Universidad de las Ciencias Informáticas

Facultad 8

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Título: "Herramienta para la gestión de torneos ajedrecísticos, versión 2.0"

Autor: Yosbel Pérez Chirino

Tutor: Ing. Alison Muñoz Capote

Ciudad de La Habana, junio 2010

"Año 52 de la Revolución"

Declaración de Autoría

Declaro que soy el único autor de esta tesis titulada: “Herramienta para la gestión de torneos ajedrecísticos, versión 2.0” y autorizo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ___días del mes ___del año _____

Autor:

Yosbel Pérez Chirino

_____

Tutor:

Alison Muñoz Capote

_____

Pensamiento

De pocas partidas he aprendido tanto como de la mayoría de mis derrotas.

José Raúl Capablanca

Agradecimiento

A la Revolución Cubana y a Fidel, nuestro eterno Comandante en jefe por darme la oportunidad de estudiar esta carrera. A mis maestros y profesores que a lo largo de mi vida como estudiante me han formado y enseñado todo lo que sé.

A mi familia en especial a mi mamá y mi papá por el apoyo que siempre me brindaron en todo momento.

A mi tutor, a mis compañeros y a todos mis amigos que de una forma u otra me ayudaron a cumplir esta labor.

Yosbel

Dedicatoria

Le dedico este trabajo y mi esfuerzo a mi familia, mi mamá, mi papá, mi hermano porque siempre estuvieron presentes ante cualquier problema.

A mis amigos, a mis profesores, a la Revolución por darnos la oportunidad a todos los ciudadanos que vivimos en Cuba de formarnos como profesionales.

Yosbel

Resumen

Actualmente en todo el mundo se celebran múltiples torneos de ajedrez, ya sean oficiales o no oficiales, incluyendo la Universidad de las Ciencias Informáticas (UCI) donde se realizan varios eventos de este tipo durante todo el año. Para la gestión de estos torneos de ajedrez se utilizan herramientas privativas, las cuales solicitan el pago de una licencia para su utilización. Producto de la política de migración hacia software libre que pretende llevar a cabo el Estado cubano, se decidió crear una aplicación web que sustituyera a estas aplicaciones privadas en los torneos celebrados en la UCI, surgiendo así, Arbitraje versión 1.0. En el presente trabajo se desarrolló la implementación de la versión 2.0 de la aplicación, la cual podrá gestionar los diferentes tipos de torneos que se celebran en la UCI. Este proceso se desarrolló utilizando a RUP (Proceso de Desarrollo de Software) como metodología de desarrollo, con el objetivo de obtener una mejor organización en el trabajo y obtener un producto de máxima calidad. Al término de este trabajo se pretende que la nueva versión sea utilizada en los eventos ajedrecísticos celebrados en la UCI.

Palabras claves: Software libre, ELO, Infodrez, Ajedrez.

Índice

Introducción

El ajedrez ha tenido en los últimos años un gran auge a nivel mundial debido a la gran cantidad de personas que se han sumado a la práctica este deporte. En nuestro país se ha llevado hasta las escuelas, ya que ayuda a desarrollar el intelecto y habilidades en los niños. En este momento Cuba cuenta con jugadores de prestigio universal, a pesar de ser un país del tercer mundo. Actualmente se desarrollan en todo el mundo muchísimos torneos de esta disciplina, donde en cada uno de ellos necesitan no sólo de la participación de los jugadores, sino también la presencia de un grupo de árbitros. Estas personas son las encargadas de controlar que se cumplan las reglas, así como, conocer el funcionamiento de las herramientas utilizadas para la gestión de los diferentes tipos de torneos de ajedrez existentes hasta el momento.

En la Cátedra Honorífica de Ajedrez “Remberto Fernández” de la Universidad de las Ciencias Informáticas (UCI) se desarrollan torneos, en los cuales participan ajedrecistas de todo el país y se guarda la información referente a los datos de los jugadores y las rondas, de las cuales se tiene en cuenta los enfrentamientos, resultados y clasificación de las mismas. Para poder realizar estas actividades los árbitros utilizan un software propietario llamado Swiss Manager. En medio de los planes de nuestro país de migrar hacia el software libre, se desea desarrollar una aplicación web con herramientas libres, y así sustituir al Swiss Manager en los eventos ajedrecísticos celebrados en la UCI y además incorporarles funcionalidades definidas por el presidente de la Cátedra de Ajedrez.

Por este motivo en el curso 2007-2008 se realizó un trabajo de diploma con el objetivo de realizar una herramienta que permitiera gestionar los contenidos relacionados con el arbitraje en la UCI. Actualmente la versión 1.0 de Arbitraje del proyecto Infodrez, sólo gestiona 2 tipos de torneos (Todos contra todos, variante individual y por equipos), por lo que no se contemplan en la misma la cantidad de torneos que se realizan en la UCI dejando fuera las variantes de juego suizo, variante individual y por equipos, muerte súbita y scheveningen los cuales se desarrollan con frecuencia en la UCI.

Por todo lo anterior planteado, el **problema a resolver** quedaría conformado de la siguiente forma: ¿Cómo mejorar la gestión de información de todos los torneos ajedrecísticos que se celebran en la UCI? Para darle solución a esta problemática se plantea como **objetivo general,** implementar un sistema que sea capaz de gestionar la información de los torneos (Suizo, variante individual y por equipos, muerte súbita y scheveningen) que se realizan en la UCI.

Como **objetivos específicos** se definen:

1 Elaborar el marco teórico de la investigación.

2 Implementar un sistema que sea capaz de gestionar los torneos (Suizo individual y por equipo, muerte súbita y sheveningen).

3 Realizar pruebas al sistema.

El **objeto de estudio** lo constituyen los Sistemas de Gestión de Información de Torneos de Ajedrez.

El **campo de acción** que abarca este trabajo es el proceso de implementación de los Sistemas de Gestión de Información de Torneos de Ajedrez en la UCI.

Como **idea a defender** se plantea: si se elabora un Sistema de Gestión Web que incluya la información de los torneos de ajedrez (Suizo, variante individual y por equipos, muerte súbita y scheveningen), entonces permitirá una mejor gestión de los torneos de este deporte que se desarrollan en la UCI.

Para darle cumplimiento a los objetivos específicos y resolver la situación problemática planteada se proponen las siguientes **tareas** y los **métodos científicos** que se utilizaron para desarrollar las mismas:

- Estudio de la modelación del sistema de arbitraje 2.0 descrita en el trabajo de diploma "Propuestas del Sistema Integrado del proyecto Infodrez".

  Métodos científicos:

  - Análisis Histórico y Lógico: Se realizará con el objetivo de conocer la evolución de esta propuesta.

- Revisión de la arquitectura propuesta en la versión 1.0 del sistema de Arbitraje de Infodrez.

  Métodos científicos:

  - Análisis: Permitirá desglosar toda la información y así poder trabajar sobre esa base.

- Propuesta de arquitectura para la versión 2.0 del sistema Arbitraje.

  Método Científico**:**

- Modelación: Para representar mediante esquemas cómo se debe realizar el sistema y que pasos deben seguirse.

- Implementación de los casos de uso de la versión 2.0 de Infodrez.

  Método Científico**:**

  - Modelación: Para mostrar mediante esquemas cómo es que se hará la implementación del sistema.

- Realización de pruebas a la aplicación.

Los **resultados** que se esperan en este trabajo son:

- Herramienta de arbitraje que gestione los torneos suizos, variantes individuales y por equipos, muerte súbita y scheveningen además de todos contra todos variantes individuales y por equipos.
- Que la herramienta sustituya la que actualmente utilizan los árbitros (Swiss-Manager) en los torneos oficiales que se celebran en la UCI.

Estructura que presenta este trabajo:

Capítulo 1. Fundamentación Teórica: En este capítulo se tratan todos los aspectos teóricos necesarios para la realización de este trabajo, entre los que se encuentran el estudio de herramientas existentes para la gestión de torneos de ajedrez, descripciones y características de la metodología y las herramientas a utilizar en este trabajo.

Capítulo 2. Características del Sistema: En este capítulo se describe la solución propuesta a través de los procesos del negocio. Se presentan los diagramas de los casos de uso del negocio, así como los requerimientos funcionales y los no funcionales.

Capítulo 3. Implementación: En este capítulo se describe cómo estará implementado el sistema a través de los diagramas de componentes y el diagrama de despliegue.

Capítulo 1. Fundamentación teórica

1.1 Introducción

En este capítulo se realizará la fundamentación teórica del módulo de Arbitraje perteneciente a la plataforma de Infodrez, donde se hará un estudio de herramientas similares utilizadas a nivel mundial para la gestión de torneos de ajedrez. Además, se presentará la metodología que se utilizará para realizar este trabajo. Así como las tendencias y tecnologías actuales en los cuales se va a sustentar el mismo.

1.2 Estudio de herramientas similares

El ajedrez es uno de los juegos de mesa más populares del mundo, aunque también es considerado como un deporte. La enseñanza del mismo puede ser útil como forma de desarrollar el intelecto. Es jugado tanto recreativa como competitivamente en clubes de ajedrez, disputando torneos, en Internet, entre y contra máquinas mediante el ajedrez por computadora, e incluso por correo (ajedrez por correspondencia).

Toda competencia o torneo de esta disciplina tienen que contar con la presencia de un grupo de árbitros, que son los encargados de que se cumplan las reglas, demandas y capaces de afrontar las dificultades y penalidades que puedan surgir durante el evento.

Para la gestión de la información generada en estos torneos de ajedrez, los árbitros utilizan herramientas especializadas en estas materias, entre las que se encuentran: Swiss Manager, Swiss Perfect, Swiss Chess, Chess Pairing, Sevilla y Protos. A nivel internacional y nacional las más utilizadas para estas cuestiones son Swiss Manager y Swiss Perfect, que son las que poseen un mayor prestigio, además de que el Swiss Manager es la única reconocida por la Federación Internacional de Ajedrez (FIDE) para gestionar torneos de ajedrez importantes. Por tal motivo solamente se le realiza el estudio a estas herramientas, ya que las demás no son permitidas en torneos oficiales.

1.2.1 Swiss Perfect

Descripción:

Es fácil de usar y posee todas las características necesarias para administrar un torneo de ajedrez. Entre las funcionalidades que presenta se encuentran: mostrar la lista de jugadores, incluyendo sus últimos ingresos al sistema, emparejamientos automáticos, cálculos de ELO,

facilita el ingreso de resultados, importación de jugadores desde archivos de texto. (Rubio Pardo, 2010)

Algunas de las características principales de este programa son:

- Trabaja en sistemas operativos de Windows.
- Fácil de usar y trabajar con el mismo, para Directores de Torneos u organizadores de los mismos.
- Se puede utilizar para sistemas: Todos contra todos y suizo.
- Sistema de Pareo de acuerdo con la FIDE.
- Presenta opciones configurables: desempates, orden de jugadores.
- Puede utilizarse con sistemas de desempate: Buchholz, Media-Buchholz, Berger, Progresivo, Número de ganadas.
- Calcula de forma automática el ELO de los participantes en base al ELO inicial.
- Se puede observar previamente: Impresión, resultados, pareos.
- Permite exportar a archivos de texto y a HTML.
- Sistema de conversión de datos: archivo en PGN.

Críticas:

Una de las principales críticas que posee este programa es que es un software propietario, por lo que para hacer uso del mismo se debe pagar un determinado importe monetario. Sus reportes son simples archivos de texto. Presenta fallas en el cálculo de los desempates. No respeta la totalidad de las reglamentaciones de la FIDE por lo que esta organización no la reconoce como una herramienta oficial. (Rubio Pardo, 2010)

1.2.2 Swiss Manager

Descripción:

Es un programa especializado en la gestión de torneos de ajedrez. Permite gestionar tanto torneos individuales como por equipos, emparejamientos por sistema suizo, ligas. Permite hacer una lista detallada de los participantes en los torneos, usando una lista consolidada de jugadores que se encuentra anexa al programa, facilitando el proceso y minimizando la posibilidad de introducir mal los nombres. Permite realizar los emparejamientos tanto manuales como automáticos. Es capaz de generar tablas con los resultados, clasificaciones variaciones de rating, tarjetas individuales de los jugadores, así como los cálculos de desempates.

Es muy completo y confiable genera toda clase de informes incluidos los reportes para la FIDE, para normas de títulos, cálculo del ELO, además genera los archivos vacíos de los pareos de cada ronda y permite subir la información del torneo en curso y las partidas a Internet a la página chess–results diseñada por su programador de una manera muy sencilla, en dicha página automáticamente se generan gran cantidad de informes sobre el torneo (resultados, pareos, partidas, estadísticas). Posee un gran número de sistemas de desempates. (Morales Zorrilla, 2009)

Este software también posibilita mostrar un listado de premios, así como, clasificaciones de jugadores por tableros en torneos por equipos. Swiss Manager ha sido el software de gestión de torneo y emparejamientos utilizado en las últimas Olimpiadas de Ajedrez (Bled 2002, Calviá 2004, Turin 2006 y Dresde 2008).

Críticas:

Es un software propietario que cuenta con tres versiones, una versión demo que es gratuita y no tiene limitación de tiempo, pero si tiene límite de jugadores y rondas. Las otras dos poseen un poco más de posibilidades, pero con un precio ya definido. (Morales Zorrilla, 2009)

Al ser un software propietario no es posible hacerle modificaciones, ni mejoras, según las necesidades que puedan surgir en algún torneo que se celebre en nuestra universidad.

1.2.3 ¿Por qué crear una aplicación web que realice estas funciones?

La razón fundamental de esta propuesta es dotar a la cátedra de ajedrez de la universidad de una aplicación de software libre capaz de realizar las mismas funciones que estas herramientas y además que se le puedan agregar funcionalidades de acuerdo con las necesidades que surjan. Esta propuesta estará soportada sobre una plataforma web la cual permitirá el acceso desde cualquier estación de trabajo de la UCI que esté conectada a la red, por lo que la información de los torneos estará centralizada, todo esto sin necesidad de instalar ninguna herramienta.

1.3 Metodología de desarrollo de software

Todo proyecto de software, en especial los que son de gran envergadura necesitan de la utilización de una determina metodología para su realización, ya que la ausencia de la misma pudiera provocar la insatisfacción de los clientes por el producto final. A la hora de determinar que metodología se va a utilizar, se debe seleccionar la más adecuada a las características del

software que se quiere desarrollar y que sirva de guía para poder desarrollar el producto de forma satisfactoria.

1.3.1 Proceso Unificado de Desarrollo (RUP)

El Proceso Unificado de Desarrollo se encuentra dentro del grupo de las denominadas metodologías robustas, la cual presenta un marco de desarrollo que sirve de guía para el proceso de elaboración de software.

RUP es una forma disciplinada de asignar tareas y responsabilidades a un equipo de desarrollo (quién hace qué, cómo y cuándo), el cual se divide en cuatro fases según la siguiente figura. (Mendoza Sánchez, 2004)

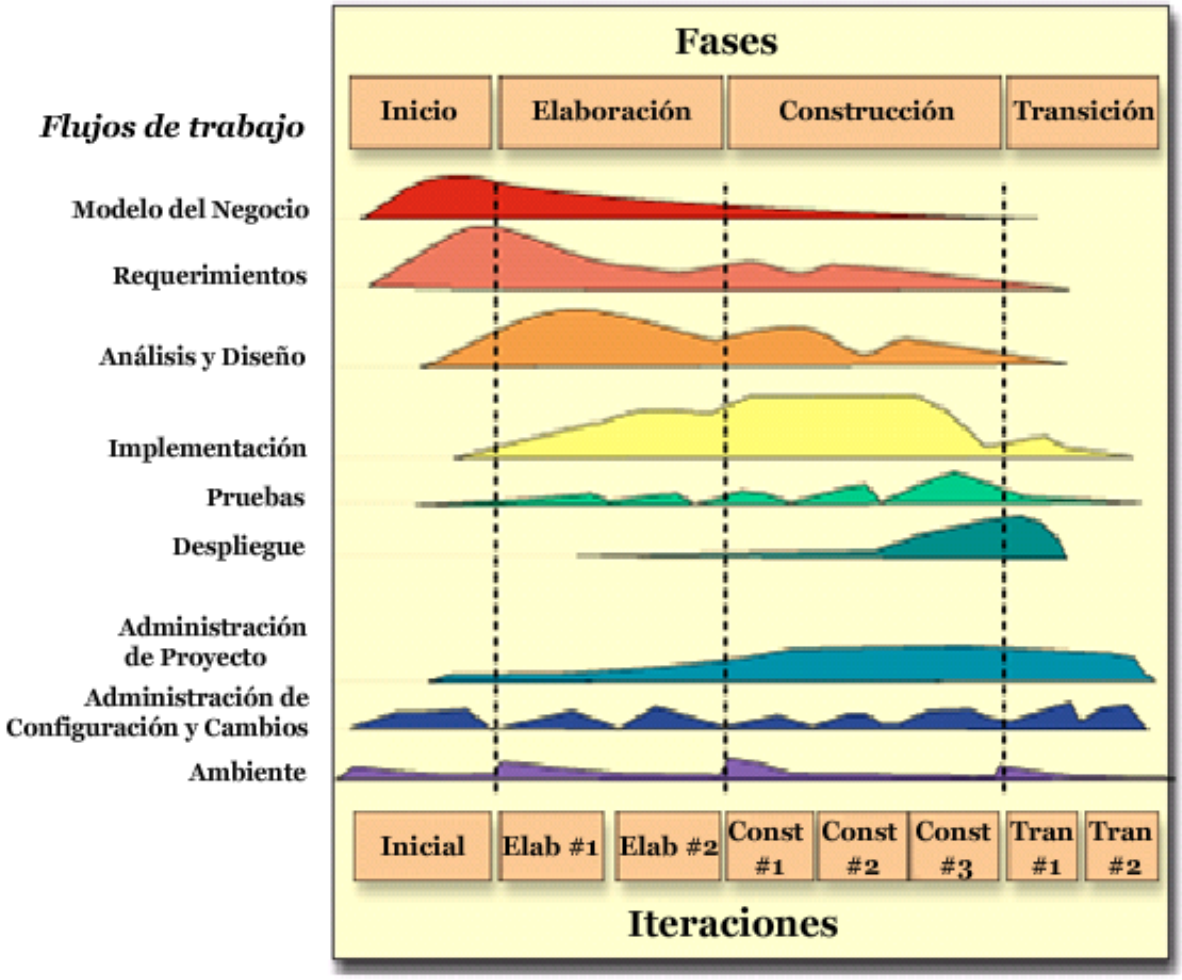


Figura 1: RUP en dos dimensiones.

El objetivo de cada una de estas fases es:

Inicio: Determinar la visión del proyecto y es donde se decide si se realiza, o no.

Elaboración: Determinar la arquitectura óptima del sistema.

Construcción: Llegar a obtener la capacidad operacional inicial del sistema.

Transición: Realizar el empaquetamiento e instalación del producto.

Cada una de estas etapas es desarrollada mediante un ciclo de iteraciones, la cual consiste en reproducir el ciclo de vida en cascada a menor escala. Los Objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes.

El ciclo de vida que se desarrolla por cada iteración es llevada a cabo bajo los diferentes flujos de trabajo que presentan cada una de ellas, los cuales se clasifican en dos tipos:

Flujos de trabajo de Ingeniería

- Modelamiento del negocio: Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
- Requerimientos: Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
- Análisis y Diseño: Describe cómo el sistema será realizado a partir de la funcionalidad prevista y las restricciones impuestas (requerimientos), por lo que indica con precisión lo que se debe programar.
- Implementación: Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación.
- Prueba: Busca los defectos a lo largo del ciclo de vida.
- Despliegue: Se produce la liberación del producto y se realizan actividades (empaque, instalación, asistencia a usuarios) para entregar el software a los usuarios finales.

Flujos de trabajo de apoyo

- Administración de proyecto: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.

- Administración de configuración y cambios: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos, control de versiones.
- Ambiente: Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto, así como el procedimiento para implementar el proceso en una organización.

RUP es una metodología de desarrollo de software que intenta integrar todos los aspectos a tener en cuenta durante todo el ciclo de vida del software, con el objetivo de hacer abarcables tanto pequeños como grandes proyectos software.

Las características principales de RUP son:

- Manejado por casos de uso: Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, son fragmentos de funcionalidad del sistema, que proporcionan al usuario un resultado importante. Representan requerimientos funcionales. Los casos de uso guían la arquitectura del sistema y la arquitectura del sistema influye en la selección de los casos de uso.
- Centrado en arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente. RUP se desarrolla mediante iteraciones, comenzando por los casos de uso relevantes desde el punto de vista de la arquitectura. La arquitectura se refleja en los casos de uso, pues cada producto tiene tanto una función como una forma, ninguna es suficiente por sí sola.
- Iterativo e Incremental: RUP propone dividir el trabajo en partes más pequeñas o mini proyectos, donde cada mini proyecto es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en los flujos de trabajo, y los incrementos, al crecimiento del producto. Cada iteración se realiza de forma planificada es por eso que se dice que son mini proyectos. Una iteración es una secuencia de actividades con un plan establecido y criterios de evaluación las cuales están estrechamente relacionadas, cuyo resultado es una versión del software.

1.3.2 ¿Por qué se escoge a RUP como la metodología a utilizar?

Se selecciona esta metodología principalmente por ser la que tenía definida el Proyecto Infodrez para la realización de los módulos del proyecto, con el objetivo de lograr un estándar entre ellos. Además, se exponen otras razones importantes que justifican la utilización de la misma en este trabajo:

- Es una metodología completa y extensa que intenta abarcar todos los aspectos del desarrollo de un software, aunque principalmente está orientada para su utilización en proyectos de gran envergadura.
- Es aplicable tanto a pequeños proyectos (como el del presente trabajo de diploma), así como para grandes proyectos de varios años de duración.
- Existe gran cantidad de documentación acerca de esta metodología tanto en libros, Internet, como en la propia universidad.
- Al decidir que el proceso de investigación fuese iterativo e incremental se pueden ir obteniendo versiones de cada iteración consiguiendo que se minimicen los riesgos al momento de implementar la aplicación.
- Es un proceso que define de manera ordenada las tareas. Es una guía para utilizar Lenguaje Unificado de Modelado (UML).

1.4 Lenguaje de Modelado Unificado (UML 2.0)

Este lenguaje de modelado de sistemas de software es el más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Ofrece un estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos de negocio y funciones del mismo, además de aspectos concretos como expresiones de lenguajes de programación, esquemas de base de datos y componentes reutilizables. (Gracia, 2005)

UML se aplica en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software como pude ser RUP, pero no especifica que metodología o proceso usar. El desarrollo de sistemas con este lenguaje siguiendo el proceso unificado incluye actividades específicas las cuales sirven como guía de cómo deben ser estas desarrolladas y secuenciadas con el fin de obtener sistemas más exitosos.

1.5 Herramienta CASE Visual Paradigm 6.0

Las herramientas CASE son utilizadas con el fin de automatizar los aspectos clave de todo un proceso de desarrollo de software de un sistema, desde su inicio, hasta completar todo el proceso. Visual Paradigm es una herramienta CASE que ofrece un entorno de creación de diagramas para UML, posee un diseño centrado en casos de uso y enfocado al negocio que ayuda a crear un software de mayor calidad. (Sierra, 2007)

Ofrece un lenguaje estándar, común a todo el equipo de desarrollo, facilitando la comunicación entre ellos. Modelo y código que permanece sincronizado en todo el ciclo de desarrollo; disponibilidad de múltiples versiones, para cada necesidad. Capacidad de integrarse en los principales IDEs (Entornos de Desarrollo Integrados). Es multiplataforma, y muy útil para la generación de código fuente en PHP.

Ventajas:

- Navegación intuitiva entre el modelo visual y el código.
- Poderosa herramienta de generación de PDF/HTML a partir de diagramas UML.
- Sincronización entre el código fuente y el modelo en tiempo real o bajo demanda.
- Posee un entorno visual de modelado avanzado.
- Posee soporte para toda la notación UML.
- Incluye sofisticados y automáticos diagramas de capas.
- Permite análisis de textos.
- Cuenta con un diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Tiene capacidades de ingeniería directa e inversa.

1.6 Solución a través de una aplicación web

Una aplicación web es aquella que los usuarios utilizan para acceder a servidores web a través del internet o de una intranet. (Cobo, y otros, 2005)

Las mismas tienen gran aceptación debido a que utilizan navegadores como clientes ligeros, además de actualizarlas y mantenerlas sin necesidad de distribuir e instalar software a miles de usuarios. Una página web puede contener elementos que permiten una comunicación activa

entre el usuario y la información. Esto permite que el usuario acceda a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos y acceder a gestores de base de datos de todo tipo.

1.6.1 Arquitectura Cliente-Servidor

Todos los servicios que ofrece Internet, entre ellos los servicios web, se basan en la arquitectura cliente/servidor. Esta arquitectura está dividida en dos partes fundamentales, la parte del servidor y un conjunto de clientes. Generalmente el servidor es una computadora muy potente que ofrece sus servicios al resto de los equipos conectados, y es donde están alojadas las páginas web. Los clientes son estaciones de trabajo que solicitan varios servicios al servidor a través de una red conectada al mismo. Entre las ventajas que presenta esta arquitectura se puede decir que el servidor no necesita tanta potencia de procesamiento, parte del proceso se reparte con los clientes.

1.6.2 Técnicas y tecnologías del lado del cliente

Las técnicas y tecnologías del lado del cliente son aquellas que se ejecutan en el navegador del usuario que pueden ser por ejemplo: Internet Explorer, Mozilla Firefox, Opera, entre otros. Esto permite que algunos procesos se ejecuten en las máquinas de los clientes y así no se sobrecarga el servidor, además de agilizar las conexiones con los mismos. Los usuarios pueden visualizar toda la información ya que los navegadores son capaces de interpretar las órdenes recibidas en forma de código HTML fundamentalmente y convertirlas en páginas que son el resultado de dicha orden.

1.6.3 Lenguajes del lado del cliente a utilizar

1.6.3.1 Hyper Text Markup Language (HTML 4.0)

HTML es la abreviatura de "Hyper Text Markup Language", es decir, "Lenguaje de marcado hipertextual", es el lenguaje con el que se definen las páginas web. Se trata de un conjunto de etiquetas que sirven para definir el texto y otros elementos que compondrán una página web. (Alvarez, 2004)

Este lenguaje de etiquetas además de mostrar texto, también permite visualizar gráficos, videos, audio, así como vínculos hacia otras partes de la misma página u otros sitios de la red. Es el encargado de decirle al navegador dónde ubicar cada texto, imágenes, videos y la forma

que van a tener estos. Entre las muchas ventajas que presenta HTML, se encuentra, que son posibles de crear y visualizar en cualquier sistema operativo.

Un documento HTML puede ser creado con la ayuda de un simple editor de texto, como el Bloc de Notas del sistema operativo Microsoft Windows. Sin embargo, también hay herramientas profesionales que nos permiten crear las páginas en un entorno gráfico, como son el caso de Dreamweaver, FrontPage.

1.6.3.2 Javascript

Se presenta como un lenguaje de desarrollo de aplicaciones cliente/servidor a través de Internet. El programa en Javascript está insertado dentro del documento HTML por lo que no es un programa aparte. Fue creado con el objetivo de darle más dinamismo a las páginas web, ya que permite una verdadera interactividad con los usuarios.

Este lenguaje evita tener que depender del servidor a la hora de hacer cálculos sencillos ya que estos son realizados en el navegador del cliente; captura los eventos realizados por el usuario y responde a los mismos sin que la información salga a la red; comprueba los datos que el usuario introduce en un formulario antes de enviarlos al servidor.

Ventajas que presenta este lenguaje:

- No requiere tiempo de compilación.
- Los scripts pueden desarrollarse en un período de tiempo relativamente corto.
- Posee características de interfaz, que son gestionados por el navegador y por el código HTML.
- Los programas Javascript tienden a ser pequeños y compactos, no requieren mucha memoria ni tiempo adicional de transmisión.
- Es independiente de la plataforma de hardware o sistema operativo, siempre y cuando exista un navegador con soporte Javascript.

1.6.3.3 Hojas de Estilo en Cascada (CSS)

Las hojas de estilo son bloques de código escrito en lenguaje HTML que se encargan de la presentación del documento, que puede abarcar desde los detalles relacionados con, cómo se muestra el documento en la pantalla (fuentes, tamaño de fuentes, interlineado). (Van Lancker, 2007)

Las CSS cubren lo relacionado a todos los aspectos de la presentación, y el papel del lenguaje HTML se limitaría a la estructura y la codificación de la información. Este lenguaje te permite descubrir nuevas posibilidades de presentación que no son posibles con HTML como son: tamaño de letra ilimitado, nuevas formas de bordes, la coloración precisa de las imágenes.

Las ventajas de utilizar CSS son:

- Obliga a crear documentos HTML/XHTML bien definidos y con significado completo.
- Mejora la accesibilidad del documento.
- Reduce la complejidad de su mantenimiento.
- Permite visualizar el mismo documento en infinidad de dispositivos diferentes.
- Evita el uso excesivo de imágenes.
- Incrementa la velocidad de transmisión de la información de los contenidos.
- Reduce el tiempo de renderización, ofreciendo al cliente una conexión más rápida y eficiente.

1.7 Tecnologías del lado del servidor

Las tecnologías del lado del servidor son aquellos programas o servicios que corren en un servidor remoto y que brindan funcionalidades al sistema. El código fuente permanece en el servidor, se conserva su privacidad y los clientes no tienen acceso al mismo. (Cobo, y otros, 2005)

1.7.1.1 Hypertext Preprocessor (PHP 5.0)

Lenguaje interpretado, de alto nivel, embebido en páginas HTML y que es ejecutado en el servidor. Es usado para la creación de aplicaciones para servidores, o para la creación de contenidos dinámicos para sitios web. Además, PHP permite la conexión de estas aplicaciones web a diferentes tipos de bases de datos como son el caso de MySQL, Postgres, Oracle, Microsoft SQL Server, entre otras; y así lograr que éstas sean más robustas.

Características:

- Velocidad**:** Posee gran velocidad y no crea demoras en la máquina, por esta razón no requiere demasiados recursos del sistema.

- Estabilidad**:** Utiliza su propio sistema de administración de recursos y dispone de un sofisticado método de manejo de variables, conformando un sistema robusto y estable.
- Seguridad**:** El sistema debe poseer protecciones contra ataques, por lo que PHP provee diferentes niveles de seguridad, estos pueden ser configurados desde el archivo *.ini.
- Simplicidad**:** Fácil de aprender sin dificultades en especial a los programadores con experiencia en los lenguajes de programación C y C++.

Ventajas de PHP:

- Este lenguaje es capaz de correr en la mayoría de las plataformas utilizando el mismo código fuente, ya sean plataformas Unix o Windows.
- Se puede ejecutar bajo diferentes servidores de aplicaciones web como son: Apache, IIS, AOLServer, Roxen y THTTPD.
- Puede interactuar con muchos motores de bases de datos tales como MS SQL, Oracle, Informix, PostgreSQL, y otros muchos.
- Generalmente es utilizado como módulo de Apache, lo que lo hace extremadamente veloz. Está completamente escrito en el lenguaje C, así que se ejecuta rápidamente utilizando poca memoria.
- Es Código Abierto (Open Source), lo cual significa que el usuario no depende de una compañía específica para arreglar problemas de funcionamiento, y no está forzado a pagar actualizaciones anuales para tener una versión que funcione.

1.8 Gestor de base de datos

Conjunto de programas que administran y gestionan la información contenida en una base de datos. Es el encargado de la privacidad, la integridad, la seguridad de los datos y la interacción con el sistema operativo. Proporciona una interfaz entre los datos, los programas que los manejan y los usuarios finales. Controla todas las operaciones realizadas por el usuario en la base de datos. (Alvarez, 2007)

1.8.1 MySQL 5.0.51a

Permite la gestión de los datos de una base de datos relacional usando un lenguaje de consulta estructurado (SQL). Mediante estas consultas, MySQL llevará a cabo una determinada acción sobre la base de datos. (Martín, 2006)

Es una aplicación de código abierto hasta esta versión, permite redistribuir una aplicación que la contenga y modificar su código para mejorarla o adaptarla a nuestras necesidades. Es un sistema fácil de instalar y configurar ya sea en servidores de Windows o Linux.

Características de MySQL:

- Acceso a las bases de datos de forma simultánea por varios usuarios y/o aplicaciones.
- Seguridad, en forma de permisos y privilegios, determinados usuarios tendrán permiso para consultar o modificar determinadas tablas. Esto permite compartir datos sin que peligre la integridad y la disponibilidad de estos.
- Potencia: SQL es un lenguaje muy potente para consulta de bases de datos.
- Portabilidad: Las consultas hechas usando SQL son fácilmente portables a otros sistemas y plataformas, debido a la estandarización que posee este lenguaje.
- Escalabilidad: es posible manipular bases de datos enormes, en torno a seis mil tablas y alrededor de cincuenta millones de registros, y hasta 32 índices por tabla.
- Conectividad: permite conexiones entre diferentes máquinas con distintos sistemas operativos. Es común que servidores Linux o Unix, usando MySQL, sirvan datos para ordenadores con sistemas operativos Windows, Linux, Solaris.
- Permite manejar registros de longitud fija o variable.

1.9 Servidor de Aplicaciones Web Apache 2.2.4

Un servidor de aplicaciones web es el encargado de recibir las peticiones emitidas por los navegadores, atienden únicamente los aspectos del cliente HTML, Javascript y Protocolo de transferencia de hipertexto (HTTP). Apache es un servidor web HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1. Es fiable, eficiente y fácilmente extensible. Es el más usado en todo el mundo, incluso más que todos los demás servidores juntos. (Márquez Díaz, y otros, 2002)

Ventajas de Apache:

- Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.
- Apache es una tecnología gratuita, de código abierto, licencia del tipo Distribución de

Software Berkeley (BSD), permite el uso del código fuente en software propietario.

- Apache es un servidor altamente configurable, de diseño modular. Es muy sencillo ampliar las capacidades del servidor Web Apache. Actualmente existen muchos módulos para Apache que son adaptables a este.
- Te permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurarlo para que ejecute un determinado script cuando ocurra un error en concreto.
- Es robusto y seguro.

1.10 Herramienta de Desarrollo Zend Studio 5.5.0

Es un completo entorno de desarrollo integrado para el lenguaje de programación PHP. Está escrito en Java, y está disponible para las plataformas Microsoft Windows, Mac OS X y GNU/Linux. Es una de las herramientas más potentes de programación para el lenguaje PHP. Este programa además de servir como editor de texto para páginas PHP, proporciona unas series de ayudas desde la creación y gestión de proyectos hasta la depuración de código. Está escrito en lenguaje de programación Java, por lo que a veces no funciona tan rápido como otras aplicaciones.

Características:

- No requiere la instalación previa de PHP ni del entorno de ejecución de Java.
- Soporte para PHP 4 y PHP 5.
- Resaltado de sintaxis, auto completado de código, ayuda de código y lista de parámetros de funciones y métodos de clase.
- Plegado de código (comentarios, cuerpo de funciones y métodos e implementación de clases).
- Inserción automática de paréntesis y corchetes de cierre.
- Sangrado automático y otras ayudas de formato de código.
- Emparejamiento de paréntesis y corchetes (si se sitúa el cursor sobre un paréntesis o corchete de apertura, Zend Studio localiza el correspondiente cierre del mismo).
- Detección de errores de sintaxis en tiempo real.

- Funciones de depuración: Botón de ejecución y traza, marcadores, puntos de parada (breakpoints), seguimiento de variables y mensajes de error del intérprete de PHP. Permite también la depuración en servidores remotos (requiere plataforma Zend Studio).
- Instalación de barras de herramientas para Internet Explorer y Mozilla Firefox.
- Soporte para gestión de grandes proyectos de desarrollo.
- Manual de PHP integrado.
- Soporte para navegación en bases de datos y ejecución de consultas SQL.

1.11 Arquitectura 4-Capas

La arquitectura por capas es un estilo de programación en el que el objetivo primordial es la separación de la lógica del negocio, de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario. (Moliner López, 2005)

Con la utilización de este estilo de desarrollo se puede llevar a cabo la programación en varios niveles, en caso de que surja algún cambio, sólo se revisa el nivel requerido sin tener que analizar todo el código. Además, permite distribuir el trabajo de creación de una aplicación por niveles; de esta forma, cada grupo de trabajo está totalmente abstraído del resto de niveles, solo basta con conocer las conexiones que existe entre las capas.

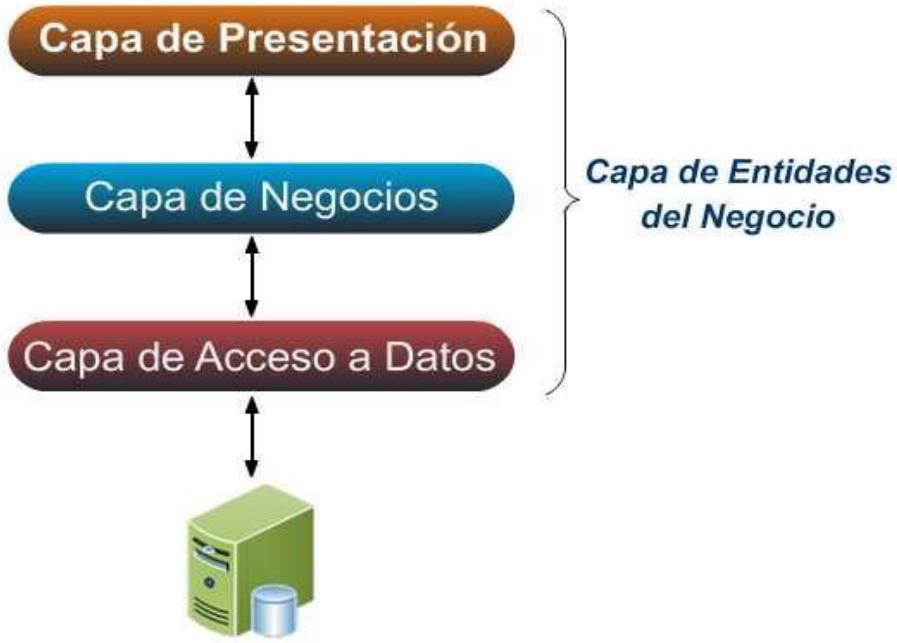


Figura 2: Estructura de la arquitectura 4-capas.

Dentro de los beneficios que presenta la arquitectura n-capas se encuentran:

- Escalabilidad**:** Habilidad de adicionar recursos para soportar mayores números de usuarios sin modificar las aplicaciones.
- Extensibilidad**:** Incrementa la funcionalidad de una aplicación sin alterar la ya existente.
- Seguridad**:** Capacidad de extender y propagar la autenticación en la capa de interfaz hacia las capas intermedias, salvaguardando la información de los niveles de datos.
- Administrabilidad**:** Fácil de administrar.
- Permite que varias interfaces diferentes puedan acceder a un mismo recurso, facilitando el trabajo del programador.

Dentro de las arquitecturas n-capas la más utilizada es la de 4-capas, que presenta una capa más que la de 3-capas, que es la que separa definitivamente la capa del negocio de la de acceso a datos. Donde cada capa se especializa absolutamente en la funcionalidad que deben brindar, ya sean las encargadas de las reglas del negocio o las encargadas de mostrar la información al usuario. Esta arquitectura les permite a los programadores la posibilidad de reutilizar el código, agilizando así el proceso de implementación. Como estas aplicaciones se

realizan en unidades separadas, permite realizarles pruebas por separado a cada una de ellas con muchos más detalles, logrando así un producto más sólido.

1.12 Conclusiones

En este capítulo se realizó un estudio de las herramientas existentes en el mundo para la gestión de torneos de ajedrez y se analizó el porqué de la propuesta planteada. Además, se llevó a cabo un estudio de las herramientas y metodologías que serán usadas en la propuesta, las cuales ya estaban definidas por el proyecto Infodrez, y forman parte de la versión 1.0 de Arbitraje. Con el desarrollo de este estudio se logró conocer las características de herramientas y metodologías, con el fin de hacer un mejor uso de las mismas a la hora de trabajar sobre este proyecto de software.

Capitulo 2. Características del sistema

2.1 Introducción

En este capítulo se tratarán las características generales que presentará el sistema, además de que se explica el funcionamiento general de la aplicación. También se transitará por la fase de inicio que propone la metodología que se está utilizando, en este caso, RUP. Se abordará desde el modelo del negocio hasta el sistema, que son los flujos de trabajos más significativos en la fase inicial que propone el Proceso Unificado de Desarrollo. Todo esto se desarrollará por medio de un conjunto de artefactos y modelos resultantes de la aplicación de la metodología de desarrollo de software.

2.2 Propuesta del sistema

Se desea crear una aplicación web, que permita controlar toda la información generada en los torneos de ajedrez celebrados en la UCI. En la misma se podrán autenticar los árbitros que estén al frente de cada evento ajedrecístico, el cual tendrá la posibilidad de controlar y gestionar los datos referentes a cada torneo que esté arbitrando. Cada torneo presenta diferentes particularidades en dependencia del tipo que sean. Los árbitros crean los torneos con todos sus datos, después de estar creado, agrega los jugadores que participarán en el mismo, podrá realizar el pareo correspondiente al tipo de torneo que haya creado. Este pareo representa los jugadores y su oponente en cada ronda. El sistema debe permitir mostrar el cuadro sinóptico del torneo, así como los resultados individuales de cada jugador.

Esta propuesta será capaz de realizar las mismas funciones que poseen herramientas similares como son el caso de Swiss Manager y el Swiss Perfect, que son software propietario. La aplicación al ser de tecnología web se podrá acceder a ella fácilmente a través de un simple navegador, sin necesidad de instalar un programa como es el caso de los mencionados anteriormente. Además, a esta herramienta se le podrán agregar funcionalidades especificadas por el presidente de la cátedra de ajedrez de la UCI.

2.3 Modelamiento del Negocio

El modelo del negocio es el primer flujo de trabajo que propone la metodología de desarrollo RUP, en el cual se describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización. En el trabajo de diploma desarrollado en cursos anteriores titulado “Propuesta de Sistemas Integrados del proyecto Infodrez” se encuentra

desarrollado el modelo del negocio correspondiente a esta aplicación, por lo cual no se hace necesario que se describa en este documento.

2.4 Requerimientos

Después de la realización del modelo de negocio se procede a ejecutar el proceso de captura de requisitos del sistema. Los requisitos son en sí, lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.

2.4.1 Requerimientos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir. Especifican acciones que el sistema debe ser capaz de realizar, sin tomar en consideración ningún tipo de restricción física. No alteran la funcionalidad del producto, lo cual quiere decir que estos se mantienen invariables sin importarle con que propiedades o cualidades se relacionen. Los requisitos funcionales que debe cumplir la aplicación fueron capturados en el trabajo de diploma “Propuesta de Sistemas Integrados del proyecto Infodrez”, en el cual se especificaron 37 requerimientos. Para el desarrollo de la versión que se quiere desarrollar fue necesario agregar algunos requisitos más, a los que ya existían, que son en este caso:

RF 38 Gestionar estructura de matches individuales.

33.1 Crear estructura de matches individuales.

33.2 Mostrar estructura de matches individuales.

33.3 Eliminar estructura de matches individuales.

RF 39 Ver resultados por ronda.

2.4.2 Requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Estas características son las que hacen al producto atractivo, usable y confiable. Normalmente, están vinculados a los requerimientos funcionales, o sea, una vez que se conozca lo que el sistema debe hacer podemos determinar cómo ha de comportarse, las cualidades que debe tener o cuán rápido o grande debe ser. Los mismos fueron especificados en el trabajo de diploma “Propuesta de Sistemas Integrados del proyecto Infodrez” desarrollado en cursos anteriores.

2.5 Modelo del sistema

Una vez que se identificaron y se clasificaron los requerimientos del sistema, es necesario identificar los actores y casos de uso. Los casos de uso son priorizados y detallados, realizando la descripción de cada uno de ellos. Este modelo del sistema fue modificado a partir del modelo del sistema desarrollado en el trabajo de diploma “Propuesta de Sistemas Integrados del proyecto Infodrez” ya que se hacía necesario para desarrollar esta versión.

2.5.1 Actores del sistema

El actor del sistema es un individuo, grupo de individuos, sistema automatizado o máquina que interactúa con este, intercambiando información. Cada trabajador del negocio o en el caso de que fuera un sistema ya existente que tienen actividades a automatizar, es un candidato a actor del sistema. Todo actor del negocio que interactúe con el sistema, también será un actor del mismo.

Descripción de los actores del sistema.

Tabla 1: Actores del sistema

Actor del sistema	Descripción
Usuario	Puede ser un Árbitro o un Administrador, el cual es el que se autentica en la aplicación.
Árbitro	Es el encargado de velar que se cumplan las leyes del ajedrez y realizar todo el proceso de arbitraje en un torneo. Es el encargado de gestionar toda la información que genere un torneo, insertar jugadores, parear jugadores, insertar resultados. Sólo podrá acceder a los torneos que ha creado el propio árbitro.
Administrador	Gestiona toda la información manejada en el sistema, relacionado con la gestión de los usuarios y los árbitros. Es el encargado de introducir los datos de los usuarios o árbitros nuevos en la aplicación.

2.5.2 Diagrama de casos de uso del sistema

Un diagrama de casos de uso del sistema representa gráficamente a los procesos y su interacción con los actores. Los casos de uso son artefactos que describen, mediante acciones y reacciones, el comportamiento que tendrá el sistema desde el punto de vista del usuario.

Establece un acuerdo entre los clientes y los desarrolladores en cuanto a las condiciones y posibilidades que debe cumplir el sistema.

En este diagrama se representan 32 casos de uso identificados en el trabajo de diploma anteriormente mencionado que son necesarios para la segunda versión, además se le agregaron casos de uso nuevos imprescindibles para este trabajo. Los casos de uso incorporados son:

- Gestionar estructura de matches individuales.
    - Crear estructura de matches individuales.
    - Mostrar estructura de matches individuales.
    - Eliminar estructura de matches individuales.
- Ver resultados por ronda.

Diagrama de casos de uso del sistema:

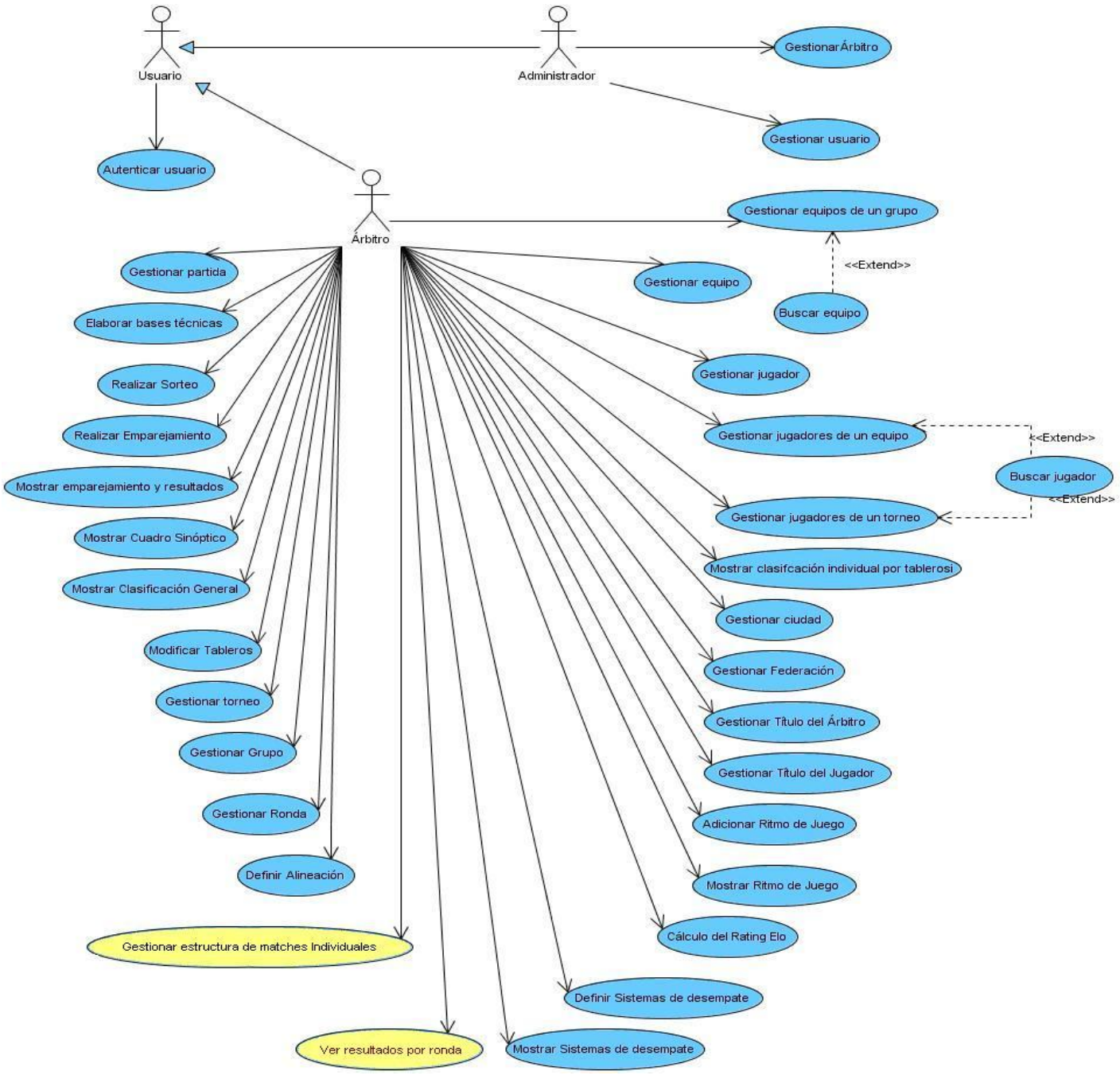


Figura 3: Diagrama de casos de uso del sistema.

2.5.3 Descripción de los casos de uso del sistema

La descripción de los casos de uso del sistema se realiza con el objetivo de entender la funcionalidad asociada a cada uno de ellos. Esta debe ser elaborada de forma breve o extendida para lograr una mejor comprensión sobre lo que debe realizar el sistema y ayudar a

un mejor entendimiento. Expresa de forma clara y precisa las acciones que se realizan durante la interacción entre el actor y el sistema, describe el flujo de actividades que realiza el actor al hacer uso del sistema y las respuestas que emite el mismo.

A continuación se muestran algunas descripciones de casos de uso críticos del sistema, los demás se pueden consultar en los Anexos.

Tabla 2: Descripción caso de uso " Gestionar Árbitro"

Caso de Uso:	Gestionar Árbitro
Actores:	Administrador
Propósito:	Permite al administrador gestionar toda la información referente a los árbitros: adicionar, ver, modificar o eliminar árbitros en el sistema.
Resumen:	El caso de uso se inicia cuando el administrador dentro de las opciones de administración va a tener la posibilidad de adicionar, modificar o eliminar árbitros.
Precondiciones:	
Poscondiciones:	-Árbitro adicionado a la base de datos. - Árbitro eliminado de la base de datos. -Datos del Árbitro actualizados.
Referencias	RF 3, RF 3.1, RF 3.2, RF 3.3, RF 3.4
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1-El administrador dentro de la página principal escoge la opción de Gestionar Árbitro del Menú de Administración	1.1- El sistema brinda las opciones de: -Adicionar Árbitro -Ver Árbitro -Modificar Árbitro. -Eliminar Árbitro.
2 -El administrador selecciona una de las opciones	2.1- Si selecciona “Adicionar Árbitro” ver sección **Adicionar Árbitro** -Si selecciona “Ver Árbitro” ver sección **Ver Árbitro** -Si selecciona “Modificar Árbitro” ver sección

	Modificar Árbitro. -Si selecciona “Eliminar Árbitro” ver sección **Eliminar Árbitro.**
Sección: “Adicionar Árbitro”	
Acción del Actor	**Respuesta del Sistema**
1-El administrador selecciona la opción de Adiciona Árbitro.	1.1-El sistema muestra el formulario con los siguientes campos a llenar: -Nombre -Apellidos -Rol -Título
2- El administrador llena lo campos necesarios pulsa el botón Aceptar.	2.1-El sistema verifica que los datos introducidos son correctos.
	2.2- El sistema verifica que el árbitro no esté registrado, en caso positivo lo registra en la base de datos y muestra un mensaje evidenciando el registro del nuevo árbitro, concluyendo con esto el caso de uso del sistema.
Flujo Alterno de Eventos	
Sección: “Adicionar Árbitro”	
	2.1-Si los datos introducidos no son correctos el sistema muestra un mensaje informando de que los datos no son válidos. Se procede al flujo normal de eventos a partir del paso 2.
	2.2- Si el árbitro está registrado el sistema muestra un mensaje informando la situación.
Sección: “Ver Árbitro”	
Acción del Actor	**Respuesta del Sistema**
1-El administrador selecciona el árbitro.	1.1-El sistema muestra los datos del árbitro, termina así el caso de uso.
Sección: “Modificar Árbitro”	

Acción del Actor	**Respuesta del Sistema**
1- El administrador selecciona la opción de Modificar Árbitro.	1.1-El sistema muestra un listado de todos los árbitros registrados.
2- El administrador selecciona el árbitro que desea y pulsa el botón Aceptar.	2.1-El sistema muestra un formulario con los datos del árbitro que se desea modificar.
3- El administrador modifica los datos necesarios y pulsa el botón Aceptar.	3.1- El sistema verifica que los datos modificados son válidos, en caso positivo estos se actualizan en la base de datos, concluyendo así el caso de uso del sistema.
Flujo Alterno de Eventos	
Sección: "Modificar Árbitro"	
	3.1-Si los datos no son correctos, éste muestra un mensaje indicando en que parte del formulario fue donde estuvo el error. Se procede al flujo normal de eventos desde la acción 3.
Sección: "Eliminar Árbitro "	
Acción del Actor	**Respuesta del Sistema**
1-El administrador selecciona la opción de Eliminar Árbitro.	1.1- El sistema muestra un listado con los árbitros registrados.
2- El administrador selecciona el árbitro que desea eliminar del sistema y pulsa el botón Aceptar.	2.1- El sistema elimina el árbitro de la base de datos y muestra un mensaje de que éste ha sido eliminado, terminando así el caso de uso.

Tabla 3: Descripción caso de uso "Gestionar Jugador"

Caso de Uso:	Gestionar Jugador
Actores:	Árbitro
Propósito:	Permite al árbitro gestionar toda la información referente a los jugadores: adicionar, ver, modificar y eliminar un jugador.
Resumen:	El caso de uso se inicia cuando el árbitro dentro del Menú de Arbitraje tiene la opción de gestionar la información de los jugadores.

Precondiciones:	
Poscondiciones:	-Jugador adicionado a la base de datos. -Jugador eliminado de la base de datos. -Datos del Jugador actualizados.
Referencias	RF 11, RF 11.1, RF 11.2, RF 11.3, RF 11.4
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro selecciona dentro de la página principal la opción Gestionar Jugador del Menú Torneo.	1.1- El sistema brinda las opciones de: - Adicionar Jugador -Ver Jugador -Modificar Jugador -Eliminar Jugador
2 -El árbitro selecciona una de las opciones.	2.1- Si selecciona “Adicionar Jugador” ver sección **Adicionar Jugador** -Si selecciona “Ver Jugador” ver sección **Ver Jugador** -Si selecciona “Modificar Jugador” ver sección **Modificar Jugador** -Si selecciona “Eliminar Jugador” ver sección **Eliminar Jugador**

Sección: “Adicionar Jugador”	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro selecciona la opción de Adicionar Jugador	1.1- El sistema muestra el formulario con todas las opciones para que el árbitro entre todos los datos que se necesitan para adicionar un jugador al sistema. -Nombre -Apellido -Sexo -Titulo

	-Código FIDE -Federación -Fecha de nacimiento -Elo -K
2- El árbitro introduce los datos necesarios para adicionar un nuevo jugador y pulsa el botón Aceptar.	2.1-El sistema verifica que los datos insertados son correctos.
	2.2-El sistema verifica que el jugador no se encuentre registrado a la base de datos, en caso afirmativo lo adiciona a la base de datos y muestra un mensaje informando de que el registro se realizó correctamente, concluyendo con esto el caso de uso del sistema.
Flujo Alterno de Eventos	
Sección: "Adicionar Jugador"	
	2.1-Si los datos introducidos son incorrectos el sistema muestra un mensaje indicando en que elemento del formulario fue donde estuvo el error. Se procede al flujo normal de eventos desde el paso 2.
	2.2-Si el jugador ya existe el sistema emite un mensaje informando la existencia del mismo.
Sección: "Ver Jugador"	
Acción del Actor	**Respuesta del Sistema**
1-El árbitro selecciona el jugador a consultar datos.	1.1-El sistema muestra los datos del jugador seleccionado, termina así el caso de uso
Sección: "Modificar Jugador "	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro selecciona el jugador que desea modificar y marca la opción de Modificar Jugador	1.1- El sistema verifica que el jugador no pertenezca a un torneo que no haya finalizado, en

	caso positivo muestra los datos del jugador a ser modificados
2- El árbitro modifica los datos y pulsa el botón Aceptar.	2.1-El sistema verifica que los datos modificados son válidos, en caso positivo estos se actualizan en la base de datos, concluyendo así el caso de uso del sistema.
Flujo Alterno de Eventos	
Sección: "Modificar Jugador "	
	1.1-Si el jugador pertenece a un torneo que no haya finalizado se muestra un mensaje informando sobre la situación
	2.1- Si los datos que se modifican son incorrectos el sistema mostrará un mensaje señalando el campo donde los datos no son válidos. Se procede al flujo normal de eventos a partir del paso 3.
Sección: "Eliminar Jugador "	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro selecciona el jugador que desea eliminar y marca la opción de Eliminar Jugador	1.1- El sistema verifica que el jugador no pertenezca a un torneo que no haya finalizado, en caso positivo muestra un mensaje de confirmación para verificar que realmente se quiere eliminar el jugador.
2- El árbitro pulsa el Aceptar del mensaje de confirmación.	2.1- El sistema elimina el jugador de la base de datos y muestra un mensaje de que ha sido eliminado, terminando así el caso de uso.
Flujo Alterno de Eventos	
Sección: "Eliminar Jugador"	
	1.1- Si el jugador pertenece a un torneo que no haya finalizado se muestra un mensaje informando sobre la situación
2-El árbitro pulsa el Cancelar del mensaje de	

confirmación.	

Tabla 4: Descripción caso de uso "Gestionar Equipo"

Caso de Uso:	Gestionar Equipo
Actores:	Árbitro
Propósito:	Brinda la posibilidad a los árbitros de gestionar la información relacionada con los equipos: adicionar, ver, modificar o eliminar equipos.
Resumen:	El caso de uso inicia cuando el usuario selecciona la opción Equipo del Menú Arbitraje, el sistema brinda la posibilidad de gestionar los equipos.
Precondiciones:	
Poscondiciones:	-Nuevo equipo del torneo registrado en la base de datos. -Equipo del torneo eliminado de la base de datos. -Actualización de los datos de un equipo del torneo.
Referencias	RF 12, RF 12.1, RF 12.2, RF 12.3
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1-1-El árbitro selecciona dentro de la página principal la opción Gestionar Equipo del Menú Torneo.	1.1- El sistema brinda las opciones de: -Adicionar Equipo -Ver Equipo - Modificar Equipo -Eliminar Equipo
2 -El árbitro selecciona una de las opciones.	2.1- Si selecciona “Adicionar Equipo” ver sección **Adicionar Equipo** - Si selecciona “Ver Equipo” ver sección **Ver Equipo** -Si selecciona “Modificar Equipo” ver sección **Modificar Equipo** - Si selecciona “Eliminar Equipo” ver sección **Eliminar Equipo**

Sección: "Adicionar Equipo"	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro selecciona la opción de Adicionar Equipo	1.1- El sistema muestra un formulario con los datos a ser llenados: - Nombre -Federación -Logo
2- El árbitro introduce los datos para llenar los campos correspondientes, y pulsa el botón Aceptar.	2.1-El sistema verifica que los datos introducidos por el árbitro son válidos.
	2.2- El sistema verifica que el quipo no se encuentre registrado, en caso positivo se insertan los mismos en la base de datos, concluyendo así el caso de uso.
Flujo Alterno de Eventos	
Sección: "Adicionar Equipo"	
	2.1- Si los datos que se introducen son incorrectos el sistema mostrará un mensaje señalando la presencia de un error al adicionar un equipo. Se procede al flujo normal de eventos a partir de la acción 2.
	2.2-Si el equipo ya existe el sistema emite un mensaje informando su existencia.
Sección: "Ver Equipo"	
Acción del Actor	**Respuesta del Sistema**
1-El árbitro selecciona el equipo.	1.1-El sistema muestra los datos del equipo seleccionado.
Sección: "Modificar Equipo"	
Acción del Actor	**Respuesta del Sistema**
1- El usuario selecciona el equipo que desea modificar y marca la opción de Modificar	1.1-El sistema verifica que el equipo no pertenezca a un torneo que no haya finalizado, en caso

Equipo.	positivo muestra todos los datos del equipo de torneo para que sean modificados.
2- El usuario modifica los datos deseados y pulsa el botón Aceptar.	2.1- El sistema verifica que los datos modificados son válidos, en caso positivo estos se actualizan en la base de datos, concluyendo así el caso de uso del sistema.
Flujo Alterno de Eventos	
Sección: "Modificar Equipo"	
	1.2-Si el equipo pertenece a un torneo que no finalizó muestra un mensaje informando sobre la situación.
	2.1- Si los datos que se introducen son incorrectos el sistema mostrará un mensaje señalando el campo donde los datos no son válidos. Se procede al flujo normal de eventos a partir del paso 2.
Sección: "Eliminar Equipo"	
Acción del Actor	**Respuesta del Sistema**
1- El usuario selecciona el equipo que desea eliminar y marca la opción Eliminar Equipo	1.1-El sistema verifica que el equipo no pertenezca a un torneo que no haya finalizado, en caso positivo muestra un mensaje de confirmación para verificar que se quiere eliminar el equipo.
2- El usuario pulsa el botón Aceptar del mensaje de confirmación.	2.1-El sistema lo elimina de la base de datos y muestra un mensaje informando de que el equipo ha sido eliminado, terminando así el caso de uso.
Flujo Alterno de Eventos	
Sección: "Eliminar Equipo "	
	1.2-Si el equipo pertenece a un torneo que no finalizó muestra un mensaje informando sobre la situación.
2- El usuario pulsa el botón Cancelar del mensaje de confirmación.	

Tabla 5: Descripción de caso de uso "Gestionar Torneo"

Caso de Uso:	Gestionar Torneo
Actores:	Árbitro
Propósito:	Brinda la posibilidad a los árbitros de gestionar la información relacionada con los torneos: adicionar, ver, modificar, y eliminar torneos.
Resumen:	El caso de uso inicia cuando el árbitro selecciona la opción Torneo del Menú Arbitraje donde el sistema brinda la posibilidad de gestionar la información relacionada con los diferentes torneos.
Precondiciones:	
Poscondiciones:	- Torneo registrado en la base de datos. - Torneo actualizado. - Torneo eliminado de la base de datos.
Referencias	RF 13, RF 13.1, RF 13.2, RF 13.3, RF 13.4
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro selecciona dentro de la página principal la opción Gestionar Torneo del Menú Torneo.	1.1- El sistema brinda las opciones de: - Adicionar Torneo -Ver Torneo -Modificar Torneo -Eliminar Torneo
2 -El árbitro selecciona una de las opciones.	2.1- Si selecciona “Adicionar Torneo” ver sección **Adicionar Torneo.** - Si selecciona “Ver Torneo” ver sección **Ver Torneo.** -Si selecciona “Modificar Torneo” ver sección **Modificar Torneo.** -Si selecciona “Eliminar Torneo” ver sección **Eliminar Torneo.**
Sección: “Adicionar Torneo”	

Acción del Actor	**Respuesta del Sistema**
1- El árbitro selecciona la opción de Adicionar Torneo.	1.1- El sistema muestra un formulario con los datos a ser llenados: -Nombre -Fecha de inicio -Fecha final -Cantidad de grupos -Árbitro Principal
2- El Árbitro introduce los datos para llenar los campos correspondientes y pulsa el botón Aceptar.	2.1- El sistema verifica que los datos introducidos por el árbitro son válidos.
	2.2-El sistema verifica que el torneo no está registrado, en caso positivo se insertan los mismos en la base de datos y muestra un mensaje informando de que el registro se realizó correctamente concluyendo así el caso de uso.
Flujo Alterno de Eventos	
Sección: "Adicionar Torneo"	
	2.1- Si los datos que se introducen son incorrectos el sistema mostrará un mensaje señalando el campo donde los datos no son válidos. Se procede al flujo normal de eventos a partir de la acción 2.
	2.2- Si el torneo ya está registrado el sistema muestra un mensaje informando la situación.
Sección: "Ver Torneo"	
Acción del Actor	**Respuesta del Sistema**
1-El árbitro selecciona el torneo.	1.1-El sistema muestra los datos del torneo, finalizando así el caso de uso.
Sección: "Modificar Torneo"	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro selecciona el torneo que desea	1.1-El sistema verifica que el torneo no hay finalizado,

modificar y marca la opción de Modificar Torneo.	en caso positivo muestra un formulario con los datos del torneo seleccionado.
2- El árbitro modifica los datos y pulsa el botón Aceptar.	2.1- El sistema verifica que los datos modificados son válidos, en caso positivo estos se actualizan en la base de datos, termina así el caso de uso.
Flujo Alterno de Eventos	
Sección: "Modificar Torneo"	
	1.1-Si el torneo ya finalizó el sistema muestra el mensaje: "No se puede eliminar el torneo".
	2.1-Si los datos que se introducen son incorrectos el sistema mostrará un mensaje señalando el campo donde los datos no son válidos. Se procede al flujo normal de eventos a partir del paso 2.
Sección: "Eliminar Torneo"	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro selecciona el torneo que desea eliminar y marca la opción de Eliminar Torneo.	1.1-El sistema verifica que el torneo no hay finalizado, en caso positivo muestra un mensaje de confirmación para verificar que realmente se quiere eliminar el torneo.
2- El árbitro pulsa el botón Aceptar del mensaje de confirmación.	2.1- El sistema elimina el torneo de la base de datos y muestra un mensaje de que ha sido eliminado, termina así el caso de uso.
Flujo Alterno de Eventos	
Sección: "Eliminar Torneo"	
	1.2-Si el torneo ya finalizó el sistema muestra el mensaje: "No se puede eliminar el torneo".
2-El árbitro pulsa el botón Cancelar del mensaje de confirmación.	

Tabla 6: Descripción caso de uso "Realizar Sorteo"

Caso de Uso:	Realizar Sorteo	
Actores:	Árbitro	
Propósito:	El propósito de este caso de uso es realizar sorteo conociendo inicialmente la lista de jugadores o de equipos que participarán en el torneo.	
Resumen:	El caso de uso comienza cuando el árbitro escoge la opción Sorteo del Menú Torneo y brinda la posibilidad de sortear lo mismo los torneos desarrollados en Round Robin individual como por equipos.	
Precondiciones:	Debe estar creada la lista de jugadores o de equipos participantes en el torneo.	
Poscondiciones:	Cada jugador o equipo tienen un número aleatorio asignado.	
Referencias:	RF 18	
Prioridad:	Crítico	
Flujo Normal de Eventos		
Sección: "Realizar Sorteo para un torneo Round Robin individual"		
Acción del Actor		**Respuesta del Sistema**
1- El árbitro selecciona dentro de la página principal la opción Sorteo del Menú Torneo.		1.1- El sistema muestra todos los jugadores participantes en el torneo, brindando la posibilidad de que el árbitro pueda realizar el sorteo.
2- El árbitro sortea los jugadores por orden de Rating Elo y Título.		2.1-El sistema asigna de forma aleatoria un número a cada jugador, mostrando dicho número al árbitro y brindando la posibilidad de deshacer el sorteo si es necesario.
Sección: "Realizar Sorteo para un torneo Round Robin por equipos "		
Acción del Actor		**Respuesta del Sistema**
1- El árbitro selecciona la opción Sorteo del Menú Torneo.		1.1- El sistema muestra los equipos participantes en el torneo, brindando la posibilidad de que el árbitro pueda realizar el sorteo.
2- El árbitro sortea los equipos por el Elo promedio del equipo.		2.1-El sistema asigna de forma aleatoria un número a cada equipo, mostrando dicho número al árbitro y brindando la posibilidad de deshacer el sorteo si es

	necesario terminando así el caso de uso.

Tabla 7: Descripción caso de uso "Realizar emparejamiento"

Caso de Uso:	Realizar emparejamiento
Actores:	Árbitro
Propósito:	Mostrar el pareo de los torneos que se desarrollan como son Round Robin individuales y por equipos, Muerte Súbita, Suizo Individual, Suizo por Equipo y Sheveningen.
Resumen:	El caso de uso inicia cuando el árbitro selecciona la opción Pareo del menú Torneo, luego se muestra el emparejamiento de todas las rondas de un grupo de torneo.
Precondiciones:	Ya debe existir un torneo creado y debe haberse realizado el sorteo de lo jugadores o de los equipos.
Poscondiciones:	Se muestra la tabla de pareo de un grupo del torneo que contiene los enfrentamientos por ronda.
Referencias	RF 19
Prioridad	Crítico
Flujo Normal de Eventos	
Sección: "Realizar emparejamiento para un torneo Round Robin individual"	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro dentro de la página principal pulsa el botón Pareo del Menú Torneo.	1.1-El sistema verifica que estén registrados todos los jugadores que van a participar en el torneo.
	1.2-El sistema verifica que todos los jugadores participantes tengan un número de sorteo asignado.
	1.3 - El sistema aplica el algoritmo del pareo, es decir, guiado por el número de sorteo de cada jugador para el torneo busca el nombre de cada uno y construye un arreglo con todas las partidas organizadas por rondas.
	1.4-El sistema muestra todas las partidas organizadas por ronda con sus resultados en caso de que ya los

	tenga.
Flujo Alterno de Eventos	
	1.1-Si no están todos los jugadores registrados en el torneo el sistema muestra un mensaje informando la situación.
	1.2- Si existe algún participante en el torneo que no cuenta aún con un número de sorteo el sistema muestra un mensaje informando de que se debe realizar primero el sorteo para luego realizar el pareo.
Sección: "Realizar emparejamiento para un torneo Round Robin por equipos "	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro pulsa el botón Pareo del Menú Torneo.	1.1-El sistema verifica que estén registrados todos los equipos que van a participar en el torneo.
	1.2-El sistema verifica que todos los equipos participantes tengan un número de sorteo asignado.
	1.3-El sistema ejecuta el algoritmo encargado de realizar el pareo.
	1.4 - El sistema guiado por el número de sorteo de cada equipo para el torneo busca el nombre de cada uno de ellos y construye un arreglo con los enfrentamientos entre los equipos en cada una de las rondas.
	1.5-El sistema muestra todos los enfrentamientos por ronda con sus resultados en caso de que ya los tenga.
Flujo Alterno de Eventos	
	1.1-Si no están todos los equipos registrados en el torneo el sistema muestra un mensaje informando la situación.
	1.2- Si existe algún equipo en el torneo que no cuenta aún con un número de sorteo el sistema muestra un mensaje informando de que se debe realizar primero el sorteo para luego realizar el pareo.
Sección: "Realizar emparejamiento para un torneo Muerte Súbita"	

Acción del Actor	**Acción del Actor**
1- El árbitro dentro de la página principal pulsa el botón Pareo del Menú Torneo.	1.1-El sistema verifica que estén registrados todos los jugadores que van a participar en el torneo.
	1.2-El sistema verifica que todos los jugadores participantes tengan un número de sorteo asignado.
	1.3 - El sistema aplica el algoritmo del pareo, es decir, guiado por el número de sorteo de cada jugador para el torneo busca el nombre de cada uno y construye un arreglo con todas las partidas organizadas por rondas.
	1.4-El sistema muestra todas las partidas organizadas por ronda con sus resultados en caso de que ya los tenga.
Flujo Alterno de Eventos	
	1.1-Si no están todos los jugadores registrados en sistema muestra un mensaje informando la situación.
Sección: "Realizar emparejamiento para un torneo Suizo Individual "	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro dentro de la página principal pulsa el botón Pareo del Menú Torneo.	1.1-El sistema verifica que estén registrados todos los jugadores que van a participar en el torneo.
	1.2-El sistema verifica que todos los jugadores participantes tengan un número de sorteo asignado.
	1.3 - El sistema aplica el algoritmo del pareo, es decir, guiado por el número de sorteo de cada jugador para el torneo busca el nombre de cada uno y construye un arreglo con todas las partidas organizadas por rondas.
	1.4-El sistema muestra todas las partidas organizadas por ronda con sus resultados en caso de que ya los tenga.
Flujo Alterno de Eventos	
	1.1-Si no están todos los jugadores registrados en sistema muestra un mensaje informando la situación.

Sección: "Realizar emparejamiento para un torneo Suizo por equipos "	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro pulsa el botón Pareo del Menú Torneo.	1.1-El sistema verifica que estén registrados todos los equipos que van a participar en el torneo.
	1.2-El sistema verifica que todos los equipos participantes tengan un número de sorteo asignado.
	1.3-El sistema ejecuta el algoritmo encargado de realizar pareo.
	1.4 - El sistema guiado por el número de sorteo de cada equipo para el torneo busca el nombre de cada uno de ellos y construye un arreglo con los enfrentamientos entre los equipos en cada una de las rondas
	1.5-El sistema muestra todos los enfrentamientos por ronda con sus resultados en caso de que ya los tenga.
Flujo Alterno de Eventos	
	1.1-Si no están todos los equipos registrados en el torneo el sistema muestra un mensaje informando la situación.
	1.2- Si existe algún equipo en el torneo que no cuenta aún con un número de sorteo el sistema muestra un mensaje informando de que se debe realizar primero el sorteo para luego realizar el pareo.
Sección: "Realizar emparejamiento para un torneo Sheveningen"	
Acción del Actor	**Respuesta del Sistema**
1- El árbitro dentro de la página principal pulsa el botón Pareo del Menú Torneo.	1.1-El sistema verifica que estén registrados todos los jugadores que van a participar en el torneo.
	1.2-El sistema verifica que todos los jugadores participantes tengan un número de sorteo asignado.
	1.3 - El sistema aplica el algoritmo del pareo, es decir, guiado por el número de sorteo de cada jugador para el torneo busca el nombre de cada uno y construye un arreglo con todas las partidas organizadas por rondas.

	1.4-El sistema muestra todas las partidas organizadas por ronda con sus resultados en caso de que ya los tenga.
Flujo Alterno de Eventos	
	1.1-Si no están todos los jugadores registrados en sistema muestra un mensaje informando la situación.

Tabla 8: Descripción de caso de uso "Gestionar estructura del match individual"

Caso de Uso:	Gestionar estructura del match individual
Actores:	Árbitro Principal
Resumen:	Permite al árbitro gestionar toda la información referente a la estructura de matches individuales.
Precondiciones:	Ya debe estar creado el torneo, debe ser solamente para torneos muerte súbita.
Referencias	RF 33, RF 33.1, RF 33.2, RF 33.3
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Negocio**
1-El árbitro selecciona dentro del menú del torneo muerte súbita la opción de gestionar estructura de matches individuales.	1.1-El sistema brinda las opciones de: - Crear estructura de matches individuales - Mostrar estructura de matches individuales -Eliminar estructura de matches individuales
2-El árbitro selecciona una de las opciones.	2.1-Si selecciona “Crear estructura de matches individuales” ver sección **Crear estructura de matches individuales.** -Si selecciona “Mostrar estructura de matches individuales” ver sección **Mostrar estructura de matches individuales.** -Si selecciona “Eliminar estructura de matches individuales” ver sección **Eliminar estructura**

	de matches individuales.
Sección: "Crear estructura de matches individuales"	
Acción del Actor	**Respuesta del Sistema**
1-El árbitro selecciona la opción de Crear la estructura de matches individuales.	1.1-El sistema muestra un formulario con las opciones para crear la estructura de matches individuales, que son los matches que tendrá el torneo y el respectivo ritmo de juego.
2- El árbitro selecciona los matches que tendrá el torneo, el ritmo de juego de cada match y pulsa el botón Enviar.	2.1-El sistema verifica que todo este correcto
Flujos Alternos de Eventos	
Sección: "Crear estructura de matches individuales"	
Acción del Actor	**Respuesta del Sistema**
	2.2-Si todo está correcto el sistema muestra un mensaje informando de que la estructura de match individual se creó correctamente.
	2.3-Si existe algún error el sistema muestra un mensaje in formando que los datos son incorrectos.
Sección: "Mostrar estructura de matches individuales"	
Acción del Actor	**Respuesta del Sistema**
1-El árbitro selecciona la opción de Mostrar estructura de matches individuales.	1.1-El sistema muestra una tabla con los matches que tendrá el torneo y su respectivo ritmo de juego.
Flujos Alternos de Eventos	
Sección: "Crear estructura de matches individuales"	
Acción del Actor	**Respuesta del Sistema**
	1.2-Si no existe ninguna estructura creada en torneo el sistema muestra un mensaje informando de que no existe ninguna estructura de matches individuales.
Sección: "Eliminar estructura de matches individuales"	

Acción del Actor	Respuesta del Sistema
1-El árbitro selecciona la opción de eliminar estructura de matches individuales.	1.1-El sistema elimina la estructura de matches individuales que posee el torneo.
Flujos Alternos de Eventos	
Acción del Actor	**Respuesta del Sistema**
	1.2-Si no existe una estructura de matches individuales del torneo el sistema muestra un mensaje informando de que no existe una estructura creada.
Poscondiciones	Estructura de matches individuales creada.

Tabla 9: Descripción de caso de uso " Ver resultados por ronda"

Caso de Uso:	Ver resultados por ronda
Actores:	Árbitro Principal
Resumen:	El caso de uso se inicia cuando el árbitro selecciona la opción Resultados por Rondas, y se muestran las rondas que presenta el torneo con los resultados de cada una de ellas.
Precondiciones:	Debe haber algún resultado de los matches individuales.
Referencias	RF 34
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1-El árbitro selecciona dentro del menú del torneo muerte súbita la opción de Resultados por rondas.	1.1-El sistema muestra la opción de seleccionar la ronda del torneo que se quiere ver el resultado.
	1.2-El sistema muestra una tabla con los resultados de cada uno de los matches en la ronda.
Flujo Alterno de Eventos	
Acción del Actor	**Respuesta del Sistema**
	1.4-Si no existe ningún resultado, el sistema muestra un mensaje informando de que no

	existe ningún resultado.
Poscondiciones	Tabla con los resultados de cada ronda del torneo.

2.6 Análisis del sistema

Durante la fase de elaboración que propone RUP el flujo de trabajo principal es el análisis y diseño del sistema. En este flujo se describe cómo el sistema será realizado a partir de los requerimientos impuestos, por lo que indica con precisión lo que se debe programar. En este caso solo se tomarán en cuenta los requisitos que se adicionaron, ya que los demás fueron realizados en el trabajo de diploma realizado anteriormente titulado “Propuesta de Sistemas Integrados del proyecto Infodrez”.

2.6.1 Diagramas de clases del análisis

Estos diagramas están centrados fundamentalmente en los requisitos funcionales del sistema y son evidentes en el dominio del problema porque representan conceptos y relaciones del dominio. Los casos de uso que se tendrán en cuenta en este epígrafe son: Gestionar estructura de matches individuales y Ver resultados por ronda. A continuación se muestran los diagramas de clases que satisfacen estos casos de uso.

Tabla 10: Diagrama de clases del análisis CU Gestionar estructura de matches individuales.

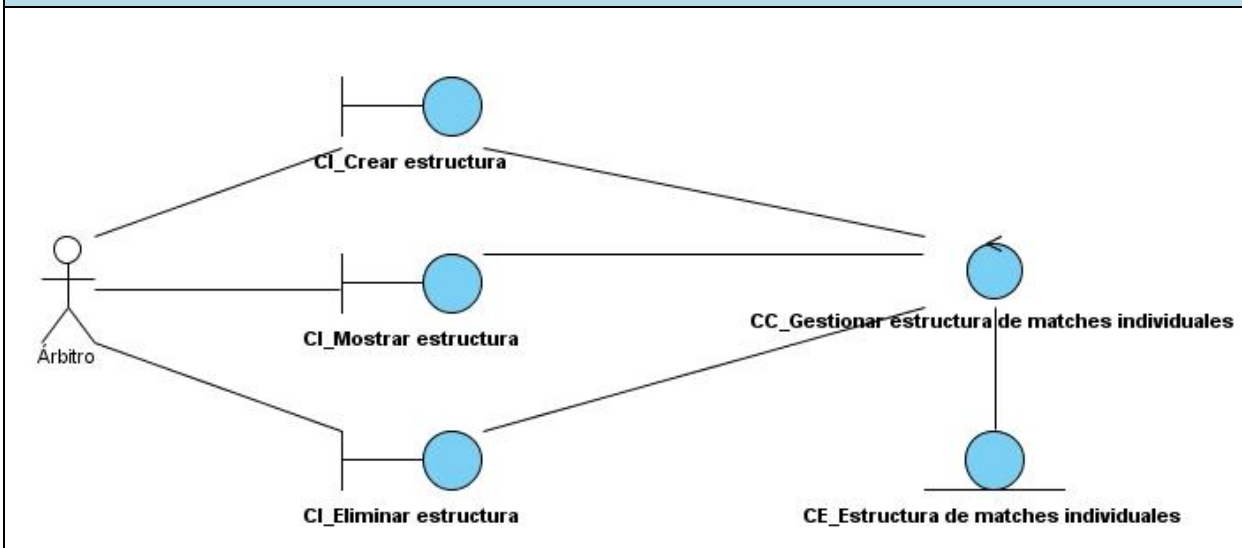
Diagrama de clases del análisis CU Gestionar estructura de matches individuales


Tabla 11: Diagrama de clases del análisis CU Ver resultados por ronda.

Diagrama de clases del análisis CU Ver resultados por ronda


2.6.1 Realización de los casos de uso del análisis.

En este epígrafe se describe fundamentalmente cómo se lleva a cabo y se ejecutan los casos de uso. En esta actividad también se comprueba que la realización del caso de uso cubre los requisitos que se hicieron y que refleja el comportamiento de su caso de uso correspondiente y sólo ese. A continuación se presentan los diagramas de colaboración de los casos de uso y cada uno de sus escenarios.

Tabla 12: Caso de Uso Gestionar estructura de matches individuales.

Caso de Uso Gestionar estructura de matches individuales
Diagrama de Colaboración del Análisis Sección “Crear estructura”

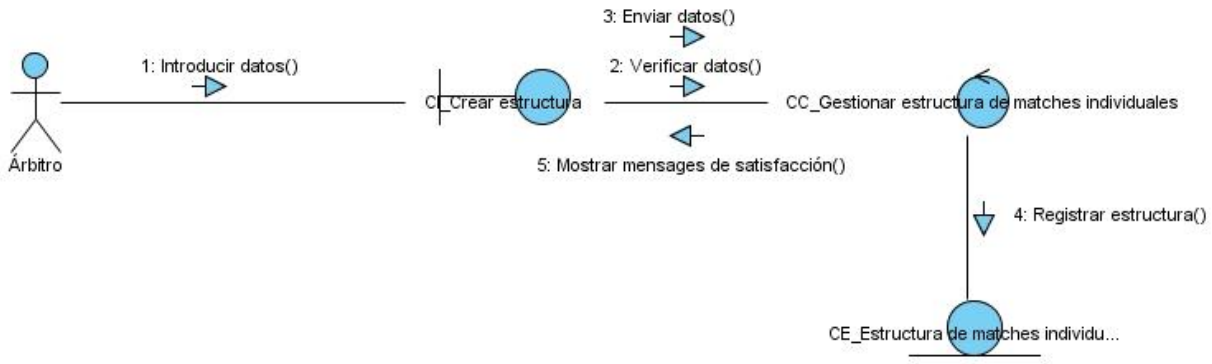



Diagrama de Colaboración del Análisis Sección “Mostrar estructura”

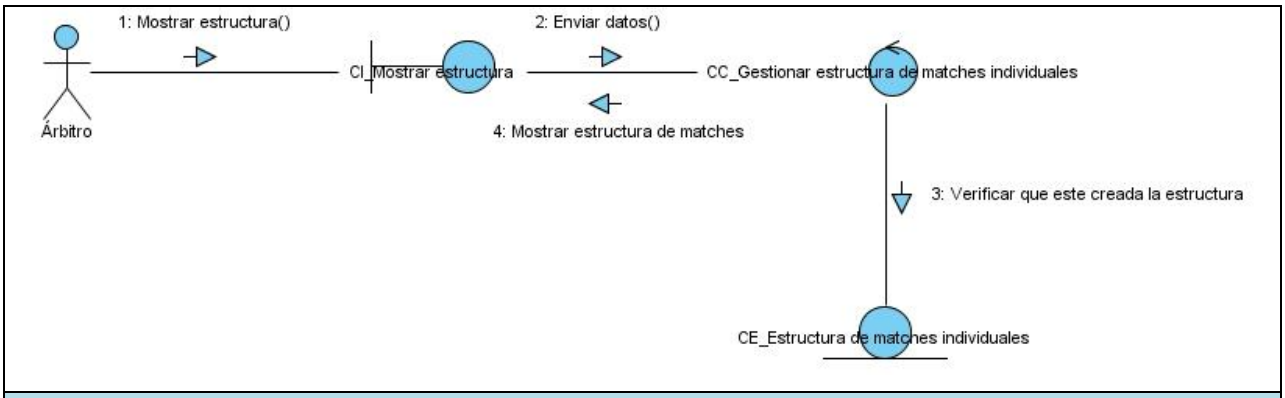


Diagrama de Colaboración del Análisis Sección “Eliminar estructura”

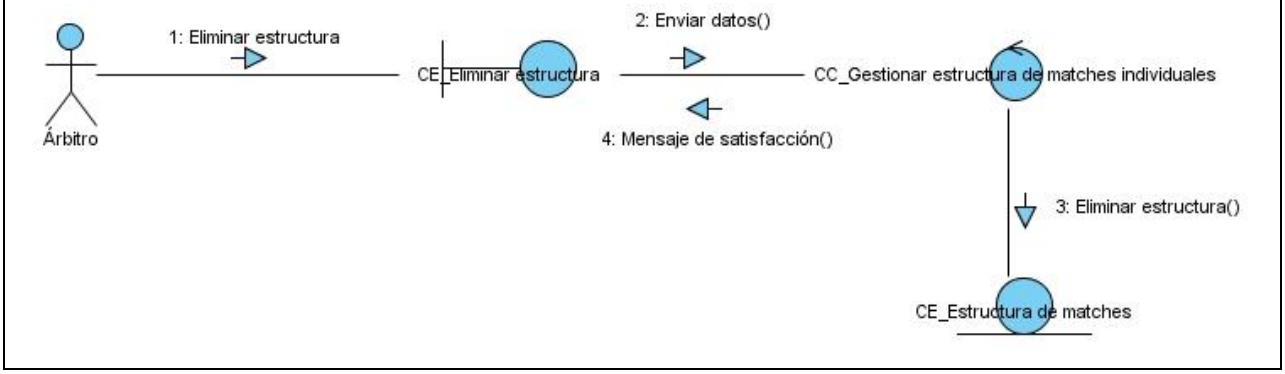
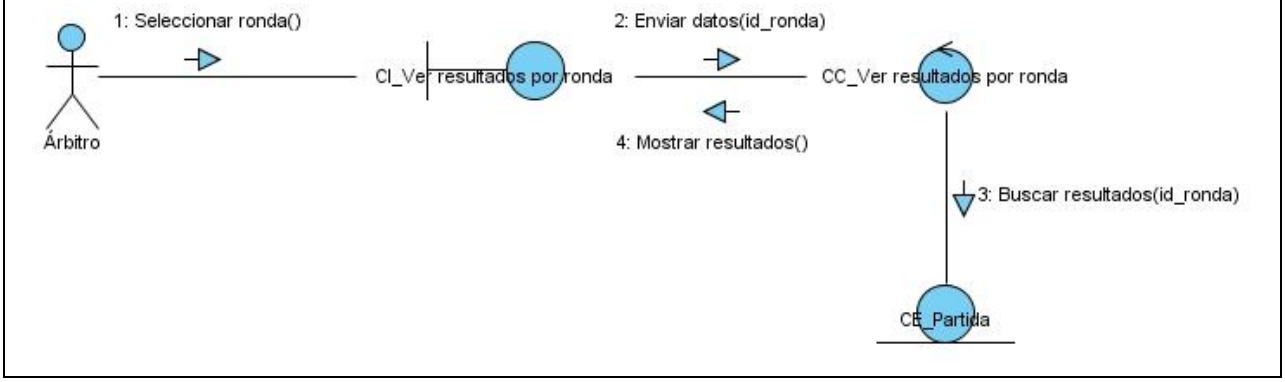


Tabla 13: Diagrama de Colaboración Caso de Uso Gestionar estructura de matches individuales.

Diagrama de Colaboración Caso de Uso Gestionar estructura de matches individuales



2.7 Diseño del sistema

El modelo de diseño, es un modelo de objetos que describe la realización física de los casos de uso, centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además, sirve de abstracción de la implementación del sistema y es de ese modo utilizado como una entrada fundamental de las actividades de implementación.

2.7.1 Diagramas de clases del diseño

Tabla 14: Diagrama de Clases del diseño del CU Gestionar estructura de matches individuales.

Diagrama de Clases del diseño del CU Gestionar estructura de matches individuales

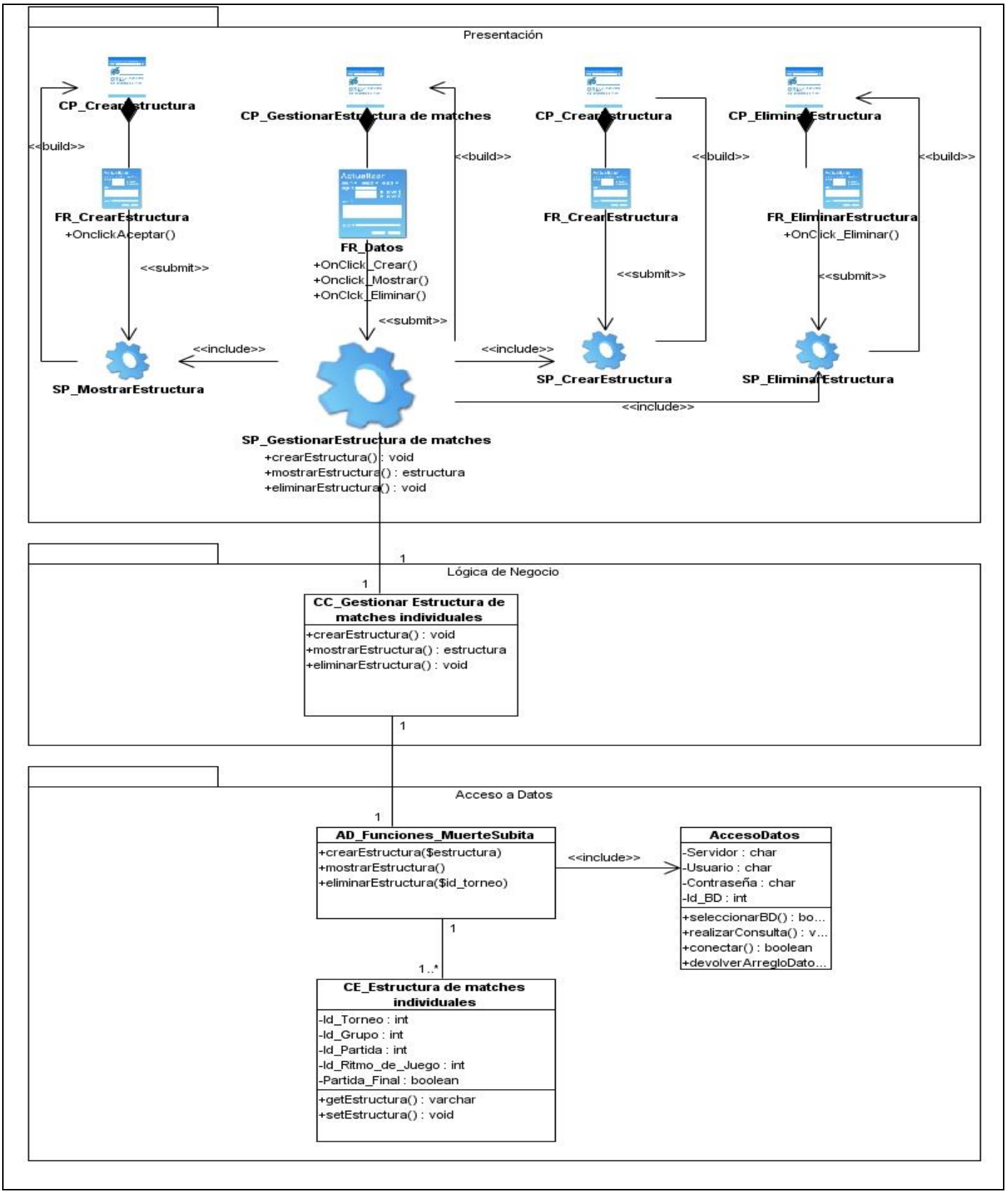
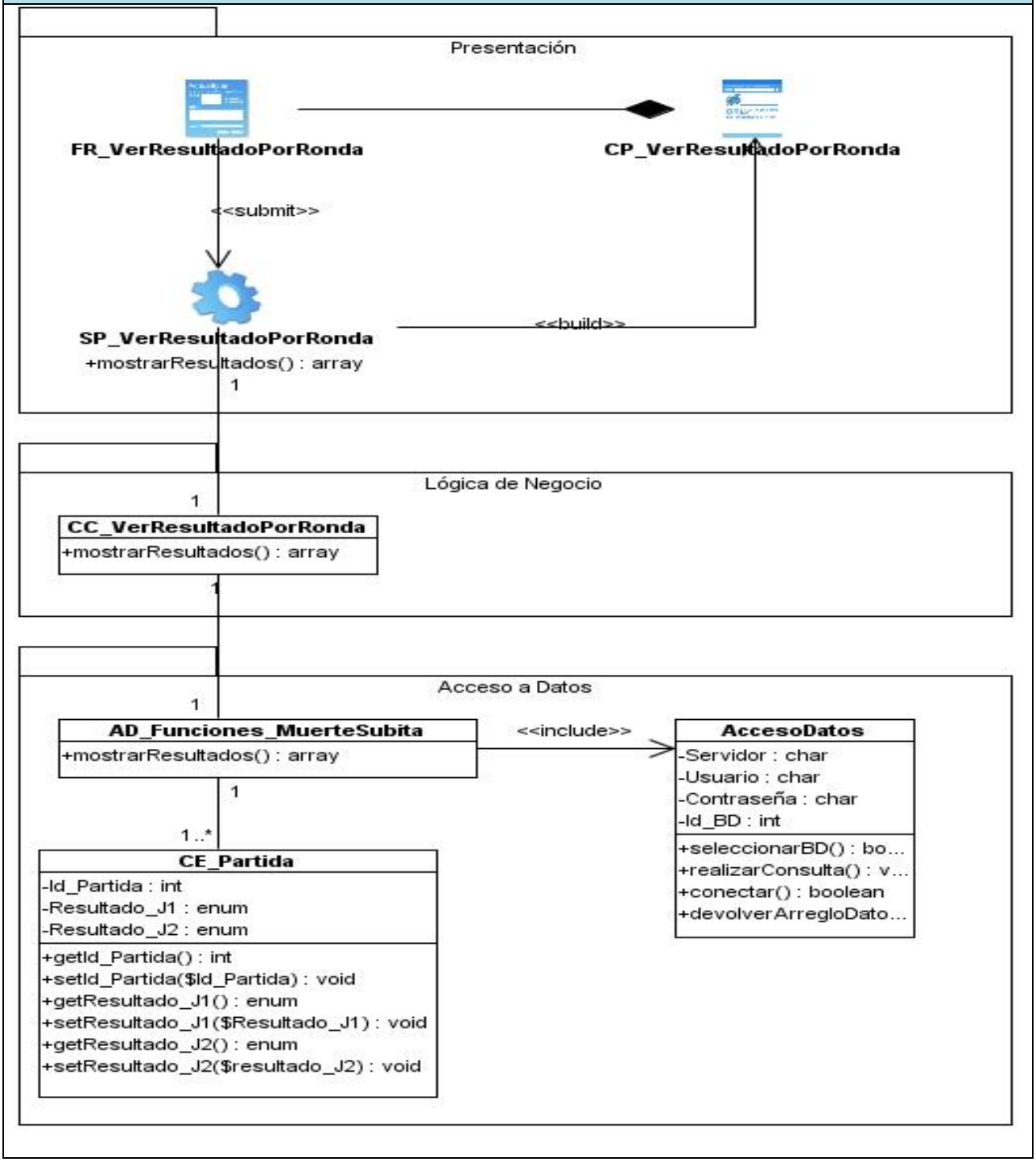

Presentación
CP_CrearEstructura
CP_GestionarEstructura de matches
CP_CrearEstructura
CP_EliminarEstructura
<<build>>
<<build>>
<<build>>
<<build>>
FR_CrearEstructura
+OnclickAceptar()
FR_Datos
+OnClick_Crear()
+Onclick_Mostrar()
+OnClck_Eliminar()
FR_CrearEstructura
FR_EliminarEstructura
+OnClick_Eliminar()
<<submit>>
<<submit>>
<<submit>>
<<submit>>
<<include>>
<<include>>
SP_MostrarEstructura
SP_CrearEstructura
SP_EliminarEstructura
<<include>>
SP_GestionarEstructura de matches
+crearEstructura() : void
+mostrarEstructura() : estructura
+eliminarEstructura() : void
1
Lógica de Negocio
1
CC_Gestionar Estructura de matches individuales
+crearEstructura() : void
+mostrarEstructura() : estructura
+eliminarEstructura() : void
1
Acceso a Datos
1
AD_Funciones_MuerteSubita
+crearEstructura($estructura)
+mostrarEstructura()
+eliminarEstructura($id_torneo)
<<include>>
AccesoDatos
-Servidor : char
-Usuario : char
-Contraseña : char
-Id_BD : int
+seleccionarBD() : bo...
+realizarConsulta() : v...
+conectar() : boolean
+devolverArregloDato...
1
1..*
CE_Estructura de matches individuales
-Id_Torneo : int
-Id_Grupo : int
-Id_Partida : int
-Id_Ritmo_de_Juego : int
-Partida_Final : boolean
+getEstructura() : varchar
+setEstructura() : void

Tabla 15: Diagrama de Clases del diseño del CU Ver resultados por ronda.

Diagrama de Clases del diseño del CU Ver resultados por ronda


2.8 Prototipos de interfaz

Se crearon interfaces de usuario amigables y sencillas para los casos de uso incorporados, que eran necesarios para la versión de la aplicación que se quiere implementar. Se presenta una interfaz para la funcionalidad de crear estructura de matches individuales y una interfaz para mostrar y eliminar la estructura de matches.

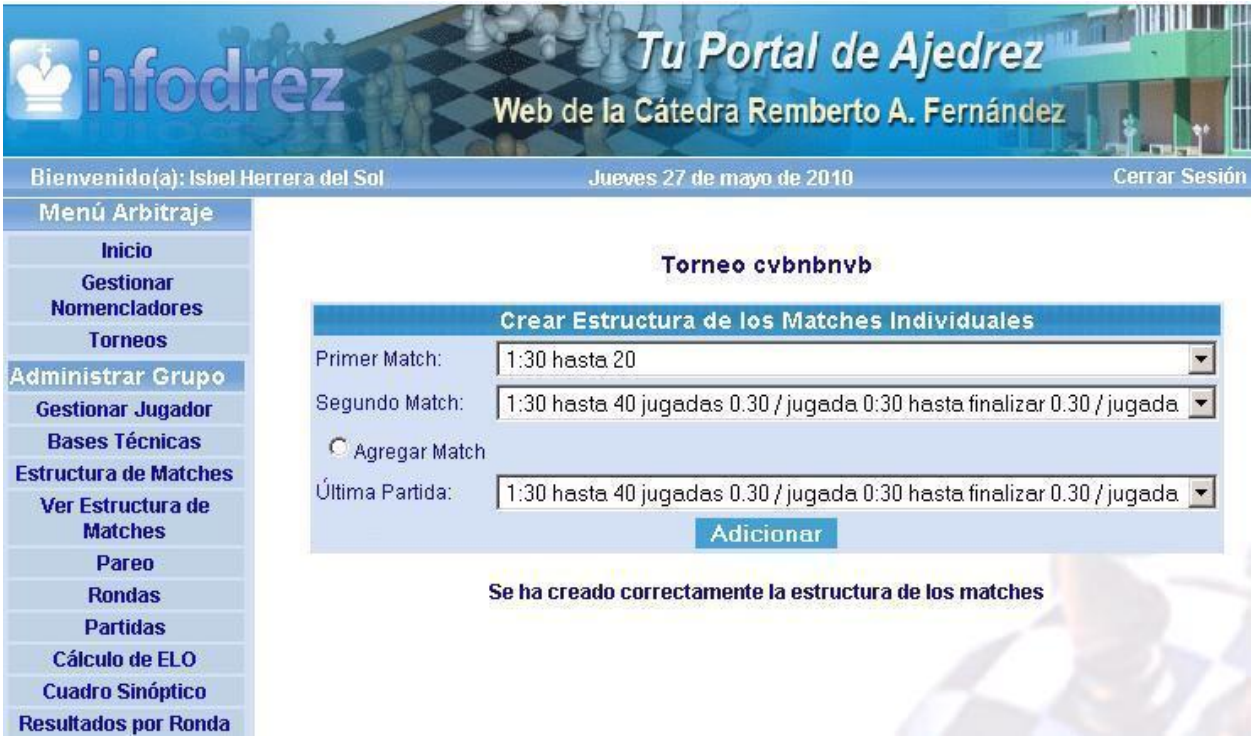


Figura 4: Interfaz de la funcionalidad: Crear estructura de matches individuales.

Figura 5: Interfaz de la funcionalidad: Mostrar y Eliminar estructura de matches individuales.

2.9 Conclusiones

A lo largo de este capítulo se elaboró una propuesta de las características funcionales que debe presentar el sistema. Se hizo referencia al modelo de dominio y a los requisitos del sistema realizado en el trabajo de diploma titulado “Propuesta de Sistemas Integrados del proyecto Infodrez”, y además de modificar el modelo del sistema. Se realizó un análisis y diseño del sistema a los casos de uso que se agregaron para la versión que se quiere implementar. Después del desarrollo de este capítulo se tiene una mejor visión de las características que tendrá el sistema, facilitando así el trabajo en los demás flujos de trabajo que propone RUP.

Capítulo 3. Implementación y pruebas

3.1 Introducción

En este capítulo describe el modelo de implementación utilizado y se muestran los diagramas de componentes y de despliegue. Para el desarrollo del mismo se tuvieron en cuenta algunos artefactos que fueron generados en el flujo de análisis y diseño, que aparecen en el trabajo de diploma “Propuesta de sistemas Integrados del Proyecto Infodrez”.

3.2 Modelo de Implementación

El resultado principal del flujo de trabajo implementación es el modelo de implementación, el cual se describe, partiendo de componentes, ficheros de código fuente, ejecutables, scripts, librerías, dentro del flujo de implementación que propone RUP. Este modelo está conformado por los diagramas de despliegue y de componentes, éstos describen los componentes a construir y su organización y dependencia entre nodos físicos en los que funcionará el sistema. El modelo de implementación es la entrada principal de las etapas de pruebas, que es el flujo de trabajo que le sigue al de implementación.

3.2.1 Diagramas de componentes

Los diagramas de componentes son usados para estructurar el modelo de implementación en subsistemas de implementación y mostrar las relaciones entre estos elementos. Es un esquema de componentes unidos a través de relaciones que pueden ser de compilación o de ejecución. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes. Los diagramas que se presentan a continuación tienen como objetivo figurar la estructura general de la aplicación en desarrollo, en términos de componentes.

Figura 6: Diagrama de Componentes CU Autenticar Usuario.

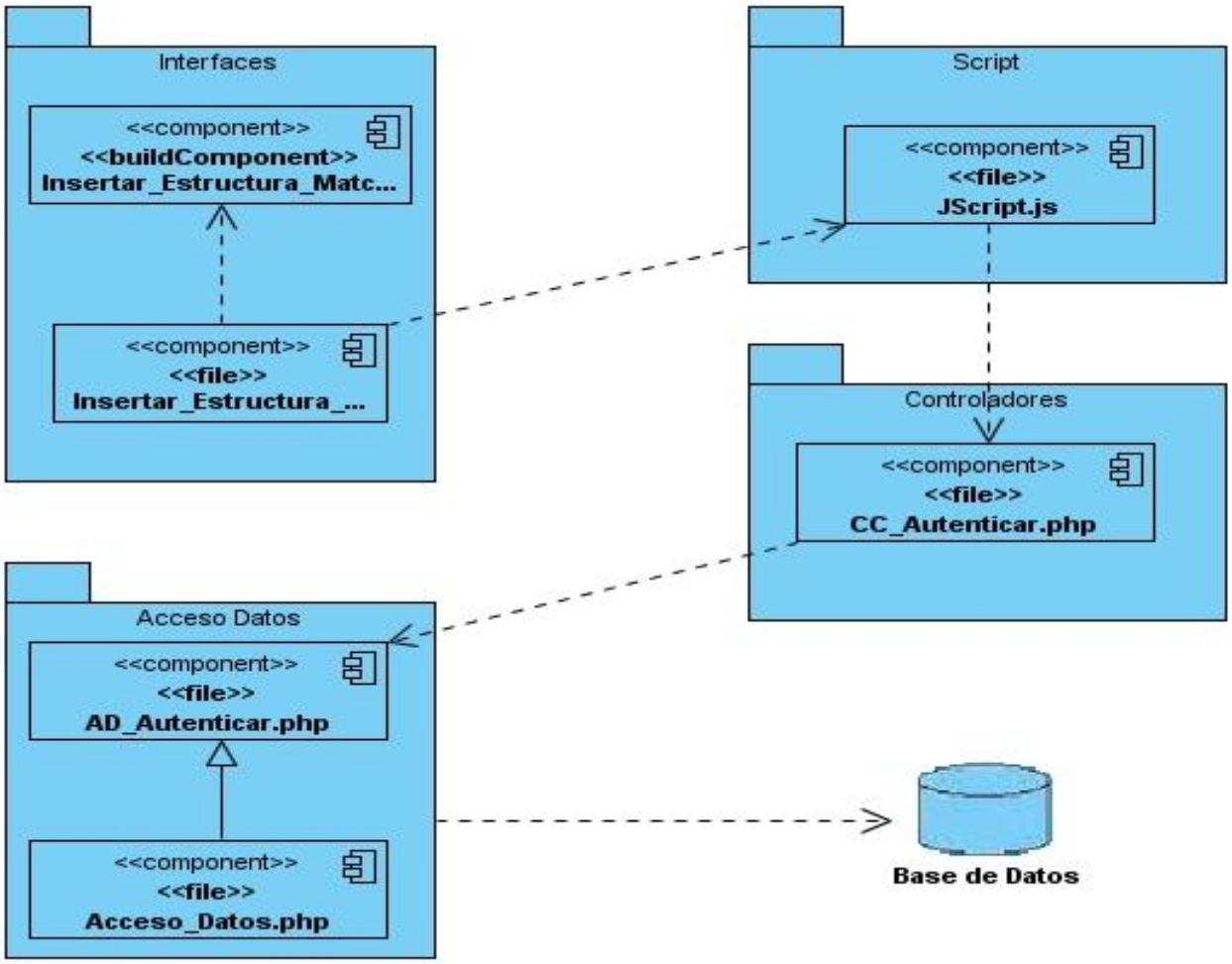


Figura 7: Diagrama de Componentes CU Insertar Estructura de Matches Individuales.

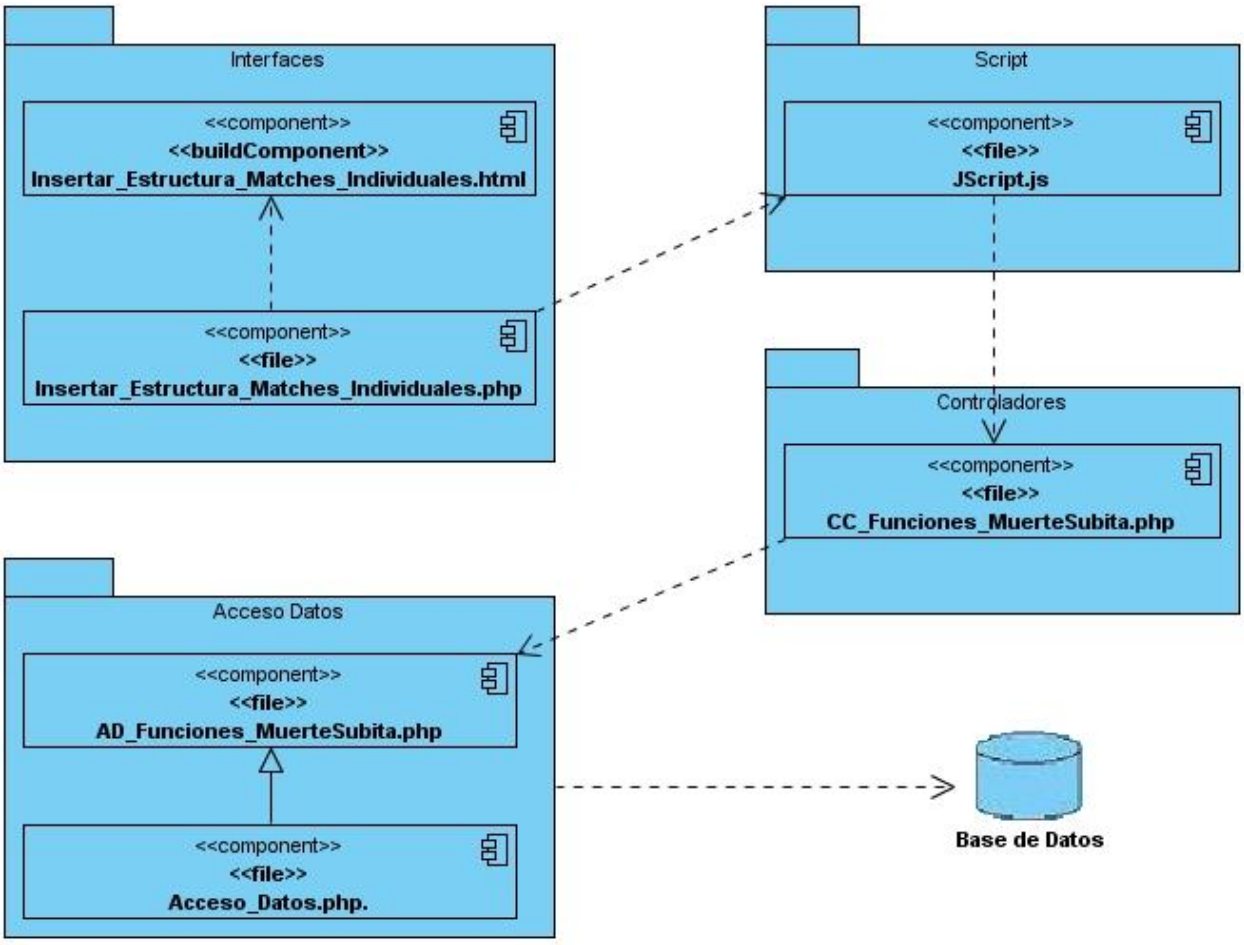


Figura 8: Diagrama de Componentes CU Mostrar Cuadro Sinóptico.

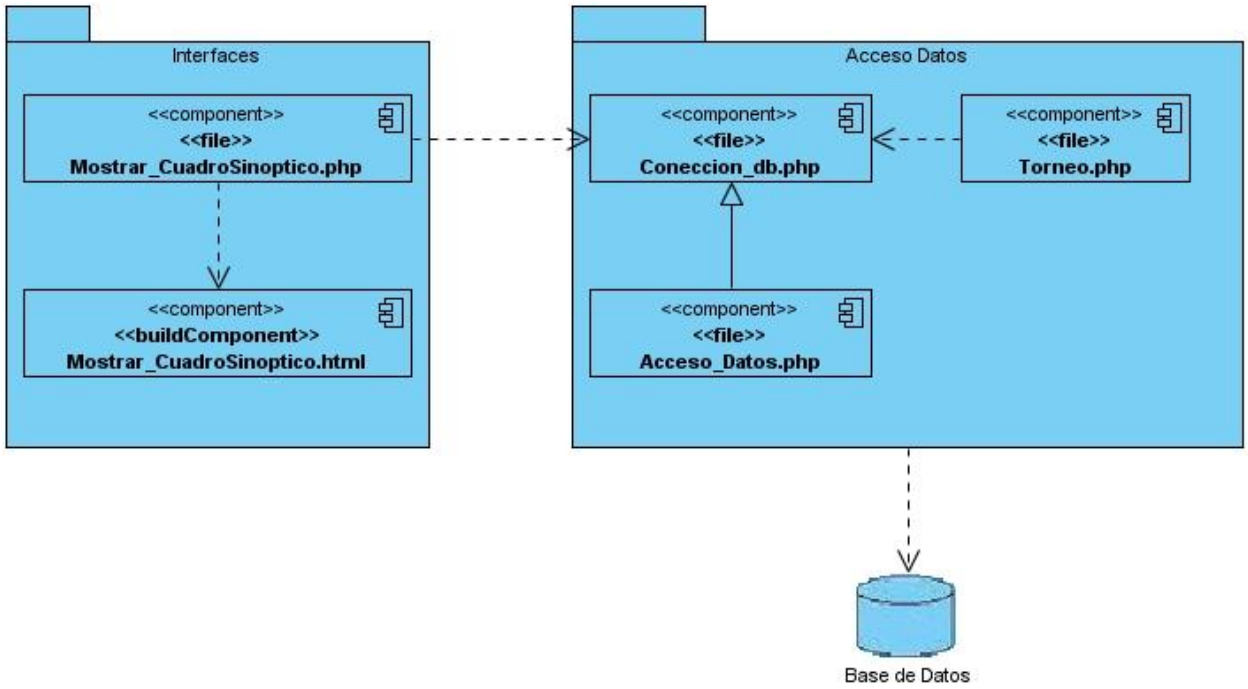


Figura 9: Diagrama de Componentes CU Mostrar Pareo Muerte Súbita.

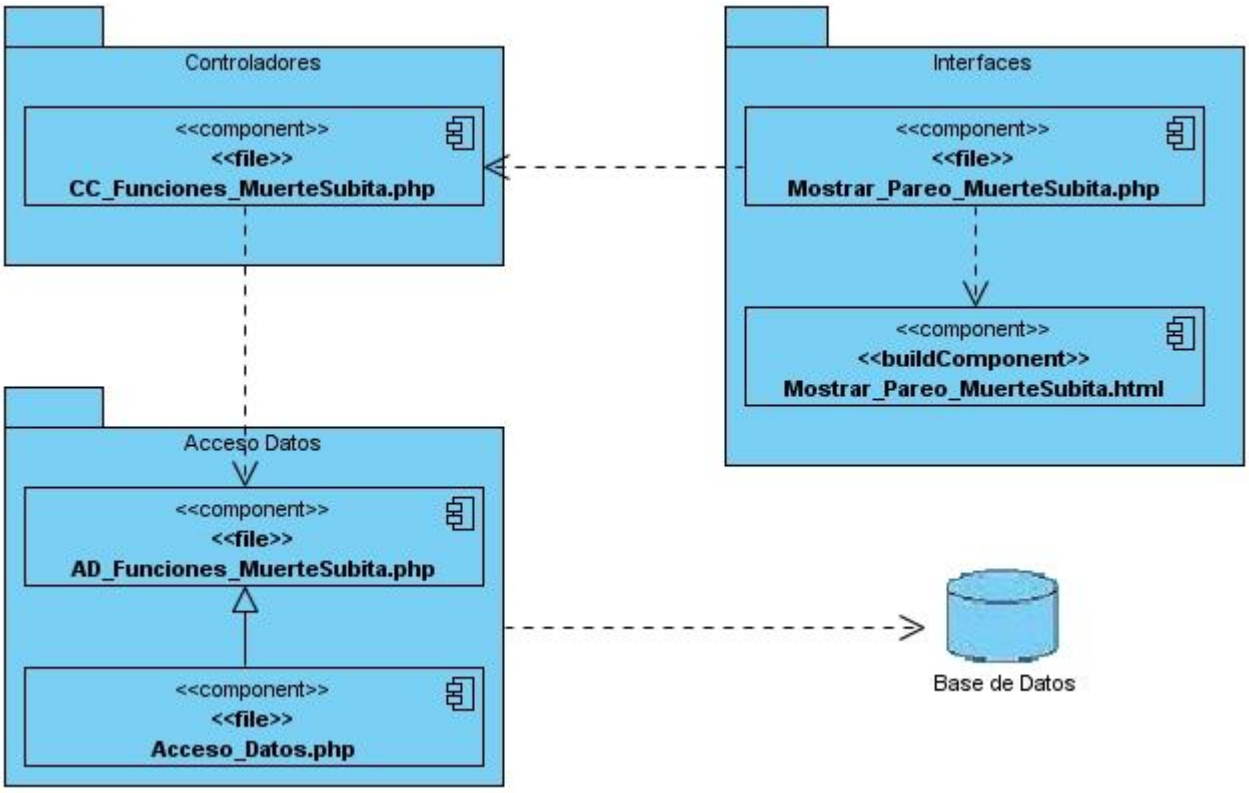
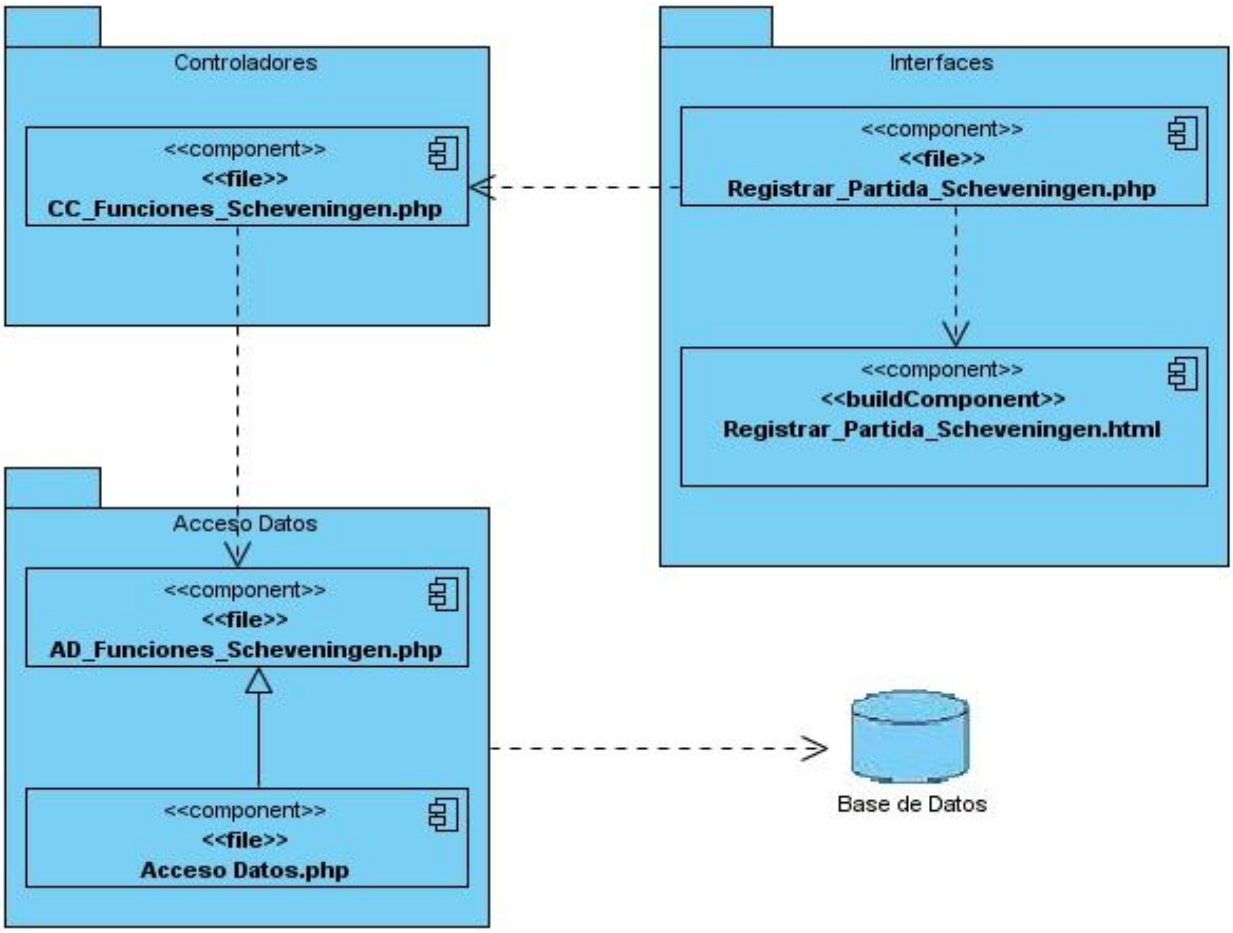
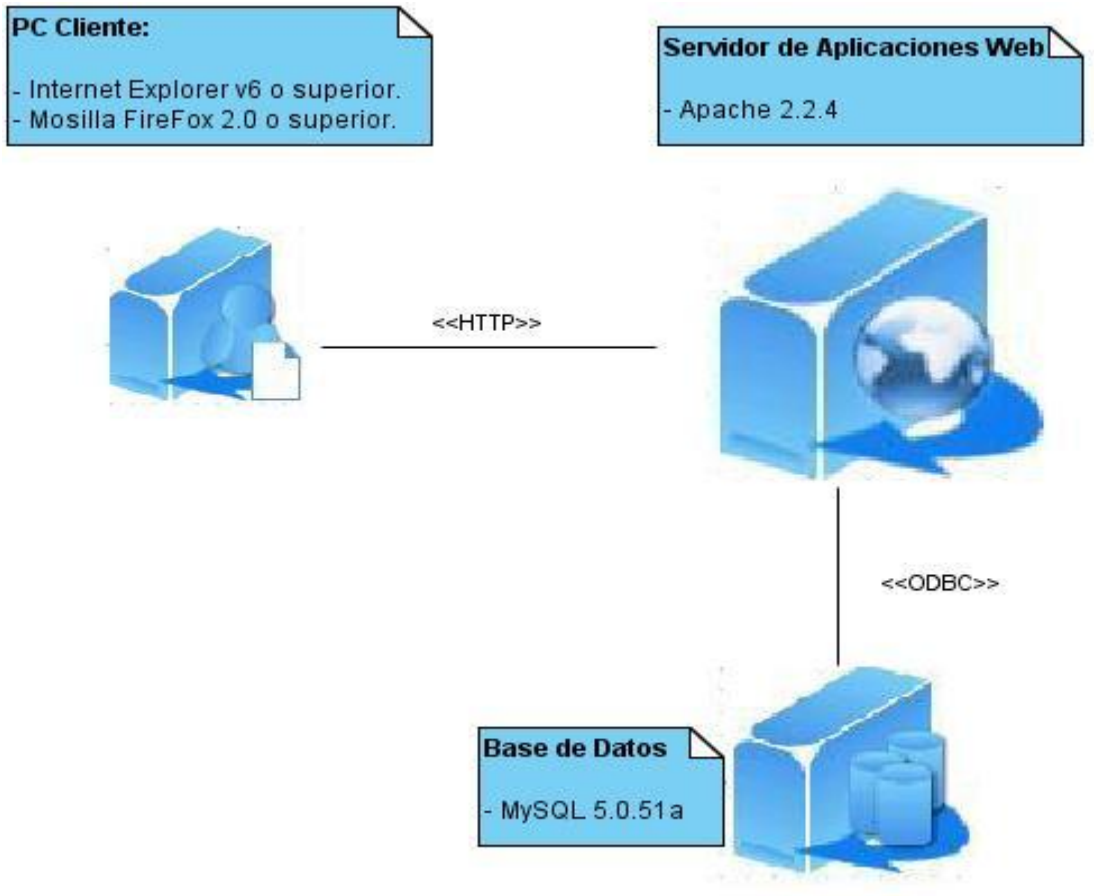


Figura 10: Diagrama de Componentes CU Registrar Partida Scheveningen.



3.2.2 Diagrama de despliegue

En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados. Un diagrama de despliegue es un grafo de nodos unidos por conexiones de comunicación, donde los nodos son máquinas físicas o procesadores. Este diagrama muestra cómo y dónde se desplegará el sistema.



3.3 Modelo de pruebas

El objetivo del flujo de trabajo de pruebas es verificar que el sistema cumpla con los requisitos del sistema y lograr una mayor calidad en el producto final. Para obtener estos resultados es necesario realizar y evaluar las pruebas como se describen en este modelo. Aquí se describe principalmente cómo se prueban los componentes ejecutables en el modelo de implementación.

3.3.1 Métodos de pruebas

Los principales tipos de pruebas son las pruebas de caja negra y las pruebas de caja blanca.

Pruebas de caja negra

Son las que se llevan a cabo sobre la interfaz de usuario, está centrada principalmente en los requisitos funcionales del software. Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de la aplicación.

Examina aspectos del modelo principalmente del sistema sin tener mucho en cuenta la estructura interna del software.

Pruebas de caja blanca

Las pruebas de caja blanca consisten en realizar un seguimiento del código fuente según se van ejecutando los casos de pruebas, determinando de manera concreta las instrucciones en las que existen errores. Requieren de la estructura interna del programa y son derivadas a partir de las especificaciones internas de diseño o el código.

3.3.2 Casos de pruebas

Un caso de prueba especifica una forma de probar el sistema, incluyendo la entrada o resultado con la que se ha de probar y las condiciones bajo las que ha de probarse. Para comprobar el funcionamiento del sistema se realizaron diferentes casos de pruebas de caja negra:

Tabla 16: Caso de prueba CU Gestionar estructura de matches individuales.

Caso de Prueba	**CU: Gestionar estructura de matches individuales.**	
Sección: Crear estructura de matches individuales		
Entrada	**Resultados**	**Descripción**
El árbitro selecciona los matches que tendrá el torneo pero no selecciona el ritmo de juego de los matches.	El sistema muestra un mensaje de error informando de que debe completar la información.	Esta operación se repite hasta que la información esta correcta.
Sección: Crear estructura de matches individuales		
El árbitro selecciona la opción de mostrar la estructura de matches individuales.	El sistema muestra la estructura de matches individuales.	La estructura de matches debe estar creada.
Sección: Eliminar estructura de matches individuales		

El árbitro selecciona la opción de eliminar la estructura de matches individuales.	El sistema elimina la estructura de matches individuales.	La estructura de matches individuales debe estar creada.

Tabla 17: Caso de prueba CU Ver resultados por ronda.

Caso de Prueba	**CU: Ver resultados por ronda**	
Entrada	**Resultados**	**Descripción**
El árbitro selecciona la ronda que desea ver.	El sistema muestra los resultados de la ronda seleccionada.	La ronda debe tener resultados en ese momento.

Tabla 18: Caso de prueba CU Mostrar emparejamiento

Caso de Prueba	**CU: Mostrar emparejamiento**	
Entrada	**Resultados**	**Descripción**
El árbitro selecciona la opción de mostrar emparejamiento.	El sistema muestra el emparejamiento que le corresponde al tipo de torneo que se gestiona en ese momento.	Los jugadores o los equipos deben estar adicionados al torneo que se está gestionando y además deben estar sorteados.

3.4 Conclusiones

En este capítulo se mostraron los principales artefactos del flujo de trabajo implementación, específicamente los diagramas de componentes y el de despliegue. Los diagramas de componentes se utilizaron para representar a través de un grafo los componentes de software unidos por medio de relaciones de dependencia; por lo cual se obtuvo la vista estática del sistema. Además, sirvió para mostrar la organización y las dependencias lógicas entre un conjunto de componentes software. El modelo de prueba desarrollado permitió que el sistema cumpliera con los requisitos funcionales y que este tuviera una mejor calidad.

Conclusiones generales

Después de finalizado este trabajo se llega a la conclusión de que los objetivos propuestos fueron cumplidos exitosamente. Se llevó a cabo un estudio de la metodología, lenguajes y herramientas que se necesitaban para el desarrollar la aplicación, el cual se hizo con el objetivo de tener un conocimiento más amplio sobre las mismas. Este estudio sirvió para hacer un mejor uso de estas, conociendo sus ventajas y facilidades. Se llevó a cabo el desarrollo de la aplicación web, así como las pruebas a este, por lo que ya es posible gestionar todos los torneos de ajedrez que se desarrollan en la Universidad de las Ciencias Informáticas, sin necesidad de utilizar un software propietario. De esta forma, queda resuelto el problema que existía en cuanto a la gestión de torneos de ajedrez en la UCI, planteado al principio de este trabajo.

El desarrollo de este trabajo ha permitido obtener un gran conocimiento en cuanto al tema de la gestión de torneos de ajedrez, así como, en cuanto al desarrollo de aplicaciones web de gestión. Después de terminado este trabajo de diploma, la aplicación de Arbitraje es capaz de gestionar los torneos: todos contra todos Individuales y por equipos, suizos individuales y por equipos, muerte súbita y sheveningen; por lo que puede sustituir así al Swiss Manager en los torneos oficiales de ajedrez en la UCI.

Recomendaciones

Como resultado del proceso de investigación e implementación de la aplicación, han surgido ideas que serían recomendables tener en cuenta para un futuro perfeccionamiento del sistema, a continuación se listan las mismas:

- Incluir en la aplicación la opción de imprimir las bases técnicas de cada uno de los torneos.

Referencias bibliográficas

1. **Alvarez, Miguel Angel. 2003.** Desarrolloweb. [En linea] 6 4, 2003. [Citado: 1 22, 2010.] http://www.desarrolloweb.com/articulos/1178.php.
2. **—. 2004.** Desarrolloweb. [En linea] 2004. [Citado: 01 22, 2010.] http://www.desarrolloweb.com/articulos/182.php.
3. **Alvarez, Sara. 2007.** Desarrolloweb. [En linea] 7 31, 2007. [Citado: 01 22, 25.] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.
4. **Aumaille, Benjamin. 2000.** *JavaScript y VBScript.* Barcelona : Ediciones ENI, 2000. ISBN: 2-7460-0982-X.
5. **Cobo, Ángel, et al. 2005.** *PHP y MySQL Tecnologías para el desarrollo de aplicaciones web.* España : Ediciones Días de Santos, 2005. ISBN: 84-7978-706-6.
6. **Díaz, José Márquez. 2002.** Instalación y configuración de Apache, un servidor Web gratis. [En linea] 10 23, 2002. http://ciruelo.uninorte.edu.co/pdf/ingenieria_desarrollo/12/instalacion_y_configur.
7. **Dondo, Agustín. 2002.** PHP en Castellano. [En linea] 2002. http://www.programacion.com/php/articulo/porquephp/.
8. **Eguíluz Pérez, Javier.** Librosweb. [En linea] http://librosweb.es/javascript/capitulo1.html.
9. **Gracia, Joaquin. 2005.** ingenierosoftware. [En linea] 5 7, 2005. [Citado: 3 15, 2010.] http://www.ingenierosoftware.com.
10. **INNOVACION Y CUALIFICACION, S.L. 2001.** *JavaScript.* Malaga : INNOVACION Y CUALIFICACION, S.L., 2001. ISBN: 84-95733-18-8.
11. **Jacobson, Ivar, Booch, Grady and Rumbaugh, James. 1999.** *The Unified Software Development Process.* s.l. : Addison-Wesley, 1999. ISBN: 0201571692.
12. **Márquez Díaz, José, Sampedro, Leonardo y Vargas, Félix. 2002.** ciruelo.uninorte.edu.co. [En linea] 10 23, 2002. [Citado: 02 5, 2010.] http://ciruelo.uninorte.edu.co/pdf/ingenieria_desarrollo/12/instalacion_y_configuracion_de_apache.pdf.

13. **Martín, Eloi de San. 2006.** Programaciónweb. [En linea] 2006. [Citado: 01 30, 2010.] http://www.programacionweb.net/articulos/articulo/?num=184.

14. **Mendoza Sanchez, María A. 2004.** [En linea] 7 6, 2004. [Citado: 02 5, 2010.] http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

15. **Mendoza Sánchez, María A. 2004.** [En linea] 6 7, 2004. http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

16. **Moliner López, Francisco Javier. 2005.** *Programación Web con Visual Studio y ASP.NET 2.0.* Valencia : Editorial Mad. S.L., 2005. ISBN: 84-665-2077-5.

17. **Monteiro Lazaro, Juliana.** Desarrolloweb. [En linea] http://www.desarrolloweb.com/articulos/26.php.

18. **Morales Zorrilla, Enrique.** Torneos de ajedrez.[En linea] 2009, abril 26. [Citado: 25 4, 2010] http://www.torneosajedrez.com/ajedrez47/eventos/668/LA_PUNTUACION_3-1-0.doc

19. **Rubio Pardo, J.** Ajedrezcastellon.[En linea] 2010, marzo 20. [Citado: abril 25, 2010] http://ajedrezcastellon.com/2010/03/20/swiss-perfect-98/

20. **Sierra, Daniel. 2007.** Slidershare. [En linea] 2007. [Citado: 02 7, 2010.] http://www.slideshare.net/vanquishdarkenigma/visual-paradigm-for-uml.

21. **Van Lancker, Luc. 2007.** *Technote CSS 1 y CSS 2.1.* Barcelona : Ediciones ENI, 2007. ISBN: 978-2-7460-3583-6.

Bibliografía

1. **Alvarez, Miguel Angel. 2003.** Desarrolloweb. [En linea] 6 4, 2003. [Citado: 1 22, 2010.] http://www.desarrolloweb.com/articulos/1178.php.
2. **—. 2004.** Desarrolloweb. [En linea] 2004. [Citado: 01 22, 2010.] http://www.desarrolloweb.com/articulos/182.php.
3. **Alvarez, Sara. 2007.** Desarrolloweb. [En linea] 7 31, 2007. [Citado: 01 22, 25.] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.
4. **Arguelles Blanco, Yadira y Abreu Hernández, Lienys. 2009.** *Propuesta de sistemas Integrados del Proyecto Infodrez.* Universidad de las Ciencias Informáticas : Ciudad de La Habana, 2009, Cantidad de páginas 324.
5. **Aumaille, Benjamin. 2000.** *JavaScript y VBScript.* Barcelona : Ediciones ENI, 2000. ISBN: 2-7460-0982-X.
6. **Cobo, Ángel, et al. 2005.** *PHP y MySQL Tecnologías para el desarrollo de aplicaciones web.* España : Ediciones Días de Santos, 2005. ISBN: 84-7978-706-6.
7. **Díaz, José Márquez. 2002.** Instalación y configuración de Apache, un servidor Web gratis. [En linea] 10 23, 2002. http://ciruelo.uninorte.edu.co/pdf/ingenieria_desarrollo/12/instalacion_y_configur.
8. **Dondo, Agustín. 2002.** PHP en Castellano. [En linea] 2002. http://www.programacion.com/php/articulo/porquephp/.
9. **Eguíluz Pérez, Javier.** Librosweb. [En linea] http://librosweb.es/javascript/capitulo1.html.
10. **Gracia, Joaquin. 2005.** ingenierosoftware. [En linea] 5 7, 2005. [Citado: 3 15, 2010.] http://www.ingenierosoftware.com.
11. **INNOVACION Y CUALIFICACION, S.L. 2001.** *JavaScript.* Malaga : INNOVACION Y CUALIFICACION, S.L., 2001. ISBN: 84-95733-18-8.
12. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 1999.** *The Unified Software Development Process.* s.l. : Addison-Wesley, 1999. ISBN: 0201571692.
13. **Larman, Craig. 2002.** UML y Patrones. s.l. : Prentice Hall, 2002.

14. **Manresa Peña, Yendri y Carmenate Garcia, Henry Herminio. 2008.** *Concepción y desarrollo del Módulo Arbitraje del Proyecto Infodrez.* Universidad de las Ciencias Informática : Ciudad de La Habana, 2008, Cantidad de páginas 110.

15. **Márquez Díaz, José, Sampedro, Leonardo and Vargas, Félix. 2002.** ciruelo.uninorte.edu.co. [En linea] 10 23, 2002. [Citado: 02 5, 2010.] http://ciruelo.uninorte.edu.co/pdf/ingenieria_desarrollo/12/instalacion_y_configuracion_de_apache.pdf.

16. **Martin Fowler, Sccott Kendall. 1999.** *UML Gota a Gota.* 1999.

17. **Martín, Eloi de San. 2006.** Programaciónweb. [En linea] 2006. [Citado: 01 30, 2010.] http://www.programacionweb.net/articulos/articulo/?num=184.

18. **Mendoza Sanchez, María A. 2004.** [En linea] 7 6, 2004. [Citado: 02 5, 2010.] http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

19. **Mendoza Sánchez, María A. 2004.** [En linea] 6 7, 2004. http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

20. **Moliner López, Francisco Javier. 2005.** *Programación Web con Visual Studio y ASP.NET 2.0.* Valencia : Editorial Mad. S.L., 2005. ISBN: 84-665-2077-5.

21. **Monteiro Lazaro, Juliana.** Desarrolloweb. [En linea] http://www.desarrolloweb.com/articulos/26.php.

22. **Morales Zorrilla, Enrique.** Torneos de ajedrez.[En linea] 2009, abril 26. [Citado: 25 4, 2010] http://www.torneosajedrez.com/ajedrez47/eventos/668/LA_PUNTUACION_3-1-0.doc

23. **Moreno, G. 2008.** *Ingeniería del Software UML .* 2008.

24. **PARADIGM, C. V.** [En línea] [Citado el: 19 de marzo de 2010.] http://www.visual-paradigm.com/product/vpuml.

25. **Rubio Pardo, J.** Ajedrezcastellon.[En linea] 2010, marzo 20. [Citado: Abril 25, 2010] http://ajedrezcastellon.com/2010/03/20/swiss-perfect-98/

26. **Sierra, Daniel. 2007.** Slidershare. [En linea] 2007. [Citado: 02 7, 2010.] http://www.slideshare.net/vanquishdarkenigma/visual-paradigm-for-uml.

27. **Van Lancker, Luc. 2007.** *Technote CSS 1 y CSS 2.1.* Barcelona : Ediciones ENI, 2007. ISBN: 978-2-7460-3583-6.

Glosario de Términos

Software: Programas de sistemas, utilerías o aplicaciones expresados en un lenguaje de máquina.

Software libre: Es un software el cual los usuarios pueden ejecutar, copiar, distribuir, estudiar, cambiar y mejorar, sin tener que pedir o pagar permisos.

Software propietario: Es un software al que no se tiene acceso al código fuente, por lo que no es posible modificarlo. Además, se necesita pagar una licencia para usar este tipo de software.

Base de datos: Conjunto no redundante de información almacenada en memoria organizada independientemente de su utilización y su implementación en máquinas accesibles en tiempo real, y compatibles con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo.

ELO: Método para calcular la fuerza relativa de los jugadores de ajedrez.

Infodrez: Proyecto de desarrollo de software para el ajedrez ubicado en la Universidad de las Ciencias Informática.