|  |
| --- |
| **GESTOR DE EVENTOS DEPORTIVOS** |
|  |
| OLIMPIADAS2MIL |
|  |
| Francisco Huertas  Gonzalo Ortíz  Alicia Pérez  Laura Reyero  Gabriela Ruíz  Rubén Tarancón |
|  |
| CURSO 2009/2010 |
|  |

Contenido

[Introducción 5](#_Toc261806277)

[Propósito 5](#_Toc261806278)

[Ámbito del sistema 5](#_Toc261806279)

[Definiciones, Acrónimos y Abreviaturas 5](#_Toc261806280)

[Referencias 7](#_Toc261806281)

[Descripción general 8](#_Toc261806282)

[Perspectiva del proyecto 8](#_Toc261806283)

[Funciones del sistema 11](#_Toc261806284)

[Características de los usuarios 11](#_Toc261806285)

[Restricciones 13](#_Toc261806286)

[Suposiciones y dependencias 13](#_Toc261806287)

[Requisitos futuros 14](#_Toc261806288)

[Requisitos específicos 15](#_Toc261806289)

[Planificación 17](#_Toc261806290)

[Casos de uso 20](#_Toc261806291)

[Diagrama de clases 30](#_Toc261806292)

[Observaciones 31](#_Toc261806293)

[Restricciones ocl 32](#_Toc261806294)

[Diagramas de estado 35](#_Toc261806295)

[Eventos 35](#_Toc261806296)

[Observaciones 35](#_Toc261806297)

[Entradas 35](#_Toc261806298)

[Observaciones 35](#_Toc261806299)

[Diagramas de actividad 36](#_Toc261806300)

[Eventos 36](#_Toc261806301)

[Observaciones 36](#_Toc261806302)

[Entradas 36](#_Toc261806303)

[Observaciones 36](#_Toc261806304)

[Diagramas de secuencia 37](#_Toc261806305)

[Obtener clasificación 37](#_Toc261806306)

[Observaciones 37](#_Toc261806307)

[Restricciones 37](#_Toc261806308)

[Consultar horarios personalizados 38](#_Toc261806309)

[Observaciones 38](#_Toc261806310)

[Restricciones 38](#_Toc261806311)

[Crear acta 39](#_Toc261806312)

[Observaciones 39](#_Toc261806313)

[Restricciones 39](#_Toc261806314)

[Reservar entradas 40](#_Toc261806315)

[Observaciones 40](#_Toc261806316)

[Restricciones 40](#_Toc261806317)

[Diagramas de componentes 41](#_Toc261806318)

[Diagramas de Componentes 41](#_Toc261806319)

[Base de datos 41](#_Toc261806320)

[Gestores 42](#_Toc261806321)

[Interfaces 43](#_Toc261806322)

[Modelo de datos 45](#_Toc261806323)

[Diagrama entidad-relación 45](#_Toc261806324)

[Modelo relacional 46](#_Toc261806325)

[Descripción de las tablas 46](#_Toc261806326)

[Funciones de la base de datos 48](#_Toc261806327)

[Patrones 55](#_Toc261806328)

[PATRÓN SINGLETON 55](#_Toc261806329)

[Interfaz de usuario 56](#_Toc261806330)

[Investigación sobre los componentes 57](#_Toc261806331)

[Propuesta para la interfaz 57](#_Toc261806332)

[Propuesta para la base de datos 58](#_Toc261806333)

[Propuesta para el blog 58](#_Toc261806334)

[Propuesta para el componente de pago 59](#_Toc261806335)

[Planificación estimada 60](#_Toc261806336)

[Propuesta de calidad 63](#_Toc261806337)

[Cuestionario CMMI 64](#_Toc261806338)

[Evolución del proyecto 69](#_Toc261806339)

# Introducción

## Propósito

Queremos ofrecer una herramienta que facilite la gestión de eventos deportivos de grandes magnitudes.

El propósito es integrar todo lo que conlleva la planificación de un acontecimiento de tan grandes magnitudes en una herramienta. Que ésta sea capaz de fusionar la organización deportiva del mismo como la interacción con el usuario a través de una página web.

## Ámbito del sistema

Queremos ofrecer una herramienta que facilite la gestión de eventos deportivos de grandes magnitudes. Para ello elaboramos una aplicación que administra los eventos desde la adjudicación del proyecto hasta la finalización del mismo.

Una vez adjudicado el proyecto, se solicita al ayuntamiento o ayuntamientos de la ciudad o ciudades organizadoras una serie de datos relativos a sus municipios que nuestra aplicación utilizará para la gestión de los eventos

La entidad organizadora del evento deportivo nos proporcionará información relativa a las distintas competiciones que se disputarán (características deportivas, equipamiento necesario para los eventos, número de participantes, número de árbitros…) así como un listado de participantes, árbitros etc. asignados a cada competición (ver el apartado [Modelo de Datos](#_Modelo_de_datos)).

Una vez recopilada toda esta información, nuestra aplicación se encarga de organizar en el tiempo y el espacio todos los eventos relativos a cada competición, atendiendo a las limitaciones impuestas por cada deporte. Con esto nos referimos a decidir la ubicación y hora de los eventos, asignar los árbitros en función de los equipos que vayan a participar, encargarse de las entradas etc.

Así mismo, nuestra aplicación actúa de portal de noticias y de visualización de las características del evento deportivo (calendario, clasificación, etc.) según el perfil al que pertenezca el usuario.

También incluye funcionalidad adicional como el visor de eventos, con el que un usuario (previo pago) podrá ver los eventos deportivos en directo online.

## Definiciones, Acrónimos y Abreviaturas

* Competición: En adelante competición es una categoría deportiva como podría ser el salto de altura o los cien metros lisos. Dada una competición puede saberse de manera unívoca sus normas.
* Evento: Un evento es cada una de las pruebas que componen una competición. Así, para salto de altura habrá varios eventos. Cada evento ha de transcurrir en un emplazamiento, un número de árbitros y de equipos participantes. Las características del emplazamiento así como el número de equipos participantes y árbitros viene impuesta por la competición a la que pertenezca el evento.
* Participante: Persona que interviene activamente en un evento, bien sea para competir (en ese caso hablaremos de un deportista) o bien para moderar el evento (en ese caso árbitro).
* Equipo: Conjunto de deportistas que se asocian para participar en una competición concreta. El equipo puede estar formado por un único deportista, y esto es así porque un deportista puede participar en varias competiciones, y aunque el deportista es el mismo, el equipo no se considerará igual. Todos los integrantes son de la misma nacionalidad y esto influirá a la hora de decidir el árbitro de un evento (no puede tener la misma nacionalidad que ninguno de los equipos que participen).
* Emplazamiento: Instalación en la que se van a desarrollar los eventos. Para que un evento se celebre en un emplazamiento éste tendrá que tener unas instalaciones acordes a las necesidades de cada competición.
* Asiento: Localidad que se encuentra en un emplazamiento susceptible de ser asociada a una entrada.
* Evento final: Ciertos tipos de competiciones de carácter eliminatorio (como todas las olímpicas) tendrán un evento final en la cual se decidirá el medallero.
* COI: Comité Olímpico Internacional. En principio la aplicación se plantea para organizar juegos olímpicos, aunque puede servir para organizar otros tipos de actividades deportivas. En este segundo caso, COI deja de referirse a este comité para referirse a la entidad gestora de la parte deportiva correspondiente.
* Nacionalidad: Aunque la legalidad internacional permita en ciertos casos a una persona tener múltiples nacionalidades, en nuestro sistema la nacionalidad se refiere a la federación nacional a la cual está federado el deportista o árbitro. Si la aplicación se usase para organizar competiciones no internacionales (como podría ser un torneo nacional), la nacionalidad de los participantes equivaldría al grupo o federación deportiva local en la que esté inscrito.
* Medallero: Cada competición tiene un medallero asignado que asignará la medalla de oro, plata y bronce al primer, segundo y tercer clasificado respectivamente.
* Clasificación: Representa un ranking de los distintos equipos junto con su puntuación en una competición en un momento determinado.
* Acta: Será un conjunto de datos introducidos por el árbitro de un evento donde refleja los puntos más importantes de éste. Está compuesta por el resultado del evento, las lesiones que se hayan producido y las sanciones aplicadas en dicho evento. También podrá introducir información adicional que sea relevante a nivel deportivo.
* Sanción: Penalización aplicada a un deportista en un evento concreto por una conducta antideportiva o antirreglamentaria. Se establece al introducir un acta. Puede tener repercusiones para el deportista sancionado a nivel de evento o incluso de competición (un deportista puede ser sancionado X partidos o ser expulsado de la competición).
* Lesión: Posible agravio sufrido por un deportista en un evento.
* Resultado: Será establecido al final de un evento y mostrará un listado de los participantes de un determinado evento junto a la puntuación obtenida en dicho evento. Esta puntuación puede ser el número de goles en fútbol, los segundos en carreras etc.
* Entrada: Billete asociado al asiento de un emplazamiento para uno o varios eventos. La duración de la validez de la entrada será desde el inicio del primer evento hasta el final del último. Cada entrada será de un tipo, y en función de ese tipo se establecerá el asiento y el precio.
* Entrada disponible: Aquella entrada que no está ni comprada ni reservada por ningún usuario.
* Entrada reservada: Aquella entrada para la que un usuario ha manifestado al sistema su intención de compra en un plazo establecido
* Entrada comprada: Aquella entrada para la que un usuario ha efectuado el pago correspondiente al importe de esa entrada.
* Tiempo de caducidad: Tiempo máximo que puede encontrarse una entrada en estado reservado sin ser comprada. Pasado este tiempo, la entrada pasará a estado disponible.
* Incidencia deportiva: Con este término nos referiremos a sanciones o lesiones que afecten a un participante.
* Núcleo de la aplicación: Se refiere a la parte de la aplicación representada por el diagrama de clases, que suponiendo un entorno que realice las correctas llamadas a las funciones, funcionará correctamente con la interfaz propuesta o dentro de cualquier otro sistema.
* Interfaz de la aplicación: Se refiere a la parte de la aplicación que interactuará con el usuario. Se trata de un módulo adyacente al núcleo que recibirá peticiones de los usuarios, las comunicará al núcleo y mostrará al usuario los resultados.

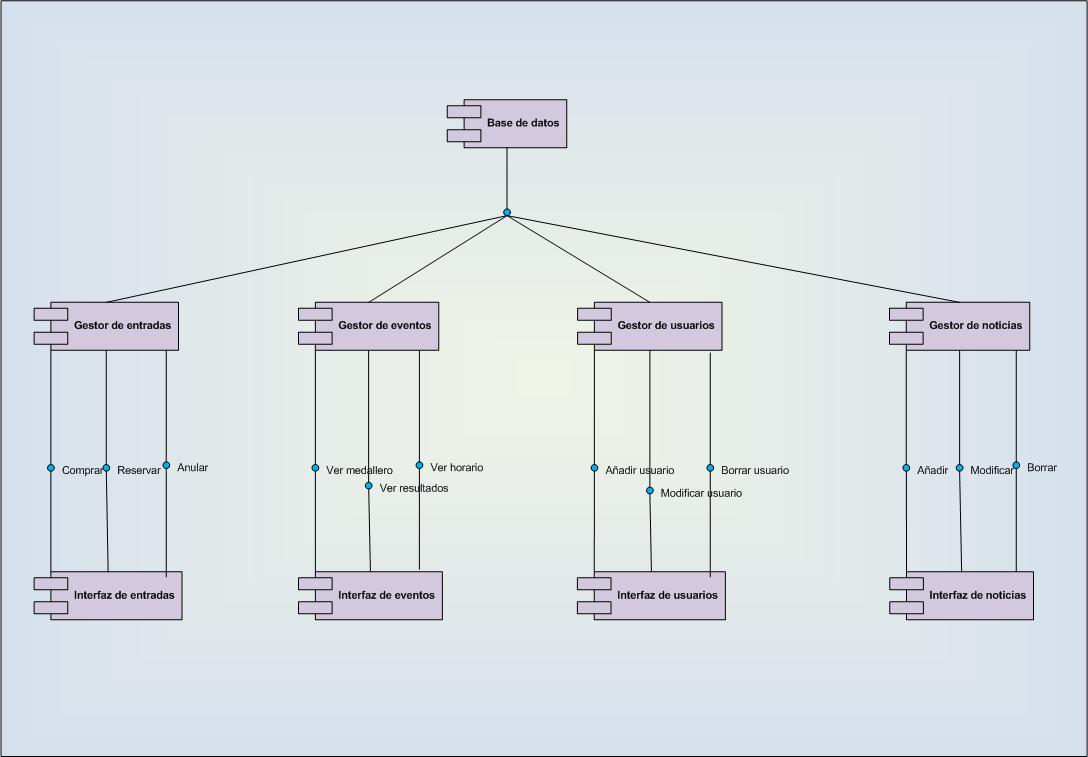
## Referencias

* <http://www.olimpiadasbeijing2008.com/>: Página oficial de las últimas olimpiadas celebradas. La hemos utilizado como referencia para la interfaz y para hacernos una idea de la organización y dimensiones del proyecto.

# Descripción general

## Perspectiva del proyecto

Diagramas de Componentes



###### Base de datos

* La base de datos de nuestra aplicación constará de varias tablas para los diferentes usos del sistema. Habrá tablas para el registro de los usuarios, para la venta de entradas, equipos participantes, instalaciones, etc.
* Alguna de esas tablas nos la enviará el COI (como la tabla equipos y la tabla competiciones) y en nuestra aplicación solo se podrán utilizar, nunca modificar. Otras tablas nos la dará el ayuntamiento de la ciudad organizadora (tabla instalaciones) y al igual que antes, nuestra aplicación no modificara esa tabla.
* El resto de tablas son las que mediante el sistema de gestión de bases de datos, abajo explicado, y los gestores harán que nuestra aplicación tenga una funcionalidad completa.
* En el apartado relativo a modelo de datos se explicara más detalladamente el formato de cada una de las tablas

###### SISTEMA DE GESTION DE BASE DE DATOS

* Es el componente que nos ayuda a gestionar la base de datos de la aplicación, el espacio físico y nos proporciona unas interfaces de acceso primitivas. Es el puente entre los gestores y las tablas de la base de datos.
* La interfaz consta de métodos de acceso a tablas (select), de creación de tablas (create), de borrado de tablas (drop) , borrado de información de una tabla (delete), modificación de información contenida en tablas (update) y de las propias tablas (alter) , y por ultimo un método para insertar información en las bases de datos (insert).

###### GESTOR DE ENTRADAS

* Es el componente a través del cual vamos a gestionar las entradas de los espectadores a las distintas disciplinas olímpicas.
* Va a ser el que va a recibir las ordenes de los compradores a través de la interfaz de entradas y el que va a tener que utilizar el SGDB para acceder y modificar las tablas. Gracias a él, los datos que los usuarios puedan introducir, serán coherentes con las tablas de la base de datos relativas a las entradas.
* Las operaciones que realizara sobre las tablas de la base de datos son:
* Reservar entrada : añadirá una fila a la tabla de entradas reservadas
* Comprar entrada: añadirá una fila a la tabla de entradas vendidas si la entrada que se intenta comprar pertenece a la tabla entradas reservadas y en caso afirmativo borrará la entrada de la tabla entradas reservadas.
* Anular entrada : eliminará una fila de la tabla de entradas reservas

###### GESTOR DE USUARIOS

* Es el componente a través del cual vamos a gestionar los usuarios que van a poder disfrutar de las ventajas de esta aplicación.
* A través del formulario de registrarse de la interfaz el gestor va a recibir todos los datos necesarios para que el usuario sea registrado en la aplicación. A partir de ahí el gestor hará las operaciones necesarias para que en la tabla de usuario queden reflejados los cambios.
* Las operaciones que va a realizar el gestor son:
* Registrar un usuario: añadir una fila a la tabla de usuarios.
* Modificar usuario: modificar los datos en la fila de la tabla de usuarios.
* Eliminar usuario: eliminar la fila de la tabla de usuarios.
* Login: comprobar que los datos introducidos están en la tabla de usuarios.

###### GESTOR DE NOTICIAS, FOTOS

* Es el componente a través del cual vamos a gestionar las noticias y fotos que van aparecer en la herramienta.
* En la interfaz de noticias, el usuario podrá introducir el titular y el cuerpo de la noticia, o si lo desea, fotos. Las noticias y fotos se almacenaran en el servidor y el encargado será el gestor.
* Las operaciones que realiza son:
* Añadir noticia
* Modificar noticia
* Añadir foto
* Eliminar foto
* Poner noticia en ultima hora
* En la parte relativa a componentes software se dará más información de este gestor y de cómo se va a integrar en nuestra aplicación

###### GESTOR DE EVENTO

* Es sin duda, el gestor más importante de nuestra aplicación a la vez que el más sofisticado.
* Utilizará las tablas de la base de datos de deportes, equipos e instalaciones que le vendrán dadas por el COI y el ayuntamiento para realizar un organigrama de toda la competición (en el diagrama de actividad Busca emplazamiento se detalla). Gracias a él se construirá el calendario completo de todos los eventos que se van a realizar en las olimpiadas y les podrá horario e instalaciones. También gestionará la tabla de resultados que después de cada partido se rellenará gracias a los actas y gracias a ello podremos visualizar inmediatamente las clasificaciones y el medallero. Por último el gestor tendrá operaciones más simples, pero no por ello menos importantes como pueden ser: modificar la información de un evento o modificar las actas de un partido.

###### Interfaces

* Es el componente a través del cual el usuario va a poder acceder a todas las utilidades que ofrece el sistema, por lo tanto debe ser sencillo y muy intuitivo.
* Habrá una serie de interfaces, cada una la gestionará uno de los componentes arriba descritos.
* Más adelante se podrán ver con más detalle estas interfaces.

## Funciones del sistema

Nuestro sistema gestiona principalmente aspectos de un proyecto deportivo:

* Gestión de competiciones: en este aspecto podemos encontrar las siguientes funciones:
  + Consultas: un usuario puede consultar información relativa al medallero, clasificación, participantes, calendario de eventos.
* Gestión de eventos:
  + Consultas: un usuario puede consultar información relativa a la fecha y la hora de un evento que aún no se haya celebrado, y consultar el resultado e incidencias de un evento celebrado.
  + Administración: un perfil administrador puede gestionar en qué lugar se va a celebrar un evento y a qué hora.
* Gestión de entradas:
  + El sistema se encarga de la venta, reserva y anulación de reserva de las entradas para los distintos eventos que estén planificados.
* Gestión de la información:
  + Consultas: cualquier usuario puede consultar el blog de noticias, el mapa de la villa deportiva, puede visualizar eventos online tanto pasados como en tiempo real y puede recibir noticias en su correo o dispositivo móvil.
  + Administración: los perfiles administradores se encargarán de gestionar la información que se ofrece al usuario, tal como el mapa de la villa deportiva, las entradas del blog, etc.

## Características de los usuarios

##### USUARIO NO REGISTRADO

Otorgaremos este perfil a cualquier individuo que acceda a nuestra aplicación sin identificarse mediante una clave de usuario y una contraseña.

##### USUARIO REGISTRADO

Otorgaremos este perfil a cualquier individuo ajeno a la organización que acceda a nuestra aplicación identificándose mediante una clave de usuario y una contraseña.

##### ÁRBITRO

Otorgaremos este perfil a aquellos individuos identificados por la aplicación como árbitros.

##### COI

Otorgaremos este perfil a aquellos individuos identificados por la aplicación como miembros del Comité Olímpico Internacional.

##### AYUNTAMIENTO

Otorgaremos este perfil a aquellos individuos identificados por la aplicación como miembro del ayuntamiento de la localidad organizadora.

##### ADMINISTRACIÓN – GESTOR DE EVENTOS

Otorgaremos este perfil a aquellos usuarios con poderes de administración total en la aplicación.

Figura: Diagrama de herencia entre los perfiles

## Restricciones

A continuación se enumeran una serie de restricciones sobre el desarrollo de la aplicación:

* Todos los usuarios que deseen adquirir una entrada para un evento deben estar registrados en la aplicación. Los datos que deben introducir son los que se han considerado mínimos para una correcta identificación personal y se gestionarán conforme a la [Ley de protección de catos de carácter personal](http://www.mir.es/SGACAVT/derecho/lo/lo15-1999.html)

## Suposiciones y dependencias

A continuación se enumeran todas aquellas cosas que se hayan dado por supuestas en la aplicación:

* En el núcleo
  + Se ha puesto especial hincapié en que los métodos que relacionan eventos con emplazamientos y entradas cumplan las invariantes del sistema. Sin embargo se ha relajado el control de las invariantes que rigen la distribución de eventos, deportistas y jugadores delegando su control en la interfaz o, en su caso, en el usuario.
  + Los requisitos que una competición pueda imponer a los emplazamientos donde se desarrollen sus eventos son tan variados como las propias competiciones. Por eso se ha simplificado el problema limitando las restricciones a tener piscinas, tener pistas de atletismo y tener circuitos de velocidad. Por ejemplo, la competición de salto triple y la de lanzamiento de peso tendrían, en nuestro sistema, los mismos requisitos: una pista de atletismo.
  + Las lesiones y sanciones son meramente informativas. El sistema no impide que un jugador lesionado o sancionado compita en un evento.
    - En el caso de los jugadores, si un jugador lesionado ha de participar en un evento es algo que se delega en las federaciones correspondientes y que un jugador sancionado participe en un evento es algo que ha de regular el árbitro de mismo.
    - En el caso de los árbitros, el sistema no contempla que pueden ser sancionados pero sí contempla que se lesionen. El obligar a sustituir a un árbitro lesionado en los eventos en los que no pueda participar es una funcionalidad que se propone como futura.
* En la interfaz
  + La interfaz es un anexo al núcleo. Por ello no se ha diseñado con tanto detalle como el núcleo. Por ejemplo, no hay diagrama de clases para la interfaz, sino que se describe su funcionamiento en un nivel más alto y abstracto.
* En las bases de datos
  + Se presupone que las bases de datos ya vienen dadas por los distintos organismos: las competiciones, deportistas y árbitros así como los datos de cada uno de estos objetos viene dada por el COI o la organización deportiva correspondiente. La lista de emplazamientos con todos sus datos viene a su vez dada por la administración local.
  + Se presupone la corrección de estas bases de datos dadas tanto con respecto a nuestros modelos de bases de datos como a las invariantes de núcleo.

## Requisitos futuros

A continuación se lista una serie de ampliaciones y mejoras propuestas:

* Núcleo:
  + Control de los invariantes en la distribución de eventos, deportistas y jugadores. Esto no se ha realizado por requerir replantear desde un comienzo el método de inserción de estos datos. Si se añadiera podría enfocarse de varias maneras, entre las que se enumeran:
    - Suponiendo dada la distribución de eventos en el tiempo, limitar la distribución de los deportistas entre los equipos de tal manera que nunca un jugador pudiera verse en dos equipos que podrían acabar teniendo eventos que coincidieran en el tiempo. Este sistema tiene el inconveniente de que sería el COI quien tiene que amoldarse a la entidad local organizativa. Por ejemplo, un mal diseño evitaría que deportistas pudieran participar en las competiciones de cien y doscientos metros lisos, algo bastante común, pero permitir a otro formar parte del equipo de fútbol y de baloncesto, algo realmente infrecuente.
    - Suponiendo dada la distribución de los jugadores en los equipos nacionales y los emplazamientos, organizar los eventos de tal manera que cumplan las invariantes. Este sistema delega en el COI la viabilidad o no del sistema. Por ejemplo, si el COI permite que un jugador esté inscrito en todos los deportes, es muy probable que dado el limitado número de emplazamientos, la olimpiada tuviera que durar muchísimo tiempo para poder cumplir que los eventos en los que potencialmente pueda estar ese jugador no se solapen.
    - Clasificando las competiciones por categorías (atletismo velocidad, atletismo salto, fútbol, etc.) de tal manera que el COI se comprometa a que un deportista o un árbitro solo estará en competiciones de una misma categoría, de tal manera que el sistema garantice que dada una categoría, los eventos de las competiciones de estas categorías no se solapan. Este posiblemente fuera el mejor sistema, pero requeriría modificar más profundamente el diseño actual.
  + Gestión de incidencias deportivas: que un deportista lesionado o sancionado participe en un evento es algo que siempre habría que delegarse en el propio deportista y su federación. Sin embargo el sistema podría requerir al COI un sustituto para el participante lesionado o sancionado por los eventos que dure su incidencia si estuviera planificado que participase en eventos posteriores.
  + Mejorar el sistema de requisitos en los emplazamientos: El núcleo utiliza un campo para cada requisito que pueda imponer una competición a un emplazamiento. Para tres posibles restricciones como se tienen actualmente esto es aceptable, pero no si se éstas se extendiesen a las restricciones reales de tantas competiciones como tendría una olimpiada. Para que el sistema fuera escalable debería usar un diseño de requisitos bastante más complejo.
* Otros:
  + Eventos asignados automáticamente: Actualmente ha de ser un usuario el que asigne los eventos una fecha y un emplazamiento. Que esto fuera hecho automáticamente es algo que se quiso hacer desde un principio. Gracias a las funciones del núcleo se podría hacer una capa externa (al mismo nivel que la interfaz gráfica) que mediante algoritmos más o menos complejos realicen esta tarea automáticamente o al menos aconsejen al usuario dónde y cuándo asignar un evento. Sin embargo esto es algo tan dependiente de la implementación que en nuestro caso no tiene demasiado sentido hacer un diseño exhaustivo.
* Añadir al sistema una gestión del alojamiento, ya que teniendo la información relativa a los participantes de los eventos y suponiendo que al igual que los emplazamientos el ayuntamiento o los ayuntamientos organizadores nos facilitasen los datos relativos a los alojamientos de los municipios en cuestión y sus alrededores, podríamos calcular el número de habitaciones necesarias para alojar a los participantes (o incluso al personal organizador y de apoyo a los participantes como médicos, entrenadores etc) y la forma de que el alojamiento fuese lo más óptimo para cada participante en función de los eventos previstos en los que fueran a participar.
  + Si se añade esta funcionalidad no sería muy costoso añadir una gestión de transporte, suponiendo que el transporte como tal se delegaría en una empresa externa y que el sistema únicamente se encargaría de proporcionar las rutas más óptimas para desplazar a los participantes o al personal indicado desde su alojamiento hasta el emplazamiento del evento que corresponda. Sin la funcionalidad del alojamiento también se podría desarrollar, suponiendo que conocemos el alojamiento del personal (aunque su distribución no sea la más óptima)

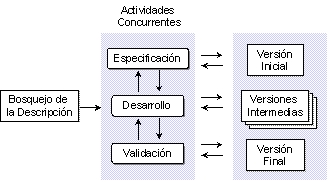
# Requisitos específicos

* Todos los deportistas que forman parte de un mismo equipo, tiene la misma nacionalidad.
* La nacionalidad de los árbitros que regulan un evento es distinta a la de los equipos que en él participan.
* Para poder reservar una entrada es necesario que el usuario esté registrado en el sistema.
* Para poder realizar la compra de una entrada, es necesario haberla reservado previamente.
* Los organizadores podrán determinar el tiempo máximo que una entrada puede estar reservada sin comprar. Pasado este tiempo, la entrada volverá a estar disponible.
* Se deben entregar funcionalidades de acuerdo al rol que tenga el usuario dentro del sistema.
* El software debe contar con un sistema de autenticación de usuarios a través de un identificador de usuario y una contraseña.
* Para poder visualizar los eventos online, es necesario que el usuario esté registrado.
* Cada comprador puede adquirir una o mas entradas para asistir a un mismo evento.
* Se distinguen tres tipos de entradas: normal, descuento y VIP; a las cuales se les aplicará un precio diferente.
* Cada competición tendrá determinadas las necesidades del emplazamiento donde poder realizarse.
* Cada competición determina el número de equipos y de árbitros necesarios para llevarla a cabo.
* Una competición está formada por varios eventos.
* Todos los emplazamientos adjuntan información sobre número y tipo de instalaciones que los componen.
* Los emplazamientos tienen un número variable de asientos.
* Cada asiento posee unas características que determinarán el precio de la misma.
* Un asiento tendrá asignadas tantas entradas como eventos se vayan a desarrollar en el emplazamiento al que pertenece.
* El horario también juega un papel importante en el evento, ya que los eventos no se pueden solapar en el tiempo; ya sea porque un equipo no pueda encontrarse en dos lugares diferentes al mismo tiempo; como el hecho de que en una misma pista no puedan tener lugar dos eventos simultáneamente.
* Para cada evento deberá buscarse un emplazamiento atendiendo a las necesidades que impone la competición y además tener en cuenta el tipo de evento que se desarrollara.
* Tras la finalización de un evento, el árbitro o árbitros encargados, deberán rellenar un acta con lo acontecido durante el mismo.
* El acta del evento contendrá los diferentes resultados que han obtenido los equipos participantes, así como las lesiones o sanciones que se hayan podido producir durante el evento.
* Toda la gestión se realiza a través de un portal web deportivo en el que se permitirá a los usuarios registrase para poder tener acceso a información particular y para la venta de entradas. Se podrá consultar la clasificación de las diferentes competiciones.
* A través del portal web, la organización puede publicar noticias y dar a conocer información sobre el estado del evento deportivo, la ciudad que lo alberga u otras noticias de interés general.
* Se podrá también visualizar en el portal un calendario con los próximos eventos que se celebrarán. Esta consulta, podrá ser más personalizada si el usuario que la realiza está registrado en ella.
* Si el usuario lo desea, podrá ver el evento a través de la página web, previo pago de la cantidad establecida por la organización.

# Planificación

Para el desarrollo de nuestra aplicación, hemos decidido utilizar un desarrollo evolutivo. La idea detrás de este modelo es el desarrollo de una implantación del sistema inicial, exponerla a los comentarios del usuario y refinarla hasta que se desarrolle el sistema adecuado.

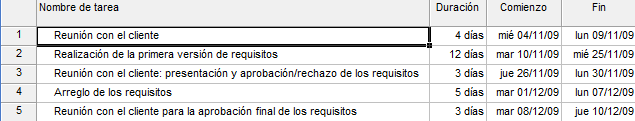
Una ventaja de este modelo es que se obtiene una rápida realimentación del usuario, ya que las actividades de especificación, desarrollo y pruebas se ejecutan en cada iteración.

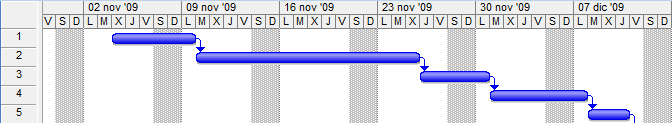


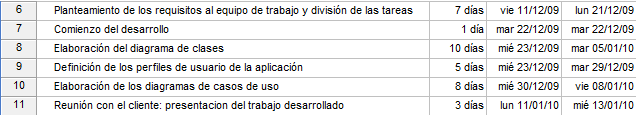
Nos hemos decantado por el Desarrollo Evolutivo Explorativo, cuyo objetivo es explorar con el usuario los requisitos hasta llegar a un sistema final. El desarrollo comienza con las partes que se tiene más claras. El sistema evoluciona conforme se añaden nuevas características propuestas por el usuario.

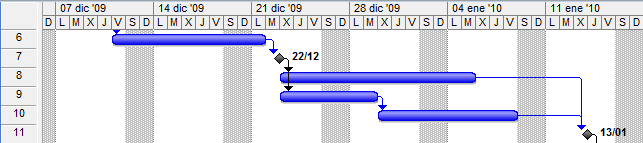
Para poder elaborar el sistema atendiendo a las necesidades del usuario, hemos planificado reuniones periódicas a lo largo del tiempo. En dichas reuniones, se expondrán los avances del proyecto y el usuario nos asesoraba sobre qué camino tomar.

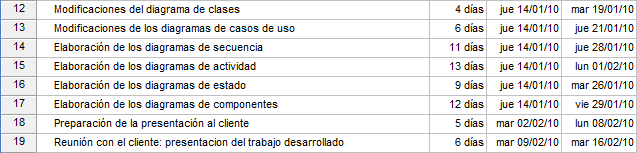
A continuación mostramos una propuesta de planificación para las tareas que hemos adoptado en la elaboración del proyecto.

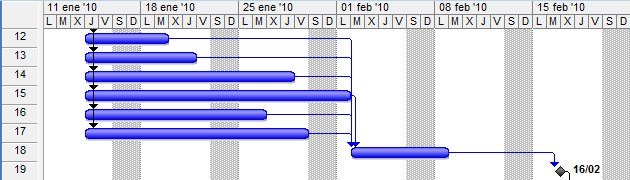


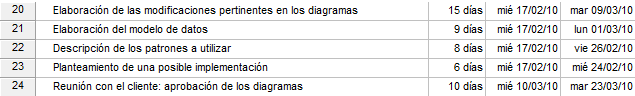


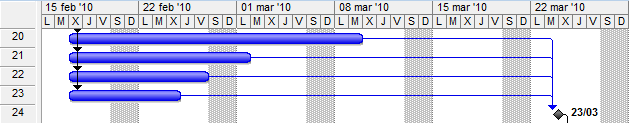




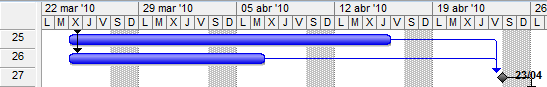


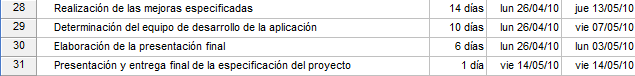


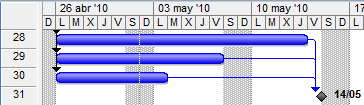












# Casos de uso

* A continuación se detallan los casos de uso correspondiente a cada rol de la aplicación. Toda la herencia entre usuarios se explica en la definición de roles:

###### Casos de uso de usuario no logueado



#### Descripción de los casos de uso

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Login |
| Actores | **Usuario** |
| Descripción | Cambiar su estado de usuario común a usuario logueado (o identificado), aumentando el número de operaciones disponibles |
| Restricciones | *Pre-condiciones:*   1. Usuario registrado   *Post-condiciones:*   1. Conversión de usuario común (o no identificado) a usuario logueado (o identificado) 2. Habilitar operaciones únicamente disponibles para usuarios identificados. |
| Objetivo | Identificación de un usuario ya registrado |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Registro |
| Actores | **Usuario** |
| Descripción | Registro de una nueva cuenta de usuario, que permitirá al mismo identificarse en la aplicación web. |
| Restricciones | *Post-condiciones:*   1. Usuario identificado. 2. Habilitar operaciones únicamente disponibles para usuarios identificados. |
| Objetivo | Creación de una nueva cuenta de usuario |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Consultar villa deportiva |
| Actores | **Usuario** |
| Descripción | Visualización de la distribución de la villa deportiva, imagen suministrada por el comité organizador (Ayuntamiento y COI) |
| Restricciones | No aplicable |
| Objetivo | Visualización de la distribución de la villa deportiva |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Consultar resultados |
| Actores | **Usuario** |
| Descripción | Visualización de los resultados de los eventos disputados, y actualizados en caso de que el acta correspondiente haya sido ampliada |
| Restricciones | *Pre-condiciones:*   1. Haber seleccionado una disciplina y evento ya disputado. 2. Existencia de un acta y/o adjuntos referida al evento seleccionado. |
| Objetivo | Visualización de los resultados de los eventos disputados |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Consultar titulares |
| Actores | **Usuario** |
| Descripción | Visualización de los titulares acerca de los eventos deportivos, suministrados por el comité organizador (Ayuntamiento y COI) |
| Restricciones | No aplicable |
| Objetivo | Visualización de los titulares acerca de los eventos deportivos |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Consultar fotos |
| Actores | **Usuario** |
| Descripción | Visualización de las fotos de los eventos deportivos, suministradas por el comité organizador (Ayuntamiento y COI) |
| Restricciones | No aplicable |
| Objetivo | Visualización de las fotos de los eventos deportivos |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Consultar disciplinas |
| Actores | **Usuario** |
| Descripción | Definición de las disciplinas existentes en los eventos, suministradas por el comité organizador (Ayuntamiento y COI) |
| Restricciones | No aplicable |
| Objetivo | Definición de las disciplinas existentes en los eventos |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Consultar última hora |
| Actores | **Usuario** |
| Descripción | Visualización de las noticias de última hora referidas a eventos deportivos, suministradas por el comité organizador (Ayuntamiento y COI) |
| Restricciones | No aplicable |
| Objetivo | Visualización de las noticias de última hora referidas a eventos deportivos |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Consultar calendario |
| Actores | **Usuario** |
| Descripción | Mostrar el calendario de eventos deportivos programados automáticamente por el cuadro de competiciones de la aplicación y con posibles actualizaciones debido a modificaciones en los diferentes eventos |
| Restricciones | No aplicable |
| Objetivo | Mostrar el calendario de eventos deportivos programados |

###### casos de uso de usuario logueado



#### Descripción de los casos de uso

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Borrarse |
| Actores | **Usuario logueado** |
| Descripción | Eliminar su cuenta como usuario de la aplicación |
| Restricciones | *Pre-condiciones:*   1. Usuario registrado e identificado 2. Sus pagos han sido realizados satisfactoriamente |
| Objetivo | Eliminar su cuenta |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Modificar datos |
| Actores | **Usuario logueado** |
| Descripción | Modificar los datos de su cuenta de usuario de la aplicación |
| Restricciones | *Pre-condiciones:*   1. Usuario registrado e identificado |
| Objetivo | Modificar los datos de su cuenta |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Anular reserva |
| Actores | **Usuario logueado** |
| Descripción | Cancelar la reserva de un evento seleccionado |
| Restricciones | *Pre-condiciones:*   1. La fecha del evento es posterior a la actual. 2. La fecha de reserva no ha superado el límite máximo tiempo de reserva.   *Post-condiciones:*   1. Aumentar el número de entradas disponibles para un evento, en relación a la anulación. |
| Objetivo | Cancelar la reserva de un evento |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Comprar entrada |
| Actores | **Usuario logueado** |
| Descripción | Compra de entradas para un evento determinado |
| Restricciones | *Pre-condiciones:*   1. La fecha del evento es posterior a la fecha actual. 2. La fecha de reserva no ha superado el límite máximo tiempo de reserva.   *Post-condiciones:*   1. Aumentar el número de entradas vendidas para el evento concreto. 2. Realizar el cobro mediante el servicio contratado |
| Objetivo | Comprar una reserva |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Reservar entrada |
| Actores | **Usuario logueado** |
| Descripción | Reserva de un evento concreto para su posible futura compra |
| Restricciones | *Pre-condiciones:*   1. Existe una o más entradas disponibles para el evento concreto.   *Post-condiciones:*   1. Disminuir el número de entradas disponibles para el evento concreto |
| Objetivo | Reserva de entrada |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Consultar entradas disponibles |
| Actores | Usuario logueado |
| Descripción | Mostrar las entradas disponibles, y sus características, de un evento seleccionado, de una disciplina específica. |
| Restricciones | *Pre-condiciones:*   1. Haber seleccionado una disciplina y un evento |
| Objetivo | Mostrar las entradas disponibles de un evento seleccionado |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Logout |
| Actores | **Usuario logueado** |
| Descripción | Cambiar su estado de usuario logueado a usuario común, restringiendo el número de operaciones permitidas |
| Restricciones | *Post-Condiciones:*   1. Convertir en usuario común (o no identificado) 2. Deshabilitar operaciones únicamente disponibles para usuarios identificados |
| Objetivo | Dejar de estar identificado en la aplicación web |

###### casos de uso de árbitro



#### Descripción de los casos de uso

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Crear acta |
| Actores | **Árbitro** |
| Descripción | Genera un acta perteneciente a un evento concreto, en el que ha participado el árbitro que la crea; y que contiene todos los hechos deportivos sucedidos durante el evento así como el resultado del mismo. |
| Restricciones | *Pre-condiciones:*   1. El evento al que se refiere el acta ha sido disputado con anterioridad 2. El árbitro, que genera el acta, ha participado en dicho evento   *Post-condiciones:*   1. Acta creada (no modificable) |
| Objetivo | Genera una nueva acta sobre un evento sucedido |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Consultar acta |
| Actores | **Árbitro** |
| Descripción | Muestra el contenido de un acta perteneciente a un evento concreto y sucedido con anterioridad. |
| Restricciones | *Pre-condiciones:*   1. Debe existir un acta relacionado con el evento concreto |
| Objetivo | Consulta el contenido de un acta sobre un evento sucedido |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Modificar acta |
| Actores | **Arbitro** |
| Descripción | Agrega una “nueva acta” o informe rectificando el acta original del evento. |
| Restricciones | *Pre-condiciones:*   1. El acta debe existir, previamente 2. El árbitro debe ser participante del evento o pertenecer al comité organizativo (COI)   *Post-condiciones:*   1. Adjunto (no modificables) |
| Objetivo | Agrega un documento adjunto a un acta ya existente |

###### casos de uso de AYUNTAMIENTO



#### Descripción de los casos de uso

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Introducir fotos |
| Actores | **Ayuntamiento** |
| Descripción | Agregar a la interfaz web fotos concernientes a los eventos deportivos. |
| Restricciones | *Pre-condiciones:*   1. Foto relativa a los eventos deportivos. 2. Foto no existente. |
| Objetivo | Aumentar el número de fotos existentes en la aplicación web. |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Introducir ultima hora |
| Actores | **Ayuntamiento** |
| Descripción | Agregar, en la interfaz web, noticias de última hora relacionadas con los eventos deportivos gestionados |
| Restricciones | *Pre-condiciones:*   1. Titular concerniente a los eventos deportivos 2. Titular no existente. |
| Objetivo | Nuevo titular de última hora en la aplicación |

|  |  |
| --- | --- |
| CasoDeUso | |
| Nombre | Introducir titulares |
| Actores | **Ayuntamiento** |
| Descripción | Agregar, en la interfaz web, las noticias relacionadas con los eventos deportivos gestionados |
| Restricciones | *Pre-condiciones:*   1. Noticia concerniente a los eventos deportivos 2. Noticia no existente |
| Objetivo | Nueva noticia en la aplicación |

###### casos de uso del coi



# Diagrama de clases



### Observaciones

Los equipos están formados por uno o más deportistas, todos de la misma nacionalidad. Y son los equipos los que participan en las diferentes competiciones.

El emplazamiento, lugar físico donde tendrá lugar el evento, posee información sobre sus instalaciones, para la posterior asignación. En función de esas instalaciones se podrá asignar el evento cuya competición tenga esos requerimientos.

Cada emplazamiento dispone de unos asientos para los espectadores en función de su aforo. Cada asiento puede tener asignada o no una entrada para asistir al evento. Dado que en un mismo emplazamiento, a lo largo del tiempo, se pueden celebrar varios eventos, el asiento tendrá asignado una entrada por cada evento que se celebra. Una vez el evento se ha celebrado las entradas se eliminan de nuestro sistema.

Desde la clase evento, se gestionan todos los detalles necesarios para la organización del mismo. Para cada competición se determinan los eventos que van a tener lugar para ultimar la misma. Desde esta misma clase se gestiona la búsqueda de un emplazamiento, que es donde tendrá lugar, atendiendo a las necesidades de la competición. Una vez encontrado un emplazamiento acorde a la competición, será necesario reservarlo, para la posterior venta de entradas.

Para que el evento tenga lugar, será también necesario determinar los equipos que competirán por alcanzar el triunfo y los árbitros que lo regularán.

Para poder asignar a los árbitros al evento, será necesario que su nacionalidad no coincida con la de ninguno de los equipos participantes.

El número de equipos y árbitros que tendrá el evento vendrá impuesto por su competición.

Tras la finalización del evento, el árbitro o árbitros serán los encargados de elaborar un acta. Dicho acta está formada por al menos dos Entrada Resultado, cada una de las cuales posee el resultado de cada equipo que ha participado en el evento. El acta también reúne las posibles lesiones que hayan podido sufrir alguno de los deportistas de los equipos, así como las sanciones que se le hayan podido aplicar.

Para cada emplazamiento en el que tendrá lugar un evento, se ponen a la venta una serie de entradas hasta completar el aforo del emplazamiento. Una persona identificada como comprador puede reservar una o varias entradas que se encuentren en estado disponible. Tras lo cual, el comprador puede decidir proceder con la compra de las mismas.

La organización será la encargada de determinar el tiempo de caducidad de la reserva de las entradas. Si pasado este tiempo, no se ha procedido a la compra de la misma, la entrada pasará a estar disponible para una futura reserva.

## Restricciones ocl

###### Funciones auxiliares

Context Evento::solaparHorarios(tiempo: Integer):Boolean

derive: self.horaIni < tiempo and self.horaFin > tiempo

Context Evento::solapaCon(otro:Evento):Boolean

derive: result = self.solaparHorarios(otro.horaIni)

Context Emplazamiento::dameMaximosSolapantes(eventos : Collection(Evento)) : Integer

derive:

result = eventos->asSet()->iterate(

e1 : Evento ;

max : Integer = 0 |

let

newMax : Integer = eventos->select(e2 |

e1.solapaCon(e2))

in

if(newMax > max)

max = newMax

else

max = max

endif

)

Context Emplazamiento::damePiscinasLibres(e1 : Evento) :: Integer

derive:

let eventos : Collection(Eventos) =

self.misEventos->select(e2 |

e2.solapaCon(e1)

and

e2.miCompeticion.requierePiscina)

in

result = numPiscinas –

dameMaximosSolapantes(eventos)

Context Emplazamiento::damePistasAtletismoLibres(e1 : Evento) :: Integer

derive:

let eventos : Collection(Eventos) =

self.misEventos->select(e2 |

e2.solapaCon(e1) and

e2.miCompeticion.requierePistaAtletismo

)

in

result = numPistasAtletismo - dameMaximosSolapantes(eventos)

Context Emplazamiento::dameCircuitosLibres(e1 : Evento) :: Integer

derive:

let eventos : Collection(Eventos) =

self.misEventos->select(e2 |

e2.solapaCon(e1) and

e2.miCompeticion.requiereCircuito

)

in

result = numCircuitos – dameMaximosSolapantes(eventos)

###### Invariantes

* El número de jugadores de un equipo estará comprendido entre un mínimo y un máximo

context Equipo inv:

self.misDeportistas->size() >= self.miCompeticion.minJugadoresPorEquipo and

self.misDeportistas->size() <= self.miCompeticion.maxJugadoresPorEquipo

* Todos los miembros de un equipo tienen que ser de la misma nacionalidad

Context Equipo inv:

Self.misDeportistas->forAll(d1,d2 | d1.nacionalidad = d2.nacionalidad)

* Si una competición tiene un único árbitro la nacionalidad del mismo debe ser distinta de la de los equipos participantes

Context Evento inv:

let nacionArbitro : Collection(String) = self.misArbitros.nacionalidad in

Self.miCompeticion.nArbitrosPorEvento = 1 implies

self.misEquipos->allInstances(e | e.misDeportistas.nacionalidad->intersection(nacionArbitro)->isEmpty())

* Si una competición tiene varios árbitros, éstos no pueden compartir nacionalidad entre ellos (esto implica que para cada equipo, a lo sumo solo hay un árbitro con la misma nacionalidad)

Context Evento inv:

Self.miCompeticion.tipoArbitraje = self.nArbitros > 1 implies

self.misArbitros->forAll(a1,a2 | a1.nacionalidad <> a2.nacionalidad)

* Los equipos de un evento tienen que estar inscritos en la competición a la que pertenece el evento

Context Evento inv:

Self.misEquipos->forAll(e| e.miCompeticion = self.miCompeticion)

* Los árbitros de un evento tienen que pertenecer a la competición a la que pertenece el evento

Context Evento inv:

Self.misArbitros->forAll(a| a.misCompeticiones->includes(self.miCompeticion))

* Un mismo equipo no puede estar en dos eventos que se solapen

Context Equipo inv:

Self.misEventos->forAll(e1, e2 | e1.solapaCon(e2) = false)

* Un mismo árbitro no puede estar en dos eventos que se solapen

Context Arbitros inv:

Self.misEventos->forAll(e1, e2 | e1.solapaCon(e2) = false)

* Un jugador no puede estar en dos equipos con eventos que se solapen

Context Deportista inv:

Self.misEquipos->forAll(eq1,eq2|eq1.misEventos->forAll(ev1 | eq2.misEventos->forAll(ev2 | ev1.solapaCon(ev2) = false)))

* El emplazamiento de todo evento cumple las restricciones de la competición

Context Evento inv:

Self.miCompeticion.necesitaPiscina implies

self.miEmplazamiento.numeroPiscina > 0

Context Evento inv:

Self.miCompeticion.necesitaPistaAtletismo implies

self.miEmplazamiento.numeroPistaAtletismo > 0

Context Evento inv:

Self.miCompeticion.necesitaCircuito implies

self.miEmplazamiento.numeroCircuito > 0

* Dado un emplazamiento no hay mas eventos asignados que necesiten recurso que recursos tiene el emplazamiento

Context evento inv:

self.necesitaPiscina implies self.miEmplazamiento.misEventos->select(e|

self.solapaCon(e) and

e.necesitaPiscina

)->size() <= self.miEmplazamiento.numeroPiscinas

Context evento inv:

self.necesitaPistaAtletismo implies self.miEmplazamiento.misEventos-

>select(e|

self.solapaCon(e) and

e.necesitaPistaAtletismo

)->size() <= self.miEmplazamiento.numeroPistaAtletismo

Context evento inv:

self.necesitaCircuito implies self.miEmplazamiento.misEventos->select(e|

self.solapaCon(e) and

e.necesitaCircuito

)->size() <= self.miEmplazamiento.numeroCircuito

Context Entrada

inv entradaTieneEvento:

self.asiento.miEmplazamiento.misEventos->exist(e|

e.solapaCon(self.instante))

inv entradaDisponibleSiYSoloSiNoTieneComprador

self.estado = EstadoReserva::disponible implies self.miComprador->isEmpty()

and

self.miComprador->isEmpty() implies self.estado = EstadoReserva::disponible

* Ningún asiento tiene dos reservas en el mismo momento

Context Asiento inv:

self.misReservas->forAll(r1,r2|r1 <> r2 and r1.instante <>r2.instante)

* Para cada evento hay tantos equipos participantes como requiere la competición

Context Evento inv:

self.misEquipos->size() = self.miCompeticion.nEquiposPorEvento

* Para cada evento hay tantos árbitros como requiere la competición

Context Evento inv:

self.misArbitros->size() = self.miCompeticion.nArbitrosPorEvento

* Un evento está finalizado si y solo si tiene un acta asignada

Context Evento inv:

self.miActa->notEmpty() implies self.isFinalizado() and

self.isFinalizado() implies self.miActa->notEmpty()

* El saldo de un comprador no puede ser negativo

Context Comprador inv:

self.saldo >= 0

* Los equipos que aparecen en los resultados de un acta son los mismos que los que los equipos del evento de acta

Context Acta inv:

self.miEvento->notEmpty() implies self.resultado.equipos = self.miEvento.misEquipos

* Las lesiones que aparecen en un acta pueden hablar de un deportista o de un árbitro del evento

Context Lesión inv:

self.miEvento->notEmpty() implies (

self.miActa.miEvento.misEquipos.misDeportistas->includes(self.persona) or

self.miActa.miEvento.misArbitros->includes(self.persona)

)

* Las sanciones que aparecen en un acta pueden hablar de un deportista del evento

Context Sanción inv:

self.miEvento->notEmpty() implies (

self.miActa.miEvento.misEquipos.misDeportistas->includes(self.deportista)

)

* Las competiciones de dos equipos por evento tienen una final y una de consolación donde se juegan el tercer y cuarto puesto. En estos dos eventos los equipos participantes han de ser distintos

Context Competicion inv

let

final : Evento = self.getEventosOrdenados()->last(),

consolacion : Evento = self.getEventosOrdenados()->at(self.getEventosOrdenados()->size()-2)

in

Competicion.nEquiposPorEvento = 2 implies

final.misEquipos->intersection(consolacion->misEquipos)->isEmpty()

###### PRE Y POST CONDICIONES

context Asiento::reservar(c : Comprador, fecha : Integer) : Entrada

pre:

self.misEntradas->select(e | a.fecha = fecha)->isEmpty()

precio >= 0

post:

result.oclIsNew() and

result.asiento = self and

result.fecha = fecha and

result.precio = damePrecio(fecha) and

result.estado = EstadoReserva::reservada and

result.miComprador->notEmpty() and

self.misEntradas->includes(result)

context Asiento::vender(e: Entrada, c:Comprador) : Boolean

post:

if

e.estado@pre = EstadoReserva::reservada and

c = e.miComprador and

c.saldo@pre > e.precio and

self.includes(e)

then

result = true and

e.estado = EstadoReserva::comprada and

c^ajustarSaldo(-e.precio)

else

result = false

endif

Context Asiento::anular(e: Entrada, c : Comprador) : Boolean

post:

if

e.estado@pre = EstadoReserva::reservada and

c = e.miComprador and

self.includes(e)

then

result = true and

e.estado = EstadoReserva::disponible and

e.miComprador->isEmpty() and

self.misEntradas->excludes(e)

else

result = false

endif

context Comprador::ajustarSaldo(incremento :: Integer) : Integer

pre:

self.saldo + incremento >= 0

post:

self.saldo = self.saldo@pre + incremento

context Entrada::isCaducada() : Boolean

body:

if self.estado = EstadoEntrada::reservada

then time() < momentoReserva + caducidadReserva

else false

endif

Context Emplazamiento::quitarEvento(e : Evento) : Boolean

post:

self.misEventos = self.misEventos@pre->excluding(e)

Context Emplazamiento::puedoAsignarEvento(e : Evento) : Boolean

post:

if

self.misEventos->includes(e)

then

result = true

else

result =

e.miCompeticion.requierePiscinas implies

self.damePiscinasLibres(e) > 1 and

e.miCompeticion.requierePiscinas implies

self.damePistasAtletismoLibres(e) > 1 and

e.miCompeticion.requierePiscinas implies

self.dameCircuitosLibres(e) > 1

endif

Context Emplazamiento::asignarEvento(e : Evento) : Boolean

post:

if self.puedoAsignarEvento(e)

then

self.misEventos = self.misEventos@pre->including(e)

result = true

else

result = false

Context Evento::reservarEmplazamiento(e : Emplazamiento) : Boolean

post:

if e = self.miEmplazamieto@pre then result = true

else

if e.puedoAsignarEvento(self)

then

(self.miEmplazamieto@pre->notEmpty() implies

self.miEmplazamiento@pre^quitarEvento(self)) and

self.miEmplazamiento = e and

e^asignarEvento(self) and

result = true

else

result = false

endif

endif

Context Acta::getResultado(e:Equipo) : Integer

pre:

self.miEvento.misEquipos->includes(e)

post:

result = self.resultado->at(self.miEvento.misEquipos->indexOf(e))

Context Evento::getClasificacion() : OrderedSet(Equipo)

pre:

self.finalizado()

post:

if self.miCompeticion.ganaElMayorResultado

then

result = self.miActa.resultado->sortedBy(resultado).equipo

else

result = self.miActa.resultado->sortedBy(-resultado).equipo

endif

Context Competicion::getEventosOrdenados() : OrderedSet(Evento)

derive:

self.misEventos->asOrderedSet()->sortBy(horaInicio)

Context Competicion::getMedallero() : OrderedSet(Equipo)

pre:

self.misEventos->select(e | not e.finalizado())->isEmpty()

post:

let eventosOrdenados : OrderedSet(Evento) = self.getEventosOrdenados() in

if eventosOrdenados->last().getClasificacion()->size() >= 3 or misEquipos->size() < 3

then

eventosOrdenados->last().getClasificacion()

else

eventosOrdenados->last().getClasificacion().prepend(

eventosOrdenados->at(eventosOrdenados->size()-2).getClasificacion()

->first())

endif

Context Evento::finalizar(a : Acta) : Boolean

post:

if self.miActa->isEmpty() then

self.miActa = a and

a.miEvento = self and

result = true and

else

result = false

endif

# Diagramas de estado

## Eventos

### Observaciones

## Entradas

### Observaciones

# Diagramas de actividad

## Eventos

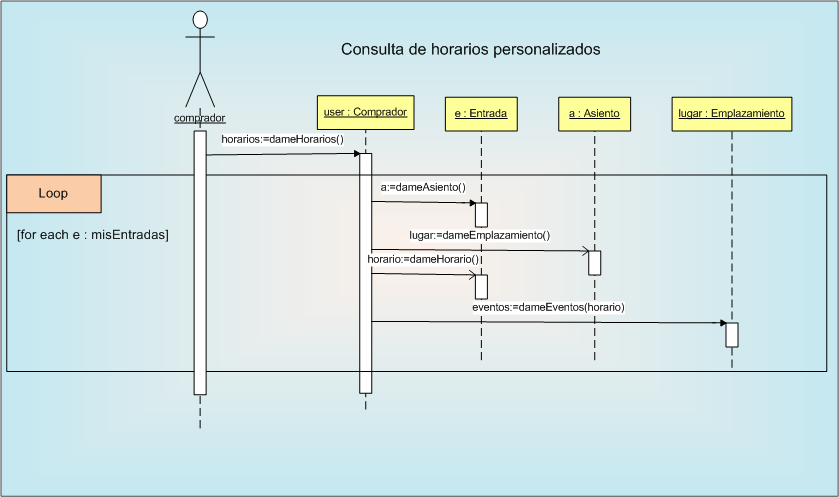
### Observaciones

## Entradas

### Observaciones

# Diagramas de secuencia

## Obtener clasificación

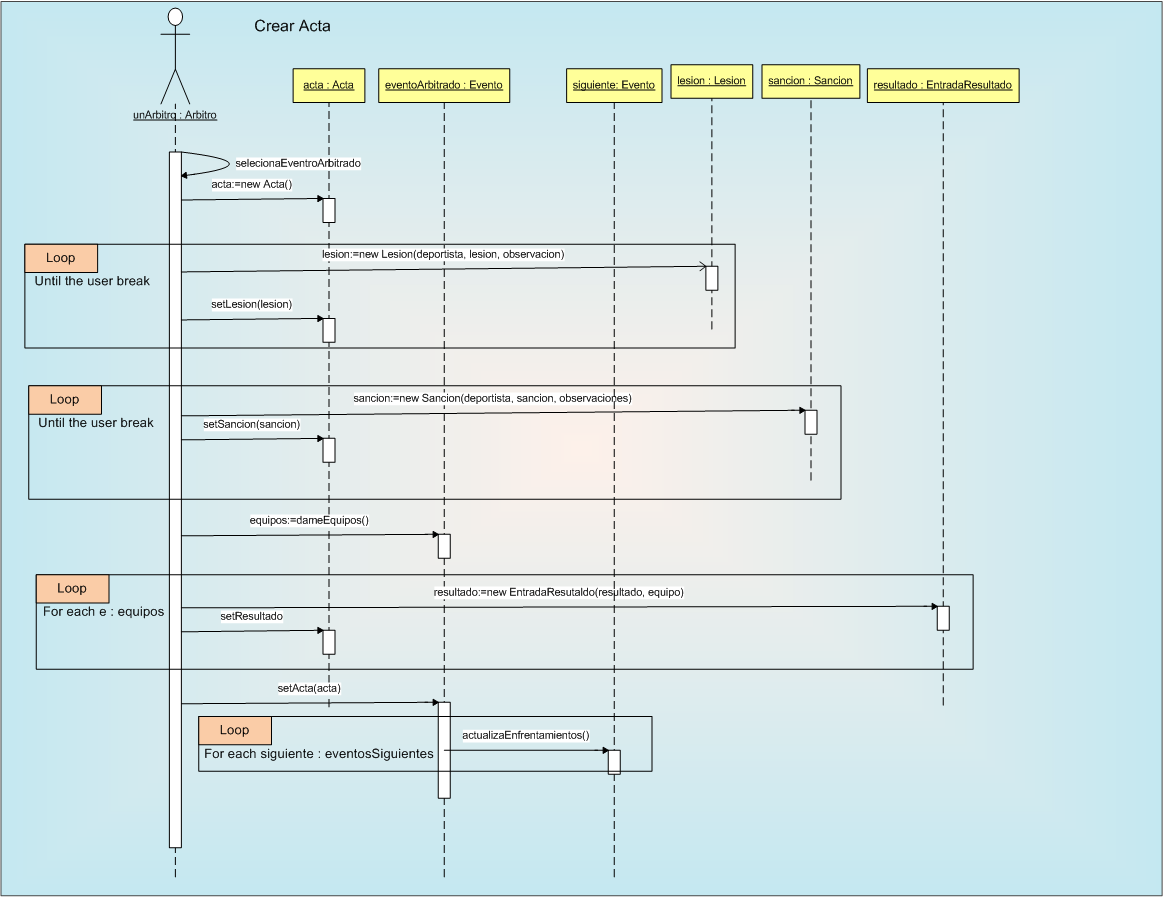


### Observaciones

Este diagrama muestra cómo un usuario puede consultar el estado de una competición. Esto incluye resultados de partidos anteriores, próximos eventos, actas de los equipos etc.

### Restricciones

## Consultar horarios personalizados

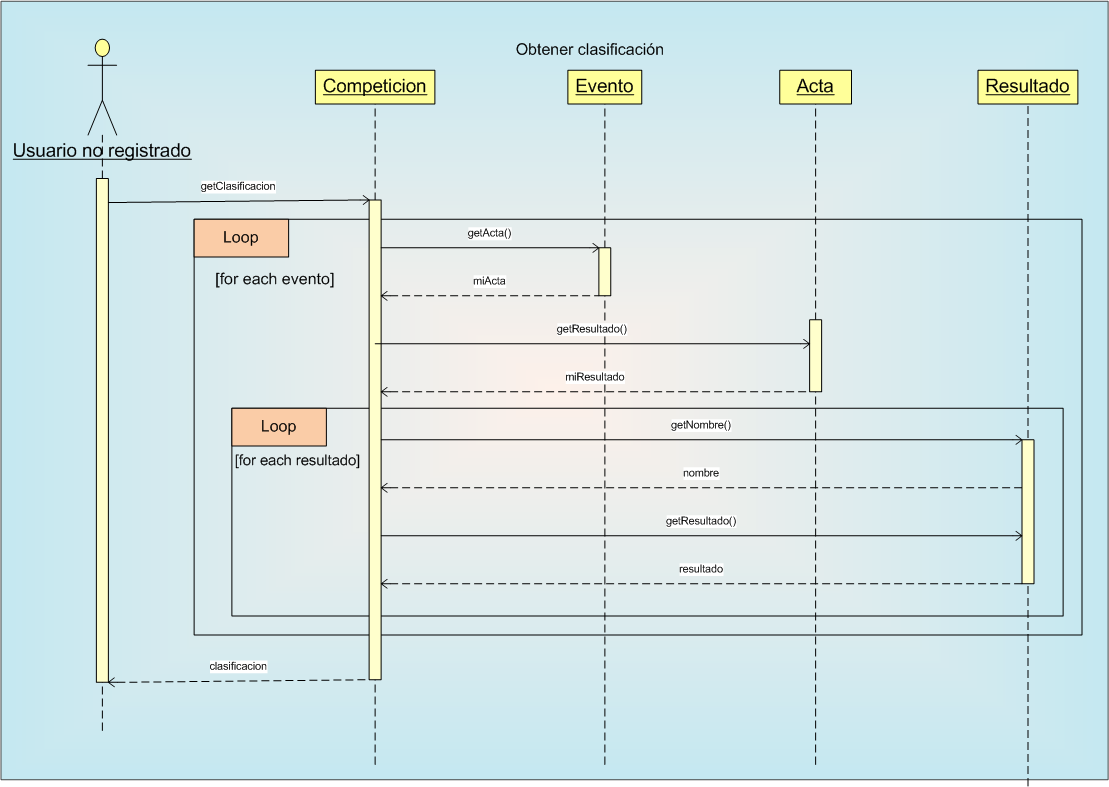


### Observaciones

Este diagrama de secuencia representa cómo se genera un horario personalizado dependiendo de las entradas. El usuario solicita los horarios que le serán representados en la interfaz web. Esto se realiza con el objeto comprador, que será el que consultando todas las entradas y sus respectivos asientos y emplazamientos, dará una lista detallada de todo el horario.

### Restricciones

## Crear acta

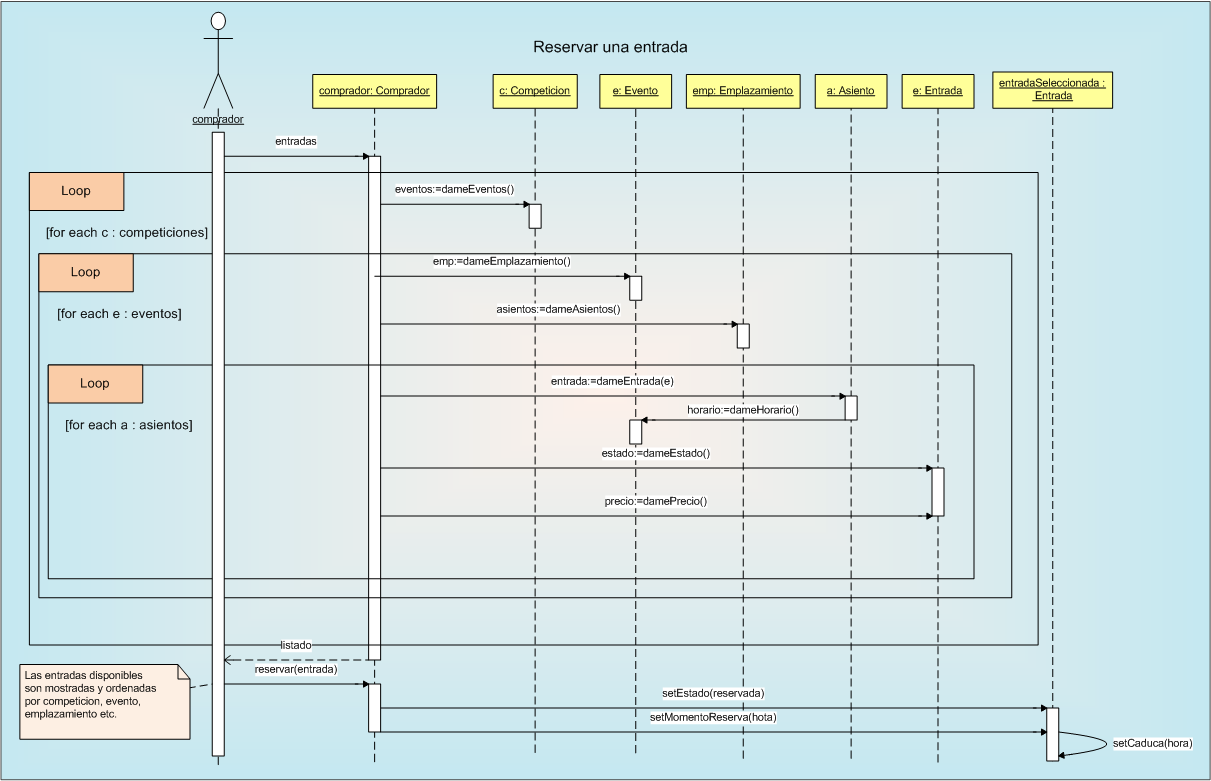


### Observaciones

Este diagrama de secuencia muestra como se crea un acta. Las actas una vez creadas no pueden ser modificadas y quedan almacenadas en el sistema. Al introducir el acta hay que especificar los atributos del acta y al almacenarla se actualizan, en caso de existir, los siguientes eventos de la competición.

### Restricciones

## Reservar entradas



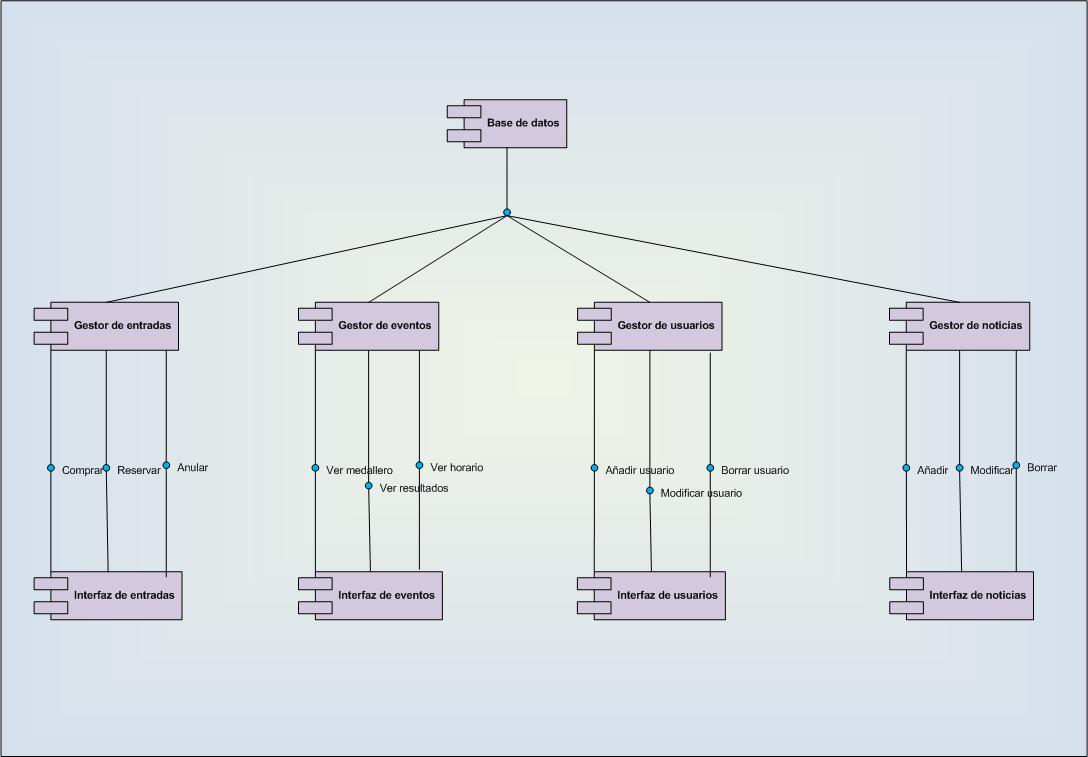
### Observaciones

Para realizar una reserva lo primero que se hace es comprobar el estado de las entradas disponibles, haciendo un listado con todas las entradas disponibles y seleccionables. En el caso de existir entradas de varios tipos, si la entrada no se ajusta al perfil de comprador (ya sea porque no está disponible para él o porque ese tipo de entrada es específico para determinado tipo de personas) no aparecerán en el listado. Una vez realizada la petición de reserva, ésta es realizada.

### Restricciones

# Diagramas de componentes

## Diagramas de Componentes



## Base de datos

* La base de datos de nuestra aplicación constará de varias tablas para los diferentes usos del sistema. Habrá tablas para el registro de los usuarios, para la venta de entradas, equipos participantes, instalaciones, etc.
* Alguna de esas tablas nos la enviará el COI (como la tabla equipos y la tabla competiciones) y en nuestra aplicación solo se podrán utilizar, nunca modificar. Otras tablas nos la dará el ayuntamiento de la ciudad organizadora (tabla instalaciones) y al igual que antes, nuestra aplicación no modificara esa tabla.
* El resto de tablas son las que mediante el sistema de gestión de bases de datos, abajo explicado, y los gestores harán que nuestra aplicación tenga una funcionalidad completa.
* En el apartado relativo a modelo de datos se explicara más detalladamente el formato de cada una de las tablas

###### SISTEMA DE GESTION DE BASE DE DATOS

* Es el componente que nos ayuda a gestionar la base de datos de la aplicación, el espacio físico y nos proporciona unas interfaces de acceso primitivas. Es el puente entre los gestores y las tablas de la base de datos.
* La interfaz consta de métodos de acceso a tablas (select), de creación de tablas (create), de borrado de tablas (drop) , borrado de información de una tabla (delete), modificación de información contenida en tablas (update) y de las propias tablas (alter) , y por ultimo un método para insertar información en las bases de datos (insert).

## Gestores

###### GESTOR DE ENTRADAS

* Es el componente a través del cual vamos a gestionar las entradas de los espectadores a las distintas disciplinas olímpicas.
* Va a ser el que va a recibir las ordenes de los compradores a través de la interfaz de entradas y el que va a tener que utilizar el SGDB para acceder y modificar las tablas. Gracias a él, los datos que los usuarios puedan introducir, serán coherentes con las tablas de la base de datos relativas a las entradas.
* Las operaciones que realizara sobre las tablas de la base de datos son:
* Reservar entrada : añadirá una fila a la tabla de entradas reservadas
* Comprar entrada: añadirá una fila a la tabla de entradas vendidas si la entrada que se intenta comprar pertenece a la tabla entradas reservadas y en caso afirmativo borrará la entrada de la tabla entradas reservadas.
* Anular entrada : eliminará una fila de la tabla de entradas reservas

###### GESTOR DE USUARIOS

* Es el componente a través del cual vamos a gestionar los usuarios que van a poder disfrutar de las ventajas de esta aplicación.
* A través del formulario de registrarse de la interfaz el gestor va a recibir todos los datos necesarios para que el usuario sea registrado en la aplicación. A partir de ahí el gestor hará las operaciones necesarias para que en la tabla de usuario queden reflejados los cambios.
* Las operaciones que va a realizar el gestor son:
* Registrar un usuario: añadir una fila a la tabla de usuarios.
* Modificar usuario: modificar los datos en la fila de la tabla de usuarios.
* Eliminar usuario: eliminar la fila de la tabla de usuarios.
* Login: comprobar que los datos introducidos están en la tabla de usuarios.

###### GESTOR DE NOTICIAS, FOTOS

* Es el componente a través del cual vamos a gestionar las noticias y fotos que van aparecer en la herramienta.
* En la interfaz de noticias, el usuario podrá introducir el titular y el cuerpo de la noticia, o si lo desea, fotos. Las noticias y fotos se almacenaran en el servidor y el encargado será el gestor.
* Las operaciones que realiza son:
* Añadir noticia
* Modificar noticia
* Añadir foto
* Eliminar foto
* Poner noticia en ultima hora
* En la parte relativa a componentes software se dará más información de este gestor y de cómo se va a integrar en nuestra aplicación

###### GESTOR DE EVENTO

* Es sin duda, el gestor más importante de nuestra aplicación a la vez que el más sofisticado.
* Utilizará las tablas de la base de datos de deportes, equipos e instalaciones que le vendrán dadas por el COI y el ayuntamiento para realizar un organigrama de toda la competición (en el diagrama de actividad Busca emplazamiento se detalla). Gracias a él se construirá el calendario completo de todos los eventos que se van a realizar en las olimpiadas y les podrá horario e instalaciones. También gestionará la tabla de resultados que después de cada partido se rellenará gracias a los actas y gracias a ello podremos visualizar inmediatamente las clasificaciones y el medallero. Por último el gestor tendrá operaciones más simples, pero no por ello menos importantes como pueden ser: modificar la información de un evento o modificar las actas de un partido.

## Interfaces

* Es el componente a través del cual el usuario va a poder acceder a todas las utilidades que ofrece el sistema, por lo tanto debe ser sencillo y muy intuitivo.
* Habrá una serie de interfaces, cada una la gestionará uno de los componentes arriba descritos.
* Más adelante se podrán ver con más detalle estas interfaces.

# Modelo de datos

## Diagrama entidad-relación



## Modelo relacional

**Equipo**(idEquipo, país, info, idCompeticion)

**Deportista**(idDeportista, nacionalidad, nombre, apellido1, apellido2)

**Pertenece\_Deportista**(idEquipo, idDeportista)

**Competicion**(idCompeticion, nombre, numPart, numArbitros, maxEquipos, minEquipos,

necPiscina,necPista, necCircuito)

**Evento**(idEvento, fase, fecha\_inicio, fecha\_fin)

**Resultado**(idResultado, idEquipo, resultado)

**Lesion**(idLesion, idEvento, lesión, idDeportista, observaciones)

**Sancion**(idSancion, idEvento, idDeportista, penalización, observaciones)

**Evento**\_Resultado(idEvento, idResultado)

**Arbitro**(idArbitro, nombre, apellido1, apellido2, nación)

**ArbitrosEvento**(idArbitro, idEvento)

**Emplazamiento**(idEmplazamiento, idMunicipio, idVia, cp, numero, numCircuitos, numPistas,

numPiscinas)

**Lugar\_Evento**(idEmplazamiento, idEvento, fecha)

**Asiento**(idAsiento, localizador, idEmplazamiento)

**Entrada**(idEntrada, idAsiento, idEmplazamiento, fecha, estado, precio, comprador,

fechaReserva, fechaCaducidad)

## Descripción de las tablas

###### TABLA DE EQUIPO

* El COI nos facilitará los equipos que van a participar en los diferentes deportes olímpicos. Cada equipo será identificado por “idEquipo”, y llevara información como el deporte al que pertenece, la nacionalidad del equipo e información adicional del equipo, como puede ser el nombre de los participantes del equipo.

###### TABLA DE DEPORTISTA

* La tabla de deportistas, al igual que la anterior, nos vendrá dada por el COI. Los únicos atributos relevantes para nuestra aplicación son el nombre, los apellidos y la nacionalidad.

###### TABLA DE PERTENECE\_DEPORTISTA

* Esta tabla se encarga de relacionar los equipos con los deportistas que van a formar cada equipo.

###### TABLA DE COMPETICION

* En esta tabla se alberga la información principal de cada competición. Los datos nos los facilitará el COI.

###### TABLA DE EVENTO

* La tabla de eventos va a recoger la información de cada uno de los partidos que se van a realizar en la olimpiada. Nuestro sistema, a partir de las tablas de deportes, instalaciones y equipos gestionará esta tabla automáticamente. El identificador de cada evento va a ser “idEvento” y va a llevar información básica, como puede ser los equipos participantes, los árbitros, el deporte, la instalación, la fase de competición o la fecha

###### TABLA DE RESULTADO

* Esta tabla va a ser utilizada por los árbitros al introducir las actas de los eventos. Se guardara el resultado en tuplas de equipo – resultado. Esto es así para facilitarnos el almacenamiento de resultados de eventos donde participen más de un equipo, ya que, por ejemplo, en atletismo se pueden almacenar el corredor y su tiempo, en fútbol el equipo junto a su resultado etc.

###### TABLA DE LESION

* Esta tabla va a ser utilizada por los árbitros al introducir las actas de los eventos. Se guardará la lesión y el deportista que la sufre. En el campo observaciones se puede añadir información adicional como la duración estimada de la lesión o la gravedad.

###### TABLA DE SANCION

* Esta tabla va a ser utilizada por los árbitros al introducir las actas de los eventos. Se guardará la sanción, el deportista que ha sido sancionado y en qué deporte. También contiene un campo observaciones donde introducir información adicional como una descripción de la acción sancionada.

###### TABLA DE EVENTO\_RESULTADO

* La misión de esta tabla es relacionar los eventos con el resultado de cada uno de los equipos que ha participado en el evento

###### TABLA DE ÁRBITRO

* Es otra tabla que nos dará el COI y simplemente se guardará el nombre y los apellidos del árbitro, el deporte que va a arbitrar y la nacionalidad de éste, muy importante a la hora de que no arbitre partidos de su propia nación.

###### TABLA DE ARBITROS\_EVENTO

* En esta tabla habrá una relación entre los árbitros y los eventos que van a arbitrar.

###### TABLA DE EMPLAZAMIENTO

* En este caso la tabla del emplazamiento nos la dará el ayuntamiento de la ciudad organizadora y contendrá la información de las instalaciones que posee la ciudad para albergar las olimpiadas. Cada instalación viene identificada por “id” y lleva información como restricciones que tienen que ser compatibles con el atributo restricciones del deporte que se va a realizar en esta instalación. También tiene la capacidad de espectadores.

###### TABLA DE LUGAR\_EVENTO

* En esta tabla se relacionara el evento con el emplazamiento que lo va a albergar.

###### TABLA DE ASIENTOS

* Cada estadio tendrá una tabla de asientos en la que cada asiento estará designado por un identificador “idAsiento”, que pertenecerá a un estadio concreto, “idEstadio”. Todos sabemos que en cada instalación hay asientos mejores y asientos peores, por eso hay que distinguirlo y como es obvio si el asiento es mejor tendrá un precio más caro. También es lógico que un asiento sea más costoso en una fase de competición que en otra, por ello esto también debemos de almacenarlo.

###### TABLA DE ENTRADA

* Existirá una tabla de entradas por cada instalación de las olimpiadas. Inicialmente estará vacía y a medida que se va reservando se va añadiendo una fila. Recordemos que para que una entrada sea vendida primero ha de ser reservada, por lo que las entradas vendidas son las que están en la tabla con el atributo “esReservada” a false. Por último es necesario añadir el identificador del usuario que compra la entrada y la fecha en la que lo hace, ya que hay que recordar que las entradas se compra por instalación y por día.

## Funciones de la base de datos

###### TABLA DE EQUIPOS

* El Access Control de esta tabla lo va tener el COI, las operaciones que va a poder realizar son:

añadirEquipo( idE, id, pais, inf)

pre:

not(Equipo.filas -> exists(f1: fila |

f1.idEquipo=idE and f1.idCompeticion=idC))

post:

Equipo.filas -> exists(f1:fila |

f1.idEquipo = idE and

f1.idCompeticion = idC and

f1.pais = pais and

f1.info = inf)

eliminarEquipo(idE, idC)

pre:

Equipo.filas -> exits(f1:fila |

f1.idEquipo=idE and f1. idCompeticion =idC)

post:

not(Equipo.filas -> exists(f1:fila |

f1.idEquipo=idE and f1. idCompeticion =idC)

modificarDeporte(idE, idC, pais, inf)

pre:

Equipo.filas -> exits(f1:fila |

f1.idEquipo=idE and f1.idCompeticion=idC)

post:

Equipo.filas -> exists(f1:fila |

f1.idEquipo = idE and

f1.idCompeticion = idC and

f1.pais = pais and

f1.info = inf)

###### TABLA DE DEPORTISTA

* El Access control en este caso también lo va a tener el COI, las operaciones que va a realizar son:

añadirDeportista(idD,nacion,nomb,ape1,ape2)

pre:

not(Deportista.filas -> exists(f1: fila |

f1.idDeportista=idE))

post:

Deportista.filas -> exists(f1:fila |

f1.idEquipo = idD and

f1.nacionalidad = nacion and

f1.nombre = nomb and

f1.apellido1 = ape1

f1.apellido2 = ape2)

eliminarEquipo(idD)

pre:

Deportista.filas -> exits(f1:fila |

f1.idDeportista=idD)

post:

not(Deportista.filas -> exists(f1:fila |

f1.idDeportista=idD)

modificarDeporte(idD,nacion,nomb,ape1,ape2)

pre:

Deportista.filas -> exits(f1:fila |

f1.idDeportista=idD)

post:

Deportista.filas -> exists(f1:fila |

f1.idEquipo = idD and

f1.nacionalidad = nacion and

f1.nombre = nomb and

f1.apellido1 = ape1

f1.apellido2 = ape2)

###### TABLA DE PERTENECE\_DEPORTISTAS

* Esta tabla al relacionar los deportistas con el equipo al que pertenecen, el Access control es también del COI. Las operaciones que se pueden utilizar son:

añadirPertenece\_Deportista(idE,idD)

eliminarPertenece\_Deportista(idE,idD)

modificarPertenece\_Deportista(idE,idD)

* Todas las operaciones anteriores son muy similares a añadirDeporte, modificarDeporte y eliminarDeporte, respectivamente. En esta tabla también van a ser importantes las operaciones de consulta como:

Equipo\_de\_deportista(idD)

pre:

Pertenece\_Deportista.filas->exits( f1:filas |

f1.idDeportista = iD

post:

return Pertenece\_Deportista.f1.idEquipo

Deportistas\_de\_Equipo(idE)

pre:

Pertenece\_Deportista.filas->exits( f1:filas |

f1.idEquipo = idE

post:

return Pertenece\_Deportista.f1.idDeportista

###### TABLA DE COMPETICION

* Esta tabla contiene toda la información relativa a la competición por lo que va a ser gestionada por el COI. Las operaciones son muy a las de la tabla deportista:

añadirCompeticion(idCompeticion, nombre, numPart, nArbitros, maxEquipos, minEquipos, necPiscina,necPista, necCircuito)

modificarCompeticion(idCompeticion, nombre, numPart, nArbitros, maxEquipos, minEquipos, necPiscina,necPista, necCircuito)

eliminarCompeticion(idCompeticion)

###### TABLA DE EVENTO

* Las operaciones que se pueden realizar son:

crearEvento(idEvento, equipos, árbitros, idDeporte, Emplazamiento, faseCompeticion, fecha)

modificarEvento(idEvento, equipos, árbitros, idDeporte, id Emplazamiento, faseCompeticion, fecha)

EliminarEvento(idEvento)

###### TABLA DE RESULTADO

* Los árbitros emplean las siguientes consultas para poner los resultados:

crearActa(idE, res, obs)

pre:

not(Resultados.filas->exits(f1:filas |

f1.idEvento = idE)

post:

Resutados.filas->exits(f1:filas |

f1.idEvento = idE and

f1.resultado = res and

f1.observaciones = obs)

modificarActa(idE, res, obs)

pre:

Resutados.filas->exits(f1:filas |

f1.idEvento = idE)

post:

Resutados.filas->exits(f1:filas |

f1.idEvento = idE and

f1.resultado = res and

f1.observaciones = obs)

eliminarActa(idE)

pre:

Resutados.filas->exits(f1:filas |

f1.idEvento = idE)

post:

not(Resultados.filas->exits(f1:filas |

f1.idEvento = idE)

###### TABLA DE LESION

* El Access control de esta tabla pertenece al árbitro, las operaciones que puede realizar son:

añadirLesion(idLesion, idEvento, lesión, idDeportista, observaciones)

modificarLesion(idLesion, idEvento, lesión, idDeportista, observaciones)

eliminarLesion(idLesion, idEvento)

###### TABLA DE SANCION

* El Access control de esta tabla pertenece al árbitro, las operaciones que puede realizar son similares a las de la tabla lesión:

añadirSancion(idSancion, idEvento, idDeportista, penalización, observaciones)

modificarSancion(idSancion, idEvento, idDeportista, penalización, observaciones)

eliminarSancion(idSancion, idEvento)

###### TABLA DE EVENTO\_RESULTADO

* La tabla está controlada también por el árbitro. Las operaciones que se pueden utilizar son:

añadirEvento\_Resultado(idE,idR)

eliminarEvento\_Resultado(idE,idR)

modificarEvento\_Resultado(idE,idR)

* Todas las operaciones anteriores son muy similares a añadirDeporte, modificarDeporte y eliminarDeporte, respectivamente. En esta tabla también van a ser importantes las operaciones de consulta como:

Resultado\_Evento(idE)

pre:

Evento\_Resultado.filas->exits( f1:filas |

f1.idEvento = idE

post:

return Evento\_Resultado.f1.idResultado

###### TABLA DE ÁRBITRO

* El Access Control de esta tabla es del COI. Las operaciones que se van a poder realizar son:

añadirArbitro(idArbitro, nombre, apellidos, idDeporte)

eliminarArbitro(idArbitro)

modificarArbitro(idArbitro, nombre, apellidos, idDeporte)

* Estas operaciones son muy similares a añadirDeporte, modificarDeporte y eliminarDeporte respectivamente.

###### TABLA DE ARBITROS\_EVENTO

* El Access control de la tabla es de Ayuntamiento. Las operaciones que se pueden utilizar son:

añadirArbitros\_Evento(idA,idE)

eliminarArbitroEvento(idA,idE)

modificarArbitro\_Evento(idA,idE)

* Todas las operaciones anteriores son muy similares a añadirDeporte, modificarDeporte y eliminarDeporte, respectivamente. En esta tabla también van a ser importantes las operaciones de consulta como:

Arbitros\_Evento(idE)

pre:

Arbitros\_Evento.filas->exits( f1:filas |

f1.idEvento = idE

post:

return Arbitros\_Evento.f1.idArbitro

###### TABLA DE EMPLAZAMIENTO

* La tabla relativa al emplazamiento nos la proporcionara el ayuntamiento de la ciudad organizadora, por lo que el Access control es del Ayuntamiento. Nosotros solo vamos a poder acceder a la tabla para hacer consultas. Las operaciones que se van a poder realizar son:

añadirEmplazamiento(idE, idD, rest, cap)

eliminarEmplazamiento(idE, idD)

modificarEmplazamiento(idE, idD, rest, cap)

* Estas operaciones son muy similares a añadirDeporte, modificarDeporte eliminarDeporte respectivamente.

exiteEmplazamiento(rest)

pre:

not( Emplazamiento.filas->isEmpty() )

post:

return Emplazamiento.filas->exits(f1:fila|

f1.restriccion=rest)

###### TABLA DE LUGAR\_EVENTO

* El Access control de la tabla es de Gestor de Eventos. Las operaciones que se pueden utilizar son:

añadirLugar\_Evento(idEmplazamiento, idE, fecha)

eliminarLugar\_Evento(idEmplazamiento, idE, fecha)

modificarLugar\_Evento(idEmplazamiento, idE, fecha)

* Todas las operaciones anteriores son muy similares a añadirDeporte, modificarDeporte y eliminarDeporte, respectivamente. En esta tabla también van a ser importantes las operaciones de consulta como:

Lugar\_Evento(idE,fecha)

pre:

Lugar\_Evento.filas->exits( f1:filas |

f1.idEvento = idE and f1.fecha = fecha)

post:

return Lugar\_Evento.f1.idEmplazamiento

###### TABLA DE ASIENTOS

* El Access control pernecene al gestor de Entradas. Las operaciones que se puede realizar en esta tabla son:

insertarAsiento(idAsiento, localizador, idEmplazamiento)

modificarAsiento(idAsiento, localizador, idEmplazamiento)

eliminarAsiento(idAsiento)

###### TABLA DE ENTRADA

* El Access Control es el gestor de entradas y las operaciones que puede realizar son:

insertarEntrada(idEntrada, idAsiento, idEmplazamiento, fecha, estado, precio, comprador, fechaReserva, fechaCaducidad)

modificarEntrada(idEntrada, idAsiento, idEmplazamiento, fecha, estado, precio, comprador, fechaReserva, fechaCaducidad)

eliminarEntrada(idEntrada, idAsiento, idEmplazamiento)

verPrecio(idE, idA, idEmpl)

pre:

Entrada.filas->exits( f1:filas |

f1.idEntrada = idE and f1.idEmplazamiento = idEmpl and f1.idAsiento = idA)

post:

return Entrada.f1.precio

# Patrones

## PATRÓN SINGLETON

El objeto de este patrón es garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella.



Usaremos este patrón cuando deba haber exactamente una instancia de una clase y deba ser accesible a los clientes desde un punto de acceso conocido. Por ejemplo:

* Acceso al archivo de configuración de una aplicación
* Gestor de impresión
* Una fábrica de objetos

# Interfaz de usuario

# Investigación sobre los componentes

Primeramente exponer que vamos a utilizar una forma de programación en 3 capas. La capa de presentación, la de negocios y la de datos. La capa de presentación va a ser el componente de la interfaz, la capa de negocios va a ser la propia aplicación y la capa de datos va a ser la base de datos.

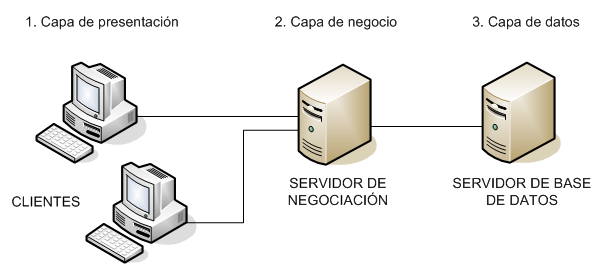


Figura: Modelo en tres capas

## Propuesta para la interfaz

La interfaz se va a desarrollar en lenguaje PHP. Algunas de las ventajas que hemos visto en utilizar este lenguaje para desarrollar nuestra interfaz son:

* Es un lenguaje multiplataforma.
* El código fuente escrito en PHP es invisible al navegador y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
* Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
* Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.

La interfaz va a consistir básicamente en la parte visual, todos los accesos a la base de datos se van hacer a través de la capa de negocio, la interfaz solo va a llamar a las funciones de esa capa para que se ejecuten. El creador de la interfaz no tiene porque saber el funcionamiento exacto del sistema.

El lenguaje que vamos a utilizar para la aplicación es Java, por lo que tenemos que llamar desde PHP a java, el siguiente enlace nos muestra como se puede llevar a cabo:

<http://gonetil.wordpress.com/2008/07/28/instalar-phpjava-bridge-en-linux/>

## Propuesta para la base de datos

Para desarrollar la nuestra base de datos nos hemos decantado por la última versión estable de Oracle, la Database 11g Enterprise Edition, ya que Oracle tiene el record mundial en rendimiento y en la relación calidad/precio, lo cual nos viene muy bien para nuestro sistema.

Las características más importantes que posee Oracle son:

* Es el gestor de bases de datos más utilizado por las compañías más grandes del mundo, por su robustez y por la seguridad.
* Tiene múltiples versiones para cada sistema operativo, entre ellos UNIX, LINUX, NT, Win 2000- 98 - 95, Novel etc.
* Ofrece soporte mundial a través de sus centros de soporte y sus sitios Web donde se puede encontrar desde scripts hasta documentos de instalación.
* Proporciona actualización de versiones gratis al cliente; siempre y cuando la licencia de soporte lo cubra.
* El 80% o más de los sitios Web en Internet tienen bases de datos Oracle.
* Oracle es más que un gestor de bases de datos. Oracle Corporation ofrece otras soluciones a la plataforma de Negocio, EBusiness, Ecommerce etc.
* Puedes programar con las últimas herramientas del mercado.

## Propuesta para el blog

Nuestra página web posee una parte de noticias, donde se actualizarán las novedades que vayan surgiendo y que gestionaremos por medio de un blog. Para esta parte nos hemos decantado por Wordpress, ya que sólo con instalarlo tienes acceso a los blogs.

Para instalar el componente blog en nuestra página web únicamente hemos incorporando Wordpress a nuestra página en PHP. La forma de hacerlo es bastante sencilla tal y como se muestra en el siguiente enlace:

<http://ezekinho.com.ar/?p=28>

Al registrarse en nuestra aplicación, damos opción al usuario de suscribirse a un servicio de noticias y actualizaciones para que las pueda consultar en su correo o en su teléfono móvil. Para ello vamos a incluir en nuestro blog el protocolo RSS, que es una herramienta para publicar o visualizar artículos de una sola fuente en diferentes medios al mismo tiempo, generalmente al instante de publicarlos. Muchos medios en línea ponen a disposición su feed (dirección única del RSS del sitio) para mostrar los últimos titulares.

En el siguiente enlace se puede ver cómo hacerlo:

<http://www.vidadigital.net/blog/2008/04/21/rss-y-tu-blog-en-wordpresscom/>

## Propuesta para el componente de pago

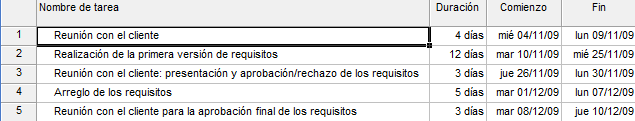
PayPal es un sistema que permite a cualquier persona que tenga una dirección de correo electrónico enviar o recibir dinero online utilizando su tarjeta de crédito de manera totalmente segura.

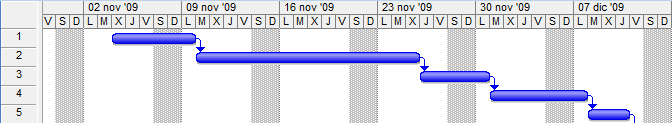
Hemos pensado que sería una forma amena y sencilla de incorporar pagos online para nuestra aplicación, ya que dispone de herramientas guiadas y sencillas para la integración del paquete en la aplicación web. Además, es gratuita.

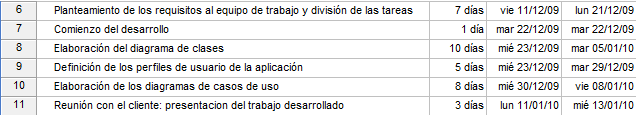
La forma de integrarlo se ve en la siguiente página:

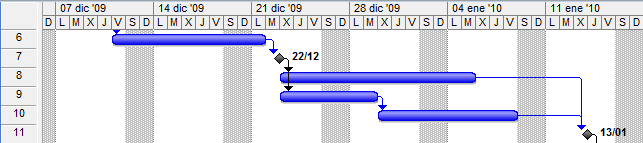
<http://www.elwebmaster.com/articulos/un-script-php-para-integrar-pagos-con-paypal-a-tu-sitio-web>

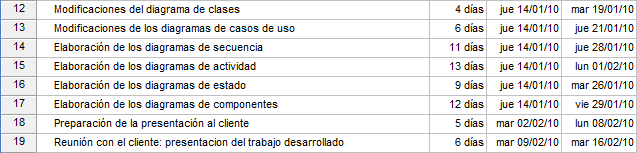
# Planificación estimada

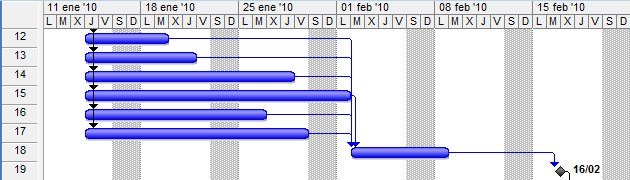


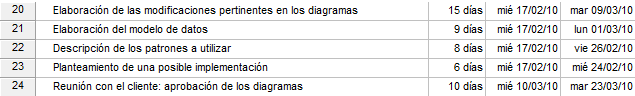


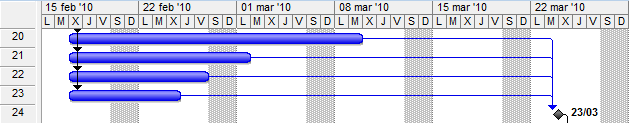




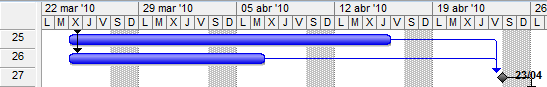


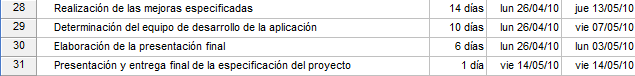


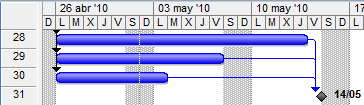












# Propuesta de calidad

# Cuestionario CMMI

###### NIVEL 2 (REPETIBLE)

**A lo largo del desarrollo del proyecto, ¿la forma de trabajo ha sido clara y controlada? ¿Por qué? ¿Se ha seguido algún método, procedimiento y/o entrenamiento?**

Se distinguen dos líneas de opinión muy diferenciadas.

* La forma de trabajo no ha sido clara y controlada desde el principio, y al obtener una visión de cómo y cuánto se avanzaba, se eligió como metodología de trabajo divide y vencerás, es decir, la repartición del trabajo de forma equitativa y con ciertas limitaciones, evitando así tener que consensuar todas las cuestiones que se presentasen (relevantes o irrelevantes), comunicando únicamente las cuestiones de importancia global y con repercusiones.
* La forma de trabajo ha sido clara pero descontrolada, ya que no se eligió ningún método de trabajo determinado y ni se realizo entrenamiento previo a la realización de tareas, lo que conllevo a múltiples suposiciones erróneas y por tanto, a un retroceso (rehacer) en el trabajo.

**¿El proyecto ha tenido una documentación correcta y coherente en cada fase? Argumenta tu respuesta**

Se manifiesta la existencia de opiniones contrapuestas:

* El proyecto no ha tenido una documentación correcta y coherente, ya que los cambios realizados y requerimientos nuevos se comunicaban en las reuniones y no se reflejaban, así como sus motivos y soluciones, en ningún documento escrito. Esta forma de trabajo tuvo como resultado la necesidad de retomar debates ya concluidos.
* El proyecto ha tenido una buena documentación, acorde al desarrollo de las tareas. Se mantuvo una documentación visual que facilitaba la necesidad de otras explicaciones, y se controlaron las versiones de la documentación mediante el repositorio donde se alojo el proyecto durante su desarrollo.

**En todo momento, ¿se ha conocido la documentación a presentar en la entrega final?**

Se distinguen dos líneas de opinión muy diferenciadas.

* Durante el desarrollo del proyecto no ha sido conocida la documentación a presentar, hasta el final del proceso.
* La documentación a presentar ha sido conocida pero de forma inexacta, es decir, no en su totalidad; ya que se desconocía, de forma exacta, el alcance y nivel de profundidad que se deseaba obtener, porque siempre iba en aumento y la principal fuente de conocimiento a lo que se accedió fueron proyectos de años anteriores.

**¿La documentación final ha sido elaborada desde el principio del proyecto?**

Se manifiesta la existencia de opiniones contrapuestas:

* La documentación ha sido elaborada desde el principio, sentando las bases más solidas posibles para obtener la coherencia y consistencia del documento, que ha sido modificado en numerosas ocasiones según aumentaban los requisitos de la entrega así como las modificaciones propias del proyecto, en cuanto a su definición y su alcance.
* La documentación final se ha relegado al último paso en el desarrollo, ya que se entiende como un documento recopilatorio de todo el trabajo realizado, que brinda la oportunidad de realizar una revisión total de todo el proyecto

**¿El tiempo ha sido distribuido de una forma correcta y necesaria en las distintas fases del proyecto?**

La distribución del tiempo no ha sido la idónea, por diversos factores:

* El método de trabajo al principio no era el adecuado, ya que todos los miembros no podían dedicarse a la misma tarea. Por esto motivo, se opto por la repartición de trabajo de forma equitativa, asumiendo una cantidad de tiempo desperdiciada por la mala organización
* La fase inicial (Especificación de Requisitos) se dio por concluida estando en un estado inexacta y demasiado ambicioso, lo que conllevo a múltiples debates acerca de las funcionalidades a desarrollar y el rediseño que obligaba a realizar los cambios decididos en los debates mencionados.

**¿La planificación del trabajo ha sido la adecuada?**

La planificación no ha sido la adecuada, por diversos motivos;

* La falta de un método de trabajo. El intento de que todos los miembros abarcasen todas las tareas conllevo al retraso y a tomar la iniciativa de dividir el trabajo, estrategia que debió emplearse desde el inicio del proyecto.
* Los innumerables debates sobre cualquier cambio. Se debería debatir solo lo imprescindible, es decir, aquellos cambios que tengan una repercusión real y significativa; y comunicar a todos los miembros en reuniones cortas los avances realizados por cada subgrupo.
* Falta de especificación en la idea a desarrollar. Se debería dedicar más tiempo en la especificación de requisitos dejando aclarado en su totalidad, y en medida de lo posible, sin lugar a ambigüedades, las funcionalidades y repercusiones de cada requisito, haciendo consciente al equipo del nivel de ambición del proyecto y el tiempo necesario para desarrollarlo, como también evitando la necesidad de rediseñar.
* Mayor conocimiento del trabajo a realizar desde el inicio del proyecto, facilitando de esta forma la metodología de repartición de tareas.
* Diseñar un estructura o jerarquía con roles bien definidos, existiendo una figura encargada de organizar al resto del equipo según las necesidades del proyecto.

**¿La especificación de requisitos ha sido bien definida desde el inicio?**

La especificación de requisitos no ha estado bien definida, debido a la falta de detalle en su definición y el exceso de ambición debido a las funcionalidades que se planteaban, lo que generó la necesidad de reducir dichas funcionalidad a medida que avanzaba el proyecto, y los innumerables debates causados por las ambigüedades o falta de exactitud en las definiciones; debiendo realizar reestructuraciones y rediseños.

**Porcentaje de requisitos modificados**

Los requisitos han sido modificados en tordo al 60%, siendo una gran parte de estas modificaciones eliminaciones de requisitos. Las modificaciones realizadas han supuesto una gran pérdida de tiempo y esfuerzo, dando lugar a replanteamientos y rediseños, pasando por extensos debates sobre cada modificación.

###### NIVEL 3 (DEFINIDO Y ESTRUCTURADO)

**¿La forma de trabajo ha funcionado bien?**

Se distinguen dos líneas de opinión muy diferenciadas.

* La forma de trabajo ha funcionado bien al final, después de la evolución del grupo hasta conseguir su total integración.
* La forma de trabajo no ha funcionando bien porque no ha sido la adecuada, el proyecto se desarrollo bajo un grupo con una metodología desordenada, en la que todos los miembros trabajaban y opinaban sobre todo generando grandes pérdidas de tiempo y exceso de modificaciones

**¿La definición de cada componente del proyecto ha sido la adecuada?**

Cada componente ha tenido una definición correcta en un punto cercano a la versión definitiva. El principal obstáculo para obtener la definición adecuada eran las distintas modificaciones que se realizaban sobre la misma, ya que cada miembro del grupo tenía una visión diferente, lo que sumía al equipo en deliberaciones hasta obtener un punto en común (olvidado y posteriormente retomado en diversas ocasiones por la falta de documentación)

**¿Las herramientas utilizadas en cada parte del desarrollo del proyecto han sido adecuadas?**

Las herramientas han tenido que reemplazarse en varias ocasiones ya que no abarcaban todas las tareas que se requerían. Entre las limitaciones de las herramientas destacan:

* Incapacidad intrínseca para dividir el trabajo con Subversion (Trabajo en paralelo).
* Incapacidad de integración con otros modelos (OCLs)
* Ausencia de corrección sintáctica (OCLs)

**¿El estado y avance del proyecto ha sido conocido, en todo momento, por cada miembro del equipo?**

Cada miembro tenía a su disposición, en todo momento y en exactitud, conocer el estado y modificaciones realizadas en el proyecto mediante el envío de correos y actualizaciones del repositorio

Aunque el equipo es unánime en cuanto a la disponibilidad para conocer el estado del proyecto, se distinguen opiniones opuestas basadas en la dificultad para conseguirlo:

Todas las modificaciones eran consensuadas y en su mayoría realizadas en las horas lectivas de la asignatura (laboratorios) con la presencia de la mayoría (o totalidad) de los miembros del equipo.

Las actualizaciones en el repositorio se realizaban sin documentar (comentarios explicativos) y de forma autoritaria o no consensuada (ya que se suponía el concilio de opiniones entre todos los miembros), generando discusiones y una gran complejidad por desconocer los motivos respecto a modificaciones hechas.

**¿El método de trabajo ha sido centralizado (reuniones periódicas)?**

El método de trabajo ha sido centralizado ya que las horas lectivas en el laboratorio facilitan esta tarea. Se realizaban reuniones 2 veces por semana de 2 horas de duración para la puesta en común y nueva división de tareas.

**¿Qué porcentaje (%) de tiempo se ha invertido en el aprendizaje necesario para desarrollar el proyecto?**

El aprendizaje necesario se aproxima a un 20 %, ya que gran parte de esta tarea se realizaba en paralelo con las tareas a las que se destinaba el aprendizaje.

En oposición a esta opinión:

* No se invirtió tiempo en aprendizaje ya que lo explicado en las horas lectivas de teoría era suficiente, en suma con otras asignaturas anteriores.

**¿Existe coordinación entre las diferentes tareas de los distintos miembros?**

Existe coordinación ya que al tener un método de trabajo centralizado, todos los miembros conocen las tareas que se desarrollan en cada etapa así como el subgrupo o miembro encargado de dicha tarea.

**¿Qué herramientas han favorecido la coordinación entre los miembros?**

* Repositorio GoogleCode: Olimpiadas2mil
* Grupo de correo (GoogleGroups): Olimpiadas2020
* Eclipse (herramienta de modelado gratuita que interactúa con Subversion)

###### NIVEL 4 (GESTIONADO)

**¿Qué porcentaje (%) de tiempo dedicado al proyecto han sido fructífero?**

La mayoría del equipo estima que aproximadamente un 60% del tiempo dedicado ha sido fructífero, mientras que existe una estimación que considera un 30%.

**¿Qué porcentaje (%) de tiempo se ha dedicado a cada fase del proyecto? ¿Ha sido el adecuado?**

* Especificación de requisitos: 20%
* Diseño (Diagramas, invariantes, precondiciones, postcondiciones,…): 60%
* Memoria y documentación: 20%

No ha sido la distribución adecuada, por no generar un resultado óptimo. La escasa dedicación en la especificación de requisitos conllevo a gran número de debates y forzó a múltiples etapas de rediseño por cambios realizados.

**¿Qué tareas han significado una pérdida de tiempo?**

* Cambios de herramientas (rehacer lo que ya estaba hecho)

Las modificaciones y correcciones en los requisitos han significado un retraso significativo y las discusiones que generaban, un aumento en gran medida de este tiempo desperdiciado, ya que cada debate significaba el replanteamiento de alguno o varios elementos y su correspondiente funcionamiento.

* Diseños realizados (inconclusos) desechados.

**¿El tiempo para realizar una tarea ha podido ser estimado/predicho?**

La estimación de una tarea dependía en gran parte de las complicaciones que generará, debido a la realización de algún cambio y las correspondientes modificaciones que se deben realizar; por este motivo, el equipo se divide en dos posturas:

* Subgrupo que considera que podrían estimarse las tareas (aproximadamente)
* Subgrupo que considera que el tiempo dedicado a realizar una tarea no es predecible.

**¿En cada parte del proyecto, se ha distinguido la figura de un líder?**

No se distingue ninguna figura líder, cada miembro era consciente y responsable de la tarea que había elegido a realizar. Existían miembros más conscientes de las tareas a realizar que servían de guía para el desarrollo del proyecto.

**¿Se han distinguido especialistas en el proyecto?**

No existen especialistas declarados. Durante gran parte del desarrollo del proyecto el trabajo se ha dividido de forma equitativa en subgrupos, permitiendo que todos los miembros pudiesen participar en distintas tareas en las diversas fases.

**¿Existen diferentes puntos de vista sobre la gestión del proyecto?**

Existen tantos puntos de vista como miembros. Al no tener una estructura jerarquizada ni una especificación de requisitos totalmente clara y definida, cada miembro intentaba desarrollar el proyecto bajo su visión personal lo que generaba enfrentamientos (grandes pérdidas de tiempo). A medida que el proyecto tomaba forma y consistencia los debates disminuían, ya que toda la idea pasaba a tener claridad, consistencia y precisión.

**¿El progreso en el proyecto ha sido lineal? Argumenta tu respuesta**

Se divide en dos subgrupos que difieren en:

* El progreso ha sido lineal.
* El progreso del proyecto ha sido por ráfagas, siendo estas positivas (avances significativos en periodos cortos) o negativos (replanteamiento de requisitos, rediseño,… lo que generaba retrasos)

Aunque las ideas de ambos subgrupos se armonizan en la opinión de que en los últimos meses, sujetos a la presión de realizar una entrega, la progresión ha sido exponencial.

**Nivel de madurez obtenido 1,4**

**MODIFICACIONES A APLICAR PARA MEJORAR EL NIVEL DE MADUREZ ALCANZADO**

* Elegir un modelo de desarrollo.
* Revisar cada fase antes de darla por concluida e intentar esclarecer lo más posible cada punto o tarea, para evitar posibles confusiones, discusiones y la necesidad de realizar cambios en tareas concluidas.
* Elección de herramientas, basadas en documentación y experiencia.
* Establecer una jerarquía (elegir un líder), formar especialistas y generar una planificación global y otra dedicada a cada fase.
* Elaborar el documento final desde el principio, para hacernos más conscientes del trabajo ya realizado y las tareas que quedan por hacer.

## Evolución del proyecto

En un principio pensábamos gestionar:

Gestión de las competiciones:

* Creación del cuadro de la competición de cada disciplina.
* Inscripción de los deportistas en las competiciones.

Gestión del emplazamiento:

* Que cumpla con los requisitos del deporte (natación – piscina, tenis – pista de tenis….)
* Que esté bien comunicado con el alojamiento
* Que tenga el aforo adecuado
* Que todos los emplazamientos de la misma disciplina estén lo más cerca posible.

Gestión del horario:

* Que no se solape con una competición de las mismas características
* Cercano a los horarios de máxima audiencia

Gestión de entradas

* Venta de entradas a espectadores
* Pases de prensa para los medios
* Invitaciones para organizadores

Gestión de los participantes:

* Gestión de alojamiento, de forma que el alojamiento este lo más cerca posible o lo mejor comunicado posible de la competición y que este lo más cerca posible del resto de participantes de tu país en otras competiciones.
* Gestión de transporte, para que desde los hoteles a las pistas deportivas haya buena comunicación.

Gestión de los árbitros:

* Gestión de alojamiento: deberían de estar en una zona de la ciudad, aislados de los participantes y todos juntos.
* Gestión del transporte: tendrían un transporte privado para ellos desde su lugar de hospedaje hasta la competición deportiva.

Gestión del personal contratado y voluntario.

* Definición de sueldos.
* Asignación tareas y horarios así como el organigrama de la organización.

Gestión de patrocinadores:

* Asignación de recursos publicitarios en base al grado de colaboración, estos recursos serán gestionados por terceros para la administración de publicidad.
* Gestión de los espacios de publicidad, que incluye el control del cumplimiento de las empresas encargadas de la gestión.
* Acuerdos de colaboración con empresas de transportes para reforzar sus servicios.

Venta de entradas por Internet.

Portal deportivo:

* Mantiene la información de los resultados, medallero, clasificaciones, así como el horario y lugar de las competiciones.

Íbamos a tener los siguientes roles y perfiles:

* ROL: Administración: Posee el control total de la aplicación.
  + - PERFIL: Administrador del Comité de la Sede
* ROL: Control parcial: Solo en algunos aspectos de la aplicación
  + - PERFIL: Autoridad pública (policías, bomberos…)
    - Ejemplo: Hay un accidente grave en la nacional de acceso al evento 3 horas antes. El policía introduce la incidencia y el sistema alerta a la organización, participantes y espectadores que hayan adquirido una entrada por mail o sms. Además en la consulta de rutas de acceso al evento muestra una ruta alternativa para llegar al estadio
    - PERFIL: Federación
    - Ejemplo: Se lesiona 3 horas antes un participante. La aplicación gestiona esa baja e informa a la administración.
    - PERFIL: Espectador
    - Ejemplo: Al comprar una entrada se modifican las localidades libres, se incrementa la recaudación del evento…
    - PERFIL: Medio de comunicación
    - Ejemplo: TVE informa que sus 2 pases de prensa los van a ocupar Pepito Pérez y Juan Rodríguez.
* ROL: Consulta
  + - Cualquier perfil tiene como mínimo el perfil de consulta.
    - Sector servicios: transporte, alojamiento, restaurantes, cines, teatros…
    - Deportivos: participantes, clasificaciones, horarios…

Tras una reunión con personal ajeno a nuestro equipo de desarrollo, decidimos centrarnos en gestionar los eventos y las entradas para estos eventos, y dejar de lado todo lo relacionado con el transporte y alojamiento de los deportistas, árbitros y participantes en general.

También eliminamos la gestión de federaciones (inscripciones de participantes en competiciones etc.)

Eliminamos la publicidad y los pases de prensa.

También eliminamos la gestión de personal porque suponemos que todos esos servicios los subcontratamos.

Añadimos el concepto de ACTA como un informe que el árbitro rellenará al final de cada partido y que proporcionará información relativa a cada evento y que servirá para obtener la información de las competiciones.

En cuanto a los ROLES:

* El COI mantendría un rol de administración
* Hemos eliminado los perfiles de Prensa, Deportista y Autoridad pública, por lo que solo tendría un rol de “Control parcial” el usuario registrado (en la gestión de entradas) y el árbitro (en la gestión de actas)

En cuanto a la funcionalidad:

* Al implementar la interfaz hemos añadido opciones de consulta (noticias, última hora, fotos, villa olímpica, etc)

Conflictos encontrados:

* Antes de eliminar la gestión de transporte hubo discusiones acerca de introducir una clase gestora del transporte que realizara las tareas de reserva de recursos, rutas etc. Pero se descartó porque no se adaptaba a la filosofía de la orientación a objetos.
* En la gestión de entradas pasó algo parecido, teníamos un gestor de entradas que se dedicaba a vender, reservar etc. Decidimos delegar esa funcionalidad en el asiento y esto también generó polémica, pero decidimos dejarlo así.
* En la mitad del proyecto nos dimos cuenta de que estábamos abordando demasiados ámbitos y no nos iba a permitir realizar un buen trabajo con todo, así que decidimos quitar funcionalidades. Sopesamos la situación y decidimos quedarnos únicamente con las entradas, eventos y competiciones.
* En un principio trabajábamos con Bouml, pero daba problemas al trabajar con el SVN porque almacenaba todos los diagramas en un mismo fichero y esto daba conflictos al reconciliar las versiones. Probamos con ArgoUml pero le resultado era el mismo, así que al final nos decantamos por Eclipse, que al guardar cada diagrama en un fichero no nos daba ese problema. Aun así, no sabemos hacer los diagramas de secuencia con el Eclipse y los hemos hecho con ArgoUML.