RESUMEN

La evolución constante y acelerada de la tecnología, genera la necesidad de contar con sistemas informáticos que se conviertan en herramientas de alto nivel para el control y administración de cada proceso y actividad desarrollada en determinada empresa, institución u organismo. El Internet, complementa estos sistemas informáticos al ser hoy en día, la fuente más importante de acceso a cualquier tipo de información; además, permite: acortar distancias, trabajar remotamente mejorando la productividad y mantener una estandarización e integración de actividades sin importar el lugar físico en el que se localice el talento humano. Estos son algunos de los motivos por los que, la Asociación de TAE KWON DO de Pichincha, ha decidido oportunamente, mejorar y automatizar sus procesos para optimizar recursos, tiempo y personal; para facilitar la ejecución de las actividades propias de su misión; los torneos y competencias. Este articulo presenta la implementación de un sistema en ambiente WEB que, permite a los diferentes usuarios acceder desde cualquier sitio a un servidor; en el que reside una aplicación informática y una base de datos robusta que, facilita el registro de todos y cada uno de los movimientos propios de la administración de torneos y competencias y cuyo resultado final es, generar llaves de competencia automáticamente y administrar flujos de efectivo para controlar el capital activo de la Asociación. Para lograr dicho resultado, se ha diseñado un algoritmo que permite: generar fácilmente los grupos de competencia filtrando parámetros básicos como categoría, peso y género; ingresar puntajes por combate y finalmente, evaluar los ingresos y egresos generados. Los resultados del uso de este sistema revelan un comportamiento rápido y eficaz de los procesos de la Asociación, permitiendo que los directivos tomen decisiones oportunas y puedan revisar todo tipo de información desde cualquier sitio y a cualquier momento.

ABSTRACT

The constant and rapid evolution of technology creates the need for computer systems that become higher-level tools for monitoring and managing each process and activity in certain company, institution or agency. The Internet complements these systems; because nowadays, it is the most important source of access to any information, it allows: closing the gap, working remotely, improve productivity and maintain standardization and integration of activities no matter where is the human talent. These are some of the reasons why the Association of Pichincha TAE KWON DO, has decided in due course, improve and automate their processes to optimize resources, time and personnel, to facilitate the implementation of the activities of its mission tournaments and skills. This article presents the implementation of a Web environment that allows different users to access from anywhere to a server, in which resides a software and a robust database that facilitates the registration of each and every one of the movements of the administration of tournaments and competitions and the final result is, competition automatically generate keys and manage cash flows to control the capital assets of the Association. To achieve this result, we designed an algorithm that can: easily create groups of basic parameters such as filtering competition category, weight and gender, enter scores by combat and finally, assess the revenues and expenditures generated. The results of using this system reveal a rapid and efficient conduct of the association processes, allowing managers to take timely decisions and to review all information from anywhere and any time

CAPÍTULO I

INTRODUCCIÓN

1.1- Antecedentes

La Asociación de Tae Kwon Do de Pichincha, fundada en julio de 1983, es una institución de Derecho Privado, es la entidad máxima del Tae Kwon Do en la Provincia de Pichincha, cuya finalidad es: fomentar, planificar, desarrollar, supervisar, controlar, autorizar y representar al Tae Kwon Do en la Provincia, con sujeción a la Ley de Educación Física, Deportes y Recreación, a los Estatutos de la Concentración Deportiva de Pichincha, a los Estatutos de la Federación Ecuatoriana de Tae Kwon Do y sus Reglamentos.

Su sede se encuentra en la ciudad de Quito, se le conoce por sus siglas ATKPD[1]. La ATKPD es una Asociación que se encuentra en constante renovación para la satisfacción de entrenadores, alumnos y personal administrativo, motivo por el cual los procesos y necesidades deben ser modificados constantemente.

1.2- Planteamiento del Problema

Gracias a la recolección de información a través de entrevistas realizadas al Personal Administrativo y Directivos de la Asociación, se puede constatar que

[1] Asociación de Tae Kwon Do de Pichincha

actualmente, la ATKDP posee únicamente un sistema encargado de administrar el montaje de las diferentes llaves y grupos de competidor@s que participan en cada torneo, pero el resto de procesos que intervienen en la Administración de la ATKDP y de Torneos se maneja de manera manual.

Es por este motivo que, la asociación necesita no solo tener automatizado una parte del proceso de competencia sino automatizar todos y cada uno de los procesos de manera integral, desde la inscripción de clubes de Tae Kwon Do hasta el registro de ingresos y egresos que tienen por cada torneo siendo éstos de diferentes rubros.

Además la aplicación de herramientas actuales para poder generar no solo un sistema de Escritorio sino ir más allá proyectando un Sistema WEB[2] que permita no solo a la Asociación sino a cada Club, a cada deportista, entrenador y administrador; mantener la información de manera organizada, poder acceder a la misma de forma oportuna y tener registro de todas y cada una de las participaciones que han tenido desde cualquier lugar del mundo que tenga acceso a Internet.

El Sistema planteado permite no solo administrar todos y cada uno de los Torneos sino que también permitirá que los clubes inscritos puedan incluir a sus competidores desde la WEB, en periodos de tiempo establecidos por la Asociación; puedan generar además los grupos, llaves y enfrentamientos

[2] (FUENTE: http://astursistemas.com/Articulos.php?articulo_no=28&) La palabra WEB es el término más usado para referirse al World Wide Web que es la Red Mundial de páginas o Documentos de texto entrelazados. Un documento entrelazado no es más que un documento que contiene enlaces a otros documentos o páginas de texto. También se le conoce con el nombre de documento de Hipertexto

automáticamente; los mismos que aparecerán en la Pagina WEB publicada, de tal forma que todos estén al tanto de los resultados y demás información de cada competencia.

Esta aplicación también maneja la administración de todos los módulos del Sistema, así como los ingresos y egresos que se gestionen, llevando un control mayor del factor y recurso económico.

1.3- Justificación

La falta de un registro digital de clubes con sus respectivos entrenadores y deportistas, así como de los diferentes torneos y competencias con datos informativos históricos sobre las mismas; control de ingresos y egresos exactos por parte de la ATKDP; inscripción ágil y segura por parte de los clubes en los diferentes torneos y competencias realizadas por la ATKDP; genera la necesidad de emprender un Proyecto que permita desarrollar un Sistema Informático para la Administración de la ATKDP.

Este Sistema permitirá a la ATKDP llevar un Registro de todos los Clubes de Tae Kwon Do de Pichincha con sus respectivos entrenadores y deportistas, permitirá un Control de Ingresos y Egresos Económicos de la misma y almacenará en una base de datos integrada, toda la información relacionada con inscripciones por parte de los clubes, torneos y competencias que lleve a cabo la ATKDP.

Con el Desarrollo de este proyecto, el beneficio que podrá alcanzar la ATKDP será de suma importancia, ya que podrá tener un registro exacto sobre los clubes de Tae Kwon Do de Pichincha así como de sus deportistas y entrenadores; información exacta de los deportistas con datos históricos sobre clubes, torneos en los que han participado, así como de las distinciones conseguidas; de la misma manera información histórica sobre los torneos y competencias realizadas con nombres de ganadores clasificados por categorías, sexo y peso; control del flujo de efectivo y reportes varios que permitan a la ATKDP tomar decisiones acertadas de manera oportuna.

1.4- Objetivos

1.4.1- Objetivo general

Desarrollar una aplicación Web que permita Administrar la Asociación de Tae Kwon Do de Pichincha utilizando Metodología eXtreme Programming.

1.4.2- Objetivos específicos

- Realizar el levantamiento y análisis de requerimientos de la Asociación para desarrollar cada uno de los módulos[3].
- Diseñar un Sistema Web que permita la administración, registro de ingresos y egresos y control de todos los procesos realizados por la ATKDP con referencia a torneos y campeonatos.
- Crear y estandarizar una base de datos para el registro e integración de información actual.

[3] (FUENTE: http://es.wikipedia.org/wiki/M%C3%B3dulo) Un módulo es un componente auto-controlado de un sistema, dicho componente posee una interfaz bien definida hacia otros componentes

- Generar todos los reportes necesarios que permitan la verificación oportuna de datos ingresados en el sistema mejorando la toma de decisiones por parte de la Administración.
- Realizar varias pruebas en cada uno de los módulos para verificar la aceptación del usuario según sugiere la metodología XP

1.5- Alcance

El alcance de este proyecto comprende el desarrollo de una Aplicación Web para la Asociación de Tae Kwon Do de Pichincha, la misma que comprende los siguientes módulos:

- Módulo de Inscripción e Ingreso de Información Preliminar
  - Registro de clubes administrados por la ATKDP.
  - Registro de entrenadores pertenecientes a los clubes administrados por la ATKDP.
  - Registro de Categorías Reglamentarias para competir en un torneo organizado por la ATKDP.
  - Registro de deportistas pertenecientes a los clubes administrados por la ATKDP.
  - Registro de torneos, campeonatos o selectivos organizados por la ATKDP.

- Módulo de Administración de Torneos, Campeonatos o Selectivos
  - Inscripción de clubes con los deportistas a participar en un determinado torneo, competencia o selectivo organizado por la ATKDP.

- Elaboración automática de las llaves para un determinado torneo, competencia o selectivo organizado por la ATKDP.
- Registro de puntaje obtenido en cada uno de los combates para un determinado torneo, competencia o selectivo organizado por la ATKDP.
- Registro de ganadores en cada una de las categorías para un torneo, competencia o selectivo organizado por la ATKDP.
- Consulta de reportes de un torneo, competencia o selectivo organizado por la ATKDP.

- Módulo de Administración de perfiles de Usuario.
  - Desarrollar interfaz para registro de usuarios y perfiles
  - Desarrollar interfaz para asignación de permisos.
  - Desarrollar interfaz para mantenimiento de permisos, usuarios, perfiles y menús.

- Módulo de Administración Financiera.
  - Ingreso de Pagos por concepto de carnetización o inscripción de los deportistas en un torneo, campeonato o selectivo organizado por la ATKDP.
  - Ingreso de Pagos por concepto de inscripción anual de clubes en la ATKDP.
  - Ingreso de valores por conceptos varios a la ATKDP.
  - Egreso de valores por conceptos varios de la ATKDP.
  - Consulta de reportes financieros de la ATKDP

CAPÍTULO II

MARCO TEÓRICO

Para la elaboración de un Sistema Informático flexible y escalable, es necesario una Metodología de Desarrollo de Software así como también, herramientas que permitan la culminación del trabajo de acuerdo al tiempo planificado, presupuesto establecido, delegación de funciones y responsabilidades.

De acuerdo a varios análisis e investigaciones realizadas, un sistema como el planteado, puede desarrollarse mediante una Metodología Ágil; la cual permita reducir tiempo y elaborar un software de calidad con un equipo de desarrollo pequeño.

Para el desarrollo del Sistema Informático de Administración de la Asociación de Tae Kwon do de Pichincha, la metodología seleccionada y las herramientas de apoyo seleccionadas son:

- Metodología de Desarrollo eXtreme Programming.
- Ambiente Web como contexto de programación.
- IDE[4] de desarrollo informático Visual Studio .NET 2005.
- SQL Server 2000 como almacenamiento de información

[4] Entorno de Desarrollo Integrado

A manera de conceptualización, en las siguientes secciones se describe claramente en qué consisten los Sistemas Web; así como la metodología XP. De la misma manera se revisan breves conceptos de las herramientas de desarrollo que se consideran para la construcción del sistema y del motor de base de datos.

2.1- Sistemas en ambientes WEB

2.1.1- Desarrollo de Aplicaciones Web

La evolución de Internet como red de comunicación global, además del surgimiento y desarrollo de la WEB como servicio imprescindible para compartir información, ha creado un excelente espacio para la interacción del hombre con la información WEB, fortaleciendo las bases para el desarrollo de una herramienta integradora de los servicios existentes en Internet.

Los sistemas Web, como expresión de sistemas de información, deben poseer los siguientes componentes:

- Usuarios.
- Mecanismos de entrada y salida de la información.
- Almacenes de datos, información y conocimiento.
- Mecanismos de recuperación de información.

Se puede definir entonces como sistema de información al conjunto de elementos relacionados y ordenados, según ciertas reglas que aportan al sistema

objeto; es decir, a la organización a la que sirve y que marca sus directrices[5] de funcionamiento y la información necesaria para el cumplimiento de sus fines; para ello, debe recoger, procesar y almacenar datos, procedentes tanto de la organización como de fuentes externas, con el propósito de facilitar su recuperación, elaboración y presentación.

Actualmente, los sistemas de información se encuentran al alcance de las grandes masas de usuarios por medio de Internet; así se crean las bases de un nuevo modelo, en el que los usuarios interactúan directamente con los sistemas de información para satisfacer sus necesidades de información. Además, las aplicaciones web se han convertido en pocos años en complejos sistemas con interfaces[6] de usuario cada vez más parecidas a las aplicaciones de escritorio, dando servicio a procesos de negocio de considerable envergadura y estableciéndose sobre ellas, requisitos estrictos de accesibilidad y respuesta. Esto ha exigido reflexiones sobre la mejor arquitectura y las técnicas de diseño más adecuadas.

2.1.2- Requerimientos para desarrollar una aplicación WEB.

Una aplicación WEB requiere de algunos niveles para su implementación y funcionamiento óptimo. Estos niveles son:

a) **Nivel de presentación**: es el encargado de generar la interfaz de usuario en función de las acciones llevadas a cabo por el mismo.

[5] (FUENTE: http://www2.valledelcauca.gov.co/SIISVC/documentos/Presentaciones/MECI/glosario/glosario.htm) Instrucciones o normas generales para la ejecución de algo

[6] Conexión entre dos entidades, en este caso entre un sistema informático y la persona que lo usa

b) **Nivel de negocio**: contiene toda la lógica que modela los procesos de negocio y es donde se realiza todo el procesamiento necesario para atender a las peticiones del usuario.

c) **Nivel de administración de datos**: encargado de hacer persistente toda la información, suministra y almacena información para el nivel de negocio.

Los dos primeros y una parte del tercero (el código[7] encargado de las actualizaciones y consultas), suelen estar en el servidor[8] mientras que la parte restante del tercer nivel se sitúa en la base de datos [9]

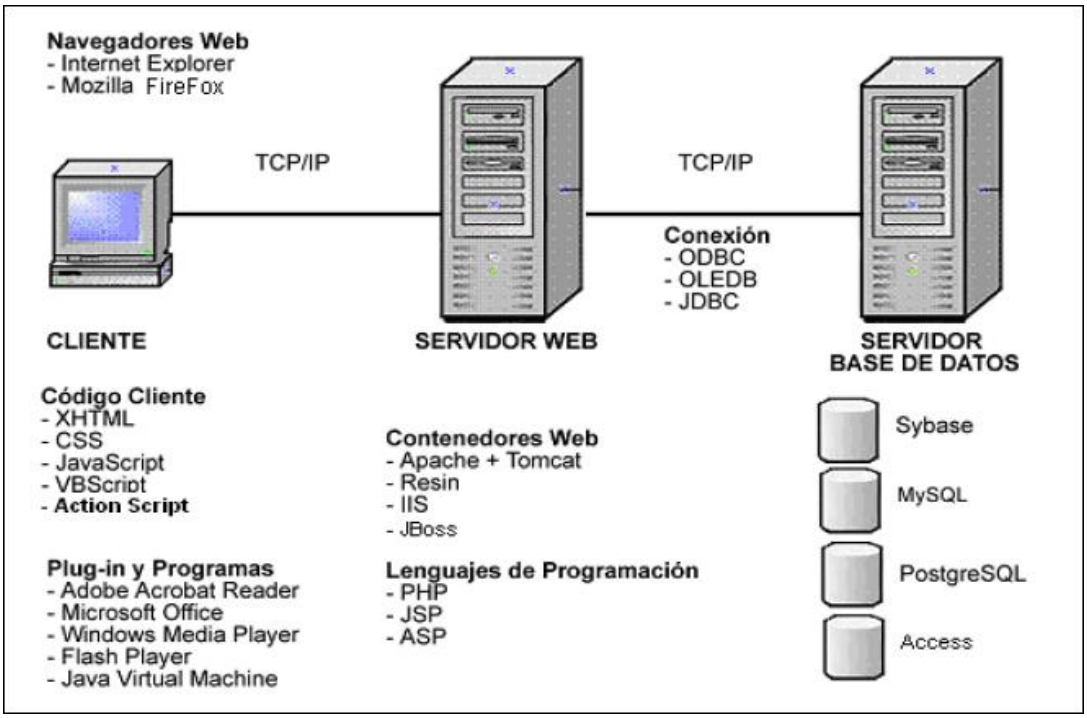


Figura 2.1: Arquitectura de una aplicación WEB[10]

[7] (FUENTE: http://es.wikipedia.org/wiki/C%C3%B3digo_(inform%C3%A1tica) Conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar dicho programa

[8] (FUENTE: http://www.economicas-online.com/glosarios/terminos.htm) Computadora que suministra información a través de una red a otras computadoras

[9] (FUENTE: http://es.wikipedia.org/wiki/Base_de_datos)Es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso

[10] Tomado de la publicación “Estándares para el uso de herramientas y plataformas de aplicaciones WEB”, página 5.

2.1.3- Beneficios de las aplicaciones WEB

Una aplicación WEB posee múltiples beneficios en cuanto a su operatividad y disponibilidad, varios de ellos son:

- **Compatibilidad multiplataforma[11].** Las aplicaciones web tienen un camino mucho más sencillo para la compatibilidad multiplataforma que las aplicaciones de software descargables.
- **Actualización.** Las aplicaciones basadas en web están siempre actualizadas con el último cambio o modificación realizada, sin requerir que el usuario tome acciones o interfiera con su trabajo para descargar una actualización o realice procedimientos de instalación que si no son completamente conocidos por el usuario puedan afectar al sistema.
- **Acceso Inmediato.** Las aplicaciones basadas en web no necesitan ser descargadas, instaladas y configuradas. Se accede a una cuenta online[12] y éstas están habilitadas para su uso.
- **Menos requerimientos de memoria.** Las aplicaciones basadas en web tienen mucha menor demanda de memoria RAM[13] por parte del usuario final que los programas instalados localmente, ya que al residir y correr en servidores WEB evitan el uso local de memoria dejando mayor cantidad de ella para correr otro tipo de aplicaciones sin que se vea afectado el rendimiento.

---

[11] (FUENTE: http://www.sitiosespana.com/paginas/diccionario_informatica/c.htm)Programa o dispositivo que puede utilizarse sin inconvenientes en distintas plataformas de hardware y sistemas operativos

[12] (FUENTE: http://www.perugrafico.com/articulos/diccionario-de-marketing.htm) Se está online cuando se efectúa la conexión entre dos ordenadores en tiempo real, sin embargo, la expresión se refiere, en la mayoría de los casos, a cuando estos ordenadores se conectan vía Internet

[13] RAM significa Memoria de Acceso Aleatorio; es el lugar lógico donde se almacenan temporalmente las aplicaciones que se están ejecutando en el computador, al momento de apagar el equipo se vacía y queda de nuevo lista para levantar aplicaciones temporales en un nuevo encendido del computador.

- **La información viaja en línea.** Con las aplicaciones basadas en web se logra acceder a información actualizada desde cualquier parte del mundo.
- **Múltiples usuarios concurrentes.** Las aplicaciones basadas en web pueden realmente ser utilizadas por múltiples usuarios al mismo tiempo. No hay más necesidad de compartir pantallas o solicitar permisos para ingresar a una interfaz en la que otro usuario se encuentra trabajando, pueden ver e incluso editar el mismo documento de manera conjunta varios usuarios en el mismo momento.
- **Desarrollar aplicaciones en cualquier lenguaje.** Una vez que las aplicaciones han sido separadas de computadoras locales y sistemas operativos específicos pueden también ser escritas en prácticamente cualquier lenguaje de programación. Ya que las aplicaciones web son esencialmente una colección de programas más que un simple programa, ellas podrían ser escritas en cualquier lenguaje de programación que esté disponible.

2.1.4- Limitaciones de las aplicaciones WEB

- **Inseguridad de la información.** Se sabe que en las aplicaciones WEB la información viaja a través del Internet por ende es susceptible a cualquier tipo de intromisión por parte de agentes externos que en ocasiones podrían causar la perdida de la misma o a su vez el plagio de ella.
- **Dependencia de factores externos para un buen rendimiento.** El tipo de conexión y ancho de banda[14] de cada cliente es un factor muy importante

[14] (FUENTE: http://www.sitiosespana.com/paginas/diccionario_informatica/)volumen de información que puede circular por un medio físico de comunicación de datos

al momento de acceder a una aplicación basada en la web, ya que si se tiene un deficiente ancho de banda muy posiblemente la aplicación presente lentitud y por ende un bajo rendimiento.

2.2- Metodologías Ágiles

2.2.1- Introducción

En la última década ha comenzado un creciente interés en Metodologías de Desarrollo Ágiles. Hasta hace poco el proceso de Desarrollo de Software llevaba énfasis en el control del proceso mediante una rigurosa definición de roles, actividades y artefactos, incluyendo modelado y documentación detallada.

Este esquema "tradicional" para abordar el desarrollo de software ha demostrado ser efectivo y necesario en proyectos de gran tamaño (respecto a tiempo y recursos). Sin embargo, en muchos de los proyectos actuales donde el entorno del sistema es muy cambiante, y en donde se exige reducir drásticamente los tiempos de desarrollo, las metodologías ágiles emergen como una posible respuesta para llenar ese vacío metodológico; por estar especialmente orientadas para proyectos pequeños.

Las metodologías ágiles constituyen una solución a medida para ese entorno, aportando una elevada simplificación que a pesar de ello no renuncia a las prácticas esenciales para asegurar la calidad del producto.

2.2.2- Justificación para uso de una metodología ágil

Las metodologías ágiles están enfocadas en proyectos pequeños con equipos de desarrollo reducidos. Estas metodologías dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Este enfoque está mostrando su efectividad en proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo pero manteniendo la calidad.

2.2.3- Diferencias entre Metodologías Ágiles y Tradicionales

Tabla 2.1: Diferencia entre Metodologías de Desarrollo de Software Ágiles y Tradicionales

METODOLOGÍAS ÁGILES	METODOLOGÍAS TRADICIONALES
Basadas en heurísticas[15] provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares[16] seguidos por el entorno de desarrollo
Preparadas para cambios durante el proyecto	Cierta resistencia a cambios
Impuestas internamente (Equipo de Desarrollo)	Impuestas Externamente
Proceso menos controlado con pocos principios	Proceso mucho más controlado con numerosas políticas/normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
Grupos pequeños (menos de 10 integrantes) trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Menos énfasis en la Arquitectura del Software	La Arquitectura del Software es esencial y se expresa mediante modelos

[15] (FUENTE: http://es.wikipedia.org/wiki/Heur%C3%ADstica)Se denomina **heurística** a la capacidad de un sistema para realizar de forma inmediata innovaciones positivas para sus fines

[16] (FUENTE: http://www.ciat.cgiar.org/agroempresas/sistema_cj/glosario.htm)Son acuerdos documentados que contienen especificaciones técnicas u otros criterios específicos para ser usados como referentes, guías o definiciones de características, para asegurar que materiales, productos, procesos y servicios son obtenidos o han sido realizados de acuerdo a sus propósitos

2.3- Metodología XP (eXTREME PROGRAMMING)

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios.

La Gráfica a continuación (Figura 2.2) muestra los largos ciclos de desarrollo en cascada (a) a ciclos más cortos (b) y a la mezcla que hace XP (c);

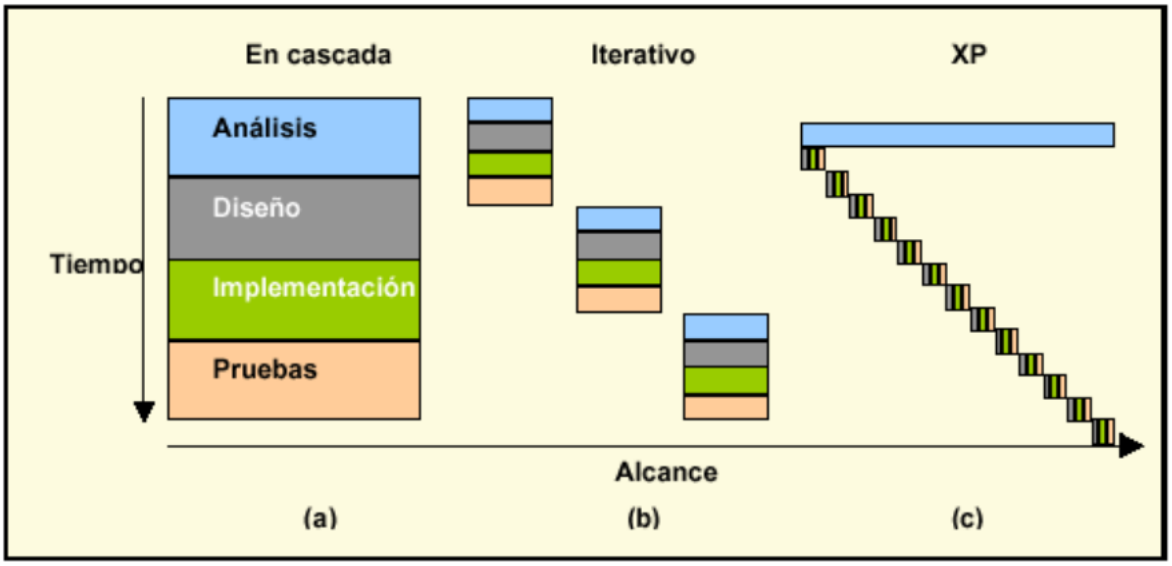


Figura 2.2: Evolución de los ciclos de la Metodología XP.[17]

IBM[18] realizó una encuesta de sus desarrolladores en el año 2000 con respecto a la Metodología XP, los resultados se pueden ver a continuación:

[17] Tomado del artículo "Introducción a EXTREME PROGRAMMING, página 4"

A) Lo he probado y no me gusta nada

B) Es una mala idea, no puede funcionar nunca

C) Es una buena idea, pero no funcionará.

D) Lo he probado y me gusta mucho

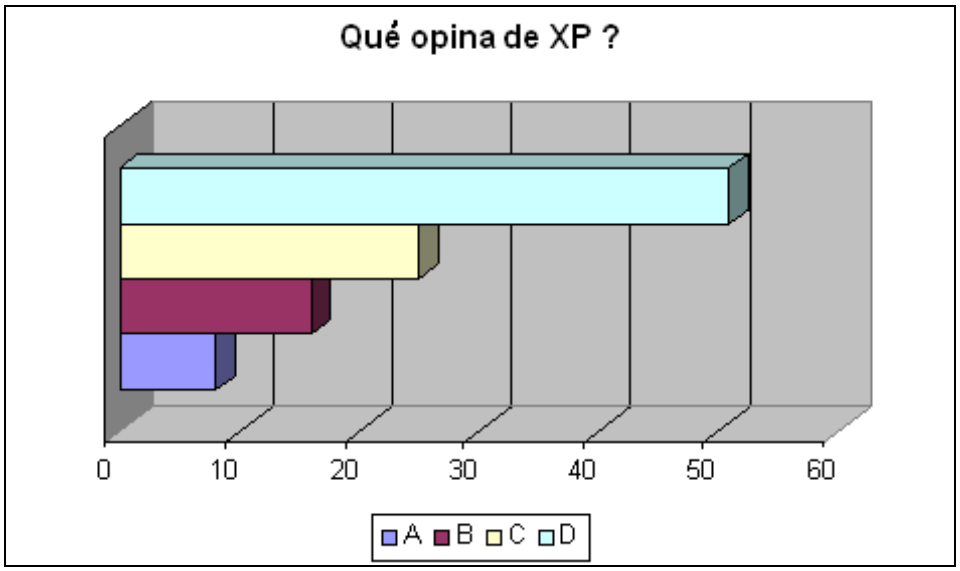


Figura 2.3: Resultado de una encuesta realizada por IBM en el año 2000[19]

2.3.1- Valores de la Metodología XP

La programación extrema, lejos de ser un proceso no controlado, es una metodología muy disciplinada y que se apoya en cuatro valores fundamentales:

- **Comunicación:** Se hace énfasis en que la comunicación, para ser efectiva, debe involucrar a todos los participantes en el proyecto, y debe ser libre y sincera.

[18] INTERNATIONAL BUSINESS MACHINES, fábrica de computadores y software
[19] Tomado del artículo "Programación Extrema, página 3"

- **Simplicidad:** Nunca debe perderse de vista que el objetivo de un proyecto es proporcionar valor al cliente; no es demostrar el desconocimiento técnico del equipo ni construir una aplicación que resuelva más problemas que los del cliente.

- **Realimentación:** No se puede dirigir adecuadamente un proceso si no se dispone de realimentación permanente sobre su progreso. La realimentación puede provenir del cliente, de los programadores, de herramientas automáticas, entre otras.

- **Coraje:** A veces, hacer lo que es correcto requiere valor. Por ejemplo, hay que tener coraje para reescribir código que funciona pero ha alcanzado su límite de escalabilidad[20].

2.3.2- Doce Prácticas de la Metodología XP

Los valores antes mencionados dan origen a cinco principios básicos:

1. Conseguir realimentación rápida,
2. No complicar las cosas con suposiciones (asumir que las cosas son simples)
3. Realizar cambios incrementales
4. Abrazar el cambio
5. Generar productos de calidad.

[20] (FUENTE: http://www2.esmas.com/emprendedor/glosario/)Capacidad de un software o de un hardware de crecer, adaptándose a nuevos requisitos conforme cambian las necesidades del negocio

Los cinco principios se manifiestan a través de las prácticas de la programación extrema, que son las actividades que el Equipo de Desarrollo lleva a cabo cada día, estas actividades se detallan en doce prácticas que se agrupan en cuatro categorías y son las siguientes:

2.3.2.1- Retroalimentación a escala fina.

- **El principio de pruebas**: Se tiene que establecer un período de pruebas de aceptación del programa donde se definirán las entradas al sistema y los resultados esperados de estas entradas. Es recomendable automatizar estas pruebas para poder hacer varias simulaciones del sistema en funcionamiento.

- **Proceso de planificación**: En esta fase, el usuario tiene que escribir sus necesidades, definiendo las actividades que realizará el sistema. Se creará un documento llamado Historias del Usuario[21]. Entre 15 y 45 historias (todo dependiendo de la complejidad del problema) se consideran suficientes para formar el llamado Plan de Iteración, el cual define de forma específica los tiempos de entrega de la aplicación para recibir retroalimentación por parte del usuario.

- **El cliente en el sitio**: el cliente debe determinar los requerimientos, definir la funcionalidad, señalar las prioridades y responder las preguntas de los programadores. Esta fuerte interacción cara a cara con el programador disminuye el tiempo de comunicación y la cantidad de documentación,

[21] (FUENTE: http://es.wikipedia.org/wiki/Historias_de_usuario)Una **historia de usuario** es una representación de un requerimiento de software escrito en una o dos frases utilizando el lenguaje común del usuario

junto con los altos costos de su creación y mantenimiento. El cliente estará con el equipo de trabajo durante toda la realización del proyecto.

- **Programación en parejas**: Uno de los principios más radicales y en el que la mayoría de gerentes de desarrollo pone sus dudas. Requiere que todos los programadores XP escriban su código en parejas, compartiendo una sola máquina. De acuerdo con los experimentos, este principio puede producir aplicaciones mejores, con iguales o menores costos.

2.3.2.2- Proceso continuo en lugar de por lotes.

- **Integración continua**: Permite al equipo hacer un rápido progreso implementando las nuevas características del software. En lugar de crear versiones estables de acuerdo a un cronograma establecido, los equipos de programadores XP pueden reunir su código y reconstruir el sistema varias veces al día.

- **Refactorización**: Permite a los equipos de programadores XP mejorar el diseño del sistema a través de todo el proceso de desarrollo. Los programadores evalúan continuamente el diseño y recodifican lo necesario. La finalidad es mantener un sistema enfocado a proveer el valor de negocio mediante la minimización del código duplicado y/o ineficiente.

- **Entregas pequeñas**: Colocar un sistema sencillo en producción rápidamente que se actualiza de forma oportuna y constante permitiendo que el verdadero valor de negocio del producto sea evaluado en un

ambiente real. Estas entregas no pueden pasar las 2 o 3 semanas como máximo

2.3.2.3- Entendimiento compartido.

- **Diseño simple**: Se basa en que el mayor valor de negocio es entregado por el programa más sencillo que cumpla los requerimientos, se enfoca en proporcionar un sistema que cubra las necesidades inmediatas del cliente.

- **Metáfora**: Desarrollada por los programadores al inicio del proyecto, define una historia y el cómo funciona el sistema completo. XP estimula historias, que son breves descripciones de un trabajo de un sistema. La metáfora expresa la visión evolutiva del proyecto que define el alcance y propósito del sistema. Las tarjetas CRC (Clase, Responsabilidad y Colaboración) también ayudan al equipo a definir actividades durante el diseño del sistema. Cada tarjeta representa una clase en la programación orientada a objetos y define sus responsabilidades (lo que ha de hacer) y las colaboraciones con las otras clases (cómo se comunica con ellas).

- **Propiedad colectiva del código**: Un código con propiedad compartida. Nadie es el propietario de nada, todos son el propietario de todo. XP argumenta que mientras haya más gente trabajando en un módulo, menos errores aparecerán.

- **Estándar de codificación**: Define la propiedad del código compartido así como las reglas para escribir y documentar el código y la comunicación

entre diferentes partes de código desarrolladas por diversos equipos. Los programadores las han de seguir de tal manera que el código en el sistema se vea como si hubiera estado escrito por una sola persona.

2.3.2.4- Bienestar del programador.

- **La semana de 40 horas**: La programación extrema sostiene que los programadores cansados escriben código de menor calidad. Minimizar las horas extras y mantener los programadores frescos, generará código de mayor calidad.

2.3.3- Fases de Desarrollo de la Metodología XP

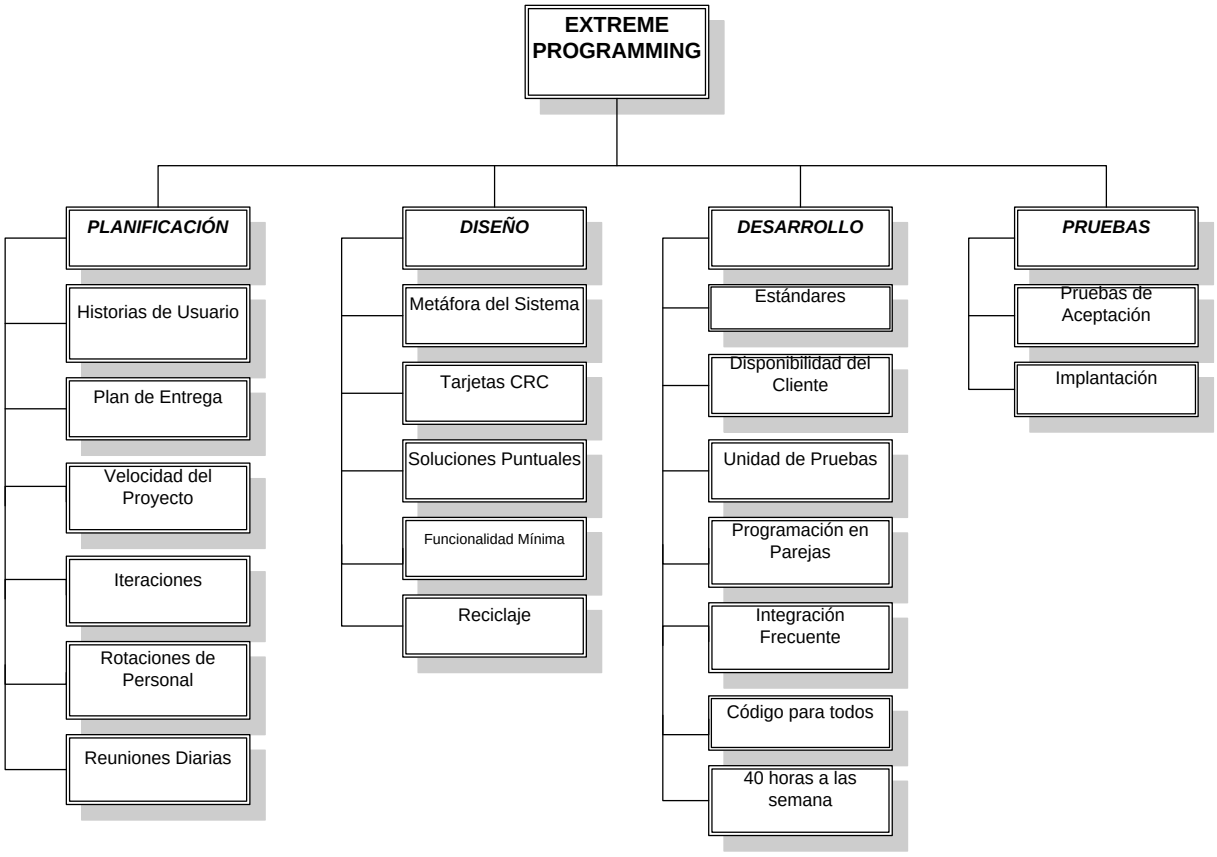


Figura 2.6: Fases de Desarrollo de la Metodología XP

2.3.3.1- Planificación

XP plantea la planificación como un diálogo permanente entre la parte empresarial y técnica del proyecto, en la que los primeros decidirán el alcance, la prioridad, la composición de las versiones y la fecha de las mismas.

En cuanto a los técnicos, son los responsables de estimar la duración requerida para implementar las funcionalidades deseadas por el cliente, informar sobre las consecuencias de determinadas decisiones, organizar la cultura de trabajo y, finalmente, realizar la planificación detallada dentro de cada versión.

- Se utilizan historias de usuario, que son las necesidades escritas por el cliente con la ayuda de los diseñadores; las cuales deberán ser resueltas con el Sistema Informático.
- Se crean los planes de entregas, los cuales estiman el tiempo de desarrollo de las historias de usuario.
- Se controla la velocidad del proyecto, se usa para planificar cuántas historias de usuario estarán implementadas antes de una fecha dada.
- Se lleva a cabo la planificación de iteración; es decir se identifica las historias de usuario que se van a desarrollar en una iteración específica.
- Se realiza la rotación de funciones al personal, esto ayuda a que todo el equipo de desarrollo conozca cómo funciona el sistema en general y distribuye el trabajo de una manera más equitativa.
- Se desarrollan reuniones diarias, con el fin de facilitar la comunicación entre el grupo de trabajo y la exposición de los diferentes problemas.

2.3.3.2- Diseño

XP establece unas recomendaciones o premisas a la hora de abordar esta etapa:

- Se escoge una metáfora de sistema, con el objetivo de facilitar el manejo consistente de los nombres de las clases[22] y los métodos[23].
- Se usan tarjetas CRC (cargo, responsabilidad, colaboración), nos permiten trabajar con una metodología basada en objetos[24].
- Se proponen soluciones a problemas técnicos o de diseño.
- Se ignoran las funcionalidades extra que podrían incorporarse al proyecto, es decir, se trata de que el equipo se centre en lo principal y lo que realmente solicita el cliente.
- Se remueve la redundancia, se eliminan las funcionalidades no necesarias y se rejuvenecen los diseños obsoletos.

2.3.3.3- Desarrollo

Esta etapa debe reunir las siguientes características o cualidades:

- Se utilizan estándares para escribir el código.
- Se debe tener al cliente siempre disponible, no sólo para ayudar al equipo de desarrollo sino para formar parte de él
- Se crean pruebas antes de empezar a codificar, esto hará más sencilla la implementación del código.

[22] (FUENTE: http://www.calinsoft.com/2008/08/definiciones-de-programacion-orientada-a-objetos.html) Una clase es una agrupación de datos (variables o campos) y de funciones (métodos) que operan sobre esos datos. A estos datos y funciones pertenecientes a una clase se les denomina variables y métodos o funciones miembro. La clase es el patrón o modelo para crear objetos

[23] (FUENTE: http://es.wiktionary.org/wiki/m%C3%A9todo)Procedimiento, técnica o manera de hacer algo, en especial si se hace siguiendo un plan, o de forma sistemática, ordenada y lógica

[24] (FUENTE: http://www.calinsoft.com/2008/08/definiciones-de-programacion-orientada-a-objetos.html) Es un paquete de variables y métodos relacionados. Los objetos en software frecuentemente son la abstracción de las propiedades y comportamientos de una entidad del mundo real

- Todo debe programarse en parejas utilizando una sola computadora, esto incrementa la calidad de software sin afectar tiempos de entrega.
- Se utiliza integración frecuente, esto con el fin de trabajar siempre con la última versión
- Todo el código es común a todos, esto permite que un programador contribuya al desarrollo de cualquier parte del proyecto.
- Se deja la optimización para el final, una vez que el código requerido este completo
- Trabajar 40 horas semanales, esto con el objetivo de que el equipo de desarrollo no se agote y disminuya la calidad del trabajo.

2.3.3.4- Pruebas

- Se crean pruebas de aceptación a partir de las historias de usuario.
- El cliente es el responsable de revisar, tanto las pruebas de aceptación, como los resultados obtenidos al ser éstas aplicadas.
- Una historia de usuario no se considera lista hasta que haya pasado todas sus pruebas de aceptación

2.4- Visual Studio .NET 2005 Professional Edition

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias

para muchos otros. Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones Web, así como Servicios Web en cualquier entorno que soporte la plataforma .NET. Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas y dispositivos móviles

Las ediciones Express se han diseñado para principiantes, aficionados y pequeños negocios, todas disponibles gratuitamente a través de la página de Microsoft se incluye una edición independiente para cada lenguaje: Visual Basic, Visual C++, Visual C#, Visual J# para programación .NET en Windows, y Visual Web Developer para la creación de sitios web ASP.NET.

Las ediciones Express carecen de algunas herramientas avanzadas de programación así como de opciones de extensibilidad.[25]

2.5- SQL Server 2000

SQL Server 2000 es un sistema de gestión de bases de datos relacionales (Relational Database Management System) diseñado para trabajar con grandes cantidades de información y con la capacidad de cumplir con los requerimientos de proceso de información para aplicaciones comerciales y sitios Web. Ofrece el soporte de información para las tradicionales aplicaciones Cliente/Servidor, las cuales están conformadas por una interfaz a través de la cual los clientes acceden a los datos por medio de una LAN[26].

[25] Fuente de consulta http://www.microsoft.com/Microsoft_Visual_Studio [Microsoft Visual Studio: definición]
[26] Red de Área Local

La plataforma .NET exige un gran porcentaje de distribución de recursos, desconexión a los servidores de datos y un entorno descentralizado, para ello sus clientes deben ser livianos, tales como los navegadores de Internet, los cuales accederán a los datos por medio de servicios como el Internet Information Services (IIS).

SQL Server puede ejecutarse sobre redes basadas en Windows Server así como sistema de base de datos de escritorio en máquinas Windows NT Work-station, Windows Millenium, Windows XP.

Los entornos Cliente / Servidor están implementados de tal forma que la información se guarda de forma centralizada en un servidor, siendo este responsable del mantenimiento de la relación entre los datos, asegurarse del correcto almacenamiento de los datos, establecer restricciones que controlen la integridad de datos, entre otras.

Del lado cliente, este corre típicamente en distintas computadoras las cuales acceden al servidor a través de una aplicación, para realizar la solicitud de datos los clientes emplean el Structured Query Language (SQL), este lenguaje tiene un conjunto de comandos que permiten indicar la información que se desea recuperar o modificar.[27]

[27] Microsoft. SQL Server Architecture. Microsoft Press, USA, 2005

CAPÍTULO III

PLANIFICACIÓN DEL PROYECTO

La Planificación del proyecto informático permite programar el tiempo, los recursos y alcances para cada etapa de desarrollo, de esta manera se permite establecer los objetivos basados en un plan de entrega previamente establecido y analizado. La especificación de requerimientos del sistema permite ver las necesidades y tener objetivos claros que se pretende cumplir para el presente proyecto. Los requerimientos del usuario se encuentran en las historias de usuario, de la misma manera la programación del tiempo y recursos a emplear en cada historia de usuario, se encuentran proyectadas en el plan de entrega.

3.1- Apreciación global

El Desarrollo de un Sistema Administrativo para ATKDP está formado por una serie de procesos que necesitan estar analizados para tener una visión de las funciones que se encuentran en cada uno de los módulos propuestos, por tal razón, en este capítulo se detallarán las historias de usuario, la planificación para su desarrollo y los requisitos necesarios para la ejecución de la aplicación web.

3.2- Descripción del Producto

En el sistema informático a desarrollar, para el manejo de información, cada usuario desempeña una función que permite tener un control eficiente de los procesos que tiene el Sistema de Administración de la ATKDP. El siguiente gráfico muestra los procesos principales de la Administración de la ATKDP.

PROCESO DE INSCRIPCIÓN Y CARNETIZACIÓN

Entrenador del Club X se acerca a la Asociación con los datos principales del Club para su registro.

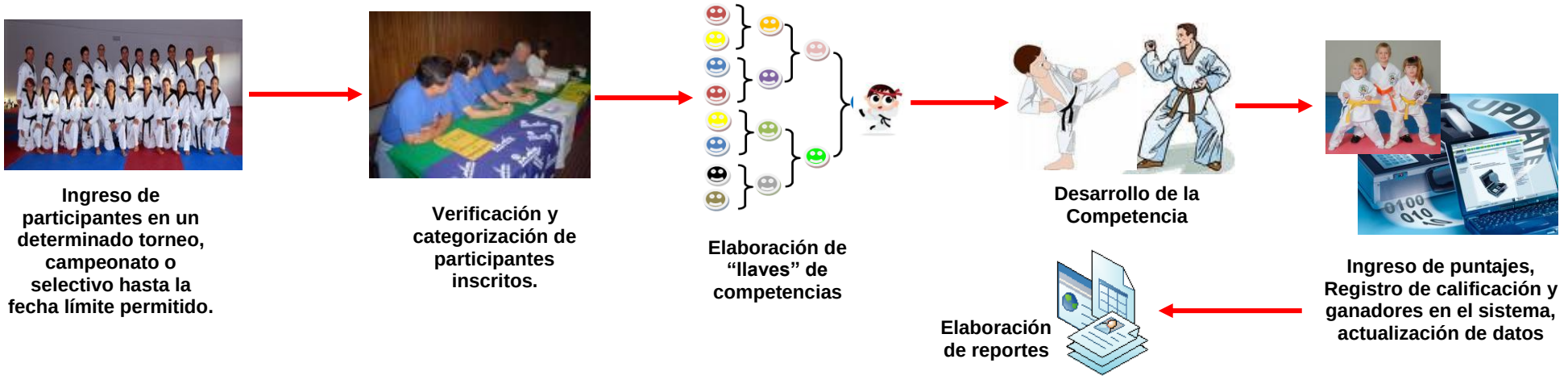
Registro de alumnos, entrenadores y clubes - Inscripción

Carnetización de alumnos y entrenadores inscritos

Pago a la Asociación por concepto de inscripción y carnetización

Registro y actualización de datos por concepto de inscripción, carnetización y pagos en el sistema

PROCESO DE SELECCIÓN DE PARTICIPANTES Y EVALUACIÓN DE TORNEO, CAMPEONATO O SELECTIVO



Ingreso de participantes en un determinado torneo, campeonato o selectivo hasta la fecha límite permitido.

Verificación y categorización de participantes inscritos.

Elaboración de "llaves" de competencias

Desarrollo de la Competencia

Ingreso de puntajes, Registro de calificación y ganadores en el sistema, actualización de datos

Elaboración de reportes

Figura 3.1: Procesos Principales para la Administración de la ATKDP

El sistema que se va a desarrollar debe procesar todos los datos ingresados por los usuarios (ATKDP y clubes), permitiendo reportes históricos y financieros.

La interfaz de usuario debe tener facilidad de interacción para cada módulo, lo cual permitirá al usuario realizar de mejor manera su trabajo.

3.3- Requisitos Funcionales

Los requisitos funcionales se representan a través de las historias de usuario que muestran las necesidades de la ATKDP.

Tabla 3.1: Historia de Usuario I - Ingreso y utilización del sistema por parte del usuario

Historia #1	**Ingreso y utilización del sistema por parte del usuario**
Objetivo: Registrar la información actualizada de los usuarios que utilizan el software y permitir su acceso al sistema de acuerdo a perfiles.	

Descripción: Los nombres y contraseñas de los usuarios son guardados previamente por parte del administrador del sistema; el usuario que desee hacer uso del sistema debe ingresar su nombre y contraseña, con lo cual el sistema realizará una verificación de usuario y de los permisos asignados al mismo y le permitirá ingresar a los módulos cuyo perfil asigne para utilizar el sistema.

Tabla 3.2: Historia de Usuario II - Registro de clubes administrados por la Asociación de Tae Kwon Do de Pichincha (ATKDP).

Historia #2	**Registro de clubes administrados por la Asociación de Tae Kwon Do de Pichincha (ATKDP).**
Objetivo: Registrar la información actualizada de los clubes que administra la ATKDP.	
Descripción: El usuario realiza el registro de los clubes administrados por la Asociación de Tae Kwon Do de Pichincha, la información registrada consta de datos generales del club como son nombre, representante legal, delegados (2), dirección, año de creación del mismo, número de acuerdo de creación, teléfono, dirección de página web, dirección de correo electrónico, usuario, contraseña.	

Tabla 3.3: Historia de Usuario III - Registro de entrenadores correspondientes a los clubes administrados por la ATKDP

Historia #3	**Registro de entrenadores correspondientes a los clubes administrados por la ATKDP**
Objetivo: Registrar la información actualizada de entrenadores de los clubes que pertenecen a la ATKDP.	
Descripción: El usuario realiza el registro de los entrenadores de los clubes que pertenecen a la ATKDP, la información registrada consta de datos generales del entrenador como cédula de identidad, nombre, club al que pertenece, fecha de ingreso al club, sexo, dirección domiciliaria, teléfono del domicilio, teléfono celular, dirección de correo electrónico	

Tabla 3.4: Historia de Usuario IV - Registro de categorías y pesos reglamentarios para competir en un torneo, campeonato o selectivo organizado por la ATKDP

Historia #4	**Registro de categorías y pesos reglamentarios para competir en un torneo, campeonato o selectivo organizado por la ATKDP**
Objetivo: Registrar la información actualizada de las categorías y pesos reglamentarios utilizadas para competir en un torneo organizado por la ATKDP.	
Descripción: El usuario realiza el registro de las categorías y pesos reglamentarios para que un club con sus deportistas asignados, pueda competir en un torneo organizado por la ATKDP, la información registrada consta de datos generales como sexo, nombre de la categoría, tipo de peso, edad.	

Tabla 3.5: Historia de Usuario V - Registro de deportistas correspondientes a los clubes administrados por la ATKDP

Historia #5	**Registro de deportistas correspondientes a los clubes administrados por la ATKDP**
Objetivo: Registrar la información actualizada de los deportistas registrados en la ATKDP.	
Descripción: El usuario realiza el registro de los deportistas registrados en la ATKDP, la información registrada consta de datos generales del deportista como cédula de identidad, nombre, fecha de nacimiento, club al que pertenece, fecha de ingreso al club, fecha de salida del club, tipo de desportista(novato-clasificado), sexo, peso, dirección domiciliaria, teléfono, teléfono celular, dirección de correo electrónico	

Tabla 3.6: Historia de Usuario VI - Registro de torneos, campeonatos o selectivos organizados por la ATKDP

Historia #6	**Registro de torneos, campeonatos o selectivos organizados por la ATKDP**
Objetivo: Registrar la información actualizada de los torneos, campeonatos o selectivos organizados por la ATKDP.	
Descripción: El usuario realiza el registro de los torneos, campeonatos o selectivos organizados por la ATKDP, la información registrada consta de datos generales como nombre del torneo/campeonato/selectivo, fecha de inicio de la misma, fecha de finalización, fecha de inicio de inscripciones, fecha fin de inscripciones, evaluador, lugar en la cual se va a llevar a cabo, categoría(novatos, clasificados, selectivos, mixtas) , tipo(torneo, campeonato, selectivo)	

Tabla 3.7: Historia de Usuario VII - Inscripción de clubes a participar en un determinado torneo, campeonato o selectivo organizado por la ATKDP

Historia #7	**Inscripción de clubes a participar en un determinado torneo, campeonato o selectivo organizado por la ATKDP**
Objetivo: Inscribir a los clubes a participar en un determinado torneo, campeonato o selectivo organizado por la ATKDP.	
Descripción: El usuario realiza la inscripción de su club en el torneo, campeonato o selectivo organizado por la ATKDP, la información necesaria para la inscripción del club consta de datos generales como nombre del torneo/campeonato/selectivo en el cuál se está inscribiendo, nombre del club; entrenador 1, entrenador 2, se ingresa una lista de deportistas que van a participar clasificados por nombre, categoría, sexo y peso, fecha de inicio de inscripciones, el sistema guarda la fecha en la que el club se inscribe, todos los clubes que deseen participar en el torneo/competencia deben realizar la inscripción en el rango de fechas indicado por la ATKDP, el sistema bloqueará automáticamente el ingreso cuando se cumpla la fecha establecida como límite de inscripción; luego el sistema debe permitir imprimir la lista de deportistas para pagar los valores correspondientes por inscripción.	

Tabla 3.8: Historia de Usuario VIII - Elaboración automática de las llaves para un determinado torneo, campeonato o selectivo organizado por la ATKDP

Historia #8	**Elaboración automática de las llaves para un determinado torneo, campeonato o selectivo organizado por la ATKDP**
Objetivo: Elaborar automáticamente las llaves (designación de contrincantes aleatoriamente) para dar inicio a un determinado torneo, campeonato o selectivo organizado por la ATKDP.	
Descripción: El usuario le indica al sistema que realice la elaboración automática de las llaves para dar comienzo a los combates de un determinado torneo/campeonato/selectivo organizado por la ATKDP, las llaves tienen que elaborarse dependiendo del número de deportistas inscritos clasificados por categoría, peso, sexo; de manera aleatoria.	

Tabla 3.9: Historia de Usuario IX - Registro de puntaje obtenido en cada uno de los combates para un determinado torneo, campeonato o selectivo organizado por la ATKDP.

Historia #9	**Registro de puntaje obtenido en cada uno de los combates para un determinado torneo, campeonato o selectivo organizado por la ATKDP.**
Objetivo: Registrar el puntaje obtenido en los combates de un determinado torneo, campeonato o selectivo organizado por la ATKDP.	
Descripción: El usuario realiza el ingreso del puntaje obtenido en cada uno de los combates organizados por la ATKDP, luego de cada fase el sistema automáticamente elabora las llaves para los siguientes combates con los deportistas que siguen en competencia.	

Tabla 3.10: Historia de Usuario X - Registro de ganadores en cada una de las categorías para un torneo, campeonato o selectivo organizado por la ATKDP

Historia #10	**Registro de ganadores en cada una de las categorías para un torneo, campeonato o selectivo organizado por la ATKDP.**
Objetivo: Registrar los ganadores de cada una de las categorías de un torneo, campeonato o selectivo organizado por la ATKDP.	
Descripción: El usuario realiza el ingreso de los ganadores en cada una de las categorías de un torneo, campeonato o selectivo organizado por la ATKDP.	

Tabla 3.11: Historia de Usuario XI - Ingreso de Pago por concepto de carnetización, o Inscripción de los deportistas en un torneo, campeonato o selectivo organizada por la ATKDP

Historia #11	**Ingreso de Pago por concepto de carnetización o Inscripción de los deportistas en un torneo, campeonato o selectivo organizada por la ATKDP.**
Objetivo: Ingresar pago por concepto de carnetización o inscripción de los deportistas en un torneo, campeonato o selectivo organizado por la ATKDP	
Descripción: El usuario realiza el ingreso de los valores por concepto de carnetización o inscripción en una competencia de los deportistas que pertenecen a la ATKDP, se ingresan datos generales como nombre del club, número de deportistas, valor por deportista, valor total de carnetización; el sistema permitirá imprimir un comprobante de pago con el detalle de los rubros pagados.	

Tabla 3.12: Historia de Usuario XII - Ingreso de Pagos por concepto de inscripción anual de clubes en la ATKDP

Historia #12	**Ingreso de Pagos por concepto de inscripción anual de clubes en la ATKDP.**
Objetivo: Ingresar pagos por concepto de inscripción anual de clubes en la ATKDP.	
Descripción: El usuario realiza el ingreso de los valores por concepto de inscripción anual de clubes en la ATKDP, se ingresan datos generales como nombre del club, valor de la inscripción, fecha, responsable y alguna observación que se tenga; el sistema permitirá imprimir un comprobante de pago con el detalle de los rubros pagados.	

Tabla 3.13: Historia de Usuario XIII - Ingreso de valores por conceptos varios a la ATKDP

Historia #13	**Ingreso de valores por conceptos varios a la ATKDP.**
Objetivo: Ingresar valores por conceptos varios a la ATKDP.	
Descripción: El usuario realiza el ingreso de los valores por conceptos varios que tiene la ATKDP, se ingresan datos generales como nombre, detalle del ingreso, valor ingresado, fecha y hora, entre otros datos que sean relevantes a manera de información adicional.	

Tabla 3.14: Historia de Usuario XIV - Egreso de valores por conceptos varios de la ATKDP

Historia #14	**Egreso de valores por conceptos varios de la ATKDP.**
Objetivo: Egresar valores por conceptos varios de la ATKDP.	
Descripción: El usuario realiza el egreso de valores por conceptos varios que tiene la ATKDP, se ingresan datos generales como nombre, detalle del egreso, valor de egreso, beneficiario, fecha y hora, nombre del que emite , nombre del que recibe	

Tabla 3.15: Historia de Usuario XV - Consulta de reportes para un torneo, campeonato o selectivo organizado por la ATKDP

Historia #15	**Consulta de reportes para un torneo, campeonato o selectivo organizado por la ATKDP.**
Objetivo: Consultar diferentes reportes para un torneo, campeonato o selectivo organizado por la ATKDP.	
Descripción: El usuario realiza la consulta de los diferentes reportes para un torneo/campeonato/selectivo organizado por la ATKDP, se ingresan datos que sirvan como filtro para obtener la información requerida como rango de fechas, código o nombre de entrenadores y deportistas, código de competencia, tipo de medallas obtenidas, categoría, sexo, peso, entre otros**.**	

Tabla 3.16: Historia de Usuario XVI - Consulta de reportes financieros de la ATKDP

Historia #16	**Consulta de reportes financieros de la ATKDP.**
Objetivo: Consultar diferentes reportes financieros de la ATKDP.	
Descripción: El usuario realiza la consulta de los diferentes reportes financieros de la ATKDP, se ingresan datos que sirvan como filtro para obtener la información requerida como rango de fechas, usuario, beneficiario, tipo de ingreso o egreso, numero de documento de respaldo o comprobante, entre otros	

3.4- Requisitos no funcionales

Un requisito no funcional especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que éstos corresponden a los requisitos funcionales.

Se han considerado los siguientes requisitos no funcionales:

- **Seguridad:** el sistema deberá mantener la seguridad de información entre los usuarios finales a través de perfiles de usuario. Estos perfiles de usuario están catalogados por prioridad de uso; por ejemplo, los diferentes clubes pueden acceder a la aplicación WEB para inscribir a sus deportistas, por lo tanto tendrán permisos únicamente para ver información relativa al ingreso de deportistas, registro de entrenadores; por lo tanto el

perfil no permitirá que el club edite, registre o elimine información propia de la ATKDP como resultados financieros, registro de competencias, reportes contables, entre otras interfaces que son de exclusividad para el personal que trabaja en la ATKDP.

- **Integridad:** La información ingresada por parte del usuario de este sistema posee integridad de datos y lógica.

  La integridad de datos se lo controla mediante las validaciones de longitud de campos, cédula de identidad, dirección de correo electrónico, fechas, entre otras; con esto se pretende que el ingreso de información sea correcto.

  La integridad lógica se refiere a las relaciones que existen entre las reglas de negocio y la base de datos, las cuáles ayudan a la ejecución correcta de los procesos.

- **Eficiencia:** el sistema deberá trabajar de manera rápida, entregando información confiable al usuario. El tiempo de respuesta generado por el sistema dependerá del ancho de banda para el enlace a internet requerido para acceder al Sistema WEB con el que cuente el equipo desde el que se accederá a la aplicación. En las pruebas realizadas en la etapa de desarrollo, el tiempo de respuesta para el ingreso a una pantalla, o para manipular ingreso, cambio o consulta de información se estima de 1 a 2 segundos con un ancho de banda adecuado, esto se logra al tener toda la información integrada correctamente en la base, las consultas generadas a través de la aplicación se realizan de manera rápida optimizando el tiempo

de respuesta que visualiza el usuario, el desarrollo del sistema con programación orientada a objetos, el uso adecuado de la metodología que nos guía en todo el proceso de desarrollo.

- **Calidad:** La calidad en cualquier clase de producto y de la misma manera en un sistema informático es la apreciación que el cliente tiene de él. La calidad abarca diversas características entre las que se destacan: seguridad, integridad, eficiencia, disponibilidad entre otras.

- **Documentación:** De acuerdo a la metodología empleada en el desarrollo del presente proyecto, la documentación juega un papel importante para el fácil entendimiento del funcionamiento del sistema, puesto que permite el auto aprendizaje a los futuros operadores del sistema que no hayan estado presentes en la capacitación previa a la implantación. Al implantar el sistema, posterior a la capacitación, se entregará al delegado encargado por la ATKDP el manual de usuario. Este manual junto a toda la documentación que sea solicitada y sea de ayuda para un correcto manejo del sistema será entregado de manera documentada al personal de la ATKDP.

- **Disponibilidad:** el sistema será multiusuario, permitiendo el ingreso de información desde varios ordenadores al mismo tiempo; además el sistema funcionará en línea a cualquier hora del día. Esto permitirá que el ingreso de datos desde varios terminales sea factible agilitando procesos como carnetización, inscripción, pagos, entre otros. Además el hecho de estar

disponible 24 horas los 7días de la semana durante 365 días al año permite el fácil acceso al sistema sin ninguna restricción de horarios.

3.5- Condiciones para el Diseño

- Los colores que maneje la interfaz deberán representar los colores de la ATKDP.
- Ningún usuario tendrá acceso a los módulos que no necesite según su perfil.
- Todos los movimientos que representen flujo de efectivo deben generar automáticamente un documento comprobante de la transacción efectuada.
- Todas las pantallas donde se almacene información deben tener las opciones de insertar eliminar y actualizar.

3.6- Plan de entrega

Se debe establecer un plan de entrega que sirva al equipo de desarrollo para conocer el cumplimiento, avances y límites de cada etapa del proyecto. El cliente establece el orden en el que se desarrollará cada historia de usuario, dependiendo de los cambios o mejoras que surjan en cada presentación de avances, se realiza una nueva iteración.

Una historia de usuario se dará por finalizada cuando ya no existan más requerimientos por parte del usuario, cuando la interfaz haya sido probada y cuando se hayan realizado las pruebas respectivas con el cliente.

Se debe considerar en el plan de entrega para cada historia de usuario:

- Tiempo de desarrollo
- Número de iteraciones que puede tener cada historia de usuario

El tiempo de desarrollo se calcula tomando en cuenta la rapidez con la que en equipo de trabajo pueda desarrollar cada historia de usuario, de acuerdo a la metodología empleada, se estiman un máximo de 8 horas de trabajo diarias que indicarán un esfuerzo de 40 horas semanales.

La estimación de cada historia de usuario a desarrollar y la prioridad u orden se detalla en la tabla (Tabla 3.17) a continuación:

Tabla 3.17: Estimación de cada historia de usuario y acciones a desarrollar

Módulo	Historias de Usuario	Tiempo Estimado		
		Semanas	Días	Horas
Pasos Previos	Instalación de Herramientas y Creación de Ambiente de Desarrollo	1	5	40
Administración de Perfiles de Usuario	Ingreso y utilización del sistema por parte del usuario	1,6	8	64
Inscripción e Ingreso de Información Preliminar	Registro de clubes administrados por la Asociación de Tae Kwon Do de Pichincha (ATKDP).	0,4	2	16
	Registro de entrenadores correspondientes a los clubes administrados por la ATKDP	0,4	2	16
	Registro de categorías reglamentarias para competir en un torneo organizado por la ATKDP	0,4	2	16
	Registro de deportistas correspondientes a los clubes administrados por la ATKDP	0,4	2	16
	Registro de torneos, campeonatos o selectivos organizados por la ATKDP	0,4	2	16
Administración de Torneos	Inscripción de clubes a participar en un determinado torneo, campeonato o selectivo organizado por la ATKDP	0,4	2	16
	Elaboración automática de las llaves para un determinado torneo, campeonato o selectivo organizado por la ATKDP	1,4	7	56
	Registro de puntaje obtenido en cada uno de los combates para un determinado torneo, campeonato o selectivo organizado por la ATKDP	0,6	3	24
	Registro de ganadores en cada una de las categorías para un torneo, campeonato o selectivo organizado por la ATKDP	0,4	2	16
	Consulta de reportes para un torneo, campeonato o selectivo organizado por la ATKDP	0,8	4	32
Administración Financiera	Ingreso de Pago por concepto de carnetización o Inscripción de los deportistas en un torneo, campeonato o selectivo organizada por la ATKDP	0,6	3	24
	Ingreso de Pagos por concepto de inscripción anual de clubes en la ATKDP	0,6	3	24
	Ingreso de valores por conceptos varios a la ATKDP	0,6	3	24
	Egreso de valores por conceptos varios de la ATKDP	0,6	3	24
	Consulta de reportes financieros de la ATKDP	0,8	4	32
	TOTALES	**11,4**	**57**	**456**

3.7- Iteraciones por cada entrega

Como se puede ver en la Tabla 3.17, se estima que el alcance del proyecto se logrará en condiciones óptimas de trabajo en un total de 11.4 semanas si se tiene un buen desempeño; para este proyecto cada iteración comprenderá un conjunto de historias de usuario que pertenezcan a un mismo módulo (alcance), con ello, se tendrán 3 iteraciones con un aproximado de 4 semanas de planificación cada una, para la ejecución del proyecto.

Varias historias de usuario requieren mayor atención debido a la complejidad de su funcionalidad, por lo que necesitarán más tiempo para su desarrollo.

El orden de prioridades establecidas por la ATKDP, se enmarca en la siguiente estructura:

- **Primera iteración:** módulos de Administración de Perfiles de Usuario e Inscripción e Ingreso de Información Preliminar.
- **Segunda iteración:** módulo de Administración de Torneos
- **Tercera iteración:** módulo de Administración Financiera

La tabla a continuación (Tabla 3.18), muestra a detalle las iteraciones y tiempos (en semanas) del sistema:

Tabla 3.18: Estimación de Iteraciones

Módulo	Historias de Usuario	Semanas Estimadas	Iteraciones		
			1	2	3
Pasos Previos	Instalación de Herramientas y Creación de Ambiente de Desarrollo	1	X		
Administración de Perfiles de Usuario	Ingreso y utilización del sistema por parte del usuario	1,6	X		
Inscripción e Ingreso de Información Preliminar	Registro de clubes administrados por la Asociación de Tae Kwon Do de Pichincha (ATKDP).	0,4	X		
	Registro de entrenadores correspondientes a los clubes administrados por la ATKDP	0,4	X		
	Registro de categorías reglamentarias para competir en un torneo organizado por la ATKDP	0,4	X		
	Registro de deportistas correspondientes a los clubes administrados por la ATKDP	0,4	X		
	Registro de torneos, campeonatos o selectivos organizados por la ATKDP	0,4	X		
Administración de Torneos	Inscripción de clubes a participar en un determinado torneo, campeonato o selectivo organizado por la ATKDP	0,4		X	
	Elaboración automática de las llaves para un determinado torneo, campeonato o selectivo organizado por la ATKDP	1,4		X	
	Registro de puntaje obtenido en cada uno de los combates para un determinado torneo/ campeonato o selectivo organizado por la ATKDP	0,6		X	
	Registro de ganadores en cada una de las categorías para un torneo, campeonato o selectivo organizado por la ATKDP	0,4		X	
	Consulta de reportes para un torneo, campeonato o selectivo organizado por la ATKDP	0,8		X	
Administración Financiera	Ingreso de Pago por concepto de carnetización o Inscripción de los deportistas en un torneo, campeonato o selectivo organizada por la ATKDP	0,6			X
	Ingreso de Pagos por concepto de inscripción anual de clubes en la ATKDP	0,6			X
	Ingreso de valores por conceptos varios a la ATKDP	0,6			X
	Egreso de valores por conceptos varios de la ATKDP	0,6			X
	Consulta de reportes financieros de la ATKDP	0,8			X
	TOTALES	**11,4**	**4,6**	**3,6**	**3,2**

3.8- Cronograma de Planificación

El cronograma de planificación estimado para desarrollar cada historia de usuario es el siguiente:

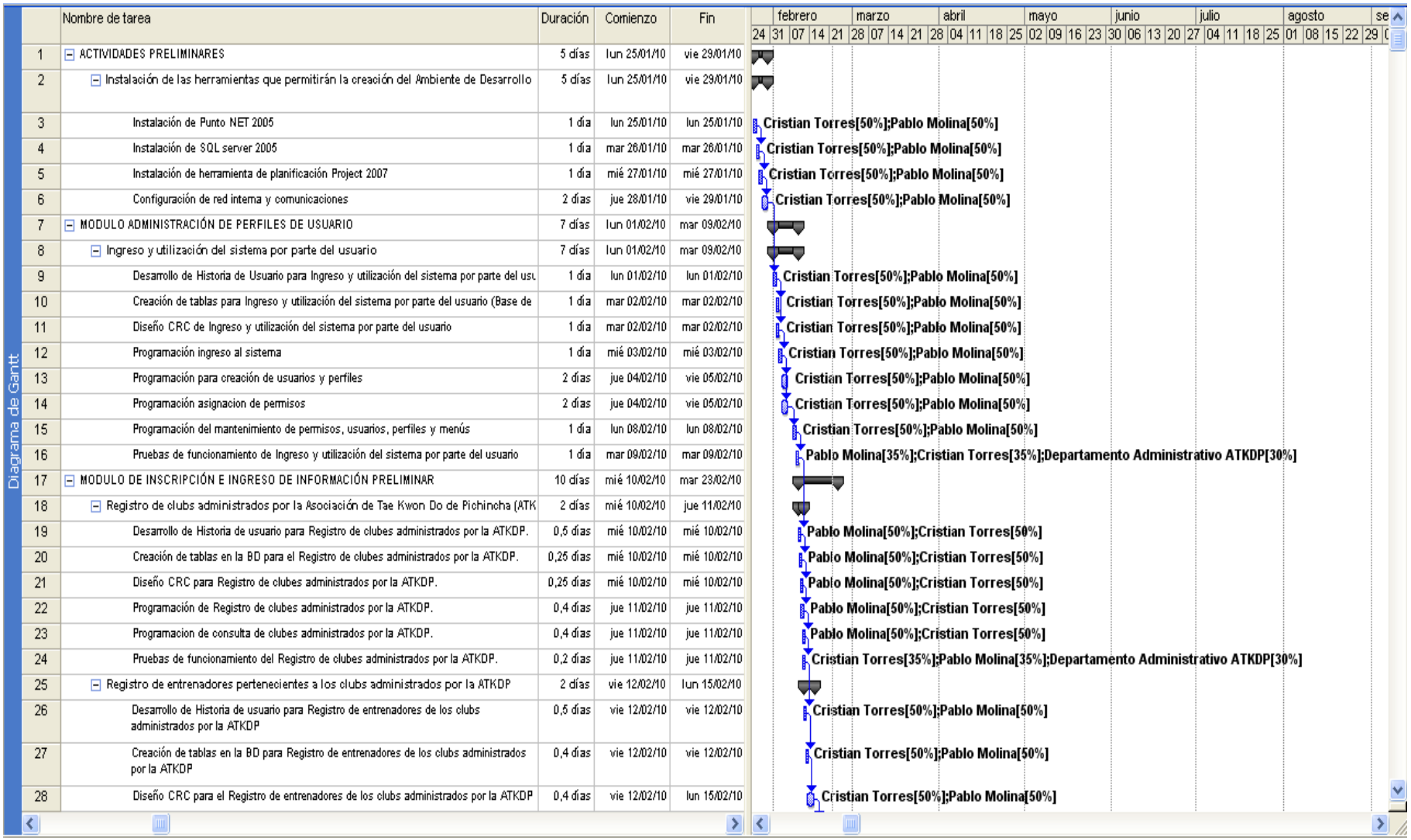


Figura 3.2: Plan de Entrega; (primera parte)

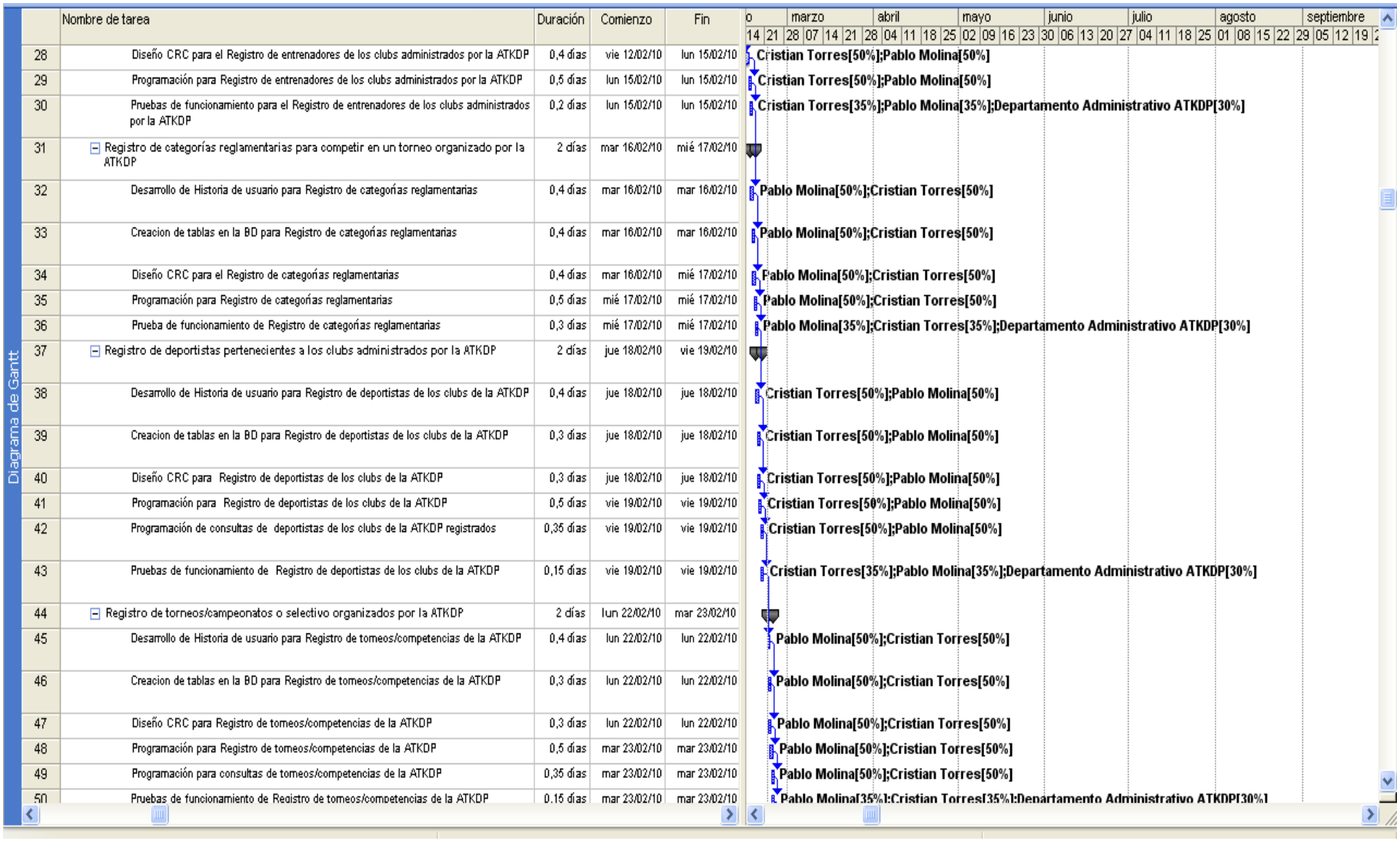


Figura 3.2: Plan de Entrega; (segunda parte)

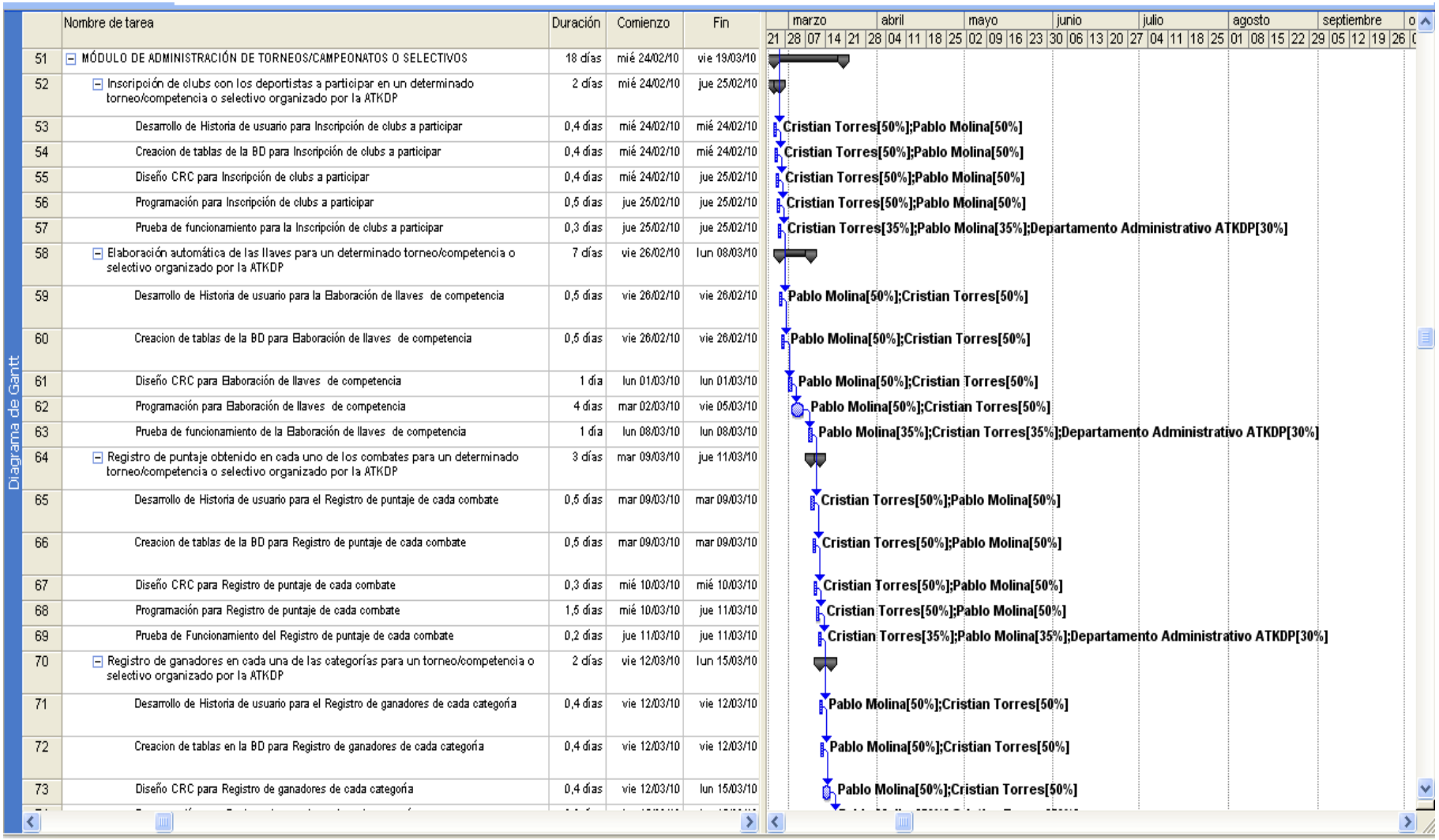


Figura 3.2: Plan de Entrega; (tercera parte)

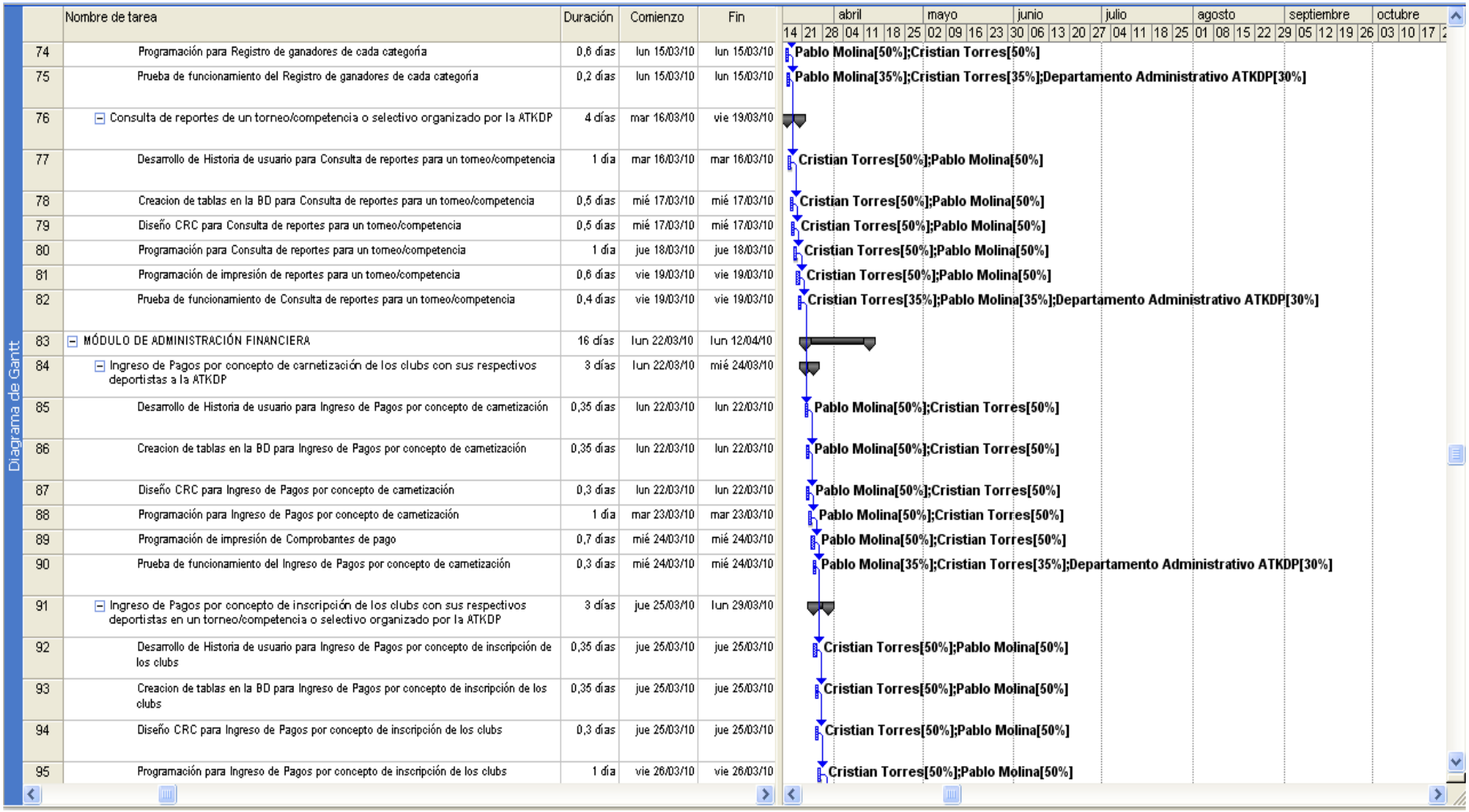


Figura 3.2: Plan de Entrega; (cuarta parte)

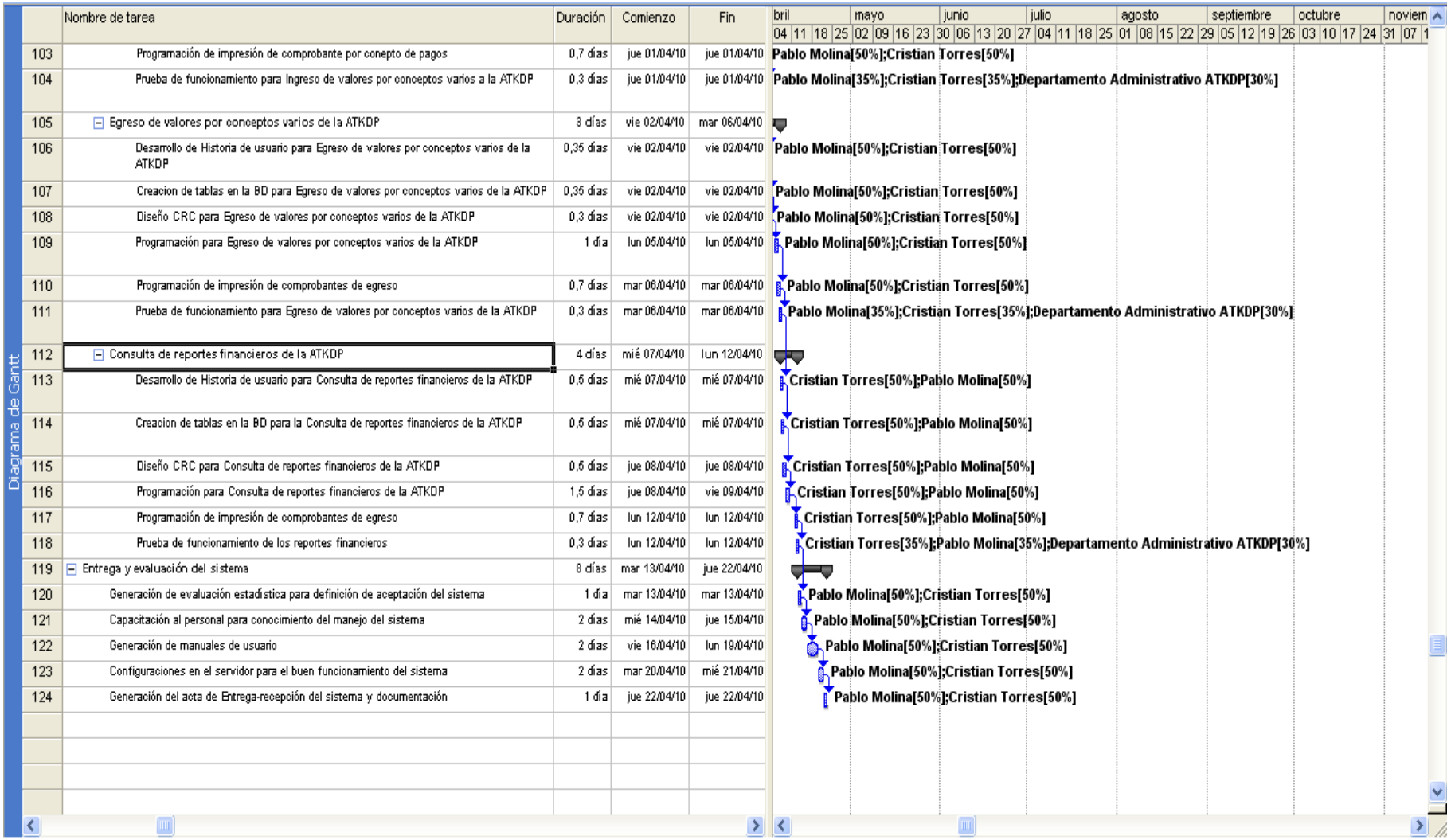


Figura 3.2: Plan de Entrega; (quinta parte)

CAPÍTULO IV

DISEÑO DEL PROYECTO

4.1- Arquitectura del Sistema

En base a un análisis detallado del Sistema de Administración para la ATKDP a desarrollar, es necesaria la aplicación de una arquitectura cliente servidor bajo el modelo de 3 capas; ya que, se ajusta de una manera adecuada a los requerimientos proporcionados por el usuario.

4.1.1- Modelo 3 Capas

Esta arquitectura consiste en un cliente que realiza peticiones a otro programa servidor que es el encargado de darle la respuesta.

La separación entre cliente y servidor es de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina, ni es obligatoriamente un sólo programa. Son varias máquinas cliente, las que acceden lógicamente a la capa de negocios donde se procesan todas las peticiones generadas desde la aplicación en tiempo real.

La programación por capas tiene el objetivo primordial de separar la lógica de negocios de la lógica de diseño. La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varias capas y, en caso de que se tenga que

realizar algún cambio, sólo se modifica la capa requerida sin tener que revisar entre código mezclado, también permite distribuir el trabajo de creación de una aplicación por niveles; de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles.

A continuación se da una breve explicación de las capas que se van a utilizar:

- **Capa de presentación**: es la interfaz de interacción con el usuario, misma que debe ser amigable y fácil de usar, es la que está en contacto con el usuario directamente en pantalla y le presenta la información del sistema; esta capa se comunica únicamente con la capa de negocio.

- **Capa de negocio:** En esta capa se encuentran los programas que se ejecutan al recibir las peticiones del usuario y envían las respuestas después de realizado el proceso. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados; y con la capa de datos.

- **Capa de datos:** Es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

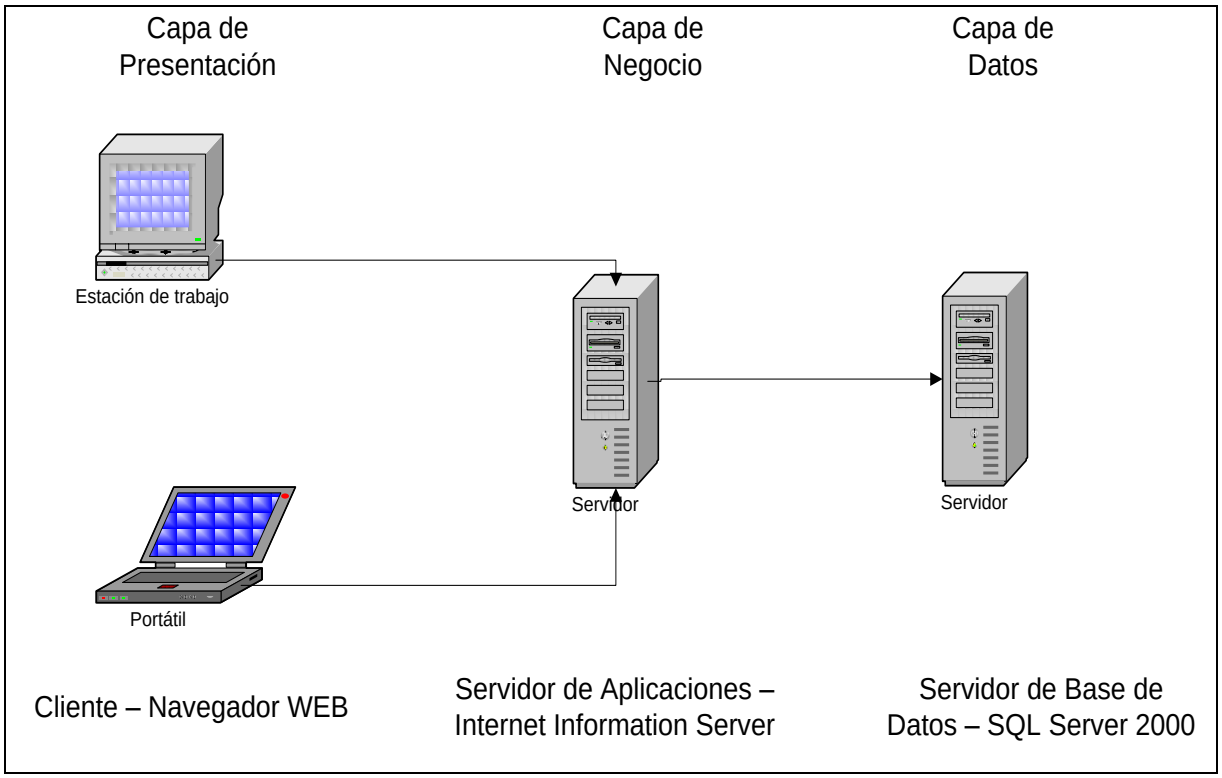


Figura 4.1: Arquitectura cliente servidor 3 capas

4.1.2- Modelo Vista Controlador(MVC)

Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos.

Se la emplea con frecuencia en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página, el controlador es el Sistema de Gestión de Base de Datos y el modelo es el modelo de datos.

4.2- Módulos del Sistema

4.2.1- Módulo de Inscripción e Ingreso de Información Preliminar:

En este módulo se realiza la inscripción de clubes, entrenadores, deportistas, categorías, pesos, torneos, campeonatos o selectivos con sus datos correspondientes; esto con el fin de mantener un registro histórico, utilizar estos datos para organizar un torneo[28], campeonato[29] o selectivo[30], y mantener un control con respecto a la administración de la ATKDP. En cada una de las pantallas, se ejecutan las opciones básicas en cuanto a administración de datos se refiere, esto es: eliminar, editar, guardar, y buscar a través de filtros de consulta por selección de registros.

4.2.2- Módulo de Administración de Torneos, Campeonatos o Selectivos:

En este módulo se realiza la inscripción de los clubes con sus respectivos deportistas en los diferentes torneos, campeonatos o selectivos organizados por la ATKDP, esta inscripción se la realiza en un rango de tiempo acordado por los directivos de la misma; además se generan las llaves para los combates y se registran los resultados de los encuentros así como los ganadores de los tres primeros lugares divididos por categorías, sexo y peso.

[28] Torneo es una competencia en la que pueden participar cualquier número de competidores por cada club.

[29] Campeonato es una competencia en la que sólo puede participar un único deportista por categoría representando a un club

[30] Selectivo es una competencia para seleccionar a los mejores deportistas por categoría para formar una selección que represente a una provincia o a un país.

4.2.3- Módulo de Administración de perfiles de Usuario:

Este módulo es el encargado de asignar permisos y autentificar la entrada a los usuarios que administren el sistema, de esta manera un usuario sólo tendrá acceso a los módulos que necesite utilizar de acuerdo a los permisos que le hayan sido asignados.

4.2.4- Módulo de Administración Financiera:

En este módulo se registran todos y cada uno de los ingresos por carnetización, inscripción e ingresos varios además de los egresos de la ATKDP, con ello se tiene en tiempo real la situación financiera de la asociación para un mejor control y manejo del flujo de efectivo.

El módulo financiero permite generar varios reportes consolidados de información que muestran los movimientos contables realizados, admitiendo la factibilidad de impresión de comprobantes enumerados automáticamente para control de dichos movimientos.

4.2.5- Reportes

La interfaz de usuario, al ser estandarizada, muestra en el menú principal la opción de reportería, misma que dependiendo del módulo en el que se esté trabajando, muestra los tipos de reportes habilitados para su visualización y en caso de requerirse su posterior impresión.

4.3- Estándares de programación

Los estándares de programación permiten normalizar la estructura de codificación de variables, tablas, vistas, y demás elementos vigentes en el desarrollo del software, mismos que ayudan a escribir código de calidad que pueda ser comprendido fácilmente por cualquier programador que en un futuro retome el desarrollo del sistema, para implementar nuevos requerimientos o simplemente necesite dar mantenimiento a los módulos ya generados.

Los controles que se van a usar tienen como prefijo el nombre por defecto que le da Visual Studio.Net 2005 seguido del nombre que se le asigne, que representa su significado en la pantalla, se citan algunos ejemplos en la tabla a continuación (Tabla 4.1):

Tabla 4.1: Estandarización de elementos de desarrollo

DESCRIPCION	SIMBOLOGÍA	EJEMPLO
Caja de Texto **Ingreso de información**	TextBox	TextBoxNombre
Vista en Grid **Despliegue de información**	GridView	GridViewDeportista
Label **Información – mensaje**	Label	LabelMensaje
Listado **Lista de varios elementos**	DropDownList	DropDownListEstado
Botón **Selección de funciones o eventos**	Button	ButtonGuardar
Caja de Selección **Confirmación alternativas**	CheckBox	CheckBoxPermiso
Paneles **Agrupación de elementos**	Panel	PanelTorneo

La codificación del sistema también conlleva una estandarización propia para definición de clases y funciones; mismas que se detallan en la siguiente tabla:

Tabla 4.2: Estandarización de código fuente

DESCRIPCION	SIMBOLOGÍA	EJEMPLO
Clase **Agrupación de funciones que abstraen un objeto específico.**	Clase_	Clase_Usuario
Función **Especifica la tarea a la que hacen referencia. Está formada por el nombre de la actividad con la primera letra en mayúscula y separa las palabras con un guión bajo.**	No aplica	Guardar_Detalle Validar_Cedula Eliminar_Ingreso_Detalle

4.4- Interfaz

De acuerdo con los módulos señalados anteriormente, el Sistema posee varias pantallas, tales como: creación de clubes, creación de deportistas, asignación de un deportista a un club, inscripción en competencias, registro de ingresos por concepto de inscripción en competencias, entre otras.

Todas las pantallas para el Sistema de la ATKDP están diseñadas de la siguiente manera:

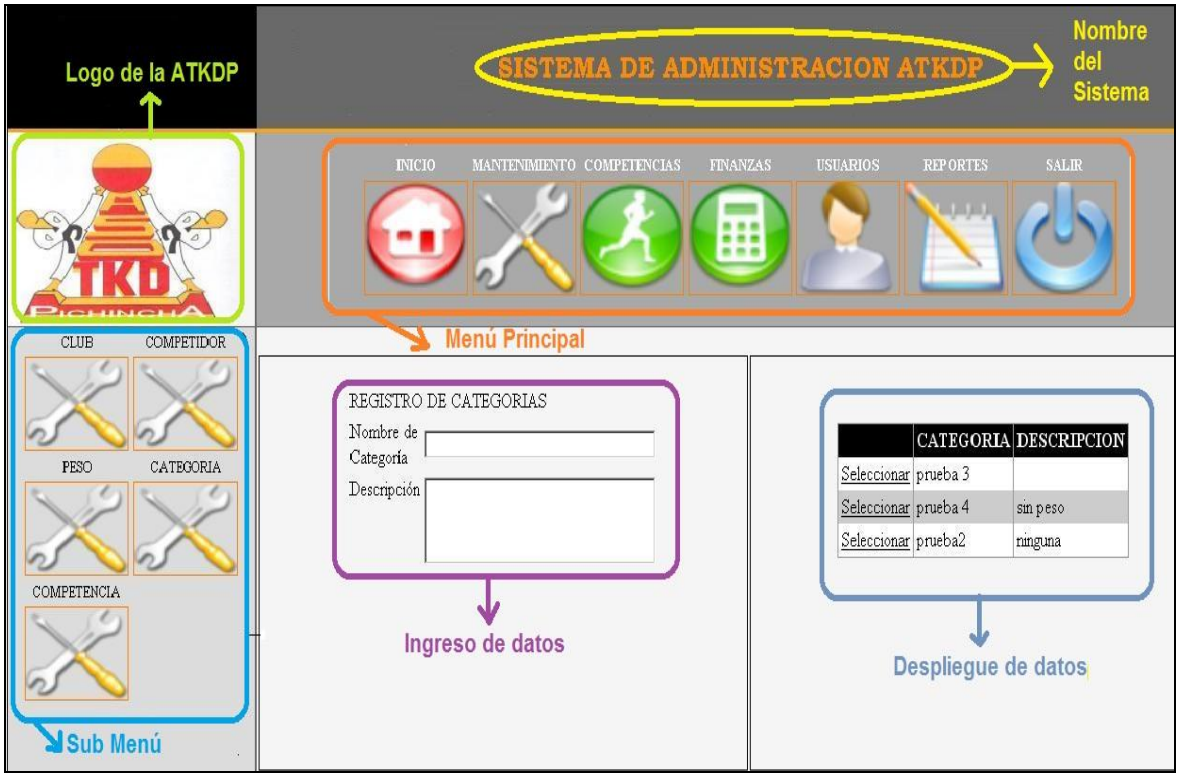


Figura 4.2: Interfaz de usuario - Estandarización

De acuerdo a la estandarización mostrada en la figura 4.2, todas las pantallas tienen:

- El nombre del Sistema en la parte Superior
- El menú principal ubicado bajo el nombre del sistema
- El Logo de la ATKDP ubicado en la parte superior izquierda
- El sub menú ubicado bajo el logo de la ATKDP
- Dos secciones centrales, uno de lado izquierdo para el ingreso de datos y otro de lado derecho para el despliegue de datos

Cada pantalla tendrá en la parte inferior de la sección para ingreso de datos, varios botones que se utilizan para manipular información; estos botones son: nuevo, buscar, guardar, eliminar y limpiar.

Cada uno de los botones posee un gráfico que representa la acción que se quiere realizar, así como el nombre; como se muestra en la figura a continuación:

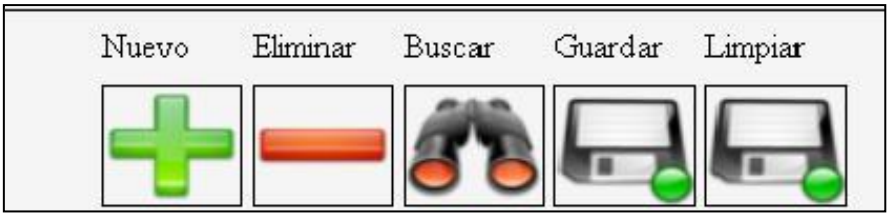


Figura 4.3: Botones básicos

En todas las pantallas para la opción de búsqueda se utiliza el filtro del dato identificador del registro a consultar, por ejemplo: si se busca a un determinado deportista, se lo consulta por la cédula o parte de ella; con el objetivo de facilitar el proceso de búsqueda por parte del usuario final.

En los casos de aquellas pantallas que no manejen un identificador numérico se podrá filtrar la búsqueda por el nombre representativo del registro buscado.

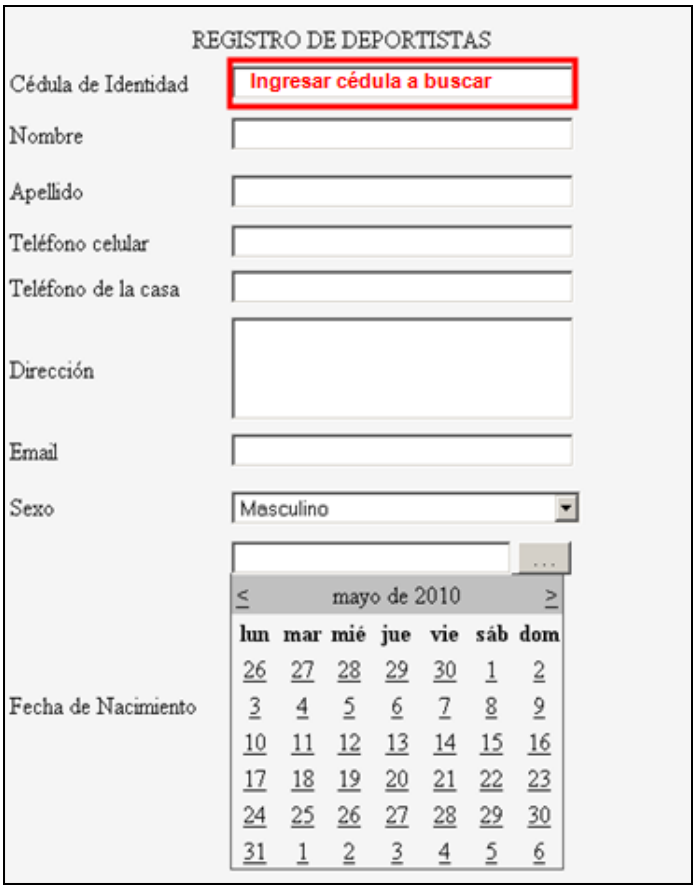


Figura 4.4: Búsquedas

Como se muestra en la figura 4.4, para ingresar la fecha de nacimiento se maneja un calendario, de esta manera funcionan todas las pantallas donde existan ingresos de fechas ya sea de nacimiento, de inscripción, entre otras; cabe mencionar que el ingreso de varias fechas es con el fin tanto informativo como de llevar un control sobre los procesos en ejecución.

En las pantallas donde se maneje ingresos o egresos de dinero, tiene un número de comprobante alfanumérico representativo para cada tipo de transacción, por ejemplo:

- RC1 = Registro de club
- RD1 = Registro de deportista
- IC1 = Inscripción en competencia
- IV1 = Ingreso vario
- EV1 = Egreso vario

PAGOS REALIZADOS

	CODIGO	FECHA	REMITENTE	COMPROBANTE	TIPO
Seleccionar	59	29/04/2010 0:00:00	ds	RC1	Registro Club
Seleccionar	60	29/04/2010 0:00:00	ded	RC2	Registro Club
Seleccionar	61	29/04/2010 0:00:00	ded	RC3	Registro Club
Seleccionar	62	22/04/2010 0:00:00	fe	RC4	Registro Club
Seleccionar	63	03/05/2010 0:00:00	JHJ	RC5	Registro Club
Seleccionar	64	08/05/2010 0:00:00	ds	RD1	Carnetizacion
Seleccionar	65	08/05/2010 0:00:00	ds	IC1	Inscripcion
Seleccionar	66	08/05/2010 0:00:00	sa	IC2	Inscripcion

Figura 4.5: Visualización de datos autogenerados

4.5- Modelado de datos

Los modelos de datos se usan para escribir una base de datos, éstos permiten describir la realidad que está reflejada en las historias de usuario que se

realizaron en la etapa de planificación del proyecto; el modelo que se va a utilizar es el de Entidad – Relación.

El modelo tiene algunos estándares, los cuáles son necesarios para que la base de datos sea de fácil entendimiento incluso para personas que no poseen conocimientos informáticos profundos; además los estándares permiten que la base de datos sea mantenida y actualizada con mayor facilidad.

Los estándares propuestos para el presente desarrollo de la base de datos están descritos a continuación:

- Los nombres de las tablas en la base de datos deben estar escritos en mayúsculas, si tienen dos palabras deben estar separadas por un guión bajo, por ejemplo:
  - DEPORTISTA
  - INSCRIPCION_COMPETENCIA
  - CATEGORIA
  - COMPETENCIA
  - PESO

Los nombres reflejan el significado de la tabla, sólo se usan caracteres alfabéticos; no se utilizan dígitos numéricos, caracteres de puntuación o símbolos.

- Los nombres de las tablas no se podrán abreviar en ningún caso.

- Toda tabla debe tener clave primaria

- Toda relación entre tablas se establece mediante claves foráneas con integridad referencial, con esto no se podrán borrar datos que estén relacionados.

- Los campos que pertenezcan a claves primarias deben ubicarse al principio de la tabla, cabe mencionar que estos campos son auto numéricos y todos están nombrados con la palabra ID, seguido de un guión bajo y el nombre de la tabla a la cuál correspondan, por ejemplo:

Tabla 4.3: Identificadores o llaves primarias en la Base de Datos.

TABLA	CLAVE PRIMARIA
DEPORTISTA	ID_Deportista
COMPETENCIA	ID_Competencia

- Las vistas en la base de datos deben estar escritos en mayúsculas, deben tener la palabra VISTA seguido de un guión bajo y del nombre, los nombres reflejan el significado de la vista, sólo se usan caracteres alfabéticos; no se utilizan dígitos numéricos, caracteres de puntuación o símbolos, se cita algunos ejemplos:
  - VISTA_ESTUDIANTE
  - VISTA_CATEGORIA
  - VISTA_CLUB

4.6- Diseño de la Base de datos

Para la construcción de la base de datos se emplea el modelo Entidad – Relación, lo que se obtiene es un modelo completo de la base de datos correctamente normalizada y relacionada constituida por tablas, campos, tipos de datos.

Cabe mencionar que esta base se construye tomando en cuenta los requerimientos plasmados en las historias de usuario ya que, el fin es obtener la visión que tienen los usuario sobre los datos y su manipulación a través del sistema informático en cuestión.

A continuación se presenta el modelo de la base de datos del Sistema para la ATKDP tomado directamente del gestor de base de datos, en este caso de SQL Server 2000.

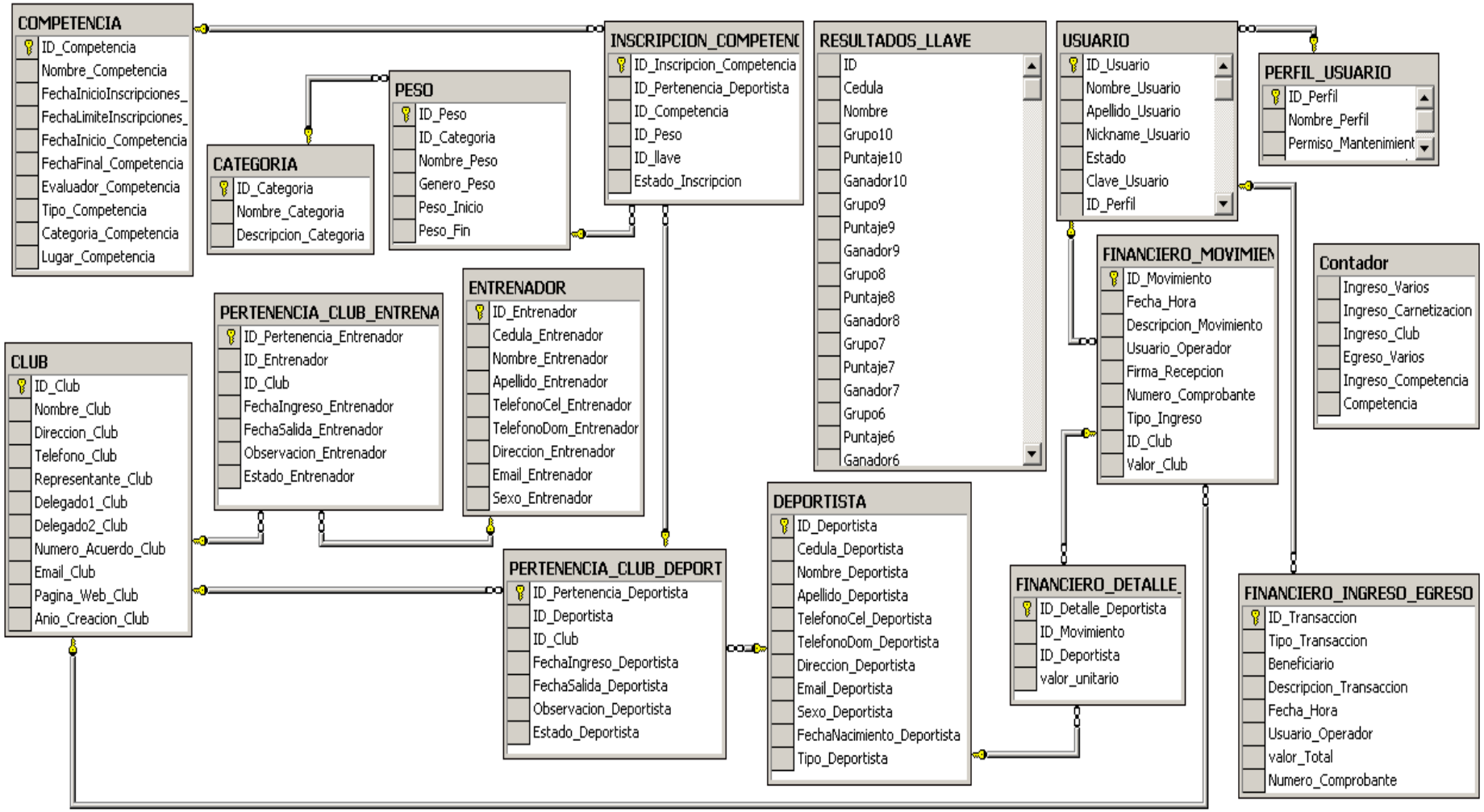


Figura 4.6: Base de Datos del Sistema

4.7- Tarjetas CRC

Las tarjetas CRC (Clase, Responsabilidad, Colaboración) se usan en el Diseño de Software, se utilizan en la metodología XP; éstas permiten describir el comportamiento de las clases y su funcionamiento dentro de un Sistema Informático; las tarjetas CRC se enfocan en todas las ideas planteadas sin dejar de lado a ninguna.

Las tarjetas que se van a presentar están elaboradas con el siguiente formato:

Tabla 4.4: Formato de una tarjeta CRC

Clase: Nombre de la Clase	
Responsabilidad: Responsabilidades que debe cumplir	**Colaboración:** Clases colaboradoras

Cabe mencionar que las tarjetas CRC se construyen tomando en cuenta las historias de usuario. A continuación se presentan las Tarjetas CRC del sistema para la ATKDP:

Tabla 4.5: Tarjeta CRC – Conexión con la base de datos

Conexión con la base de datos	
Responsabilidad	**Colaboración**
• Establecer comunicación entre el sistema y la base de datos	

Tabla 4.6: Tarjeta CRC – Validaciones utilizadas en el Sistema

Validaciones utilizadas en el Sistema	
Responsabilidad	**Colaboración**
• Validar el ingreso de datos no duplicados como nombres de deportistas, nombres de competencias, entre otras. • Validar el ingreso de campos en blanco en datos que son esenciales en el sistema. • Establecer un valor alfa numérico automático de documento contable.	• Clase_Conexion • Clase_Club • Clase_Entrenador • Clase_Deportista • Clase_Usuario • Clase_Ingresos • Clase_Ingresos_Egresos

Tabla 4.7: Tarjeta CRC –Gestión de Datos Preliminares

Gestión de Datos Preliminares	
Responsabilidad	**Colaboración**
• Gestionar datos principales de deportistas • Gestionar datos principales de clubes • Gestionar datos principales de entrenadores • Gestionar datos principales de categorías • Gestionar datos principales de pesos • Gestionar datos principales de competencias Estos datos son indicados en las historias de usuario, son necesarios para el funcionamiento del sistema	• Clase_Conexion

Tabla 4.8: Tarjeta CRC – Asignaciones

Asignaciones	
Responsabilidad	**Colaboración**
• Realizar las asignaciones de entrenadores a un club específico, misma que puede durar un lapso de tiempo determinado; de esta manera un entrenador puede pertenecer a uno o varios clubes • Realizar las asignaciones de deportistas a un club específico, misma que puede durar un lapso de tiempo determinado; de esta manera un deportista puede pertenecer a uno o varios clubes	• Clase_Conexion • Clase_Club • Clase_Entrenador • Clase_Deportista

Tabla 4.9: Tarjeta CRC – Gestión de Competencias

Gestión de Competencias	
Responsabilidad	**Colaboración**
• Realizar la inscripción de un club con sus respectivos deportistas a una competencia determinada, dicha competencia puede ser un campeonato, torneo o selectivo • Construir las llaves de enfrentamientos de los deportistas participantes ordenados por competencia, categoría y peso • Ingresar los puntajes obtenidos por los deportistas en los diferentes enfrentamientos a los que fueron asignados	• Clase_Conexion • Clase_Competencia

Tabla 4.10: Tarjeta CRC – Gestión de Ingresos / Egresos

Gestión de Ingresos/Egresos	
Responsabilidad	**Colaboración**
• Realizar el ingreso por concepto de inscripción de clubes • Realizar el ingreso por concepto de carnetización de deportistas • Realizar el ingreso por concepto de inscripción en una competencia determinada por parte de un club con sus respectivos deportistas • Realizar el ingreso por conceptos varios como puede ser donación, multas, entre otras • Realizar el egreso por conceptos varios como puede ser gasto para refrigerios, transporte, insumos de papelería, entre otras	• Clase_Conexion • Clase_Ingresos • Clase_Ingresos_Egresos

Tabla 4.11: Tarjeta CRC – Gestión de Usuarios

Gestión de Usuarios	
Responsabilidad	**Colaboración**
• Registrar a los usuarios que van a utilizar el sistema de administración para la ATKDP con sus respectivos datos personales los mismos que constan en las historias de usuario • Asignar permisos a los usuarios de acuerdo a su perfil en la ATKDP, de esta manera los usuarios van a utilizar sólo las pantallas que necesiten para su desempeño óptimo dentro del sistema	• Clase_Conexion • Clase_Usuario

CAPÍTULO V

DESARROLLO DEL SISTEMA

5.1- Construcción de la interfaz del sistema de Administración de la Asociación de TAEKWONDO de Pichincha.

Para el desarrollo de la interfaz de usuario del sistema, se utilizó la plantilla que se puede visualizar en la figura 5.1, que es la base de todas las pantallas o interfaces; en esta plantilla se encuentra el menú principal de la aplicación, además se mediante el uso del patrón de diseño fachada se organiza la estructura que se mostrará en pantalla de forma estandarizada, para presentar el mismo esquema en cuanto a la interfaz de cada página.

El sistema permite también la aplicación del Modelo Vista Controlador (MVC), que permite darle una mejor estructura al sistema.

Figura 5.1: Plantilla de Menú General

5.1.1- Patrón fachada

El patrón Fachada simplifica las interfaces de los distintos módulos; ya que, en cuanto a presentación, todas las páginas son similares mientras que en funcionalidad cada una tiene su propia particularidad.

5.1.2- Modelo vista controlador (MVC)

Para el desarrollo de la aplicación se ha utilizado el patrón de diseño MVC o Modelo Vista Controlador, que permite separar de manera estructurada la parte gráfica, la lógica del negocio y la conexión a la base de datos.

Normalmente existen mayor cantidad de cambios en la interfaz grafica de las aplicaciones, por este motivo, el patrón en mención facilita el mantenimiento de la aplicación en cuanto a cambios, ya que no es necesario cambiar la lógica del negocio sino solamente lo que se necesita disminuyendo de esta forma la cantidad de programación y por ende el tiempo en el desarrollo.

5.2- Generación de código del sistema de Administración de la Asociación de TAEKWONDO de Pichincha.

La codificación se muestra de manera resumida detallando el estilo empleado, mostrando clases importantes, variables, entre otras.

5.2.1- Vista

Para la parte grafica de la aplicación se utilizan las siguientes librerías:

- using System;
- using System.Collections;
- using System.ComponentModel;
- using System.Data;
- using System.Drawing;
- using System.Web;
- using System.Web.SessionState;
- using System.Web.UI;
- using System.Web.UI.WebControls;
- using System.Web.UI.HtmlControls;
- using System.Data.SqlClient;
- using System.Configuration;
- using System.IO;
- using ATKDP_Negocio;

La librería ATKDP_Negocio es aquella que contiene toda la lógica del negocio, y es ahí donde se encuentran todas las funciones de la aplicación.

Para la declaración de las clases como formato se registra el nombre representativo de la clase tomando en cuenta que la primera letra de la o de las palabras, debe ir en mayúscula y separadas por el símbolo "_" por ejemplo:

public class **Clase_Deportista**

En el desarrollo de la codificación se usan varios comentarios, que, en caso de ser cortos y tan solo se requiera una línea para su descripción, tendrán en el inicio los símbolos "***//***"; en caso de contar con un comentario que abarque varias líneas se lo representara entre los símbolos "***/* */***", estos comentarios informaran acerca de todo aquello que explique el uso de alguna/as líneas de código.

A lo largo de la codificación, se tendrán varias variables de sesión las mismas que serán escritas de la siguiente manera:

Session["Nombre_Variable"] = "Dato_Asignado";

Todas las funciones serán nombradas de tal forma que sean entendibles a la vista y estarán escritas con una letra mayúscula en cada inicio de palabra así como del símbolo "_" que separa cada palabra. Las funciones a utilizarse en la parte de la vista tendrán la siguiente representación:

Protected void **Limpiar_Cajas_Texto ()**

5.2.2- Lógica de negocio:

Las librerías que se usan en la lógica de negocio son las siguientes:

- using System;
- using System.Collections.Generic;

- using System.Text;
- using System.Data;
- using System.Data.SqlClient;
- using ATKDP_Conexion;

Al igual que en la Vista se usa una librería propia que administra la conexión a la base de datos y el acceso a la misma.

Cada clase tiene funciones muy similares en cuanto a inserción, actualización, búsqueda y eliminación de información.

Así por ejemplo, a continuación se detalla una de las funciones mas recurridas en los procesos de administración del sistema, la misma que permite establecer la conexión con la base de datos.

```
public SqlConnection Abrir_Conexion()
{
   objConexion=new SqlConnection("Data Source=SISTEMAS;Initial catalog=ATKDP;Integrated
            Security=True; User=xxxxx; Password=xxxxxxx");
    if (objConexion.State == ConnectionState.Closed)
        objConexion.Open();
    return objConexion;
}
```

5.3- Desarrollo del módulo de Mantenimiento

5.3.1- Creación de Clubes

La creación de clubes se realiza para ingresar un nuevo Club al sistema o a su vez para modificar ciertos datos de alguno ya creado con anterioridad.

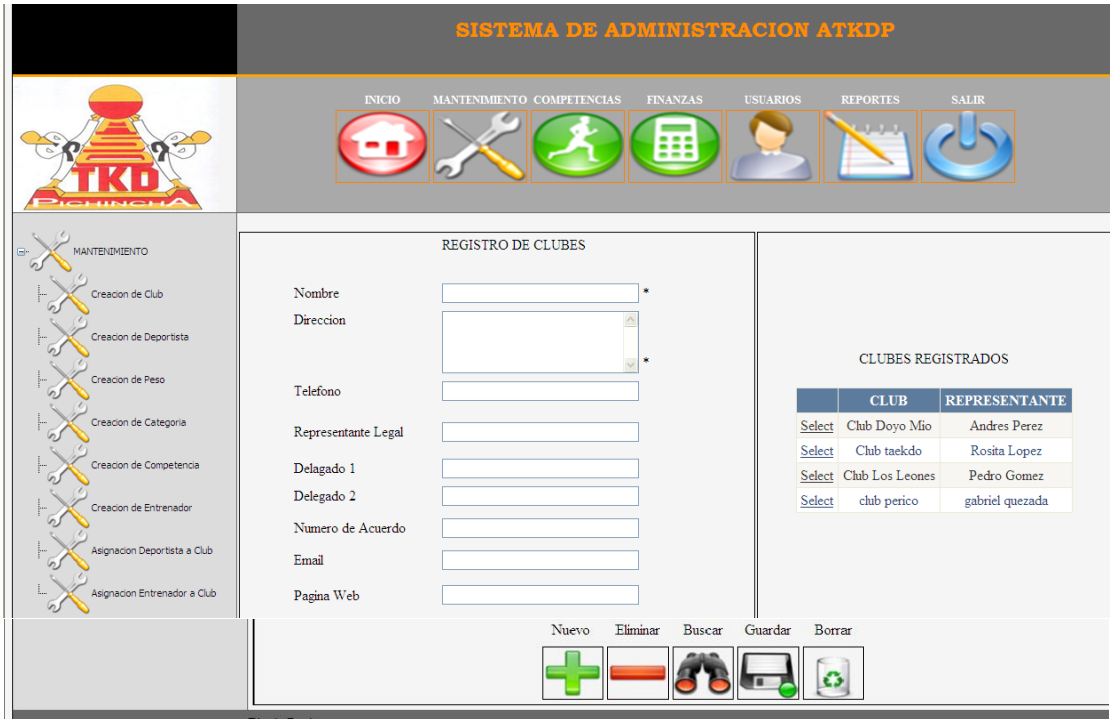


Figura 5.2: Pantalla creación de clubes

Para la creación de un club se deben ingresar los datos en las cajas de texto que se muestran y se podra visualizar en varias pantalla en ciertos campos obligatorios el símbolo “*”.

La funcion que realiza la inserción de la información a la Base de Datos es la siguiente:

```
protected void ImageButtonGuardar_Click(object sender, ImageClickEventArgs e)
   {
      try
      {
         Boolean registros;
         if (LabelResultado_Operacion.Text.Equals("true"))
            insertar = true;
         else
            insertar = false;
// ---- Se insertan o se actualizan los datos del club.
         if (insertar)
            registros = objClase_Club.Registrar_Club(TextBoxNombre.Text,
TextBoxDireccion.Text, TextBoxTelefono.Text, TextBoxRepresentante.Text,
TextBoxDelegado1.Text, TextBoxDelegado2.Text, TextBoxAcuerdo.Text, TextBoxEmail.Text,
TextBoxPagina.Text, TextBoxCreación.Text);
         else
```

```
            registros =
objClase_Club.Actualizar_Club(LabelResultado_Operacion.Text,TextBoxNombre.Text,
TextBoxDireccion.Text, TextBoxTelefono.Text, TextBoxRepresentante.Text,
TextBoxDelegado1.Text, TextBoxDelegado2.Text, TextBoxAcuerdo.Text, TextBoxEmail.Text,
TextBoxPagina.Text, TextBoxCreación.Text);
            if (registros && insertar)
            {
                LabelResultado_Operacion.Text = "Club registrado exitosamente.";
                LabelResultado_Operacion.Visible = true;
            }
            else if (registros && !insertar)
            {
                LabelResultado_Operacion.Text = "Club actualizado exitosamente.";
                LabelResultado_Operacion.Visible = true;
            }
            else
            {
                LabelResultado_Operacion.Text = "Error de Inserción. Vuelva a intentar.";
                LabelResultado_Operacion.Visible = true;
            }
            Limpiar_Cajas_Texto();
            DataSet ds = new DataSet();
            ds=objClase_Club.Recuperar_Club("completa", "", "");
            GridViewClub.DataSource = ds.Tables[0];
            GridViewClub.DataBind();
        }
        catch (Exception error)
        {
            LabelResultado_Operacion.Text = error.Message.ToString();
            LabelResultado_Operacion.Visible = true;
        }
    }
```

El botón guardar tiene doble funcionalidad; guardar información nueva o actualizar información existente. La opción a escogerse va de acuerdo a una bandera que se coloca con valor de “True” si se presiona el botón Nuevo o con valor “False” si se seleccionan datos del Datagrid (grilla o tabla en la que se despliegan datos almacenados en la base) que muestra el listado de clubes.

La funcion de Registrar_Club es la siguiente:

```
public Boolean Registrar_Club(string nombre,String direccion,String telefono, String representante,
String delegado1,String delegado2, String acuerdo, String mail, String paginaweb, String
aniocreación)
        {
            int registros;
            objConexion = objClase_Conexion.Abrir_Conexion();
            sentencia = String.Format("INSERT INTO CLUB(Nombre_Club, Direccion_Club,
Telefono_Club, Representante_Club, Delegado1_Club, Delegado2_Club, Numero_Acuerdo_Club,
Email_Club, Pagina_Web_Club, Anio_Creación_Club) VALUES('{0}', '{1}', '{2}', '{3}', '{4}', '{5}', '{6}',
```

```
'{7}', '{8}', '{9}')", nombre,
direccion,telefono,representante,delegado1,delegado2,acuerdo,mail,paginaweb,aniocreación);

        objComando = new SqlCommand(sentencia, objConexion);
        registros = Convert.ToInt32(objComando.ExecuteNonQuery());

        if (registros > 0)
            return true;
        else
            return false;
    }
```

Esta función recibe como parámetros todos y cada uno de los campos que despliega la pantalla, y es aquí donde se arma la sentencia SQL que se ejecutará para la inserción. Una vez que la sentencia se ejecuta la función devolverá como resultado el valor “True” si el número de registros afectados es mayor a 0, en caso contrario se devolverá el valor de “False” como indicador de que no se realizó la inserción.

5.3.2- Creación de Deportistas

La creación de deportistas se realiza para ingresar un nuevo deportista al sistema o a su vez para modificar ciertos datos de alguno ya creado con anterioridad.

Figura 5.3: Pantalla creación de deportistas

Para la creación de un deportista se debe ingresar los datos en las cajas de texto que se muestran y se podra visualizar en pantalla en ciertos campos obligatorios el símbolo “*”.

La función que realiza la inserción de la información a la Base de Datos es la siguiente:

```
protected void ImageButtonGuardar_Click(object sender, ImageClickEventArgs e)
   {
      try
      {
         Boolean validacion_duplicadoDeportista = false;
         validacion_duplicadoDeportista =
objClase_Deportista.Autentificar_Duplicados_Deportista(TextBoxCedula.Text);
         if (TextBoxCedula.Text != "" & TextBoxNombre.Text != "" & TextBoxApellido.Text != "" &
TextBoxDireccion.Text != "" & TextBoxFecha.Text != "")
         {
            Boolean registros;
            if (Label_Insertar.Text == "verdadero" & validacion_duplicadoDeportista == false)
               registros = objClase_Deportista.Registrar_Deportista(TextBoxCedula.Text,
TextBoxNombre.Text, TextBoxApellido.Text, TextBoxCelular.Text, TextBoxTelefono.Text,
```

```
TextBoxDireccion.Text, TextBoxEmail.Text, DropDownListSexo.Text,
Convert.ToDateTime(TextBoxFecha.Text), DropDownListTipo.Text);
                else
                    registros =
objClase_Deportista.Actualizar_Deportista(LabelResultado_Operacion.Text, TextBoxCedula.Text,
TextBoxNombre.Text, TextBoxApellido.Text, TextBoxCelular.Text, TextBoxTelefono.Text,
TextBoxDireccion.Text, TextBoxEmail.Text, DropDownListSexo.Text,
Convert.ToDateTime(TextBoxFecha.Text), DropDownListTipo.Text);
                if (registros && Label_Insertar.Text == "verdadero")
                {
                    LabelResultado_Operacion.Text = "Deportista registrado exitosamente.";
                    LabelResultado_Operacion.Visible = true;
                }
                else if (registros && !(Label_Insertar.Text == "verdadero"))
                {
                    LabelResultado_Operacion.Text = "Deportista actualizado exitosamente.";
                    LabelResultado_Operacion.Visible = true;
                }
                else
                {
                    LabelResultado_Operacion.Text = "Error de Inserción. Vuelva a intentar.";
                    LabelResultado_Operacion.Visible = true;
                }
                Limpiar_Cajas_Texto();
                DataSet ds = new DataSet();
                ds = objClase_Deportista.Recuperar_Deportista("completa", "", "");
                GridViewDeportista.DataSource = ds.Tables[0];
                GridViewDeportista.DataBind();
            }
            else
            {
                LabelResultado_Operacion.Text = "Error de Inserción. No se permiten deportistas
iguales ni en blanco.";
            }
        }
        catch (Exception error)
        {
            LabelResultado_Operacion.Text = error.Message.ToString();
            LabelResultado_Operacion.Visible = true;
        }
    }
```

En la pantalla, el botón guardar tiene doble funcionalidad; guardar información nueva o actualizar información existente. La opcion a escogerse va de acuerdo a una bandera que se coloca con valor de “True” si se presiona el botón Nuevo o con valor “False” si se seleccionan datos del Datagrid que muestra el listado de los deportistas registrados al momento.

La funcion de Registrar_Deportista es la siguiente:

```
public Boolean Registrar_Deportista(String cedula, String nombre, String apellido, String celular,
String telefono, String direccion, String email, String sexo, DateTime fecha, String tipo)
      {
         int registros;
         objConexion = objClase_Conexion.Abrir_Conexion();
         sentencia = String.Format("INSERT INTO DEPORTISTA(Cedula_Deportista,
Nombre_Deportista, Apellido_Deportista, TelefonoCel_Deportista, TelefonoDom_Deportista,
Direccion_Deportista, Email_Deportista, Sexo_Deportista, FechaNacimiento_Deportista,
Tipo_Deportista) VALUES('{0}', '{1}', '{2}', '{3}', '{4}', '{5}', '{6}', '{7}','{8}','{9}')", cedula, nombre,
apellido, celular, telefono, direccion, email, sexo, fecha.ToString("dd/MM/yyyy hh:mm:ss") , tipo);
         objComando = new SqlCommand(sentencia, objConexion);
         registros = Convert.ToInt32(objComando.ExecuteNonQuery());
         if (registros > 0)
            return true;
         else
            return false;
      }
```

Esta función recibe como parámetros todos y cada uno de los campos que despliega la pantalla, y es aquí donde se arma la sentencia SQL que se ejecutará para la inserción.

Una vez que la sentencia se ejecuta, la función devolverá como resultado el valor “True” si el número de registros afectados es mayor a 0, en caso contrario se devolverá el valor de “False” como indicador de que no se realizó la inserción.

5.3.3- Creación de Peso

La creación de pesos se realizará para ingresar un nuevo peso al sistema o a su vez para modificar ciertos datos de alguno ya creado con anterioridad.

Figura 5.4: Pantalla creación de peso

Para la creación de un peso ingresan los datos en las cajas de texto que se muestran en pantalla, se visualizará en pantalla, ciertos campos obligatorios, mismos que se diferencian por tener junto a la caja de texto el simbolo “*”.

La función que realiza la inserción de la información a la Base de Datos es la siguiente:

```
protected void ImageButtonGuardar_Click(object sender, ImageClickEventArgs e)
   {
      try
      {
         Boolean validacion_duplicadoPeso = false;
         validacion_duplicadoPeso =
objClase_Peso.Autentificar_Duplicados_Peso(DropDownListCategoria.Text,TextBoxNombre.Text);
            Boolean registros;
            if (Label_Insertar.Text == "verdadero" && validacion_duplicadoPeso == false)
               registros =
objClase_Peso.Registrar_Peso(DropDownListCategoria.Text,TextBoxNombre.Text,DropDownList
Sexo.Text, TextBoxInicio.Text, TextBoxFin.Text);
            else
```

```
                registros = objClase_Peso.Actualizar_Peso(LabelResultado_Operacion.Text,
DropDownListCategoria.Text, TextBoxNombre.Text, DropDownListSexo.Text, TextBoxInicio.Text,
TextBoxFin.Text);
            if (registros && Label_Insertar.Text == "verdadero")
            {
                LabelResultado_Operacion.Text = "Peso registrado exitosamente.";
                LabelResultado_Operacion.Visible = true;
            }
            else if (registros && !(Label_Insertar.Text == "verdadero"))
            {
                LabelResultado_Operacion.Text = "Peso actualizado exitosamente.";
                LabelResultado_Operacion.Visible = true;
            }
            else
            {
                LabelResultado_Operacion.Text = "Error de Inserción. Vuelva a intentar.";
                LabelResultado_Operacion.Visible = true;
            }
            Limpiar_Cajas_Texto();
            DataSet ds = new DataSet();
            ds = objClase_Peso.Recuperar_Peso("completa", "", "");
            GridViewPeso.DataSource = ds.Tables[0];
            GridViewPeso.DataBind();
        }
        catch (Exception error)
        {
            LabelResultado_Operacion.Text = error.Message.ToString();
            LabelResultado_Operacion.Visible = true;
        }
    }
```

La funcion de Registrar_Peso es la siguiente:

```
    public Boolean Registrar_Peso(string categoria, string nombre, string genero, String peso1,
String peso2)
    {
        int registros;
        objConexion = objClase_Conexion.Abrir_Conexion();
        sentencia = String.Format("INSERT INTO PESO(ID_Categoria, Nombre_Peso,
Genero_Peso,Peso_Inicio, Peso_Fin) VALUES('{0}', '{1}', '{2}','{3}','{4}')", categoria, nombre,
genero, peso1, peso2);
        objComando = new SqlCommand(sentencia, objConexion);
        registros = Convert.ToInt32(objComando.ExecuteNonQuery());
        if (registros > 0)
            return true;
        else
            return false;
    }
```

Esta función recibe como parámetros todos y cada uno de los campos que despliega la pantalla, y es aquí donde se arma la sentencia SQL que se ejecutará

para la inserción. Una vez que la sentencia se ejecuta la función devolverá como resultado el valor “True” si el número de registros afectados es mayor a 0, en caso contrario se devolverá el valor de “False” como indicador de que no se realizó la inserción.

5.3.4- Creación de Categoría

La creación de categorías se realiza para ingresar una nueva categoría al sistema o a su vez para modificar ciertos datos de alguna ya creada con anterioridad.

Figura 5.5: Pantalla creación de categoría

Para la creación de una categoría se ingresan los datos en las cajas de texto que se muestran en pantalla, muchos de las cuales contienen el símbolo “*” a un contado, este símbolo indica que el dato es obligatorio. La función que realiza la inserción de la información a la Base de Datos es la siguiente:

```
protected void ImageButtonGuardar_Click(object sender, ImageClickEventArgs e)
{
    try
    {
        Boolean validacion_duplicadoCategoria = false;
        validacion_duplicadoCategoria = objClase_Categoria.Autentificar_Duplicados
(TextBoxNombre.Text);
        if (TextBoxNombre.Text != "")
        {
            Boolean registros;

            if (Label_Insertar.Text == "verdadero" & validacion_duplicadoCategoria == false)
                registros = objClase_Categoria.Registrar_Categoria(TextBoxNombre.Text,
TextBoxDescripcion.Text);
            else
                registros = objClase_Categoria.Actualizar_Categoria(LabelResultado_Operacion.Text,
TextBoxNombre.Text, TextBoxDescripcion.Text);
            if (registros && Label_Insertar.Text == "verdadero")
            {
                LabelResultado_Operacion.Text = "Categoria registrada con éxito.";
                LabelResultado_Operacion.Visible = true;
            }
            else if (registros && !(Label_Insertar.Text == "verdadero"))
            {
                LabelResultado_Operacion.Text = "Categoria actualizada exitosamente.";
                LabelResultado_Operacion.Visible = true;
            }
            else
            {
                LabelResultado_Operacion.Text = "Error de Inserción. Vuelva a intentar.";
                LabelResultado_Operacion.Visible = true;
            }
            Limpiar_Cajas_Texto();
            DataSet ds = new DataSet();
            ds = objClase_Categoria.Recuperar_Categoria("completa", "", "");
            GridViewCategoria.DataSource = ds.Tables[0];
            GridViewCategoria.DataBind();
        }
        else
        {
            LabelResultado_Operacion.Text = "Error de Inserción. No se permiten nombres iguales
ni en blanco.";
        }
    }
    catch (Exception error)
    {
        LabelResultado_Operacion.Text = error.Message.ToString();
        LabelResultado_Operacion.Visible = true;
    }
}
```

Dentro del código mostrado se encuentra la función Registrar_Categoria, misma que se encuentra en la clase Categoría en la capa de Logica del Negocio. Esta función contiene el siguiente código:

```
public Boolean Registrar_Categoria(string nombre, String descripcion)
{
    int registros;
    objConexion = objClase_Conexion.Abrir_Conexion();
    sentencia = String.Format("INSERT INTO CATEGORIA(Nombre_Categoria,
Descripcion_Categoria) VALUES('{0}', '{1}')", nombre, descripcion);
    objComando = new SqlCommand(sentencia, objConexion);
    registros = Convert.ToInt32(objComando.ExecuteNonQuery());
    if (registros > 0)
        return true;
    else
        return false;
}
```

Esta función recibe como parámetros el nombre de la categoría y la descripción, el identificador, el sistema lo genera automáticamente en la base de datos. Estos dos parámetros permiten formar la cadena SQL que se ejecutará para la inserción. Una vez que la sentencia se ejecuta la función devolverá como resultado el valor “True” si el número de registros afectados es mayor a 0, en caso contrario se devolverá el valor de “False” como indicador de que no se realizó la inserción.

5.4- Desarrollo del módulo de Competencias

5.4.1- Inscripción de Clubes

Una vez creada la competencia, cada club podrá inscribirse para participar en la misma. En la pantalla de inscripción de clubes se despliegan las competencias que se encuentren activas, es decir, de acuerdo a la fecha de inicio

y fin de inscripciones las competencias aparecerán o desaparecerán de la lista mostrada.

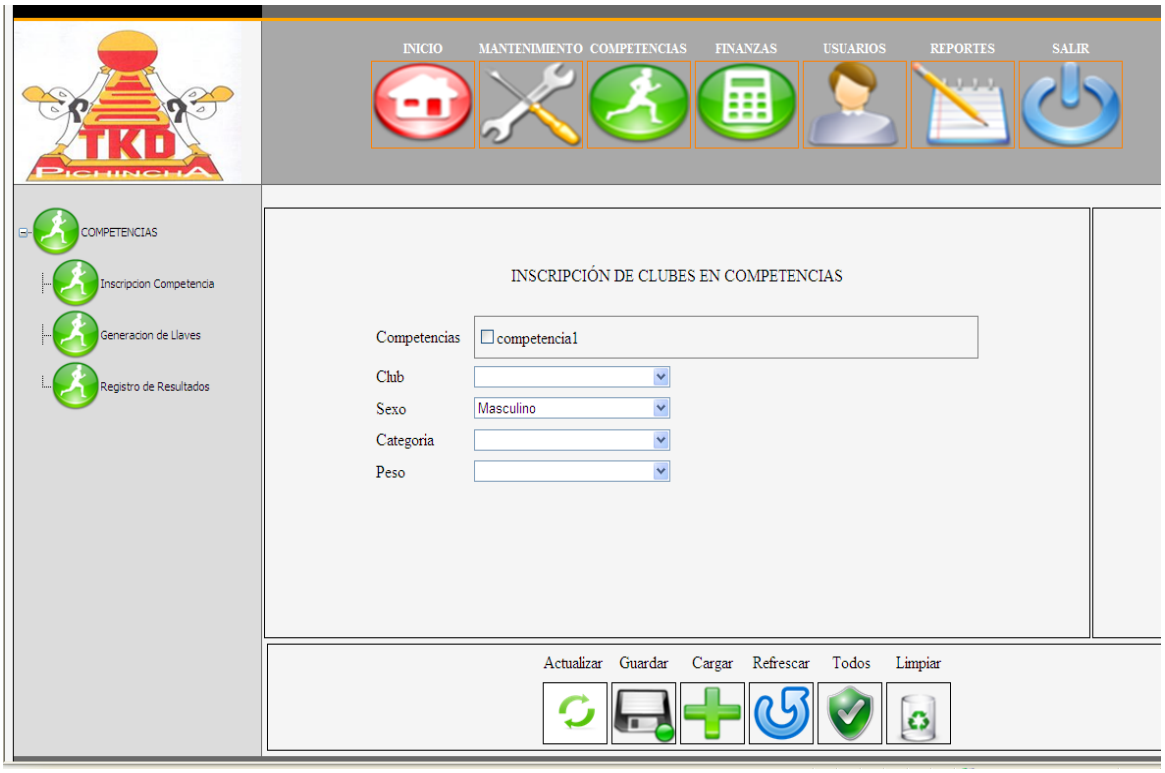


Figura 5.6: Pantalla Inscripción de Clubes

Para esta pantalla se utilizan como funciones principales de carga y registro las siguientes:

- Recuperar_Deportistas
- Inscribir_Club_Competencia

La funcion Recuperar_Deportistas es la siguiente:

```
public DataSet Recuperar_Deportistas(string club, String Sexo)
{
   try
   {
      DataSet dsDeportista = new DataSet();
```

```
            objConexion = objClase_Conexion.Abrir_Conexion();
               sentencia = String.Format("select id_pertenencia_deportista, d.id_deportista,
nombre_deportista +' '+ apellido_deportista as Nombre from deportista d join
pertenencia_club_deportista p on (d.id_deportista = p.id_deportista) where p.id_club = '{0}' and
d.sexo_deportista='{1}' and p.estado_deportista = 1", club, Sexo);

               objAdaptador = new SqlDataAdapter(sentencia, objConexion);
               objAdaptador.Fill(dsDeportista);
               return dsDeportista;
            }

            catch (Exception err)
            {
               Console.WriteLine(err.Message.ToString());
               return null;
            }
         }
```

Esta función permite obtener el listado de deportistas que pertenecen al club seleccionado para poder marcarlos e inscribirlos.

La función Inscribir_Club_Competencia es la siguiente:

```
         public Boolean Inscribir_Club_Competencia(String deportista,String competencia, String
Peso, String llave, int estado)
         {

            int registros;
            objConexion = objClase_Conexion.Abrir_Conexion();
            sentencia = String.Format("INSERT INTO
INSCRIPCION_COMPETENCIA(ID_Pertenencia_Deportista, ID_Competencia, ID_Peso, id_llave,
estado_inscripcion) VALUES('{0}', '{1}', '{2}', '{3}', '{4}')", deportista, competencia, Peso, llave,
estado);

            objComando = new SqlCommand(sentencia, objConexion);
            registros = Convert.ToInt32(objComando.ExecuteNonQuery());

            if (registros > 0)
               return true;
            else
               return false;
         }
```

Esta función guarda los deportistas que se inscriben para una competencia.

5.4.2- Creación de Llaves

Para la generación de llaves, es necesario seleccionar una competencia, el sexo de los participantes, la categoría y el peso.

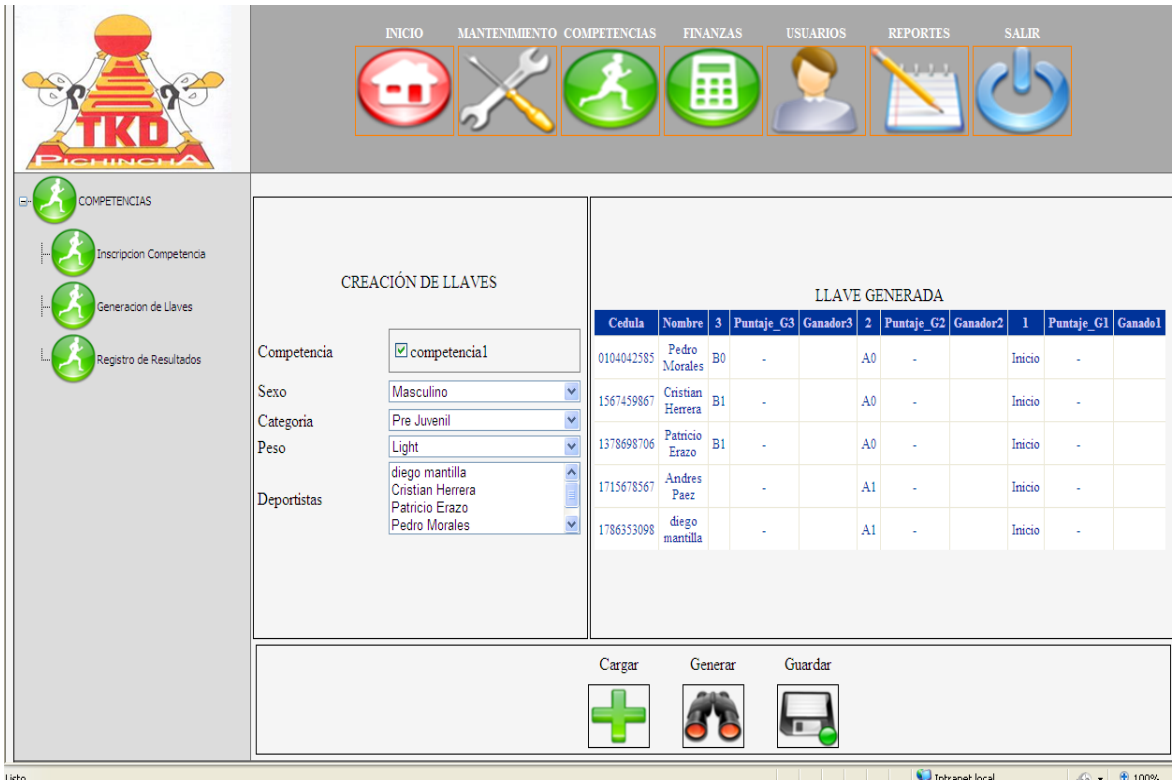


Figura 5.7: Pantalla Creación de Llaves

Para esta pantalla se utilizan como funciones principales de carga y registro las siguientes:

- Recuperar_Deportistas_Inscritos
- Crear_Llave
- Recuperar_Resultados_Llave

La función Recuperar_Deportistas_Inscritos es la siguiente:

```
public DataSet Recuperar_Deportistas_Inscritos(string competencia, String peso)
{
    try
    {
        DataSet dsDeportista = new DataSet();
        objConexion = objClase_Conexion.Abrir_Conexion();

        sentencia = String.Format("select id_llave,d.cedula_deportista as Cedula,
nombre_deportista +' '+ apellido_deportista as Nombre from inscripcion_competencia i join
pertenencia_club_deportista p on (i.id_pertenencia_deportista = p.id_pertenencia_deportista) join
deportista d on (d.id_deportista = p.id_deportista) where i.id_competencia = '{0}' and i.id_peso =
'{1}' and i.estado_inscripcion = 1", competencia, peso);
        objAdaptador = new SqlDataAdapter(sentencia, objConexion);
        objAdaptador.Fill(dsDeportista);
        return dsDeportista;
    }
    catch (Exception err)
    {
        Console.WriteLine(err.Message.ToString());
        return null;
    }
}
```

Esta función recupera los deportistas inscritos de acuerdo a la competencia y al peso en el que fueron inscritos, con este listado se procede a generar la llave de participación respectiva.

La función Crear_Llave es la siguiente:

```
public DataTable Crear_LLave(DataTable dtGruposFinales)
    {
        if (!dtGruposFinales.Columns.Contains("Grupo"))
            dtGruposFinales.Columns.Add("Grupo");
        foreach (DataRow dr in dtGruposFinales.Rows)
            dr["Grupo"] = "Inicio";
        dtGruposFinales.AcceptChanges();
        dtGruposFinales = CrearGrupos(dtGruposFinales,
dtGruposFinales.Rows[0]["Grupo"].ToString(), objUtilGrupos.SiguienteSubgrupo());
        DataTable dtCompara = new DataTable();
        for (; ; ) //bucle dinamico
        {
            dtCompara = new DataTable();
            dtCompara = CrearSubgrupos(dtGruposFinales);
            if (dtCompara.Equals(dtGruposFinales))
                break;
            else
```

```
            dtGruposFinales = dtCompara;
        }
        DataSet dsPrueba = new DataSet();
        dsPrueba.Tables.Add(dtCompara.Copy());
        dsPrueba.WriteXml(strPath);
        //Tabla desordenada dsPrueba
        return Organiza(dsPrueba);
    }
```

Esta función crea randomicamente la llave de participación de cada deportista, esta función a su vez llama subfunciones que van organizando de manera estructurada la llave ubicando a los participantes aleatoriamente en posiciones para competir.

La función Recuperar_Resultados_Llave es la siguiente:

```
    public Boolean Registrar_Resultados_Llave(object[] arreglo)
    {
        int registros;
        objConexion = objClase_Conexion.Abrir_Conexion();
        sentencia = String.Format("INSERT INTO resultados_llave(ID, cedula, nombre, grupo10,
puntaje10, ganador10, grupo9, puntaje9, ganador9, grupo8, puntaje8, ganador8, grupo7, puntaje7,
ganador7, grupo6, puntaje6, ganador6, grupo5, puntaje5, ganador5, grupo4, puntaje4, ganador4,
grupo3, puntaje3, ganador3, grupo2, puntaje2, ganador2, grupo1, puntaje1, ganador1)
VALUES('{0}', '{1}', '{2}','{3}', '{4}', '{5}','{6}', '{7}', '{8}','{9}', '{10}', '{11}','{12}', '{13}', '{14}','{15}', '{16}',
'{17}','{18}', '{19}', '{20}','{21}', '{22}', '{23}','{24}', '{25}', '{26}','{27}', '{28}', '{29}','{30}', '{31}','{32}')",
arreglo[0], arreglo[1], arreglo[2], arreglo[3], arreglo[4], arreglo[5], arreglo[6], arreglo[7], arreglo[8],
arreglo[9], arreglo[10], arreglo[11], arreglo[12], arreglo[13], arreglo[14], arreglo[15], arreglo[16],
arreglo[17], arreglo[18], arreglo[19], arreglo[20], arreglo[21], arreglo[22], arreglo[23], arreglo[24],
arreglo[25], arreglo[26], arreglo[27], arreglo[28], arreglo[29], arreglo[30], arreglo[31], arreglo[32]);
        objComando = new SqlCommand(sentencia, objConexion);
        registros = Convert.ToInt32(objComando.ExecuteNonQuery());
        if (registros > 0)
        {
            objClase_Conexion.Cerrar_Conexion();
            return true;
        }
        else
        {
            objClase_Conexion.Cerrar_Conexion();
            return false;
        }
    }
```

Esta función registra la llave que se obtiene como resultado de la generación hecha previamente, esta inserción está diseñada para manejar

algunos niveles de participaciones en función de la cantidad de competidores inscritos en la competencia.

5.5- Desarrollo del módulo Financiero

5.5.1- Pago de Inscripción de Clubes

La inscripción de clubes se da cuando un nuevo club es creado en el sistema, una vez que se ha dado de alta al club el mismo debe pagar un rubro por concepto de inscripción.

Figura 5.8: Pantalla Pago Inscripción de Clubes

Para esta pantalla se utilizan como funciones principales de registro las siguientes:

- Registrar_Financiero_Movimientos
- Actualizar_Financiero_Movimientos

La función Registrar_Financiero_Movimientos es la siguiente:

```
public Boolean Registrar_Financiero_Movimientos(DateTime fecha, string remitente, string
descripcion, int user, string comprobante, string tipo, string club, string valor)
    {
        int registros;
        objConexion = objClase_Conexion.Abrir_Conexion();
        sentencia = String.Format("INSERT INTO FINANCIERO_MOVIMIENTOS(Fecha_Hora,
Descripcion_Movimiento, Usuario_Operador, Firma_Recepcion, Numero_Comprobante,
Tipo_Ingreso, ID_Club, Valor_Club) VALUES('{0}', '{1}', '{2}', '{3}', '{4}', '{5}','{6}', '{7}')",
fecha.ToString("dd/MM/yyyy"), descripcion, user, remitente, comprobante, tipo, club, valor);
        objComando = new SqlCommand(sentencia, objConexion);
        registros = Convert.ToInt32(objComando.ExecuteNonQuery());
        if (registros > 0)
            return true;
        else
            return false;
    }
```

Esta funcion registra el pago hecho por cada club que se inscribe, recibe como parámetros la fecha, remitente, descripcion, usuario que realiza la transaccion y guarda en la tabla respectiva los datos del club inscrito.

La función Actualizar_Financiero_Movimientos es la siguiente:

```
public Boolean Actualizar_Financiero_Movimientos(string movimiento, DateTime fecha, string
remitente, string descripcion, int user, string tipo, string club, string valor)
    {
        int registros;
        objConexion = objClase_Conexion.Abrir_Conexion();
        sentencia = String.Format("UPDATE FINANCIERO_MOVIMIENTOS SET
Fecha_Hora='{0}', Descripcion_Movimiento='{1}',Usuario_Operador = '{2}',
Firma_Recepcion='{3}',Tipo_Ingreso ='{4}', ID_Club ='{5}', Valor_Club = '{6}'  where ID_Movimiento
= '{7}'", fecha.ToString("dd/MM/yyyy"), descripcion, user, remitente, tipo, club, valor, movimiento);
        objComando = new SqlCommand(sentencia, objConexion);
        registros = Convert.ToInt32(objComando.ExecuteNonQuery());
        if (registros > 0)
            return true;
        else
            return false;
    }
```

Esta funcion actualiza los datos de un pago de inscripción de un club que previamente ha sido inscrito. Recibe como parámetros la fecha, remitente, descripcion, usuario que realiza la transaccion, codigo previo de inscripcion y actualiza en la tabla respectiva los datos del club inscrito.

5.5.2- Pago de Inscripción de Deportistas

La inscripción de deportistas se da por dos motivos, cuando es un pago por carnetización o cuando existe una competencia, en cualquiera de los dos casos el deportista debe cancelar un rubro por uno de los dos conceptos.

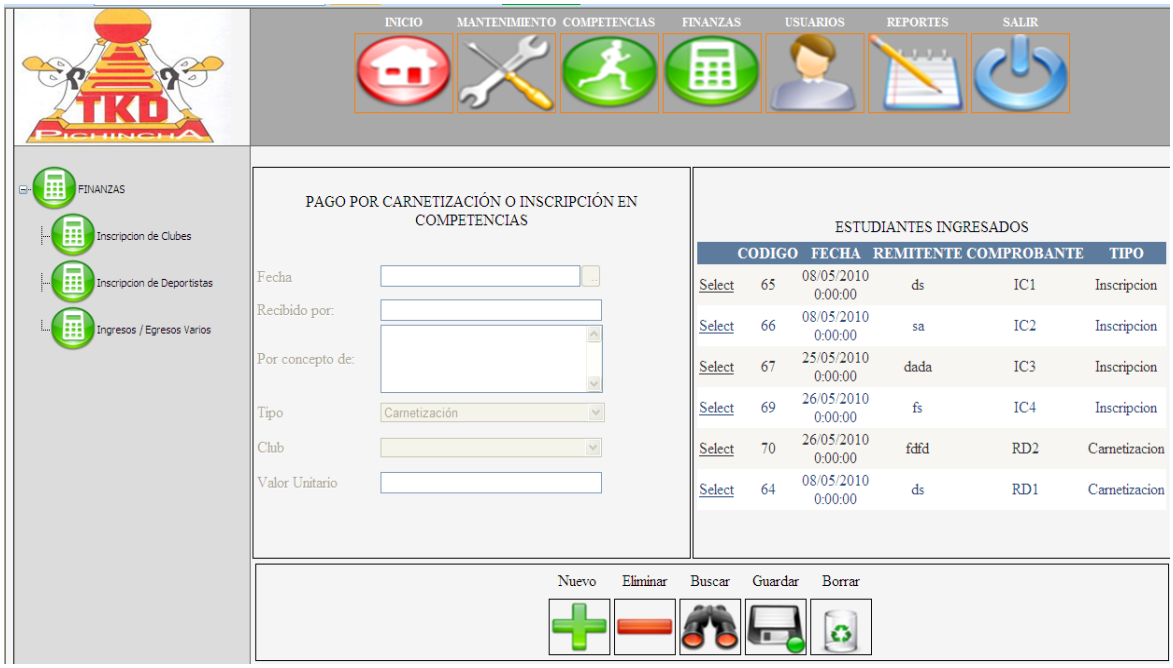


Figura 5.9: Pantalla Pago Inscripción de Deportistas Clubes

Para esta pantalla se utiliza como función principal de registro la siguiente:

- Registrar_DetalleMovimientoDeportista

La función Registrar_DetalleMovimientoDeportista es la siguiente:

```
public Boolean Registrar_DetalleMovimientoDeportista(string movimiento, string deportista,
double valor)
      {
         int registros;

          objConexion = objClase_Conexion.Abrir_Conexion();
         sentencia = String.Format("INSERT INTO
FINANCIERO_DETALLE_DEPORTISTA(ID_Movimiento, ID_Deportista, valor_unitario)
VALUES('{0}', '{1}', '{2}')", movimiento, deportista, valor);

          objComando = new SqlCommand(sentencia, objConexion);
         registros = Convert.ToInt32(objComando.ExecuteNonQuery());

          if (registros > 0)
            return true;
         else
            return false;
      }
```

Esta función permite almacenar los datos en la tabla de detalle movimiento discriminando por carnetización o por competencia. Recibe como parámetros el tipo de movimiento, el código del deportista y el valor que se paga.

5.5.3- Ingresos y Egresos Varios

Los ingresos y egresos varios corresponden a todo valor o rubro que sale o entra a la asociación por diferentes conceptos, ya sean por competencias, gastos de caja chica, etc.

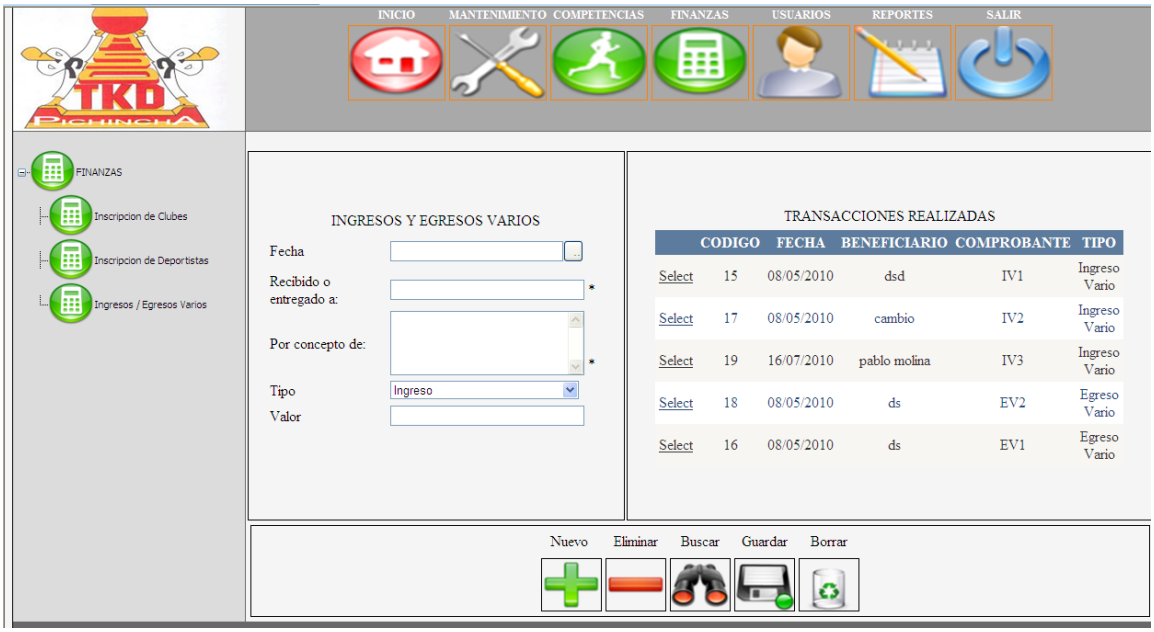


Figura 5.10: Pantalla Ingresos y Egresos Varios

Para esta pantalla se utilizan como funciones principales de registro las siguientes:

- Registrar_Ingreso_Egreso
- Actualizar_Ingreso_Egreso

La función Registrar_Ingreso_Egreso es la siguiente:

```
public Boolean Registrar_Ingreso_Egreso(DateTime fecha, string remitente, string descripcion,
int user, string comprobante, string tipo, string valor)
    {   int registros;
        objConexion = objClase_Conexion.Abrir_Conexion();
        sentencia = String.Format("INSERT INTO
FINANCIERO_INGRESO_EGRESO(Tipo_Transaccion, Beneficiario, Descripcion_Transaccion,
Fecha_Hora, Usuario_Operador, Valor_Total, Numero_Comprobante) VALUES('{0}', '{1}', '{2}',
'{3}', '{4}', '{5}','{6}')", tipo, remitente, descripcion, fecha.ToString("dd/MM/yyyy"), user, valor,
comprobante);
        objComando = new SqlCommand(sentencia, objConexion);
        registros = Convert.ToInt32(objComando.ExecuteNonQuery());
        if (registros > 0)
            return true;
        else
            return false;
    }
```

Esta función permite registrar cualquier ingreso o egreso que se realice separándolos por el tipo de movimiento. Los parámetros que recibe son fecha,

remitente, descripción, usuario, número de comprobante, tipo de transacción(Ingreso o Egreso) y el valor correspondiente.

La función Actualizar_Ingreso_Egreso es la siguiente:

```
public Boolean Actualizar_Ingreso_Egreso(string movimiento, DateTime fecha, string remitente,
string descripcion, int user, string comprobante, string tipo, string valor)
    {
        int registros;
        objConexion = objClase_Conexion.Abrir_Conexion();
        sentencia = String.Format("UPDATE FINANCIERO_INGRESO_EGRESO SET
Tipo_Transaccion = '{0}', Beneficiario ='{1}', Descripcion_Transaccion = '{2}', Fecha_Hora ='{3}',
Usuario_Operador ='{4}', Valor_Total ='{5}' where ID_Transaccion = '{6}'", tipo, remitente,
descripcion, fecha.ToString("dd/MM/yyyy"), user, valor, movimiento);
        objComando = new SqlCommand(sentencia, objConexion);
        registros = Convert.ToInt32(objComando.ExecuteNonQuery());
        if (registros > 0) return true;
        else return false;
    }
```

Esta función permite actualizar cualquier ingreso o egreso que se realice de acuerdo al tipo de movimiento. Los parámetros que recibe son el codigo de ingreso_egreso generado previamente, fecha, remitente, descripción, usuario, número de comprobante, tipo de transacción(Ingreso o Egreso) y el valor correspondiente. Con esto se actualiza en la tabla de ingresos y egresos.

5.6- Desarrollo del módulo de Usuarios

5.6.1- Creación de Usuarios

La creación de usuarios permite crear y/o modificar la información de usuarios que accederán al sistema, asignándoles un nivel de permisos para que puedan acceder a cada módulo.

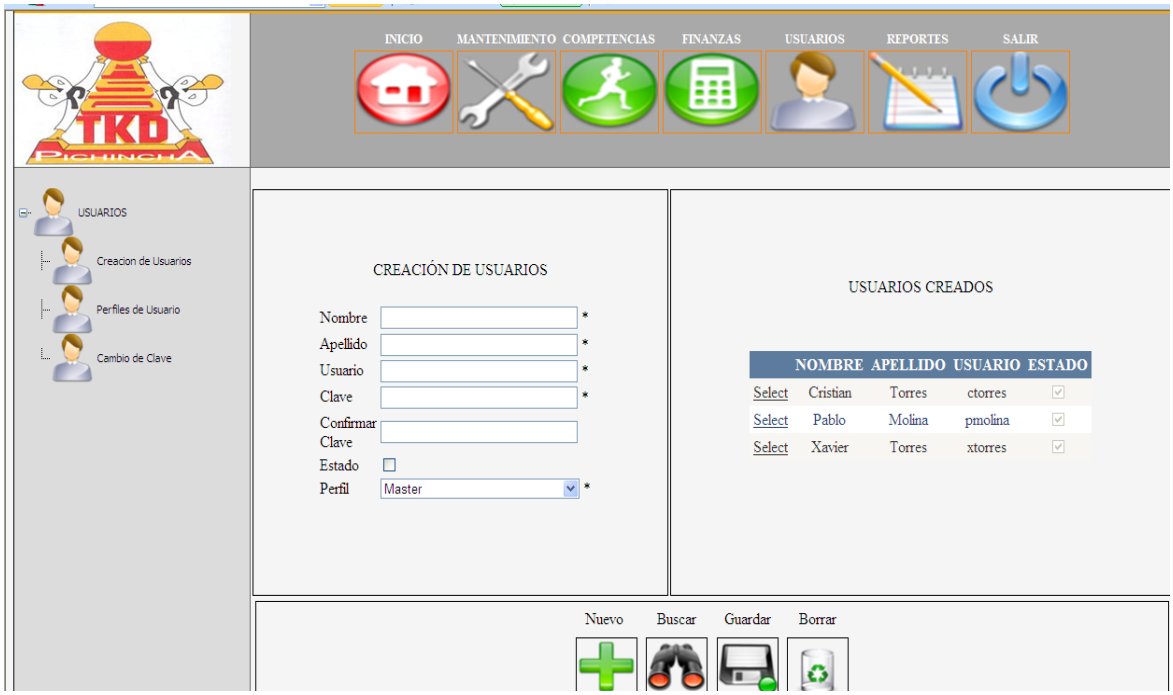


Figura 5.11: Pantalla Creación de Usuarios

Para esta pantalla se utiliza como funciones principales de registro las siguientes:

- Registrar_Usuario
- Actualizar_Usuario

La función Registrar_Usuario es la siguiente:

```
public Boolean Registrar_Usuario(string nombre, string apellido, string Usuario, Int32 estado,
string clave, string perfil)
    {
        int registros;
        objConexion = objClase_Conexion.Abrir_Conexion();
        sentencia = String.Format("INSERT INTO USUARIO(Nombre_Usuario, Apellido_Usuario,
Nickname_Usuario, Estado, Clave_Usuario, ID_Perfil) VALUES('{0}', '{1}', '{2}', '{3}', '{4}', '{5}')",
nombre, apellido, Usuario, estado, clave, perfil);
        objComando = new SqlCommand(sentencia, objConexion);
        registros = Convert.ToInt32(objComando.ExecuteNonQuery());
        if (registros > 0)
            return true;
        else
            return false;
    }
```

Esta función permite guardar la información en la base de datos de un usuario nuevo. Los parámetros que recibe son todos los datos del usuario y adicional el perfil de acceso que tendrá el mismo.

La función Registrar_Usuario es la siguiente:

```
public Boolean Actualizar_Usuario(string codigo, string nombre, string apellido, string Usuario,
Int32 estado, string clave, string perfil)
      {
         int registros;
         objConexion = objClase_Conexion.Abrir_Conexion();
         sentencia = String.Format("UPDATE USUARIO SET Nombre_Usuario = '{0}',
Apellido_Usuario = '{1}', Nickname_Usuario = '{2}', Estado = '{3}', Clave_Usuario = '{4}', ID_Perfil =
'{5}'  WHERE ID_Usuario = '{6}'", nombre, apellido, Usuario, estado, clave, perfil, codigo);
         objComando = new SqlCommand(sentencia, objConexion);
         registros = Convert.ToInt32(objComando.ExecuteNonQuery());
         if (registros > 0)
            return true;
         else
            return false;
      }
```

Esta función permite actualizar la información en la base de datos de un usuario creado previamente. Los parámetros que recibe son todos los datos del usuario y adicional el perfil de acceso y el código de usuario creado al guardar la información por primera vez.

5.6.2- Perfiles de Usuarios

Los perfiles de usuario hacen referencia al nivel de acceso que tendrán los usuarios, es decir, a que módulos o sub-módulos podrán ingresar de acuerdo a las asignaciones y permisos habilitados en cada perfil de usuario creado.

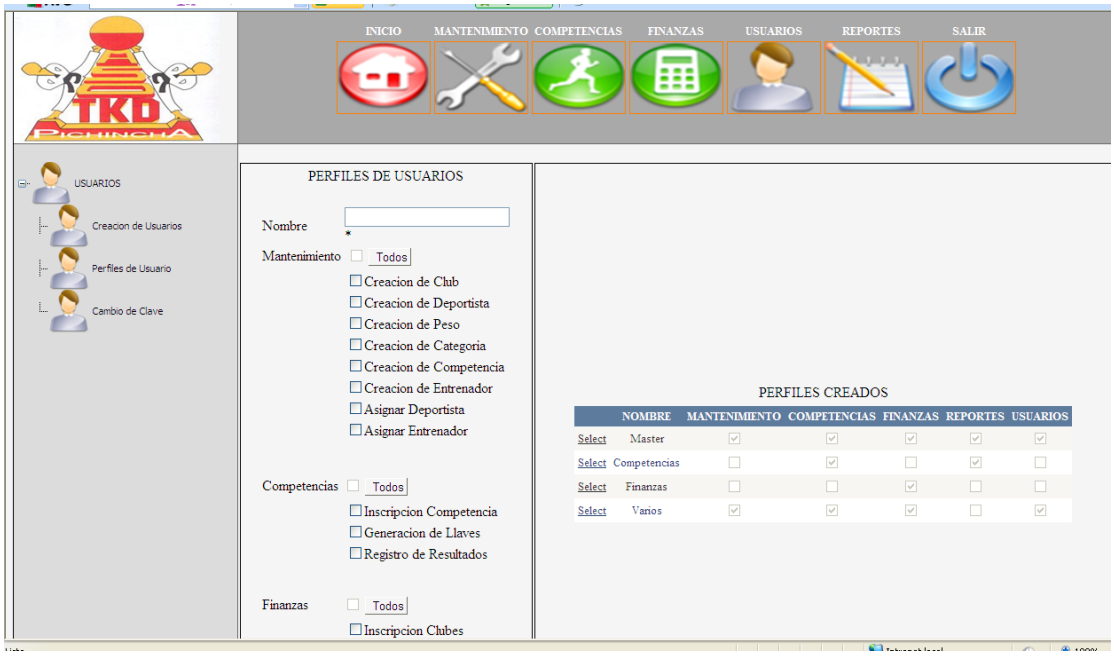


Figura 5.12: Pantalla Perfiles de Usuarios

Para esta pantalla se utiliza como funciones principales de registro las siguientes:

- Registrar_Perfil_Usuario
- Actualizar_Perfil_Usuario

La función Registrar_Perfil_Usuario es la siguiente:

```
public Boolean Registrar_Perfil_Usuario(string nombre, int mantenimiento , int creacion_club, int
creacion_deportista, int creacion_peso, int creacion_categoria, int creacion_competencia, int
creacion_entrenador, int asignacion_deportista, int asignacion_entrenador, int competencia, int
inscripcion_competencia, int generacion_llaves, int registro_resultados, int finanzas, int
inscripcion_club, int inscripcion_deportista, int ingresos, int reportes, int usuarios)
      {
         int registros; objConexion = objClase_Conexion.Abrir_Conexion();
         sentencia = String.Format("INSERT INTO PERFIL_USUARIO(Nombre_Perfil,
Permiso_Mantenimiento, Permiso_Creacion_Club, Permiso_Creacion_Deportista,
Permiso_Creacion_Peso, Permiso_Creacion_Categoria, Permiso_Creacion_Competencia,
Permiso_Creacion_Entrenador, Permiso_Asignacion_Deportista,
Permiso_Asignacion_Entrenador, Permiso_Competencias, Permiso_Inscripcion_Competencia,
Permiso_Generacion_Llaves, Permiso_Registro_Resultados, Permiso_Finanzas,
Permiso_Inscripcion_Club, Permiso_Inscripcion_Deportista, Permiso_Ingresos,
Permiso_Reportes, Permiso_Usuarios) VALUES('{0}', '{1}', '{2}', '{3}', '{4}', '{5}', '{6}', '{7}', '{8}', '{9}',
```

```
'{10}', '{11}', '{12}', '{13}', '{14}', '{15}', '{16}', '{17}', '{18}', '{19}')", nombre, mantenimiento,
creacion_club, creacion_deportista, creacion_peso, creacion_categoria, creacion_competencia,
creacion_entrenador, asignacion_deportista, asignacion_entrenador, competencia,
inscripcion_competencia, generacion_llaves, registro_resultados, finanzas, inscripcion_club,
inscripcion_deportista, ingresos, reportes, usuarios);
            objComando = new SqlCommand(sentencia, objConexion);
            registros = Convert.ToInt32(objComando.ExecuteNonQuery());
            if (registros > 0) return true;
            else return false;
        }
```

Esta función permite guardar en la base de datos la información de cada perfil de usuario creado, cada check de la pantalla representa un campo en la base de datos que asigna el permiso a cada opción del sistema. Los parámetros que recibe son nombre de perfil, cada check marcado o desmarcado en la pantalla.

La función Actualizar_Perfil_Usuario es la siguiente:

```
public Boolean Actualizar_Perfil_Usuario(string codigo, string nombre, int mantenimiento, int
creacion_club, int creacion_deportista, int creacion_peso, int creacion_categoria, int
creacion_competencia, int creacion_entrenador, int asignacion_deportista, int
asignacion_entrenador, int competencia, int inscripcion_competencia, int generacion_llaves, int
registro_resultados, int finanzas, int inscripcion_club, int inscripcion_deportista, int ingresos, int
reportes, int usuarios)
        {
            int registros;
            objConexion = objClase_Conexion.Abrir_Conexion();
            sentencia = String.Format("UPDATE PERFIL_USUARIO SET Nombre_Perfil = '{0}',
Permiso_Mantenimiento = {1}, Permiso_Creacion_Club = {2}, Permiso_Creacion_Deportista = {3},
Permiso_Creacion_Peso = {4}, Permiso_Creacion_Categoria = {5},
Permiso_Creacion_Competencia = {6}, Permiso_Creacion_Entrenador = {7},
Permiso_Asignacion_Deportista = {8}, Permiso_Asignacion_Entrenador = {9},
Permiso_Competencias = {10}, Permiso_Inscripcion_Competencia = {11},
Permiso_Generacion_Llaves = {12}, Permiso_Registro_Resultados = {13}, Permiso_Finanzas =
{14}, Permiso_Inscripcion_Club = {15}, Permiso_Inscripcion_Deportista = {16}, Permiso_Ingresos
= {17}, Permiso_Reportes = {18}, Permiso_Usuarios = {19}  WHERE ID_Perfil = '{20}'", nombre,
mantenimiento, creacion_club, creacion_deportista, creacion_peso, creacion_categoria,
creacion_competencia, creacion_entrenador, asignacion_deportista, asignacion_entrenador,
competencia, inscripcion_competencia, generacion_llaves, registro_resultados, finanzas,
inscripcion_club, inscripcion_deportista, ingresos, reportes, usuarios, codigo);
            objComando = new SqlCommand(sentencia, objConexion);
            registros = Convert.ToInt32(objComando.ExecuteNonQuery());
            if (registros > 0)
                return true;
            else
                return false;
        }
```

Esta función permite actualizar en la base de datos la información de cada perfil de usuario creado previamente, cada check de la pantalla representa un campo en la base de datos que asigna el permiso a cada opción del sistema. Los parámetros que recibe son nombre de perfil, cada check marcado o desmarcado en la pantalla.

5.6.3- Cambio de Clave de Usuarios

El cambio de clave de usuario va relacionado con el usuario que se encuentra conectado al sistema, es decir, cada usuario una vez que se conecte podrá cambiar solo su contraseña.

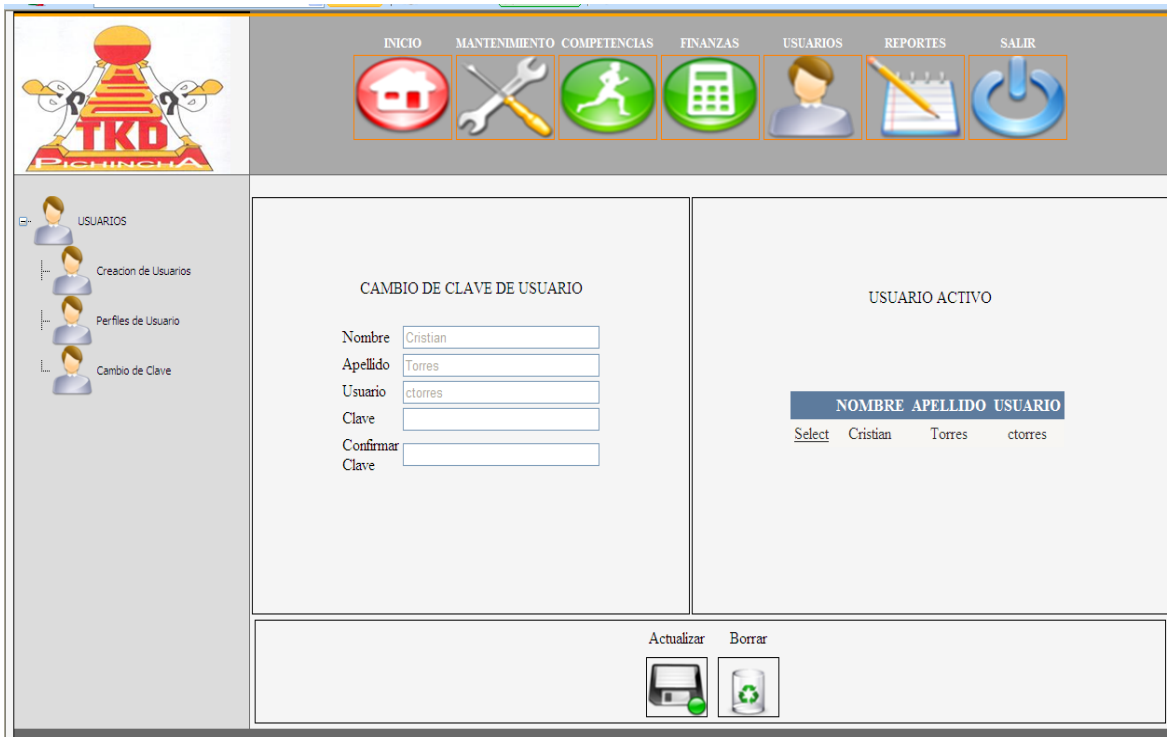


Figura 5.13: Pantalla Clave de Usuarios

Para esta pantalla se utiliza como función principal de registro la siguiente:

- Actualizar_Clave_Usuario

La función Actualizar_Clave_Usuario es la siguiente:

```
public Boolean Actualizar_Clave_Usuario(string codigo, string clave)
      {
         int registros;
         objConexion = objClase_Conexion.Abrir_Conexion();
         sentencia = String.Format("UPDATE USUARIO SET Clave_Usuario = '{0}' WHERE
ID_Usuario = '{1}'", clave, codigo);
         objComando = new SqlCommand(sentencia, objConexion);
         registros = Convert.ToInt32(objComando.ExecuteNonQuery());
         if (registros > 0)
            return true;
         else
            return false;
      }
```

Esta función permite realizar la actualización o cambio de contraseña de cada usuario que se encuentre conectado al sistema. Los parámetros que recibe son el código del usuario y la nueva clave del mismo.

CAPÍTULO VI

PRUEBAS

Las pruebas se realizan integralmente durante todas las etapas de la metodología XP; de manera previa en la toma de requerimientos, planificación y diseño se realizan test conjuntamente con el usuario de los prototipos de pantallas y funciones que se van a codificar en el proceso de desarrollo.

Con éstas pruebas lo que se busca es plasmar en papel, la ideología del sistema de cómo lo visualiza el usuario para dar solución a una necesidad; éstas pruebas son conocidas como de aceptación; es decir son el análisis del sistema con el cliente basándose en las historias de usuario, las que deben ser cumplidas a cabalidad.

Conforme se va construyendo el código para las clases de negocio, se realizan varias pruebas con el objetivo de controlar la simplicidad, estandarización, flexibilidad y ausencia de código repetitivo, éstas pruebas son conocidas como de unidad y son aplicadas a las funciones y clases más importantes del sistema ya sea con alguna herramienta informática o a través de pruebas de escritorio[31] realizadas por el mismo programador.

Una vez finalizado el sistema se realizan pruebas con el usuario para verificar que se han cumplido las historias de usuario de manera satisfactoria y

[31] Prueba de escritorio: es una simulación con datos y variables ficticias que permiten validar el código de manera manual

cada uno de los módulos actúa de manera integral dentro del sistema, éstas pruebas son conocidas como pruebas de integración y son generadas a través de una plantilla o formulario en el que el usuario junto al programador validan el impacto global de los datos de simulación utilizados en el sistema

6.1- Pruebas de Aceptación de Usuario

Para el desarrollo de éstas pruebas, se ha creado una plantilla de aceptación para cada historia de usuario; en la que se plasma la necesidad del cliente, la evaluación de datos preliminares en caso de ser necesario, descripción de las pantallas y su funcionalidad para el cumplimiento del requerimiento solicitado en la historia de usuario y el grado de aceptación por parte del cliente

Para la elaboración correcta de estas pruebas se debe tener contacto permanente con el cliente; es decir, que el cliente es parte activa del equipo de desarrollo.

A continuación se describen las pruebas de aceptación de usuario para el sistema:

Tabla 6.1: Prueba de Aceptación para la historia de usuario; Ingreso y utilización del sistema por parte del usuario

<table>
<tr><th>HISTORIA DE USUARIO</th><th>OBSERVACIONES</th><th>RESULTADO</th></tr>
<tr><td>Ingreso y utilización del sistema por parte del usuario</td><td rowspan="2"><u>No se necesitan datos preliminares.</u>
Se diseñan tres pantallas; Perfiles de usuario, Creación de Usuarios y Cambio de Clave.
En la pantalla <u>Perfiles de Usuario</u> se registra el nombre que describe al perfil, se asignan las pantallas a las que tiene acceso a través de checks de aceptación en un listado, mismo que está ordenado por módulos.
En la pantalla <u>Creación de usuarios</u> se registra el Nombre del usuario, una clave y se le asigna un perfil.
En la pantalla <u>Cambio de Clave</u> se realiza el proceso de cambio de clave para esto se escribe la clave actual y a continuación la clave nueva.
Esta pantalla tiene un grupo de botones de administración de datos como: nuevo, buscar, guardar, borrar y una tabla donde se visualizan los datos ingresados.</td><td rowspan="2">Aceptado por la Directiva de la ATKDP</td></tr>
<tr><td>Registrar la información actualizada de los usuarios que utilizan el software y permitir su acceso al sistema de acuerdo a perfiles</td></tr>
</table>

Tabla 6.2: Prueba de Aceptación para la historia de usuario; Registro de clubes administrados por la Asociación de Tae Kwon Do de Pichincha (ATKDP).

<table>
<tr><th>HISTORIA DE USUARIO</th><th>OBSERVACIONES</th><th>RESULTADO</th></tr>
<tr><td>Registro de clubes administrados por la Asociación de Tae Kwon Do de Pichincha (ATKDP)</td><td rowspan="2"><u>No se necesitan datos preliminares.</u>

• Se diseña una pantalla; Creación de club.

• En la pantalla <u>Creación de club</u> se registran los datos principales de cada club como son: nombre, dirección, teléfono, teléfono, representantes, email, página web, año de creación.

• Esta pantalla tiene un grupo de botones de administración de datos como: nuevo, eliminar, buscar, guardar, borrar y una tabla donde se visualizan los datos ingresados.</td><td rowspan="2">Aceptado por la Directiva de la ATKDP.</td></tr>
<tr><td>Registrar la información actualizada de los clubes que administra la ATKDP.</td></tr>
</table>

Tabla 6.3: Prueba de Aceptación para la historia de usuario; Registro de entrenadores correspondientes a los clubes administrados por la ATKDP.

HISTORIA DE USUARIO	OBSERVACIONES	RESULTADO
Registro de entrenadores correspondientes a los clubes administrados por la ATKDP **Registrar la información actualizada de entrenadores de los clubes que pertenecen a la ATKDP.**	***Se necesitan datos preliminares.*** • Se diseñan dos pantallas; Creación de Entrenadores y Asignación de Entrenadores a un club. • En la pantalla *Creación de Entrenadores* se registran los datos principales de un entrenador como son: cédula de identidad, nombre, teléfono, email, sexo • En la pantalla *Asignación de Entrenadores a un club* se realiza la asignación de un entrenador a un club, para esta pantalla se necesita como dato preliminar al club y al entrenador; además datos propios de la pantalla como son: fecha de asignación, alguna observación, y el estado para ver si esta activo o inactivo el entrenador en un club • Esta pantalla tiene un grupo de botones de administración de datos como: nuevo, eliminar, buscar, guardar, borrar y una tabla donde se visualizan los datos ingresados.	**Aceptado por la Directiva de la ATKDP**

Tabla 6.4: Prueba de Aceptación para la historia de usuario; Registro de categorías y pesos reglamentarios para competir en un torneo, campeonato o selectivo organizado por la ATKDP.

<table>
<tr><th>HISTORIA DE USUARIO</th><th>OBSERVACIONES</th><th>RESULTADO</th></tr>
<tr><td>Registro de categorías y pesos reglamentarios para competir en un torneo, campeonato o selectivo organizado por la ATKDP</td><td rowspan="2"><u>Se necesitan datos preliminares.</u>

• Se diseñan dos pantallas; Creación de Categoría y Creación de Peso.
• En la pantalla <u>Creación de Categoría</u> se registran los datos principales de una categoría como son: nombre, descripción
• En la pantalla <u>Creación de Peso</u> se realiza el registro de los datos principales de un peso como son: nombre, sexo, rango de peso; y la asignación de un peso a una categoría, para esto se necesita como dato preliminar a la categoría
• Esta pantalla tiene un grupo de botones de administración de datos como: nuevo, eliminar, buscar, guardar, borrar y una tabla donde se visualizan los datos ingresados.</td><td rowspan="2">Aceptado por la Directiva de la ATKDP</td></tr>
<tr><td>Registrar la información actualizada de las categorías y pesos reglamentarios utilizados para competir en un torneo organizado por la ATKDP.</td></tr>
</table>

Tabla 6.5: Prueba de Aceptación para la historia de usuario; Registro de deportistas correspondientes a los clubes administrados por la ATKDP.

HISTORIA DE USUARIO	OBSERVACIONES	RESULTADO
Registro de deportistas correspondientes a los clubes administrados por la ATKDP **Registrar la información actualizada de los deportistas registrados en la ATKDP.**	***Se necesitan datos preliminares.*** • Se diseñan dos pantallas; Creación de Deportistas y Asignación de Deportistas a un club. • En la pantalla *Creación de Deportistas* se registran los datos principales de un deportista como son: cédula de identidad, nombre, teléfono, dirección, email, sexo, fecha de nacimiento, tipo de deportista • En la pantalla *Asignación de Deportistas a un club* se realiza la asignación de un deportista a un club, para esta pantalla se necesita como dato preliminar al club y al deportista; además datos propios de la pantalla como son: fecha de asignación, alguna observación, y el estado para ver si esta activo o inactivo el deportista en un club • Esta pantalla tiene un grupo de botones de administración de datos como: nuevo, eliminar, buscar, guardar, borrar y una tabla donde se visualizan los datos ingresados.	**Aceptado por la Directiva de la ATKDP**

Tabla 6.6: Prueba de Aceptación para la historia de usuario; Registro de torneos, campeonatos o selectivos organizados por la ATKDP.

<table>
<tr><th>HISTORIA DE USUARIO</th><th>OBSERVACIONES</th><th>RESULTADO</th></tr>
<tr><td>Registro de torneos, campeonatos o selectivos organizados por la ATKDP</td><td rowspan="2"><u>No se necesitan datos preliminares.</u>

• Se diseña una pantalla; Creación de Competencia.

• En la pantalla <u>Creación de Competencia</u> se registran los datos principales de un campeonato, torneo o selectivo como son: nombre, fechas de inicio y fin de inscripciones, evaluador, tipo de competencia, categoría, lugar, fecha de inicio y fin de la competencia.

Esta pantalla tiene un grupo de botones de administración de datos como: nuevo, eliminar, buscar, guardar, borrar y una tabla donde se visualizan los datos ingresados</td><td rowspan="2">Aceptado por la Directiva de la ATKDP</td></tr>
<tr><td>Registrar la información actualizada de los torneos, campeonatos o selectivos organizados por la ATKDP.</td></tr>
</table>

Tabla 6.7: Prueba de Aceptación para la historia de usuario; Inscripción de clubes a participar en un determinado torneo, campeonato o selectivo organizado por la ATKDP

<table>
<tr><th>HISTORIA DE USUARIO</th><th>OBSERVACIONES</th><th>RESULTADO</th></tr>
<tr><td>Inscripción de clubes a participar en un determinado torneo, campeonato o selectivo organizado por la ATKDP</td><td rowspan="2"><u>Se necesitan datos preliminares.</u>

• Se diseña una pantalla; Inscripción de clubes en competencias.

• En la pantalla <u>Inscripción de clubes en competencias</u> se realiza el proceso de inscripción de un club en un determinado torneo, campeonato o selectivo; para esta pantalla se necesita como dato preliminar a la competencia, club, sexo, categoría, peso, deportista.

• Estas pantallas tienen un grupo de botones de administración de datos como: actualizar, guardar, cargar, refrescar, todos, limpiar.</td><td rowspan="2">Aceptado por la Directiva de la ATKDP</td></tr>
<tr><td>Inscribir a los clubes a participar en un determinado torneo, campeonato o selectivo organizado por la ATKDP.</td></tr>
</table>

Tabla 6.8: Prueba de Aceptación para la historia de usuario; Elaboración automática de las llaves para un determinado torneo, campeonato o selectivo organizado por la ATKDP

<table>
<tr><th>HISTORIA DE USUARIO</th><th>OBSERVACIONES</th><th>RESULTADO</th></tr>
<tr><td>Elaboración automática de las llaves para un determinado torneo, campeonato o selectivo organizado por la ATKDP.</td><td rowspan="2"><u>Se necesitan datos preliminares.</u>

• Se diseña una pantalla; Generación de llaves.

• En la pantalla <u>Generación de llaves</u> se realiza el proceso de generar la llave tantas veces como se quiera ya que la generación es aleatorio con respecto a los deportistas, y guardar la llave generada; para esta pantalla se necesita como dato preliminar a la competencia, club, sexo, categoría, peso, deportista.

• Estas pantallas tienen un grupo de botones de administración de datos como: generar, guardar, cargar; y una tabla donde se visualiza la llave generada.</td><td rowspan="2">Aceptado por la Directiva de la ATKDP</td></tr>
<tr><td>Elaborar automáticamente las llaves (designación de contrincantes aleatoriamente) para dar inicio a un determinado torneo, campeonato o selectivo organizado por la ATKDP</td></tr>
</table>

Tabla 6.9: Prueba de Aceptación para las historia de usuario; Registro de puntaje obtenido en cada uno de los combates para un determinado torneo, campeonato o selectivo organizado por la ATKDP

HISTORIA DE USUARIO	OBSERVACIONES	RESULTADO
Registro de puntaje obtenido en cada uno de los combates para un determinado torneo, campeonato o selectivo organizado por la ATKDP.	***Se necesitan datos preliminares.*** • Se diseña una pantalla; Registro de Resultados • En la pantalla *Registro de Resultados* se registran los puntajes de cada participante hasta llegar al campeón de una determinada llave, cabe mencionar que el programa escoge automáticamente al deportista ganador, tomando en cuenta el puntaje obtenido con respecto a su contrincante; para esta pantalla se necesita como dato preliminar a la competencia, club, sexo, categoría, peso. • Estas pantallas tienen un grupo de botones de administración de datos como: refrescar, encerar, guardar; y una tabla donde se visualiza la llave generada.	**Aceptado por la Directiva de la ATKDP**
Registrar el puntaje obtenido en los combates de un determinado torneo, campeonato o selectivo organizado por la ATKDP		

Tabla 6.10: Prueba de Aceptación para las historia de usuario; Registro de ganadores en cada una de las categorías para un torneo, campeonato o selectivo organizado por la ATKDP

HISTORIA DE USUARIO	OBSERVACIONES	RESULTADO
Registro de ganadores en cada una de las categorías para un torneo, campeonato o selectivo organizado por la ATKDP.	***Se necesitan datos preliminares.*** • Se diseña una pantalla; Registro de Ganadores • En la pantalla Registro de Resultados se guardan automáticamente el campeón y subcampeón correspondiente a cada llave; a continuación en la pantalla *Registro de Ganadores* se registran los deportistas que obtuvieron el tercer puesto, además se permite hacer un cambio de campeón y subcampeón, esto tomando en cuenta alguna variación en los resultados finales; para esta pantalla se necesita como dato preliminar a la competencia, club, sexo, categoría, peso. • Esta pantalla tiene el botón guardar para administrar los datos; y una tabla donde se visualiza la llave generada.	**Aceptado por la Directiva de la ATKDP**
Registrar los ganadores de cada una de las categorías de un torneo, campeonato o selectivo organizado por la ATKDP		

Tabla 6.11: Prueba de Aceptación para las historia de usuario; Ingreso de Pago por concepto de carnetización o Inscripción de los deportistas en un torneo, campeonato o selectivo organizada por la ATKDP

HISTORIA DE USUARIO	OBSERVACIONES	RESULTADO
Ingreso de Pago por concepto de carnetización o Inscripción de los deportistas en un torneo, campeonato o selectivo organizada por la ATKDP	***Se necesitan datos preliminares.*** • Se diseña una pantalla; Ingreso por carnetización o inscripción de deportistas • En la pantalla *Ingreso por carnetización o inscripción de deportistas* se registra el ingreso de valores por carnetización o inscripción en una competencia de deportistas, se ingresan datos como fecha, responsable, tipo, club, valor y una observación, para esta pantalla se necesita como dato preliminar a los deportistas con su respectivo club.	**Aceptado por la Directiva de la ATKDP**
Ingresar pago por concepto de carnetización o inscripción de los deportistas en un torneo, campeonato o selectivo organizado por la ATKDP	• Esta pantalla tiene un grupo de botones de administración de datos como: nuevo, eliminar, buscar, guardar, borrar, registrar detalle y una tabla donde se visualizan los datos generales de inscripción o carnetización; se menciona además que para mejorar la visualización de la lista de deportistas se utiliza otra tabla y otro grupo de botones de manipulación de datos como: todos, limpiar, guardar. Ésta última tabla tiene una lista con checks de aceptación para los deportistas, cuando se selecciona el botón todos se activan todos los deportistas filtrados y cuando se selecciona el botón limpiar se desactivan todos los deportistas.	

Tabla 6.12: Prueba de Aceptación para las historia de usuario; Ingreso de Pagos por concepto de inscripción anual de clubes en la ATKDP

HISTORIA DE USUARIO	OBSERVACIONES	RESULTADO
Ingreso de Pagos por concepto de inscripción anual de clubes en la ATKDP	***Se necesitan datos preliminares.*** • Se diseña una pantalla; Ingreso por inscripción de clubes • En la pantalla *Ingreso por inscripción de clubes* se registra el ingreso de valores por inscripción anual de clubes, se ingresan datos como: fecha, responsable, tipo, club, valor y una observación, para esta pantalla se necesita como dato preliminar a los clubes. • Esta pantalla tiene un grupo de botones de administración de datos como: nuevo, eliminar, buscar, guardar, borrar y una tabla donde se visualizan los datos generales de inscripción ingresados.	**Aceptado por la Directiva de la ATKDP**
Ingresar pagos por concepto de inscripción anual de clubes en la ATKDP		

Tabla 6.13: Prueba de Aceptación para las historias de usuario; Ingreso y Egreso de valores por conceptos varios a la ATKD.

HISTORIA DE USUARIO	OBSERVACIONES	RESULTADO
Ingreso y Egreso de valores por conceptos varios a la ATKD	***No se necesitan datos preliminares.*** • Se diseña una pantalla; Ingresos / Egresos varios • En la pantalla *Ingresos / Egresos varios* se registra el ingreso o egreso de valores por distintos motivos como multas, donaciones, gastos de viáticos, entre otros, se ingresan datos como: fecha, responsable, tipo, valor y una observación. • Esta pantalla tiene un grupo de botones de administración de datos como: nuevo, eliminar, buscar, guardar, borrar y una tabla donde se visualizan los datos generales de inscripción ingresados	**Aceptado por la Directiva de la ATKDP**
Realizar el Ingreso y Egreso valores por conceptos varios a la ATKDP		

Tabla 6.14: Prueba de Aceptación para las historias de usuario; Consulta de reportes financieros y de competencia.

<table>
<tr><th>HISTORIA DE USUARIO</th><th>OBSERVACIONES</th><th>RESULTADO</th></tr>
<tr><td>Consulta de reportes financieros y de competencia.</td><td rowspan="2"><u>Se necesitan datos preliminares.</u>

• Se diseñan varias pantallas donde se despliegan los reportes financieros y de competencias.

• Se tienen reportes financieros como: total de ingresos y egresos por fechas, total movimientos realizados a la fecha.

• Se tienen reportes de competencias como: categorías y pesos, deportistas por competencia, ganadores por competencia, deportistas, clubes, deportistas.

• Se tienen filtros en los reportes donde sean necesarios.</td><td rowspan="2">Aceptado por la Directiva de la ATKDP</td></tr>
<tr><td>Consultar diferentes reportes tanto financieros como de competencia</td></tr>
</table>

6.2- Pruebas de Unidad

Para la evaluación del sistema mediante una prueba de unidad, se ha tomado en cuenta una función crítica, la misma que realiza el proceso “inserción de pago de deportista por inscripción en competencia”; para examinar esta función se realizan los siguientes pasos:

6.2.1- Código fuente de la función

```
protected void ImageButtonGuardarDeportistas_Click(object sender,
ImageClickEventArgs e)
{
   int i, maximo = DgridMovimiento.Items.Count;
   string aux_comprobante = TextBoxAux.Text;
   dsDeportistaDetalle=objClase_DeportistaDetalle.
Recuperar_MovimientoID(aux_comprobante);
   String id_mov = Convert.ToString(dsDeportistaDetalle.Tables[0].Rows[0][0]);
   DataSet dsDeportistaID = new DataSet();
   for (i = 0; i< maximo; i++)
   {
      CheckBox valorCH = (CheckBox)DgridMovimiento.
      Items[i].FindControl("ChkDeportista");
      if (valorCH.Checked == true)
      {
        string CI_deportista = this.
        DgridMovimiento.Items[i].Cells[1].Text;
        string nombre_deportista = this.
        DgridMovimiento.Items[i].Cells[2].Text;
        double valorUnitario = Convert.ToDouble(TextBoxValorAux.Text);
        dsDeportistaID = objClase_DeportistaDetalle.
        Recuperar_DeportistaID(CI_deportista,nombre_deportista);
        string id_deportista = Convert.ToString(dsDeportistaID.Tables[0].Rows[0][0]);
        registros_detalle = objClase_Ingresos.
        Registrar_DetalleMovimientoDeportista(id_mov,id_deportista,valorUnitario);
      }
 }
 Panel1.Visible = false;
 ImageButtonDetalle.Visible = false;
 LabelRegistrar.Visible = false;
 TextBoxFlag.Text = "0";
}
```

6.2.2- Diagrama de flujo de la función

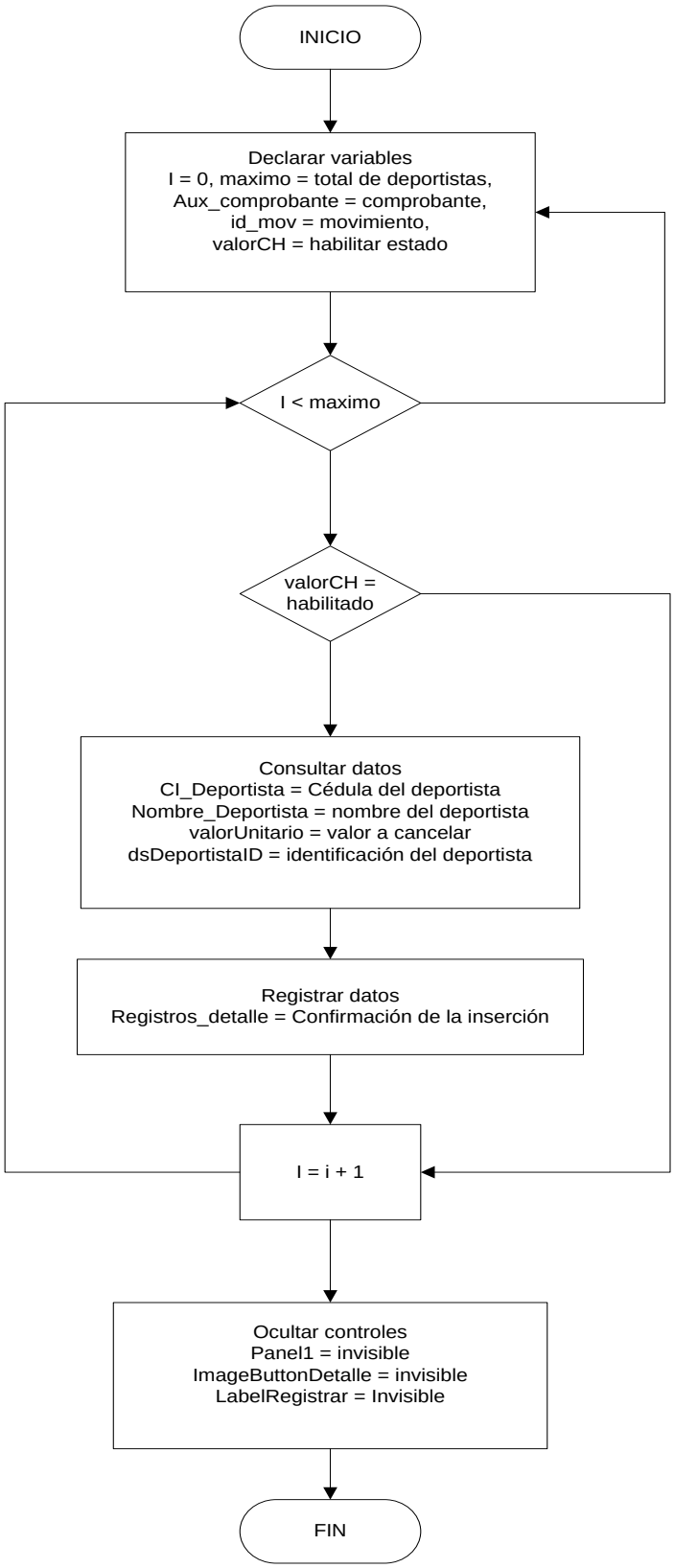


Figura 6.1: Flujograma Función “inserción de pago de deportista por inscripción en competencia”

6.2.3- Prueba de Escritorio

Para desarrollar la prueba de escritorio se tomarán en cuenta que se tiene registrada en la base de datos, 3 deportistas; dos de los cuales están en estado activo y uno deshabilitado. Se considera también que al ser una función de ingreso de detalle la que se va a comprobar, se conoce claramente los datos de la cabecera correspondiente entre estos datos se encuentran el numero de comprobante y el identificador del movimiento a realizar. Por lo tanto los datos que se disponen son los mostrados a continuación:

Tabla 6.15: Datos simulados para la prueba de escritorio

DATOS DE PRUEBA						
ID DEPORTISTA	CEDULA DEPORTISTA	NOMBRE DEPORTISTA	VALOR	HABILITADO	COMPROBANTE	MOVIMIENTO
1	12345	JUAN PEREZ	25	SI	IC1	MOV1
2	67890	PEDRO TAPIA	15	NO		
3	10293	JUANA PALACIOS	20	SI		

Con los datos disponibles se genera la prueba de acuerdo al flujograma de la figura 6.1; en la prueba se seguirá el orden de los procesos de acuerdo al flujo planteado en la función con el objetivo de segmentar la información disponible y generar un registro con los datos que realmente cumplen cada una de las condiciones mostradas en el flujograma:

Tabla 6.16: Prueba de escritorio

PRUEBA DE ESCRITORIO									
I	**Maximo**	**Aux_comprobante**	**Id_mov**	**Valorch**	**Id deportista**	**Cedula deportista**	**Nombre deportista**	**Valor**	**Registro_detalle**
0	3	IC1	MOV1						
1				1	1	12345	JUAN PEREZ	25	**TRUE**
2				0					
3				1	3	10293	JUANA PALACIOS	20	**TRUE**

De acuerdo a los datos arrojados por la prueba de escritorio realizada, se constata que; de los tres deportistas disponibles, solo dos cancelarán por concepto de inscripción a una competencia, y los datos correspondientes a estos dos únicos deportistas son los que se insertan en la base de datos.

Tabla 6.17: Datos finales obtenidos de la Prueba de escritorio

DATOS INSERTADOS EN BD - PROCESO: PAGO DE DEPORTISTAS POR INSCRIPCION DE COMPETENCIA					
ID DEPORTISTA	**CEDULA DEPORTISTA**	**NOMBRE DEPORTISTA**	**VALOR**	**COMPROBANTE**	**MOVIMIENTO**
1	12345	JUAN PEREZ	25	IC1	MOV1
3	10293	JUANA PALACIOS	20	IC1	MOV1

6.3- Prueba de Integración

Las pruebas de integración se utilizan para verificar que se cumplen todas necesidades del cliente en el sistema, se toma como base las historias usuario y

las pruebas de aceptación de usuario, además para comprobar que los módulos del sistema se retroalimenten de una manera correcta para el correcto funcionamiento del sistema. Por ejemplo:

- En el módulo financiero cuando se realiza un ingreso de pago por inscripción de deportistas en algún torneo, campeonato o selectivo; se filtra sólo los deportistas inscritos previamente en el módulo de competencias.

- En el módulo de competencias para inscribir deportistas en un torneo, campeonato o selectivo; se filtra sólo los deportistas que estén asignados a un club específico en el módulo de mantenimiento.

A continuación se presenta un listado con las actividades de integración que se deben cumplir para el sistema de administración para la ATKDP, con el objetivo de mostrar las validaciones e integraciones que el sistema debe cumplir para llevar un control total de integridad de los datos.

Tabla 6.18: Prueba de Integración

Descripción	Validación
El control de acceso al sistema se lo realiza en base a perfiles de usuario	SI
El usuario puede hacer un cambio de clave únicamente cuando esté activo	SI
Para la creación de clubes se deben ingresar datos obligatorios como nombre y dirección	SI
Para la creación de deportistas se deben ingresar datos obligatorios como cédula de identidad, nombre, apellido, dirección, fecha de nacimiento	SI
Para la creación de deportistas no se permiten el ingreso de dos o más deportistas con el mismo número de cédula, principalmente para evitar deportistas duplicados	SI
Para la creación de entrenadores se deben ingresar datos obligatorios como cédula de identidad, nombre, apellido, dirección	SI
Para la creación de entrenadores no se permiten el ingreso de dos o más deportistas con el mismo número de cédula, principalmente para evitar entrenadores duplicados	SI
Para la creación de categorías se debe ingresar el nombre como dato obligatorio	SI
No se permite el ingreso de dos categorías con el mismo nombre	SI
Para la creación de peso se debe ingresar el nombre y el peso inicial como datos obligatorios	SI
No se permite el ingreso de dos pesos con el mismo nombre que pertenezcan a la misma categoría y que sean del mismo sexo	SI
Para la creación de competencias se deben ingresar datos obligatorios como el nombre de la misma, la fecha de inicio de inscripciones y la fecha de finalización de inscripciones	SI
Para la creación de competencias se establece validaciones sobre el rango de fechas tales como: la fecha de inicio de competencia debe ser menor a la fecha fin de competencia, la fecha de inicio de inscripciones debe ser menor a la fecha fin de inscripciones, la fecha fin de inscripciones debe ser menor o igual a la fecha de inicio de competencia	SI
Para la asignación de un deportista en un club no se permite que un deportista esté asignado a dos o más clubes al mismo tiempo, para esto se utiliza el estado activo o inactivo	SI
Para la asignación de deportista a un club se deben ingresar el deportista, el club y la fecha de ingreso como datos obligatorios	SI
Cuando se inhabilita un deportista en un club determinado, se establece automáticamente la fecha de salida con la del sistema operativo	SI
Para la asignación de entrenador a un club se deben ingresar el entrenador, el club y la fecha de ingreso como datos obligatorios	SI
Cuando se inhabilita un entrenador en un club determinado, se establece automáticamente la fecha de salida con la del sistema operativo	SI
No se puede inscribir en un competencia si no está dentro del rango de fechas de inicio y fin de inscripciones	SI
Se crean llaves de competición tomando en cuenta la categoría, el peso y el sexo	SI
Las llaves se crean de manera aleatoria y se pueden generar tantas veces como se crea conveniente	SI
Para la generación de llaves sólo se toma en cuenta los deportistas inscritos	SI
Para el pago de un club por concepto de inscripción anual ingresar la fecha, el remitente y una observación como datos obligatorios	SI
Para el pago por concepto de inscripción anual de un club se crea un número de comprobante de manera automática que se usa como respaldo tanto de la ATKDP como para el club	SI
Para el pago de deportistas por concepto de carnetización o inscripción en competencia se debe ingresar la fecha, el remitente y una observación como datos obligatorios	SI
Para el pago de deportistas por concepto de carnetización o inscripción en competencia se crea un número de comprobante de manera automática que se usa como respaldo tanto de la ATKDP como para el club	SI
Para el ingreso o egreso de valores por conceptos varios se debe ingresar la fecha, el remitente y una observación como datos obligatorios	SI
Para el ingreso o egreso de valores por conceptos varios se crea un número de comprobante de manera automática que se usa como respaldo tanto de la ATKDP como para el club	SI
Cuando un deportista ha participado en más de 3 competencias o ha estado inscrito por más de un año en la ATKDP pasa automáticamente de tipo novato a clasificado	SI

CAPÍTULO VII

CONCLUSIONES Y RECOMENDACIONES

Una vez finalizado el Sistema de Administración para la ATKDP, se han observado varias experiencias, las mismas que permiten generar varias conclusiones y recomendaciones sobre el proyecto realizado.

7.1- Conclusiones

- Es importante levantar los requerimientos funcionales y no funcionales, y que queden validados por los diferentes usuarios para posteriormente definir la metodología a usar al igual que la tecnología.

- Con el desarrollo de este sistema, los procesos que se llevan a cabo en la ATKDP se los realiza de manera más ágil y óptima; ya que debido a la automatización, los datos se presentan de una manera más rápida y confiable, con el fin de ofrecer un mejor servicio a los clubes administrados por la ATKDP así como al público en general.

- Con la ayuda del sistema, se evita confusiones por parte de clubes, entrenadores y deportistas con respecto a transferencia de deportistas, información en competencias, entre otras; ya que la información se encuentra albergada en una base de datos de la cual se pueden extraer datos correctos sobre la administración de la ATKDP en tiempo real.

- La utilización de la metodología XP en el desarrollo del sistema de administración para la ATKDP, permite que se tenga un contacto continuo con el cliente, se realice una toma de requerimientos de forma clara y

sencilla para automatizar los procesos de una manera correcta; con esto las pruebas realizadas al sistema en su funcionalidad e integración de módulos genera resultados positivos.

- La programación orientada a objetos y por capas utilizada en el sistema de administración para a ATKDP permite estructurar la aplicación por niveles lo cual es beneficioso; ya que, si se quiere realizar algún cambio se lo realiza solamente en el nivel requerido, con ello no se mezcla código innecesariamente.

- Considerando que las personas que van operar el sistema, incluyendo a autoridades, entrenadores, deportistas, entre otros; no se encuentran trabajando dentro de la ATKDP; ya que, es una asociación deportiva sin fines de lucro; el sistema es de gran utilidad, puesto que permite realizar transacciones vía web para realizar los diferentes procesos dentro de misma desde cualquier sitio.

7.2- Recomendaciones

- De acuerdo a la metodología XP, los programadores de la aplicación se deben intercambiar módulos y funciones con el fin de que todos conozcan el código y puedan reutilizar funciones y métodos, por lo tanto se recomienda que la codificación del sistema siga un estándar establecido por el equipo de desarrollo.

- Para el desarrollo de sistemas pequeños y medianos que requieran cortos tiempos de desarrollo, se recomienda el uso de una metodología ágil como es el caso de XP, ya que se tiene agilidad en la documentación y en el desarrollo, permanente contacto con el cliente y se obtiene un producto de calidad.

- Se recomienda usar programación orientada a objetos y por capas para el desarrollo de cualquier sistema informático, de esta manera el sistema puede afrontar con mayor facilidad cambios en el futuro y mantenimiento; teniendo en cuenta que siempre se debe utilizar una metodología de desarrollo que es la base de un sistema informático

- Se recomienda de manera especial que se impartan conocimientos de metodologías ágiles en la carrera de ingeniería en sistemas e informática de la ESPE, para que los estudiantes las tengan en cuenta para el desarrollo de sistemas informáticos con características como permanente contacto con el cliente que incluso se convierte en parte del equipo de desarrollo, documentación concreta del sistema, menores tiempos de entrega con resultados ampliamente satisfactorios, entre otros.

- En la Asociación de Tae Kwon Do de Pichincha existen procesos que no está dentro del alcance de nuestra tesis de grado, pero que pueden ser de mucho interés para otros estudiantes que necesiten un tema como proyecto de tesis, no sólo en esta asociación sino muchas otras que estén ligadas con el deporte ecuatoriano.

- Se puede considerar: desarrollar otros reportes no estimados actualmente, envío de información obtenida desde el sistema vía correo electrónico tanto a directivos de la asociación como a personal autorizado de los diferentes clubes inscritos, automatización de procesos no considerados en el sistema actual y un control de fotografías e imágenes indexadas en los registros de deportistas, entrenadores y clubes.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS