UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

# Sistema de información para el Club Baloncesto Gran Canaria


Autor: Vicente José Hernández García
Tutor: José Juan Hernández Cabrera
Cotutor: Ana María Plácido Castro

Fecha 10 de Julio de 2013

ESCUELA DE INGENIERÍA INFORMÁTICA

# Agradecimientos

Quiero darles las gracias a todas las personas que me han apoyado y ayudado, tanto en el proyecto como a lo largo de todos estos años de carrera.

A mi familia y en especial a mis padres, que siempre me han dado todo y han hecho un enorme esfuerzo para que yo pudiera llegar donde estoy.

A mis tutores José Juan y Ana por guiarme ayudarme siempre todo lo que han podido con la realización de este proyecto.

Finalmente agradecer a los compañeros tanto del SIANI como de la universidad, que siempre me ayudaron, dedicándome tiempo incluso cuando ellos no lo tenían.

# Índice


Motivación ..... 1
Objetivos ..... 3
Descripción del problema ..... 5
Clubes de baloncesto ..... 6
Sistemas de información ..... 16
Tecnologías y herramientas ..... 25
Monet ..... 25
Joomla ..... 28
Eclipse ..... 32
XML ..... 33
Java ..... 34
Metodología ..... 37
Modelos desarrollo evolutivo ..... 37
Ingeniería dirigida de modelos ..... 38
Arquitectura orientada a servicios (SOA) ..... 40
Rest ..... 51
Descripción del proceso ..... 55
Arquitectura del sistema ..... 61
Sitio web ..... 62
Unidad de negocio “club” ..... 68
Unidad de negocio “Feb” ..... 82
Funciones para descargar datos del servidor web de la FEB ..... 85
Resultado ..... 89
Conclusiones ..... 91
Trabajo futuro ..... 93
Bibliografía ..... 95
Referencias ..... 96
Anexo 1: Descripción servicio web de la FEB ..... 97
Anexo 2: Estructura de los ficheros XML ..... 99

# Motivación

Existen diversos motivos que hicieron que me decidiera a la hora de la elección del Proyecto Fin de Carrera. Uno de los principales es la rama, ya que la ingeniería del software es una de las ramas informáticas que más me gusta, y este proyecto me permitiría utilizar de forma amplia e imaginativa los conocimientos técnicos aprendidos a lo largo de la carrera.

Se me brindo una oportunidad única, ya que iba a trabajar con un club de baloncesto de la categoría del Gran Canaria 2014, podría ver su funcionamiento interno y estar en contacto directo con el baloncesto que es una de mis aficiones.

Destacar también un extra de motivación que era el hecho de trabajar con requisitos impuestos por clientes reales, lo que podría considerarse como un primer contacto con el mundo laboral. Además de tener un proyecto que va a ser utilizado y no sea solo realizar un trabajo únicamente por que es obligatorio. Aunque sabía que esto le daba un extra de dificultad, ya que necesitabas que la aplicación agradara a los clientes y cumpliera sus requisitos, también el hecho de tratar con un club requería el visto bueno de otros directivos para la realización de algunos cambios, por lo que algunas cosas fue necesario modificarlas más de una vez.

Técnicamente el proyecto también me pareció muy interesante por las tecnologías y herramientas que iba a utilizar, ya que son muy demandadas en el mercado laboral.

# Sistema de información

La implementación de sistemas de información en una compañía, brindan la posibilidad de obtener grandes ventajas, incrementar la capacidad de organización de la empresa, y tornar de esta manera los procesos a una verdadera competitividad

En los últimos años las organizaciones han reconocido la importancia de administrar los principales recursos como la mano de obra y las materias primas, por lo que la información se ha colocado en un buen lugar como uno de los principales recursos que poseen las empresas actualmente y se han vuelto uno de los elementos fundamentales para lograr el éxito comercial.

Un sistema de información adecuado además ofrece una importante y notable satisfacción en los usuarios que lo operan, debido a su facilidad de uso y su acceso constante, que puede resultar en que los empleados logren alcanzar los objetivos planteados por la compañía.

Por otra parte, es de vital importancia utilizar tecnologías de información y comunicación adecuadas para el procesamiento y transmisión de los datos que se gestionarán en el sistema de información. De esta manera, en los últimos años ha aumentado la necesidad de profesionales encargados de elaborar dichos sistemas para las empresas.

# Joomla

Hacía tiempo tenía ganas de probar esta plataforma y el proyecto me brindo una "excusa" perfecta para hacerlo, tiene muchas ventajas, es un software libre, por lo que no hay que pagar licencias, además de tener un código fuente abierto, por lo que es posible modificar cualquier aspecto del sitio web manualmente en caso de que Joomla no te lo permita en sus opciones.

Otro de sus puntos fuertes es las facilidades que da a la hora de actualizar y organizar el contenido o de cambiar el estilo visual del sitio web.

Finalmente comentar que es compatible con todos los sistemas operativos ya que la herramienta te permite hacer modificaciones a través del navegador internet, por lo que no tienes ningún tipo de limitación si deseas hacer algún cambio y será posible realizarlo desde cualquier lugar.

Por lo que considero que es una herramienta interesante para aprender, tarea además sencilla debido a las facilidades que ofrece y a la multitud de guías y tutoriales que existen.

# Interoperabilidad y servicios Web

Finalmente otro de los aspectos que se tocan en el proyecto y me parece muy interesante es el tema de los servicios Web y la interoperabilidad entre unidades.
En el modelo de aplicación web "tradicional" encontramos una importante limitación: la interacción comienza y termina en dos puntos claramente definidos: la petición del usuario y la respuesta de la aplicación informática.

Un servicio web se suele definir como una unidad de aplicación capaz de ofrecer datos o servicios de procesamiento a otras aplicaciones informáticas.

Por lo que tendríamos una aplicación que da servicio a otras aplicaciones abriendo un mundo de posibilidades, permitiendo hacer aplicaciones muy interesantes utilizando estas tecnologías.

# Objetivos

El objeto principal del presente proyecto es llevar a cabo el ***desarrollo de un sistema de información para un club de baloncesto***, en este caso el Club de baloncesto Gran Canaria 2014. Aunque se dispone de una serie de objetivos más específicos.

- Realizar un análisis de los requerimientos de nuestro cliente e intentar cumplirlos o dar alternativas factibles.- Como estamos tratando con un cliente real es necesario tener en cuenta sus requisitos por lo que una de nuestros principales requisitos será ese.

- Intuitivo y fácil de usar.- Este es un requisito fundamental en la actualidad para cualquier tipo de aplicación, ya que los usuarios no tienen por qué utilizar tiempo en aprender a utilízala.

- Visualmente atractiva y con una interfaz amigable.- Una aplicación como esta, por la que te moverás, leyendo datos y buscando información tiene que ser agradable a la vista

- Conseguir que se disponga de gran cantidad de información.- Para que el sistema de información sea lo más completo posible

- La información tiene que estar bien interconectada y ser accesible de la forma más sencilla posible.- Ya que al disponer de un gran volumen de datos, si no están correctamente ordenados será imposible acceder a ellos y utilizar la aplicación.

- Fácilmente actualizable.- Todas las semanas hay partidos, se modifican las clasificaciones, estadísticas, en cada temporada hay equipos que cambian de categoría etc., por lo que es necesario tener en cuenta esto a la hora de hacer la aplicación, y conseguir que se actualice de la forma más sencilla posible.

- Sistema flexible y escalable constituido por unidades independientes.-Se dividirá en unidades independientes, de tal forma que si queremos añadir nuevas fuentes de información, o las actuales cambia la forma en que nos envían los datos, no afectan directamente al funcionamiento global del sistema, además facilitará su mantenimiento futuro.

- Adaptación de la herramienta a cada tipo de usuario y sus privilegios.- En nuestro sistema de información necesitamos la figura del administrador, por lo que existirán distintos roles o usuarios que podrán hacer uso del sistema. La apariencia de esta y las actividades que pueda desempeñar cada usuario dependerán del rol que tenga asignado, así como los privilegios asociados a su rol.

- Conseguir diferenciarnos del resto de aplicaciones similares.-Es algo indispensable hoy en día para que una herramienta tenga éxito. Al cumplir los objetivos anteriores estaríamos cumpliendo este, ya que nos diferenciaría el hecho de poder adaptarla a cualquier deporte, debido a las múltiples unidades de negocio, además

de contar con la potencia y sencillez de Monet que visualmente le dan un aspecto muy atractivo. Y finalmente la facilidad para los administradores, que con un simple botón actualizarían la aplicación. Además el valor añadido, de contar con el apoyo de un gran club de baloncesto, que es una gran ventaja.

- Crear el sitio web del club.- Cumpliendo con los requisitos y las indicaciones proporcionadas por los responsables, dando un especial énfasis en las redes sociales y los patrocinadores.

# Descripción del problema

El deporte hoy en día es un fenómeno social, practicado y seguido por millones de personas, existe un enorme seguimiento a través de los medios de comunicación, redes sociales o sitios web, que han conseguido aumentar el nivel de interés de los aficionados y fomentado su divulgación.

Ello provoca que una enorme cantidad de personas los disfruten como espectáculos de masas, convirtiéndolos en importantes negocios que mueven millones y financian a los deportistas, agentes, medios y también indirectamente, a otros sectores de la economía.

La directiva del club nos explica que es indispensable para que un club sobreviva tener una gran cantidad de aficionados, ellos son los que compran entradas, se abonan, apoyan al equipo, y consiguen que se firmen acuerdos con los patrocinadores. Pero para ello es necesario implicarlos, y hacerlos sentir parte de dicho club, y en la actualidad la mejor manera es mediante internet. De hecho nos explican como otros clubs como por ejemplo el UCAM Murcia en los últimos años han ido actualizando su sitio web y que sus resultados han sido muy favorables.

Por lo que nos indican su interés por mejorar en ese campo, ya que ellos no disponían ningún tipo de base de datos de sus jugadores, o una aplicación para guardar o consultar resultados o estadísticas. Desde su web únicamente se podía consultar la clasificación, y todo lo demás eran enlaces externos. Además nos comentaron que tampoco hacían uso de la mayoría de redes sociales, y que el diseño de su web se había quedado obsoleto, por lo que habían notado un descenso importante en el número de visitas.

Una vez planteados estos puntos comprendemos las necesidades por parte del club, que serían actualizar el sitio web del club con un aspecto más actual, dando un gran énfasis a todas las cuentas que el club disponga en redes sociales. Además de crear cuentas en otras redes sociales de las que no disponen, como por ejemplo Goear para las grabaciones de los partidos o Flickr para colgar fotos.

Otro sitio importante que querían fomentar era su YouTube, que lo tenían un poco abandonado, para ello se llegó a la conclusión de que además de tener en la web un enlace a su canal sería interesante poder incrustar videos para que se vieran directamente, pero esto debería ser dinámico, es decir que el personal de prensa pudiera subir los videos sin necesidad de depender de ningún informático. Por lo que así conseguirían tener informados a los usuarios desde su página web.

También como comentamos anteriormente se nos informó de la importancia de los patrocinadores, por lo que era necesario un buen lugar para ellos dentro de la web, y también buscar alguna solución para poder mostrar sus logos, dando importancia a unos sobre otros, dependiendo de la cantidad de dinero que aporten al club. Esto fomentaría que los patrocinadores que dan menos dinero intentaran dar más para conseguir una mejor posición además de que los mayores patrocinadores, vean justificado su gasto.

Del mismo modo que los patrocinadores, también se necesitaban unos espacios para la publicidad, que normalmente seria venta de abonos o merchandising del club.

Por otro lado estudiando las posibilidades, se llegó a la conclusión de que el club necesitaba un sistema de información, donde estuviera almacenada toda esa información de interés para los usuarios que ellos carecen a día de hoy, y que se actualice periódicamente, como clasificaciones, resultados, etc. Además de toda la información adicional, como trayectorias de jugadores o estadísticas de temporadas pasadas que se les pueda brindar, que les seria de utilidad tanto a la afición como a los propios directivos del club. Y esta petición se convertiría en la parte principal del proyecto, debido a la complejidad y volumen de los datos.

A continuación hablaremos del dominio del problema, mostrando toda la información ha sido necesario consultar antes de la realización del proyecto.

# Clubes de baloncesto

Se consideran clubes deportivos de baloncesto las asociaciones privadas que tengan por objeto la práctica y promoción de esta modalidad deportiva, así como la participación en actividades y competiciones deportivas, y que estén inscritos en la Federación Española de Baloncesto a través de las respectivas Federaciones Autonómicas integradas en aquélla.

Dentro de un club pueden existir diversos equipos, a los que deberá llamar de formas distintas, bien sea denominando cada uno con el nombre del club seguido de algún elemento diferenciador, o bien adjudicándoles nombres distintos. Estos equipos participaran en las diferentes categorías o ligas. En caso de ser profesional estará regido por una determinada federación.

La ACB (Asociación de Clubes de Baloncesto) organiza la Liga del mismo nombre, compuesta por los 18 equipos de mayor potencial. Cada año los dos últimos de la clasificación liguera descienden a la división LEB, es decir se puede perder la categoría.

La Federación Española de Baloncesto organiza la LEB, EBA, ligas femeninas y los campeonatos de las categorías de formación.

## Derechos

- Intervenir como electores y elegibles en las elecciones a representante, por su estamento de la Asamblea General de la Federación Española de Baloncesto.

- Participar con sus equipos en las competiciones oficiales que les correspondan por su categoría, en los términos y observando los requisitos determinados reglamentariamente, así como los que establezca la FEB.

- Concertar y participar en encuentros amistosos con otros clubes federados o extranjeros, en fechas compatibles con las señaladas para las competiciones oficiales con la previa autorización, según el caso, de la FEB, Federación Autonómica y Asociaciones de Clubes reconocidas por la FEB, e incluso si fuera preciso, el Ministerio de Asuntos Exteriores.

- Asistir e intervenir en las reuniones de los órganos federativos que les correspondan con arreglo a las normas vigentes.

- Optar a las ayudas económicas que anualmente establezca la Federación Española de Baloncesto para los clubes, en la medida y ocasión que determinen las correspondientes normas.

- Asociarse para el cumplimiento de sus fines.

- Fusionarse, permutar o ceder los derechos deportivos a competir de uno o varios de sus equipos

## Obligaciones

- Participar con sus equipos en las competiciones oficiales que les correspondan por su categoría, en los términos y observando los requisitos determinados reglamentariamente, así como los que establezca la FEB, debiendo señalar su terreno oficial de juego dentro de su propia Comunidad Autónoma o fuera de ella, siempre que se obtenga la correspondiente autorización de las Federaciones Autonómicas implicadas y el visto bueno de la FEB.

- Cumplir los Estatutos, Reglamentos y demás normas de la Federación Española de Baloncesto y de su Federación Autonómica, así como sus propios Estatutos.

- Poner a disposición de la Federación Española de Baloncesto y de la Federación Autonómica correspondiente sus campos de juego, cuando dichas entidades los necesiten.

- Contribuir al sostenimiento económico de la Federación Española de Baloncesto y de su Federación Autonómica correspondiente, abonando las correspondientes cuotas, derechos y compensaciones económicas que se determinen, así como las sanciones que les hayan sido impuestas por los órganos correspondientes.

- Abonar los derechos de arbitraje que, según las normas federativas, les correspondan, así como liquidar las deudas generadas con la Federación española de Baloncesto y con la Federación Autonómica que corresponda, con sus jugadores o técnicos y con otros Clubes, como consecuencia de su participación en la competición.

- Mantener la disciplina deportiva evitando situaciones de violencia o animosidad con otros miembros o estamentos del Baloncesto.

- Comunicar a la FEB las modificaciones estatutarias, el nombramiento y cese de directivos o administradores y los acuerdos de fusión, escisión o disolución.

- Cubrir mediante un seguro obligatorio deportivo que respete las prestaciones mínimas exigidas por el Real Decreto 849/1993, de 4 de Junio, por el que se determinan las prestaciones mínimas del Seguro Deportivo Obligatorio, los riesgos derivados de la práctica del baloncesto a los jugadores, entrenadores y delegados.

- Cuidar de la más perfecta formación deportiva de sus jugadores, facilitando los medios precisos para ello y garantizando su efectiva y plena actividad deportiva.

- Disponer para sus encuentros oficiales de un terreno de juego que cumpla los requisitos reglamentarios.

- Contar con entrenador con título oficial o certificado otorgado por la FEB para cada uno de los equipos que mantengan en las diferentes categorías, en los términos previstos reglamentariamente.

- Reconocer a todos los efectos las tarjetas de identidad expedidas por la Federación Española de Baloncesto y la Federación Autonómica correspondiente.

- Facilitar la entrada gratuita en el recinto deportivo, a los miembros del equipo contrario provistos de licencia federativa.

- Facilitar los datos necesarios que deben figurar en el libro-registro de Clubes a tenor de las disposiciones vigentes.

- Cumplimentar, atender y contestar con la mayor diligencia las comunicaciones que reciban de los órganos federativos y auxiliarles, facilitando cuantos datos soliciten. Además deberán facilitar sus Cuentas Anuales y presupuestos en los términos previstos en la normativa específica de cada competición.

- Facilitar la asistencia de sus jugadores y técnicos a los equipos nacionales y a las actividades federativas de perfeccionamiento técnico.

- Presentar la documentación y demás requisitos que sean necesarios para formalizar la inscripción de cada equipo, según lo establecido en este Reglamento General y en las Bases de cada Competición.

## Estructura organizativa

Los clubs se dividen en diferentes departamentos, aunque esto depende de la estructura y envergadura del club este es el esquema más usado aunque con ligeros cambios en los equipos de la ACB.

Pero independientemente del tamaño del club podríamos decir que existen cuatro áreas principales:

### Área deportiva

Es la esencia de un club de baloncesto, en tanto que la finalidad del club está ligada a la promoción y desarrollo del deporte y el objetivo principal es la formación de un equipo que represente al club en competiciones nacionales e internacionales. El área deportiva no solo es la base de las actividades de estas organizaciones, sino también lo que determina su supervivencia, su mayor o menor promoción y su potencial de crecimiento.

El área deportiva debe contemplar actividades como la selección, formación y preparación de dicho equipo, en lo físico, lo técnico y lo táctico.

La profesionalización del deporte ha supuesto importantes cambios en esta área, no sólo por la incorporación de jugadores con unos sueldos muy elevados y una gran carga mediática, sino también, porque diseñar una política deportiva, a corto, medio y largo plazo, exige una formalización, una especialización y una coordinación de las distintas actividades que deben desarrollarse de cara a conseguir un funcionamiento ágil y eficaz que sean el soporte necesario para alcanzar éxitos deportivos.

### Área comercial

La comercialización de la actividad deportiva ha supuesto la aparición y creciente relevancia de un área comercial capaz de generar nuevos recursos que permitan la financiación de las actividades deportivas. El producto que ofrecen los clubes de baloncesto profesional, y que se relaciona con el objetivo primero de formar un equipo competitivo, es el espectáculo deportivo y todo lo que ello conlleva. A medida que el club amplía su actividad comercial, aparecen nuevos productos, como son el merchandising, el uso de las instalaciones deportivas para fines no deportivos, la venta de derechos de imagen, de los derechos de televisión, etc.

Socios y aficionados son el cliente directo, quienes compran los diferentes productos que un club de baloncesto ofrece, empezando, por supuesto, por la entrada al campo o el abono de socio. Además, como se ha expuesto anteriormente, es el motor del círculo virtuoso de creación de valor para el sector, de manera que el club debe de dar gran relevancia a la relación con el aficionado. En este sentido, dentro de las áreas de trabajo de un club, tiene que haber un área especialmente dedicada al trato con el cliente directo, sin olvidar a las

distintas “peñas”, orientada a captar a nuevos socios y a fidelizar a los que ya existen y están integrados en el club.

Las empresas se relacionan con estas entidades a través de los patrocinios y de sus anuncios. Esta relación es también muy importante, debido a que aportan una parte importante de los recursos que los clubes necesitan. En este sentido, el club deberá contar con un área especialmente orientada a la captación y fidelización de las empresas que ya se han asociado al club.

Para diseñar la estructura de un club no solo habrá que tener en cuenta las actividades principales y los actores con los que se relaciona, sino también otras variables relacionadas con las particularidades de cada club y que refieren al grado de complejidad particular de cada organización, al tamaño y grado de profesionalización y comercialización.

## Área de comunicación

Los medios de comunicación son otro actor fundamental en el entorno de un club profesional. Ellos, a diario, manejan y publican una enorme cantidad de información y son verdaderos creadores de opinión. Por tanto, resulta fundamental tener una buena relación con los medios de comunicación, de manera que la exposición mediática fortalezca la imagen del club y ayude a captar nuevos socios y aficionados. Un club de baloncesto profesional mantiene una constante relación con los medios de comunicación y necesitará contar con un área que defina una estrategia y facilite orientación para aprovechar al máximo dicha relación. Lo mismo sucede con el club y las autoridades locales y asociaciones deportivas, donde una buena gestión puede ser muy beneficiosa para la organización.

La relevancia de los medios de comunicación para el deporte, por un lado, y la necesidad de mantener un contacto constante con los socios y aficionados por otro suponen la necesidad de definir un área especialmente dedicada al tema de la comunicación que se plantee incluso el conseguir medios propios, ya sea un periódico o una televisión.

Un club de baloncesto profesional debe dar conferencias de prensa, conceder entrevistas, además de las transmisiones de los partidos propios de la competición que también se transmiten a través de los medios de comunicación. La relación entre medios de comunicación y clubes de baloncesto profesional no suele ser fácil, porque:

Los clubes buscan en los medios un canal para acercarse a sus clientes y transmitir una imagen concreta del club.

Los medios buscan la noticia de primera página, lo más noticioso que les permita despertar el interés potencial del aficionado y vender ejemplares o ganar oyentes. El impacto que causen en el club las noticias no les preocupa en demasía.

Se hacen necesarias unas normas de actuación y una gestión correcta con los medios de comunicación, de manera que se consiga cumplir las expectativas de ambos actores y se eviten conflictos, sin olvidar el cuidado de la relación directa con los aficionados.

### Área de administración

Considerando los recursos económicos que gestionan los clubes de baloncesto profesional, resulta fundamental que se contemple un área que gestione dichos recursos, que diseñe presupuestos, que planifique gastos y lleve el control y el seguimiento de estos temas. La correcta administración de los recursos también determina las posibilidades de supervivencia y de crecimiento de la organización, por lo que es un área de vital importancia en toda la organización.

Estas áreas se relacionan entre ellas y se subdividen en departamentos, de diferente manera, dependiendo del club, pero un esquema bastante generalizado sería el mostrado en la figura 1.

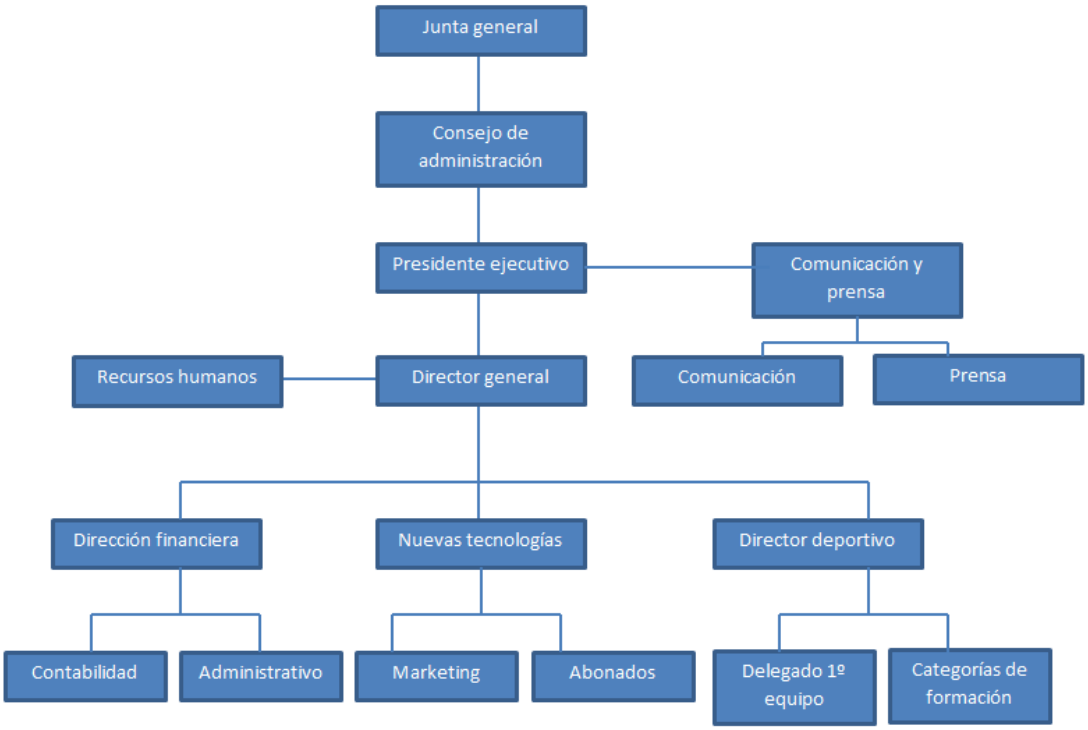



Figura 1: Diagrama de ejemplo de organización en un club de baloncesto

## Categorías

Las categorías pueden ser: masculina, femenina y de formación. Dentro de la categoría masculina están las siguientes competiciones:

Liga ACB (18 equipos)

- Es la principal liga profesional española, está formada por 18 equipos que compiten en una primera fase o fase regular en donde cada uno se enfrenta dos veces a todos los demás (una vez como local y otra como visitante).

- Al finalizar la primera vuelta los ocho primeros clasificados disputan la Copa del Rey, se puede dar el caso que el octavo clasificado no la dispute. Eso se debería a que una de las plazas es para un equipo ACB anfitrión de la ciudad que organiza el torneo. Se disputa a partido único (siete encuentros: cuartos de final, semifinales y final) en una misma sede. Lo que hace que se concentre el espectáculo y la disputa de un acontecimiento en un único fin de semana entre los mejores equipos, lo que le da espectacularidad y emoción a esta competición.

- Al final de la fase regular, los ocho primeros clasificados se enfrentan en los Play off, esto es, se emparejan esos ocho equipos de acuerdo con su clasificación en la fase regular y disputan unas eliminatorias al mejor de tres partidos (cinco en la final). Los primeros clasificados tienen ventaja de campo y los emparejamientos se hacen favoreciendo la clasificación previa, es decir el primero juega con el octavo, el segundo con el séptimo, el tercero con el sexto, y el cuarto con el quinto. El ganador de la final de estas eliminatorias es proclamado campeón de la Liga ACB.

LEB Oro (18 equipos)

- Es una competición organizada por la Federación Española de Baloncesto y es la categoría anterior a la máxima categoría profesional, la Liga ACB.

- Se divide en dos fases: la fase regular y los Play off.

- En la fase regular se enfrentan entre sí todos los equipos. El primer clasificado asciende directamente a la Liga ACB y los equipos clasificados entre el 2º y el 9º puesto pasan a disputar los Play off de ascenso, que son unas eliminatorias de cuartos de final, semifinales y final, en las que el campeón también consigue el ascenso a la Liga ACB.

- En cuartos de final, el segundo clasificado de la fase regular se enfrenta al noveno, el tercero al octavo, el cuarto al séptimo y el quinto al sexto. La primera eliminatorias es al mejor de tres partidos. El primer encuentro se disputa en la cancha del equipo mejor clasificado y el siguiente en la del rival. En caso de empate, se disputa un tercer encuentro en el pabellón del primero de ellos.

- Los cuatro equipos clasificados disputan una final a cuatro, estilo Final Four, a un único partido de semifinal y una final entre los vencedores. El ganador, si cumple los requisitos económicos y técnicos, disputará la siguiente edición de la Liga ACB.

- Este formato se impone a partir de la temporada 2007-08. Anteriormente, los 8 primeros clasificados se enfrentaban en unos play-off de características similares a los play-offs de la ACB, con la importante diferencia de que la final se disputaba a partido único en la cancha del equipo con mejor puesto en la fase regular. Esto es debido a que los dos equipos que jugaban la final eran los ascendidos a la ACB, así que la única motivación era conseguir el campeonato LEB, aspecto mucho menos valorado que conseguir el ascenso a la ACB. Este cambio de formato quiso dejar de perjudicar al campeón de la fase regular, ya que no han sido pocas las veces en las que el campeón de la fase regular no conseguía el ascenso en los play-offs.

- Al terminar la primera mitad de la temporada, los dos mejores equipos disputan la Copa Príncipe

- Descienden de categoría, a la LEB Plata, dos equipos, el último y el penúltimo clasificados de la fase regular.

LEB Plata (15 equipos)

- Actualmente denominada Adecco Plata por temas de patrocinio. Anteriormente se denominó LEB-2. Comenzó su andadura en el año 2000. Se divide en dos fases: la fase regular y los Play off.

- En la fase regular se enfrentan entre sí todos los equipos. El primer clasificado asciende directamente a la Liga LEB Oro y los equipos clasificados entre el 2º y el 9º puesto pasan a disputar los Play off de ascenso, que son unas eliminatorias de cuartos de final, semifinales y final, en las que el campeón también consigue el ascenso a la Liga LEB Oro.

- En cuartos de final, el segundo clasificado de la fase regular se enfrenta al noveno, el tercero al octavo, el cuarto al séptimo y el quinto al sexto. La primera eliminatorias es al mejor de tres partidos. El primer encuentro se disputa en la cancha del equipo mejor clasificado y el siguiente en la del rival. En caso de empate, se disputa un tercer encuentro en el pabellón del primero de ellos.

- Los cuatro equipos clasificados disputan una final a cuatro, estilo Final Four, a un único partido de semifinal y una final entre los vencedores. El ganador consigue el pase a la Liga LEB Oro.

- Este formato se impone a partir de la temporada 2007-08. Anteriormente, los 8 primeros clasificados se enfrentaban en unos play-off de características similares a los play-offs de la ACB, con la importante diferencia de que la final se disputaba a partido único en la cancha del equipo con mejor puesto en la fase regular. Esto es debido a que los dos equipos que jugaban la final eran los ascendidos a LEB Oro, así que la única motivación era conseguir el campeonato LEB Plata, aspecto mucho menos valorado que conseguir el ascenso a la LEB Oro. Este cambio de formato quiso dejar de perjudicar al campeón de la fase regular, ya que no han sido pocas las veces en las que el campeón de la fase regular no conseguía el ascenso en los play-offs.

- Al terminar la primera mitad de la temporada, los dos mejores equipos disputan la Copa LEB Plata

- Descienden de categoría, a la Liga EBA dos equipos, el último directo y el penúltimo y antepenúltimo clasificados de la fase regular se enfrentan en un Play Off del que saldrá la otra plaza.

Liga EBA (5 grupos)

- Se trata de la cuarta liga en importancia, tras la Liga ACB, la LEB Oro y LEB Plata, no obstante la Liga EBA fue durante dos años el segundo escalón del Baloncesto

Español tras la ACB, antes de la creación de las Ligas LEB. La liga se disputa simultáneamente en 5 grupos (A, B, C, D y E) donde se integran los equipos por cercanía geográfica cubriendo todo el territorio español. Durante la temporada 2011-12, el grupo "A" se encuentra subdividido en dos grupos, el "A" "A" con 12 equipos y el "A" "B" con 11, el grupo "B" consta de 16 equipos, el grupo "C" de 14 equipos, el grupo "D" de 9 equipos y el grupo "E" de 12 equipos.

Dentro de la categoría femenina están las siguientes competiciones:

Liga Femenina de Baloncesto (14 equipos)

- La Liga Femenina es la máxima competición de clubes femeninos de baloncesto que se disputa en España.

- La copa de la Reina de Baloncesto es la segunda competición española de baloncesto en importancia, aunque es la más antigua. Se inició en la temporada 1942-1943, aunque sufrió interrupciones, hasta que se regularizó a partir de la temporada 1977 - 1978.

- La disputan los equipos que ocupan las 4 primeras posiciones de la clasificación al término de la primera vuelta de la liga. Los dos primeros ocupan puesto de cabeza de serie y se enfrentan a tercero y cuarto.

- La Copa de la Reina tiene una duración de 2 días (sábado y domingo) en los cuales se disputan semifinales y final.

Liga Femenina de Baloncesto 2 (2 grupos de 14 equipos cada una)

- Al finalizar la temporada se hacen Play off para ascender a dos equipos a la categoría Liga Femenina de Baloncesto

Un club puede tener una o varias categorías de formación, se divide según edad y la federación reconoce las siguientes:

- **Escuela:** menores de 7 años
- **Pre benjamín**: 8 años
- **Benjamín:** 9 años
- **Preminibasket:** 10 años
- **Minibasket:** 11 años.
- **Infantil:** 12-13. años
- **Cadete:** 14-15. años
- **Juvenil o Junior:** 16-17

A estas categorías de formación se les denomina comúnmente cantera y son extremadamente importante para cualquier club, el tener una misma filosofía de trabajo en todas sus categorías, desde infantiles a juveniles, puede suponer (además de un extraordinario ahorro en cuanto a fichajes), que los jugadores hayan jugado desde pequeños de la misma manera, por lo que de una forma natural, se pueden encontrar

jugando en el primer equipo y adaptarse a conceptos y exigencias del juego desde el primer día.

## Equipos

Como comente anteriormente un club se compone de varios equipos, cada uno de estos equipos funcionará de forma individual, ya que no tienen relación unos con otros, aunque responden ante los mismos directivos.

Cada equipo estará únicamente en una categoría, dispone de un nombre que puede variar de una temporada a otra, aunque no se cambien ni sus directivos ni sus jugadores. Cada equipo está formado por diez jugadores, uno de los cuales asume la función de capitán. Cuando se trate de torneos internacionales, con cinco o más partidos consecutivos, cada equipo puede disponer de 12 jugadores. Cinco de ellos estarán en el terreno de juego y el resto en el banquillo, en la parte exterior de la pista, junto a su entrenador.

## Jugadores

Cada jugador lleva en la camiseta el número en la parte delantera y en la trasera, claramente visibles y comprendidos entre el 4 y el 15, ambos inclusive. El color de las camisetas de ambos equipos será distinto y, a ser posible, de tonos opuestos (claro y oscuro).

Existen cinco posiciones posibles que se les asignará a los jugadores, según el entrenador estime conveniente

- **Base:** Normalmente el jugador más bajo del equipo. En ataque sube la pelota hasta el campo contrario y dirige el juego de ataque de su equipo, mandando el sistema de juego. Sus características recomendables son un buen manejo de balón, visión de juego, capacidad de dar buenos pases, buena velocidad y un acertado tiro exterior. En los bases son apreciadas las asistencias como los puntos conseguidos, aunque un buen jugador debe conseguir ambas cosas. En defensa han de dificultar la subida del balón de la base contrario, tapar las líneas de pase y estar atento a recoger los rebotes largos. Normalmente estos jugadores no son de una elevada estatura, pues lo realmente importante es la capacidad organizativa y de dirección de juego. Conocidos como ***1*** en la terminología empleada por los entrenadores.

- **Escolta**: Jugador normalmente más bajo, rápido y ágil que el resto, exceptuando a veces el base. Debe aportar puntos al equipo, con un buen tiro incluyendo el tiro de tres puntos, un buen dominio del balón y una gran capacidad de entrar a canasta. Conocidos como ***2*** en la terminología empleada por los entrenadores.

- **Alero**: Es generalmente una altura intermedia entre los jugadores interiores y los exteriores. Su juego está equilibrado entre la fuerza y el tiro. Es un puesto

importante, por su capacidad de combinar altura con velocidad. En ataque debe ser buen tirador de tres puntos y saber culminar una entrada hasta debajo del tablero contrario, son piezas básicas en lanzar el contraataque y suelen culminar la mayoría de ellos. Conocidos como ***3*** en la terminología empleada por los entrenadores.

- **Ala-Pivot**: Es un rol más físico que el del alero, en muchos casos con un juego muy similar al pívot. Mantiene la mayoría de los puntos en el poste bajo, aunque algunos pueden llegar a convertirse en tiradores muy efectivos. Sirven de ayuda al pívot para impedir el juego interior del equipo contrario, y cierran el rebote. Conocidos como ***4*** en la terminología empleada por los entrenadores.

- **Pivot**: Suelen ser los jugadores de mayor altura del equipo, y los más fuertes muscularmente. Normalmente, el pívot debe usar su altura y su potencia jugando cerca del aro. Un pívot que conjunte fuerza con agilidad es una pieza fundamental para su equipo. Son los jugadores que más sorprenden a los aficionados noveles, por su gran altura. En Europa el pívot medio ha evolucionado más y es capaz de abrirse hacia afuera para tirar. En defensa buscan recoger el rebote corto, impedir el juego interior del equipo contrario y taponar las entradas de jugadores exteriores. Conocidos como ***5*** en la terminología empleada por los entrenadores.

# Sistemas de información

Un sistema de información es un conjunto de elementos orientados al tratamiento y administración de datos e información, organizados y listos para su uso posterior.

- Personas
- Datos
- Actividades
- Recursos

Todos estos elementos interactúan para procesar los datos (incluidos los procesos manuales y automáticos) y dan lugar a información más elaborada, que se distribuye de la manera más adecuada posible en una determinada organización, en función de sus objetivos.

En el siguiente esquema podemos observar cómo se clasifican los sistemas de información

Figura 2: Clasificación de sistemas de información

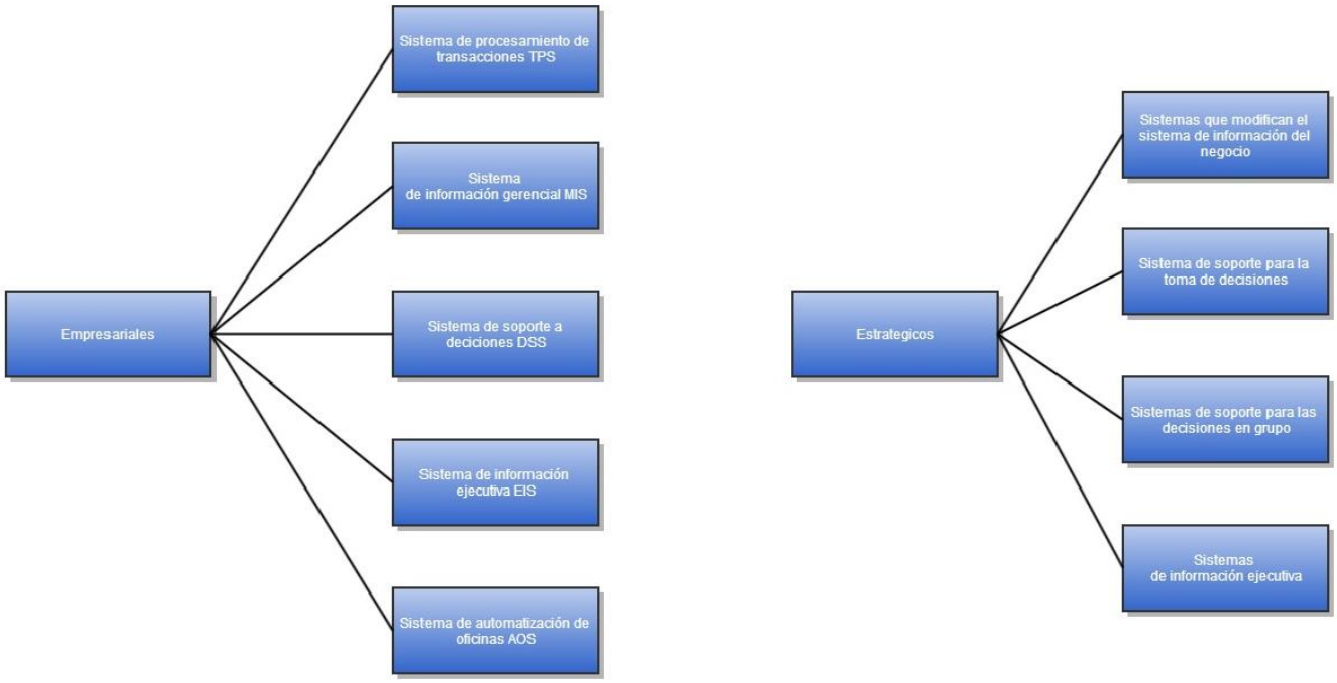



Figura3: Tipos de sistemas de información

## Clases de sistemas

- nivel operativo: apoya a los encargados de las operaciones rutinarias de la empresa, como vender, comprar, depósitos, etc.
- nivel de conocimientos: ayuda a las personas que trabajan con documentos, contadores
- nivel de administración: ayuda a los encargados de tomar decisiones (no rutinarias)
- nivel estratégico : ayuda a los administradores encargados de determinar la estrategia de la organización

Esquema de los niveles y algunos de los sistemas que se utilizan en cada uno de ellos:

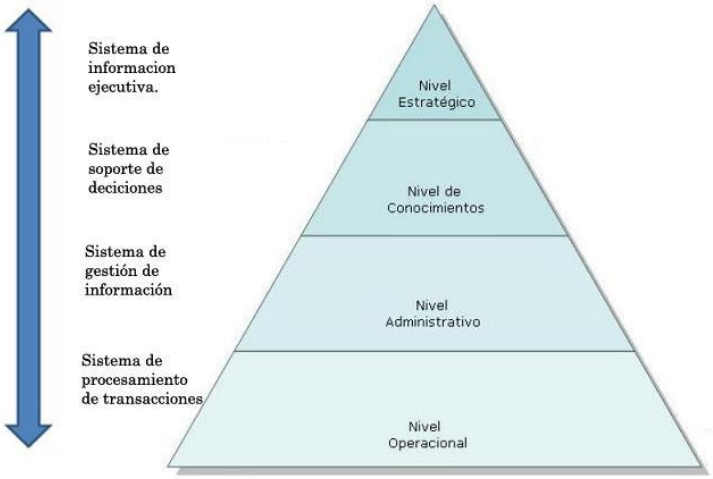



Figura 4: Pirámide de sistemas de información

## Clasificación según entorno de aplicación

- Entorno transaccional: Una transacción es un suceso o evento que crea/modifica los datos. El procesamiento de transacciones consiste en captar, manipular y almacenar los datos, y también, en la preparación de documentos; en el entorno transaccional, por tanto, lo importante es qué datos se modifican y cómo, una vez que ha terminado la transacción.

- Entorno decisional: Este es el entorno en el que tiene lugar la toma de decisiones; en una empresa, las decisiones se toman a todos los niveles y en todas las áreas (otra cosa es si esas decisiones son estructuradas o no), por lo que todos los SI de la organización deben estar preparados para asistir en esta tarea, aunque típicamente, son los DSS los que se encargan de esta función.

## Tipos de Sistemas

Sistemas de Procesamiento de Transacciones (TPS):

- Es un tipo de sistema de información que recolecta, almacena, modifica y recupera toda la información generada por las transacciones producidas en una organización. Una transacción es un evento que genera o modifica los datos que se encuentran eventualmente almacenados en un sistema de información. Para que un sistema informático pueda ser considerado como un TPS, este debe superar el test ACID.

- Con frecuencia son el primer tipo de Sistemas de Información que se implanta en las organizaciones. Se empieza apoyando las tareas a nivel operativo de la organización para continuar con los mandos intermedios y posteriormente con la alta administración conforme evolucionan.

- Suelen logran ahorros significativos en mano de obra, debido a la automatización de tareas, además requieren mucho manejo de datos, y como resultado también generan grandes volúmenes de información.

- Son fáciles de justificar ante la dirección general, ya que sus beneficios son visibles y palpables. El proceso de justificación puede realizarse enfrentando ingresos y costos. Esto se debe a que en el corto plazo se pueden evaluar los resultados y las ventajas que se derivan del uso de este tipo de sistemas. Entre las ventajas que pueden medirse se encuentra el ahorro de trabajo manual.

- Son fácilmente adaptables a paquetes de aplicación que se encuentran en el mercado, ya que automatizan los procesos básicos que por lo general son similares o iguales en otras organizaciones. Ejemplos de este tipo de sistemas son la facturación, nóminas, cuentas por cobrar, cuentas por pagar, contabilidad general, conciliaciones bancarias, inventarios, etcétera.

- Respuesta rápida: En este tipo de sistemas resulta crítico que exista un rendimiento elevado con tiempos de respuesta cortos. Una empresa no puede permitirse tener clientes esperando por una respuesta del SPT; el tiempo total transcurrido desde que se inicia la transacción hasta que se produce la salida correspondiente debe ser del orden de unos pocos segundos o menos.

- Fiabilidad: Muchas organizaciones basan su fiabilidad en los SPT; un fallo en un SPT afectará negativamente a las operaciones o incluso parará totalmente el negocio. Para que un SPT sea efectivo, su tasa de fallos debe ser muy baja. En caso de fallo de un SPT, debe existir algún mecanismo que permita una recuperación rápida y precisa del sistema. Esto convierte en esencial la existencia procedimientos de copia de seguridad y de recuperación ante fallos correctamente diseñados.

- Inflexibilidad: Un SPT requiere que todas las transacciones sean procesadas exactamente de la misma forma, independientemente del usuario, el cliente o la hora del día. Si los SPT fuesen flexibles, habría entonces demasiadas posibilidades de ejecutar operaciones no estándar. Por ejemplo, una aerolínea comercial necesita aceptar de forma consistente reservas de vuelos realizadas por un gran número de agencias de viaje distintas; aceptar distintos datos de transacción de cada agencia de viajes supondría un problema.

- Procesamiento controlado: El procesamiento en un SPT debe apoyar las operaciones de la organización. Por ejemplo, si una organización establece roles y responsabilidades para determinados empleados, el SPT debe entonces mantener y reforzar este requisito.

Sistemas de Trabajo de Conocimientos y de automatización de oficina:

- Sistema de computación, por ejemplo, procesador de textos, correo electrónico y calendarización, diseñado para aumentar la productividad de los trabajadores de datos en la oficina.

- Los sistemas de trabajo de conocimientos (KWS) ayudan a quienes crean nueva información, como contadores, ingenieros, etc.

Sistema de información Gerencial (MIS)

Estos sistemas son el resultado de interacción colaborativa entre personas, tecnologías y procedimientos -colectivamente llamados sistemas de información- orientados a solucionar problemas empresariales. Los SIG o MIS (también denominados así por sus siglas en inglés: *Management Information System*) se diferencian de los sistemas de información comunes en que para analizar la información utilizan otros sistemas que se usan en las actividades operacionales de la organización. Académicamente, el término es comúnmente utilizado para referirse al conjunto de los métodos de gestión de la información vinculada a la automatización o apoyo humano de la toma de decisiones

Todas las funciones gerenciales; Planificación, Organización, Dirección y Control son necesarias para un buen desempeño organizacional. Los Sistemas de Información Gerencial son necesarios para apoyar estas funciones, en especial la Planificación y el Control. El valor de la información proporcionada por el sistema, debe cumplir con los siguientes cuatro supuestos básicos:

- Calidad: Para los gerentes es imprescindible que los hechos comunicados sean un fiel reflejo de la realidad planteada.

- Oportunidad: Para lograr un control eficaz, las medidas correctivas en caso de ser necesarias, deben aplicarse a tiempo, antes de que se presente una gran desviación respecto de los objetivos planificados con anterioridad.

- Cantidad: Es probable que los gerentes casi nunca tomen decisiones acertadas y oportunas si no disponen de información suficiente, pero tampoco deben verse desbordados por información irrelevante e inútil, pues esta puede llevar a una inacción o decisiones desacertadas.

- Relevancia: La información que le es proporcionada a un gerente debe estar relacionada con sus tareas y responsabilidades.

Los sistemas de información gerencial son una necesidad hoy en día, ya que las empresas manejan grandes cantidades de datos los cuales deben ser analizados, de tal manera que se pueda encontrar información relevante para tomar diferentes cursos de

acción. Los SIG actualmente son conocidos como Business intelligent (Inteligencia de negocios), esto es debido a que influyen a la toma de decisiones.

Los SIG forman parte de las estrategias corporativas, ya que la comunicación e información son de gran valor en las organizaciones o empresas, porque representan poder.

Se pueden considerar una pirámide:

1. La parte inferior de la pirámide está comprendida por la información relacionada con el procesamiento de las transacciones.

2. El siguiente nivel comprende los recursos de información para apoyar las operaciones diarias de control.

3. El tercer nivel agrupa los recursos del sistema de información para ayudar a la planificación táctica y la toma de decisiones relacionadas con el control Administrativo.

4. El nivel más alto comprende los recursos de información necesarios para apoyar la planificación estratégica y la definición de políticas de los niveles más altos de la administración.

## Sistema de Apoyo a Decisiones (DSS)

Ayuda a quienes deben tomar decisiones que son semiestructuradas, únicas o que cambian rápidamente. Son más analíticos que otros sistemas. Son interactivos y suelen introducirse después de haber implantado los Sistemas Transaccionales más relevantes de la empresa, ya que estos últimos constituyen su plataforma de información.

La información que generan sirve de apoyo a los mandos intermedios y a la alta administración en el proceso de toma de decisiones.

Suelen ser intensivos en cálculos y escasos en entradas y salidas de información. Así, por ejemplo, un modelo de planeación financiera requiere poca información de entrada, genera poca información como resultado, pero puede realizar muchos cálculos durante su proceso.

No suelen ahorrar mano de obra. Debido a ello, la justificación económica para el desarrollo de estos sistemas es difícil, ya que no se conocen los ingresos del proyecto de inversión.

Suelen ser Sistemas de Información interactivos y amigables, con altos estándares de diseño gráfico y visual, ya que están dirigidos al usuario final.

Apoyan la toma de decisiones que, por su misma naturaleza son repetitivas y de decisiones no estructuradas que no suelen repetirse. Por ejemplo, un Sistema de Compra de Materiales que indique cuándo debe hacerse un pedido al proveedor o un Sistema de Simulación de Negocios que apoye la decisión de introducir un nuevo producto al mercado.

Este tipo de sistemas puede incluir la programación de la producción, compra de materiales, flujo de fondos, proyecciones financieras, modelos de simulación de negocios, modelos de inventarios, etcétera.

## Sistemas de Apoyo a Ejecutivos (ESS)

Sirven al nivel superior de administradores, y le brinda información del entorno.

Los sistemas de apoyo a ejecutivos, son creados específicamente para la alta dirección, posicionándose como factor clave, debido a que buscan que se pueda observar, monitorear y dar seguimiento a los factores críticos para el éxito de la organización. Éstos entrelazan información interna y externa de la empresa, planteando un panorama completo para la toma correcta de decisiones.

Para conducir una empresa al éxito, no importando su naturaleza, se debe estar saturado de la información necesaria, es decir, se debe conocer, comprender, analizar nuestros pro y contra, hacer énfasis en nuestras áreas fuertes y mejorar las débiles. Es ahí donde la información proporcionada o suministrada, juega un papel importante debido a que un sistema que apoye directamente a los altos ejecutivos, le permitirá mejorar su panorama para dirigir a la organización.

El sistema de apoyo a ejecutivos, debe contar con ciertas características para que sea efectivo, destacando:

- Buscar cubrir las necesidades particulares de la alta administración.
- Filtrar y dar seguimiento a la información crítica de la organización.
- Se debe desarrollar información por gráficas y tabulares de alta calidad.
- La información debe estar accesible en línea. Adicionalmente, a las anteriores características es necesario que se cumplan algunos factores como por ejemplo, que la información suministrada se vea bien; que sea relevante; rápida aparte de disponible y actualizada.

Estos sistemas de apoyo, constituyen una herramienta muy útil para la toma de decisiones, sin embargo, no significa que por tener los sistemas más avanzados tecnológicamente se tomarán buenas decisiones, o que guiará a la empresa hacia el éxito. Sino más bien, se trata de que al contar con un sistema enfocado directamente hacia los ejecutivos, se tiene la oportunidad de solucionar una situación que se presente, de forma

más eficaz y eficiente buscando la mejor alternativa que la alta dirección considere correcta y oportuna según sea el caso buscando como objetivo común un resultado positivo que saque a flote la organización.

Toda información antes de serlo, debe ser procesada, según los datos proporcionados de la compañía vista desde dentro y fuera de ella, posteriormente esos datos deben ser unificados para transformarse en información. Es fundamental que el sistema de información implementado funcione a la perfección y se convierta en una herramienta totalmente confiable para los ejecutivos que la utilizan. Debido a que de nada serviría, contar con un sistema que no permita contar con un ambiente claro y que sea inseguro o en el peor de los casos equivocado.

Se debe tener conciencia de que estos sistemas de apoyo para los ejecutivos deben estar totalmente actualizados, con el fin de que las decisiones que se tomen, vayan de la mano a los datos reales, y si fuera necesario ir continuamente modificando y actualizando los sistemas de apoyo, por la evolución que se vaya dando por factores internos y externos.

La implementación y desarrollo de los sistemas de apoyo para ejecutivos, se debe diseñar para el usuario del sistema, es decir, que se debe desarrollar en base a lo que el ejecutivo requiera determinando sus necesidades, sin olvidar los objetivos principales de la empresa a corto y mediano plazo. Se debe definir la información que se necesita para dar seguimiento al cumplimiento de estos objetivos paso a paso para que se cumpla el objetivo común y global que es crear una empresa de éxito dentro del mercado. Este sistema de apoyo es importante que cuando se aplique a la organización se haga partiendo de cero, y que a su vez se modifique el sistema actual ajustándose a las necesidades de los ejecutivos, para ello es preferible que se cuente de preferencia con la ayuda de técnicos externos con experiencia en el tema.

Este sistema cuenta con grandes ventajas como las que ya se mencionaron, pero también cuenta con desventajas tales como: La aplicación de este sistema requiere de un alto costo dependiendo del tamaño de la organización. Además se debe contar con el personal adecuado que tenga el tiempo suficiente para crear, planificar y desarrollar el sistema. Se debe tener en cuenta que hay que implementar un control que mida el desarrollo de la aplicación. Por otro lado, se debe manejar la reducción de resistencia al cambio, ya que no todos los ejecutivos aceptan algo tan drásticamente diferente en donde se ven comprometidos a emitir su juicio y definir una orientación de la empresa.

Hoy en día, la tecnología juega un papel sumamente importante para cualquier compañía en el planteamiento de posibles soluciones para los distintos problemas que pueden darse dentro de una organización. Continuamente expertos en el tema, van mejorando y afinando cada vez más los sistemas a utilizar con el fin de convertir a las empresas vanguardistas en compañías cada vez más y más competitivas.

## Ciclo de vida de los sistemas de información

Las pautas básicas para desarrollar un sistema de información

- Conocimiento de la Organización: analizar y conocer todos los sistemas que forman parte de la organización, así como los futuros usuarios del SI. En las empresas (fin de lucro presente), se analiza el proceso de negocio y los procesos transaccionales a los que dará soporte el SI.

- Identificación de problemas y oportunidades: el segundo paso es relevar las situaciones que tiene la organización y de las cuales se puede sacar una ventaja competitiva(Por ejemplo: una empresa con un personal capacitado en manejo informático reduce el costo de capacitación de los usuarios), así como las situaciones desventajosas o limitaciones que hay que sortear o que tomar en cuenta(Por ejemplo: el edificio de una empresa que cuenta con un espacio muy reducido y no permitirá instalar más de dos computadoras).

- Determinar las necesidades: Se procede identificar a través de algún método de recolección de información (el que más se ajuste a cada caso) la información relevante para el SI que se propondrá.

- Diagnóstico: En este paso se elabora un informe resaltando los aspectos positivos y negativos de la organización. Este informe formará parte de la propuesta del SI y, también, será tomado en cuenta a la hora del diseño.

- Propuesta: contando ya con toda la información necesaria acerca de la organización es posible elaborar una propuesta formal dirigida hacia la organización donde se detalle el presupuesto, relación costo-beneficio, presentación del proyecto de desarrollo del SI.

- Diseño del sistema: Una vez aprobado el proyecto, se comienza con la elaboración del diseño lógico del SI; la misma incluye el diseño del flujo de la información dentro del sistema, los procesos que se realizarán dentro del sistema, etc. En este paso es importante seleccionar la plataforma donde se apoyará el SI y el lenguaje de programación a utilizar.

- Codificación: con el algoritmo ya diseñado, se procede a su reescritura en un lenguaje de programación establecido (programación), es decir, en códigos que la máquina pueda interpretar y ejecutar.

- Implementación: Este paso consta de todas las actividades requeridas para la instalación de los equipos informáticos, redes y la instalación del programa generado en el paso anterior.

- Mantenimiento: proceso de retroalimentación, a través del cual se puede solicitar la corrección, el mejoramiento o la adaptación del SI ya creado a otro entorno. Este paso incluye el soporte técnico acordado anteriormente.

# Tecnologías y herramientas

## Monet

Monet, cuyo nombre viene de Modelling on Network, es una plataforma software orientada a desarrollar software de sistemas de información en pequeñas organizaciones, basado en unidades de negocio que permite realizar la definición de los submodulos de manera muy eficaz.

La misión principal de esta solución es facilitar el desarrollo y posterior evolución del software, de forma que se pueda construir en poco tiempo y a la vez sea flexible para el cambio cuando sea necesario. Para facilitar el proceso de desarrollo, se plantea un proceso de Ingeniería Dirigida por Modelos (MDE) en la que los modelos son la base para llevar a cabo el proceso de desarrollo.

Los motivos que nos han llevado a utilizar Monet son que dicha solución que no sólo ayuda al desarrollo del software del sistema de información, sino también a la reingeniería de la organización. Con la visión de replantear el funcionamiento de la organización integrando las nuevas tecnologías dentro de su estructura, el objetivo no debe consistir simplemente en dotar a la empresa de nuevas tecnologías, se trata de una nueva forma de hacer las cosas en la que la tecnología entra en las organizaciones de forma controlada.

El objeto de esta solución es ayudar a mejorar la gestión de los procesos, gestionar eficazmente el conocimiento, cooperar con otras organizaciones de forma flexible y simplificar las tareas diarias que deben llevar a cabo los trabajadores dentro de las organizaciones.

Otra de las ventajas de usar esta solución es que permite abordar la incertidumbre de los requisitos facilitando la introducción de cambios en el software en cualquier momento, puesto que es un proceso de ingeniería dirigida por modelos, siendo estos la base para realizar el proceso de desarrollo. Además de facilitar mucho las tareas de interoperabilidad.

Monet se compone de múltiples tipos de definiciones que utilizaremos en nuestro proyecto.

- Role
- Escritorios
- Colección
- Contenedor
- Formulario
- Índice
- Tarea
- Servicio
- Importadores

### Role

En Monet se utilizan para dividir los tipos de usuario, agrupándolos por Role, y asignándole a cada Role permisos para acceder a los distintas partes del sistema y permitir o prohibir realizar determinadas acciones.

Aclarar como dato, aunque no tiene mayor importancia, que existen ciertos Roles que quizás no tengan usuarios, en nuestro caso por ejemplo necesitamos de ellos para la interoperabilidad entre las unidades, cosa que explicaremos más adelante.

### Escritorios

Mediante estos ficheros definimos los datos accesibles desde la pantalla principal, que puede ser diferente dependiendo de los Roles.

### Colecciones

Son conjuntos de datos homogéneos, en nuestro caso son colecciones de formularios o de contenedores, estos ficheros se dividen en dos partes:

En la primera parte especificaremos los parámetros por los que vamos a filtrar u ordenar los datos que se mostraran. También se especificara porque atributo se ordenaran los datos inicialmente antes de que el usuario toque nada o si queremos aplicar algún tipo de filtro.

La siguiente parte es el índice, que todas las colecciones necesitan y podrá estar incluido en el mismo fichero o en otro, en nuestro caso lo utilizamos de ambas maneras. Aunque en el caso de estar en otro fichero es necesario especificar su ruta. El contenido del índice lo explicaremos a continuación.

### Contenedor

Es un conjunto de datos heterogéneos, en nuestro caso todos los contenedores, contienen un formulario y varias colecciones, cuyos ficheros se dividen en dos partes.

En la primera especificamos los tipos de nodos que podrá almacenar el contenedor, que por definición no serán iguales.

Finalmente en la segunda parte ponemos la referencia a todas las vistas de dichos nodos, que se encuentran en los formularios, para saber los datos que se van a mostrar.

### Formulario

Es lo que podríamos denominar la “unidad básica” o “nivel más bajo”, y es donde almacenaremos la información, especificando una lista con el nombre y tipo de los campos., donde existen multitud de tipos , como pueden ser Link, Picture, String, Date, Number… Además de una vista donde se define que datos se muestran y su orden, es posible evitar mostrar ciertos campos aunque están almacenados, para internamente realizar búsquedas o filtrar.

Es necesario especificar la ruta del índice en el que se mapearan los datos.

### Índice

Se utilizan para definir lo que veremos desde la aplicación, por ejemplo la lista de jugadores, nos parecerá solo el nombre, foto y su categoría, y una vez seleccionamos, entraremos en el formulario y dispondremos de toda la información que haya sobre el jugador seleccionado. Esta información se especifica mediante el índice.

Los índices se dividen en dos partes, una de ellas es la lista de atributos, donde se mapearan los campos de los formularios, donde el nombre y el tipo tiene que ser igual.

En la segunda parte crearemos la vista, indicando cuales de estos datos queremos que se muestren y en qué orden.

### Tarea

Una tarea es una actividad que se va a llevar a cabo en un momento dado, por lo que le añadiremos código dentro, y cuando llegue el momento se ejecutara.

### Servicio

Un servicio está a la espera de que le llegue una solicitud, para ejecutar el código que tiene en su interior y finalizar.

Se encargan de dados unos datos actualizar el sistema, es necesario especificar de donde le vienen los datos, que en nuestro caso es un XML, dividiendo cada una de las etiquetas del fichero por su nombre y tipo. Y a continuación relacionar dicha información con los campos de los formularios del sistema.

# Joomla

Es una aplicación Open Source o de código abierto programada en lenguaje PHP bajo una licencia GPL y que utiliza una base de datos MySQL para almacenar el contenido y los parámetros de configuración del sitio. Ofrece, en general, las mismas ventajas que otros gestores de contenidos web (CMS o Content Management System) en la medida en que hace posible convertir una web estática tradicional en un portal con diferentes funcionalidades y características dinámicas e interactivas además de facilitar la actualización de contenidos.

Al igual que otros CMS, Joomla es un sistema "administrado". El administrador o administradores acceden con su clave desde cualquier ordenador conectado a Internet a un panel de control desde el que realizan todas las operaciones, incluidas las relacionadas con la instalación de nuevos componentes y módulos en el sistema

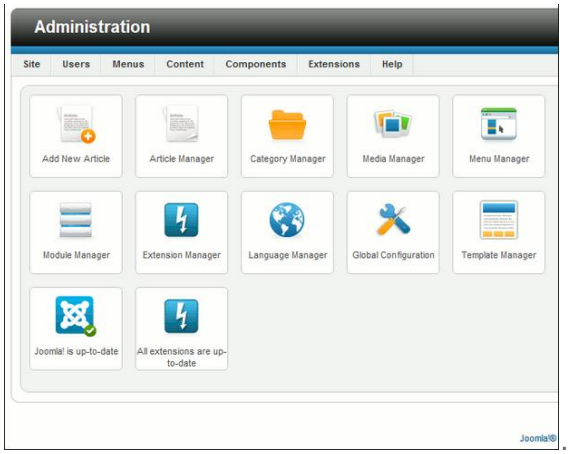



Figura 5: Panel de administración de Joomla

Al igual que otros gestores de contenidos, Joomla permite la participación de varios autores en el desarrollo y mantenimiento del sistema gracias a la posibilidad de asignar diferentes niveles de uso y administración a diferentes usuarios. Con ello se puede hacer una gestión colaborativa a la vez que controlada del sitio web y conseguir un mejor reparto de funciones y responsabilidades que estimule la participación. El sistema de gestión de usuarios de Joomla permite también restringir al acceso a determinados contenidos a usuarios con permisos especiales.

Otra de las grandes ventajas de sistemas CMS como Joomla es la facilidad para editar y organizar los contenidos. Además de páginas de contenido estático para la información que sea

más estable y permanente podemos añadir artículos agrupados en secciones y categorías temáticas que permiten su clasificación y que ofrecen diferentes opciones para su presentación: en forma de lista o índice, en forma de blog, en una o más columnas, etc.

Para la edición de artículos o páginas de contenido, Joomla dispone de un editor visual con una barra de herramientas semejante a la de los procesadores de textos con las que se puede dar formato al contenido, añadir imágenes, insertar hipervínculos, etc. Los administradores disponen también de un Gestor Multimedia con el que pueden subir fácilmente imágenes, archivos y otros elementos multimedia al servidor para luego insertarlos en las páginas y artículos de contenido.

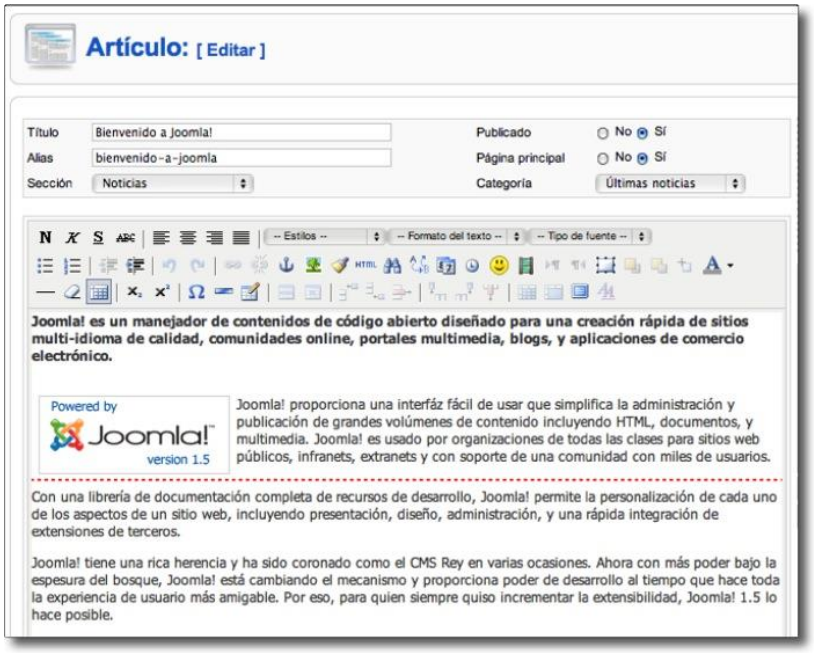



Figura 6: Editor de textos de Joomla

Otra característica importante de los gestores de contenido como Joomla es que con ellos diseño y contenido se manejan de forma independiente. La estética del portal se basa en plantillas o templates prediseñadas que nos permiten modificar en cualquier momento el aspecto visual del sitio sin tener que dar de nuevo formato al contenido. Basta con seleccionar una plantilla diferente en el Gestor de plantillas para variar los colores, tipografías, fondos o la distribución de las zonas de contenido en la página. Además podemos personalizar fácilmente las plantillas disponibles para adaptar su diseño a nuestros gustos y necesidades o podemos también instalar nuevas plantillas eligiendo entre las miles que podemos encontrar listas para su descarga gratuita en Internet.

Pero además de añadir páginas a nuestra web o de publicar artículos de noticias, con Joomla podemos gestionar otras muchas funcionalidades y tipología de contenidos. El sistema incorpora de serie una serie de componentes que nos permiten, entre otras muchas cosas

- Insertar y administrar la publicación de anuncios gráficos o banners
- Realizar encuestas en línea a los visitantes o usuarios de nuestra web

- Distribuir noticias en formato RSS
- Sindicar noticias externas procedentes de otros sitios que dispongan de un canal RSS
- Habilitar formularios de contacto para que visitantes y/o usuarios puedan enviar mensajes a los administradores y otros usuarios de la web.
- Enviar mensajes masivos a todas las personas registradas en el sistema.

## Extensiones

Existen 5 tipos de extensiones distintas en Joomla que permiten añadir otras muchas utilidades y funcionalidades para manejar otro tipo de contenidos o modificar aspecto visual de nuestra web:

- Módulos
- Componentes
- Plugins
- Plantillas
- Lenguajes.

## Componentes

Un componente es el tipo de extensión más extensa y compleja, son pequeñas aplicaciones que presentan el contenido principal de la página.

Algunas características de Joomla se proporcionan mediante el uso de componentes predeterminados como:

- Contactos
- Página de Inicio
- Noticias Externas
- Banners
- Correo Masivo
- Encuestas
- Sindicación RSS

Los Componentes gestionan datos, muestran información, proporcionan funciones y en general pueden ejecutar cualquier operación que no recaiga en las funciones generales del código del núcleo.

Actúan en conjunto con los Módulos y Plugins para proporcionar una amplia variedad de tipos de información y funcionalidades, al margen de la presentación estándar de artículos

y contenido. Los componentes hacen posible transformar completamente a Joomla y Ampliar en gran medida sus posibilidades.

## Módulos

Una extensión más ligera y flexible para la presentación de contenido es un módulo. Los módulos se usan para pequeñas partes de la página generalmente menos complejos y que pueden asociarse a diferentes componentes. Son pequeñas mini-aplicaciones que pueden situarse en cualquier lugar del sitio. En algunos casos trabajan en conjunción con componentes y en otros son fragmentos de código aislados y completos que se usan para mostrar algunos datos de la base de datos, como Contenido. Los Módulos se utilizan habitualmente para la salida de información pero también pueden ser formularios para la entrada de datos (como ejemplos, el Modulo de Acceso o las Encuestas).

Los módulos se pueden asignar a posiciones de módulo que están definidas tanto en la plantilla como en el Administrador (Back-end) en el artículo de menú Administrador de Posiciones de los Módulos (en el Administrador de Plantillas del Sitio).

Joomla trae por defecto módulos predefinidos, como por ejemplo

El Menú Principal, cuando crea un nuevo artículo de menú en el administrador de menús realmente está copiando el código del módulo Menú Principal y dándole el nombre del nuevo menú. Cuando copia un módulo no copia todos sus parámetros, simplemente permite que Joomla use el mismo código con dos configuraciones diferentes.

Otro ejemplo de modulo predefinido es el “Destacamos” que presenta los artículos de contenido del sitio en una posición de módulo asignable. Puede usarse y configurarse para presentar una categoría, todas las categorías o escoger al azar los artículos de contenido que se mostrarán a los usuarios. Este módulo presenta la porción de artículo que haya seleccionado, y muestra un enlace “Leer más” que permite al usuario ver la página de vista completa.

## Plugins

Una de las extensiones más avanzadas de Joomla es el plugin. En las versiones antiguas, anteriores a la 1.0x se les conocían como mambots. Con el desarrollo de Joomla 1.5, los mambots fueron renombrados y se ha ampliado su funcionalidad. Son una porción de código que se ejecuta cuando tiene lugar un evento predefinido. Los editores son plugins, por ejemplo, que se ejecutan cuando tiene lugar el evento “onGetEditorArea” .Permiten al desarrollador cambiar el modo en que se comporta el código, dependiendo de los plugins instalados para reaccionar a un determinado evento.

La extensión más básica, creada a partir de la versión 1.5, se empaquetan bien como paquetes de lenguaje o como paquetes de extensión de lenguaje. Estos permiten internacionalizar tanto el núcleo de Joomla como los componentes y módulos de terceras partes.

Existen infinidad de extensiones adicionales desarrolladas por la comunidad Joomla que nos permiten añadir nuevas prestaciones y funcionalidades a nuestro portal, además de tener la opción de crear nosotros una nueva extensión e insertarla. Esta facilidad para añadir extensiones y componentes adicionales otra de las grandes bazas a favor de Joomla. La comunidad de desarrolladores de Joomla es además muy amplia y cada día aparecen nuevas extensiones y plugins. Incorporar un sistema de foros para debates o consultas, añadir galerías de fotos a las que los usuarios puedan enviar sus propias imágenes o crear un sitio multilingüe son algunas de las posibilidades más interesantes

# Eclipse

Eclipse es un IDE de código abierto multiplataforma, típicamente ha sido usada para desarrollar entornos de desarrollo integrados, como el IDE de Java llamado Java Development Toolkit (JDT) y el compilador (ECJ) que se entrega como parte de Eclipse (y que son usados también para desarrollar el mismo Eclipse).

Eclipse es también una comunidad de usuarios, extendiendo constantemente las áreas de aplicación cubiertas. Un ejemplo es el recientemente creado Eclipse Modeling Project, cubriendo casi todas las áreas de Model Driven Engineering.

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Eclipse es ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios.

La versión actual de Eclipse dispone de las siguientes características: Editor de texto, resaltado de sintaxis, compilación en tiempo real, pruebas unitarias con JUnit, control de control de versiones con CVS, integración con Ant, asistentes para creación de proyectos, clases, test y refactorización.

Asimismo, a través de plugins libremente disponibles es posible añadir nuevas funcionalidades, que será así, como añadiremos Monet

# XML

XML, siglas en inglés de eXtensible Markup Language ('lenguaje de marcas extensible'), es un lenguaje de marcas desarrollado por el World Wide Web Consortium (W3C). Deriva del lenguaje SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML) para estructurar documentos grandes. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones se deben comunicar entre sí o integrar información.

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 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.

- 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.

- Si un tercero decide usar un documento creado en XML, es sencillo entender su estructura y procesarla. Mejora la compatibilidad entre aplicaciones. Podemos comunicar aplicaciones de distintas plataformas, sin que importe el origen de los datos, es decir, podríamos tener una aplicación en Linux con una base de datos Postgres y comunicarla con otra aplicación en Windows y Base de Datos MS-SQL Server.

- Transformamos datos en información, pues se le añade un significado concreto y los asociamos a un contexto, con lo cual tenemos flexibilidad para estructurar documentos.

La tecnología XML busca dar solución al problema de expresar información estructurada de la manera más abstracta y reutilizable posible. Que la información sea estructurada quiere decir que se compone de partes bien definidas, y que esas partes se componen a su vez de otras partes. Entonces se tiene un árbol de trozos de información. Ejemplos son un tema musical, que se compone de compases, que están formados a su vez por notas. Estas partes se llaman elementos, y se las señala mediante etiquetas.

Una etiqueta consiste en una marca hecha en el documento, que señala una porción de éste como un elemento. Un pedazo de información con un sentido claro y definido. Las etiquetas tienen la forma <nombre>, donde nombre es el nombre del elemento que se está señalando.

## Documentos XML bien formados y control de errores

Los documentos denominados como «bien formados» son aquellos que cumplen con todas las definiciones básicas de formato y pueden, por lo tanto, analizarse correctamente por cualquier analizador sintáctico (parser) que cumpla con la norma. Se separa esto del concepto de validez que se explica más adelante.

- Los documentos han de seguir una estructura estrictamente jerárquica con lo que respecta a las etiquetas que delimitan sus elementos. Una etiqueta debe estar correctamente incluida en otra, es decir, las etiquetas deben estar correctamente anidadas. Los elementos con contenido deben estar correctamente cerrados.

- Los documentos XML sólo permiten un elemento raíz del que todos los demás sean parte, es decir, solo pueden tener un elemento inicial.

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

- El XML es sensible a mayúsculas y minúsculas. Existe un conjunto de caracteres llamados espacios en blanco (espacios, tabuladores, retornos de carro, saltos de línea) que los procesadores XML tratan de forma diferente en el marcado XML.

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

- 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.

# Java

Es un lenguaje de programación desarrollado por James Gosling de Sun Microsystems (la cual fue adquirida por la compañía Oracle) y publicado en el 1995 como un componente fundamental de la plataforma Java de Sun Microsystems. Su sintaxis deriva mucho de C y C++, pero tiene menos facilidades de bajo nivel que cualquiera de ellos

- Es multiplataforma: un programa Java realizado en Windows funcionará igual que si se ejecuta en Linux, con tal de que en ambos sistemas la máquina virtual de java (JVM) se encuentre correctamente instalada y sean de las mismas versiones.

- Es multisistema: una aplicación Java también funcionará perfectamente ya se ejecute en un Pc o en un Mac.

- Es un lenguaje de alto nivel orientado a objetos: La orientación a objetos (POO) es un paradigma de programación que utiliza el concepto de objeto para la construcción de aplicaciones. Los conceptos básicos de la orientación a objetos son:

    - Objeto: sería una entidad con un conjunto de propiedades y métodos que definen su comportamiento.

    - Clase: definición de las propiedades y los métodos de un objeto. Los objetos se crean (se instancian) a partir de ellas.

    - Mensaje: la comunicación entre un objeto y otro. Ésta se consigue cuando un objeto ejecuta un método del otro.

    - Propiedad: representa el estado del objeto, que puede ser alterado a través de sus métodos.

    - Método: descripción de una acción que el objeto realizará cuando reciba un mensaje.

    - Este tipo de paradigma de programación permite características avanzadas que facilitan la labor del programador al mismo tiempo que ofrecen una gran potencia (como recolección de basura, tratamiento de errores, etc.) siguiendo la forma de programar de lenguajes como C y C++, con lo que su aprendizaje y manejo son relativamente sencillos.

- Está extensamente documentada: cualquier clase que se desee utilizar contará, casi con total seguridad, con la correspondiente documentación Javadoc explicando su funcionamiento y forma de uso. Los dos primeros aspectos (clave para la elección de Java en este proyecto, ya que deberá funcionar tanto en Windows como en Linux) son debidos a su naturaleza de lenguaje interpretado, esto es, la compilación de su código fuente no genera código máquina directamente, sino un pseudocódigo interpretable por la JVM, que será quien lo ejecute como si fuera un "ordenador" que a su vez se ejecuta dentro de nuestro ordenador.

- Tiene una amplia librería de clases: los programadores Java tienen a su disposición una enorme variedad de clases y utilidades proporcionadas tanto por Sun como por desarrolladores independientes. Java es el lenguaje que mayor cantidad de librerías ofrece a sus desarrolladores en todo el mundo.

En nuestro proyecto hemos utilizado las siguientes librerías:

- **Apache Axis (axis-1.4.jar):** Apache Axis es una implementación del protocolo SOAP. Esta librería nos provee de la clase necesaria para conectar con el servicio web (org.apache.axis.client.Service) así como la clase que nos permitirá invocar los distintos métodos que nos ofrece el webservice (org.apache.axis.client.Call)

- **Java API for XML-based RPC (JAX-RPC) (axis-jaxrpc-1.1.0.jar)**: La API Java para RPC basado en XML nos permite invocar un servicio web basado con una

descripción conocida sin dejar de ser consistente con su descripción WSDL.
Necesitaremos esta librería para el acceso a las clases: javax.xml.rpc.ParameterMode, javax.xml.rpc.ServiceException,java.net.MalformedURLException.

- **Web Services Description Language for Java Toolkit (javax.wsdl_1.6.2.v201012040545.jar):** Este paquete de herramientas ofrece la capacidad de creación, representación y manipulación de documentos WSDL.
  En nuestro código será necesaria para el acceso a la clasejavax.wsdl.OperationType, de lo contrario recibiremos una excepción en tiempo de ejecución, de tipo ClassNotFoundException, durante la invocación de los métodos ofrecidos por el webservice.

- **Commons Discovery: Service Discovery component(commons-discovery-0.2.jar):** Proporciona comodidades para crear instancias de las clases en general, y para la gestión del ciclo de vida de los clases bajo el patrón singleton. Fundamentalmente, Commons Discovery localiza clases que implementen una interfaz Java dada.
  Necesitaremos esta librería para evitar un error de tipo ClassNotFoundException al intentar acceder a la claseinterface.org.apache.commons.discovery.tools.DiscoverSingleton.

- **Logging Component(org.apache.commons.logging_1.0.4.v201101211617.jar)**: El paquete Logging Component es un ligero puente entre las distintas implementaciones de registro. Commons-logging ofrece soporte para las implementaciones más populares de registro, y facilita notablemente la escritura de adaptadores para otras.
  En este caso, necesitaremos declarar dicha librería para acceder a la clase (task.org.apache.commons.logging.LogFactory).

- **SOAP with Attachments API for Java(Saaj.jar):** Suministra una manera estándar de enviar documentos XML sobre Internet desde la plataforma Java.

# Metodología

## Modelos desarrollo evolutivo

Los modelos de desarrollo evolutivo se basan en el desarrollo de sucesivas versiones del sistema, donde no es necesario que todos los requisitos este definidos desde el inicio, existen multitud de modelos, nosotros utilizaremos el modelo en espiral que explicaremos a continuación.

### Desarrollo en espiral

En este modelo, los productos software son creados a través de múltiples repeticiones del proceso del ciclo de vida, dividiéndose éstas en "mini-proyectos". Cada vuelta en la espiral representa una fase del proceso, es decir, supone un avance en el proceso de desarrollo. No hay "etapas" fijas tradicionales, ligadas a actividades como la especificación o diseño, cada vuelta en la espiral determina las actividades a realizar. El desarrollo en espiral se puede dividir en las siguientes fases o etapas:

- Determinar objetivos: En esta fase se definen los objetivos específicos identificando las restricciones del proceso y el producto, y se traza un plan detallado de gestión. También se identifican los riesgos del proyecto, planteando estrategias alternativas en función de los mismos.

- Evaluación de riesgos. Se lleva a cabo un análisis detallado para cada uno de los riesgos identificados en el proyecto, definiendo los pasos necesarios para reducirlos

- Desarrollo y validación. Se elige un modelo para el desarrollo del sistema. Es en esta fase donde tiene sentido la utilización del modelo de desarrollo evolutivo. Gracias a este modelo, tras las iteraciones oportunas, se obtendrá un prototipo válido que dé por concluida esta etapa.

- Planificación. El proyecto se revisa y se toma la decisión de si se debe continuar con un ciclo posterior de la espiral. Si se decide continuar se desarrolla la planificación de la siguiente fase del proyecto.

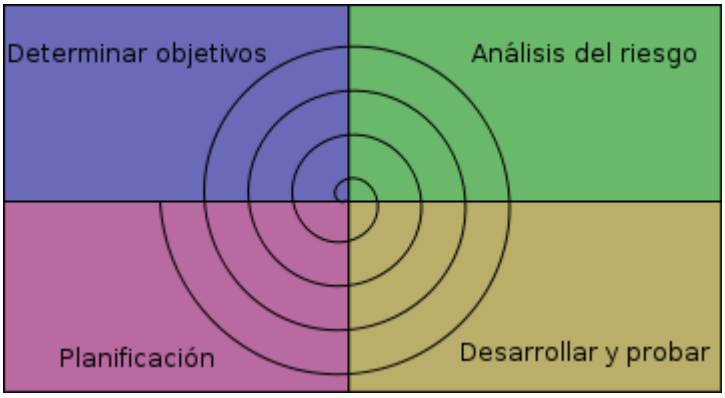



Figura 7: Iteraciones en modelos evolutivos

En nuestro caso no ha habido ningún tipo de plazos que cumplir, el planteamiento ha sido terminar el proyecto cumpliendo con nuestras expectativas.

La definición de objetivos del proyecto ha quedado marcada por la interacción con el personal del Club de Baloncesto Gran Canaria y la dirección de mi tutor.

La evaluación y reducción de riesgos se ha llevado a cabo por mi parte y la del tutor del proyecto. Donde se han identificado los principales riesgos críticos del desarrollo para poder evitarlos.

En cuanto a la fase de desarrollo y validación hay que destacar la aplicación del modelo de proceso evolutivo, que explicaré en una sección posterior debido a la importancia de esta fase.

En último lugar, y como el modelo de desarrollo en espiral determina, se lleva a cabo un plan de elaboración de la siguiente fase, es decir, tiene lugar la planificación. Este plan se elabora en base a los resultados obtenidos en la fase anterior. El prototipo es evaluado en profundidad, si los resultados han sido satisfactorios, la siguiente etapa se planteara como un incremento de la funcionalidad del sistema, en caso contrario intentaremos solventar los errores o buscar soluciones alternativas.

# Ingeniería dirigida de modelos

El desarrollo de Software dirigido por modelos (DSDM) es una propuesta para la construcción de software en la que se le atribuye a los modelos el papel principal de todo el proceso, frente a las propuestas tradicionales basadas en lenguajes de programación, plataformas de objetos y componentes software.

Persigue elevar el nivel de abstracción en el desarrollo de software, convirtiendo a los modelos y a las transformaciones entre ellos en los principales artefactos de todas las fases del proceso de desarrollo de software: captura y gestión de los requisitos, diseño, análisis, implementación, despliegue, configuración, mantenimiento, evolución, etc.

Esta iniciativa ha evolucionado originado distintos paradigmas, entre ellos el Model Driver Engineering (Ingeniería Dirigida por Modelos) conocida por sus siglas MDE, y que hace referencia al uso sistemático de modelos, como los elementos principales en la ingeniería del software, durante el ciclo de vida completo del proyecto.

Concibe al desarrollo de software bajo la idea central de que los artefactos son los modelos (y no los programas).

Las principales tecnologías de la ingeniería dirigida por modelos son:

- DSML (Lenguajes específicos de dominio). Formalizan la estructura, comportamiento y requisitos mediante dominios particulares, como por ejemplo misiones de aviónica, servicios financieros, gestión de almacenes o incluso dominios de plataformas de middleware. Los DSMLs se describen utilizando meta-modelos, que definen las relaciones entre conceptos en un dominio y especifican con precisión la semántica y las restricciones asociadas con estos conceptos.

- Motores de transformación y generadores. Analizan ciertos aspectos de los modelos y sintetizan varios tipos de artefactos, tales como código fuente, descriptores en XML o representaciones alternativas de los modelos. La capacidad de construir artefactos a partir de modelos asegura la consistencia entre la implementación de la aplicación y la información del análisis funcional capturada por los modelos.

Un modelo formal debe contener suficiente información para permitir la generación de una herramienta automática para comprobar y construir modelos sujetos al formalismo descrito. La ventaja del uso de meta-modelos es claro, en lugar de construir una aplicación desde cero, sólo se necesita especificar el tipo de modelos que se van a emplear. Si la especificación se hace gráficamente, el tiempo para desarrollar una aplicación puede reducirse a una pocas horas. Otros beneficios, tales como la reducción de las pruebas, facilidad de cambio y mantenimiento también se logran.

En modelado y simulación existe gran interés en manipulaciones de los modelos como:

- Simulación de modelos.

- Optimización de modelos, por ejemplo reducir su complejidad.

- Transformación de modelos en otros (equivalentes en comportamiento) con diferentes formalismos. Esto puede ser útil para diferentes situaciones:

    - En un modelo compuesto por diferentes componentes heterogéneos, es posible transformar cada uno de estos componentes en un formalismo común. Luego, una vez que todos los componentes están descritos en el mismo formalismo se puede procesar el significado completo del sistema. Esta aproximación es el núcleo de la propuesta de multi formalismos para modelar sistemas complejos

    - Para resolver problemas que son más simples en otros formalismos. Un ejemplo es el caso del modelo ODE que no tiene una solución analítica. Para

resolver este problema transformamos este modelo en ecuaciones en diferencias (que normalmente se resuelve numéricamente) y resolverlas en este dominio. En la desratización se perderá algo de información.

- Generación de código para simuladores existentes.

# Arquitectura orientada a servicios (SOA)

La Arquitectura Orientada a Servicios de cliente, es un concepto de arquitectura de software que define la utilización de servicios para dar soporte a los requisitos del negocio. Permite la creación de sistemas altamente escalables que reflejan el negocio de la organización, a su vez brinda una forma bien definida de exposición e invocación de servicios (comúnmente pero no exclusivamente servicios web), lo cual facilita la interacción entre diferentes sistemas propios o de terceros. SOA define las siguientes capas de software:

- Aplicaciones básicas - Sistemas desarrollados bajo cualquier arquitectura o tecnología, geográficamente dispersos y bajo cualquier figura de propiedad

- De exposición de funcionalidades - Donde las funcionalidades de la capa aplicativa son expuestas en forma de servicios (generalmente como servicios web)

- De integración de servicios - Facilitan el intercambio de datos entre elementos de la capa aplicativa orientada a procesos empresariales internos o en colaboración;

- De composición de procesos - Que define el proceso en términos del negocio y sus necesidades, y que varía en función del negocio;

- De entrega - donde los servicios son desplegados a los usuarios finales.

## Elementos esenciales de una Arquitectura Orientada a Servicios

En las Arquitecturas Orientadas a Servicios, el elemento básico es el servicio. Pero únicamente con este concepto, no podríamos diseñar una arquitectura SOA.
Cuatro son los elementos esenciales necesarios para la construcción de una Arquitectura Orientada a Servicios:

- Operación: Es la unidad de trabajo o procesamiento en una arquitectura SOA.

- Servicio: Es un contenedor de lógica. Estará compuesto por un conjunto de operaciones, las cuales las ofrecerá a sus usuarios.

- Mensaje: Para poder ejecutar una determinada operación, es necesario un conjunto de datos de entrada. A su vez, una vez ejecutada la operación, esta devolverá un resultado. Los mensajes son los encargados de encapsular esos datos de entrada y de salida.

- Proceso de negocio: Son un conjunto de operaciones ejecutadas en una determinada secuencia (intercambiando mensajes entre ellas) con el objetivo de realizar una determinada tarea.

Por lo tanto, una aplicación SOA estará formada por un conjunto de procesos de negocio. A su vez esos procesos de negocio estarán compuestos por aquellos que servicios que proporcionan las operaciones que se necesitan ejecutar para que el proceso de negocio llegue a buen término. Por último para ejecutar esas operaciones es necesario el envío de los datos necesarios mediante los correspondientes mensajes.

### Principios

- **Los Servicios deben ser reusables:** Todo servicio debe ser diseñado y construido pensando en su reutilización dentro de la misma aplicación, dentro del dominio de aplicaciones de la empresa o incluso dentro del dominio público para su uso masivo.

- **Los Servicios deben proporcionar un contrato formal:** Todo servicio desarrollado, debe proporcionar un contrato en el cual figuren: el nombre del servicio, su forma de acceso, las funcionales que ofrece, los datos de entrada de cada una de las funcionalidades y los datos de salida. De esta manera, todo consumidor del servicio, accederá a este mediante el contrato, logrando así la indepencia entre el consumidor y la implementación del propio servicio. En el caso de los Servicios Web, esto se logrará mediante la definición de interfaces con WSDL.

- **Los Servicios deben tener bajo acoplamiento:** Logrando así la independencia entre el servicio que se va a ejecutar y el que lo llama. Si conseguimos este bajo acoplamiento, entonces los servicios podrán ser totalmente reutilizables.

- **Los Servicios deben permitir la composición:** Todo servicio debe ser construido de tal manera que pueda ser utilizado para construir servicios genéricos de más alto nivel, el cual estará compuesto de servicios de más bajo nivel.

- **Los Servicios deben de ser autónomos:** Todo Servicio debe tener su propio entorno de ejecución.

- **Los Servicios no deben tener estado:** Un servicio no debe guardar ningún tipo de información. La solución, es que un servicio sólo contenga lógica, y que toda información esté almacenada en algún sistema de información sea del tipo que sea.

- **Los Servicios deben poder ser descubiertos:** En el caso de los Servicios Web, el descubrimiento se logrará publicando los interfaces de los servicios en registros UDDI.

## Diseño y desarrollo de SOA

La metodología de modelado y diseño para aplicaciones SOA se conoce como análisis y diseño orientado a servicios. La arquitectura orientada a servicios es tanto un marco de trabajo para el desarrollo de software como un marco de trabajo de implementación. Para que un proyecto SOA tenga éxito los desarrolladores de software deben orientarse ellos mismos a esta mentalidad de crear servicios comunes que son orquestados por clientes o middleware para implementar los procesos de negocio. El desarrollo de sistemas usando SOA requiere un compromiso con este modelo en términos de planificación, herramientas e infraestructura. Cuando la mayoría de la gente habla de una arquitectura orientada a servicios están hablando de un juego de servicios residentes en Internet o en una intranet, usando servicios web. Existen diversos estándares relacionados a los servicios web. Incluyen los siguientes:

- XML
- HTTP
- SOAP
- WSDL
- UDDI

Hay que considerar, sin embargo, que un sistema SOA no necesariamente necesita utilizar estos estándares para ser "orientado a servicios" pero es altamente recomendable su uso. En un ambiente SOA, los nodos de la red hacen disponibles sus recursos a otros participantes en la red como servicios independientes a los que tienen acceso de un modo estandarizado. La mayoría de las definiciones de SOA identifican la utilización de Servicios Web (empleando SOAP y WSDL) en su implementación, no obstante se puede implementar SOA utilizando cualquier tecnología basada en servicios.

## Beneficios

Los beneficios que puede obtener una organización que adopte SOA son:

- Mejora en los tiempos de realización de cambios en procesos.
- Facilidad para evolucionar a modelos de negocios basados en tercerización.
- Facilidad para abordar modelos de negocios basados en colaboración con otros entes (socios, proveedores).
- Poder para reemplazar elementos de la capa aplicativa SOA sin disrupción en el proceso de negocio.
- Facilidad para la integración de tecnologías disímiles.

# SOAP

Un servicio Web es una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet.

La importancia de esta tecnología en el proyecto es que la FEB (Federación Española de Baloncesto) dispone de un servicio Web para acceder a sus datos.

Accederemos a dichos datos mediante SOAP (Simple Object Access Protocol), que lo explicaremos a continuación pero resumiendo es un protocolo de comunicación, el cual permite la comunicación entre aplicaciones a través de mensajes por medio de Internet. Es independiente de la plataforma, y del lenguaje. Está basado en XML, y sus mensajes son básicamente documentos XML, y basados en los Web Services mencionados anteriormente.

Sus características principales son:

- Establecer un protocolo estándar de invocación de servicios remotos, basado en protocolos estándares de Internet: HTTP (Protocolo de transporte de Hipertexto) para la transmisión y XML (lenguaje de marcado extensible) para la codificación de datos.

- Independencia de plataforma, lenguaje de desarrollo e implementación (modelo de objetos).

El protocolo de comunicación HTTP es el empleado intrínsecamente para la conexión sobre Internet. Garantiza que cualquier cliente con un navegador estándar pueda conectarse con un servidor remoto. La transmisión de datos se empaqueta con XML, que se ha convertido en el estándar del intercambio de datos, salvando las incompatibilidades entre otros protocolos, tales como el NDR (Network Data Representation) o el CDR (Common Data Representation).

Por otra parte, los servidores Web pueden procesar las peticiones de usuario, empleando las tecnologías de Servlets, paginas ASP (Active Server Pages) o JSP (Java Server Pages), o un servidor de aplicaciones, invocando objetos de tipos CORBA, COM o EJB.

Soap, es un marco extensible y descentralizado que permite trabajar sobre múltiples pilas de protocolos de redes informáticas. Los procedimientos de llamadas remotas pueden ser modelados en la forma de varios mensajes SOAP interactuando entre sí.

Estos mensajes constan de 3 secciones: envelope, header y body.

Dónde:

- envelope (envoltura): Es el elemento raíz del mensaje para describir su contenido y la forma de procesarlo.

- header (encabezado): Es la información de identificación del contenido. Un grupo de reglas de codificación para expresar las instancias de tipos de datos definidos por la aplicación.

- body (cuerpo): Es el contenido del mensaje. Una convención para representar las llamadas y las respuestas a procedimientos remotos.

### Modelo de intercambio de mensajes

Los mensajes SOAP son transmisiones unidireccionales desde un emisor a un receptor.

Se suelen combinar mensajes para implementar patrones, como petición/respuesta.

Las implementaciones SOAP se pueden optimizar para explotar las características específicas de sistemas de red concretos.

### Procesamiento de mensajes

Una aplicación SOAP debe procesar un mensaje siguiendo un orden de acciones:

Identificar las partes del mensaje SOAP dirigido a dicha aplicación.

Aceptar las partes obligatorias identificadas en el paso 1 y procesarlas de la forma adecuada. De lo contrario, descartar el mensaje.

Si la aplicación SOAP no es el destino final del mensaje, quitar todas las partes identificadas en el paso 1 antes de reenviar el mensaje.

También hay que tener en cuenta que este protocolo es extensible.

### Extensiones al protocolo SOAP

SOAP puede ser extendido realizando adiciones de módulos de funcionalidad. Este enfoque permite a los desarrolladores usar los módulos y funcionalidad que ellos necesitan, sin tener la necesidad de implementar la totalidad de estos.

Algunas de las extensiones que pueden ser deseables en los proveedores son las siguientes:

- Attachments – La posibilidad de incluir un documento no XML, archivo binario ó de enviar documentos de fax, o imágenes, codificadas en Base64.

- Routing/Intermediaries – Relacionadas al proceso de rutear mensajes SOAP a través de intermediarios. Esto ofrece la posibilidad de agregar varios Web Services

(WS) y ofrecerlos como parte del paquete, es una manera de hacer a los WS escalables, a través del direccionamiento, incluso hacia múltiples servidores

- Security – Dar un marco de seguridad a la comunicación. Algunos de los aspectos podrían ser aplicar SSL, firma digital, etc. XML referent nos ayuda a responder: quien envía el mensaje y si el mensaje fue alterado en la ruta.

- Quality of Services – QoS es una medida que puede ser comparada con el número o calificación dada a los ASP o ISP, que mide la calidad del servicio, un concepto similar puede manejarse para los Web Services.

- Context/Privacy – Hace referencia a guardar el contexto y privacidad, del entorno de los usuarios.

- Transaction Support – Permitir que un grupo de operaciones o acciones se comporten como si fueran una simple unidad (o todo falla o todo es un éxito).

- Message Syntax – el formato tiene un área separada para extensiones que sean adicionadas.

- Data – SOAP puede contener cualquier tipo de datos. Provee un método para serialización de datos, pero las aplicaciones pueden definir sus propias reglas.

- Transport – No define como son transportados los mensajes durante el intercambio. SOAP muestra como podrían ser intercambiados sobre http, pero cualquier protocolo o método puede sustituir a http.

- Purpose – SOAP no define qué es lo que hay dentro del mensaje. Hay una diferencia entre los datos y su propósito o finalidad.

### Ventajas

- Es sencillo de implementar, probar y usar

- Atraviesa "firewalls" y routers, pues estos "piensan" que es una comunicación HTTP.

- Tanto los datos como las funciones se describen en XML, lo que permite que el protocolo no sólo sea más fácil de utilizar sino que también sea muy sólido.

- Es independiente del sistema operativo y procesador.

- Se puede utilizar tanto de forma anónima como con autenticación (nombre/clave).

- Facilidad para utilizar cualquier lenguaje: Los desarrolladores involucrados en nuevos proyectos pueden elegir desarrollar con el último y mejor lenguaje de programación que exista. SOAP no especifica una API, por lo que la

implementación de la API se deja al lenguaje de programación, como en Java, y la plataforma como Microsoft .Net.

- No se encuentra fuertemente asociado a ningún protocolo de transporte: La especificación de SOAP no describe como se deberían asociar los mensajes de SOAP con HTTP. Un mensaje de SOAP no es más que un documento XML, por lo que puede transportarse utilizando cualquier protocolo capaz de transmitir texto.

- No está atado a ninguna infraestructura de objeto distribuido: La mayoría de los sistemas de objetos distribuidos se pueden extender, y alguno de ellos admiten SOAP.

- Aprovecha los estándares existentes en la industria: Los principales contribuyentes a la especificación SOAP evitaron, intencionalmente, reinventar las cosas. Optaron por extender los estándares existentes para que coincidieran con sus necesidades. Por ejemplo, SOAP aprovecha XML para la codificación de los mensajes, en lugar de utilizar su propio sistema de tipo que ya están definidas en la especificación esquema de XML. Y como ya se ha mencionado SOAP no define un medio de trasporte de los mensajes, los mensajes de SOAP se pueden asociar a los protocolos de transporte existentes como HTTP y SMTP.

- Permite la interoperabilidad entre múltiples entornos: SOAP se desarrolló sobre los estándares existentes de la industria, por lo que las aplicaciones que se ejecuten en plataformas con dichos estándares pueden comunicarse mediante mensaje SOAP con aplicaciones que se ejecuten en otras plataformas. Por ejemplo, una aplicación de escritorio que se ejecute en un PC puede comunicarse con una aplicación del back-end ejecutándose en un mainframe capaz de enviar y recibir XML sobre HTTP.

### Desventajas

Entre las desventajas de SOAP se tiene que:

- Las desventajas de la utilización de SOAP recaen en la dificultad para entender las especificaciones del protocolo, puesto que es un complejo esquema de codificación en el cual es necesario precisar que todos los mensajes se incluyan en un sobre, con el contenido del mensaje dentro de un elemento de cuerpo para que puedan ser entendidos por cada una de las aplicaciones Web que procesan el mensaje.

- SOAP convierte en opcionales elementos como encabezados y ofrece un amplio margen con respecto a lo que se puede incluir en el elemento de cuerpo y además cambia los nombres de métodos en etiquetas secundarias del cuerpo y los argumentos en etiquetas secundarias del nombre del método, lo que puede generar ciertos problemas de interoperabilidad.

- Las especificaciones SOAP indican que si recibe un encabezado SOAP con un atributo mustUnderstand establecido como "1", deberá entenderlo o generar un error. Numerosas implementaciones no lo hicieron al principio lo que implicó problemas de interoperabilidad.

### Estandarización

Los WS están basados en el estándar XML, que ha sido universalmente aceptado. SOAP es el único protocolo que ha sido aceptado en este momento por el World Wide Web Consortium y se encuentra estandarizado. SOAP y WSDL están siendo ampliamente usados.

Algunas de las empresas más importantes en el desarrollo de Negocio Electrónico como IBM, Intel, Microsoft u Oracle, han creado el WS-I:organización para la Interoperabilidad de los Web Services. El objetivo de dicha organización es la promoción de la estandarización de los WS de modo que se fomente la cooperación e interoperabilidad entre las compañías y mercados utilizando este protocolo.

Para acceder a esta información he utilizado librerías Soap para eclipse, que es un kit de herramientas para desarrollar Web Services bajo el lenguaje Java. . Provee soporte para el desarrollo de clientes (aquellos que consumen los Web Services) y de servidores (aquellos que los proveen).

## WSDL

El esquema XML por sí solo no puede describir totalmente un Servicio Web.
En un esfuerzo para describir cómo interacciona un cliente con el servicio Web se define un esquema para los mensajes que se intercambiarán entre el cliente y el servidor. El esquema contiene una definición de un tipo de complejo para los mensaje de petición y repuesta para los métodos definidos. Teniendo en cuenta que el objetivo último es que los desarrolladores no tengan que investigar en las definiciones del esquema intentando descifrar cómo interaccionar con el servicio Web. En lugar de ello se quiere describir el servicio de forma que una herramienta pueda descifrarlo y crear un proxy por el cliente.

Como el cuerpo de un mensaje de SOAP puede contener cualquier cosa que no invalide el XML los mensajes de SOAP se pueden combinar para disponer de una amplia variedad de patrones de intercambio de mensajes. Mediante una asociación formal en los patrones de intercambio de mensajes para el Servicio Web y sus mensajes de respuesta asociados eliminarían cualquier posible ambigüedad.

Una descripción formal de los patrones de mensaje resulta más importante en el caso de Servicio Web complejos. Algunos servicios podrían aceptar una petición pero no enviar la respuesta correspondiente devuelta al cliente. Otros podrían solamente enviar mensajes al

cliente.
Además, el esquema no contiene información sobre cómo acceder al Servicio Web. Como SOAP es independiente del protocolo, se intercambiarán los mensajes entre el cliente y el servidor de numerosas formas.

El WSDL es un dialecto basado en XML sobre el esquema que describe un servicio Web. Un documento WSDL proporciona la información necesaria al cliente para interaccionar con el servicio Web. WSDL es extensible y se pude utilizar para describir, prácticamente, cualquier servicio de red, incluyendo SOAP sobre HTTP e incluso protocolos que no se basan en XML como DCOM sobre UDP.

Dado que los protocolos de comunicaciones y los formatos de mensajes están estandarizados en la comunidad del Web, cada día aumenta la posibilidad e importancia de describir las comunicaciones de forma estructurada. WSDL afronta esta necesidad definiendo una gramática XML que describe los servicios de red como colecciones de puntos finales de comunicación capaces de intercambiar mensajes. Las definiciones de servicio de WSDL proporcionan documentación para sistemas distribuidos y sirven como fórmula para automatizar los detalles que toman parte en la comunicación entre aplicaciones.

Los documentos WSDL definen los servicios como colecciones de puntos finales de red o puertos. En WSDL, la definición abstracta de puntos finales y de mensajes se separa de la instalación concreta de red o de los enlaces del formato de datos. Esto permite la reutilización de definiciones abstractas: mensajes, que son descripciones abstractas de los datos que se están intercambiando y tipos de puertos, que son colecciones abstractas de operaciones. Las especificaciones concretas del protocolo y del formato de datos para un tipo de puerto determinado constituyen un enlace reutilizable. Un puerto se define por la asociación de una dirección de red y un enlace reutilizable; una colección de puertos define un servicio. Por esta razón, un documento WSDL utiliza los siguientes elementos en la definición de servicios de red:

### Elemento types

El elemento Types contiene información de esquema referenciado en el documento WSDL. El sistema de tipos predeterminado que admite WSDL es de esquema de XML. Si se usa esquema de XML para definir los tipos que contiene el elemento Types el elemento schema aparecerá inmediatamente como elemento hijo.

Se pueden utilizar otros sistemas de tipo tipos por extensión. Si desea, utilizar otro sistema de tipo pude aparecer un elemento de extensibilidad bajo el elemento Types. El nombre de este elemento debería identificar el sistema de tipos utilizados. En este capítulo se limitará a tratar el esquema de XML porque es el sistema de tipos dominante en los documento WSDL.

Elemento message

El elemento Message proporciona una abstracción común para el paso de mensajes entre el cliente y el servidor. Como puede utilizar múltiples formatos de definición de esquema en documento WSDL es necesario de disponer de un mecanismo común de identificar los mensajes. El elemento Message proporciona este nivel común de abstracción al que se hará referencia en otras partes del documento WSDL.

Pude Aparecer, y normalmente aparecerán, múltiples elementos Message en un documento WSDL, uno para cada mensaje que se comunica entre el cliente y el servidor. Cada mensaje contiene uno o más elementos "Part" que describen las piezas del contenido del mensaje. Un ejemplo de una parte es el cuerpo de un mensaje de SOAP o un parámetro que forma parte de una cadena de petición, un parámetro codificado en el cuerpo del mensaje de SOAP o todo el cuerpo de un mensaje de SOAP.

Elemento portType

El elemento porType contiene un conjunto de operaciones abstractas que representan los tipos de correspondencia que pueden producirse entre el cliente y el servidor. Para los Servicios Web de estilo RPC se pude pensar en un porType como una definición de internas en donde cada método se pude definir como una operación.

Un tipo puerto se compone de un conjunto de electos operation que define una determinada acción. Los electos operation se componen de mensajes definidos en el documento WSDL. WSDL define cuatro tipos de operaciones denominadas tipo operaciones:

- Request-response (petición-respuesta) comunicación del tipo RPC en la que el cliente realiza una petición y el servidor envía la correspondiente respuesta.

- One-way (un sentido) Comunicación del estilo documento en la que el cliente envía un mensaje pero no recibe una respuesta del servidor indicando el resultado del mensaje procesado.

- Solicit-response (solicitud-respuesta) La contraria a la operación petición-respuesta. El servidor envía una petición y el cliente le envía de vuelta una respuesta.

- Notification(notificación) La contraria a la operación un sentido, el servidor envía una comunicación estilo documento al cliente

### Elemento binding

El elemento binding contiene las definiciones de la asociación de un protocolo como SOAP a un determinado bindingType. Las definiciones binding especifican detalles de formatos del mensaje y el protocolo. Por ejemplo, la información de asociación especifica si se puede acceder a una instancia de un portType de forma RPC.

Las definiciones binding también indican el número de comunicaciones de re red que se requieren para realizar una determinada acción. Por ejemplo, una llamada RPC de SOAP sobre HTTP podría involucrar un intercambio de comunicación HTTP, pero esa misma llamada sobre SMTP podría involucrar dos intercambios de comunicaciones de SMTP discretas.

La asociación de logra utilizando elementos de extensión. Cada protocolo tiene su propio conjunto de elementos de extensión para especificar los detalles del protocolo y el formato de los mensajes. Para un determinado protocolo los elementos de extensión se suelen utilizar para decorar las acciones individuales de una operación y la propia operación con la información de asociación del protocolo. A veces los elementos de extensión se utilizan en el propio nivel portType.

### Elemento service

Un servicio es un grupo de puertos relacionados y se definen en el elemento service. Un puerto es un extremo concreto de un Servicio Web al que se hace referencia por una dirección única. Los puertos que se definen en determinado servicio son independientes. Por ejemplo, la salida de un puerto que no puede utilizarse como una entrada de otro.

### Elementos de Extensibilidad

Los elementos de extensibilidad se utilizan para representar determinadas tecnologías. Por ejemplo, se puede utilizar los elementos de extensibilidad para especificar el idioma en que se utiliza en el esquema de los elementos types.

El esquema para un determinado conjunto de elementos de extensibilidad se debe definir dentro de distintos espacios de nombres que WSDL. La definición de los propios elementos puede contener un atributo wsdl:requiered que indique un valor boolean si el atributo requiered se establece a true en una definición de elementos una asociación que haga referencia a ese conjunto concreto de electos de extensibilidad tiene que incluir dicho elemento

Lo más habitual es que los elementos de extensibilidad se utilicen para especificar especificación de asociación. La especificación WSDL define conjunto de elementos de

extensibilidad para la asociación SOAP, HTTP GET, HTTP POS, MIME. Sin embargo, la especificación sólo define las asociaciones para dos de los cuatro tipos de operaciones. Un sentido y petición repuesta.

# Rest

REST (Representational State Transfer) es un estilo de arquitectura de software para sistemas hipermedias distribuidos tales como la Web. El término fue introducido en la tesis doctoral de Roy Fielding en 2000, quien es uno de los principales autores de la especificación de HTTP.
En realidad, REST se refiere estrictamente a una colección de principios para el diseño de arquitecturas en red. Estos principios resumen como los recursos son definidos y diseccionados. El término frecuentemente es utilizado en el sentido de describir a cualquier interfaz que transmite datos específicos de un domino sobre HTTP sin una capa adicional, como hace SOAP.
Cabe destacar que REST no es un estándar, ya que es tan solo un estilo de arquitectura.
Aunque REST no es un estándar, está basado en estándares:

- HTTP
- URL
- Representación de los recursos: XML/HTML/GIF/JPEG/…
- Tipos MIME: text/xml, text/html, …

## Principios

- Escalabilidad de la interacción con los componentes. La Web ha crecido exponencialmente sin degradar su rendimiento. Una prueba de ellos es la variedad de clientes que pueden acceder a través de la Web: estaciones de trabajo, sistemas industriales, dispositivos móviles,…

- Generalidad de interfaces. Gracias al protocolo HTTP, cualquier cliente puede interactuar con cualquier servidor HTTP sin ninguna configuración especial. Esto no es del todo cierto para otras alternativas, como SOAP para los Servicios Web.

- Puesta en funcionamiento independiente. Este hecho es una realidad que debe tratarse cuando se trabaja en Internet. Los clientes y servidores pueden ser puestas en funcionamiento durante años. Por tanto, los servidores antiguos deben ser capaces de entenderse con clientes actuales y viceversa. Diseñar un protocolo que permita este tipo de características resulta muy complicado. HTTP permite la extensibilidad mediante el uso de las cabeceras, a través de las URIs, a través de la habilidad para crear nuevos métodos y tipos de contenido.

- Compatibilidad con componentes intermedios. Los más populares intermediaros son varios tipos de proxys para Web. Algunos de ellos, las caches, se utilizan para mejorar el rendimiento. Otros permiten reforzar las políticas de seguridad:

firewalls. Y por último, otro tipo importante de intermediarios, gateway, permiten encapsular sistemas no propiamente Web. Por tanto, la compatibilidad con intermediarios nos permite reducir la latencia de interacción, reforzar la seguridad y encapsular otros sistemas.

- REST logra satisfacer estos objetivos aplicando cuatro restricciones:

- Identificación de recursos y manipulación de ellos a través de representaciones.

- Esto se consigue mediante el uso de URIs. HTTP es un protocolo centrado en URIs. Los recursos son los objetos lógicos a los que se le envían mensajes. Los recursos no pueden ser directamente accedidos o modificados. Más bien se trabaja con representaciones de ellos. Cuando se utiliza un método PUT para enviar información, se coge como una representación de lo que nos gustaría que el estado del recurso fuera. Internamente el estado del recurso puede ser cualquier cosa desde una base de datos relacional a un fichero de texto.

- Mensajes autodescriptivos. REST dicta que los mensajes HTTP deberían ser tan descriptivos como sea posible. Esto hace posible que los intermediarios interpreten los mensajes y ejecuten servicios en nombre del usuario. Uno de los modos que HTTP logra esto es por medio del uso de varios métodos estándares, muchos encabezamientos y un mecanismo de direccionamiento. Además saben cómo consultar las cabeceras para controlar la caducidad de la información. HTTP es un protocolo sin estado y cuando se utiliza adecuadamente, es posible interpretar cada mensaje sin ningún conocimiento de los mensajes precedentes.

- Hipermedia como un mecanismo del estado de la aplicación. El estado actual de una aplicación Web debería ser capturado en uno o más documentos de hipertexto, residiendo tanto en el cliente como en el servidor. El servidor conoce sobre el estado de sus recursos, aunque no intenta seguirle la pista a las sesiones individuales de los clientes. Esta es la misión del navegador, él sabe cómo navegar de recurso a recurso, recogiendo información que el necesita o cambiar el estado que el necesita cambiar.

- En la actualidad existen millones de aplicaciones Web que implícitamente heredan estas restricciones de HTTP. Hay una disciplina detrás del diseño de sitios Web escalables que puede ser aprendida de los documentos de arquitectura Web o de varios estándares.

- Por otra parte, también es verdad que muchos sitios Web comprometen uno más de estos principios, como por ejemplo, seguir la pista de los usuarios moviéndose a través de un sitio. Esto es posible dentro de la infraestructura de la Web, pero daña la escalabilidad, volviendo un medio sin conexión en todo lo contrario.

Los defensores de REST han creído que estas ideas son tan aplicables a los problemas de integración de aplicaciones como los problemas de integración de hipertexto.
Fielding es bastante claro diciendo que REST no es la cura para todo. Algunas de estas características de diseño no serán apropiadas para otras aplicaciones. Sin embargo,

aquellos que han decidido adoptar REST como un modelo de servicio Web sienten que al menos articula una filosofía de diseño con fortaleza, debilidades y áreas de aplicabilidad documentada.

## Soap vs Rest

Muchos diseñadores de Servicios Web están llegando a la conclusión que SOAP es demasiado complicado. Por tanto, están comenzando a utilizar Servicios Web basados en REST para mostrar cantidades de datos masivos. Este es el caso de grandes empresas como eBay y Google.

El problema principal surge del propósito inicial de SOAP. Esta tecnología fue originalmente pensada para ser una versión, sobre Internet, de DCOM o CORBA. Así lo demuestra su predecesor, el protocolo XML-RPC. Estas tecnologías lograron un éxito limitado antes de ser adaptadas. Esto es debido a que este tipo de tecnologías, las basadas en modelos RPC (Remote Procedure Call) son más adecuadas para entornos aislados, es decir, entornos donde se conoce perfectamente el entorno. La evolución en este tipo de sistemas es sencilla, se modifica cada usuario para que cumpla con los nuevos requisitos. Pero cuando el número de usuarios es muy grande es necesario emplear una estrategia diferente. Se necesita organizar frameworks que permitan evolucionar, tanto por el lado del cliente como del servidor. Se necesita proponer un mecanismo explícito para la interoperabilidad de los sistemas que no poseen la misma API. Protocolos basados en RPC no son adecuados para este tipo de sistemas, ya que cambiar su interfaz resulta complicado.

Por esta razón, se intenta tomar como modelo a la Web. A primera vista se puede pensar que SOAP lo hace, ya que utiliza HTTP como medio de transporte. Pero Fielding argumenta que la Web funciona mejor cuando se utiliza en el estilo que lo hace REST.

Utilizar HTTP como medio de transporte para protocolos de aplicación a través de firewalls es una idea equivocada. Esto reduce la efectividad de tener un firewall. Lo cual aumenta las posibilidades de nuevos agujeros de seguridad.

Sin embargo, los partidarios de SOAP argumentan que gracias a la tecnología existente que permite a los diseñadores encapsular la complejidad del sistema, dando lugar a interfaces generadas automáticamente que permiten facilitar el diseño del sistema.

| | REST | SOAP |
|---|---|---|
| Características | Las operaciones se definen en los mensajes.<br><br>Una dirección única para cada instancia del proceso.<br><br>Cada objeto soporta las operaciones estándares definidas.<br><br>Componentes débilmente acoplados | Las operaciones son definidas como puertos WSDL.<br><br>Dirección única para todas las operaciones.<br><br>Múltiple instancias del proceso comparten la misma operación.<br><br>Componentes fuertemente acoplados |
| Ventajas | Bajo consumo de recursos.<br>Las instancias del proceso son creadas explícitamente.<br><br>El cliente no necesita información de enruta-miento a partir de la URI inicial.<br><br>Los clientes pueden tener una interfaz "listener" (escuchadora) genérica para las notificaciones.<br><br>Generalmente fácil de construir y adoptar | Fácil (generalmente) de utilizar.<br><br>La depuración es posible.<br>Las operaciones complejas pueden ser escondidas detrás de una fachada.<br><br>Envolver APIs existentes es sencillo Incrementa la privacidad.<br><br>Herramientas de desarrollo. |
| Desventajas | Gran número de objetos.<br><br>Manejar el espacio de nombres (URIs) puede ser engorroso.<br><br>La descripción sintáctica /semántica muy informal (orientada al usuario).<br><br>Pocas herramientas de desarrollo. | Los clientes necesitan saber las operaciones y su semántica antes del uso.<br><br>Los clientes necesitan puertos dedicados para diferentes tipos de notificaciones.<br><br>Las instancias del proceso son creadas implícitamente. |

# Descripción del proceso

El proyecto consta de varias iteraciones, teniendo cuatro fases por iteración.

# Iteraciones

## Primera iteración

Fase 1. Determinar objetivos

El desarrollo comenzó con una entrevista con los responsables del club de baloncesto Gran Canaria, junto con el tutor. El objetivo de esta reunión era tener un primer contacto con el mundo del baloncesto, y conocer a las personas con las que íbamos a tratar. También nos explicaron el funcionamiento del club y lo que ellos esperaban de la aplicación, que serían nuestros requisitos iniciales. Finalmente se acordó que lo primero sería desarrollar el sitio web, para una vez tener la aplicación lista subirla. Por lo que también obtuvimos los requisitos de la web, que en este punto no eran muy específicos.

Fase 2.Análisis

Se estudian los requisitos para ver los posibles riesgos, y proponer soluciones a algunos planteamientos que nos han realizado.

Fase 3. Diseño

Se hace el diseño de la plantilla del sitio web, teniendo en cuenta una serie de posiciones privilegiadas en la página, para por ejemplo las publicidades y los patrocinadores, además de intentar que el diseño sea moderno y limpio. Aunque en este caso es sencillo, ya que en gran medida el diseño nos lo han proporcionado los propios clientes.

Fase 4. Desarrollo y pruebas

Se modifica la plantilla Joomla para que su diseño se adapte a nuestras necesidades, modificamos colores y añadimos logos del club. En esta iteración las pruebas es reunirse nuevamente con los miembros del club, para enseñarles la platilla, y que nos marquen nuevos objetivos.

## Segunda iteración

Fase 1. Determinar objetivos

En la reunión nos muestran los cambios que hay que realizar en la plantilla, y nos indican una lista de componentes que es necesario desarrollar y añadir, como son Social, Media, Patrocinadores e Información, que comentaremos en profundidad más adelante.

Fase 2.Análisis

Se estudia las formas de desarrollar componentes para Joomla y si es factible realizar todas las peticiones.

Fase 3. Diseño

En esta iteración esta fase es bastante amplia, ya que es necesario pensar en cómo vamos a desarrollar los 4 componentes, además de las modificaciones de la plantilla.

Fase 4. Desarrollo y pruebas

Se desarrollan todos los componentes y se hacen los cambios pertinentes en el sitio web, además se hacen todo tipo de pruebas para comprobar que los componentes funcionan correctamente.

## Tercera iteración

Fase 1. Determinar objetivos

Nuevamente les mostramos los avances en la página y nos comentan que está casi terminada, únicamente hay que hacer unos pequeños ajustes y presentarla a la prensa, por lo que esta iteración es muy pequeña.

Fase 2.Análisis

En esta iteración la fase de análisis no es necesaria, ya que únicamente es arreglar pequeños errores.

Fase 3. Diseño

En esta iteración esta fase no es necesaria, ya que es ultimar los detalles para la página web.

Fase 4. Desarrollo y pruebas

El desarrollo consiste en arreglar pequeños errores y presentar la web a los periodistas.

## Cuarta iteración

Fase 1. Determinar objetivos

Al comienzo de esta iteración se hablan todos los detalles sobre el sistema de información, nos indican los datos importantes que les gustaría que se pudieran consultar y lo que se espera de la aplicación.

Fase 2.Análisis

Este es el análisis más largo de todo el desarrollo, ya que se nos plantean todos los requisitos del sistema, y es necesario estudiarlos minuciosamente para evitar futuros problemas.

Fase 3. Diseño

Diseñamos todo el sistema, este es una de las fases más críticas, ya que dispondremos de multitud de información interconectada, y la manera en que se pueda navegar por ella determinara la facilidad para usar el sistema.

Fase 4. Desarrollo y pruebas

Desarrollamos una primera versión funcional del sistema, aunque con muchas limitaciones, no se actualiza ni contiene muchos datos, únicamente es una demo para mostrar el funcionamiento.

## Quinta iteración

Fase 1. Determinar objetivos

Se muestra la demo y los responsables nos dan su opinión, nos indican ciertos cambios que les gustaría que hiciéramos y se plantea la necesidad de que la aplicación se actualice automáticamente, por lo que el objetivo principal de esta operación es conseguir descargar los datos actualizados y almacenarlos para más adelante en futuras iteraciones pasárselos a la aplicación.

En esta iteración tenemos las últimas reuniones con el club, ya que por problemas internos cambiaron de directiva y no están interesados en el sistema de información, por lo que a partir de este momento las reuniones serán únicamente con mi tutor.

Fase 2.Análisis

Se analiza el servicio web de la FEB para comprobar los datos que podemos extraer y cómo hacerlo, ya que en el servidor de la FEB no esta los datos organizados de la manera que nos interesa, por lo que tenemos que filtrarlos y ordenarlo desde la aplicación. Se estudia como almacenaremos toda esta información al descargarlo.

Fase 3. Diseño

Se diseñan las librerías que nos proporcionan todas las funciones necesarias para descargar los datos y tratar con ellos, además de poder actualizar la aplicación. Hay que tener en cuenta que toda la información tiene que estar interconectada, por lo que se requiere de un diseño bastante minucioso.

Fase 4. Desarrollo y pruebas

Se desarrollan las librerías mencionadas anteriormente, y realizamos todo tipo de pruebas para comprobar que funcionan correctamente.

## Sexta iteración

Fase 1. Determinar objetivos

El siguiente objetivo planteado es hacer los importadores para que la aplicación se actualice automáticamente, e incluir las librerías creadas anteriormente para que todo el proceso se haga desde el sistema de información.

Fase 2.Análisis

En esta iteración esta fase es relativamente pequeña, ya que disponemos de todos los conocimientos necesarios, y esta no conlleva excesivos riesgos.

Fase 3. Diseño

Se diseñan los importadores, y la manera en que interaccionara con el sistema de información y los ficheros XML descargados.

Fase 4. Desarrollo y pruebas

Se desarrollan los importadores, y llenamos de información todo el sistema, comprobando que todos los datos sean accesibles y estén bien ordenados. También se comprueba que funcione el actualizar.

## Séptima iteración

Fase 1. Determinar objetivos

Llegado este punto se plantea una interesante mejora, y es separar en dos unidades de negocio el actual sistema, una es la encargada de descargar los datos y enviársela a la otra que los mostrara. Para ello habrá que introducir interoperabilidad entre ellas, lo que facilita mucho añadir nuevas fuentes de información en el futuro, o arreglar cualquier tipo de incidencia con la fuente actual.

Fase 2.Análisis

Analizamos todos los riesgos que se nos plantean en la solución que hemos decidido implementar, y como solucionaremos dichos problemas.

Fase 3. Diseño

Diseñamos la segunda unidad de negocio, y como interaccionan las dos unidades.

Fase 4. Desarrollo y pruebas

Desarrollamos la segunda unidad de negocio, y realizamos todo tipo de pruebas, ya que llegado este punto todo debería funcionar perfectamente.

## Octava iteración

Fase 1. Determinar objetivos

En este caso los objetivos que se plantean es dejar la aplicación en una versión final funcional, corrigiendo todos los pequeños errores que puedan quedar.

Fase 2. Análisis

Revisamos toda la aplicación en busca de cambios a realizar, como por ejemplo filtrar categorías para que todas tengan el mismo formato, evitar errores de codificación con caracteres raros en el XML etc.

Fase 3. Diseño

Llegado este punto la fase de planificación es muy pequeña, ya que no hay que realizar grandes cambios

Fase 4. Desarrollo y pruebas

Finalmente desarrollamos los cambios y hacemos las pruebas pertinentes para comprobar que la aplicación ha quedado correctamente.

# Arquitectura del sistema

El sistema de información consta de cuatro subsistemas claramente diferenciados:

- Sitio web
- Unidad de negocio: Club
- Unidad de negocio: FEB
- Servicio Web de la FEB

Se conectan entre ellas de la manera que se muestra a continuación

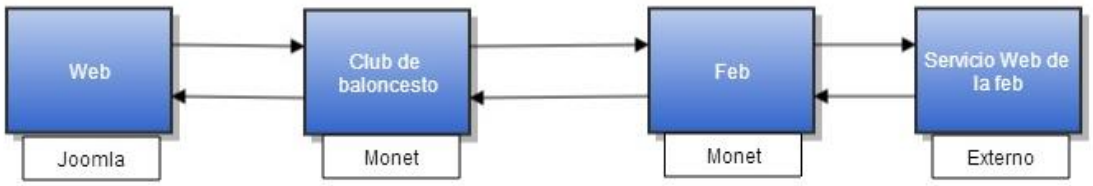



Figura 8: Arquitectura del sistema

El sistema de información se divide en dos unidades de negocio independientes que están interconectadas entre sí. Una de ellas denominado club de baloncesto que se encarga de mostrar todos los datos de los que disponemos y será accesible para el usuario mediante la página web del Club Baloncesto Gran Canaria.

La otra unidad de negocio, llamada Feb, se ocupa una vez recibida la solicitud correspondiente de comunicarse con el servidor Web de la FEB, descargar los datos, filtrarlos, ordenarlos... Para lo que hará uso de unas librerías que hemos desarrollado mediante Java.

Existe interoperabilidad entre ambas unidades, la unidad Club solicitara los datos a la Feb, que tras descargarlos y realizar todos los procesos internos necesarios le enviara un XML con la información estructurada.

Se ha decidido dividir en dos unidades el tratamiento de los datos para tener los procesos separados. Por lo que si en un futuro cambiara la fuente desde la que nos descargamos los datos o añadimos algunas nuevas no será necesario modificar la unidad Club de baloncesto, teniendo como único requisito que estas nuevas fuentes utilicen el mismo formato para enviar los datos; por lo que nos facilita mucho las tareas de mantenimiento o ampliar la aplicación en el futuro.

# Sitio web

Antes de empezar el proyecto se organizó una reunión con los responsables del club para conocernos, hablar de sus necesidades y expectativas. En dicha reunión nos comentaron que les gustaría renovar la página web del club, y nos encargaron dicha tarea.

En principio querían una web que fuera agradable a la vista, y se le diera especial importancia a la parte de Social y los patrocinadores. Dicha página se decidió hacer en Joomla ya que se necesitaba de urgencia y esta herramienta nos permitía construir la web de forma rápida.

La construcción del sitio web se dividió en las siguientes tareas:

- Diseño de plantilla
- Crear modulo Social
- Crear componente Media
- Crear modulo Patrocinadores
- Crear componente Información

## Diseño de plantilla

Esta parte es la más sencilla ya que el Joomla nos proporciona todas las herramientas necesarias y el club los colores y logos. Simplemente creamos la web ajustándonos a las especificaciones que nos indicaron, para ello nos descargamos una plantilla base, la que más se adaptaba a lo que necesitábamos y se realizaron en ella las modificaciones pertinentes.

A continuación se necesitaban una serie de extensiones

Figura 9: Sitio web terminado

## Social

Desde el club nos informaron que estaban muy interesados en tener todas redes sociales del club a disposición de los visitantes del sitio. Para ello se programó un módulo denominado Social para poder añadir los logos de las redes sociales con sus enlaces correspondientes en la esquina superior de la plantilla. En este caso no fue necesario realizar ningún tipo de interfaz para que ellos mismos subieran los logos, se hace todo desde el Joomla, y como no cambiara muy a menudo la responsabilidad recaería sobre el informático.

## Media

Este componente consta de cuatro partes:

- Twitter
- Facebook
- GrancaTV
- Fotos

Las extensiones tanto del Twitter como del Facebook ya estaban hechas, únicamente tuve que adaptarlas y acoplarlas a nuestro componente.

En la parte central del componente existen dos pestañas, una denominada GrancaTV, donde se verán videos, y para la que fue necesario crear una interfaz web para que el personal de Comunicaciones subiera los videos sin necesidad de contar con el informático, ya que se actualizarían con bastante frecuencia. Únicamente era necesario subir el video a YouTube o cualquier otra página, y pasarle el enlace a nuestra interfaz, a continuación internamente lo redimensionaba e incrustaba en la web.

Finalmente el último elemento del componente es la pestaña Fotos, que nos mostrara las fotos alojadas en un álbum Flickr, cuya URL especificaremos en la misma interfaz que utilizamos para GrancaTV.

Figura 10: Interfaz para subir videos y fotos a la web

Cabe destacar que no es necesario llenar los dos campos a la vez, si un campo está vacío o la URL no es correcta simplemente da un mensaje y no actualiza la web, por lo que seguiríamos viendo las fotos o videos anteriores.

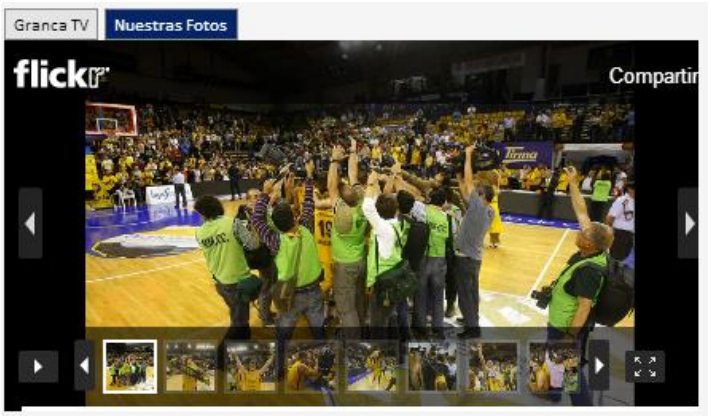



Figura 11: Captura de componente para ver las fotos

## Patrocinadores

El siguiente modulo necesario era muy específico, ya que ellos tenían en mente exactamente lo que querían, y así nos lo transmitieron. Necesitaban un componente que estuviera dividido en dos partes, la de la izquierda tendrá los tres logos en grande de sus principales patrocinadores con su enlace correspondiente al hacer click sobre ellos. En el lado derecho tenemos patrocinadores de menos importancia cuyos logos irán rotando cada pocos segundos, dependiendo del dinero aportado el logo dispondrá de link a la página o no.

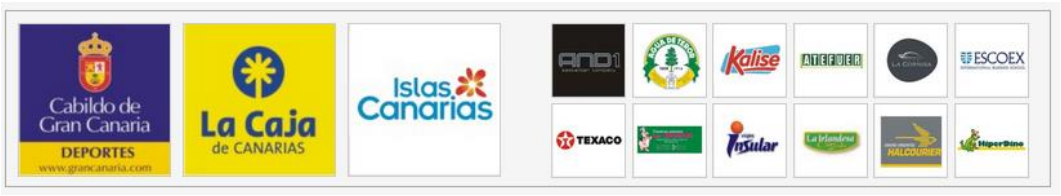



Figura 12: Captura del módulo de patrocinadores

Este es el componente más extenso, y consta de 6 pestañas, donde las tres primeras son bastante sencillas, ya que son estáticas, es decir no varían, estas son “Plantilla”,” Técnicos” y “Foto oficial”. mientras que las tres últimas, “Próximos partidos”,” Resultados” y “Clasificaciones” , hay que actualizarlas a diario o semanalmente. En principio este módulo era temporal, hasta que se implementara el sistema de información, que se actualizaría automáticamente y se incluyera en la web. Por lo que era necesario era necesario actualizarla a mano, tarea que sería responsabilidad del personal de prensa del club, pero para ello teníamos que buscar alguna manera sencilla de que la pudieran llevar a cabo de la manera más fácil posible para ellos, y que fuera rápidamente implementable por nosotros. Pero debía ser accesible desde cualquier lugar y en cualquier momento, por lo que llegamos a la conclusión de que la mejor manera era hacer uso del Google Docs como comentare a continuación.

Los responsables de actualizar esta información disponían de un enlace, que les llevaba a unas plantillas en Google Docs, por lo que era bastante sencillo y solo se necesitaba un ordenador con acceso a internet, aunque era un trabajo bastante rutinario y repetitivo, ya que tenían que ir actualizando todos los datos. Pero era factible puesto que tampoco eran una cantidad excesiva de datos. Por ejemplo la plantilla para actualizar clasificaciones la mostramos en la figura 13

| NombreEquipo | Ganados | Perdidos |
|---|---|---|
| FC Barcelona Regal | 28 | 5 |
| Real Madrid | 25 | 8 |
| Gran Canaria 2014 | 13 | 20 |

Figura 13: Plantilla de Google docs para actualizar clasificación

Como se puede ver simplemente se modificaba los partidos ganados y perdidos, y la aplicación calculaba la posición.

Una vez modificados los datos se actualizaba la página, ya al entrar en la web el componente estaba desarrollado de manera que solicitaba los datos al Google Docs. Aunque en caso de mucho tráfico ralentizaba un poco, pero como los datos solicitados eran muy pocos, apenas unas líneas de texto, el tiempo no era excesivo, unos pocos segundos.

Para llevar a cabo este componente se hizo uso de unas librerías descargadas de internet llamadas Google Spreadsheet, para PHP, que nos dotaban de todas las funciones necesarias para acceder al Google Docs y leer su contenido. Esta información la solicitaba el modulo a la hora de mostrar los datos.

El resultado final del módulo fue bastante satisfactorio, quedando como mostramos a continuación.

- Plantilla: Las fotos de todos los jugadores, con algunos datos personales, y en caso que tuviera Twitter su enlace correspondiente

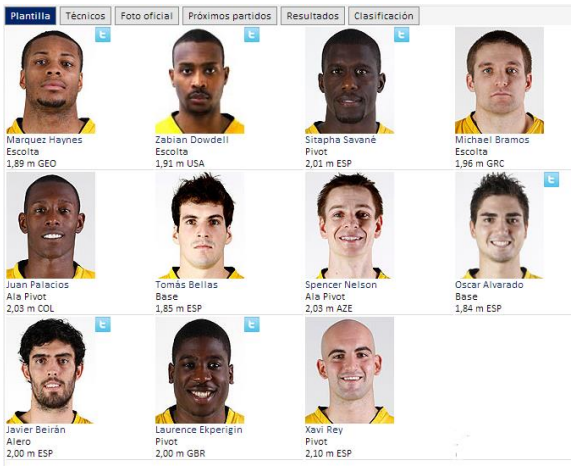



Figura 14: Componente mostrando jugadores

- Técnicos : Fotos de todos los técnicos del equipo

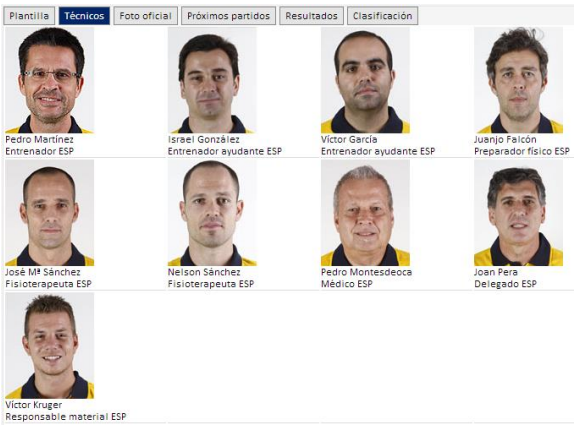



Figura 15: Componente mostrando técnicos

- Foto oficial: Foto de toda la plantilla con el nombre de sus jugadores

Figura 16: Componente mostrando plantilla

- Próximos partidos :Muestra cual son los próximos partidos y la hora a la que será

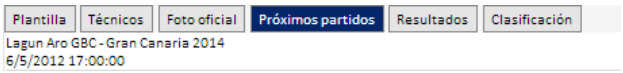



Figura 17: Componente mostrando próximos partidos

- Resultados: Muestra los resultados de los últimos partidos con un enlace a sus estadísticas en la web de la ACB

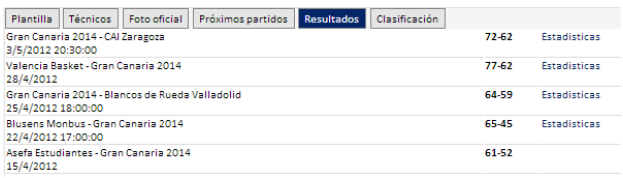
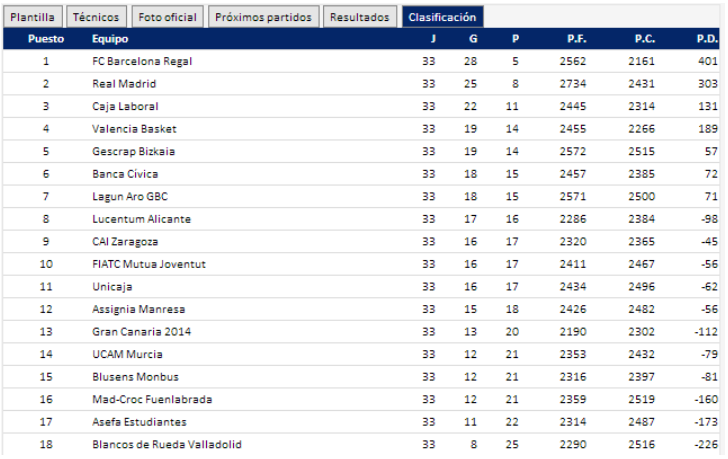



Figura 18: Componente mostrando resultados

- Clasificación: Muestra la clasificación actual



Plantilla | Técnicos | Foto oficial | Próximos partidos | Resultados | Clasificación

| Puesto | Equipo | J | G | P | P.F. | P.C. | P.D. |
|---|---|---|---|---|---|---|---|
| 1 | FC Barcelona Regal | 33 | 28 | 5 | 2562 | 2161 | 401 |
| 2 | Real Madrid | 33 | 25 | 8 | 2734 | 2431 | 303 |
| 3 | Caja Laboral | 33 | 22 | 11 | 2445 | 2314 | 131 |
| 4 | Valencia Basket | 33 | 19 | 14 | 2455 | 2266 | 189 |
| 5 | Gescrap Bizkaia | 33 | 19 | 14 | 2572 | 2515 | 57 |
| 6 | Banca Cívica | 33 | 18 | 15 | 2457 | 2385 | 72 |
| 7 | Lagun Aro GBC | 33 | 18 | 15 | 2571 | 2500 | 71 |
| 8 | Lucentum Alicante | 33 | 17 | 16 | 2286 | 2384 | -98 |
| 9 | CAI Zaragoza | 33 | 16 | 17 | 2320 | 2365 | -45 |
| 10 | FIATC Mutua Joventut | 33 | 16 | 17 | 2411 | 2467 | -56 |
| 11 | Unicaja | 33 | 16 | 17 | 2434 | 2496 | -62 |
| 12 | Assignia Manresa | 33 | 15 | 18 | 2426 | 2482 | -56 |
| 13 | Gran Canaria 2014 | 33 | 13 | 20 | 2190 | 2302 | -112 |
| 14 | UCAM Murcia | 33 | 12 | 21 | 2353 | 2432 | -79 |
| 15 | Blusens Monbus | 33 | 12 | 21 | 2316 | 2397 | -81 |
| 16 | Mad-Croc Fuenlabrada | 33 | 12 | 21 | 2359 | 2519 | -160 |
| 17 | Asefa Estudiantes | 33 | 11 | 22 | 2314 | 2487 | -173 |
| 18 | Blancos de Rueda Valladolid | 33 | 8 | 25 | 2290 | 2516 | -226 |

Figura 19: Componente mostrando clasificaciones

# Unidad de negocio “club”

## Roles

Esta unidad de negocio es el sistema de información en sí, contiene toda la información ordenada y a disposición del usuario, existen cuatro roles en esta unidad de negocio:

- Usuario: En este role declaramos los usuarios que podrán entrar en el escritorio principal y consultar toda la información.
- Administrador: En este role declaramos los usuarios que podrán entrar en el escritorio de administración, desde donde se podrá actualizar la aplicación
- RoleDescargarActualizacion: Este role se utiliza para la comunicación entre las dos unidades de negocio, en el caso que queramos actualizar los datos.
- RoleDescargarTodo: Este role se utiliza para la comunicación entre las dos unidades de negocio, en el caso de que queramos descargar todos los datos.

## Escritorios

Inicio

- Es el escritorio de los usuarios normales, desde donde podemos consultar toda la información, en el que hemos especificado todas las colecciones que queremos que sean accesibles, en nuestro caso podemos acceder desde aquí a las cinco colecciones principales de datos.

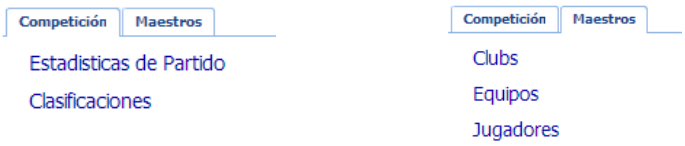



Figura 20: Captura de escritorio de usuario normal

InicioAdministrador

- Es el escritorio de los usuarios administrador para actualizar la información o añadir nuevos datos.

Figura 21: Captura escritorio de usuario administrador

## Importadores

Importadores: Existe un importador para cada colección, y se encargan de, una vez recibido el XML subir todas los datos al sistema.

- ImportadorDeClasificaciones
- ImportadorDeClubs
- ImportadorDeEquipos
- ImportadorDePartidos
- ImportadorDeJugadores

Todos los ficheros importadores tienen al principio una sección donde especificamos el formato de los XML de entrada, el nombre de las etiquetas y el tipo de dato de entrada.

A continuación recorremos todos los datos del fichero, creando un nodo cada vez que es necesario y rellenando todos los campos del formulario, las fotos, los links etc.

## Colecciones

La forma en que está organizada la información es mediante colecciones, en nuestro caso disponemos de cinco colecciones de contenedores, que serían las principales, y siete colecciones de formularios. Las cinco colecciones principales son

- Clasificaciones
- Clubs
- Equipos
- Jugadores
- Partidos

Cada una de ellas contiene un contenedor, y a la vez ese contenedor contiene una colección, a menos en el caso de jugadores y partidos que contiene dos, esto se explica más adelante.

## Contenedores

Disponemos de un contenedor por cada Colección, dichos contenedores almacenan una o varias colecciones y un formulario, con la información principal de la colección que contiene, por ejemplo el contenedor Clasificación dispone de una ficha clasificación, que almacena la temporada, la jornada, la categoría… y una colección de equipos que sería la clasificación en sí.

# Procesos

## Importación de datos

Cuando un usuario administrador quiere importar datos, y pulsa el botón correspondiente en el sistema se le dan tres opciones.

- Actualizar datos: Actualiza partidos, estadísticas y clasificaciones, y es bastante rápida.
- Actualizar todo: para cuando tenemos una nueva temporada y queremos importar los equipos o queremos actualizar lo que ya tenemos
- Cancelar: Termina la tarea

Si elegimos Cancelar, simplemente nos lleva al place Terminado que es "is-finaly" y como habíamos dicho anteriormente termina la tarea.

El motivo de crear un actualizar y un descargar todo es que el actualizar se supone que se hará cada poco tiempo y nos interesa que sea muy rápido, actualizando únicamente los valores cambiantes. Si queremos descargar todos los equipos, datos antiguos, jugadores con sus trayectorias etc. la aplicación tardara unas horas, por lo que nos interesa únicamente hacerlo al empezar una temporada para que nos descargue los nuevos equipos con sus categorías, o cuando queremos asegurarnos de que todos los datos están completos, ya que el actualizar no comprueba que los datos antiguos estén completos, simplemente actualiza a partir de la última jornada añadida.

En el caso de descargar todo o actualizar, comenzara el proceso de interoperabilidad que explicaremos en profundidad más adelante, mediante el cual recibirá unos ficheros XML. Dichos ficheros se los enviara a los importadores que les correspondan, donde se iniciara un proceso para subir toda la información.

El primer paso es enviar un Request de la unidad club de Baloncesto a la Feb

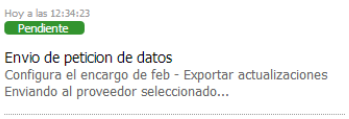



Figura 22: Aviso enviando petición de datos

La unidad de Club se queda en espera mientras la otra unidad descarga los datos, los filtra, almacena en XML, adjunta y envía junto con el Response.

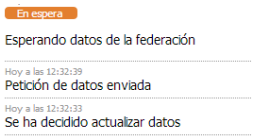



Figura 23: Aviso esperando datos

Una vez recibidos los datos, la unidad club se los envía a los importadores, que son los encargados de almacenar los datos en la aplicación, y al finalizar envían una señal de aviso a la unidad FEB de que todo ha salido correctamente. Al recibir dicha señal la unidad FEB termina, y Club nos informa que los datos están actualizados.

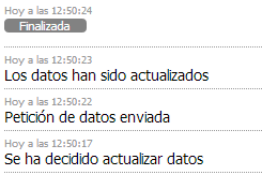



Figura 24: Aviso de datos actualizados

**Consultar información**

El sistema de información nos permite consultar gran cantidad de información, y permite muchas opciones para ello.

Equipos

Esta es la pantalla donde tenemos la información sobre equipos, como se puede apreciar, por cada equipo aparece su escudo, temporada y categoría.

Las opciones de filtrado son también estas, al igual que las de ordenar, también nos da la opción de buscar por la primera letra o escribir en el buscador el patrón de búsqueda.

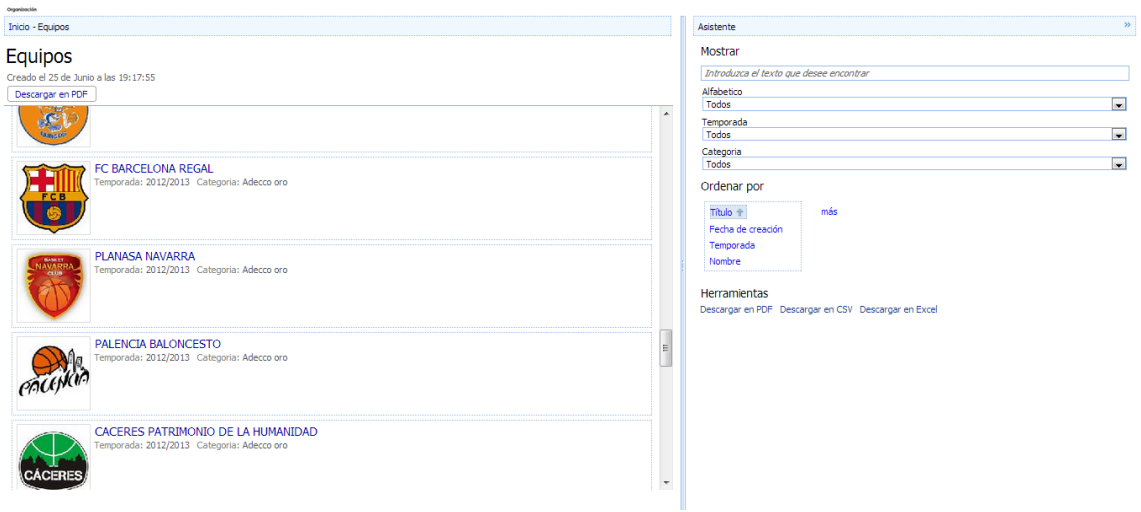



Figura 25: Pantalla de equipos

Una vez seleccionamos un equipo entramos en su ficha, desde la que podemos ver los datos del club, o seleccionar la pestaña Plantilla

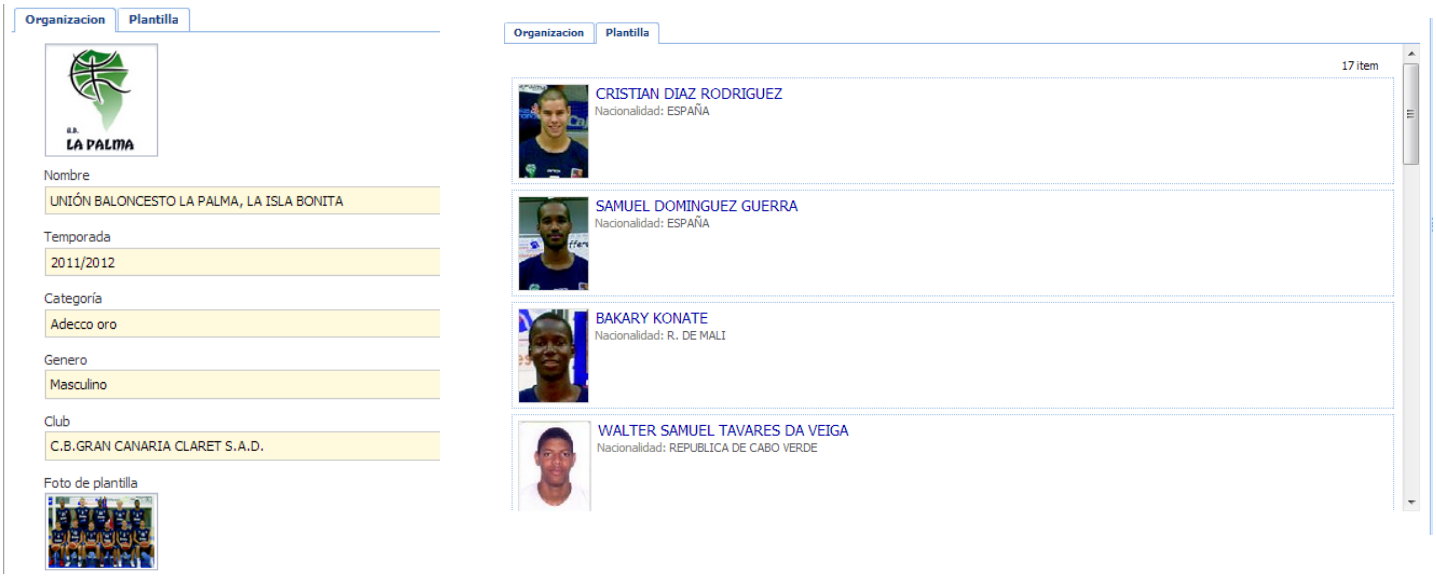



Figura 26: Pantalla con información de equipo y plantilla

Al entrar en la ficha del jugador correspondiente nos encontramos con la siguiente pantalla, donde vemos los datos del jugador y podemos consultar su trayectoria o estadísticas pasadas. A esta pantalla es posible llegar desde equipo o directamente desde el menú principal en la pestaña jugadores, pero esta segunda opción la explicaremos más adelante.

Figura 27: Pantalla información de jugador

Desde la pestaña trayectoria se puede observar los equipos en los que el jugador ha estado, indicando la temporada y categoría.

Figura 28: Trayectoria de jugador

Al seleccionar un equipo accederíamos a una pantalla con la información un poco más detallada, como se puede ver en la imagen. Si hacemos clic sobre el nombre del equipo iríamos a la ficha de dicho equipo, con las opciones mencionadas anteriormente.

Figura 29: Información detallada de equipo de trayectoria

Si seleccionamos la pestaña estadísticas aparecerían todas las temporadas para las que se dispone estadísticas de ese jugador, como vemos a continuación

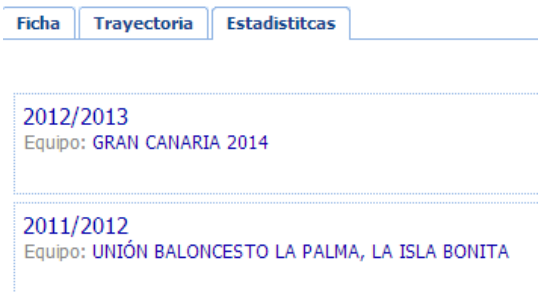



Figura 30: Pestaña estadísticas de jugador por temporadas

Y al seleccionar la que nos interesa nos aparecerán todas las estadísticas acumuladas de esa temporada

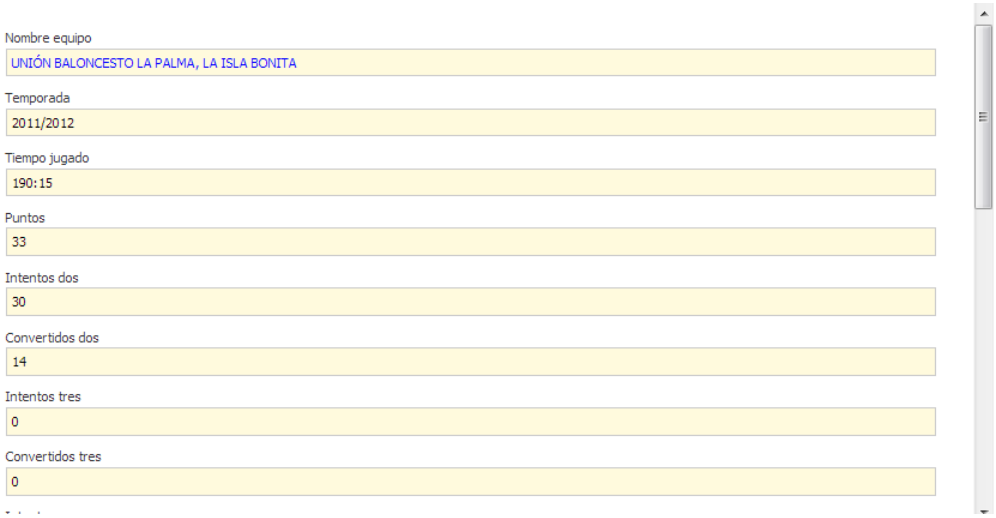



Figura 31: Estadísticas de jugador

## Jugadores

En esta pestaña nos sale la lista con todos los jugadores de los que disponemos en el sistema de información, que los podremos filtrar por varias opciones, al igual que los equipos.

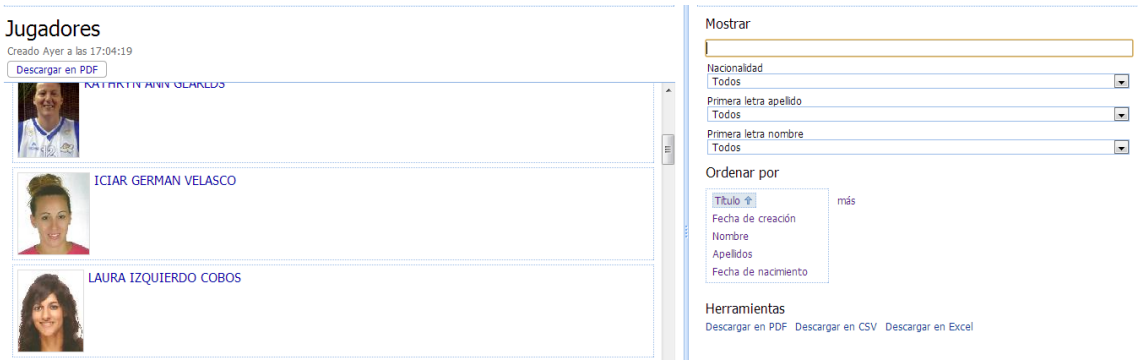



Figura 32: Pantalla de jugadores

Al seleccionar un jugador vamos a la pantalla ficha de jugador comentada anteriormente

## Estadísticas de partido

Aquí disponemos de todas las estadísticas de un partido, y de los jugadores que participaron en el, al igual que hemos mencionado antes, nos salen todos los partidos de que disponemos información, con sus filtros a la derecha

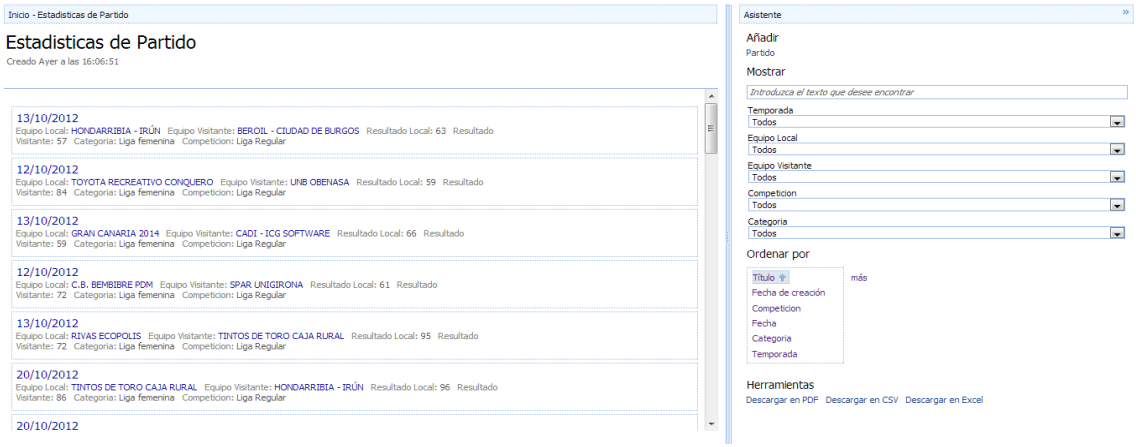



Figura 33: Pantalla estadísticas de partido

Nos muestra todos los datos del partido

Figura 34: Pantalla datos de partido

Sus resultados y las estadísticas por jugador

Figura 35: Pantalla resultados de partido

Clasificación

La pantalla principal de clasificación nos muestra todas las jornadas de las que disponemos información, que estarán divididas en categoría y temporada. Al entrar vemos la clasificación de esa jornada, y al seleccionar un equipo podemos ver sus estadísticas más detalladamente

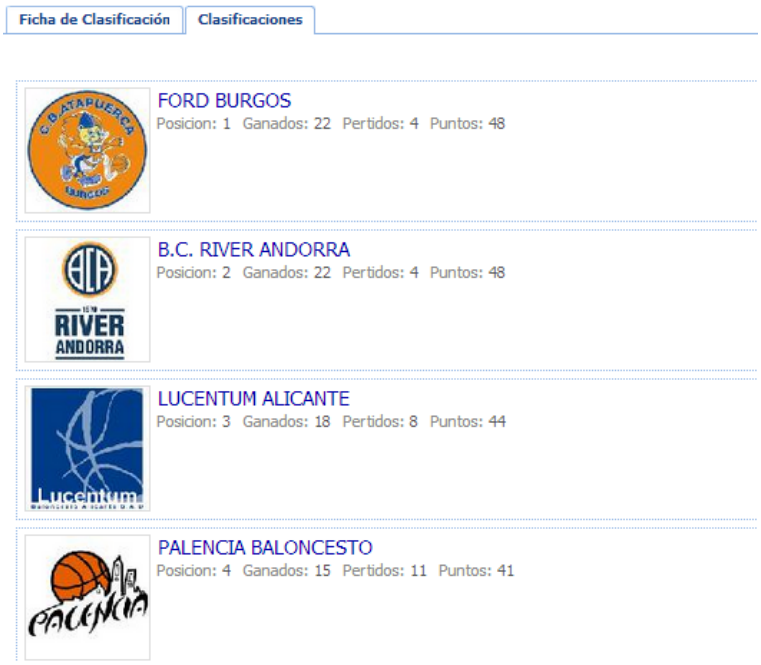



Figura 36: Pantalla clasificación

A continuación vemos la forma en que se relacionan las colecciones

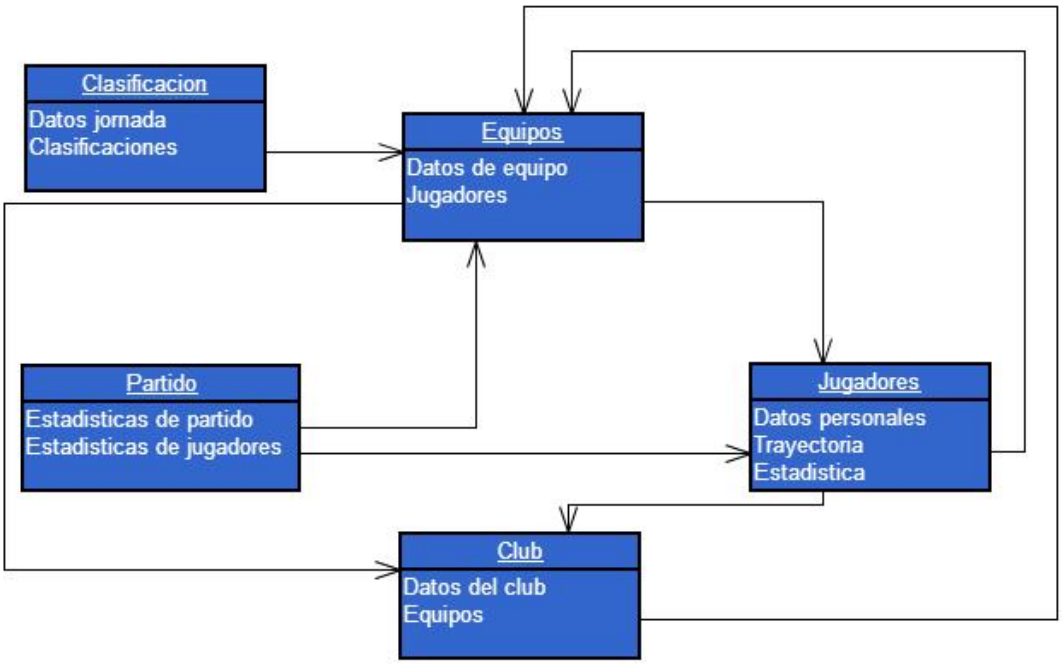



Figura 37: Relación de colecciones

A continuación mostramos los diagramas de clases de la unidad de negocio

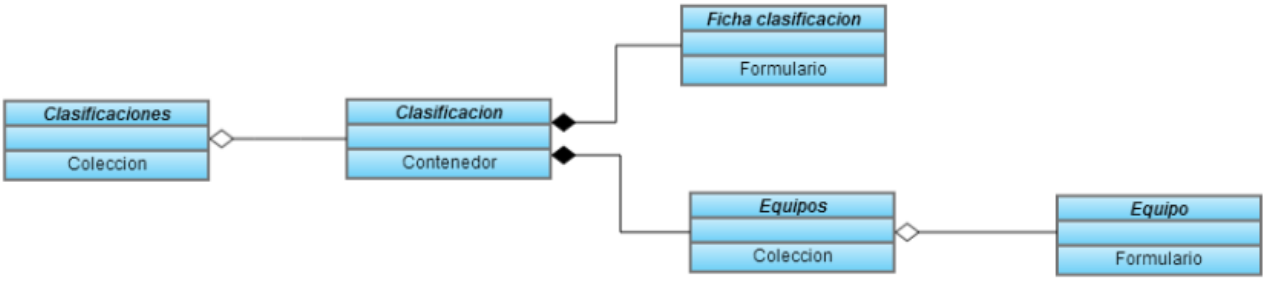



Figura 38 :Diagrama de clases de clasificaciones

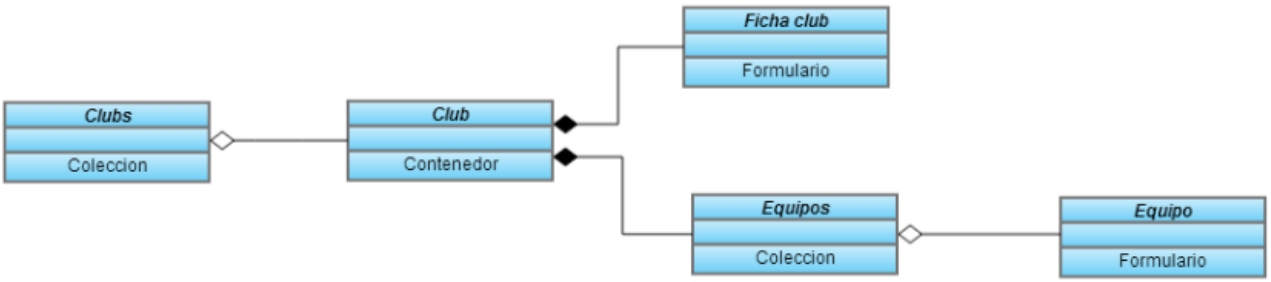



Figura 39 :Diagrama de clases de clubs

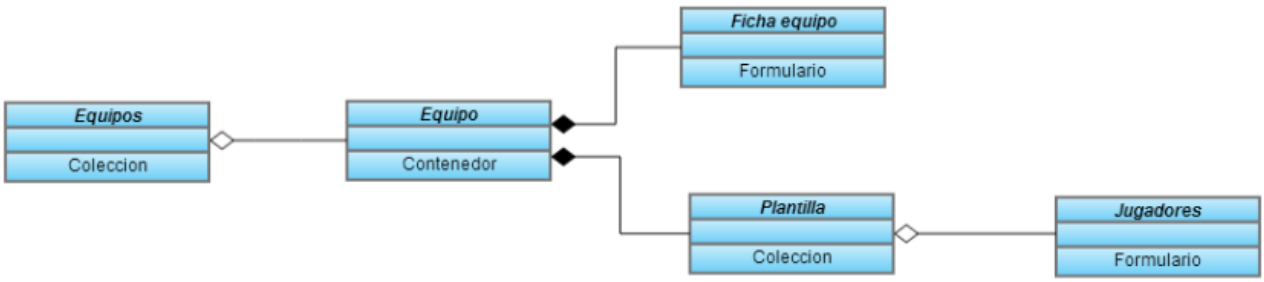



Figura 40 :Diagrama de clases de equipos

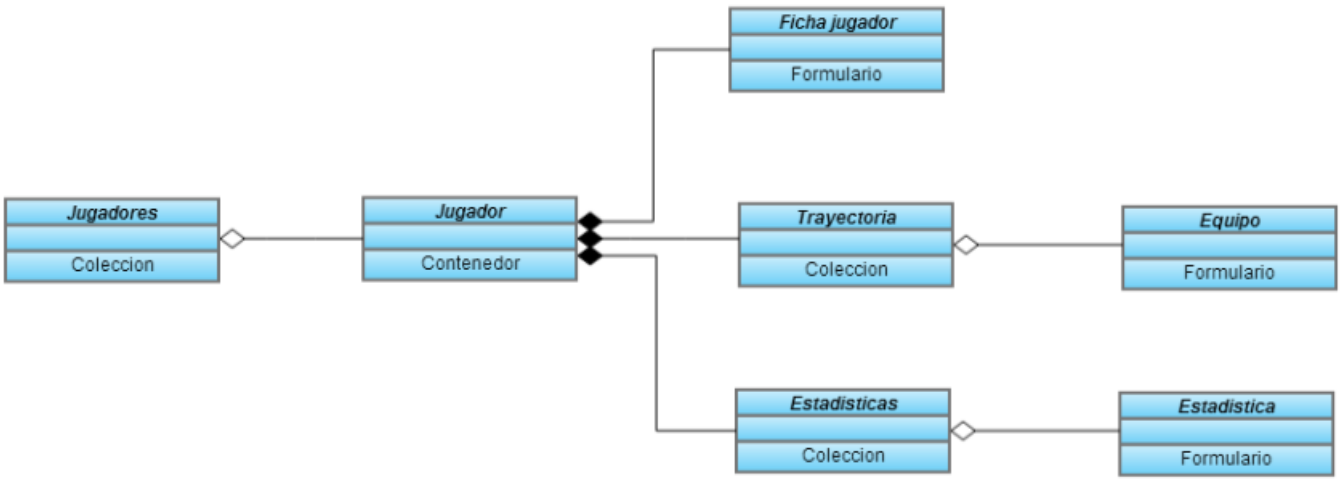



Figura 41 :Diagrama de clases de jugadores

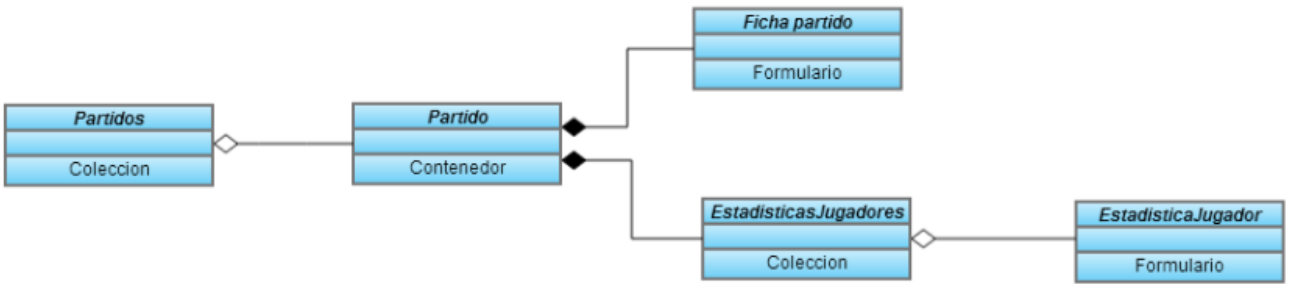



Figura 42 :Diagrama de clases de partidos

# Unidad de negocio “Feb”

Esta unidad de negocio se compone únicamente de dos Servicios, que son los que responden a las tareas de la otra unidad, se mantendrán a la espera para una vez recibida una petición activar los procesos correspondientes.

DescargarActualización

DescargarTodo

Ambos procesos son exactamente iguales, la única diferencia es la llamada a la función de las librerías creadas para trabajar con la FEB, en el caso de actualizar llamamos a una y en el de descargar todo a la otra. Estas librerías hacen todo el trabajo internamente, llamándose todas las funciones necesarias, y el resultado es una serie de ficheros XML que adjuntamos junto a la señal de respuesta.

En el caso del actualizar tiene una peculiaridad y es que leemos de un fichero el código de la última jornada añadida, así en el momento de descargar los datos sabemos a partir de qué jornada necesitamos descargar, y finalmente actualizamos la jornada en dicho fichero para futuras actualizaciones.

En este caso no existen roles, ya que a esta unidad solo sirve para conectarse a el servicio web de la federación y descargar los datos.

# Interoperabilidad entre ambas unidades

A continuación se va a explicar los procesos de intercambio de datos entre las dos unidades de negocio, tanto para descargar todos los datos, como para actualizar estadísticas y partidos.

Para ello se dispone de dos tipos de definiciones, las actividades que están alojadas en la unidad de negocio que solicita la información (en nuestro caso disponemos únicamente de una), y los servicios que están en la unidad que la envía (existen dos servicios en esta unidad).

Al acceder al escritorio de administrador, se nos ofertan tres opciones, que son Actualizar, Descargar todo o cancelar, estas opciones están indicadas dentro de la actividad, y se denominan líneas. Estas líneas simplemente muestran el texto de dicha opción, y según elijamos una u otra nos reenvían a la tarea correspondiente.

Esta tarea en el código se observa como una serie de “places”, que son los estados por los que va pasando el proceso, por lo que disponemos de un estado inicial que llama al

proveedor de servicios correspondiente mediante un “request”, como habíamos comentado anteriormente disponemos de dos, actualizar o descargar todo.

Y a continuación va a un estado de espera en el que se queda bloqueado hasta que recibe mediante un response el aviso desde el proveedor de servicio de que los datos están. Dichos datos los ha mandado el servidor adjuntos al response, mediante ficheros XML.

A continuación una vez desbloqueado sigue el flujo de ejecución enviando dichos ficheros a los importadores correspondientes, que se encargan de actualizar la información y una vez hecho esto finalizamos.

La estructura del servicio es similar a la de las actividad, ya que al igual que esta se consta de una serie de tareas o “places”, simplemente, al ser invocada mediante el request utiliza las librerías construidas para descargar los datos solicitados desde el servidor de la FEB. Una vez dichos datos están descargados, filtrados y ordenados genera los ficheros XML, y los adjuntara a un response, que a continuación enviara a la otra unidad de negocio para después finalizar.

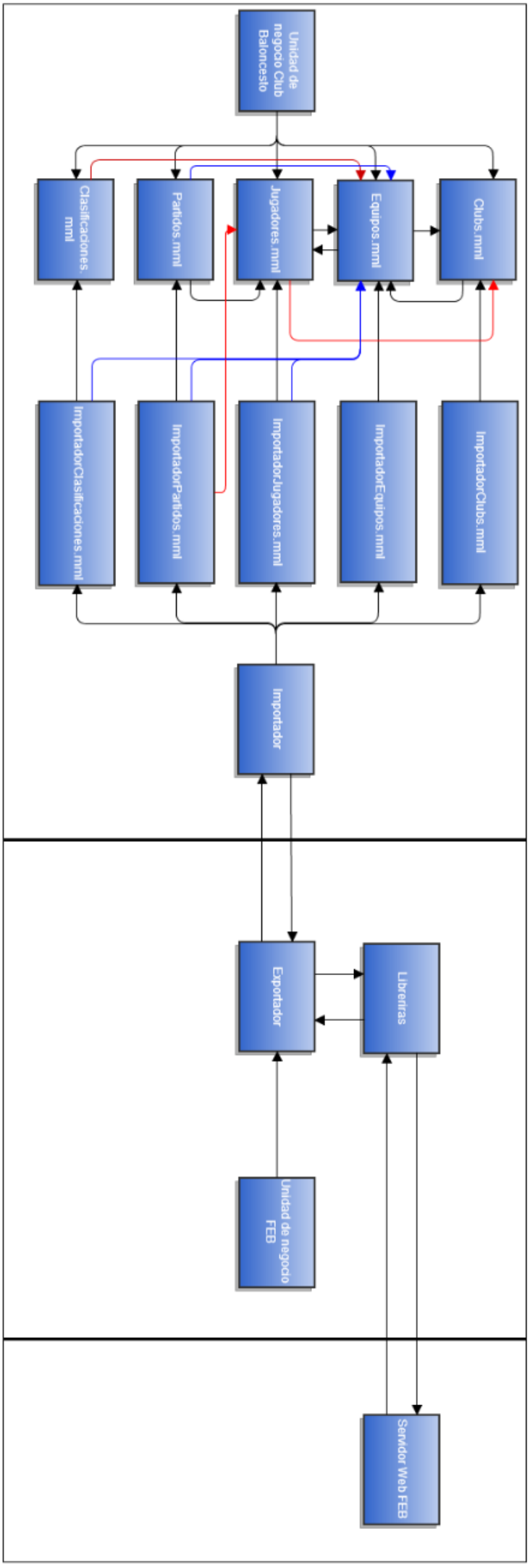



Figura 43: Diagrama de interoperabilidad entre unidades de negocio

# Funciones para descargar datos del servidor web de la FEB

Empezaremos comentando el funcionamiento de descarga y ordenado de datos, para ello disponemos de cinco paquetes:

- basketcollector
- basketcollector.filter
- basketcollector.pdfgenerator
- basketcollector.reader
- FebService

## Basketcollector

Dentro de basketcollector definimos todas las clases existentes en el proyecto, con sus constructores y las funciones necesarias para acceder a sus datos.

- Category
- ClasificationTeam
- Club
- Day
- Group
- Match
- Phase
- Player
- Season
- StatisticMatch
- StatisticMatchPlayer
- StatisticPlayer
- Team
- Trajectory

## Basket.reader

Dentro de basket.reader, disponemos de funciones para leer cada una de las clases declaradas anteriormente, su estructura básica es la misma, por lo que explicaremos solo una a continuación, aunque cada una tiene funciones específicas distintas de las otras.

En todas ellas existe una función pública readAllCateogry, ReadAllClasifications etc, a la que le pasamos los datos que necesitemos y nos devuelve la lista de elementos, por ejemplo en el caso de clasificación la cabecera de la función es:

```
public static ArrayList<ClasificationTeam> readAllClasifications(
ArrayList<Day> AllDays)
```

En este caso le tenemos que pasar un ArrayList<Day> que nos lo devuelve la function ReadAllDays, y asi vamos encadenando unas con otras.

Internamente esta función va recorriendo todas las Jornadas(Day) y pasándole su código a otra función llamada

```
private static EquipoClasificacionItem[] clasificacionXJornada(
int idJornada, java.lang.String key)
```

Esta función está en las librerías que me descargue, pero devuelve el dato en formato EquipoClasificacionItem[], por lo que necesitamos una función que nos transforme a nuestro formato ArrayList<ClasificationTeam>, que en este caso es:

```
private static ArrayList<ClasificationTeam> readClasificationTeams(
EquipoClasificacionItem[] EquipoClasificacionItems, String Date,
String season)
```

Finalmente esta función llama al constructor de Clasification Team

```
private static ClasificationTeam readClasificationTeam(
EquipoClasificacionItem item, String date, String season)
```

Como he mencionado anteriormente esta estructura básica se repite en todos los casos. Existen muchas más funciones auxiliares, pero básicamente se utilizan para ayudar a las mencionadas anteriormente.

Destacar únicamente que en jornadas (Day) disponemos de dos funciones, una de ellas es como la mencionada anteriormente, que descarga todas las jornadas, pero también disponemos de otra que es

```
public static ArrayList<Day> readAllDay(ArrayList<Group> AllGroups,
int idDay)
```

Esta función descarga todas las jornadas a partir de una concreta, y se utiliza para ir actualizando la aplicación, ya que guardaríamos la última jornada de la que disponemos información y cada cierto tiempo ejecutamos esta función, en caso que haya datos nuevos se los pasamos al resto de funciones (Clasification, StatisticMatch etc) para que se vayan actualizando.

### Basketcollector.pdfgenerator

La primera vez que ejecutamos el sistema de información se sube automáticamente todos los datos, para ello tienen que estar almacenados en ficheros XML, en este paquete tenemos las funciones que los generas, simplemente van leyendo los datos e imprimiendo las etiquetas necesarias

### Basketcollector.filter

Cabe destacar que los datos son externos a nosotros, y no siguen ningún tipo de estándar a la hora de escribirlos, por ejemplo las categorías de equipos masculino a veces está representado por una m, o más , masc, M, Masculino, masculino… y lo mismo ocurre con femenino o las abreviaturas de juveniles, cadetes etc, por lo que es necesario filtrar todo ese tipo de cosas, y eso es lo que hacemos desde aquí. También hay que sustituir una serie de caracteres que el XML no detecta bien como puede ser por ejemplo º que se sustituye por °

### FebService

Son unas librerías proporcionadas por la Federación de Baloncesto para poder acceder a través de SOAP a su servicio web y descargar todos los datos.

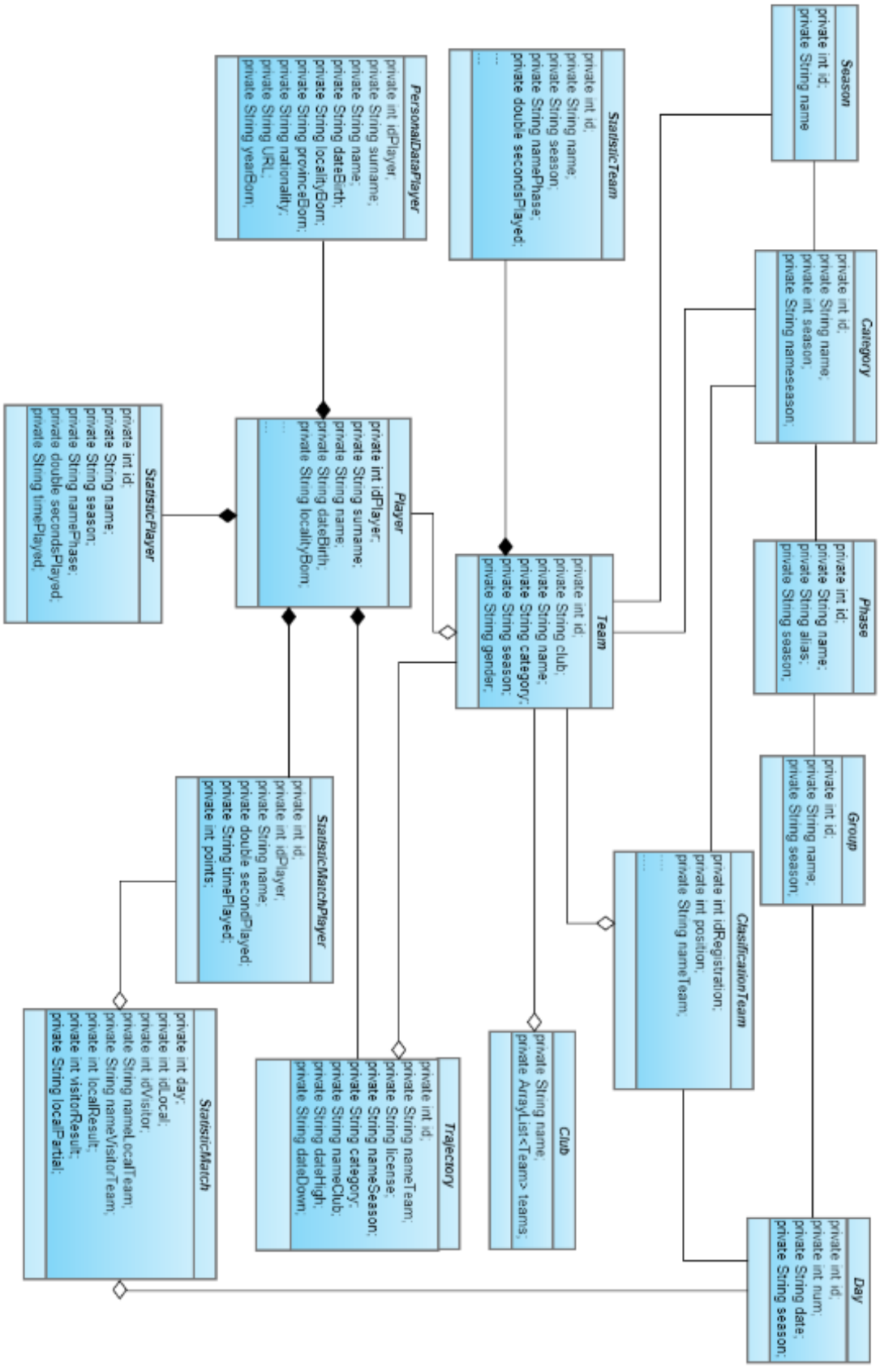



Figura 44: Diagrama de clases de las librerias

# Resultado

El resultado es el diseño y desarrollo de un sistema de información sobre baloncesto, junto con el sitio web del club de Baloncesto Gran Canaria 2014.

- Se ha construido el sitio web cumpliendo todos los requisitos especificados por el club, por lo que finalmente disponemos de una web, cuyo diseño ha sido el elegido por los responsables. Dicha web dispone de acceso a todas las redes sociales del club, además de tener un lugar especial para el módulo de patrocinadores que hemos desarrollado.

- El personal de prensa dispone de la interfaz propuesta para actualizar los videos y fotos de la web a su voluntad sin necesidad de contactar con informáticos.

- Hemos construido un SI, con gran cantidad de datos sobre jugadores, equipos, así como estadísticas, resultados y clasificaciones, con datos de varias temporadas.

- Además de disponer de todas las herramientas necesarias para que el sistema de información este siempre actualizado.

- Dicha aplicación, gracias a la potencia de Monet dispone de todo tipo de filtros para que sea más sencillo buscar información, además de posibilidad de ordenar los datos, según las opciones propuestas. Existe también la posibilidad de descargar cualquier información en Pdf, que lo convierte la aplicación automáticamente. Otra ventaja que nos proporciona haberla desarrollado en Monet es la sencillez y limpieza de su interfaz que hace que utilizarla sea totalmente intuitivo, por lo que toda la información es bastante accesible al usuario, y no es necesario que este pierda tiempo en tener que aprender cómo se utiliza.

- Se ha desarrollado el sistema de modo que este dividido en unidades de negocio que se encargan de dividir en dos unidades el tratamiento de los datos para separar los procesos y así si en un futuro cambiara la fuente desde la que nos descargamos los datos o añadimos algunas nuevas no será necesario modificar la unidad Club de baloncesto, teniendo como único requisito que estas nuevas fuentes utilicen el mismo formato para enviar los datos; por lo que nos facilita mucho ampliar la aplicación en el futuro.

- La herramienta es capaz de adaptarse a cada tipo de usuario y sus privilegios mostrando las opciones que cada perfil necesite. Existe una identificación segura para entrar a la herramienta, la cual se mostrará distinta de acuerdo al sistema de módulos y privilegios asociados al rol que se ha diseñado.

- Se encuentra alojado en un servidor Apache Tomcat. El sistema puede ser utilizado en diferentes máquinas con diferentes sistemas operativos. Solo es necesario que dispongan de un navegador web y conexión a Internet.

- Se han desarrollado librerías tanto para descargar todos los datos desde el servidor de la Feb, como para filtrarlos, ordenarlos, acceder a ellos y finalmente generar ficheros XML.

- A la otra unidad de negocio no se tiene acceso, ya que tanto para el usuario como para el administrador esa parte de la aplicación es totalmente transparente, no les interesa saber desde donde se descargan los datos o que procesos existen detrás, por lo que no tiene sentido crear un escritorio para la otra unidad.

- Por lo general se han cumplido todos los objetivos esperados.

# Conclusiones

La realización del proyecto fin de carrera, aunque ha sido dura me ha enseñado mucho, tanto a nivel técnico como personal, además de que me ha permitido poner en práctica los conocimientos adquiridos en muchas de las asignaturas cursadas a lo largo de toda la carreara.

A nivel técnico he aprendido que el análisis de un sistema de información es la base de un exitoso desarrollo, ya que la mayoría de las veces por las prisas o por falta de medios esta labor es relegada a un segundo plano pasándose muy ligeramente por ella.

También es importante la planificación, para evitar que una determinada tarea sea demasiado grande o empezar a programar sin tener todas las cosas claras, ya que esto provocara que tengas que trabajar en vano o repetir trabajo. Una de las partes más complicadas de la planificación es como dividir el trabajo en fases.

Otro aspecto importante es las tecnologías utilizadas, que actualmente están en auge y son muy utilizadas, lo que bajo mi punto de vista le da un valor añadido, han aumentado mis conocimientos en java, utilizando múltiples librerías y utilidades que no conocía, además de entrar en contacto con los Servicios Web y Soap, herramientas que tampoco había utilizado nunca.

También me sirvió para familiarizarme con Joomla, que da muchísimas facilidades a la hora de crear una web.

Finalmente Monet, una herramienta con una gran potencia que me ha facilitado mucho el trabajo, dando todo lo necesario para organizar y mostrar la información.

Como conclusiones a nivel personal comentar que este proyecto me ha servido como una gran experiencia, ya que se puede considerar una primera aproximación al mundo laboral, ya que durante la carrera hacemos muchas prácticas, no tan extensas ni que requieran de tanta planificación. Durante el proyecto tenemos la libertad y responsabilidad de tomar muchas decisiones, además de tener que realizar un análisis que permita poder tener un buen desarrollo. Me ha dado la oportunidad de tratar con clientes reales, realizando reuniones y analizando requisitos .Además en este caso tiene el valor añadido de ser una entidad como el club de baloncesto Gran Canaria, lo que también me ha permitido ver parte del funcionamiento interno de una organización de ese tamaño y características.

La memoria presentada incluye un análisis realizado con metodologías evolutivas, de una aplicación que en principio iba a ser implantada en un Club real, lo que hizo que el análisis fuese más laborioso y complicado pues tuvo que existir mucha comunicación, tanto presencial como por email, con los que sería los futuros usuarios de la aplicación. Una dificultad añadida ha sido el hecho de que las pautas que daban los responsables, una vez llevadas a cabo no gustaban a los directivos superiores, lo que conlleva un sobreesfuerzo, pero esto era inevitable, ya que los directivos de más alto nivel no se iban a involucrar de manera activa en el proyecto, únicamente querían ver el resultado y dar el visto bueno.

Pero todo ello me ha ayudado a entender cómo funciona el mundo real, la importancia de una buena planificación, y de tener el código ordenado, comentado y limpio por si es necesario hacer futuras modificaciones.

El resultado final ha sido muy satisfactorio, ya que como se ha comentado anteriormente se ha cumplido con los objetivos impuestos, por lo que finalmente se ha conseguido desarrollar una herramienta de fácil uso, en la que consultar cualquier dato relacionado con el baloncesto, que se mantendrá actualizada automáticamente.

Toda la información disponible en el sistema que podrá ser accedida por los usuarios:

- Datos de las clasificaciones de todas las jornadas y categorías.
- Estadísticas de todos los partidos.
- Datos, trayectoria y estadísticas por temporada te jugadores.
- Datos y jugadores de un equipo.
- Datos y equipos de un club.

Aunque no se dispone de toda la información para todas las categorías, disponemos de datos para:

- Liga Endesa
- EBA
- LEB Oro
- LEB Plata
- Junior masculino
- Cadete masculino
- Infantil masculino
- Mini masculino
- Liga femenina
- Liga femenina 2
- Junior femenina
- Cadete femenina
- Infantil femenina
- Mini femenino
- ACB
- SUB 20

# Trabajo futuro

Son varias las mejoras que se podrían realizar en el sistema. Con el diseño realizado, estas mejoras no supondrían grandes cambios en el código fuente. A continuación pasaremos a analizar algunas de ellas agrupándolas en dos grandes grupos:

Mejoras en la funcionalidad:

- Posibilidad de comparar ciertos datos como pueden ser estadísticas de equipos o partidos
- Posibilidad de ver estadísticas mediante graficas
- Traducir la aplicación a otros idiomas
- Posibilidad de insertar videos

Mejoras de la información:

- Insertar fotos de partidos
- Añadir nuevos sitios desde los que descargar datos para completar la información, es decir más unidades de negocio como la feb
- Calendario de futuros partidos

Otra posibilidad interesante seria modificar el sistema de información para que sirva para cualquier otro deporte.

# Bibliografía

Análisis y Diseño de Sistemas de Información

Editorial: Prentice-Hall

Autor: Kenneth Kendall & Julie Kendall

Desarrollo de los Sistemas de Información: Una Visión Práctica

Autor: Juan Bravo Carrasco

Editorial: Evolución

Ingeniería del software

Autor: Ian Sommerville

Editorial: ADDISON-WESLEY

JOOMLA! 1.6

Autor: Barrie M. Notrth

Editorial: Anaya-Multimedia

# Referencias


- Wikipedia (en línea).
- URL: http://es.wikipedia.org
- Google (en línea)
- URL: https://www.google.es/
- Joomla.org(en línea)
- URL: http://www.joomla.org/
- Feb (en línea)
- URL: http://www.feb.es/index.aspx
- W3Scools
- URL: http://www.w3schools.com/soap/
- Java
- URL: http://www.java.com/es/
- Eclipse
- URL: http://www.eclipse.org/
- XML
- URL: http://www.w3.org/XML/

# Anexo 1: Descripción servicio web de la FEB

La FEB oferta las siguientes funciones para hacer uso de su Web service

```
    /**
     * Devuelve todas las temporadas disponibles
     */
    public FebService.TemporadaItem[] temporadasDisponibles(java.lang.String
key) throws java.rmi.RemoteException;

    /**
     * Devuelve todas las categorias disponibles para la temporada
     * indicada
     */
    public FebService.CategoriaItem[] categoriasDisponiblesXTemporada(int
idTemporada, java.lang.String key) throws java.rmi.RemoteException;

    /**
     * Devuelve todas las temporadas disponibles para la categoria
     * indicada
     */
    public FebService.TemporadaItem[] temporadasDisponiblesXCategoria(int
codCategoria, java.lang.String key) throws java.rmi.RemoteException;

    /**
     * Devuelve la categoria a la que pertenece un equipo
     */
    public FebService.CategoriaItem categoriaXEquipoInscrito(int
idInscripcion, java.lang.String key) throws java.rmi.RemoteException;

    /**
     * Devuelve todas las fases disponibles para la categoría y temporada
     * indicada
     */
    public FebService.FaseItem[] fases(int codCategoria, int idTemporada,
java.lang.String key) throws java.rmi.RemoteException;

    /**
     * Devuelve todas los grupos disponibles para la fase, categoría
     * y temporada indicada
     */
    public FebService.GrupoItem[] grupos(int idFase, java.lang.String key)
throws java.rmi.RemoteException;

    /**
     * Devuelve las jornadas del grupo indicado
     */
    public FebService.JornadaItem[] jornadas(int idGrupo, java.lang.String
key) throws java.rmi.RemoteException;

    /**
     * Devuelve los partidos para la jornada indicada
     */
```

```
    public FebService.PartidoItem[] partidos(int idJornada, java.lang.String
key) throws java.rmi.RemoteException;

    /**
     * Devuelve toda la información del partido indicado
     */
    public FebService.InfoPartidoItem partido(int idPartido, java.lang.String
key) throws java.rmi.RemoteException;

    /**
     * Devuelve la clasificacion de la jornada indicada
     */
    public   FebService.EquipoClasificacionItem[]   clasificacionXJornada(int
idJornada, java.lang.String key) throws java.rmi.RemoteException;

    /**
     * Devuelve todas las estadisticas acumuladas de los jugadores
     * del equipo indicado
     */
    public   FebService.EstadisticaEquipo   estadisticasAcumuladasXEquipo(int
idInscripcion, java.lang.String key) throws java.rmi.RemoteException;

    /**
     * Devuelve la trayectoria personal del jugador del equipo indicado
     */
    public        FebService.Jugador        estadisticasAcumuladasXJugador(int
idInscripcion,     int     idJugador,     java.lang.String     key)     throws
java.rmi.RemoteException;

    /**
     * Devuelve los tipos de rankings
     */
    public  FebService.TipoRankingItem[]  tiposRanking(java.lang.String  key)
throws java.rmi.RemoteException;

    /**
     * Devuelve el ranking indicando el Grupo, Jornada y Tipo de Ranking
     */
    public  FebService.Rankings  ranking(int  idGrupo,  int  idJornada,  int
idTipoRanking, java.lang.String key) throws java.rmi.RemoteException;
```

# Anexo 2: Estructura de los ficheros XML

### *Clubs*

```
<Club>
	<schema>
		<Nombreclub>E.M.B. ALCAZAR</Nombreclub>
		<Equipos>
			<EquiposItem>
				<Nombreequipo>E.M.B. ALCAZAR D</Nombreequipo>
				<Temporada>2004/2005</Temporada>
				<Categoria>Infantil</Categoria>
			</EquiposItem>
			<EquiposItem>
				<Nombreequipo>EMB ALCAZAR A</Nombreequipo>
				<Temporada>2006/2007</Temporada>
				<Categoria>Alevin</Categoria>
			</EquiposItem>
			<EquiposItem>
				<Nombreequipo>E.M.B. ALCAZAR C</Nombreequipo>
				<Temporada>2007/2008</Temporada>
				<Categoria>Infantil</Categoria>
			</EquiposItem>
		</Equipos>
	</schema>
	<schema>
	.
	.
	</schema>
</Club>
```

### *Equipos*

```
<Equipos>
	<schema>
		<Id>608169</Id>
		<Temporada>2012/2013</Temporada>
		<Nombre>UCAM MURCIA CB</Nombre>
		<Categoria>Liga endesa</Categoria>
		<Genero>Masculino</Genero>
	</schema>
	<schema>
		<Id>608168</Id>
		<Temporada>2012/2013</Temporada>
		<Nombre>CAJASOL</Nombre>
		<Categoria>Liga endesa</Categoria>
		<Genero>Masculino</Genero>
	</schema>
	<schema>
	.
	.
	</schema>
</Equipos>
```

### *Jugadores*

```
<jugadores>
	<schema>
		<Id>1249531</Id>
		<NombreCompleto>AYOZE ALONSO RU&#205;Z</NombreCompleto>
		<Apellidos>ALONSO RU&#205;Z</Apellidos>
		<Nombre>AYOZE</Nombre>
		<FechaNac>04/04/1991</FechaNac>
		<Nacionalidad>ESPA&#209;A</Nacionalidad>
		<Localidad> </Localidad>
		<Provincia> </Provincia>
		<Url>
		http://competiciones.feb.es/estadisticas/foto.aspx?c=1249531
		</Url>
		<Edad>1991</Edad>
		<Trayectoria>
			<TrayectoriaItem>
				<NombreEquipo>C.B.UNI - PNEUS GUANCHA
				</NombreEquipo>
				<Licencia>Jugador/a</Licencia>
				<NombreTemporada>2003/2004</NombreTemporada>
				<Categoria>Infantil m</Categoria>
				<NombreClub>C.B. GUANCHA</NombreClub>
				<FechaAlta>10/10/2003</FechaAlta>
				<FechaBaja></FechaBaja>
			</TrayectoriaItem>
			<TrayectoriaItem>
				<NombreEquipo>XIEL DECO HOGAR</NombreEquipo>
				<Licencia>Entrenador/a</Licencia>
				<NombreTemporada>2008/2009</NombreTemporada>
				<Categoria>Infantil masculino</Categoria>
				<NombreClub>C.B. HERCULES DE ICOD</NombreClub>
				<FechaAlta>06/11/2008</FechaAlta>
				<FechaBaja></FechaBaja>
			</TrayectoriaItem>
		</Trayectoria>
		<Estadisticas>
			<EstadisticaJugadorItem>
				<Temporada>2012/2013</Temporada>
				<SegundosJugados>37752.0</SegundosJugados>
				<TiempoJugado>629:12</TiempoJugado>
				<Puntos>240</Puntos>
				<IntentosDos>117</IntentosDos>
				<ConvertidosDos>62</ConvertidosDos>
				<IntentosTres>62</IntentosTres>
				<ConvertidosTres>24</ConvertidosTres>
				<IntentosUno>63</IntentosUno>
				<ConvertidosUno>44</ConvertidosUno>
				<RebotesOf>13</RebotesOf>
				<RebotesDef>41</RebotesDef>
				<Asistencias>50</Asistencias>
				<BalonesRec>19</BalonesRec>
				<BalonesPer>53</BalonesPer>
				<TaponesFav>1</TaponesFav>
				<TaponesCon>6</TaponesCon>
				<Mates>0</Mates>
				<FaltasCom>60</FaltasCom>
				<FaltasRec>67</FaltasRec>
				<Valoracion>200</Valoracion>
```

```
                        <NumPartidos>28</NumPartidos>
                        <NombreEquipo>CC MERIDIANO SANTA CRUZ
                        </NombreEquipo>
                    </EstadisticaJugadorItem>
                    <EstadisticaJugadorItem>
                        <Temporada>2011/2012</Temporada>
                        <SegundosJugados>19088.0</SegundosJugados>
                        <TiempoJugado>318:08</TiempoJugado>
                        <Puntos>95</Puntos>
                        <IntentosDos>48</IntentosDos>
                        <ConvertidosDos>24</ConvertidosDos>
                        <IntentosTres>30</IntentosTres>
                        <ConvertidosTres>11</ConvertidosTres>
                        <IntentosUno>22</IntentosUno>
                        <ConvertidosUno>14</ConvertidosUno>
                        <RebotesOf>18</RebotesOf>
                        <RebotesDef>17</RebotesDef>
                        <Asistencias>24</Asistencias>
                        <BalonesRec>18</BalonesRec>
                        <BalonesPer>20</BalonesPer>
                        <TaponesFav>0</TaponesFav>
                        <TaponesCon>2</TaponesCon>
                        <Mates>0</Mates>
                        <FaltasCom>53</FaltasCom>
                        <FaltasRec>26</FaltasRec>
                        <Valoracion>72</Valoracion>
                        <NumPartidos>26</NumPartidos>
                        <NombreEquipo>CC MERIDIANO SANTA CRUZ
                        </NombreEquipo>
                    </EstadisticaJugadorItem>
                </Estadisticas>
        </schema>
        <schema>
        .
        .
        </schema>
        <schema>
        .
        .
        </schema>
</jugadores>
```

### *Partidos*

```
<partido>
        <schema>
                <Competicion>Liga Regular</Competicion>
                <Categoria>Eba</Categoria>
                <EquipoLocal>ASEFA ESTUDIANTES</EquipoLocal>
                <EquipoVisitante>CD COVIBAR RIVAS</EquipoVisitante>
                <ResultadoLocal>66</ResultadoLocal>
                <ResultadoVisitante>62</ResultadoVisitante>
                <ParcialesLocal>9|26|16|15</ParcialesLocal>
                <ParcialesVisitante>11|18|20|13</ParcialesVisitante>
                <ProrrogasLocal></ProrrogasLocal>
                <ProrrogasVisitante></ProrrogasVisitante>
                <Fecha>28/09/2012</Fecha>
                <Hora>21:00</Hora>
                <ArbitroPrincipal>VALLE IGLESIAS, JOSE M.</ArbitroPrincipal>
                <ArbitroAuxiliar>BARTOLOME GARCIA, MANUEL</ArbitroAuxiliar>
```

```
<Cancha>1018-ANTONIO MAGARIÑOS, PABELLON</Cancha>
<Localidad>1018-ANTONIO MAGARIÑOS, PABELLON</Localidad>
<Provincia>Madrid</Provincia>
<NombreTemporada>2012/2013</NombreTemporada>
<Estadisticasjugadoreslocal>
	<Estadisticaslocal>
		<IdJugador>1223583</IdJugador>
		<Nombre>ORTIZ BAUTISTA, ALEJANDRO</Nombre>
		<SegundosJugados>248.0</SegundosJugados>
		<TiempoJugado>04:08</TiempoJugado>
		<Puntos>0</Puntos>
		<IntentosDos>0</IntentosDos>
		<ConvertidosDos>0</ConvertidosDos>
		<IntentosTres>2</IntentosTres>
		<ConvertidosTres>0</ConvertidosTres>
		<IntentosUno>0</IntentosUno>
		<ConvertidosUno>0</ConvertidosUno>
		<RebotesOf>0</RebotesOf>
		<RebotesDef>0</RebotesDef>
		<Asistencias>0</Asistencias>
		<BalonesRec>0</BalonesRec>
		<TaponesFav>0</TaponesFav>
		<TaponesCon>0</TaponesCon>
		<Mates>0</Mates>
		<FaltasCom>0</FaltasCom>
		<FaltasRec>0</FaltasRec>
		<Valoracion>-2</Valoracion>
		<CincoInicial>No</CincoInicial>
		<NumeroDorsal>4</NumeroDorsal>
	</Estadisticaslocal>
	<Estadisticaslocal>
	.
	.
	</Estadisticaslocal>
</Estadisticasjugadoreslocal>
<EstadisticasjugadoresVisitante>
	<EstadisticasVisitante>
		<IdJugador>393613</IdJugador>
		<Nombre>GONZALEZ GALLEGO, JAVIER</Nombre>
		<SegundosJugados>1069.0</SegundosJugados>
		<TiempoJugado>17:49</TiempoJugado>
		<Puntos>5</Puntos>
		<IntentosDos>1</IntentosDos>
		<ConvertidosDos>1</ConvertidosDos>
		<IntentosTres>0</IntentosTres>
		<ConvertidosTres>0</ConvertidosTres>
		<IntentosUno>4</IntentosUno>
		<ConvertidosUno>3</ConvertidosUno>
		<RebotesOf>0</RebotesOf>
		<RebotesDef>3</RebotesDef>
		<Asistencias>0</Asistencias>
		<BalonesRec>3</BalonesRec>
		<TaponesFav>0</TaponesFav>
		<TaponesCon>0</TaponesCon>
		<Mates>0</Mates>
		<FaltasCom>3</FaltasCom>
		<FaltasRec>2</FaltasRec>
		<Valoracion>6</Valoracion>
		<CincoInicial>No</CincoInicial>
```

```
                        <NumeroDorsal>4</NumeroDorsal>
                    </EstadisticasVisitante>
                    <EstadisticasVisitante>
                    .
                    .
                    </EstadisticasVisitante>
              </EstadisticasjugadoresVisitante>
       </schema>
       <schema>
       .
       .
       </schema>
       <schema>
       .
       .
       </schema>
</partido>
```

### *Clasificaciones*

```
<clasificacion>
       <schema>
              <Competicion>Liga Regular</Competicion>
              <Jornadas>30/09/2012</Jornadas>
              <Temporada>2012/2013</Temporada>
              <Categoria>Eba</Categoria>
              <Mes>Septiembre</Mes>
              <Informaciones>
                    <PartInf>
                          <Posicion>1</Posicion>
                          <NombreEquipo>REAL CANOE N.C.</NombreEquipo>
                          <Jugados>1</Jugados>
                          <Ganados>1</Ganados>
                          <Perdidos>0</Perdidos>
                          <PtosFavor>77</PtosFavor>
                          <PtosContra>51</PtosContra>
                          <Puntos>2</Puntos>
                    </PartInf>
                    <PartInf>
                          <Posicion>2</Posicion>
                          <NombreEquipo>UCAM MURCIA</NombreEquipo>
                          <Jugados>1</Jugados>
                          <Ganados>1</Ganados>
                          <Perdidos>0</Perdidos>
                          <PtosFavor>82</PtosFavor>
                          <PtosContra>61</PtosContra>
                          <Puntos>2</Puntos>
                    </PartInf>
              </Informaciones>
       </schema>
       <schema>
       .
       .
       .
       </schema>
       <schema>
       .
       .
       .
```

```
        </schema>
    </clasificacion>
```