UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DISEÑO DE UN SISTEMA COMPUTACIONAL PARA EL SEGUIMIENTO, CONTROL Y MANEJO DE INFORMACIÓN EN LAS ORGANIZACIONES DEPORTIVAS

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

P R E S E N T A :

IGNACIO BELTRÁN BOTELLO

DIRECTOR DE TESIS:

ING. SEBASTIÁN POBLANO ORDÓÑEZ

CIUDAD UNIVERSITARIA **2005**

AGRADECIMIENTOS.

A la Universidad Nacional Autónoma de México, y en especial a la Facultad de Ingeniería por brindarme la capacitación necesaria para desarrollarme y realizarme como profesionista.

Al M. En C. Gerardo Ferrando Bravo, Director de la Facultad de Ingeniería por su apoyo cada vez que se lo he solicitado.

Al Dr. Salvador Landeros Ayala por proporcionarme los recursos necesarios para sustentar este proyecto.

Al Ing. Sebastián Poblano Ordóñez por brindarme su valiosa asesoría, experiencia, por su apoyo y ánimo para concluir este proyecto.

Al Arq. Carlos Guzmán Pérez y al Ing. José Arturo Landeros Ayala por otorgarme todos los medios existentes para respaldar mi proyecto de tesis.

Al Ing. Juan Manuel Martínez Villalobos por motivarme a terminar lo que muchas veces había empezado y abandonado.

A la Institución Educativa-Deportiva Pumitas C.U. Fútbol, A.C. por las facilidades otorgadas en la elaboración de la presente tesis.

A mis sinodales por dedicarme su valioso tiempo, apoyo y atención.

Muy pero muy en especial, a mi princesa Boudek Morrigan, por ser mi razón, mi pensamiento, mi motivación para ser mejor, un apoyo constante, te amo.

ÍNDICE

TEMARIO **Página**

INTRODUCCIÓN.

El amplio y variado espectro de la actividad físico-deportiva está compuesto de tantos deseos, motivos e intereses como practicantes. De ahí que haya que seleccionar contenidos y segmentar el mercado.

En las actividades de conjunto, el servicio y la organización deportiva son factores fundamentales para lograr la satisfacción del cliente.

Así, para que Pumitas C.U. Fútbol, A.C., caso específico de las organizaciones deportivas, pueda cumplir con la importante tarea de crear un desarrollo integral de los niños que la forman, se debe conocer los recursos humanos con los que cuenta, para así poder administrarlos racionalmente y lograr un óptimo rendimiento.

Por esto resulta imperativo registrar, controlar, predecir y decidir sobre los movimientos a realizarse en las distintas categorías y áreas (técnica, médica y administrativa) que componen a la organización, a fin de propiciar planificaciones de temporadas futuras con una óptima toma de decisiones.

La presentación de esta tesis está dividida en diferentes partes, las cuales permiten llevar a cabo un control efectivo del sistema durante toda su etapa de diseño e implementación, cada una de éstas se desarrolla en los diferentes capítulos que a continuación se detallan.

En el capítulo uno se da un panorama general sobre los antecedentes o conocimientos previos que se deben tener en cuenta para resolver de una manera estructurada la problemática de Pumitas, así como los objetivos que se persiguen para desarrollar el sistema.

Dentro del capítulo dos se desarrollan los fundamentos teóricos en materia de Ingeniería de Software, analizando el uso de técnicas especializadas para la realización de sistemas, además se realiza una explicación de la técnica utilizada para la solución de nuestra problemática; por último se revisan conceptos fundamentales de bases de datos.

Dentro del capítulo tres se presenta la fase de Análisis del Sistema, es la encargada de estudiar a partir de la problemática, organización y el medio ambiente existente en Pumitas, los objetivos y los requerimientos necesarios para su solución.

Se proponen o evalúan diversas alternativas de solución identificando sus entradas y salidas, para finalmente elaborar diagramas, antes de codificarse en un lenguaje de alto nivel.

Dentro del cuarto capítulo se realiza la fase correspondiente al Diseño del Sistema, empezando por realizar el diseño de la base de datos, el diagrama entidad-relación y el diccionario de datos, finalizando con la propuesta y selección de las herramientas de desarrollo y el lenguaje de programación a utilizar

En el capítulo cinco, se lleva a cabo la fase de implementación del sistema donde se especifica la codificación realizada para el sistema Pumidata 1.0 para realizar el desarrollo de la base de datos y la interfaz gráfica.

También, se presenta la fase de pruebas del sistema, donde se especifican los tipos de prueba que existen y las técnicas para la preparación de éstas con el objetivo de aplicarlas al problema real. Asimismo, se encuentra la actividad de mantenimiento, que consiste en poder satisfacer nuevos requerimientos de los usuarios, mediante posibles cambios en la programación ya existente.

Para finalizar se presentan las conclusiones de la presente tesis.

Adicionalmente, se presentan dos apéndices, en el A se presenta el manual de instalación del programa y en el B el manual de usuario, el cual proporciona las indicaciones necesarias para la operación del sistema.

CAPÍTULO 1. Antecedentes.

Actualmente, el manejo de la información es parte fundamental de cualquier organización deportiva, sin importar si persiga un fin de lucro o no, para la realización de sus actividades en forma rápida y eficiente. Con los adelantos tecnológicos en el área computacional, área de comunicaciones y tecnologías de información las organizaciones deportivas, han dado suma importancia al uso de sistemas de información computacionales, aprovechando los beneficios que éstos les otorgan en el procesamiento de la información en forma rápida y confiable, en la ayuda de toma de decisiones a gerentes y ejecutivos o como sistemas expertos en la resolución de problemas de alto grado de especialización en el área, etc. Las organizaciones conociendo las ventajas y la necesidad de uso de éstos han considerado a los sistemas como parte dinámica de su estructura.

La esencia organizativa y la metodología operativa de las distintas organizaciones deportivas es la misma en todos los casos, por lo cual la presente tesis, se aplicará a una de las organizaciones deportivas que se dirige al sector infantil y juvenil dentro de la Universidad Nacional Autónoma de México, la Organización Pumitas UNAM, dependiente del Deporte Formativo y Recreación, de la Dirección General de Actividades Deportivas y Recreativas (DGADyR), y más específicamente a la Organización Pumitas C.U. Fútbol A.C. (desde ahora Pumitas).

Pumitas sólo cuenta con un limitado equipo de cómputo para labores administrativas, y sólo algunos de los integrantes de ésta tienen acceso a dicho equipo, lo cual hace que la información sólo sea accesada por algunos “privilegiados”, quienes realizan labores administrativas auxiliándose de los actuales medios informáticos mediante formatos creados principalmente en Hojas de Cálculo (Excel) y bajo un esquema que ignora la información generada en años anteriores, pues no se registra la información resultante de las administraciones pasadas, y mucho menos se interpretan los datos que puede arrojar, lo cual hace que cada año prácticamente se empiece de cero la formación de los jóvenes deportistas, además, esto impide que se cumpla un proceso ideal de formación y preparación.

En la actualidad existen programas que se venden por Internet pero que su rigidez no permite personalizar las necesidades de Pumitas, por lo cual resultaría en un gasto innecesario.

Por lo anterior se hace necesaria la intervención de sistemas computacionales para Pumitas, ya que periódicamente se procesa una inmensa masa de información que no es posible interpretar y analizar correcta y oportunamente, pues al hacerlo de forma manual resulta muy tardado y eso evita sacarle provecho oportuno a la posibilidad de procurarles a los integrantes de Pumitas un tratamiento sistemático, efectivo y al día de su educación deportiva.

PUMITAS C.U. FÚTBOL, A.C.

OBJETIVOS.

Pumitas surge en 1976 con el objetivo primordial de coadyuvar en la formación integral de los niños, hijos del personal académico y universitarios egresados, primordialmente, en un ambiente integrador que fomenta los valores de nuestra Universidad Nacional Autónoma de México, mediante la práctica del Fútbol Asociación, motivándolos a ser mejores estudiantes, a amar y respetar a nuestra Universidad. Se pretende desarrollar en los niños el deseo de superación personal, siendo el fútbol asociación la principal actividad deportiva, considerándolo sólo un medio para lograr el desarrollo integral de la personalidad de los niños y fomentar la unión familiar, infundiéndoles los hábitos esenciales de disciplina, respeto, orden, dedicación a la escuela y buena conducta en todas sus actividades, además de desarrollar también sus habilidades motoras, afectivas, cognoscitivas y sociales.

Pumitas es una asociación de servicio social, cuyo objetivo fundamental es el fomento y difusión de sentimientos y actitudes de respeto y cariño hacia la Universidad Nacional Autónoma de México y la afirmación de los valores sociales y humanos dentro de un ambiente universitario que ella sustenta, así como lograr el desarrollo integral del niño.

El lema de la Asociación es, “Convivir más que competir”, enfatizándose que Pumitas no es una escuela de fútbol. A pesar de no ser considerada una escuela de fútbol, existe la preocupación permanente de tener capacitado a su personal para influir positivamente en el desarrollo físico, espiritual, emocional y afectivo de los niños.

El entorno de Pumitas se encuentra determinado por la familia, la escuela, el deporte, el arte y la cultura.

Son miembros de Pumitas los padres de familia cuyos hijos hayan sido aceptados en esta organización, ya sean autoridades, investigadores, profesores, técnicos, empleados, alumnos de la UNAM y aquellas personas que ostenten un cargo en la Mesa Directiva, los cuales entre sí están especialmente obligados a guardarse el mayor respeto y consideración, fomentando por los medios a su alcance, el mutuo compañerismo derivado de los ideales comunes y del cariño que profesan a nuestra Alma Mater.

VISIÓN.

Pumitas se proyecta como una Organización deportiva–educativa de prestigio nacional e internacional. Garantiza a sus egresados una filosofía de vida congruente con los valores de la Universidad por su

formación como individuos sanos, disciplinados, colaboradores y con los fundamentos básicos de la práctica del fútbol.

ESTRUCTURA.

- Mesa Directiva.
- Coordinadores técnicos.
- Coordinadores administrativos.
- Personal administrativo y de mantenimiento.
- Monitores.
- Delegados.
- Padres de familia y familiares.
- Niños y niñas.
- Área Médica.
- Árbitros.

ORGANIZACIÓN TÉCNICA DE PUMITAS.

- COORDINADOR TÉCNICO GENERAL.
- COORDINADOR TÉCNICO GENERAL ADMINISTRATIVO.
- COORDINADORES TÉCNICOS DE CATEGORÍAS MAYORES Y MENORES.
- COORDINADOR TÉCNICO DE ACONDICIONAMIENTO FÍSICO.
- COORDINADOR TÉCNICO DE CATEGORÍA.
- MONITOR.

1.1 Definición del problema.

Al parecer, todo está íntegramente planificado para que Pumitas funcione con unos altos estándares de calidad y eficiencia, que le daría la mejor reputación en cuanto a organizaciones de fútbol asociación infantil se refiere, pero en la práctica no es así, está desperdiciado gran parte del potencial que tiene.

Los Coordinadores Técnicos se encuentran frecuentemente más preocupados por cuestiones administrativas y organizativas, que en supervisar y procurar el desarrollo técnico y la asimilación de conocimientos de los niños y niñas que integran Pumitas, lo cual hace factible la realización de un sistema computacional que represente una solución para minimizar el tiempo que se le dedica a tales

funciones y dejarles más espacio a coordinar el trabajo en cancha, el cual rinde más frutos al aprovechar su experiencia.

Debido a la necesidad de llevar un buen seguimiento, control y manejo de información de Pumitas, se vuelve de suma importancia la implementación de un sistema que permita minimizar los tiempos de espera y proporcione una forma eficaz de acceso a la información, la cual permita una toma de decisiones acertada y con total conocimiento de las necesidades de la Organización.

PROBLEMÁTICA ACTUAL.

- **Inscripción de los niños.** Existe una falta de comunicación entre las distintas áreas organizativas de Pumitas, los coordinadores administrativos frecuentemente no concilian información con los coordinadores técnicos y viceversa, lo cual hace que las necesidades de las distintas categorías no se cubran con prontitud, y que se enfrenten problemas como el sobrecupo de unos equipos o falta de niños en otros, cuando afortunadamente se cuenta con una gran demanda de lugares para ingresar a Pumitas.

- **Categorías.** No se tienen catálogos de ejercicios para las distintas categorías y la información no se comparte entre las mismas.

    - No se tienen parámetros planeados indispensables para tomar en consideración si se alcanzan los objetivos pretendidos por categoría, y que hace que muchos niños o niñas pasen a categorías que no les corresponden o permanezcan en categoría que ya no cumplen sus expectativas ni permiten su desarrollo, y lo que provocan es un estancamiento en su desarrollo.

- **Instalaciones.** Hay instalaciones que casi no se ocupan y son desperdiciadas, como el auditorio, donde se pueden proyectar videos, usarlo como recinto para reuniones, etc.

- **Coordinadores técnicos de mayores y menores.**

    - **Seguimiento.** Nunca se ha dado un seguimiento adecuado de las actividades realizadas por los coordinadores técnicos, el trabajo a veces muy bien hecho por parte de unos es desperdiciado por otros, y el proceso de desarrollo deportivo se trunca cada temporada, donde se vuelve a empezar de cero y cuando vuelven a retomar el control de la situación resulta que ya ha pasado la mitad de la temporada, cuando ya se ha desperdiciado demasiado tiempo.

- **Planificación.** Los cronogramas de actividades no se tienen a tiempo y por consiguiente el trabajo llevado a cabo por los monitores es arbitrario y no sigue un proceso, no se cuentan con planes anuales de trabajo, ni se implementa metodología del deporte, aunque cada año, se refuercen estos conocimientos en teoría en los cursos de capacitación.

- **Evaluaciones a jugadores.** No se programan adecuadamente, ni se proporciona la información respectiva para llevarlas a cabo, incluso, muchas veces ni se realizan.

- **Evaluaciones del desempeño de los coordinadores.** No se cuenta con ningún esquema que permita evaluar su desempeño y con base en esto buscar una mejora continua.

- **Distribución de funciones.** No todos los coordinadores trabajan al parejo y este desequilibrio provoca una mala impresión ante los padres de familia, monitores, niños y niñas.

- **Coordinadores técnicos de categoría.**

  - **Trabajo en bloque.** Las sesiones de trabajo en bloque no cumplen la calidad deseada; asimismo el trabajo de los talleres de gimnasia y de técnica individual no tiene un seguimiento adecuado.

  - **Supervisión.** Los coordinadores no supervisan adecuadamente el trabajo de los monitores, esto trae como consecuencia tener pocos argumentos para evaluar su desempeño. No revisan ni registran las unidades de entrenamiento, listas de asistencia ni los reportes mensuales (mucho menos los anuales), por lo cual no existe un seguimiento del trabajo de los monitores y no se puede procesar esa información para interpretarla, por ende, no proporcionan retroalimentación para corregir errores y por lo tanto, no se mejora como cuadro de trabajo.

  - **Premios.** La asignación de estímulos económicos, resultado del buen desempeño de los monitores por su labor mensual muchas veces son mal designados o entregados a monitores que no lo merecen y los parámetros considerados para la entrega del estímulo no se hace pública, propiciando el enojo y descontento de los mismos, lo cual genera un mal ambiente de trabajo.

- **Talleres.** El taller de acondicionamiento físico ha sido un lugar para infinidad de niños que tienen muy buen desempeño físico y que lo único que quieren es aprovechar más días de entrenamiento y superarse, pero no cumple con su finalidad principal que es acondicionar físicamente a los menos capacitados, y en consecuencia se encuentra desaprovechado tal trabajo, lo mismo sucede con el taller de psicomotricidad.

- **Monitores.**

    - **Capacitación.** La "capacitación" que se proporciona a los monitores al inicio de cada temporada en el curso no logra los fines convenidos, pues también ahí se presenta una gran falta de interés por parte de los monitores (inasistencias). Se requiere una mejor capacitación de los monitores para que sepan lo que enseñan, que muestren interés por su equipo y por cada uno de los integrantes del mismo.

    - Los monitores no entregan a tiempo su documentación, de esto se deduce que el entrenamiento no sea planificado.

    - **Organización.** Se siguen viendo filas al momento de ejecutar los ejercicios por parte de los niños, lo cual representa tiempos perdidos, tiempos que no permiten que la carga de trabajo que se debe llevar a cabo pueda tener un efecto benéfico en el desarrollo de los niños.

- **Niños y niñas.**

    - **Asistencia.** Existe un gran número de familias que no cumplen con la asistencia mínima que por reglamento, les permite permanecer dentro de Pumitas, y que desgraciadamente hacen que el trabajo "planificado" tenga que ser modificado constantemente y hace que empiecen a perder interés los monitores en su desarrollo como educadores, pues la inasistencia de los jugadores de los equipos causa incremento de apatía por buscar la mejora continua.

- **Arbitraje.** El arbitraje dentro de Pumitas deja mucho que desear y no se encuentran bajo capacitación constante ni bajo criterios mínimos de capacidad teórica y física para soportar la carga de trabajo actual.

- **Continuidad.** La falta de información que al término de cada temporada se presenta en las

distintas categorías hace que el proceso de creación de nuevos equipos se convierta en un proceso aleatorio, y provoca equipos con supremacía física, futbolística, de asistencia, etc., creando unos desequilibrios competitivos a veces enormes.

- **Área Médica:** No se tiene un seguimiento adecuado de las lesiones en que incurren los niños, lo que ocasiona recaídas más severas de las mismas.

Al hacer esta revisión de toda la problemática existente, se determina que lo más urgente será realizar el correcto seguimiento de los niños, para lo cual se buscarán los medios necesarios para solucionarlo.

1.2 Objetivos.

Crear un sistema que facilite el control de la información que se genera diariamente para darle el seguimiento correcto a los niños y permitirles el rendimiento óptimo al tener mejor conocimiento y control de todos los factores que le rodean y que lo afectan tanto de manera negativa como positiva.

Cubrir la necesidad de respaldo, edición y consulta de información de Pumitas.

Unificar criterios entre las distintas categorías de Pumitas para que sea el posible el proceso idóneo enseñanza-aprendizaje.

Facilitar una evaluación más eficaz siguiendo los principios de rigor, objetividad y continuidad.

Simplificar el proceso de adquisición de datos, proceso e interpretación de los mismos para una óptima toma de decisiones.

Tener actualizado el historial de los coordinadores generales, coordinadores técnicos, entrenadores, equipos y jugadores con el afán de tener un mayor control en el proceso de formación deportiva de los niños, así como el historial de los conceptos que permitirá que los coordinadores generales, técnicos, y monitores de la organización operen funcionalmente con total conocimiento y permita tener un proceso óptimo en la enseñanza de la actividad deportiva de todos los niños integrantes de la organización.

Elevar el nivel de rendimiento atlético para optimizar su proceso y rendimiento deportivo.

Facilitar y tener actualizada la información correspondiente para generar una mejor distribución de los recursos humanos y materiales en beneficio del proceso enseñanza-aprendizaje buscado.

Evitar malas organizaciones y la toma de malas decisiones por falta de conocimiento o información al momento de distribuir los recursos materiales y humanos de la organización.

CAPÍTULO 2. Fundamentos Teóricos.

2.1 Ingeniería de software.

Software es la suma total de los programas de computadora, procedimientos, reglas, la documentación asociada y los datos que pertenecen a un sistema de cómputo. Un producto de software es un producto diseñado para un usuario.

La Ingeniería de Software es un enfoque sistemático del desarrollo, operación, mantenimiento y retiro del software. Es la rama de la Ingeniería que aplica los principios de la ciencia de la computación y las matemáticas para lograr soluciones costo-efectivas a los problemas de desarrollo de software y permite elaborar consistentemente productos correctos y utilizables.

El proceso de ingeniería de software es un conjunto de etapas parcialmente ordenadas con la intención de lograr un objetivo, en este caso, la obtención de un producto de software de calidad. Traduce las necesidades del usuario en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código, el código será probado, documentado y certificado para su uso operativo. Requiere por un lado un conjunto de conceptos, una metodología y un lenguaje propio. A este proceso también se le llama el ciclo de vida del software que comprende cuatro grandes fases: concepción, elaboración, construcción y transición. La concepción define el alcance del proyecto. La elaboración define un plan del proyecto, especifica las características y fundamenta la arquitectura. La construcción crea el producto y la transición transfiere el producto a los usuarios.

2.1.1 Metodologías de desarrollo de sistemas.

Las metodologías proponen seguir un proceso para que el diseño de software sea flexible, de bajo costo y confiable, es decir, indican cómo construir técnicamente el software. A estas metodologías se les conoce como paradigmas y nos permiten situar el proceso de desarrollo de un sistema en diferentes etapas.

La elección de un paradigma se hace de acuerdo con la naturaleza del proyecto y de la aplicación, los métodos y herramientas a usar.

Ciclo de vida clásico.

También conocido como modelo en cascada, exige un enfoque sistemático y secuencial; cada una de sus partes se relaciona entre sí para lograr un software eficiente y en consecuencia coherente. Tal y como se muestra en la figura.

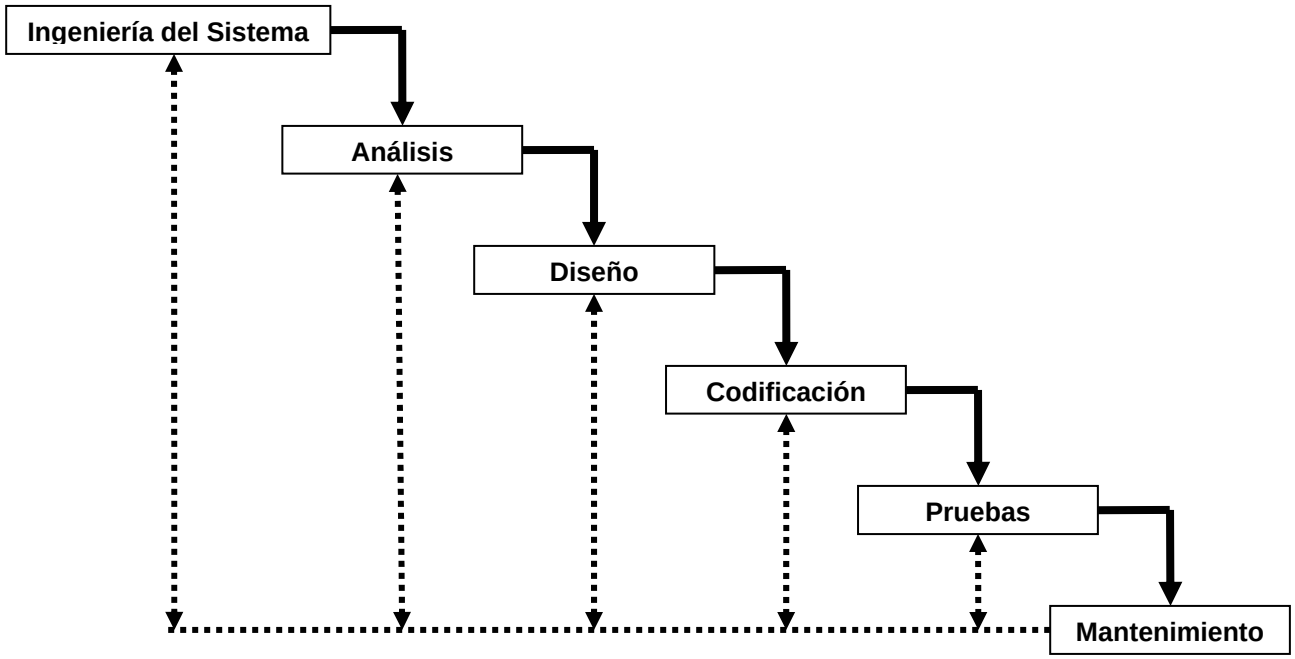


Figura 2.7 Ciclo de vida.

Las fases del modelo de cascada.

- Ingeniería y análisis del sistema. Se comienza estableciendo los requisitos de todos los elementos del sistema, entre ellos se establece los requisitos del software. En esta fase se revisan los requerimientos del sistema. Este planteamiento es necesario porque el software tiene que interactuar con el hardware, con personas y con la base de datos.
- Análisis de los requisitos del software. En esta etapa se intensifica la recopilación de los requisitos especialmente para el software. La comprensión del ámbito de la información, así como la función, el rendimiento y las interfases requeridas se revisan en compañía del cliente.
- Diseño. Es aquí donde se traducen los requerimientos en una representación del software considerando la estructura de los datos, la arquitectura del software, el detalle procedimental y la caracterización de la interfaz, todo esto previo a la codificación para obtener la calidad requerida.
- Codificación.

- Implementación y pruebas. Una vez generado el código, la prueba se centra en la lógica interna del software, asegurando que todas las sentencias se han probado, y en las funciones externas verificando que la entrada definida produce los resultados requeridos.
- Mantenimiento. Con certeza, el software sufrirá cambios después de que se entregue al cliente. Estos cambios pueden atribuirse a fallas encontradas durante la ejecución del software, a problemas de comunicación con otras aplicaciones, a actualizaciones del sistema operativo, actualizaciones de hardware; o bien, a que el cliente requiera ampliación de funciones o del rendimiento. El mantenimiento del software aplica cada uno de los pasos precedentes del ciclo de vida a un programa existente en vez de a uno nuevo.

Ventajas:

- Se tiene un seguimiento secuencial del proyecto, el cual permite tener un orden durante el desarrollo del software.
- Evita un crecimiento incontrolable, debido a que no se pueden añadir requerimientos una vez finalizada la etapa de análisis.

Desventajas:

- El desarrollo de un proyecto raramente sigue la secuencia que el modelo propone.
- Siempre ocurre la iteración y crea problemas en la aplicación de esta metodología.
- El método de ciclo de vida clásico necesita una definición clara de los requerimientos, la cual generalmente el cliente no expone de manera precisa. Esto último genera cierta incertidumbre en cuanto a lo que se espera del sistema y hace difícil el inicio del proyecto.
- Una versión desarrollada del programa no estará disponible hasta mucho tiempo después de haberse iniciado el proyecto, por lo que el cliente debe esperar.
- Una falla o error detectado al final del desarrollo y no durante el mismo, puede ser desastroso.
- Dificultad de permitir cambio después de que el proceso haya iniciado.
- División inflexible del proyecto en fases distintas.
- Esto hace difícil responder a los requerimientos cambiantes del cliente.
- Apropiado sólo cuando los requerimientos son bien comprendidos.

Construcción de prototipos.

Esta metodología presenta un diseño rápido sobre la representación de los aspectos del software visibles al usuario, el modelo que se crea puede diseñarse de la siguiente forma:

- Un prototipo en papel o modelo basado en computadora que describa la interacción hombre-máquina.
- Un prototipo que implemente algunos subconjuntos de funciones requeridas en el programa.
- Un programa existente que ejecute parte o toda la función deseada, pero que tenga características que deban ser mejoradas en el programa final que se dará al cliente.

Comienza con la recolección de requerimientos, luego se produce un diseño rápido, el cual muestra al usuario los métodos de entrada, los formatos de salida y las diversas funciones del programa, según los requerimientos. A partir de este diseño se construye el prototipo, el cual es evaluado por el cliente, y con base en esta evaluación se mejora el software; de esta manera, se genera un proceso interactivo en el que el prototipo es mejorado una y otra vez hasta que satisfaga las necesidades del cliente.

Ventajas:

- El cliente puede ver a corto plazo una primera versión quizá muy elemental de su sistema, pero ésta le puede ayudar para comenzar a familiarizarse con la computadora.
- Al tener un prototipo del proyecto de software, se pueden detectar con mayor facilidad las características faltantes, proponerse nuevas ideas, mejorar el diseño y en general ampliar la perspectiva del proyecto que se quiera desarrollar.

Desventajas:

- El cliente ve una versión del sistema en la cual no se consideran los aspectos de calidad ni mantenimiento de software a largo plazo.
- Cuando se le informa que el producto debe ser reconstruido, el cliente cree que presenta fallas porque no comprende que haciendo pequeñas modificaciones al prototipo, permitirá que se ajuste a las nuevas especificaciones para ser puesto en ejecución.

Modelo en espiral.

Tiene como objetivo aprovechar las mejores características tanto del modelo del ciclo de vida clásico como las de creación de prototipos añadiendo una etapa, el análisis de riesgo.

Etapas:

- Planificación: Se determinan los objetivos, alternativas y restricciones. Esta actividad involucra la recolección de requerimientos, donde el cliente debe exponer sus necesidades y dar una idea

general de lo que espera; una vez recopilada la información se procede a realizar el estudio inicial para el diseño del software.

- Análisis de riesgo: Aquí se hace un análisis de alternativas, se identifican y resuelven los posibles riesgos existentes. Esta actividad permite realizar un análisis del diseño que se ha definido para el desarrollo del software; así, se pueden detectar los inconvenientes y limitaciones que se tuvieran antes de comenzar la programación del sistema. También esta etapa permite predecir las expectativas a futuro del proyecto, ya que se pueden visualizar anticipadamente las fallas que pudieran ocurrir por actualizaciones en el hardware y software, lo cual permite proponer soluciones a los problemas que muy probablemente se presentarán en el futuro.
- Ingeniería: Se desarrolla lo que será el producto de “siguiente nivel”. En este caso, la espiral va girando y creciendo hasta que obtengamos el producto final, un software de calidad. Si comenzamos la primera actividad del modelo en el centro de la espiral y se va avanzando hacia fuera, el desarrollo del producto de siguiente nivel se refiere a la creación de un prototipo del sistema final, el cual ya consideró todas las características en su totalidad al cubrir la primera curva de la espiral.
- Evaluación del cliente: Una vez tenido el prototipo de la primera vuelta se hace una evaluación de los resultados de la ingeniería. Esta actividad sugiere ir revisando con el cliente tanto el diseño como el programa que se están empleando para la realización de sus sistema, esto con la finalidad de que si el desarrollo va por buen camino se continúe en la misma línea, de lo contrario, se debe analizar el proceso y corregir las fallas encontradas.

En este modelo, las actividades se realizan conforme se va desenvolviendo la espiral; debe considerarse el tiempo de desarrollo del software puesto que la espiral puede crecer desordenadamente, y lo que el modelo pretende es tener un sistema más completo y satisfactorio para el cliente, conforme se avanza en la espiral y no un desarrollo que crezca indefinidamente.

Este esquema ayuda a que en cada vuelta de la espiral, se construyan sucesivas versiones del software, cada vez más depuradas y completas, hasta llegar a obtener el producto final.

Ventajas:

- Se desarrolla un sistema que, por las diferentes etapas en las que estuvo envuelto, promete ser el óptimo y adecuado para las necesidades del cliente.
- Al tener una actividad de análisis de riesgo, el producto final garantiza un acoplamiento a los cambios del software o de hardware que pudieran presentarse en el futuro; es decir, la etapa de mantenimiento llegaría automáticamente sin necesidad de realizar un largo análisis para evaluar las alternativas de solución.

Desventajas:

- Es difícil convencer al cliente de que el enfoque evolutivo es controlable.
- Se requiere de cierta habilidad para la valoración de los riesgos.
- Si no se delimita bien el número de iteraciones, la espiral crecerá de manera desordenada y sin saber en qué momento se detendrá.

2.2 Bases de datos.

Antes de que existieran las Bases de Datos existieron los Sistemas de procesamiento de archivos, basados en registros guardados en archivos, escribiéndose varios programas para su manipulación (obtener y guardar información). Estos tenían varias desventajas:

- Redundancia de información (repetición de información).
- Inconsistencia en la información (problemático actualizarla, por lo tanto las distintas copias no concuerdan).
- Falta de manejo de concurrencia (usuarios múltiples con actualizaciones al mismo tiempo).
- Generalmente va ligado programa con archivo.

2.2.1 Conceptos.

Base de datos:

- Conjunto de información organizada (Archivo lógico (uno o varios archivos físicos)).
- Conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquina accesibles en tiempo real y compatibles con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo.

Las bases de datos proporcionan la infraestructura requerida para los sistemas de apoyo y para los sistemas de información estratégicos, ya que estos sistemas explotan la información contenida en las bases de datos de la organización para apoyar el proceso de toma de decisiones o para lograr ventajas competitivas.

Finalidad de una Base de datos: Servir a una aplicación o más de la mejor manera posible. Existen varios niveles en que puede observarse la base de datos. Esto se conoce como Abstracción de la Información.

NIVEL FÍSICO: Describe cómo se almacenan los datos (bytes). Persona que administra la Base de datos.
NIVEL CONCEPTUAL: Esquematización de los datos y sus relaciones (diagramas). Analista y Desarrollador .
NIVEL DE VISIÓN: Abstracción más alta, cómo interpretan la información los usuarios finales.
Un **Sistema de Base de datos** es un conjunto de:

- **Datos**: Valores registrados físicamente en la base de datos. La información es el significado de los datos.
- **Hardware**: Dispositivos de almacenamiento físico en donde reside la base de datos, así como los dispositivos periféricos necesarios para su uso.
- **Software**: Conjunto de programas para manejar la base de datos. Este sistema maneja todas las solicitudes formuladas por los usuarios a la base de datos.
- **Usuarios**:
    - **Analista**: Persona encargada de esquematizar los datos y sus relaciones, hace el Diagrama Entidad Relación.
    - **Programador** o **desarrollador**: Persona encargada de programar la aplicación con la base de datos.
    - **DBA**: Persona encargada de la construcción de la base de datos y su administración.
    - **Finales**: Manipulan e interpretan la información.

Ventajas de una base de datos:

- Reducir redundancia.
- Evitar inconsistencia.
- Mantener integridad.
- Controlar concurrencia.
- Aplicar restricciones de seguridad.
- Los datos son independientes de los programas.

Desventajas de una base de datos:

- La seguridad y la integridad pueden ser contraproducentes sin buenos controles.

Las bases de datos manejan conceptos básicos para su tratamiento de información, como son:

- **Registro**: Es una colección de campos (atributos). Un registro, es el conjunto de información referida a una misma persona u objeto.

- **Campo**: Unidad básica de una base de datos.
- **Tabla**: Conjunto de registros contenidos en campos.

Un sistema administrador de bases de datos (**DBMS** DataBase Management System) consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a ellos. La colección de datos se denomina base de datos. Se compone de:

- Base de datos.
- Software para manipular los datos.

Funciones:

- Almacena, recupera y modifica los datos.
- Guarda la consistencia de los datos.
- Soluciona problemas de concurrencia.

Objetivo del DBMS:

- Crear un ambiente en que sea posible guardar y recuperar información de la base de datos en forma eficiente.
- Controlar el acceso concurrente, evitar redundancia, vigilar que se cumplan las restricciones y reglas de integridad, usar elementos que aceleren el acceso físico a los datos, distribuir los bloques del modo más adecuado para el crecimiento y uso de los datos, controlar el acceso y privilegio de los usuarios, recuperar ante fallos ente otras cosas.

El manejo de datos del DBMS incluye tanto la definición como la manipulación y seguridad de los mismos.

Actividades del DBA (persona que administra la base de datos, Database Administrator, por sus siglas en ingles):

- Definir y modificar la estructura de almacenamiento (espacio).
- Definir permisos de acceso.
- Definir estrategias de respaldo y recuperación en caso de fallas.

DICCIONARIO DE DATOS.

Proporciona la siguiente información:

- Nombre de los usuarios.
- Privilegios que tienen los usuarios.
- Nombres de los objetos (Tablas, vistas, índices, sinónimos, procedimientos).
- Espacio ocupado por los objetos.

MODELO DE DATOS.

Es el resultado del análisis de la información y consiste en la representación conceptual de ésta (debe ser claro).

TIPOS DE MODELOS DE DATOS.

- **Modelo de datos jerárquico.** Se puede especificar relaciones de un registro padre y múltiples registros hijos, de manera similar a la estructura de un árbol.
- **Modelo de datos red.** Se puede especificar relaciones entre múltiples registros padres y múltiples registros hijos.
- **Modelo de datos relacional.** Se representa lógicamente la base de datos mediante entidades y sus relaciones entre sí. En este modelo toda la información se representa a través de arreglos bidimensionales o tablas. Las operaciones básicas son:

    - Seleccionar renglones de alguna tabla (SELECT).
    - Seleccionar columnas de alguna tabla (PROJECT).
    - Unir o juntar información de varias tablas (JOIN).

2.2.2 Modelo Entidad-Relación.

Se basa en la percepción del mundo real, que consiste en un conjunto de objetos llamados entidades y las relaciones entre éstas.

Representa la estructura lógica general de la base de datos gráficamente.

Entidad: Es un objeto que se distingue de otros por medio de un conjunto específico de ATRIBUTOS (características propias). Para cada atributo existe un rango de valores permitidos llamado Dominio del atributo.

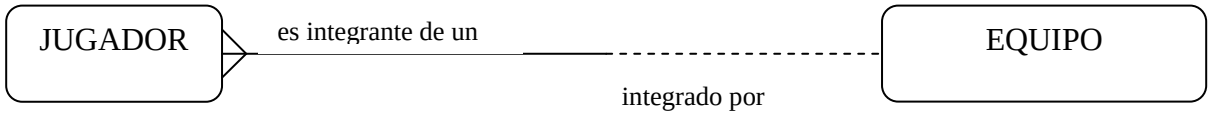
Conjunto de entidades: Es un grupo de entidades del mismo tipo.

Relación: Asociación entre entidades.

Conjunto de relaciones: Es un grupo de relaciones del mismo tipo.

Tipos de Relaciones:

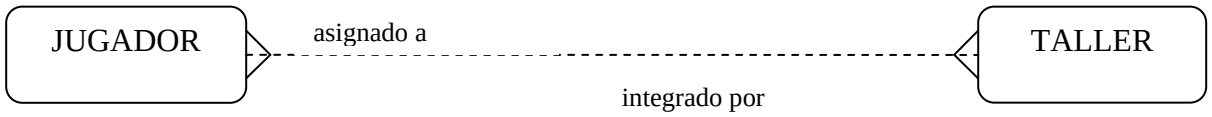
- Una relación muchos a uno (M a 1 ó M:1) tiene el grado de uno o más en una dirección y el grado de uno y solamente uno en la otra dirección. Estas relaciones son las más comunes.



Cada JUGADOR es integrante de uno y solamente un EQUIPO.
Cada EQUIPO está integrado por uno o más JUGADORES.

Las relaciones M:1 obligatorias de ambas direcciones son poco comunes.

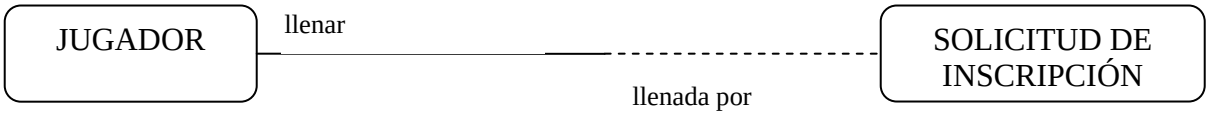
- Una relación Muchos a Muchos (M a M ó M:M) tiene el grado de uno o más en ambas direcciones.



Cada JUGADOR puede estar ASIGNADO A uno o más TALLERES.
Cada TALLER puede estar INTEGRADO POR uno o más JUGADORES.

Las relaciones muchos a muchos son muy comunes, generalmente son opciones en ambas direcciones, pero puede ser opcional en una dirección.

- Una relación uno a uno (1 a 1 ó 1:1) tiene un grado uno y solamente uno en ambas direcciones.



Cada JUGADOR debe llenar una y solamente una SOLICITUD DE INSCRIPCIÓN.
Cada SOLICITUD DE INSCRIPCIÓN debe ser llenada por uno y solamente un JUGADOR.
Las relaciones 1:1 son muy escasas.
Una relación 1:1 que es obligatoria en ambas direcciones es muy difícil de encontrar.
Las entidades que tienen una relación 1:1 pueden ser en realidad la misma entidad.

2.2.3 Modelo Relacional.

El modelo relacional es una forma de ver los datos; es decir, para representar los datos mediante tablas y manipular esa representación mediante operadores. El modelo relacional se ocupa de tres aspectos de los datos: su estructura, su integridad y su manipulación.

La base de datos relacional es un conjunto de tablas bajo una misma identificación que trabajan con base a relaciones entre las mismas, las relaciones pueden ser entre dos o más tablas y puede generarse una nueva a partir de los registros que cumplen con el criterio de correspondencia, las relaciones se llevan a cabo a través de campos denominados llaves.

Una base de datos relacional es un conjunto de relaciones normalizadas.

El modelo relacional representa la segunda generación de los Sistemas Manejadores de Bases de datos (SMBD). En él, todos los datos están estructurados a nivel lógico como tablas formadas por filas y columnas. Un punto fuerte del modelo relacional es la sencillez de su estructura lógica.

El modelo relacional tiene que ver con tres aspectos de los datos:

- **Estructura de datos**. Una base de datos relacional consiste en una colección de tablas, a cada una de las cuales se les asigna un nombre único. Una fila de la tabla representa una relación entre un conjunto de valores, puesto que una tabla es una colección de dichas relaciones, hay una estrecha correspondencia entre el concepto de tabla y el concepto matemático de relación.
- **Integridad de datos**.
- **Manejo de datos**.

Estructura.

El modelo relacional se basa en el concepto matemático de relación, que gráficamente se representa mediante una tabla.

Una **relación** es una tabla con columnas y filas. Los Sistemas de Gestión de Bases de Datos (SGBD) necesitan que el usuario pueda percibir la base de datos como un conjunto de tablas. Esta percepción sólo se aplica a la estructura lógica de la base de datos. No se aplica a la estructura física de la base de datos, que se puede implementar con distintas estructuras de almacenamiento.

Un **atributo** es el nombre de una columna de una relación. En el modelo relacional, las relaciones se utilizan para almacenar información sobre los objetos que se representan en la base de datos. Una relación se representa gráficamente como una tabla bidimensional en la que las filas corresponden a registros individuales y las columnas corresponden a los campos o atributos de esos registros. Los atributos pueden aparecer en la relación en cualquier orden.

Un **dominio** es el conjunto de valores legales de uno o varios atributos. Los dominios constituyen una poderosa característica del modelo relacional. Cada atributo de una base de datos relacional se define sobre un dominio, pudiendo haber varios atributos definidos sobre el mismo dominio.
El concepto de dominio es importante porque permite que el usuario defina, en un lugar común, el significado y la fuente de los valores que los atributos pueden tomar. Esto hace que haya más información disponible para el sistema cuando éste va a ejecutar una operación relacional, de modo que las operaciones que son semánticamente incorrectas, se pueden evitar.

Una **tupla** es una fila de una relación. Los elementos de una relación son las tuplas o filas de la tabla. Las tuplas de una relación no siguen ningún orden.

El **grado de una relación** es el número de atributos que contiene. Esto quiere decir que cada fila de la tabla es una tupla con n valores. El grado de una relación no cambia con frecuencia.

La **cardinalidad** de una relación es el número de tuplas que contiene. Ya que en las relaciones se van insertando y borrando tuplas a menudo, la cardinalidad de las mismas varía constantemente.

Reglas de integridad.

Al definir cada atributo sobre un dominio se impone una restricción sobre el conjunto de valores permitidos para cada atributo. A este tipo de restricciones se les denomina restricciones de dominios. Hay

además dos reglas de integridad muy importantes que son restricciones que se deben cumplir en todas las bases de datos relacionales y en todos sus estados o instancias (las reglas se deben cumplir todo el tiempo). Estas reglas son la regla de integridad de entidades y la regla de integridad referencial. Antes de definirlas, es preciso conocer el concepto de nulo.

Nulos.

Cuando en una tupla un atributo es desconocido, se dice que es nulo. Un nulo no representa el valor cero ni la cadena vacía, éstos son valores que tienen significado. El nulo implica ausencia de información, bien porque al insertar la tupla se desconocía el valor del atributo, o bien porque para dicha tupla el atributo no tiene sentido. Ya que los nulos no son valores, deben tratarse de modo diferente, lo que causa problemas de implementación.

Regla de integridad de entidades.

La primera regla de integridad se aplica a las claves primarias de las relaciones base: ninguno de los atributos que componen la clave primaria puede ser nulo.

Una clave primaria es un identificador irreducible que se utiliza para identificar de modo único las tuplas. Ningún subconjunto de la clave primaria sirve para identificar las tuplas de modo único.

Regla de integridad referencial.

La segunda regla de integridad se aplica a las claves ajenas: si en una relación hay alguna clave ajena, sus valores deben coincidir con valores de la clave primaria a la que hace referencia, o bien, deben ser completamente nulos.

Gráficamente se suelen representar las relaciones mediante tablas. Los nombres de las columnas corresponden a los nombres de los atributos y las filas son cada una de las tuplas de la relación. Los valores que aparecen en cada una de las columnas pertenecen al conjunto de valores del dominio sobre el que está definido el atributo correspondiente.

Propiedades de las relaciones.

- Cada relación tiene un nombre y éste es distinto del nombre de todas las demás.
- Los valores de los atributos son atómicos: en cada tupla, cada atributo toma un solo valor. Se dice que las relaciones están normalizadas.

- No hay dos atributos que se llamen igual.
- El orden de los atributos no importa: los atributos no están ordenados.
- Cada tupla es distinta de las demás: no hay tuplas duplicadas.
- El orden de las tuplas no importa: las tuplas no están ordenadas.

Tipos de relaciones.

- **Relaciones base**. Son relaciones reales que tienen nombre y forman parte directa de la base de datos almacenada (son autónomas).
- **Vistas**. También denominadas relaciones virtuales, son relaciones con nombre y derivadas. Se representan mediante su definición en términos de otras relaciones con nombre, no poseen datos almacenados propios.
- **Instantáneas**. Son relaciones con nombre y derivadas. Pero a diferencia de las vistas, son reales, no virtuales. Están representadas no sólo por su definición en términos de otras relaciones con nombre, sino también por sus propios datos almacenados. Son relaciones de sólo lectura y se refrescan periódicamente.
- **Resultados de consultas**. Son las relaciones resultantes de alguna consulta especificada. Pueden o no tener nombre y no persisten en la base de datos.
- **Resultados intermedios**. Son las relaciones que contienen los resultados de las subconsultas. Normalmente no tienen nombre y tampoco persisten en la base de datos.
- **Resultados temporales**. Son relaciones con nombre, similares a las relaciones base o a las instantáneas, pero la diferencia es que se destruyen automáticamente en algún momento apropiado.

Una tabla es un conjunto de registros. En las bases de datos relacionales se emplean tablas bidimensionales las cuales son una de las maneras más naturales de representar y visualizar datos. Las tablas deben organizarse y definirse de tal forma que no se pierdan las relaciones entre datos. Una tabla es una matriz bidimensional por lo que un renglón es un registro y una columna es un campo.

Las tablas tienen las siguientes características:

- Cada columna de la tabla representa un campo.
- Son homogéneas por columna, es decir, todos los datos de una columna son del mismo tipo o clase de datos.
- Cada columna tiene nombre propio.
- Todos los renglones son diferentes, deben variar en al menos una columna.

- Tanto los renglones como las columnas pueden ser considerados en cualquier secuencia y en cualquier momento sin afectar la información ni la semántica de cualquier función que utilice la tabla.

Las entidades son objetos tangibles o intangibles del mundo real sobre las cuales se almacena información y que tiene propiedades o atributos que eventualmente se deben registrar. A cada registro debe referirse por medio de un identificador de entidad para ser relacionado con la entidad. Uno de los atributos de la entidad es la que toma el carácter de identificador de entidad.

Características de las bases de datos relacionales.

- Integridad de datos. La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar. Estas restricciones se pueden aplicar tanto a los datos, como a sus relaciones, y es el SGBD relacional quien se debe encargar de mantenerlas.
- Seguridad. La seguridad de la base de datos es la protección de la base de datos frente a usuarios no autorizados. Sin buenas medidas de seguridad, la integración de datos en los sistemas de bases de datos hace que éstos sean más vulnerables que en los sistemas de ficheros. Sin embargo, los SGBD relacionales permiten mantener la seguridad mediante el establecimiento de claves para identificar al personal autorizado a utilizar la base de datos. Las autorizaciones se pueden realizar a nivel de operaciones, de modo que un usuario pueda estar autorizado a consultar ciertos datos pero no a actualizarlos, por ejemplo.
- Accesibilidad a los datos. Muchos SGBD relacionales proporcionan lenguajes de consultas o generadores de informes que permiten al usuario hacer cualquier tipo de consultas sobre los datos, sin que sea necesario que un programador escriba una aplicación que realice tal tarea.
- Productividad. El SGBD relacional proporciona muchas de las funciones estándar que el programador necesita escribir en un sistema de ficheros. A nivel básico, el SGBD relacional proporciona todas las rutinas de manejo de ficheros típicas de los programas de aplicación. El hecho de disponer de estas funciones permite al programador centrarse mejor en la función específica requerida por los usuarios, sin tener que preocuparse de los detalles de implementación de bajo nivel. Muchos SGBD relacionales también proporcionan un entorno de cuarta generación consistente en un conjunto de herramientas que simplifican, en gran medida, el desarrollo de las aplicaciones que acceden a la base de datos. Gracias a estas herramientas, el programador puede ofrecer una mayor productividad en un tiempo menor.
- Mantenimiento gracias a la independencia de datos. En los sistemas de ficheros, las descripciones de los datos se encuentran inmersas en los programas de aplicación que los manejan. Esto hace que los programas sean dependientes de los datos, de modo que un cambio

en su estructura, o un cambio en el modo en que se almacena en disco, requiere cambios importantes en los programas cuyos datos se ven afectados. Sin embargo, los SGBD relacionales separan las descripciones de los datos de las aplicaciones. Esto es lo que se conoce como independencia de datos, gracias a la cual se simplifica el mantenimiento de las aplicaciones que acceden a la base de datos.

- Aumento de la concurrencia. En algunos sistemas de ficheros, si hay varios usuarios que pueden acceder simultáneamente a un mismo fichero, es posible que el acceso interfiera entre ellos de modo que se pierda información o, incluso, que se pierda la integridad. La mayoría de los SGBD relacionales gestionan el acceso concurrente a la base de datos y garantizan que no ocurran problemas de este tipo.
- Servicios de copias de seguridad y de recuperación ante fallos. Muchos sistemas de ficheros dejan que sea el usuario quien proporcione las medidas necesarias para proteger los datos ante fallos en el sistema o en las aplicaciones. Los usuarios tienen que hacer copias de seguridad cada día, y si se produce algún fallo, utilizar estas copias para restaurarlos. En este caso, todo el trabajo realizado sobre los datos desde que se hizo la última copia de seguridad se pierde y se tiene que volver a realizar. Sin embargo, los SGBD relacionales funcionan de modo que se minimiza la cantidad de trabajo perdidos cuando se produce un fallo.

Objetivos.

Los objetivos que se persiguen con la organización de la base de datos pueden ser divididos en primarios y secundarios.

Objetivos primarios de la organización de la base de datos.

- Los datos podrán usarse de múltiples maneras.
- Proteger la inversión intelectual.
- Bajo costo.
- Menor proliferación de datos.
- Desempeño.
- Claridad.
- Facilidad de uso.
- Flexibilidad.
- Facilidad para el cambio.
- Precisión y coherencia.
- Reserva.
- Disponibilidad.

Objetivos secundarios.

- Independencia física de los datos.
- Independencia lógica de los datos.
- Referencia controlada.
- Normalización de los datos dentro de un organismo.
- Diccionario de datos.
- Lenguaje de usuario final.
- Controles de integridad.
- Fácil recuperación en caso de fallo.

Lenguajes relacionales.

Son varios los lenguajes utilizados por los SMBD relacionales para manejar las relaciones. Algunos de ellos son procedurales, lo que quiere decir que el usuario dice al sistema exactamente cómo debe manipular los datos. Otros son no procedurales, que significa que el usuario dice qué datos necesita, en lugar de decir cómo deben obtenerse.

La base de los lenguajes relacionales es el álgebra relacional y el cálculo relacional, definidos por Codd. Se puede decir que el álgebra es un lenguaje procedural (de alto nivel), mientras que el cálculo relacional es un lenguaje no procedural. Sin embargo, ambos lenguajes son equivalentes: para cada expresión del álgebra, se puede encontrar una expresión equivalente en el cálculo, y viceversa.

El álgebra relacional (o el cálculo relacional) se utilizan para medir la potencia de los lenguajes relacionales. Si un lenguaje permite obtener cualquier relación que se pueda derivar mediante el álgebra relacional, se dice que es relacionalmente completo. La mayoría de los lenguajes relacionales son relacionalmente completos, pero tienen más potencia que el álgebra o el cálculo porque se les han añadido operadores especiales.

Tanto el álgebra como el cálculo son lenguajes formales no muy “amigables". Pero se deben estudiar porque sirven para ilustrar las operaciones básicas que todo lenguaje de manejo datos debe ofrecer. Además, han sido la base para otros lenguajes relacionales de manejo de datos de más alto nivel.

El modelo relacional.

En 1970, E. F. Codd publicó un artículo en el que aplicaba los conceptos de una rama de las matemáticas llamada álgebra relacional, a los problemas de almacenar enormes cantidades de datos y

este artículo dio inicio a lo que la comunidad de las bases de datos nombró como el modelo de bases de datos relacionales. Este modelo es una forma particular de estructurar y procesar una base de datos.

Ventajas.

Los datos se almacenan de un modo en el que los usuarios lo entienden con más facilidad. Los datos se almacenan como tablas y las relaciones entre las filas y las tablas son visibles en los datos. Cuando se usa el modelo relacional el usuario sólo debe especificar cuáles registros quiere procesar.

Resistencias.

Los sistemas de bases de datos relacionales requieren de más recursos computacionales y, por lo tanto, al principio eran mucho más lentos que los sistemas basados en modelos anteriores por lo que resultaron imprácticos hasta los años 80, cuando se desarrolló un hardware más rápido para computadoras y la relación precio-desempeño de las computadoras cayó de un modo dramático. Los programadores tuvieron que aprender un nuevo modo de pensar acerca del procesamiento de datos.

Una de las grandes ventajas del modelo relacional es que define también un álgebra, llamada "álgebra relacional". Todas las manipulaciones posibles sobre las relaciones se obtienen gracias a la combinación de tan solo cinco operadores: RESTRICT, PROJECT, TIME, UNION y MINUS. Por comodidad, se han definido también tres operadores adicionales que de todos modos se pueden obtener aplicando los cinco fundamentales: JOIN, INTERSECT y DIVIDE. Los operadores relacionales reciben como argumento una relación o un conjunto de relaciones y restituyen una única relación como resultado.

Operadores:

- RESTRICT. Restituye una relación que contiene un subconjunto de las tuplas de la relación a la que se aplica. Los atributos se quedan como estaban.
- PROJECT. Restituye una relación con un subconjunto de los atributos de la relación a la que viene aplicado. Las tuplas de la relación resultado se componen de las tuplas de la relación original, de manera que siguen siendo un conjunto en sentido matemático.
- TIME. Se aplica a dos relaciones y efectúa el producto cartesiano de las tuplas. Cada tupla de la primera relación está concatenada con cada tupla de la segunda.
- JOIN. Se concatenan las tuplas de dos relaciones de acuerdo con el valor de un conjunto de sus atributos.

- UNION. Aplicando este operador a dos relaciones compatibles, se obtiene una que contiene las tuplas de ambas relaciones. Dos relaciones son compatibles si tienen el mismo número de atributos y los atributos correspondientes en las dos relaciones tienen el mismo dominio.
- MINUS. Aplicado a dos relaciones compatibles restituye una tercera que contiene las tuplas que se encuentran sólo en la primera relación.
- INTERSECT. Aplicado a dos relaciones compatibles restituye una relación que contiene las tuplas que existen en ambas.
- DIVIDE. Aplicado a dos relaciones que tengan atributos comunes, restituye una tercera que contiene todas las tuplas de la primera relación que se puede hacer que correspondan con todos los valores de la segunda relación.

Álgebra relacional.

Las operaciones de álgebra relacional manipulan relaciones. Esto significa que estas operaciones usan uno o dos relaciones existentes para crear una nueva relación. Esta nueva relación puede entonces usarse como entrada para una nueva operación. Esto hace considerablemente más fácil la solución de las consultas, debido a que se puede experimentar con soluciones parciales hasta encontrar la proposición con la que se trabajará.

El álgebra relacional consta de las siguientes operaciones:

- Unión.
- Intersección.
- Diferencia.
- Producto.
- Selección.
- Proyección.
- Reunión.
- División.
- Asignación.

Las cuatro primeras se toman de la teoría de conjunto de las matemáticas; las cuatro siguientes son operaciones propias del álgebra relacional y la última es la operación estándar de dar un valor a un elemento.

Unión. Combina datos de varias relaciones. No siempre es posible realizar consultas de unión entre

varias tablas, para poder realizar esta operación es necesario e imprescindible que las tablas a unir tengan las mismas estructuras y que sus campos sean iguales.

Intersección. Identifica filas que son comunes en dos relaciones.

Diferencia. Identifica filas que están en una relación y no en otra.

Producto. Consiste en la realización de un producto cartesiano entre dos tablas dando como resultado todas las posibles combinaciones entre los registros de la primera y los registros de la segunda.

Selección. Consiste en recuperar un conjunto de registros de una tabla o de una relación indicando las condiciones que deben cumplir los registros recuperados, de tal forma que los registros devueltos por la selección han de satisfacer todas las condiciones que se hayan establecido. Esta operación es la que normalmente se conoce como consulta.

En este tipo de consulta se emplean los diferentes operadores de comparación (=, >, <, >=, <=, <>), los operadores lógicos (and, or, xor) o la negación lógica (not).

Proyección. Una proyección es un caso concreto de la operación selección, esta última devuelve todos los campos de aquellos registros que cumplen la condición que se ha establecido. Una proyección es una selección en la que seleccionamos aquellos campos que deseamos recuperar.

Reunión. La reunión se utiliza para recuperar datos a través de varias tablas conectadas unas con otras mediante cláusulas JOIN, en cualquiera de sus tres variantes INNER, LEFT, RIGHT. La operación reunión se puede combinar con las operaciones selección y proyección.

División. La operación división es la contraria a la operación producto.

Asignación. Esta operación algebraica consiste en asignar un valor a uno o varios campos de una tabla.

Cálculo relacional.

El cálculo relacional usa un enfoque completamente diferente al álgebra relacional. No obstante, los dos lenguajes son lógicamente equivalentes. Esto significa que cualquier consulta que pueda resolverse en un lenguaje puede resolverse en el otro.

La solución para toda consulta en este tipo de cálculo se define por:

a) Una lista de resultados.
b) Una sentencia de cualificación.

La lista de resultados son aquellos registros que cumplen las condiciones que deseamos. La sentencia de cualificación contiene las condiciones que deseamos que cumplan los registros de la lista de resultados. La diferencia entre el cálculo y el álgebra radica en que el cálculo realiza la operación en un único paso, sin necesidad de tener que obtener tablas intermedias, el álgebra realiza las operaciones paso a paso.

El cálculo relacional se apoya en algún lenguaje de interrogación de bases de datos como puede ser el SQL. Incluye un concepto nuevo denominado cuantificador, los cuantificadores tratan de averiguar el número de registros afectados por una determinada operación, incluso antes de realizarla. Según su naturaleza los podemos dividir en dos grupos:

- **Cuantificadores existenciales.** Son aquellos que tratan de averiguar el número de registros que devolvería un tipo de consulta.
- **Cuantificadores universales.** Son aquellos que indican que una condición se aplica a todas las filas de algún tipo. Se usa para brindar la misma capacidad que la operación división del álgebra relacional.

2.2.4 Normalización.

El proceso de normalización es un estándar que consiste, básicamente, en un proceso de conversión de las relaciones entre las entidades, evitando:

- La redundancia de los datos: repetición de datos en un sistema.
- Anomalías de actualización: inconsistencias de los datos como resultado de datos redundantes y actualizaciones parciales.
- Anomalías de borrado: pérdidas no intencionadas de datos debido a que se han borrado otros datos.
- Anomalías de inserción: imposibilidad de adicionar datos en la base de datos debido a la ausencia de otros datos.

La razón de ser de las formas normales consiste en la estandarización de los conceptos relacionados al diseño eficiente de las estructuras y esquemas de una base de datos. La aplicación de las formas normales permiten la aplicación de un estándar de eficiencia en niveles ascendentes.

El proceso de normalización nos conduce hasta el modelo físico de datos y consta de varias fases denominadas formas normales.

Definición de la clave.

Antes de proceder a la normalización de la tabla, primero se debe definir el concepto de clave, esta clave deberá contener un valor único para cada registro (no podrán existir dos valores iguales en toda la tabla) y podrá estar formado por un único campo o por un grupo de campos.

Primera forma normal (1NF).

Se dice que una tabla se encuentra en primera forma normal (1NF) si y sólo si cada uno de los campos contiene un único valor para un registro determinado. Una vez normalizada la tabla en 1NF, podemos pasar a la segunda forma normal.

Segunda forma normal (2NF).

La segunda forma normal compara todos y cada uno de los campos de la tabla con la clave definida. Si todos los campos dependen directamente de la clave se dice que la tabla está es segunda forma normal (2NF).

Tercera forma normal (3NF).

Se dice que una tabla está en tercera forma normal si y sólo si los campos de la tabla dependen únicamente de la clave, dicho en otras palabras los campos de las tablas no dependen unos de otros.

Cuarta forma normal (4NF).

Una tabla está en cuarta forma normal si y sólo si para cualquier combinación clave - campo no existen valores duplicados.

CAPÍTULO 3. Análisis del Sistema.

Para el desarrollo del sistema se optó por seguir el ciclo de vida clásico. Partiremos por conocer los requerimientos y necesidades del cliente, para identificar los problemas y así proponer la alternativa de solución del sistema.

Buscando una óptima administración de los recursos y las tareas que se tienen en Pumitas, se ha realizado un levantamiento de información para conocer las necesidades del área técnica. Los requerimientos se han desglosado en la siguiente sección en donde se destacan los puntos más importantes que deben estar contenidos en el presente desarrollo.

3.1 Requerimientos.

Primeramente, se necesita almacenar toda la información respectiva del personal que labora e interactúa en Pumitas:

- Presidente.
- Administrador Tesorero.
- Coordinadores Administrativos.
- Coordinadores Técnicos.
- Cuerpo arbitral.
- Delegados de los equipos.
- Monitores.
- Jugadores.
- Servicio médico.

Se necesita tener la información correspondiente de las distintas categorías para utilidad de la coordinación técnica de Pumitas.

De las categorías:

- Desplegar información por categoría de los coordinadores, monitores, equipos y jugadores que la componen.
- Relación concerniente a los bloques de trabajo por categoría.

De los coordinadores de categoría:

- Relación de los equipos asignados a cada coordinador dependiendo del bloque asignado, para un mejor seguimiento por parte de la coordinación técnica.
- Datos generales y datos médicos.

De los monitores:

- Tener el historial de participaciones en las distintas categorías, es decir, el número de años que ha participado en cada una de ellas, el equipo con el que actualmente se encuentra, así como sus antecedentes o datos escolares.
- Datos generales y datos médicos.

De los equipos:

- Sus características, su monitor y coordinador asociado, así como la lista de asistencia de los integrantes del equipo.

De los jugadores:

- Datos generales, técnicos y médicos.
- Historial del avance por las distintas categorías y equipos por los que ha incursionado en Pumitas.
- Registrar la evaluación de cada jugador para facilitar la asignación de equipos de futuras temporadas.

De los talleres:

- Tener actualizada la información de la asistencia de los jugadores a cada taller (Acondicionamiento Físico, Motricidad, Multilateralidad o Porteros) por categoría, para tener un seguimiento del trabajo desarrollado a lo largo de la temporada.

Además de:

- Facilitar y acelerar el acceso a la información, la cual permita una toma de decisiones acertada y con total conocimiento de las necesidades de Pumitas.

- Cubrir la necesidad de respaldo, edición y consulta de información de Pumitas. Uno de los aspectos más importantes es la seguridad del sistema la cual deberá estar basada en la utilización de usuarios y contraseñas, lo que nos permitirá que sólo el personal autorizado ingrese al sistema. El nivel de seguridad se encontrará en cada pantalla que se desee editar.

Todo esto nos permitirá llevar un buen seguimiento, control y manejo de información de Pumitas.

3.2 Propuesta de Solución.

Crear un sistema que trabaje con una base de datos relacional que podrá interactuar con el usuario mediante una interfaz gráfica, dando con ello confiabilidad, seguridad y rapidez en el manejo de la información.

El sistema se organizará en forma modular y su acceso estará supeditado a la validación de usuarios y a la asignación de privilegios a usuarios.

Los privilegios permiten a los usuarios el manejo de la siguiente información:

- Captura.
- Edición o actualización.
- Consulta.
- Generación de reportes de las evaluaciones realizadas.
- Impresión.

Una vez identificadas las necesidades y requerimientos de Pumitas y observando que el sistema es viable, se crea un modelo del mismo donde queden reflejadas las interrelaciones de los distintos elementos que lo conforman.

Atendiendo a los requerimientos anteriores, se propone la creación de un sistema de software que permita tener el control y manejo de la información de Pumitas, así, se propone el sistema Pumidata 1.0.

Toda la información estará almacenada en una base de datos Access 2002 que soporta todo el volumen de información y que pueda controlar ella misma la integridad referencial.

3.2.1 Justificación.

La finalidad de Pumidata es ver de forma inmediata y completa todas las áreas que conforman a Pumitas. También permitirá ver las relaciones específicas por categoría y de los diversos talleres impartidos.

Debido al acceso que cualquier persona puede tener al sistema se hace necesario agregar accesos controlados por medio de un usuario y contraseña, éstas serán controladas por dos mecanismos; el primero será la contraseña de la base de datos Access en donde se tendrá la información y la segunda será controlada por cada página o formulario al que se pretenda tener acceso.

3.3 Modelado de Datos.

El modelado de datos hace uso de diagramas de entidad-relación en donde se describen e identifican los datos y sus relaciones. Adicionalmente nos permite conocer los detalles del almacenamiento de datos y sus relaciones con los procesos dentro del modelo de flujo de datos.

A través de una representación gráfica vamos a realizar un análisis del comportamiento de la información que manejará nuestro sistema.

3.3.1 Diagrama Entidad-Relación.

El diagrama entidad-relación nos mostrará las relaciones entre las diferentes entidades de Pumidata 1.0 que son:

- La entidad Datos Generales.
- La entidad Jugadores.
- La entidad Monitores.
- La entidad Equipos.
- La entidad Coordinadores.
- La entidad Categorías.
- La entidad Facultades.
- La entidad Asistencia a Entrenamientos.
- La entidad Asistencia a Juegos.
- La entidad Asistencia a Acondicionamiento Físico.
- La entidad Asistencia a Motricidad.

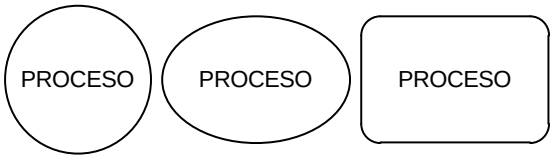
- La entidad Asistencia a Multilateralidad.
- La entidad Asistencia a Porteros.

3.3.2 Diagrama de Flujo de Datos

Los diagramas de flujo de datos se usan para representar el flujo de la información. En su forma más simple constan de los siguientes elementos: entradas, salidas y proceso. A las entradas y salidas se les denomina como unidades externas. Con más detalle se pueden figurar los elementos que a continuación se listan: unidad o entidad externa, proceso, elementos de datos (este nos muestra la dirección del flujo de información) y el almacén de datos.

- **La unidad externa o terminal:**
  - Representa: Entes generadores o receptores de información.
  - Representación gráfica: Rectángulo.

- **El proceso:**
  - Representa: Una transformación de la información.
  - Representación gráfica: Círculo, óvalo o rectángulo con esquinas redondeadas.



- **Flujo de datos:**
  - Representa: La dirección del flujo de información.
  - Representación gráfica: Una flecha. Estas flechas deben etiquetarse.

- **Almacén de datos:**
  - Representa: Depósito de datos que se almacenan para ser usados por uno o más procesos.
  - Representación gráfica: Mediante dos líneas paralelas etiquetadas.

3.3.2.1 Diagrama de procesos.

En el nivel cero, el diagrama de flujo de datos sólo brinda una visión muy genérica del sistema. Será en diagramas posteriores cuando se pueda comprender el funcionamiento del sistema.

En este nivel, el diagrama suele conocerse como diagrama de contexto y sólo da idea de los flujos de datos de entrada y de salida del software.

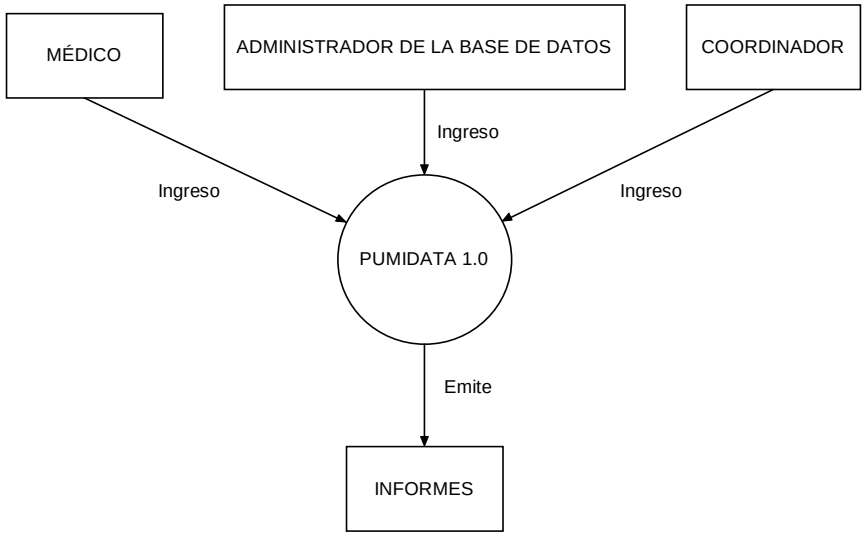


Figura 3.1 Nivel 0.

En el nivel 1 del diagrama de datos se obtiene una visión más específica del sistema como se muestra en la siguiente figura.

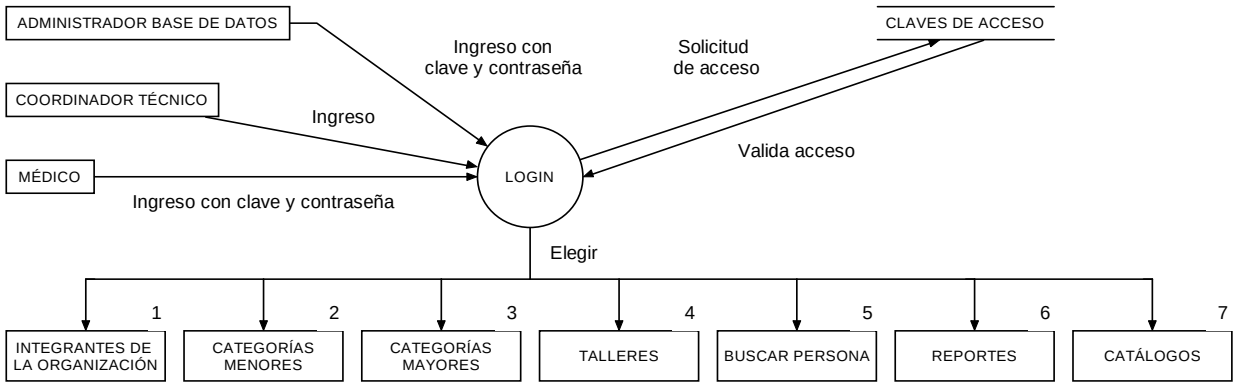


Figura 3.2 Nivel 1.

Como se observa, con el flujo de datos del Administrador de la Base de Datos, el Coordinador Técnico o el Médico, se procede a la validación, la cual revisa el rol que desempeña el usuario al entrar a Pumidata 1.0.

Una vez que el usuario haya ingresado al sistema, tiene la opción de revisar todas las opciones, y sólo podrá modificar información dependiendo de los permisos que el Administrador le haya otorgado.

Integrantes de la Organización. Se presentan todos los posibles protagonistas de la organización, a fin de tener un mejor conocimiento del entorno de Pumitas.

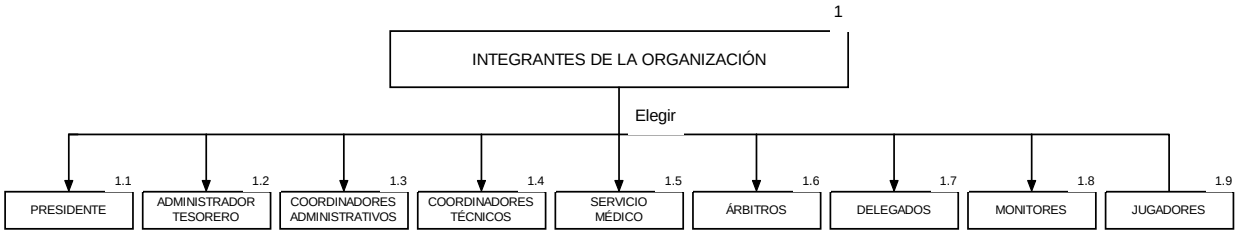


Figura 3.3 Integrantes de la Organización

Presidente. Se presentan sus datos generales y médicos, así como sus antecedentes.

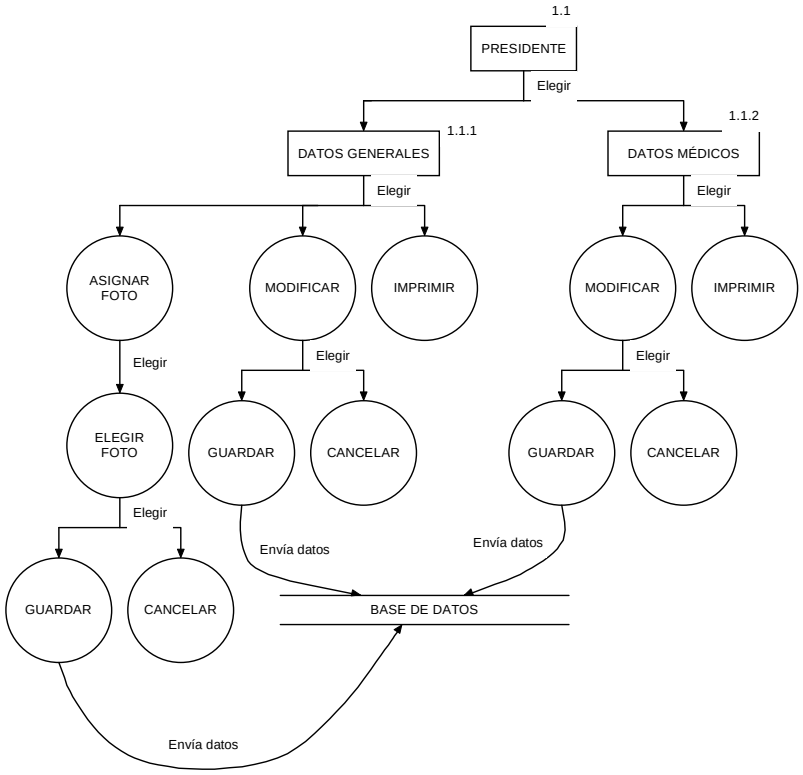


Figura 3.4 Presidente.

Administrador tesorero. Se presentan sus datos generales y médicos, así como sus antecedentes.

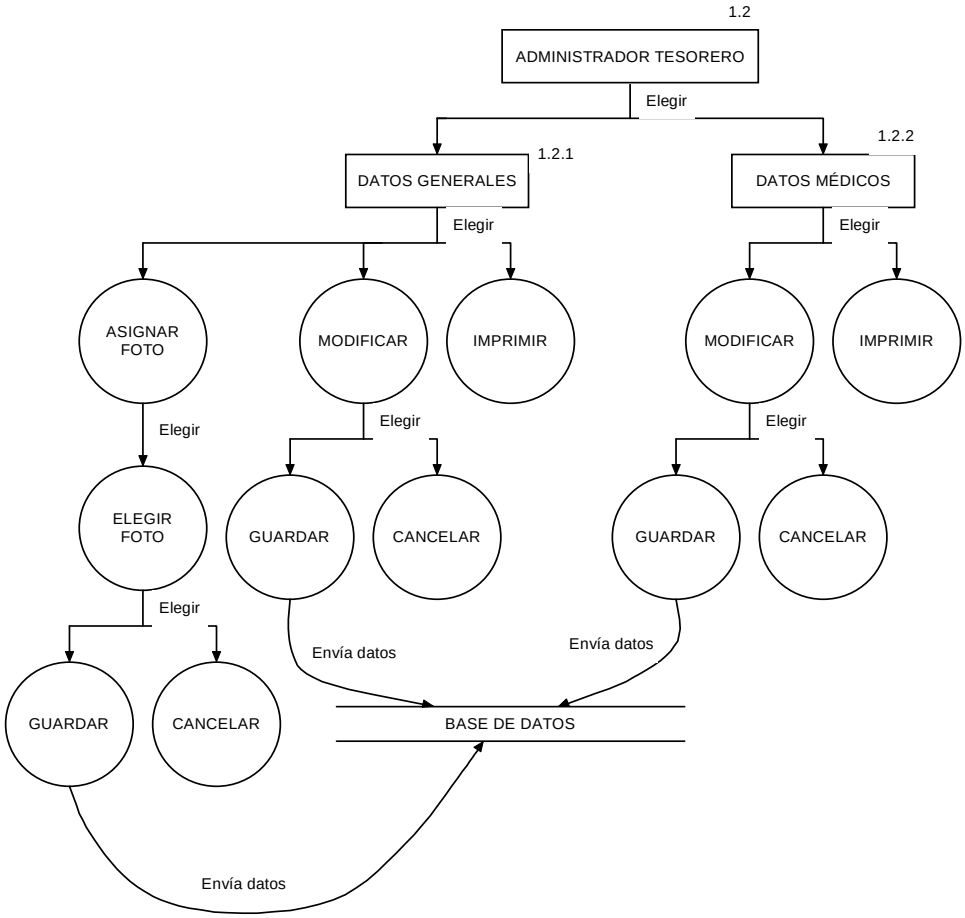


Figura 3.5 Administrador Tesorero.

Coordinadores Administrativos. Se tienen las distintas categorías de Pumitas para tener la información correspondiente y oportuna de los mismos.

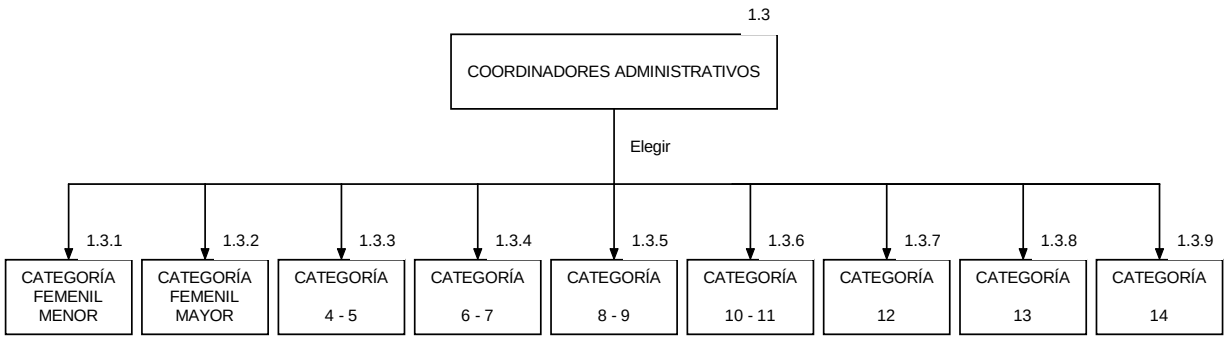


Figura 3.6 Coordinadores Administrativos.

Coordinadores Técnicos. Se cuenta con la información de las distintas categorías.

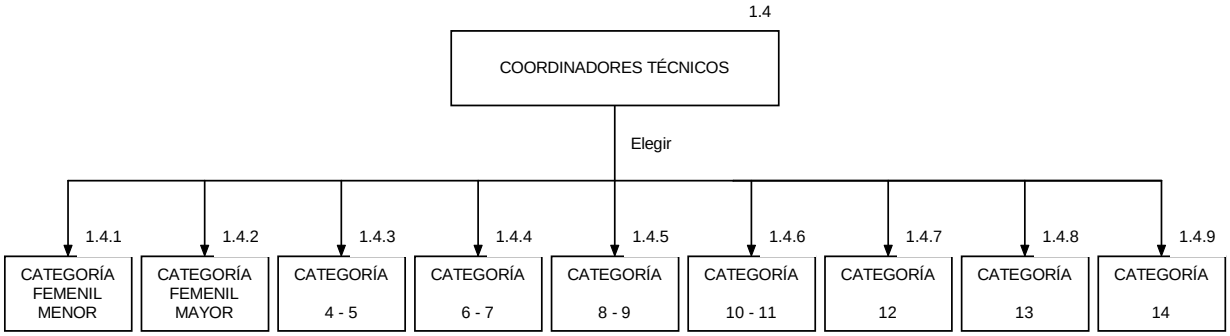


Figura 3.7 Coordinadores Técnicos.

Servicio Médico.

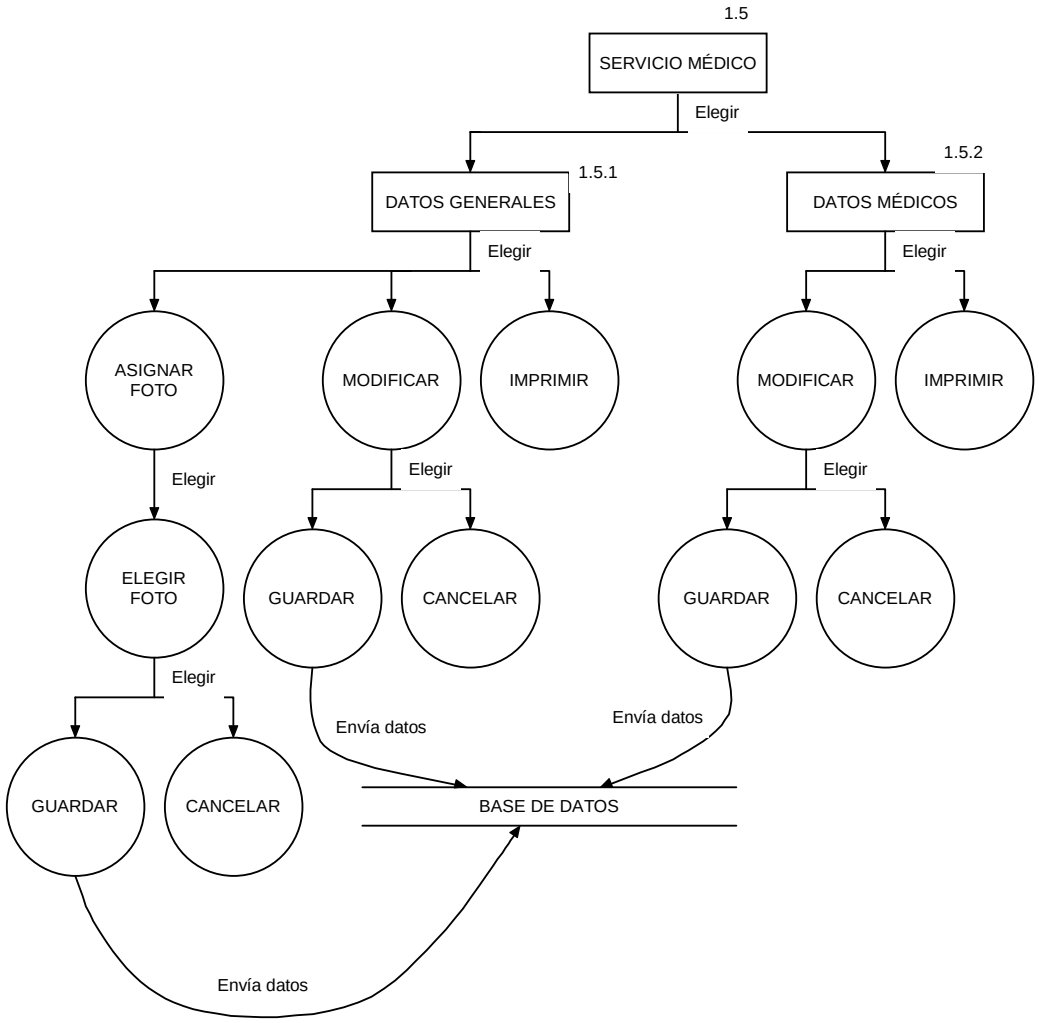


Figura 3.8 Servicio Médico.

Árbitros.

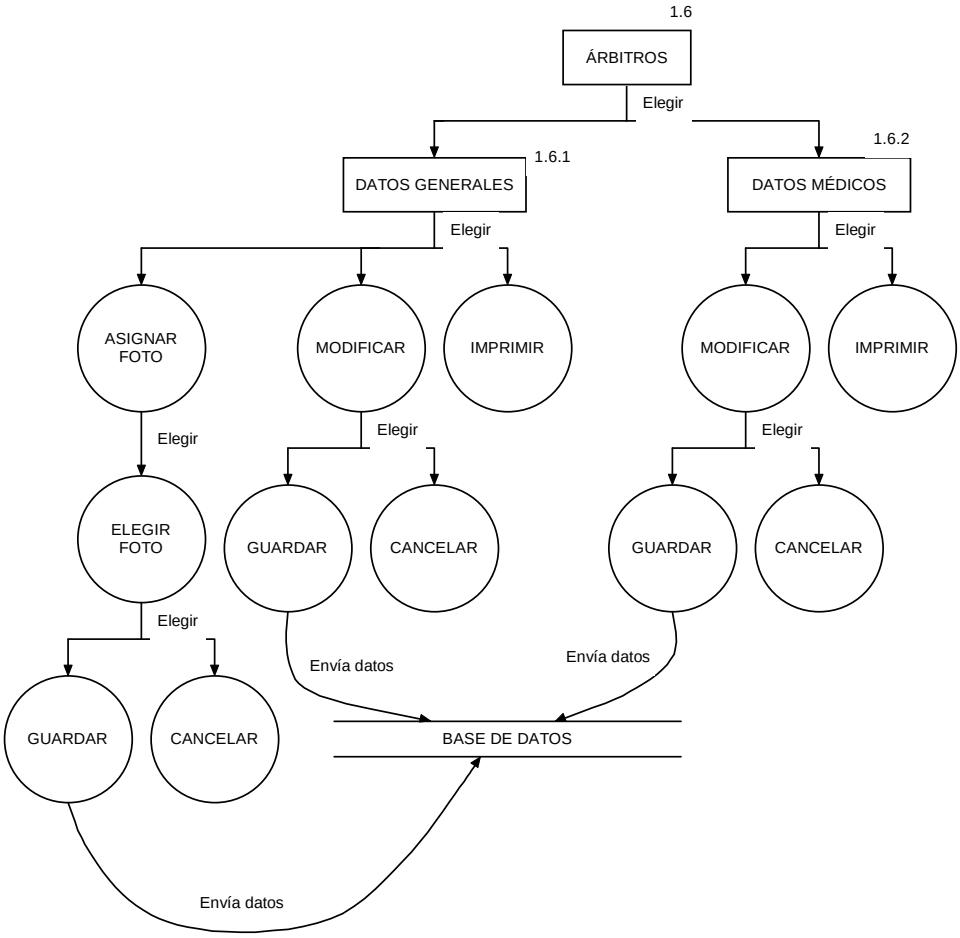


Figura 3.9 Árbitros.

Delegados.

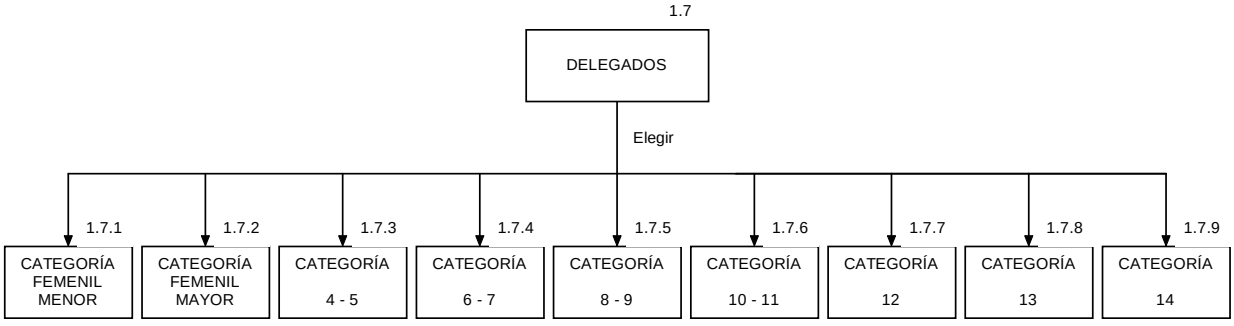


Figura 3.10 Delegados.

Monitores.

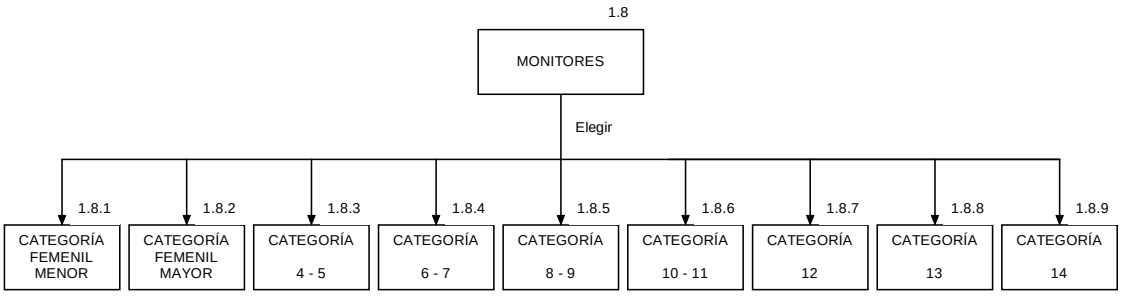


Figura 3.11 Monitores.

Jugadores.

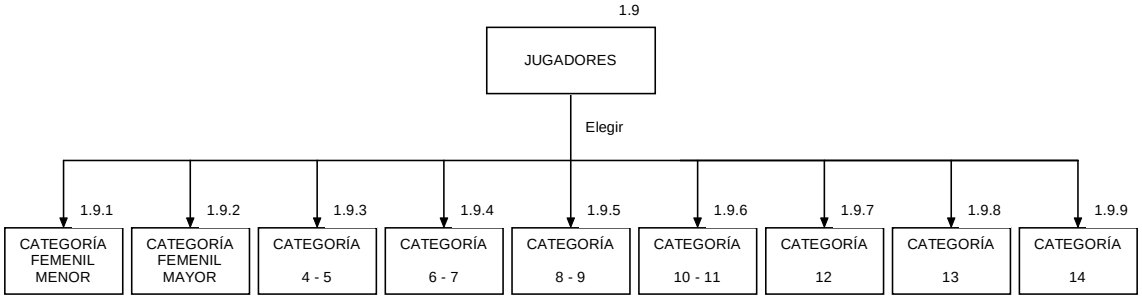


Figura 3.12 Jugadores.

Coordinador Administrativo.

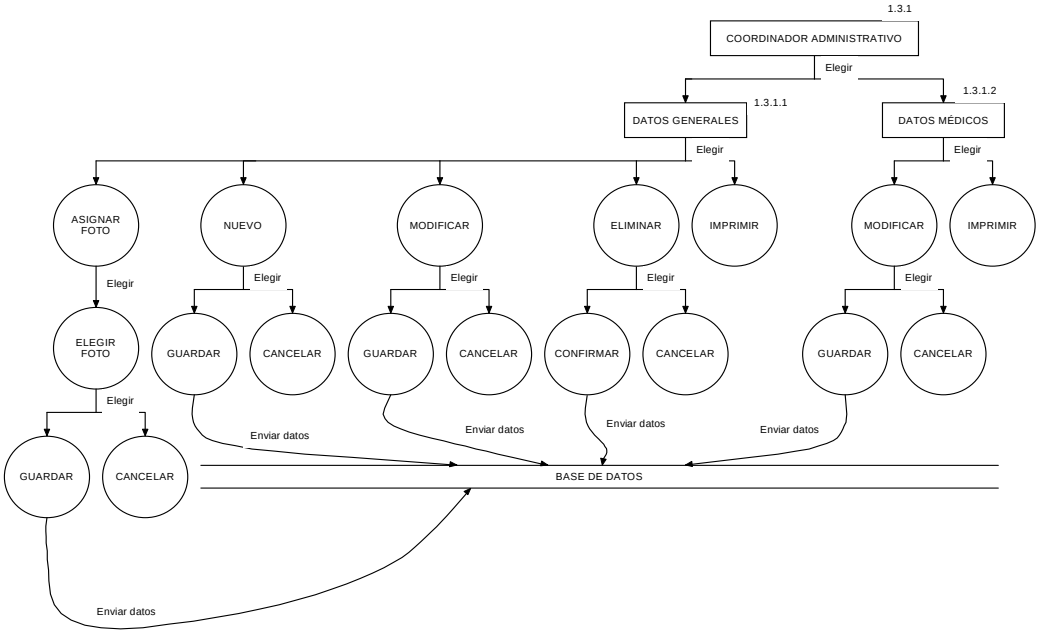


Figura 3.13 Coordinador Administrativo.

Coordinador Técnico.

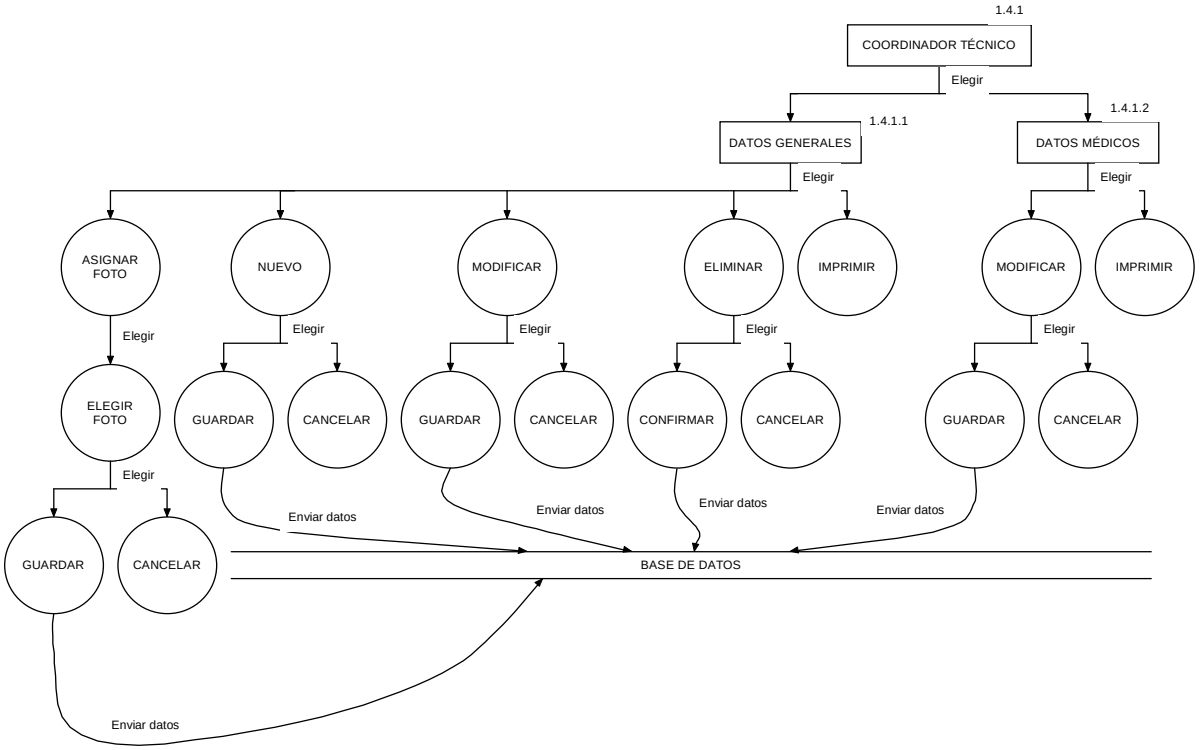


Figura 3.14 Coordinador Técnico.

Delegados.

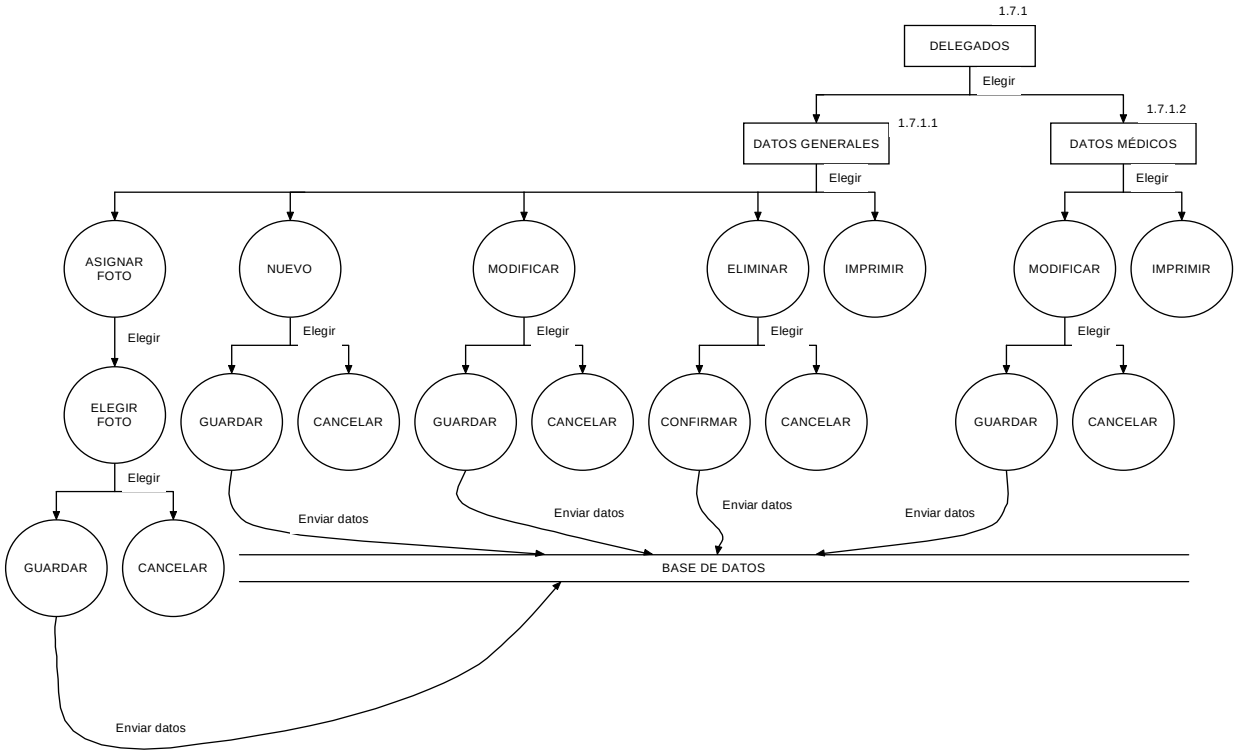


Figura 3.15 Delegados.

Monitores.

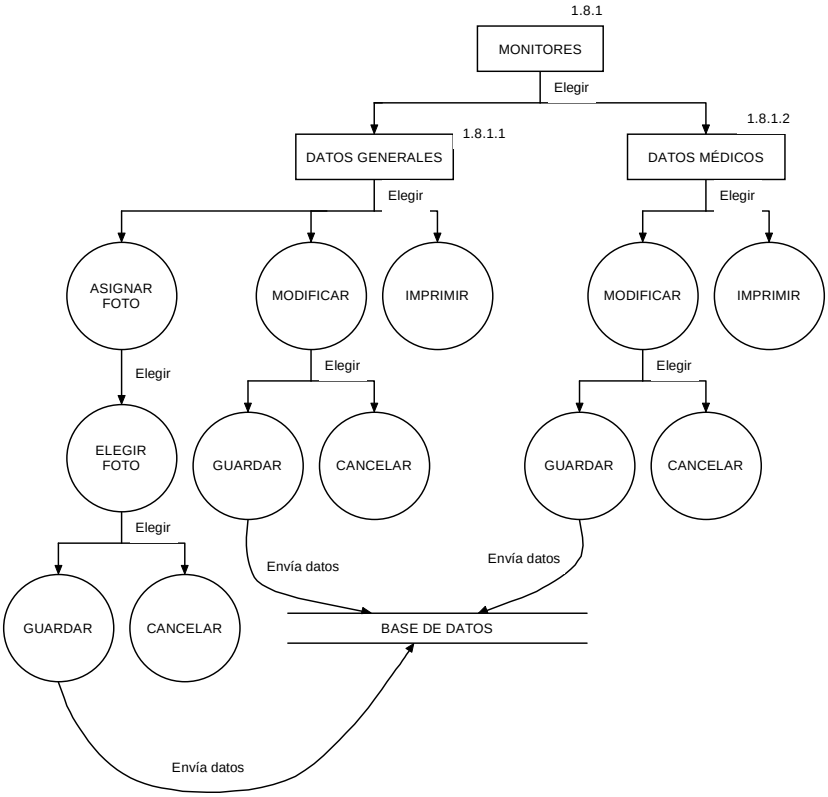


Figura 3.16 Monitores.

Jugadores.

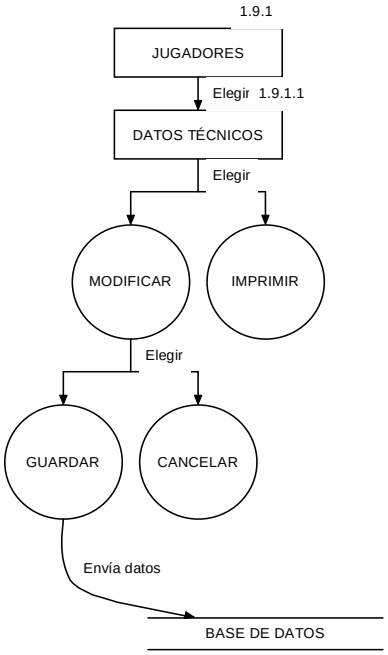


Figura 3.17 Jugadores

Categoría Menores.

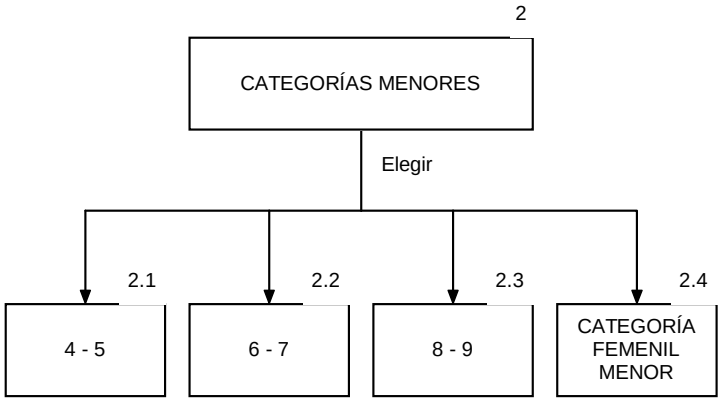


Figura 3.18 Categoría Menores.

Categorías Mayores.

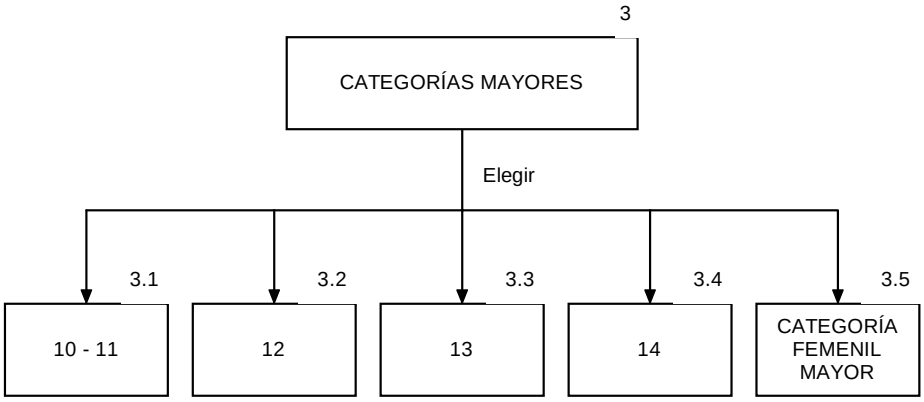


Figura 3.19 Categorías Mayores.

Talleres.

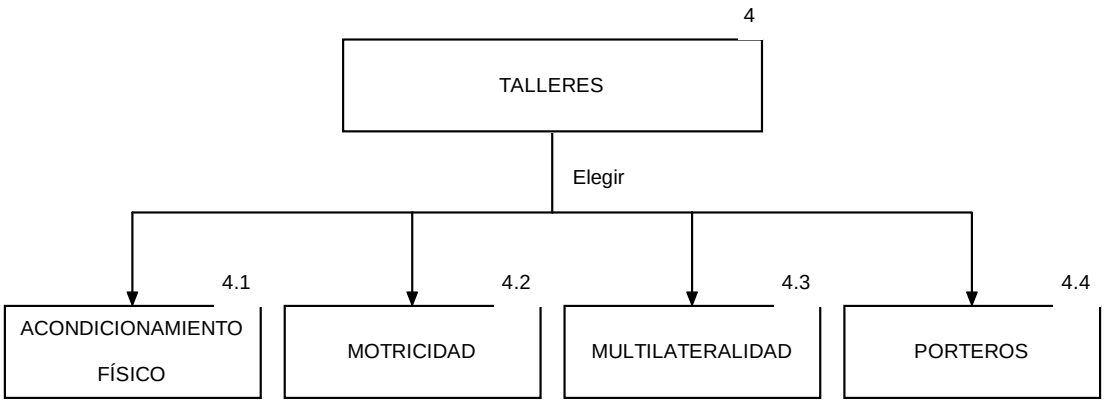


Figura 3.20 Talleres.

Acondicionamiento Físico.

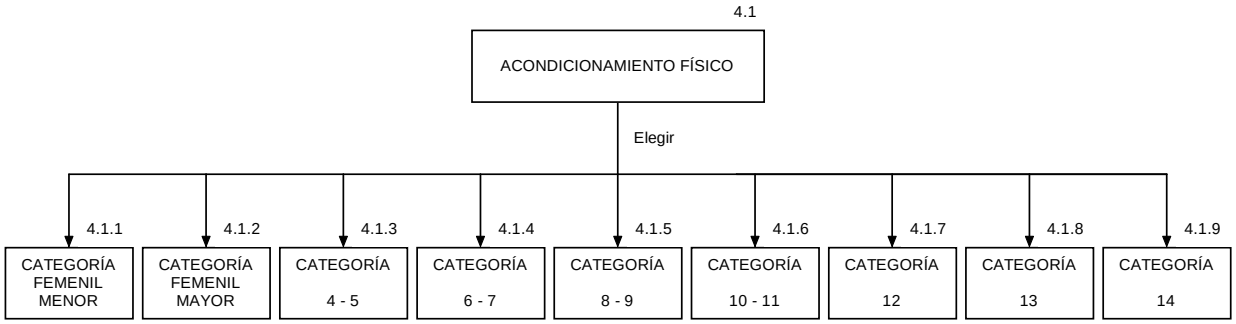


Figura 3.21 Acondicionamiento Físico.

Motricidad.

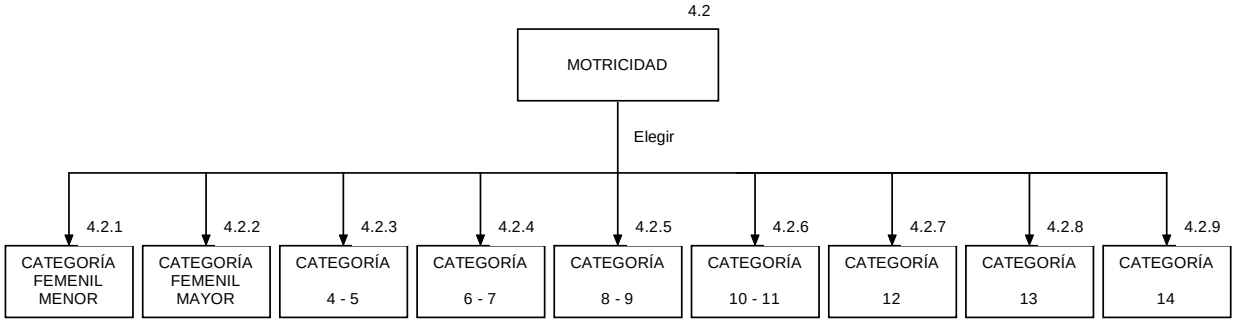


Figura 3.22 Motricidad.

Multilateralidad.

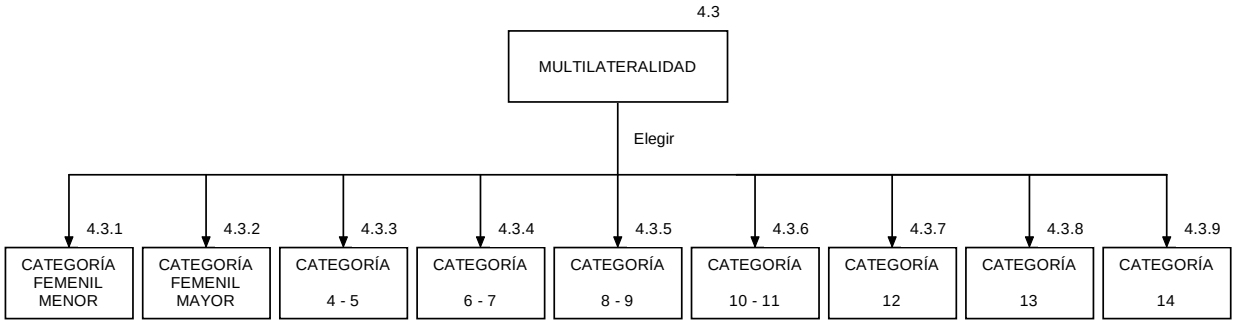


Figura 3.23 Multilateralidad.

Porteros.

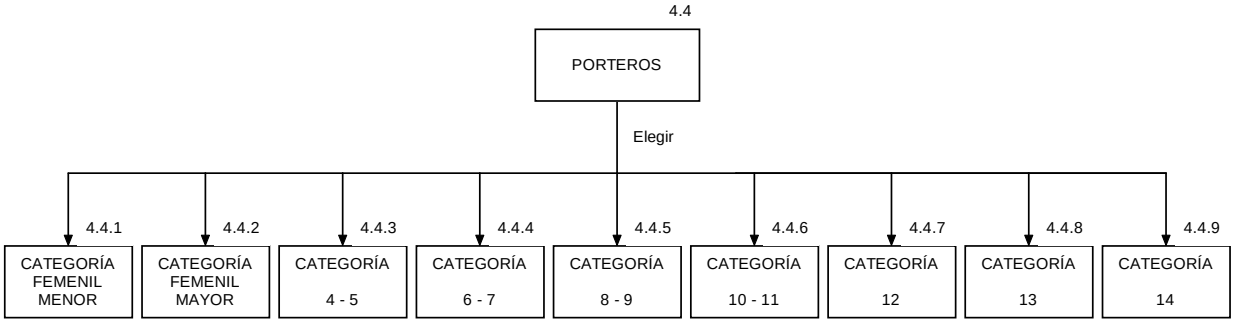


Figura 3.24 Porteros

Reportes.

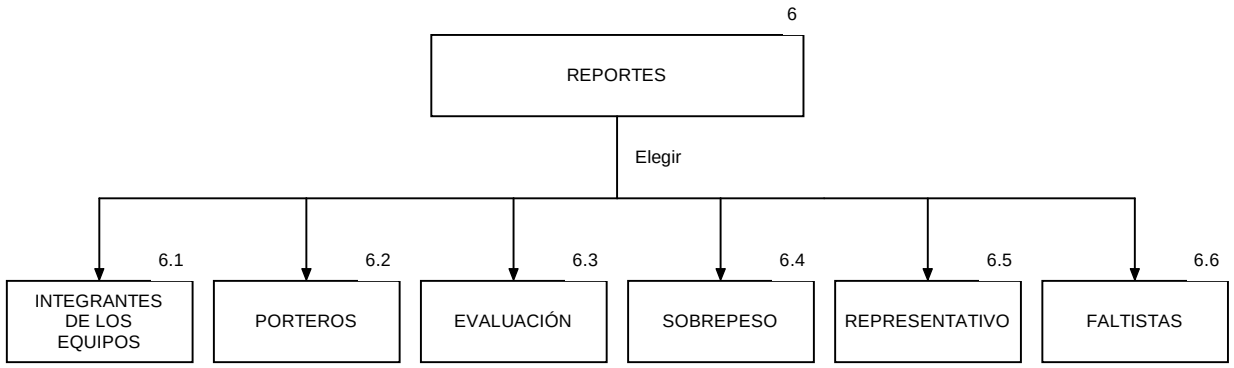


Figura 3.25 Reportes.

Integrantes de los equipos.

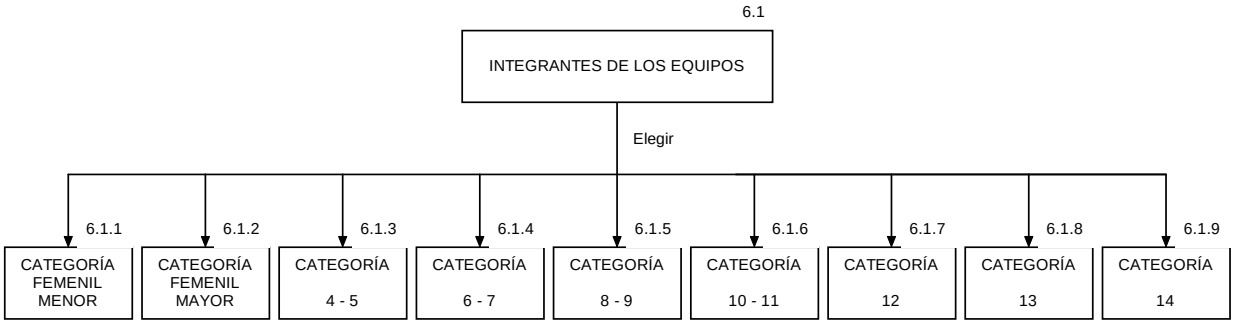


Figura 3.26 Integrantes de los equipos.

Porteros.

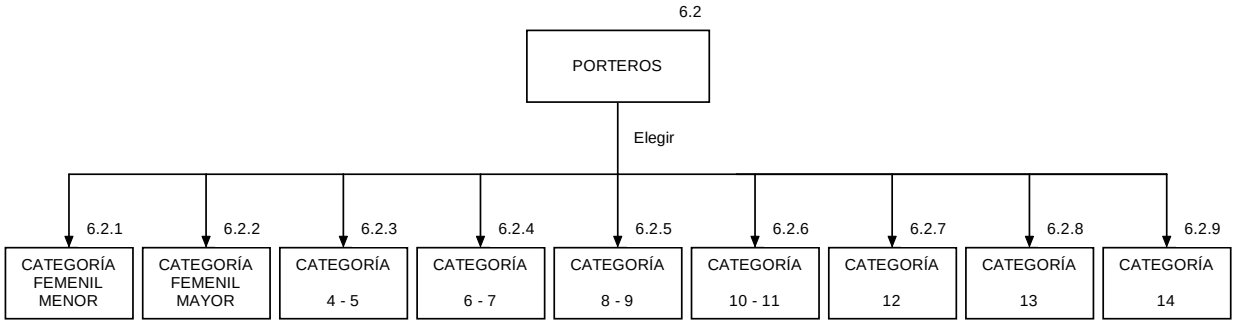


Figura 3.27 Porteros.

Evaluación.

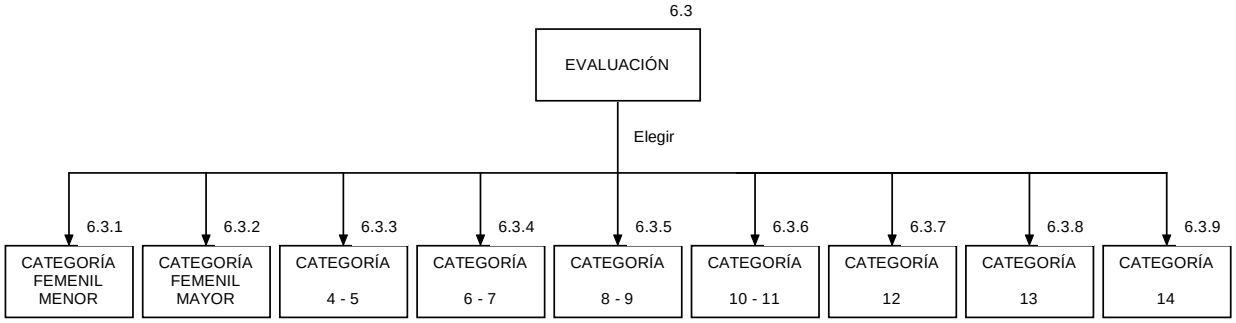


Figura 3.28 Evaluación.

Sobrepeso.

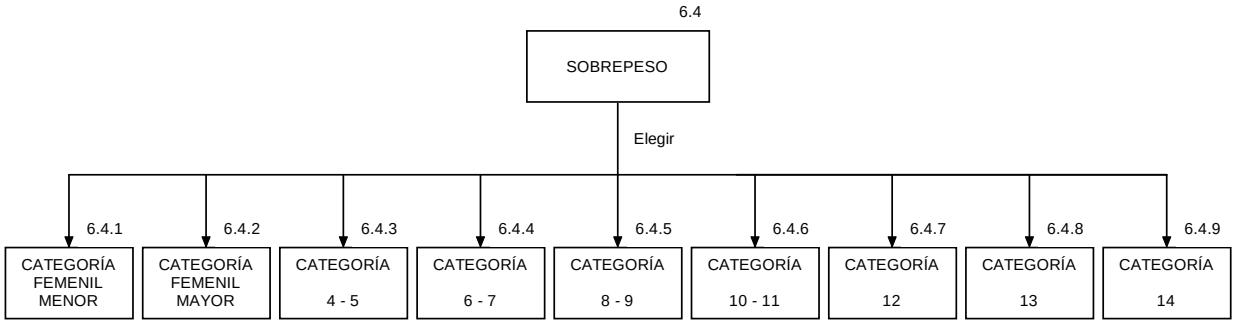


Figura 3.29 Sobrepeso.

Representativo.

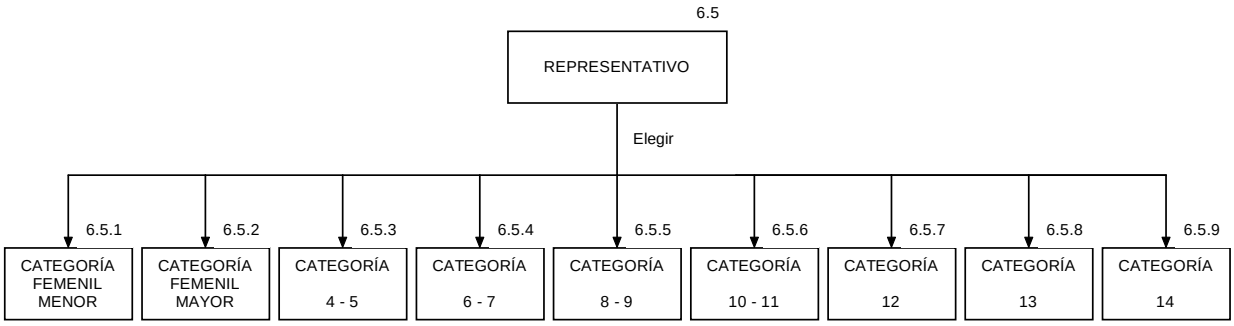


Figura 3.30 Representativo.

Faltistas.

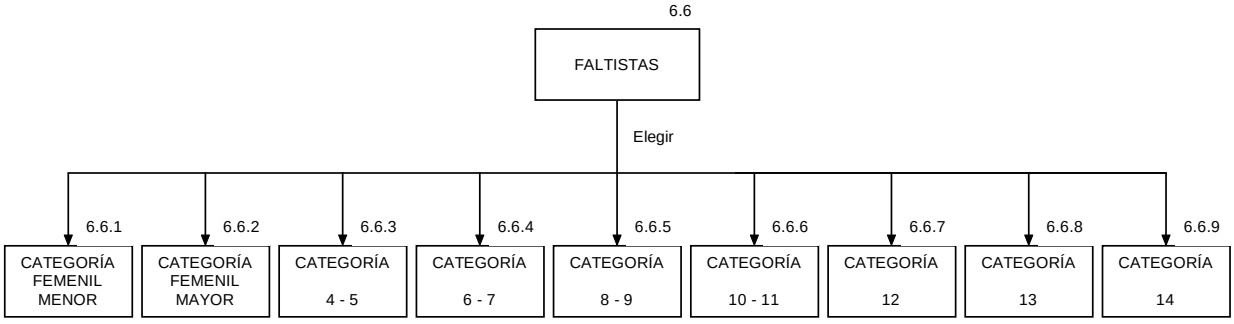


Figura 3.31 Faltistas.

Integrantes de los equipos por categoría.

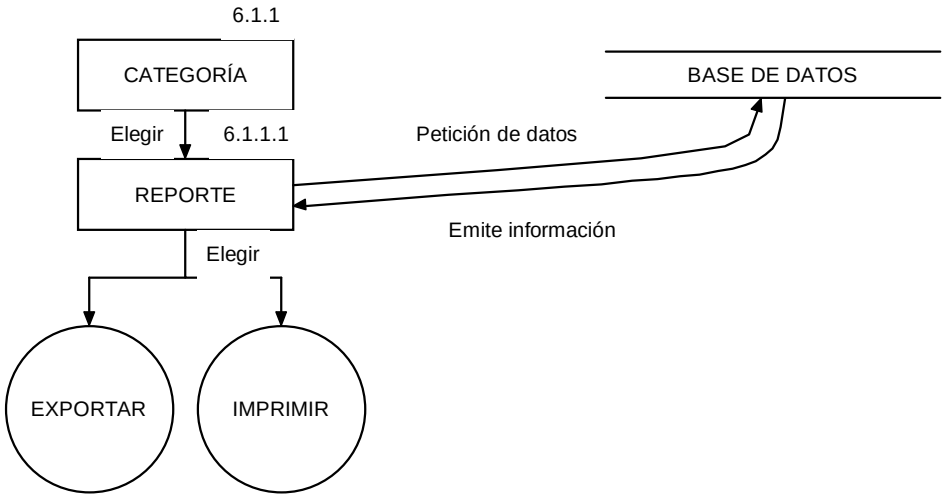


Figura 3.32 Integrantes de los equipos por categoría.

Porteros por categoría.

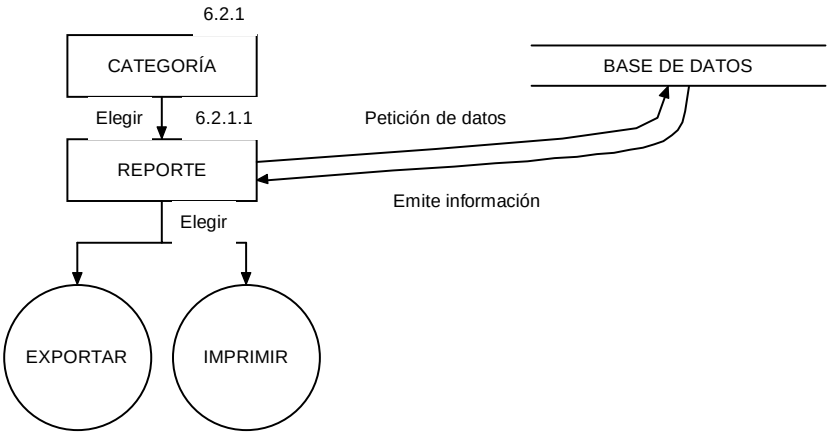


Figura 3.33 Porteros por categoría.

Evaluación por categoría.

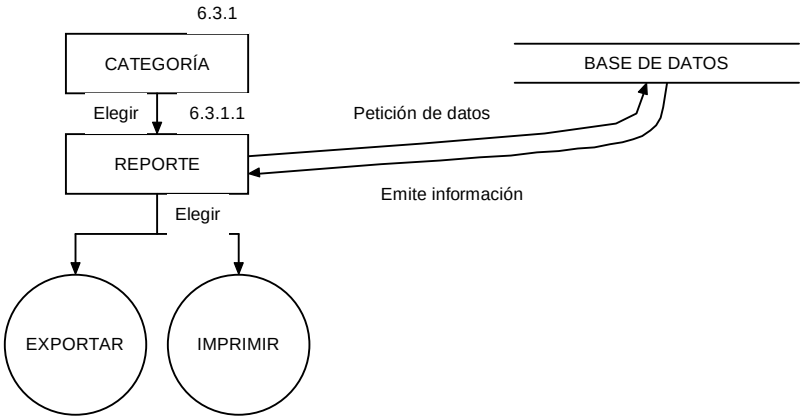


Figura 3.34 Evaluación por categoría.

Sobrepeso por categoría.

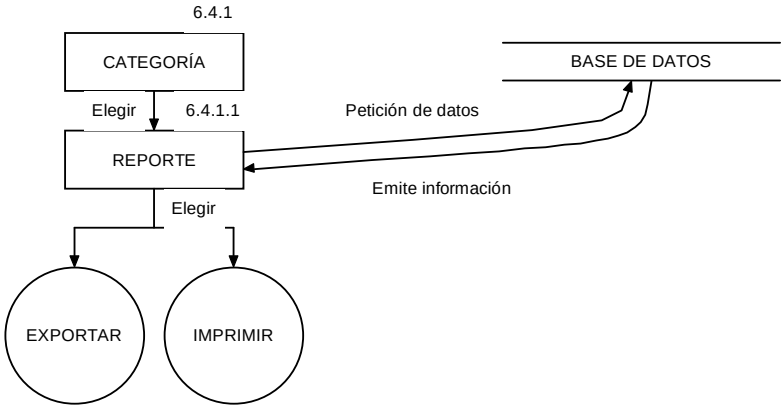


Figura 3.35 Sobrepeso por categoría.

Representativo por categoría.

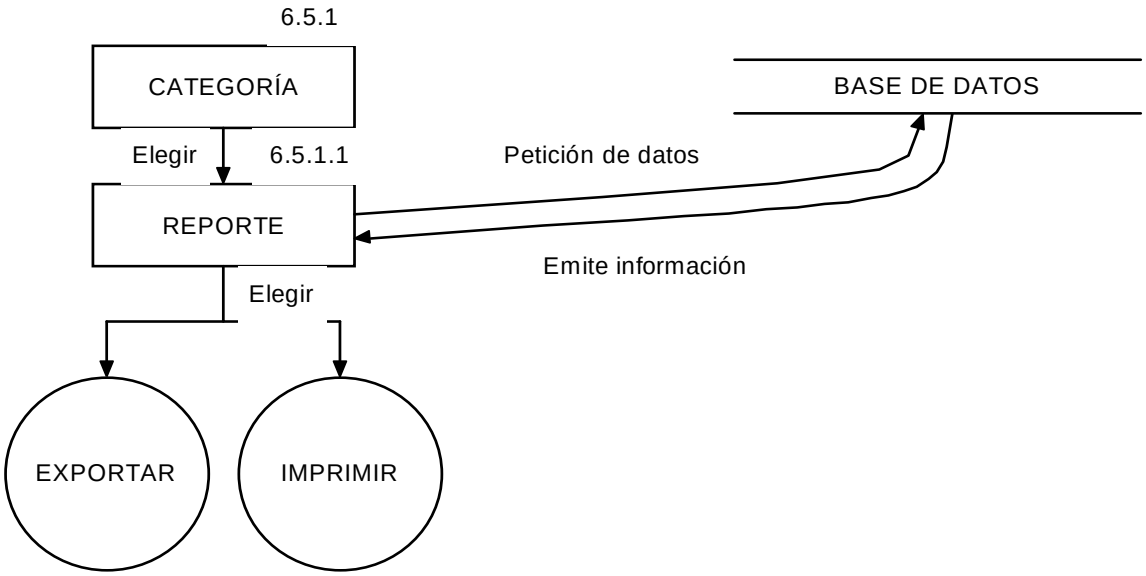


Figura 3.36 Representativo por categoría.

Faltistas por categoría.

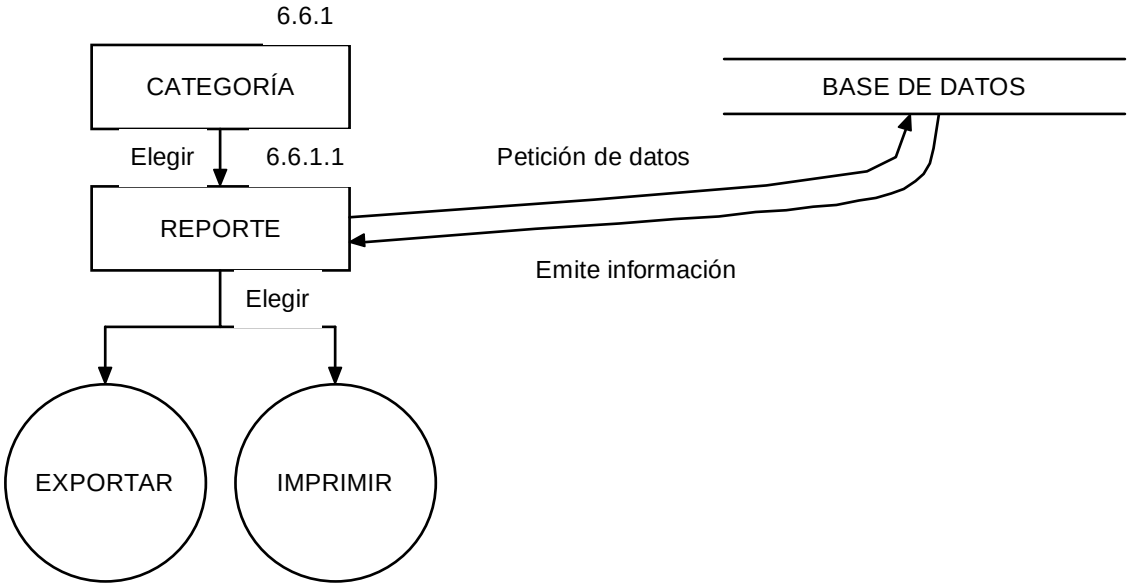


Figura 3.37 Faltistas por categoría.

Tomando en cuenta la primer categoría, se tiene:

Categoría.

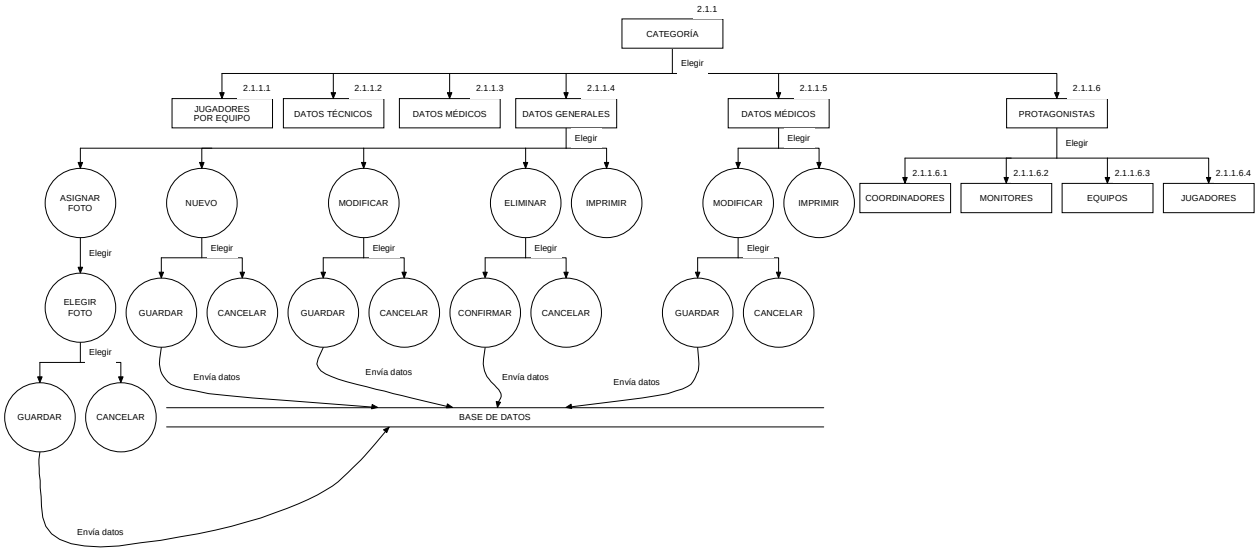


Figura 3.38 Categoría.

Jugadores por equipo.

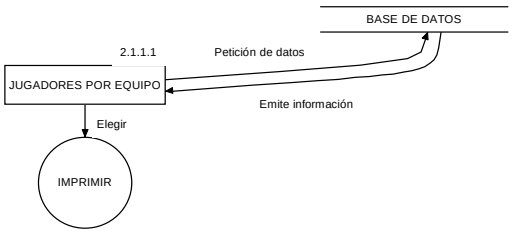


Figura 3.39 Jugadores por equipo.

Datos técnicos.

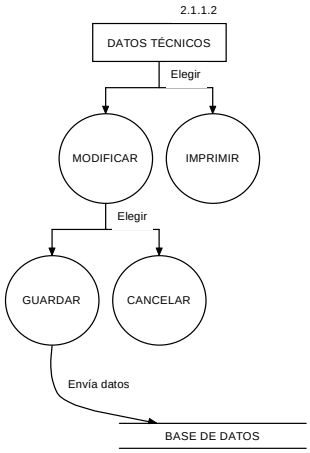


Figura 3.40 Datos técnicos.

Datos Médicos.

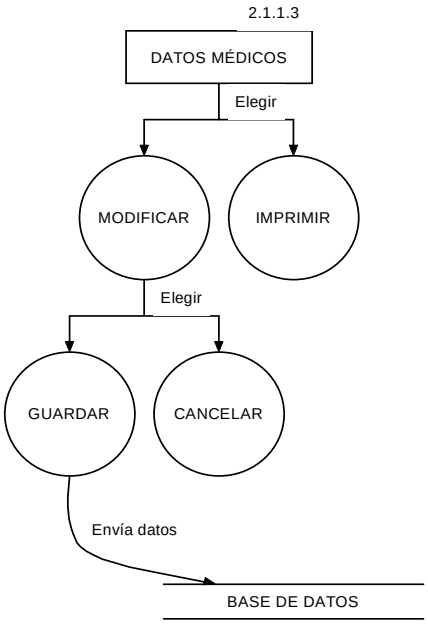


Figura 3.41 Datos Médicos.

Coordinadores.

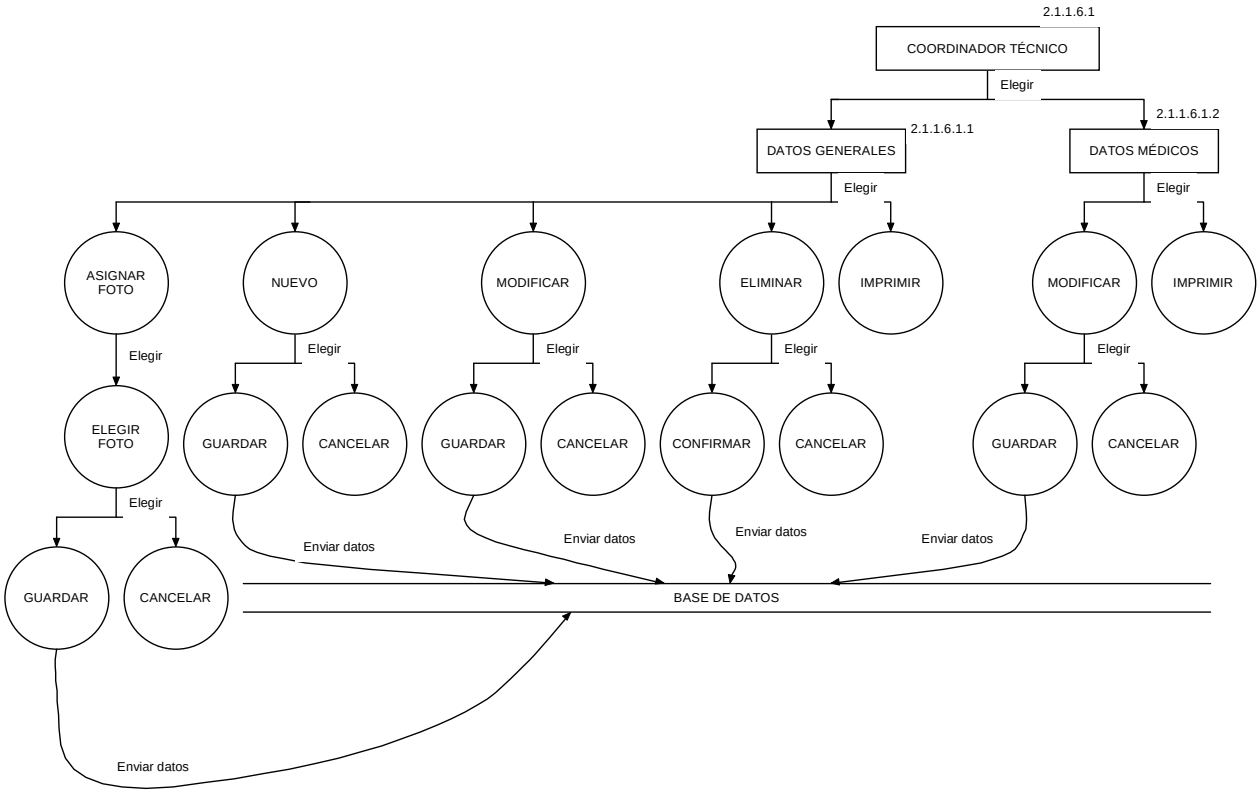


Figura 3.42 Coordinadores.

Monitores.

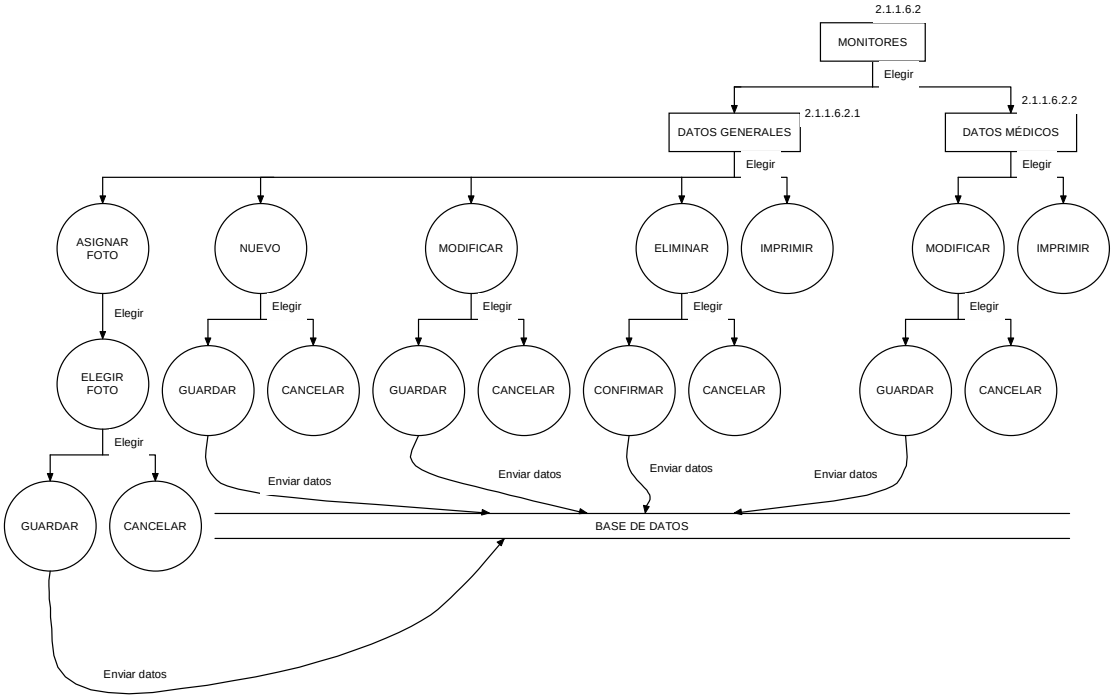


Figura 3.43 Monitores.

Equipos.

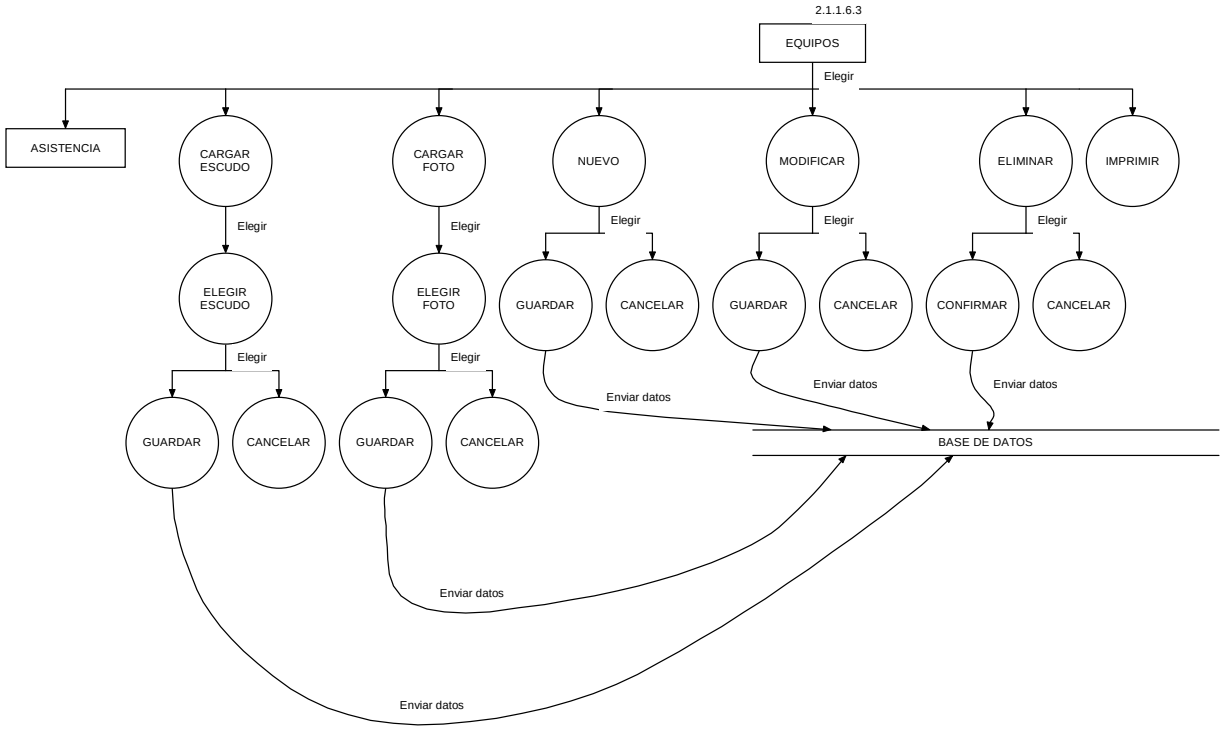


Figura 3.44 Equipos.

Asistencia.

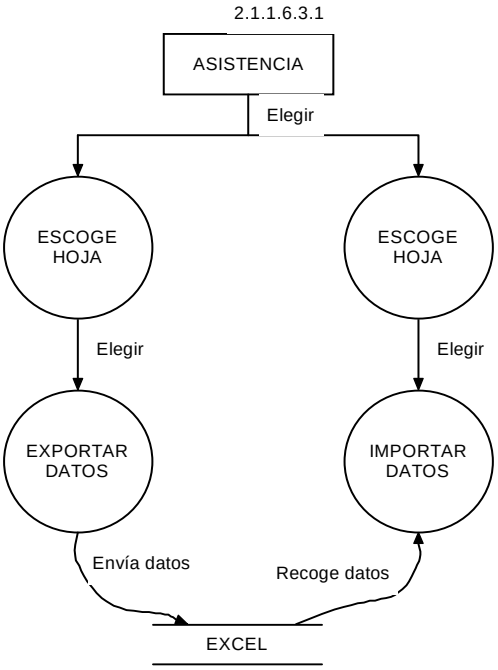


Figura 3.45 Asistencia.

Búsqueda.

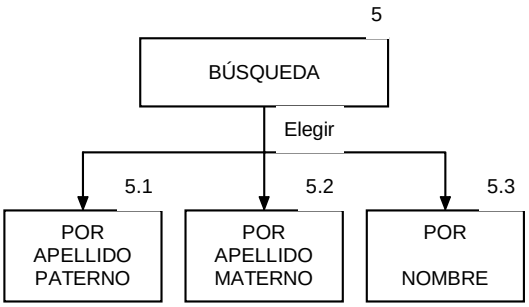


Figura 3.46 Búsqueda.

Catálogos.

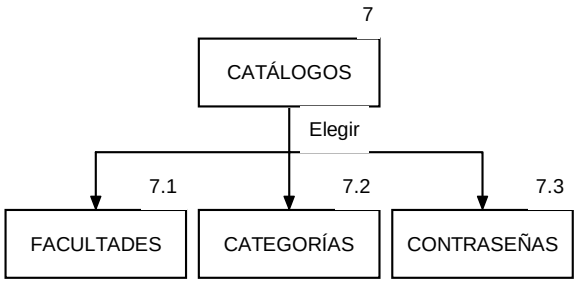


Figura 3.47 Catálogos.

CAPÍTULO 4. Diseño del Sistema.

Conceptos.

Una base de datos relacional es una base de datos que es percibida por el usuario como una colección de relaciones o tablas de 2 dimensiones.

Una base de datos relacional debe tener integridad de datos, sus datos deben ser precisos y consistentes.

LLAVE PRIMARIA. Es la llave candidato elegida para identificar una entidad como única. Una llave primaria (PK) es una columna o grupo de columnas que identifican de manera única a cada renglón en una tabla. Cada tabla debe tener una llave primaria.

Una llave primaria que consta de múltiples columnas se llama llave primaria compuesta. Las columnas de una llave primaria compuesta deben ser únicas en combinación. Las columnas pueden tener duplicados en forma individual, pero en combinación, no se permiten duplicados. Ninguna parte de la llave primaria puede ser nula.

LLAVE FORÁNEA. Una llave foránea (FK) es una columna o combinación de columnas en una tabla, que se refieren a una llave primaria en otra tabla.

Las llaves foráneas son utilizadas para hacer “JOIN” entre tablas.

INTEGRIDAD DE DATOS: Exactitud y consistencia de los datos.

Las restricciones de integridad de datos aseguran que los usuarios realicen únicamente operaciones en las cuales dejarán a la base de datos en un estado correcto y consistente.

Tipo de Restricción	Explicación
Integridad de Entidades	Ninguna parte de la llave primaria puede ser NULA
Integridad Referencial	Una llave foránea debe coincidir con un valor de una llave primaria
Integridad de Columnas	Una columna debe contener sólo valores consistentes con el formato de datos definido para la columna
Integridad definida por el Usuario	Los datos almacenados en la base de datos deben cumplir con las reglas de Pumitas

Un dato es inconsistente si existen múltiples copias de un registro y no todas las copias han sido actualizadas. Una base de datos inconsistente provee información incorrecta o contradictoria a los usuarios.

4.1 Diseño de la Base de datos.

El diseño de base de datos se lleva a cabo por medio de dos actividades:

- Pasar el modelo E-R a tablas.
- Refinar el diseño inicial para producir un diseño completo de la base de datos.

En el diseño se definen las tablas, índices, vistas y espacio de almacenamiento.

Para cada atributo seleccionar un nombre corto pero significativo.

El nombre de las columnas debe ser fácil de identificar en un modelo E-R.

No utilizar palabras reservadas de SQL para nombres de columnas.

Utilizar abreviaciones consistentes que no causen confusión.

4.1.1 Diagrama Entidad-Relación.

Este modelo representa la realidad a través de un esquema gráfico, mediante las relaciones entre las tablas de datos. Este diagrama se centra sólo en los datos, representando una red que existe para el sistema. Donde las entidades son los elementos principales que se identifican en el problema a resolver, se caracterizan por sus componentes denominados atributos, el enlace que determina la unión de las entidades está representado por la relación del modelo siguiente.

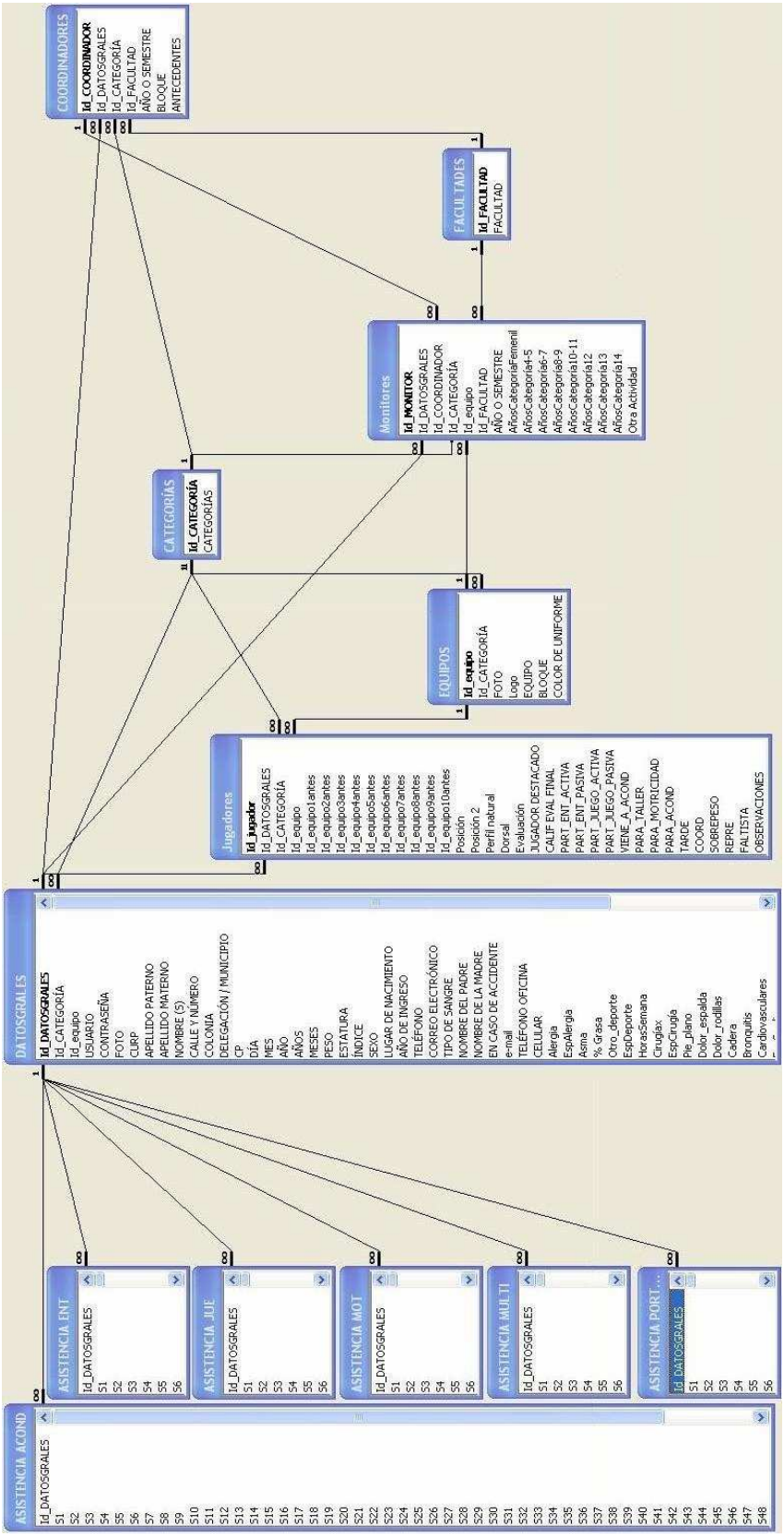


Figura 4.1 Diagrama Entidad-Relación.

4.1.2 Normalización.

Las reglas de normalización están diseñadas para la prevención de anomalías e inconsistencia en los datos, así como en las operaciones realizadas y sus relaciones.

Normalización de tablas.

Categorizar las tablas de acuerdo a su grado de normalización.

Regla de Forma Normal	Descripción
Primera Forma Normal (1FN)	La tabla no puede contener grupos de repetición
Segunda Forma Normal (2FN)	La tabla debe estar en 1FN. Cada columna que no es llave debe ser dependiente de la llave primaria como un todo.
Tercera Forma Normal (3FN)	La tabla debe de estar en 2FN. Una columna que no es llave primaria no debe ser funcionalmente dependiente de otra columna no llave primaria. Cada valor de una llave no primaria debe ser dependiente solamente de la llave y no de ningún otro campo.

La normalización minimiza la redundancia de los datos y ayuda a identificar entidades, relaciones y tablas faltantes. Un dato sin normalizar es redundante. La redundancia de los datos causa problemas de integridad. Las transacciones de actualización y borrado pueden no ser consistentes en todas las copias de los datos causando inconsistencia en la base de datos.

La 3FN es un objetivo normalmente aceptado para un diseño de base de datos para eliminar la redundancia.

Las formas normales mayores no son ampliamente utilizadas.

CONVERSIÓN A LA PRIMERA FORMA NORMAL.

- Eliminar los grupos de repetición.
- Crear una nueva tabla con la PK de la tabla base y el grupo de repetición.

CONVERSIÓN A LA SEGUNDA FORMA NORMAL.

- Eliminar cualquier columna no llave que no dependa de la llave primaria completa de la tabla.

- Determinar cuáles columnas que no son llave no dependen de la llave primaria completa de la tabla.
- Eliminar esas columnas de la tabla base.
- Crear una segunda tabla con esas columnas y la(s) columna(s) de la PK de la cual dependen.
- Cualquier tabla con una llave primaria de una sola columna está automáticamente en la 2FN.

CONVERSIÓN A LA TERCERA FORMA NORMAL.

- Eliminar cualquier columna que sea dependiente de otra columna no llave.
- Determinar qué columnas son dependientes de otra columna no llave.
- Eliminar esas columnas de la tabla base.
- Crear una segunda tabla con esas columnas y con la columna no llave de la cual son dependientes.

4.1.3 Diccionario de Datos.

El diccionario de datos es un listado organizado de todos los elementos de datos que son pertinentes para el sistema, con definiciones precisas que permiten que el usuario y el analista del sistema tengan una misma comprensión de las entradas y de las salidas.

Entidad Datos Generales.

En esta entidad se tiene almacenada toda la información referente a los datos generales de los integrantes de Pumitas, así como de los usuarios del sistema.

Llave	Nombre	Tipo	Tamaño	Observaciones
PK	Id_DATOSGRALES	Autonumérico	Entero largo	Identificador de sus datos generales
FK	Id_CATEGORÍA	Número	Entero largo	Identificador de la categoría a la que pertenece
FK	Id_equipo	Número	Byte	Identificador del equipo al que pertenece
	USUARIO	Texto	15	Nombre de usuario para entrar a Pumidata 1.0
	CONTRASEÑA	Texto	15	Contraseña del usuario para entrar a Pumidata 1.0
	Foto	Texto	255	Ruta de la ubicación de la foto del integrante en cuestión
	CURP	Texto	20	Clave Única de Registro de Población
	APELLIDO PATERNO	Texto	20	Apellido Paterno
	APELLIDO MATERNO	Texto	20	Apellido Materno
	NOMBRE (S)	Texto	30	Nombre o nombres
	CALLE Y NÚMERO	Texto	50	Calle y número de su dirección
	COLONIA	Texto	30	Colonia
	DELEGACIÓN / MUNICIPIO	Texto	20	Delegación o municipio
	CP	Número	Entero largo	Código Postal
	DÍA	Número	Byte	Día de su fecha de nacimiento

	MES	Número	Byte	Mes de su fecha de nacimiento
	AÑO	Número	Entero	Año de su fecha de nacimiento
	AÑOS	Número	Byte	Años a la fecha actual
	MESES	Número	Byte	Meses transcurridos de su último cumpleaños a la fecha actual
	PESO	Número	Byte	Peso expresado en kilos
	ESTATURA	Número	Byte	Estatura expresada en centímetros
	ÍNDICE	Número	Entero	Índice que nos determina a qué categoría se dirige
	SEXO	Texto	10	Sexo: masculino o femenino
	LUGAR DE NACIMIENTO	Texto	30	Lugar de nacimiento
	AÑO DE INGRESO	Número	Entero	Año de ingreso a Pumitas
	TELÉFONO	Número	Entero largo	Teléfono particular
	CORREO ELECTRÓNICO	Texto	30	Correo electrónico
	TIPO DE SANGRE	Texto	20	Tipo de sangre
	NOMBRE DEL PADRE	Texto	50	Nombre completo del padre
	NOMBRE DE LA MADRE	Texto	50	Nombre completo de la madre
	EN CASO DE ACCIDENTE	Texto	50	Teléfono a dónde comunicarse en caso de accidente
	e-mail	Texto	50	Correo electrónico para comunicarse en caso de accidente
	TELÉFONO OFICINA	Número	Entero largo	Teléfono de oficina para comunicarse en caso de accidente
	CELULAR	Número	Doble	Teléfono celular para comunicarse en caso de accidente
	Alergia	Sí/No		Saber si tiene alergia(s) o no
	EspAlergia	Texto	255	Si tiene alergia(s), especificar
	Asma	Sí/No		Saber si tiene asma o no
	% Grasa	Texto	3	Porcentaje de grasa corporal
	Otro_deporte	Sí/No		Saber si practica otro(s) deporte(s) organizado(s) o no
	EspDeporte	Texto	255	Si practica otro(s) deporte(s), especificar
	HorasSemana	Número	Byte	Horas a la semana que practica otro(s) deporte(s)
	Cirugía	Sí/No		Saber si le han practicado cirugías o no
	EspCirugía	Texto	255	Si le han practicado cirugías, especificar
	Pie_planp	Sí/No		Saber si tiene pie plano o no
	Dolor_espalda	Sí/No		Saber si tiene pie dolor de espalda o no
	Dolor_rodillas	Sí/No		Saber si tiene pie dolor en rodillas o no
	Cadera	Sí/No		Saber si tiene pie molestias en la cadera o no
	Bronquitis	Sí/No		Saber si tiene bronquitis o no
	Cardiovasculares	Sí/No		Saber si tiene antecedentes cardiovasculares o no
	EspCardio	Texto	255	Si tiene antecedentes cardiovasculares, especificar
	Otros	Sí/No		Saber si tiene otro antecedentes médico de importancia o no
	EspOtros	Texto	255	Si tiene otro antecedente médico, especificar
	Esguinces	Sí/No		Saber si ha tenido esguinces (torceduras graves) o no
	EspEsguinces	Texto	255	Si ha tenido esguinces, especificar
	Lesión	Sí/No		Saber si ha tenido lesiones musculares o no
	EspLesión	Texto	255	Si ha tenido lesiones musculares, especificar
	Fracturas	Sí/No		Saber si ha tenido fracturas o no
	EspFracturas	Texto	255	Si ha tenido fracturas, especificar
	Fecha	Fecha/Hora		Fecha de atención de la consulta médica
	Diagnóstico	Texto	255	Diagnóstico de la lesión o padecimiento
	Manejo inicial	Texto	255	Manejo inicial para curar la molestia o lesión
	Evolución	Memo		Seguimiento de la lesión o molestia

Entidad Monitores.

En esta entidad se tiene almacenada toda la información referente a los monitores de Pumitas.

Llave	Nombre	Tipo	Tamaño	Descripción
PK	Id_MONITOR	Autonumérico	Entero largo	Identificador único del monitor en cuestión
FK	Id_DATOSGRALES	Número	Entero largo	Identificador de sus datos generales
FK	Id_COORDINADOR	Número	Entero largo	Identificador del coordinador asignado
FK	Id_CATEGORÍA	Número	Byte	Identificador de la categoría a la que pertenece
FK	Id_equipo	Número	Entero largo	Identificador del equipo al que pertenece
FK	Id_FACULTAD	Número	Entero largo	Identificador de la facultad en la que estudia o estudió
	AÑO O SEMESTRE	Texto	30	Año o semestre en curso de su carrera
	AñosCategoríaFemenil	Número	Byte	Años que ha participado en la Categoría Femenil
	AñosCategoría4-5	Número	Byte	Años que ha participado en la Categoría 4-5
	AñosCategoría6-7	Número	Byte	Años que ha participado en la Categoría 6-7
	AñosCategoría8-9	Número	Byte	Años que ha participado en la Categoría 8-9
	AñosCategoría10-11	Número	Byte	Años que ha participado en la Categoría 10-11
	AñosCategoría12	Número	Byte	Años que ha participado en la Categoría 12
	AñosCategoría13	Número	Byte	Años que ha participado en la Categoría 13
	AñosCategoría14	Número	Byte	Años que ha participado en la Categoría 14
	Otra Actividad	Texto	50	Alguna otra actividad que desempeñe dentro de Pumitas

Entidad Jugadores.

En esta entidad se tiene almacenada toda la información referente a los jugadores de Pumitas.

Llave	Nombre	Tipo	Tamaño	Descripción
PK	Id_jugador	Autonumérico	Entero largo	Identificador del jugador en cuestión
FK	Id_DATOSGRALES	Número	Entero largo	Identificador de sus datos generales
FK	Id_CATEGORÍA	Número	Entero largo	Identificador de la categoría a la que pertenece
FK	Id_equipo	Número	Entero largo	Identificador del equipo al que pertenece
FK	Id_equipo1antes	Número	Byte	Identificador del equipo en el que se encontraba 1 año antes
FK	Id_equipo2antes	Número	Byte	Identificador del equipo en el que se encontraba 2 años antes
FK	Id_equipo3antes	Número	Byte	Identificador del equipo en el que se encontraba 3 años antes
FK	Id_equipo4antes	Número	Byte	Identificador del equipo en el que se encontraba 4 años antes
FK	Id_equipo5antes	Número	Byte	Identificador del equipo en el que se encontraba 5 años antes
FK	Id_equipo6antes	Número	Byte	Identificador del equipo en el que se encontraba 6 años antes
FK	Id_equipo7antes	Número	Byte	Identificador del equipo en el que se encontraba 7 años antes
FK	Id_equipo8antes	Número	Byte	Identificador del equipo en el que se encontraba 8 años antes
FK	Id_equipo9antes	Número	Byte	Identificador del equipo en el que se encontraba 9 años antes
FK	Id_equipo10antes	Número	Byte	Identificador del equipo en el que se encontraba 10 años antes
	Posición	Texto	20	Posición dentro del terreno de juego
	Posición 2	Texto	20	Segunda posición que desempeña dentro del terreno de juego
	Perfil natural	Texto	15	Perfil natural: derecho o izquierdo
	Dorsal	Número	Byte	Dorsal o número que porta durante el juego

	Evaluación	Texto	3	Evaluación que refleja el nivel de cualidades técnicas, tácticas
	JUGADOR DESTACADO	Sí/No		Saber si es jugador destacado o no
	CALIF EVAL FINAL	Número	Decimal	Calificación final resultado de las evaluaciones
	PART_ENT_ACTIVA	Sí/No		Saber si participa activamente en los entrenamientos o no
	PART_ENT_PASIVA	Sí/No		Saber si participa pasivamente en los entrenamientos o no
	PART_JUEGO_ACTIVA	Sí/No		Saber si participa activamente en los juegos o no
	PART_JUEGO_PASIVA	Sí/No		Saber si participa pasivamente en los juegos o no
	VIENE_A_ACOND	Sí/No		Saber si se presenta al taller de acondicionamiento o no
	PARA_ACOND	Sí/No		Saber si necesita el taller de acondicionamiento físico o no
	TARDE	Sí/No		Saber si asiste a la escuela por la tarde o no
	COORD	Sí/No		Saber si necesita mejorar su coordinación o no
	SOBREPESO	Sí/No		Saber si tiene sobrepeso o no
	REPRE	Sí/No		Saber si es integrante del representativo o no
	FALTISTA	Sí/No		Saber si es faltista o no
	OBSERVACIONES	Memo		Observaciones generales

Entidad EQUIPOS.

En esta entidad se tiene almacenada la información de los equipos Pumitas.

Llave	Nombre	Tipo	Tamaño	Descripción
PK	Id_equipo	Autonumérico	Entero largo	Identificador del equipo
FK	Id_CATEGORÍA	Número	Entero largo	Identificador de la categoría a la que corresponde el equipo
	FOTO	Texto	255	Ruta que describe la localización de la foto del equipo
	Logo	Texto	255	Ruta que describe la localización del logotipo del equipo
	EQUIPO	Texto	30	Equipo
	BLOQUE	Número	Byte	Número de Bloque que tiene en la coordinación técnica
	COLOR DE UNIFORME	Texto	20	Color de uniforme del equipo

Entidad COORDINADORES.

En esta entidad se tiene almacenada la información concerniente a los coordinadores de Pumitas.

Llave	Nombre	Tipo	Tamaño	Descripción
PK	Id_COORDINADOR	Autonumérico	Entero largo	Identificador del coordinador
FK	Id_DATOSGRALES	Número	Entero largo	Identificador de sus datos generales
FK	Id_CATEGORÍA	Número	Entero largo	Identificador de la categoría en la cual labora
FK	Id_FACULTAD	Número	Entero largo	Identificador de la facultad en la que realiza o realizó sus estudios
	AÑO O SEMESTRE	Texto	50	Año o semestre que cursa
	BLOQUE	Número	Byte	Número de bloque asignado
	ANTECEDENTES	Memo		Antecedentes dentro de Pumitas

Entidad CATEGORÍAS.

En esta entidad se tiene almacenada la información de las distintas categorías de Pumitas.

Llave	Nombre	Tipo	Tamaño	Descripción
PK	Id_CATEGORÍA	Autonumérico	Entero largo	Identificador único de la categoría
	CATEGORÍAS	Texto	50	Categorías existentes dentro de Pumitas

Entidad FACULTADES.

En esta entidad se tiene almacenada la información de las distintas facultades de la UNAM.

Llave	Nombre	Tipo	Tamaño	Descripción
PK	Id_FACULTAD	Autonumérico	Entero largo	Identificador de la facultad
	FACULTAD	Texto	40	Nombre de la facultad

Entidad ASISTENCIA ENT, ASISTENCIA JUE, ASISTENCIA ACOND, ASISTENCIA MOT, ASISTENCIA MULTI Y ASISTENCIA PORTEROS.

En estas entidades se tiene almacenada la asistencia de cada uno de los integrantes de Pumitas durante la temporada y todas tienen los mismos campos. Así:

ASISTENCIA ENT: Asistencia a entrenamientos de los integrantes de un equipo en particular.
ASISTENCIA JUE: Asistencia a juegos de los integrantes de un equipo en particular.
ASISTENCIA ACOND: Asistencia al taller de Acondicionamiento Físico.
ASISTENCIA MOT: Asistencia al taller de Motricidad.
ASISTENCIA MULTI: Asistencia al taller de Multilateralidad.
ASISTENCIA PORTEROS: Asistencia al taller de Porteros.

Llave	Nombre	Tipo	Tamaño	Observaciones
PK	Id_DATOSGRALES	Número	Entero largo	Identificador de datos generales
	S1	Texto	2	Asistencia o falta de la semana 1
	S2	Texto	2	Asistencia o falta de la semana 2
	S3	Texto	2	Asistencia o falta de la semana 3
	S4	Texto	2	Asistencia o falta de la semana 4
	S5	Texto	2	Asistencia o falta de la semana 5
	S6	Texto	2	Asistencia o falta de la semana 6
	S7	Texto	2	Asistencia o falta de la semana 7
	S8	Texto	2	Asistencia o falta de la semana 8
	S9	Texto	2	Asistencia o falta de la semana 9
	S10	Texto	2	Asistencia o falta de la semana 10

	S11	Texto	2	Asistencia o falta de la semana 11
	S12	Texto	2	Asistencia o falta de la semana 12
	S13	Texto	2	Asistencia o falta de la semana 13
	S14	Texto	2	Asistencia o falta de la semana 14
	S15	Texto	2	Asistencia o falta de la semana 15
	S16	Texto	2	Asistencia o falta de la semana 16
	S17	Texto	2	Asistencia o falta de la semana 17
	S18	Texto	2	Asistencia o falta de la semana 18
	S19	Texto	2	Asistencia o falta de la semana 19
	S20	Texto	2	Asistencia o falta de la semana 20
	S21	Texto	2	Asistencia o falta de la semana 21
	S22	Texto	2	Asistencia o falta de la semana 22
	S23	Texto	2	Asistencia o falta de la semana 23
	S24	Texto	2	Asistencia o falta de la semana 24
	S25	Texto	2	Asistencia o falta de la semana 25
	S26	Texto	2	Asistencia o falta de la semana 26
	S27	Texto	2	Asistencia o falta de la semana 27
	S28	Texto	2	Asistencia o falta de la semana 28
	S29	Texto	2	Asistencia o falta de la semana 29
	S30	Texto	2	Asistencia o falta de la semana 30
	S31	Texto	2	Asistencia o falta de la semana 31
	S32	Texto	2	Asistencia o falta de la semana 32
	S33	Texto	2	Asistencia o falta de la semana 33
	S34	Texto	2	Asistencia o falta de la semana 34
	S35	Texto	2	Asistencia o falta de la semana 35
	S36	Texto	2	Asistencia o falta de la semana 36
	S37	Texto	2	Asistencia o falta de la semana 37
	S38	Texto	2	Asistencia o falta de la semana 38
	S39	Texto	2	Asistencia o falta de la semana 39
	S40	Texto	2	Asistencia o falta de la semana 40
	S41	Texto	2	Asistencia o falta de la semana 41
	S42	Texto	2	Asistencia o falta de la semana 42
	S43	Texto	2	Asistencia o falta de la semana 43
	S44	Texto	2	Asistencia o falta de la semana 44
	S45	Texto	2	Asistencia o falta de la semana 45
	S46	Texto	2	Asistencia o falta de la semana 46
	S47	Texto	2	Asistencia o falta de la semana 47
	S48	Texto	2	Asistencia o falta de la semana 48
	S49	Texto	2	Asistencia o falta de la semana 49
	S50	Texto	2	Asistencia o falta de la semana 50
	S51	Texto	2	Asistencia o falta de la semana 51
	S52	Texto	2	Asistencia o falta de la semana 52
	S53	Texto	2	Asistencia o falta de la semana 53
	Tot	Texto	2	Total de asistencias en la temporada
	Porc	Texto	3	Porcentaje de asistencias en la temporada

4.2 Selección de Elementos y Herramientas de Desarrollo.

En general, una herramienta es cualquier dispositivo que, cuando se emplea en forma adecuada, mejora el desempeño de una tarea, tal como el desarrollo de sistemas de información basados en computadora. En general las herramientas se agrupan en las siguientes categoría: análisis, diseño y desarrollo.

Herramientas para análisis.

Estas herramientas ayudan a documentar un sistema existente, ya sea éste manual o automatizado, y a determinar los requerimientos de una nueva aplicación.

- Herramientas de especificación.
- Herramientas para presentación.

Herramientas para diseño.

Las herramientas para diseño apoyan el proceso de formular las características que el sistema debe tener para satisfacer los requerimientos detectados durante las actividades de análisis.

- Herramientas de especificación.
- Herramientas para presentación.

Herramientas para el desarrollo.

Estas herramientas ayudan al análisis a trasladar los diseños en aplicaciones funcionales.

- Herramientas para ingeniería de software.
- Herramientas para pruebas.

Para desarrollar un sistema de calidad, conviene utilizar herramientas de desarrollo que permitan lograr esa meta. La programación estructurada es una metodología de organización y programación de sistemas que mantiene una lógica ordenada, simple y directa, para facilitar su entendimiento y modificación. Con esto se logra un proceso coherente y disciplinado, disminuyendo el número de errores y tiempo invertido.

A esta metodología se le denomina así porque hace uso de estructuras lógicas de programación. Para el diseño de cualquier programa son suficientes tres estructuras: secuencia, iteración y selección.

Con la programación estructurada se pretende dividir el proyecto original en una serie de tareas o pequeños programas, que al ser unidos nuevamente, resuelven el problema planteado en el proyecto original. La programación estructurada es un caso especial de la programación modular, el diseño de un programa estructurado se realiza construyendo bloques pequeños, que pueden ser codificados fácilmente.

Se cuenta también con las herramientas de programación, que permiten a los desarrolladores de software editar, interpretar, compilar, probar y depurar, todos los programas fuentes de una aplicación o un sistema además de detectar errores y hacer comparaciones.

Lenguaje de programación orientada a eventos.

El lenguaje orientado a eventos brinda al usuario al usuario la posibilidad de construir sus propias aplicaciones utilizando interfaces gráficas sobre la base de la ocurrencia de eventos.

Para soportar este tipo de desarrollo interactúan dos tipos de herramientas; por un lado la que permite realizar diseños gráficos y por el otro un lenguaje de alto nivel que permite codificar los eventos. Con dichas herramientas es posible desarrollar cualquier tipo de aplicaciones basadas en el entorno Windows.

Visual Basic es un lenguaje de programación visual, también llamado lenguaje de 4ª Generación. Esto quiere decir que un gran número de tareas se realizan sin escribir código, simplemente con operaciones gráficas realizadas con el ratón sobre la pantalla; utiliza objetos con propiedades y métodos, al ejecutar la aplicación lo única que hace es quedarse a la espera de las acciones del usuario, que en este caso son llamados eventos; por tanto es un lenguaje orientado a eventos. La programación orientada a eventos está dirigida a la realización de programas para Windows, pudiendo incorporar todos los elementos de este entorno informático: ventanas, botones, cajas de diálogo y de texto, botones de opción y de selección, barras de desplazamiento, gráficos, menús, etc. Prácticamente todos los elementos de interacción con el usuario de los que dispone Windows pueden ser programados en Visual Basic.

Eventos.

Son las acciones del usuario sobre el programa. Son eventos típicos: el click sobre un botón, el hacer doble click sobre el nombre de un fichero para abrirlo, el arrastrar un icono, el pulsar una tecla o combinación de teclas, el elegir una opción de un menú, el escribir en una caja de texto, o simplemente mover el ratón, etc. Cada vez que se produce un evento sobre un determinado tipo de control, arranca

una determinada función o procedimiento que realiza la acción programada por el usuario para ese evento concreto.

Además de los eventos, la mayor parte de los objetos, como los formularios y los controles, son suministrados con propiedades y métodos.

Propiedades.

Una propiedad es una asignación que describe algo sobre un objeto como un formulario. Dependiendo de la propiedad, se le puede modificar en tiempo de diseño usando la ventana de Propiedades y/o en tiempo de ejecución al programar.

Métodos.

Son funciones que también son llamadas desde programa, pero a diferencia de los procedimientos no son programadas por el usuario, sino que vienen ya preprogramadas con el lenguaje. Los métodos realizan tareas típicas, previsibles y comunes para todas las aplicaciones, de ahí que vengan con el lenguaje y que se libere al usuario de la tarea de programarlos. Cada tipo de objeto o de control tiene sus propios métodos.

Visual Basic provee las herramientas necesarias para elaborar el sistema en donde el desarrollador de software puede editar, interpretar, compilar, probar y depurar los programas fuentes y detectar errores en un solo paquete; además, contiene instrucciones que permiten administrar una o más bases de datos y permite manipular los datos de la base de datos a fin de presentar los datos como información útil para el cliente.

Bases de datos.

Algunas opciones dentro de los manejadores de bases de datos más comerciales tenemos:

SQL Server.

Escalabilidad. Se adapta a las necesidades de la empresa, soportando desde unos pocos usuarios a varios miles.

Gestión. Con una completa interfaz gráfica que reduce la complejidad innecesaria de las tareas de administración y gestión de la base de datos.

Cuenta con un optimizador de consultas, el cual analiza las sentencias SQL que el usuario introduce y si la consulta que el usuario está realizando no es la óptima, genera una nueva consulta que de manera más eficiente obtenga los datos requeridos.

Maneja datos distribuidos, puede hacer llamadas a procedimientos remotos servidor-servidor.

Para mejorar la seguridad y facilitar la administración de la base de datos, sólo maneja un usuario tanto para la red como para la base de datos.

Microsoft Access.

La base de datos es un archivo lógico que puede contener uno o varios archivos físicos, en los cuales se almacena la información que genera un sistema, gestionando y organizando los datos. Access es un manejador de bases de datos (DBM por sus siglas en inglés) que facilita las funciones de:

- Crear y organizar la base de datos.
- Establecer y mantener las trayectorias de acceso a la base de datos de tal forma que los datos puedan ser consultados rápidamente.
- Manejar los datos de acuerdo a las peticiones de los usuarios.
- Registrar el uso de las bases de datos.
- Interacción con el manejador de archivos. Así, el manejador de base de datos es el responsable del verdadero almacenamiento de datos.
- Respaldo y recuperación. Cuenta con mecanismos implantados que permiten la recuperación fácilmente de los datos en caso de ocurrir fallas en el sistema de base de datos.
- Control de concurrencia. Consiste en controlar la interacción entre los usuarios concurrentes para no afectar la inconsistencia de datos.
- Seguridad e integridad. Consiste en contar con mecanismos que permitan el control de la consistencia de los datos evitando que éstos se vean perjudicados por cambios no autorizados o previstos.

Access trabaja en entorno Windows; contiene herramientas de diseño y programación y cuenta con las siguientes herramientas:

- **Tablas**: unidad donde se crea el conjunto de datos de Pumitas. Estos datos estarán ordenados en columnas verticales. Aquí definiremos los campos y sus características.

- **Consultas**: aquí se definen las preguntas que se van a formular a la base de datos con el fin de extraer y presentar la información resultante de diferentes formas (pantalla, impresora).
- **Formulario**: elemento en forma de ficha que permite la gestión de los datos de una forma más cómoda y visiblemente más atractiva.
- **Informe**: permite preparar los registros de la base de datos de forma personalizada para imprimirlos.
- **Macro**: conjunto de instrucciones que se pueden almacenar para automatizar tareas repetitivas.
- **Módulo**: programa o conjunto de instrucciones en lenguaje Visual Basic.

Para la definición de campos en Access se siguen las siguientes reglas:

- Los nombres de los campos, no pueden empezar con espacios en blanco ni caracteres especiales.
- No pueden llevar puntos, ni signos de exclamación o corchetes.
- Si pueden tener espacios en blanco intermedios.
- La descripción de un campo, permite aclarar información referida a los nombres del campo.

EL tipo de campo, permite especificar el tipo de información que se introduzca en dicho campo, esta puede ser:

- **Texto**: para introducir cadenas de caracteres hasta un máximo de 255.
- **Memo**: para introducir un texto extenso. Hasta 65.535 caracteres.
- **Numérico**: para introducir números.
- **Fecha/Hora**: para introducir datos en formato fecha u hora.
- **Moneda**: para introducir datos en formato número y con el signo monetario.
- **Auto numérico**: en este tipo de campo, Access numera automáticamente el contenido.
- **Sí/No**: campo lógico. Este tipo de campo es sólo si queremos un contenido del tipo Sí/No, Verdadero/Falso, etc.
- **Objeto OLE**: para introducir una foto, gráfico, hoja de cálculo, sonido, etc.
- **Hipervínculo**: podemos definir un enlace a una página Web.
- **Asistente para búsquedas**: crea un campo que permite elegir un valor de otra tabla o de una lista de valores mediante un cuadro de lista o un cuadro combinado.

Las características de cada campo son las siguientes:

- **Tamaño del campo:** define el tamaño máximo de caracteres que podemos introducir.

- **Formato:** dependiendo del tipo de campo, podemos escoger un formato. (Por ejemplo: monetario, fecha, etc.).
- **Máscara de entrada**: Access coloca una serie de signos automáticamente para facilitarnos la introducción de los datos. Por ejemplo, podemos introducir una fecha tecleando sólo los números del día, mes y año, y Access nos colocará automáticamente las barras de separación. **2/sep/05.**
- **Título:** se utiliza para dar un título al campo para posteriormente utilizarlo en formularios.
- **Valor predeterminado:** si deseamos que por defecto este campo contenga un valor que se repite a menudo para no tener que teclearlo.
- **Regla de validación:** podemos obligar al usuario a que introduzca los datos según unos criterios. Por ejemplo, podemos obligar a que se introduzcan datos numéricos inferiores a una cantidad.
- **Texto de validación**: cuando se incumplen las reglas de validación al introducir los datos, aparece un texto explicativo. Desde aquí podemos definir qué texto aparecerá.
- **Requerido:** si se habilita esta opción, el usuario está obligado a introducir datos.
- **Permitir longitud cero:** si se habilita esta opción, se permite la introducción de cadenas de longitud cero.
- **Indexado**: los campos indexados permiten acelerar las búsquedas. Podemos hacer que un dato se repita o no en la tabla con las opciones permitir duplicado si o no.

Elementos de una base de datos Access.

- Tablas: hasta 32,768 tablas en una base de datos. Trabajar con 254 a la vez.
- Consultas: máximo 16 tablas diferentes. Máximo 256 campos.
- Formularios: muestra y edición de datos. Imágenes, gráficos y sonidos.
- Informes: tablas, consultas, formularios.

4.3 Elección del software.

Microsoft Access.

Siendo Microsoft Access un sistema gestor de bases de datos relacionales, se tomó la determinación de su uso para el alojamiento y administración de la información de que será objeto Pumidata 1.0 por las siguientes razones:

- El sistema requerido es relativamente pequeño en cuanto a su contenido por lo que el manejador lo va a soportar fácilmente.

- El número de usuarios que van a usar el sistema es muy reducido por lo que la base de datos que se genera va a satisfacer las necesidades de los mismos.
- El manejador es el motor de base de datos nativo de varias herramientas de programación.
- Tiene todas las herramientas que se necesitan para generar todas las opciones que necesita el sistema.

Herramientas de desarrollo.

En este apartado se exponen algunas opciones sobre el lenguaje y herramientas para el desarrollo de los sistemas manejadores de bases de datos para la implantación del sistema de información.

Para la definición del lenguaje y la herramienta de desarrollo, se tomaron en cuenta a los más importantes y convenientes, resultando los siguientes:

- Visual C++.
- Power Builder.
- Visual Basic.
- Visual Fox Pro.

Viendo ventajas y desventajas de uno y otro, para desarrollar el sistema Pumidata 1.0, la herramienta que satisface las necesidades de desarrollo es Visual Basic, por las siguientes razones:

- Es una herramienta rápida para crear aplicaciones generales.
- Es un sistema productivo para crear soluciones en Windows.
- Tiene controles visuales preconstruidos.
- Tal vez lo más importante, acceso a bases de datos.
- Permite un ensamblaje fácil y rápido entre la interfaz de un usuario y los componentes prefabricados.
- Ofrece una gran capacidad y velocidad en su debugger sofisticado.
- Permite manipular otras aplicaciones para utilizarlas como componentes en aplicaciones propias (tales como Word, Excel, etc.) siempre y cuando dichas aplicaciones soporten OLE.
- Soporta una gran variedad de manejadores de bases de datos.
- Incluye como base de datos nativa a Microsoft Access, la cual provee acceso simultáneo con FoxPro, Dbase, Paradox, etc.

Sistemas operativos para el DBMS.

Se utiliza Windows XP Home Edition; tanto el manejador de la base de datos Access 2002 como la herramienta de programación Visual Basic 6.0 se pueden ejecutar bajo este ambiente.

ACCESS 2002.

Las características que nos llevan a la utilización e implementación de Access 2002 son las siguientes.

Requerimientos del Sistema.

- Procesador: Intel Pentium III a 133 MHz o superior con 64 MB de memoria RAM. Si se va a ejecutar más de una aplicación de Office al mismo tiempo, necesita como mínimo, un Pentium III a 250 MHz o superior con 128 MB de memoria. Para conseguir un funcionamiento excelente se recomienda un Pentium III a 350 MHz o superior, con al menos 128 MB de memoria RAM.
- Disco Duro: Entre 260 y 550 MB para una instalación estándar de Microsoft Office 2002. Si instala sólo Access 2002 necesita como mínimo unos 140 MB (o 200 MB para instalar todas las opciones de Access y los componentes comunes necesarios de Office).
- Monitor: VGA como mínimo, pero es recomendable una pantalla SVGA o superior.
- CD ROM y Ratón.
- Sistema Operativo: Windows 95, 98, ME, 2000 o Windows NT.

Microsoft Access es el principal sistema de gestión de bases de datos relacionales para la creación de aplicaciones de bases de datos en el escritorio. Es un sistema ameno y fácil de usar. Microsoft ha conseguido un gran éxito al proporcionar un entorno de interfaz gráfico.

Microsoft proporciona Visual Basic para Aplicaciones (VBA) en Access 2002 como una arma poderosa de desarrollo para automatizar su base de datos. Microsoft incorporó VBA en este producto para conseguir que Access 2002 fuera un sistema poderoso y versátil de gestión de bases de datos para los usuarios y programadores en la actualidad.

VBA proporciona que los programadores controlen la interfaz del usuario y manipulen los acontecimientos para crear una solución de base de datos que sea funcional, efectiva y fácil de utilizar.

La limitación de una base de datos de Access no se mide por cantidad de registros, sino por el tamaño de la base de datos, el límite de Access 97 o inferior es de 1GB, de Access 2000 o posterior es de 2GB.

En este caso se decide trabajar en Microsoft Access pues es el software instalado en el equipo de Pumitas y trabajar con otro sistema implicaría gastos que no están dispuestos a cubrir por la compra de licencias de software.

PROGRAMAS SECUENCIALES, INTERACTIVOS Y ORIENTADOS A EVENTOS.

Existen distintos tipos de programas. En los primeros tiempos de los ordenadores, los programas eran de tipo secuencial (también llamados tipo batch). Un programa secuencial es un programa que se arranca, lee los datos que necesita, realiza los cálculos e imprime o guarda en el disco los resultados. Mientras un programa secuencial está ejecutándose no necesita ninguna intervención del usuario. A este tipo de programas se les llama también programas basados u orientados a procedimientos o a algoritmos (procedural languages). Este tipo de programas siguen utilizándose ampliamente en la actualidad, pero la difusión de las PC's ha puesto de actualidad otros tipos de programación.

Los programas interactivos exigen la intervención del usuario en tiempo de ejecución, bien para suministrar datos, bien para indicar al programa lo que debe hacer por medio de menús. Los programas interactivos limitan y orientan la acción del usuario. Un ejemplo de programa interactivo podría ser Matlab.

Por su parte los programas orientados a eventos son los programas típicos de Windows, tales como Netscape, Word, Excel y PowerPoint. Cuando uno de estos programas ha arrancado, lo único que hace es quedarse a la espera de las acciones del usuario, que en este caso son llamadas eventos. El usuario dice si quiere abrir y modificar un archivo existente, o bien comenzar a crear un archivo desde el principio. Estos programas pasan la mayor parte de su tiempo esperando las acciones del usuario (eventos) y respondiendo a ellas. Las acciones que el usuario puede realizar en un momento determinado son variadísimas, y exigen un tipo especial de programación: la programación orientada a eventos. Este tipo de programación es sensiblemente más complicada que la secuencial y la interactiva, pero Visual Basic 6.0 la hace especialmente sencilla y agradable.

VISUAL BASIC 6.0.

Es un lenguaje de programación visual, también llamado lenguaje de 4ª generación. Esto quiere decir que un gran número de tareas se realizan sin escribir código, simplemente con operaciones gráficas realizadas con el ratón sobre la pantalla.

La programación en Visual Basic es una forma ágil y simple de crear aplicaciones para Microsoft Windows, es el método que se utiliza para desarrollar la interfaz gráfica de usuario. Prácticamente todos los elementos de interacción con el usuario de los que dispone Windows 95/98/NT pueden ser

programados en Visual Basic de un modo muy sencillo. El lenguaje de programación en Visual Basic proporciona todas las herramientas necesarias para el desarrollo rápido de aplicaciones, pudiendo incorporar todos los elementos de este entorno informático como: proyectos, formularios, plantillas de objetos, controles personalizados, ventanas, botones, cajas de diálogo y de texto, botones de opción y de selección, barras de desplazamiento, gráficos, menús, y un gestor de base de datos. En ocasiones bastan unas pocas operaciones con el ratón y la introducción a través del teclado de algunas sentencias para disponer de aplicaciones con todas las características de Windows 95/98/NT.

Características fundamentales de Programación en Visual Basic 6.0:

- La posibilidad de acceder a datos de la base de datos.
- Creación de archivos .exe, lo que permite distribuir la aplicación con gran libertad.

Requerimientos del sistema.

- Microprocesador Pentium a 90 MHZ o superior.
- Pantalla VGA de 640X480 o de resolución superior compatible con Microsoft Windows.
- 24 MB de RAM para Windows 95, 32 MB para Windows NT.
- Microsoft Internet Explorer versión 4.01 o posterior.
- Requisitos de espacio en disco duro:
    - Edición Estándar: instalación típica 48 MB, instalación completa 80 MB.
    - Edición Profesional: instalación típica 48 MB, instalación completa 80 MB.
    - Edición Empresarial: instalación típica 128 MB, instalación completa 147 MB.
- MSDN (para documentación) 67 MB.
- Internet Explorer 4.x aproximadamente 66 MB.
- CD-ROM.
- Mouse.
- Microsoft Windows 95 o posterior, Microsoft Windows NT Workstation 4.0 o posterior (se recomienda Service Pack 3).
- 486 DX/66 MHz o modelo superior de procesador (se recomienda Pentium o superior).

EXCEL 2002.

La configuración recomendada del sistema para Microsoft Excel versión 2002 es una computadora que corre bajo Microsoft Windows Profesional de XP con un procesador Pentium III y 128 MB de RAM.

Requerimientos mínimos del sistema.

- Microprocesador Pentium a 133 MHz o superior; Pentium III recomendado
- Memoria: Los requisitos de RAM dependen del sistema operativo utilizado:
    - Windows 98, o Windows 98 Segunda Edición. 24 MB de RAM mas 8 MB de RAM adicionales para Excel.
    - Windows Me, o Microsoft Windows NT. 32 MB de RAM mas 8 MB de RAM adicionales para Excel.
    - Windows 2000 Profesional. 64 MB de RAM mas 8 MB de RAM adicionales para Excel.
    - Windows XP Profesional, o Windows XP Home Edition.128 MB de RAM mas 8 MB de RAM adicionales para Excel.
- Disco Duro: 140 MB del espacio disponible de disco duro. 115 MB adicionales si se instala el sistema operativo en el disco duro. Los usuarios sin Windows XP, Windows 2000, Windows Me, u Office 2000 Service Release 1 (SR-1) requieren 50 MB adicionales del espacio de disco duro para el Sistema de Actualización de Archivos.
- Sistema operativo: Windows 98, Windows 98 Segunda Edición, Windows Millennium (Windows Me), Windows NT 4.0 con el Service Pack 6 (SP6) or superior, Windows 2000, o Windows XP o superior.
- CD-ROM.
- Monitor: Super VGA (800 × 600) o monitor con mayor resolución con 256 colores.
- Mouse.

WINDOWS XP HOME EDITION.

Requisitos del Sistema.

- Se recomienda un PC con procesador de 300 MHz o superior; se requiere un mínimo de 233 MHz; se recomienda la familia Intel Pentium/Celeron, la familia AMD K6/Athlon/Duron o procesador compatible.
- Se recomiendan 128 MB de memoria o más (soporte mínimo de 64 MB; puede limitar el rendimiento y algunas funciones).
- 1.5 GB de espacio disponible en disco duro.
- Adaptador y monitor de video Super VGA (800 x 600) o de resolución superior.
- Unidad de CD-ROM o DVD.
- Teclado y Microsoft Mouse o compatible.

CAPÍTULO 5. Implementación y pruebas.

5.1 Implementación del sistema.

La etapa de implementación implica el inicio de la codificación de algoritmos y estructuras de datos, definidos en la etapa de diseño, esto se lleva a cabo para el lenguaje de programación seleccionado, Visual Basic, y para el sistema de base de datos elegido, Access 2002.

Uno de los motivos de esta etapa es que la eficacia de las etapas anteriores a la implementación, sólo se demuestran una vez que el sistema se implementa y se utiliza. También es de gran importancia efectuar revisiones formales al producto, esto con el fin de:

- Descubrir errores en la función, la lógica o en la implementación del software.
- Verificar que el software cumple con los requisitos establecidos en el análisis.
- Verificar que el software se ha desarrollado bajo ciertos estándares (definidos en el análisis) .

Al realizar un desarrollo acorde al análisis y realizar revisiones formales al software, se garantiza la calidad del mismo, y se minimizan los posibles errores en el producto que se entrega al usuario final.

5.1.1 Desarrollo de la Base de datos.

El desarrollo de la base de datos se ha desarrollado a través de las herramientas propias de Access 2002. La instalación se realizó con las opciones que presenta por default.

Una vez instalado Access 2002 fue creada la base de datos llamada Pumitas2.

Para la creación de las tablas se hizo uso de las características propias de Access 2002, las cuales permiten diseñar la base de datos agregando campos, incluyendo el tipo de datos que almacenará, así como su longitud y la opción de almacenamiento de valores nulos. Se establecen también las llaves de cada tabla.

5.1.2 Implementación de la interfaz gráfica.

El desarrollo de la interfaz gráfica es la presentación final para los usuarios que nos permite interactuar con la base de datos. En el caso de Pumidata 1.0 se programó en Visual Basic.

5.2 Pruebas realizadas al sistema.

Las pruebas son una parte muy significativa del proyecto de adecuación de las aplicaciones, no sólo por su importancia en el logro de resultados sino por el tiempo y los recursos requeridos.

Plan de pruebas.

Se seguirá un plan de pruebas consistente en la ejecución de todos los pasos básicos de cada módulo, introduciendo información de prueba para ver si se reflejan de forma adecuada en la base de datos y las posteriores acciones son admitidas según lo permitido en cada estado de funcionalidad, como por ejemplo que el sistema deje pasar a un usuario aún introduciendo una contraseña errónea.

El objetivo de las pruebas reside en descubrir algún error. El éxito de una prueba se mide en función de la capacidad de detectar un error que estaba oculto.

Se plantean los siguientes niveles de prueba:

- Pruebas de aceptación.
- Pruebas de integración.
- Pruebas del sistema.
- Pruebas de especificaciones.
- Pruebas unitarias.

Pruebas de aceptación del sistema. Se llevan a cabo con el fin de validar que Pumidata cumple los requisitos básicos de funcionamiento esperado y permitir al usuario que determine la aceptación del sistema. Por este motivo, estas pruebas son realizadas por el usuario final del sistema y es durante este periodo de tiempo, cuando debe plantear todas las deficiencias o errores que encuentre antes de dar aprobado el sistema definitivamente.

Pruebas de integración. Su objetivo es realizar pruebas para detectar errores asociados con la interacción de los módulos, así como las interfaces entre componentes de la arquitectura del software.

Realización de las pruebas de integración. Apertura de ventanas asociadas a cada acción.

Pruebas del sistema. Proceso de prueba de un sistema integrado de hardware y software para comprobar si cumple los requisitos especificados. No prueba el software, sino la integración de cada módulo en el sistema, también hace pruebas para encontrar discrepancias entre el sistema y su objetivo

original, las especificaciones actuales y la documentación de sistemas. La preocupación principal es la compatibilidad de los módulos individuales.

Pruebas de especificaciones (código). El analista examina las especificaciones que el programa debe cumplir y cómo debe desempeñarlas bajo diferentes condiciones, no es un prueba completa, pero se piensa que si el programa cumple con las especificaciones entonces no fallará.

Pruebas unitarias. Se prueban los programas que conforman un sistema, a veces se denomina prueba de programas. Primero se enfoca a los módulos independientes, para localizar errores en la lógica y en la codificación. Debe llevarse a cabo de abajo hacia arriba, es decir, comenzando con los módulos más pequeños y de menor nivel uno a la vez. Se llama prueba ascendente.

En una prueba descendente, se comienza por los módulos de arriba hacia los de abajo, pero como no se llega a probar los módulos inferiores se utilizan los que se llaman módulos esclavos, simplemente para mandar mensajes de OK o de error para los módulos superiores, los módulos de menor nivel no se prueban, por eso es recomendable combinar la prueba ascendente con la descendente.

Las primeras pruebas consistieron en la aceptación por los usuarios. La prueba se llevó a cabo desde el inicio del diseño en donde a los usuarios se les mostraron las pantallas en un documento para su autorización y validación. Una vez validadas las pantallas por los usuarios se dio inicio a su desarrollo.

Terminado el sistema se comenzó con las pruebas unitarias con el propósito de detectar errores en los programas. Las pruebas se realizaron de forma ascendente y descendente.

Después de haber probado cada módulo y pantalla iniciamos las pruebas sobre el sistema completo, observando la secuencia de la información desde la captura con información real. Para cada uno de los usuarios se fueron grabando los datos sin ningún problema y con una respuesta inmediata.

Al llevarse a cabo las pruebas se evalúan los resultados, es decir, se comparan los resultados obtenidos de la prueba con los que se esperaban.

La depuración comienza cuando se descubre que existe un error. Este proceso de depuración es bastante complicado por tener que localizar la fuente del error. Una vez más, dependiendo de la modularidad del sistema con que fue construido el software, la detección del problema será más o menos rápida, para lo cual existen fundamentalmente dos enfoques: pruebas de caja blanca y pruebas de caja negra.

Pruebas de la caja blanca.

Este tipo de pruebas son muy útiles para probar aplicaciones desarrolladas con Visual Basic, ya que Visual Basic más que crear líneas de código lo que hace es crear formularios para introducir datos. Las pruebas de caja blanca se basan en el conocimiento del código para el diseño de casos de prueba que ejecutan todas y cada una de las partes del código para encontrar errores, por lo que se requiere acceso al código fuente para el diseño de las mismas.

Pruebas de caja negra.

Los métodos de la caja negra enfocan los requisitos funcionales del software permitiendo disponer de conjuntos de valores de entrada que ejerciten de forma completa todos los requisitos del programa. Es una prueba funcional que se basa en el comportamiento del sistema según las especificaciones internas del sistema.

Las pruebas de caja negra intenta encontrar errores de los siguientes tipos fundamentalmente:

- Funciones incorrectas o inexistentes.
- Errores relativos a las interfaces.
- Errores en estructuras de datos o en accesos a las bases externas.
- Errores debido al rendimiento.
- Error de inicialización o terminación.

La técnica de prueba de caja negra forma parte en gran medida de las pruebas de integración del software pues se ignora la estructura de control.

Pruebas en Visual Basic y Access.

Pumidata 1.0 está realizado con Visual Basic, el cual es un programa que soporta aplicaciones con bases de datos, por tanto un punto importante es comprobar que las estructuras de datos (tablas de Access) creadas para mantener la base de datos son las adecuadas.

Básicamente se probaron los formularios (estructuras básicas con las que trabaja Visual Basic) y los eventos que poseen los componentes de éstos.

Principalmente, Visual Basic crea aplicaciones que no son sólo código, más bien crea formularios que son una especie de interfaz con el usuario. Por tanto, las pruebas de caja blanca, que examina los

posibles caminos en la ejecución del código, fueron muy útiles para probar la aplicación. Se utilizó la prueba de la caja negra, para probar los formularios mediante diversos datos de entrada que comprobaron si el programa funcionaba como se desea. Es decir, aceptando los datos correctos y rechazando los datos no válidos según la especificación del programa.

Otro aspecto que se probó son los eventos que poseen los componentes de un formulario. Un evento es una acción que puede ocurrir sobre un componente del formulario. Por lo tanto, se comprobó que para cada componente del formulario y el evento asociado a él, se ejecutara correctamente el código asociado.

Pumidata 1.0 se ve sometido al siguiente plan de actuación:

- **Recuperación ante fallos del sistema:** Se procede a insertar datos erróneos en cada uno de los elementos lógicos del programa de capturas.
- **Errores de tipo.** Introducir cadenas de texto donde se pide un valor numérico.
- **Situaciones inesperadas.** Forzar actuaciones atípicas para tratar de ver fallos del sistema.
- **Seguridad.** Se comprueba que los métodos de seguridad funcionen frente a intentos de acceso no permitidos.

La realización de las pruebas ha sido llevada con éxito sin hallar fallo alguno del sistema, siendo todas las respuestas esperadas.

Los cambios en la base de datos han sido los esperados, es decir, todo ha funcionado correctamente. Tras realizar las pruebas se ha examinado cada una de las tablas de la base de datos para comprobar que todo es correcto.

En cuanto a la funcionalidad permitida en cada estado de la ejecución, también se ha tenido éxito, mostrándose en cada caso las ventajas adecuadas con la información correspondiente a cada situación.

5.3 Factibilidad técnica, operativa y económica.

Los sistemas en la actualidad deben cumplir con características indispensables de compatibilidad y factibilidad para el usuario. Actualmente el ambiente Windows permite crear sistemas gráficos que son amigables para el usuario permitiendo operar de forma sencilla. Además se brinda la seguridad, confiabilidad y veracidad que son parte fundamental para la eficiencia operativa de los sistemas.

Pumidata 1.0 se instalará, previa verificación de los mínimos requerimientos para poder ejecutar la aplicación.

Factibilidad técnica.

Es aquí donde se estudia si el trabajo para el proyecto puede desarrollarse con el software y el hardware, además del personal existente, y en el caso de necesitar nueva tecnología, cuáles son las posibilidades de desarrollarla.

Es importante hacer una evaluación de la factibilidad técnica que se requiere para poder realizar la implantación del sistema, ésta depende en gran medida de los recursos mínimos indispensables con los que debemos de trabajar.

Para la implantación de Pumidata 1.0 se requiere realizar las siguientes actividades.

- Instalación en el equipo de cómputo de Pumitas bajo Windows XP.
- Instalación de Visual Basic 6.0.
- Configuración e instalación de la base de datos Access 2002.
- Configuración e instalación del sistema en Pumitas.

Todo esto conlleva un tiempo de aprendizaje muy pequeño por parte del usuario para que éste pueda manejar y dominar completamente el sistema.

La aplicación se ha diseñado para que sea fácil de modificar en el caso de actualizaciones en el futuro.

Factibilidad operativa.

En esta fase se investiga si los usuarios usarán el sistema, si habrá algún tipo de capacitación y si se le tendrá que dar algún tipo de mantenimiento al sistema.

El éxito de Pumidata 1.0 es que tenga una buena aceptación de las diferentes personas que laboran dentro de Pumitas, principalmente aquellas personas que utilizarán de una forma constante el sistema. Tomando en consideración que con el uso de Pumidata 1.0 se verán de una gran forma beneficiadas ya que el sistema ofrece versatilidad y simplicidad al ser utilizado. Pumidata 1.0 es un sistema que por su sencilla forma de captura, nos permite ahorrar tiempo, esfuerzo debido a sus pantallas de inserción de datos.

Todo esto supone una factibilidad operativa ya que la aplicación tiene una interfaz gráfica de usuario fácil de utilizar y que puede correr bajo el ambiente gráfico de Windows XP.

Factibilidad económica.

Existen muchos factores que influyen en el costo de un producto de programación. El efecto de éstos factores es difícil estimar y, por ende también lo es el costo del esfuerzo en el desarrollo o en el mantenimiento.

Los factores que influyen en la estimación del costo son:

- Las capacidades individuales del personal asignado al proyecto y su familiaridad con el área de aplicación.
- La complejidad del producto, y el tamaño de éste.
- El tiempo asignado.
- El nivel de confiabilidad.
- El nivel tecnológico utilizado.
- La disponibilidad, familiaridad y estabilidad del sistema donde se desarrolla el producto.

El costo del software constituirá un pequeño porcentaje del costo total de los sistemas basados en computadora. Una mala estimación en el costo del software traería problemas en la aceptación. Y debido a que los sistemas son el elemento más caro; puede ser lo que marque la diferencia entre beneficios y pérdidas.

Sobrepasarse en el costo puede ser desastroso para el desarrollo. La estimación del costo y del esfuerzo del software nunca será una ciencia exacta, son demasiadas las variables humanas, técnicas, de entorno, políticas que pueden afectar el costo final de éste y al esfuerzo aplicado para el desarrollo.

La estimación de costos puede llevarse a cabo en forma jerárquica hacia abajo (top-down) o en forma jerárquica hacia arriba (bottom-up). La estimación jerárquica hacia abajo se enfoca primero a los costos del nivel del sistema, así como a los costos de manejo de la configuración, del control de calidad, de la integración del sistema, del entrenamiento y de las publicaciones de documentación. Los costos del personal relacionado se estiman mediante el examen del costo de proyectos anteriores que resulten similares.

En la estimación jerárquica hacia arriba, primero se estima el costo del desarrollo de cada módulo o subsistema; tales costos se integran para obtener un costo total. Esta técnica tiene la ventaja de

enfocarse directamente a los costos del sistema, pero se corre el riesgo de despreciar diversos factores técnicos relacionados con algunos módulos que se desarrollan.

La técnica subraya los costos asociados con el desarrollo independiente de cada módulo o componente individual del sistema, aunque puede fallar al no considerar los costos del manejo de la configuración o del control de la calidad. En la práctica, ambas técnicas deben de compararse para que interactivamente se eliminen las diferencias obtenidas.

Otras técnicas con las que se puede estimar el costo de un sistema es realizando una estructura de divisiones de trabajo, esta técnica es de tipo jerárquico en donde se establecen diferentes partes de un sistema.

A continuación se muestran las diversas cotizaciones del valor del sistema en las siguientes tres tablas de costo.

Estimación de costos por obra terminada.

Para estimar el costo contamos con el auxilio del diagrama de Gantt de las etapas utilizadas para el desarrollo del sistema, el cual fue iniciado el 4 de abril de 2005 y terminado el 31 de octubre de 2005.

No.	ACTIVIDAD	DÍAS																								
		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
1	Investigación preliminar																									
2	Aplicación de entrevistas																									
3	Definición del problema																									
4	Determinación de requerimientos																									
5	Análisis de los requerimientos																									
6	Diseño lógico																									
7	Diseño de diagramas del sistema																									
8	Diagrama entidad-relación																									
9	Diseño de la Base de Datos																									
10	Creación de la Base de Datos																									
11	Codificación																									
12	Pruebas																									
13	Implantación																									
14	Capacitación																									
15	Documentación																									

Figura 5.1 Diagrama de Gantt. Tiempo en días.

Así, la estimación de costos queda como sigue:

ETAPA	DÍAS	COSTO / DÍA	SUBTOTAL
Análisis	15	800	$12,000
Diseño	15	500	$7,500
Codificación	30	1500	$45,000
Pruebas	15	400	$6,000
Capacitación	2	800	$1,600
Implantación	2	650	$1,300
Documentación	25	400	$10,000
Total			$83,400

Estimación de costos por el número total de líneas programadas para el sistema.

En este caso, nos auxiliamos del número de líneas de código por módulo:

Módulo	Número aproximado de líneas de código
1	32
2	870
3	1170
4	2670
5	300
6	330
7	450
8	3
9	120
10	450
11	330
12	120
13	150
14	150
15	150
16	150
17	70
18	65
19	65
20	40
21	40
22	75
23	100
24	240
25	4
26	4
27	10
28	150
Total	8308

Y la estimación de costos queda:

Número aproximado de líneas de código	Costo por línea	Total
8308	$10	$83,080

Estimación de costos por el número total de módulos del sistema.

Número de módulos	Costo por módulo
33	$3,000
Total	$99,000

El costo del sistema es aproximado en cualquier tipo de forma de venta. Y por lo tanto Pumidata 1.0 será vendido en $83.000.

Los precios de costo de software (herramienta de desarrollo) y hardware (equipos destinados al sistema) corren por parte de Pumitas al momento de aprobar el proyecto.

5.4 Mantenimiento.

El mantenimiento está asociado a la corrección de errores, a nuevas adaptaciones requeridas por la evolución del entorno del software, así como a los cambios y nuevas especificaciones propuestas por el cliente.

Mientras que el cambio tecnológico afecta indirectamente a los sistemas en el software, el entorno de trabajo y los usuarios lo hacen directamente, produciendo demandas de mantenimiento adaptativo y perfectivo, respectivamente.

Mantenimiento correctivo.

A pesar de las pruebas y verificaciones que aparecen en etapas anteriores del ciclo de vida del software, los programas pueden tener defectos. El mantenimiento correctivo tiene por objetivo localizar y eliminar los posibles defectos de los programas.

Mantenimiento adaptativo.

Este tipo de mantenimiento consiste en la modificación de un programa debido a cambios en el entorno (hardware o software) en el cual se ejecuta.

Mantenimiento perfectivo.

Cambios en la especificación, normalmente debidos a cambios en los requerimientos de un producto de software, implican un nuevo tipo de mantenimiento llamado perfectivo. Se puede definir el mantenimiento perfectivo como el conjunto de actividades para mejorar o añadir nuevas funcionalidades requeridas por el usuario.

Mantenimiento preventivo.

Este último tipo de mantenimiento consiste en la modificación del software para mejorar las propiedades de dicho software sin alterar sus especificaciones funcionales.

Con respecto al proyecto, si surge algún problema con respecto a una librería que Visual Basic utiliza para funcionar al momento de compilar el programa, pues será necesario la instalación completa del sistema, no sin antes respaldar la base de datos para que al momento de instalar nuevamente, la información ya guardada en la base de datos no se pierda.

Y con respecto a la base de datos, si por alguna razón se daña, Access 2002 tiene una herramienta para poder reparar la base de datos sin perder información. Pero ha habido casos en donde si se pierden datos debido a tal reparación, esto es debido a que algunos datos o registros se perdieron o se dañaron de forma definitiva y Access no puede restaurarlos, por lo que borra esa información que se convierte en basura y reacomoda la información en buen estado.

Aparte de que repara la base de datos, Access 2002 también la compacta. Esto es de mucha ayuda cuando se usa Access con tablas, formularios, reportes, consultas, etc., ya que cuando generalmente se usan todas las propiedades de un archivo de Access, éste tiende a crecer conforme se vayan introduciendo datos, por lo que es necesario depurar el archivo para que no crezca demasiado. Esta herramienta nos ayuda cuando el archivo crece desmesuradamente hasta llegar el momento en que ya no puede abrirla o se daña.

Puesta en operación.

En la puesta en marcha o instalación del sistema Pumidata 1.0 fue necesario contar con algunos factores que condujeran al éxito de la instalación, éstos factores los podemos resumir como:

- Contar con todas las aplicaciones que se integran para funcionamiento del sistema: Access, Excel y herramientas multimedia, todas éstas en la plataforma donde se va a instalar el sistema.

- Verificar que se tenga la versión apropia de los componentes para evitar alguna incompatibilidad entre ellos.
- Un proceso automático donde se copien los archivos con la ruta correspondiente.
- Definir todas las rutas de carpetas y archivos necesarios para la correcta ejecución de la aplicación.
- Contar con la base de datos a la que hace referencia el sistema: Pumitas2.mdb.
- Una vez instalada la aplicación se capacitó a los usuarios para manipular adecuadamente el nuevo sistema.
- La instalación del sistema Pumidata1.0 debe contar con:
    - Pumidata 1.0 (aplicación).
    - Pumitas2.mdb
    - ASISTENCIA ANUAL EN BLANCO.xls

La capacitación para los usuarios se facilita dado que la aplicación se encuentra bajo un ambiente gráfico y amigable y con validaciones que evitan que el usuario de Pumidata 1.0 cometa errores de captura. La capacitación básica para los usuarios consistirá de los siguientes pasos:

- Inicio del sistema.
- Presentación de las pantallas del sistema.
- Uso y alcance del sistema (altas, consultas, modificaciones, eliminación, altas y reportes).
- Manual del usuario.

La capacitación tendrá una duración de una sesión con duración aproximada de 3 horas, procurando así responder a todas las preguntas o inquietudes que se fueran presentando sobre el sistema. El uso y operación deberán estar apegados a los procedimientos propios de Pumitas y a la responsabilidad del usuario al buen uso del sistema.

Requerimientos Técnicos.

- Procesador Pentium III a 800 MHz o posterior (recomendado).
- 128 MB de memoria RAM mas lo que se recomienda dependiendo de la versión de Windows que se tenga (128 MB para Windows 98, 256 hasta 512 MB para Windows 2000 y XP).
- Disco duro: El programa compilado y listo para instalar requiere de aproximadamente 50 MB de espacio, pero generalmente las computadoras vienen con discos de 40 GB.
- De preferencia, Windows 98 o posterior. Si hubiese el caso de Windows NT 4.0 hay que instalar también el Service Pack 3 o posterior.
- Unidad de CD-ROM.

- Monitor: Resolución VGA o posterior.
- Periféricos: Impresora predeterminada por la versión de Windows.

El sistema Pumidata 1.0 está instalado en:

Nombre del sistema operativo: Microsoft Windows XP Professional
Versión 5.1.2600 Service Pack 2 Compilación 2600
Memoria física total: 256 MB

Respaldo de la base de datos.

Como sucede con todo programa de manejo de información, Pumidata 1.0 deberá de contar con una estrategia de respaldos eficiente. La estrategia se deberá determinar con base en la carga de trabajo del sistema y el riesgo de pérdida de datos, tomando en cuenta esta información, se recomienda tener una estrategia de respaldos utilizando discos compactos re-grabables y que se deberá llevar a cabo como se menciona a continuación.

En cada sesión de trabajo, al finalizar el sistema le da la opción al usuario de respaldar la información, así, esta estrategia se puede aplicar cada vez que se use el programa, además de respaldar cada semana (por ejemplo, cada viernes), respaldar cada mes, y cada seis meses.

5.5 Generación de reportes.

Los reportes presentados en el sistema se generan con la información almacenada en los campos de la base de datos.

Los reportes son documentos internos que indican los diversos procesos que se efectúan durante el transcurso de las actividades de Pumitas, y que significan el punto de partida para la planeación de la siguiente temporada.

La creación de reportes es de gran ayuda para los usuarios del sistema, y se generan haciendo uso del diseñador de informe de datos de Visual Basic, que es su generador de informes de datos versátil con capacidad integrada de creación de informes.

El generador de informes puede usarse con un origen de datos como el diseñador de entorno de datos (DataEnvironment) para crear informes con datos procedentes de diferentes tablas relacionales.

Visual Basic permite imprimir los informes y exportarlos como archivos de tipo texto o HTML.

Los reportes que genera el sistema son:

- Integrantes de los equipos por categoría.
- Porteros por categoría.
- Evaluaciones por categoría.
- Sobrepeso por categoría.
- Representativo por categoría.
- Faltistas por categoría.

El formato general que presentan los reportes es el siguiente:

PUMITAS C.U. FÚTBOL, A.C.

INSTITUCIÓN EDUCATIVA - DEPORTIVA

CATEGORÍA 12 AÑOS

TEMPORADA 2005

INTEGRANTES DE LOS EQUIPOS

EQUIPO	APELLIDO PATERNO	APELLIDO MATERNO	NOMBRE (S)	POSICIÓN
XXXXXXXXXXXX	XXXXXXXXXXXX	XXXXXXXXXXXX	XXXXXXXXXXXX	XXXXXXXXXXXX

CONCLUSIONES.

No cabe duda que el uso de la tecnología y la automatización de procesos en todos los niveles de cualquier organización deportiva son un requisito indispensable en la actualidad para que resulte altamente recomendable y funcional. La dependencia tecnológica se ha incrementado de una manera significativa en los últimos años, por lo que no existe organización que no esté sujeta de una u otra forma a ella.

Los sistemas de información desarrollados a la medida de las necesidades del usuario resulta en muchas ocasiones más baratos en recursos (dinero, conocimientos, tiempo, etc.) que adquirir un sistema ya liberado, pero que resulta muy difícil que se adecue a todos los requerimientos que plantea, pues debe ser el soporte informático quien se adapte a solucionar el problema y no al revés.

Pumidata 1.0 refleja la información proporcionada por los monitores de forma inmediata a la base de datos, con lo cual los responsables de las categorías pueden conocer inmediatamente la situación real de cada una de ellas.

Tiene un mejor seguimiento del desarrollo individual de cada integrante de Pumitas, así como de cada equipo, y será posible tomar medidas antes que se generen los problemas, anticipando soluciones que eviten que la organización tenga situaciones problemáticas que se salgan de control.

Mantiene un control estricto sobre el seguimiento del desarrollo de las actividades por parte de los integrantes de Pumitas.

El sistema contempla una administración completa de las actividades organizativas de Pumitas, esto nos permite toma de decisiones acertadas y la proyección de futuras temporadas.

Toda esta información brinda a Pumitas la oportunidad de tener mejores resultados en la planificación, en beneficio de las familias y el crecimiento de la organización.

En el desarrollo de este proyecto se hizo todo el proceso para la construcción de un sistema, desde el análisis hasta la capacitación del usuario final.

Pumidata 1.0 cumplió con los requisitos solicitados por el usuario y se cubrieron todos los puntos a satisfacción de éste.

APÉNDICE A. Manual de instalación.

Para distribuir la aplicación de Pumidata 1.0 se necesita primero crear el instalador para que el sistema pueda ser utilizado en cualquier equipo de cómputo.

Muchas veces resulta innecesario crear el paquete de instalación del programa, basta con crear el ejecutable, pero como en este caso se requiere utilizar librerías, es indispensable contar con el instalador.

Primero se crea el ejecutable del programa, que en este caso es PUMIDATA.EXE, en Visual Basic desde:

>File
>>Make PUMIDATA.EXE...

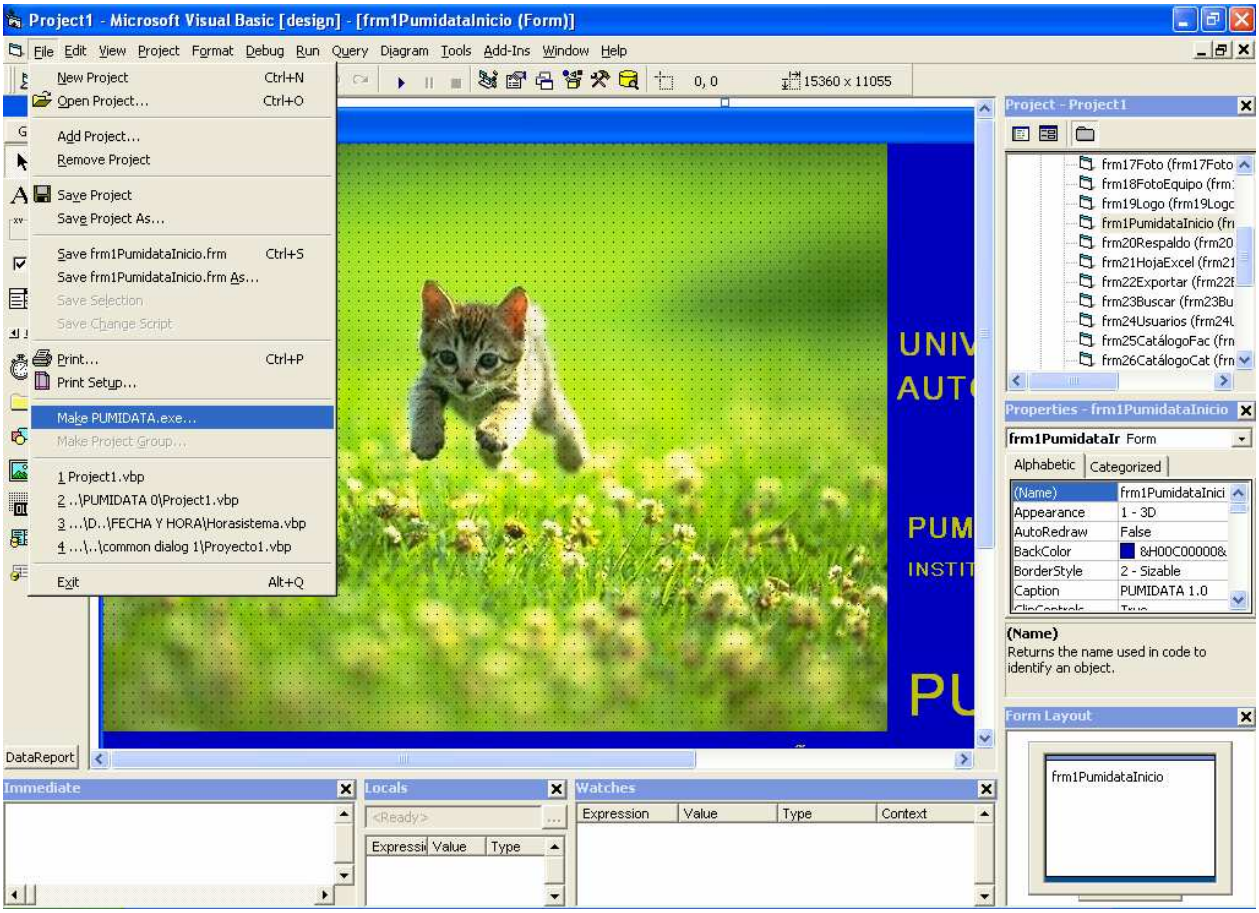


Figura 1. Haciendo el programa ejecutable.

Para crear el instalador, Visual Basic dispone de una herramienta llamada “Package & Deployment Wizard” que se encuentra dentro de las opciones de Visual Studio 6.0.

Nos vamos a:

>Inicio
>> Todos los programas
>>> Microsoft Visual Studio 6.0
>>>> Microsoft Visual Studio 6.0 Tools
>>>>>Package & Deployment Wizard

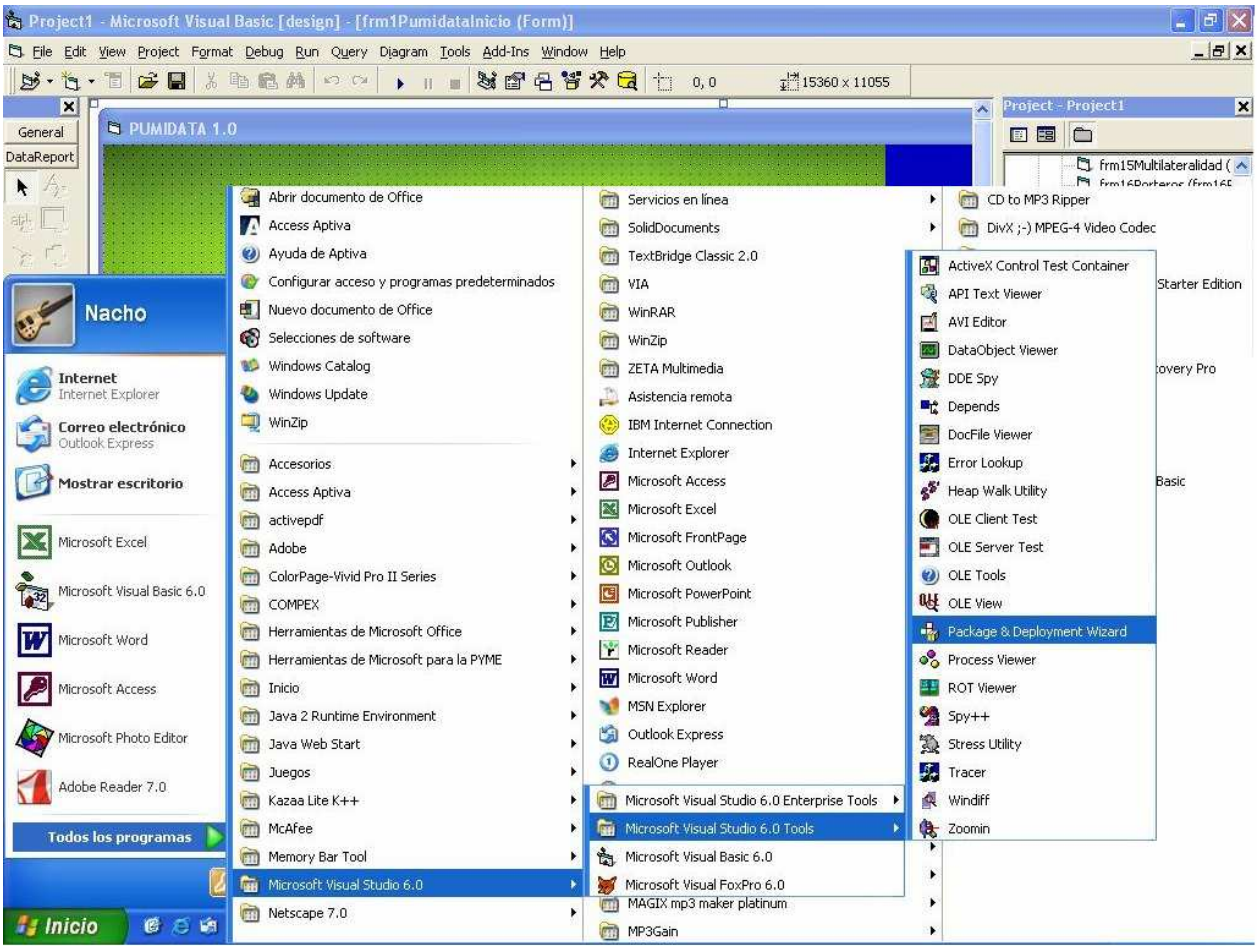


Figura 2. Abrir el Asistente.

Se busca el proyecto a distribuir, en este caso Project1.vbp y se da click en el botón “Package”.

Figura 3. Seleccionar el proyecto.

Se etiqueta el paquete de instalación como Pumidata 1.0 2005 y se da click en “Next”.

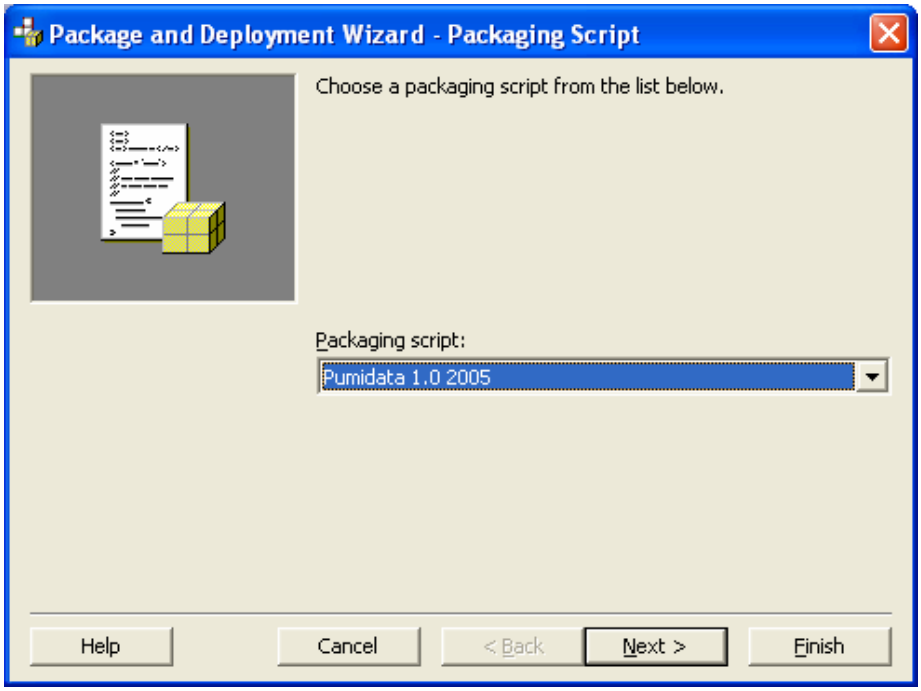


Figura 4. Escritura del paquete de instalación.

Se elige “Standard Setup Package” y se da click en “Next”.

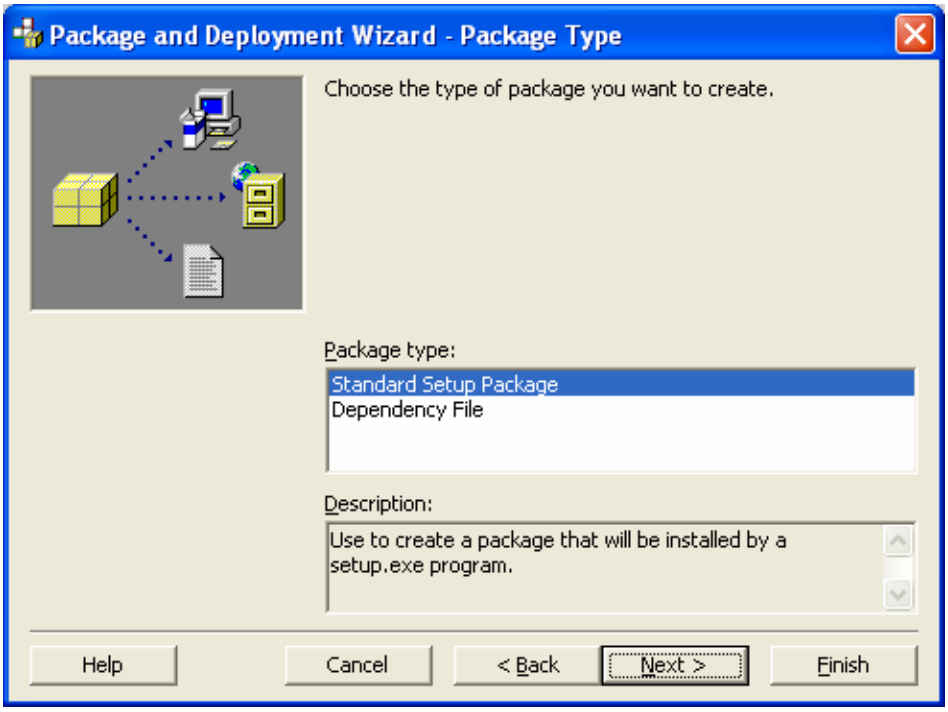


Figura 5. Instalación estándar del paquete de instalación.

Se elige la ruta donde se creará el instalador, se crea una nueva carpeta con el nombre Pumidata 1.0 y se da click en “Next”.

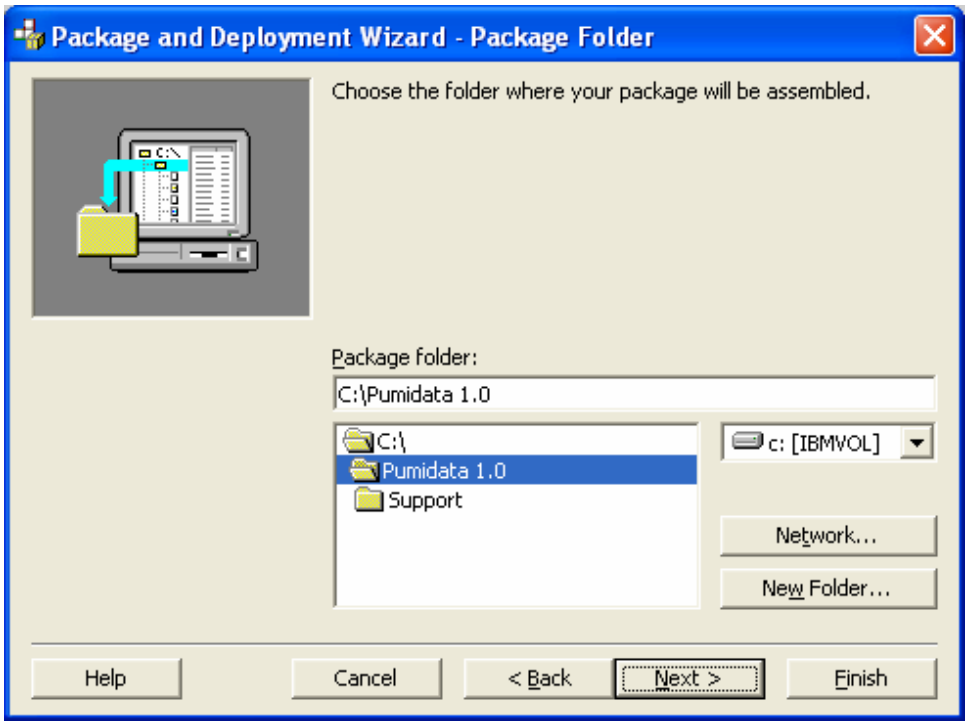


Figura 6. Carpeta del paquete de instalación.

Se activan las casillas de verificación de todos los componentes que aparecen, excepto Excel quien ya se encuentra instalado, se da click en “Next”.

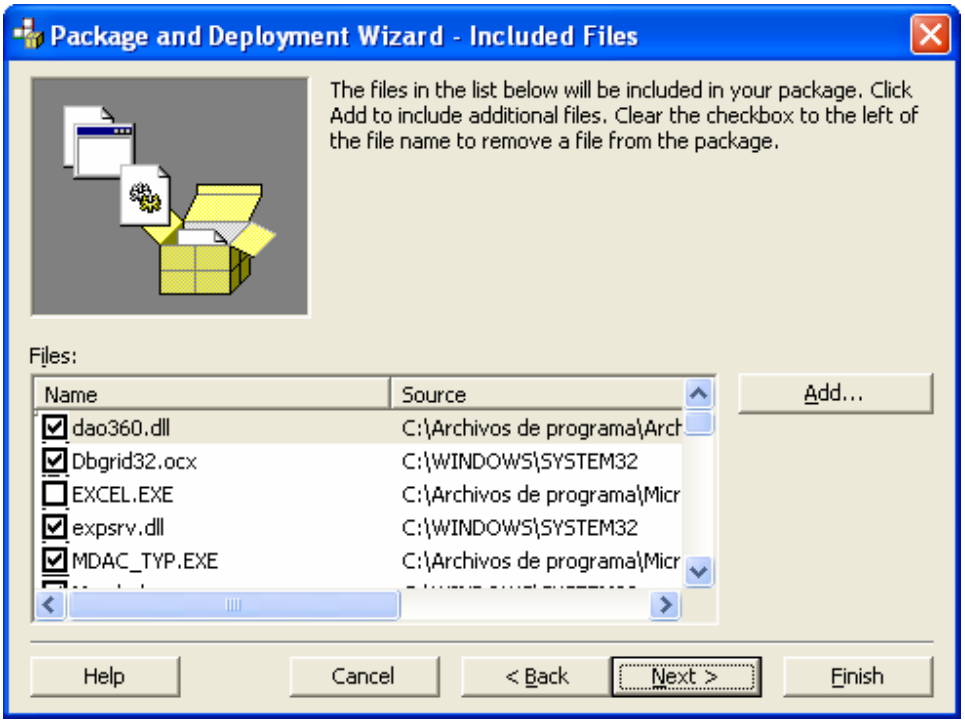


Figura 7. Archivos incluidos en el paquete de instalación.

Dejamos seleccionado “Single cab” y se da click en “Next”

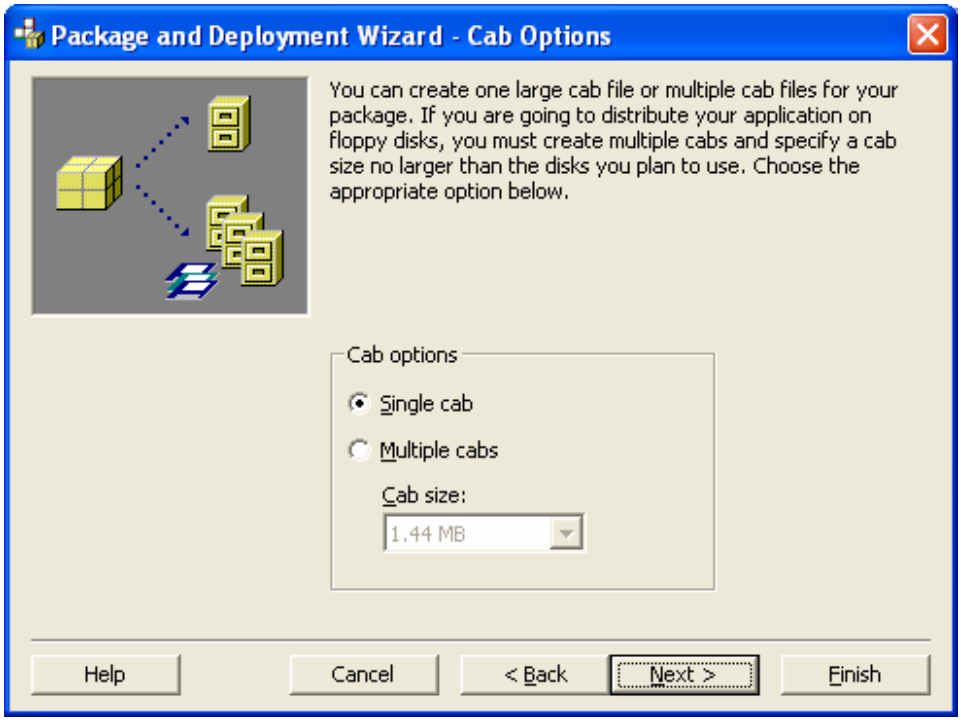


Figura 8. Distribución en un cab.

Colocamos como titulo de instalación Pumidata 1.0 y se da click en “Next”.

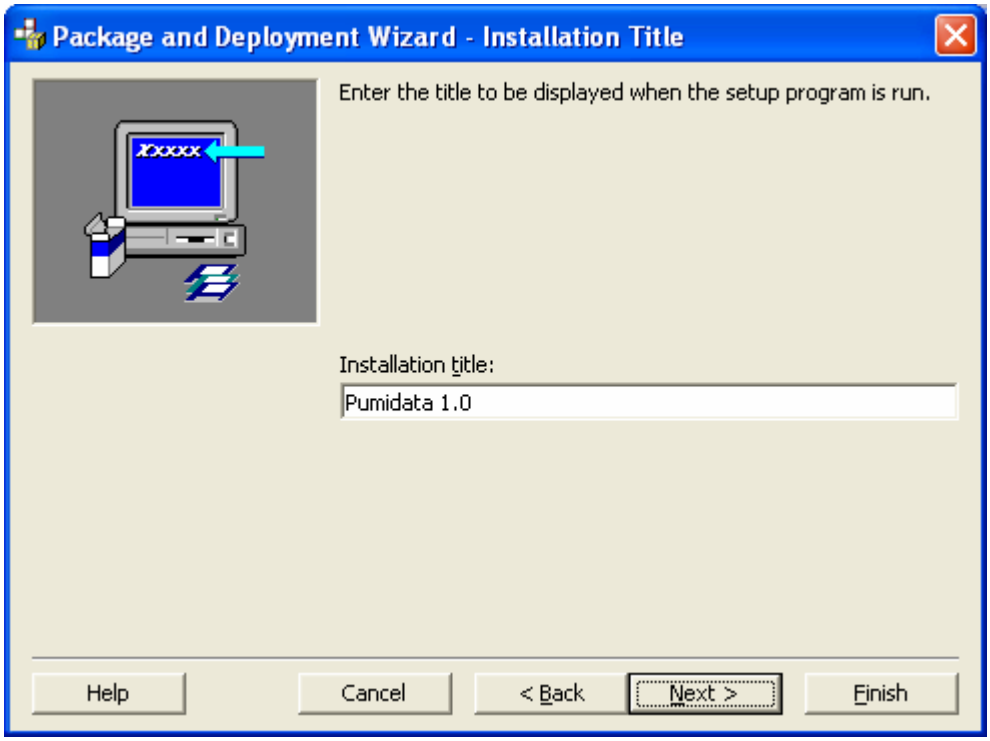


Figura 9. Título del programa de instalación.

Se indica la ruta desde el menú Inicio, y se da click en “Next”.

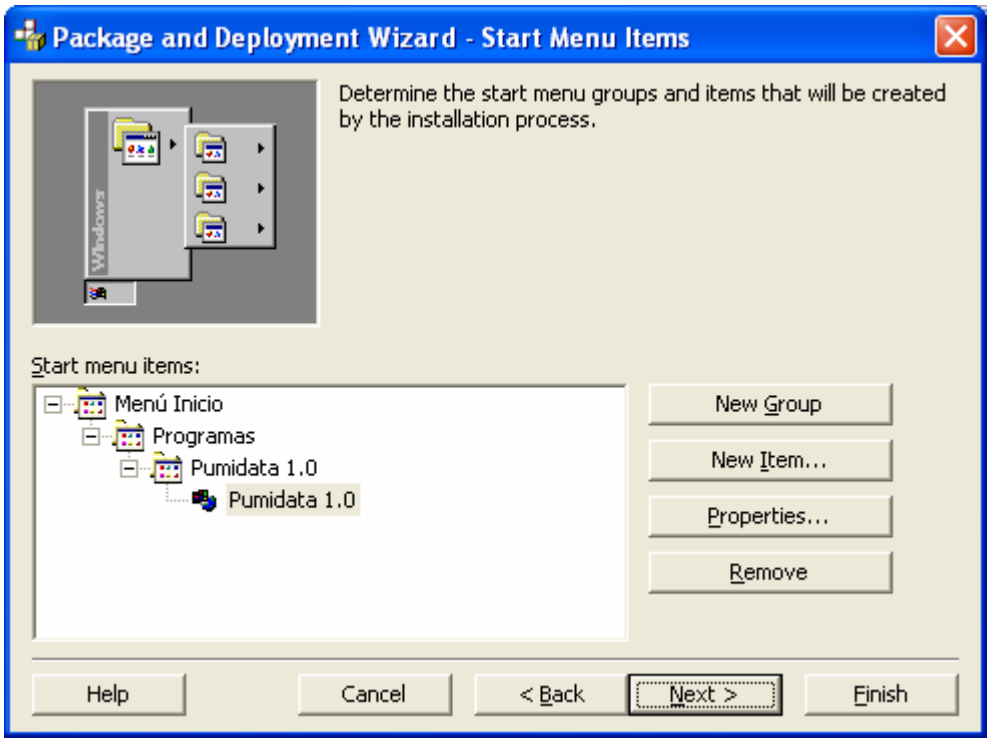


Figura 10. Elementos creados por el proceso de instalación a partir del menú Inicio.

Otra vez “Next”.

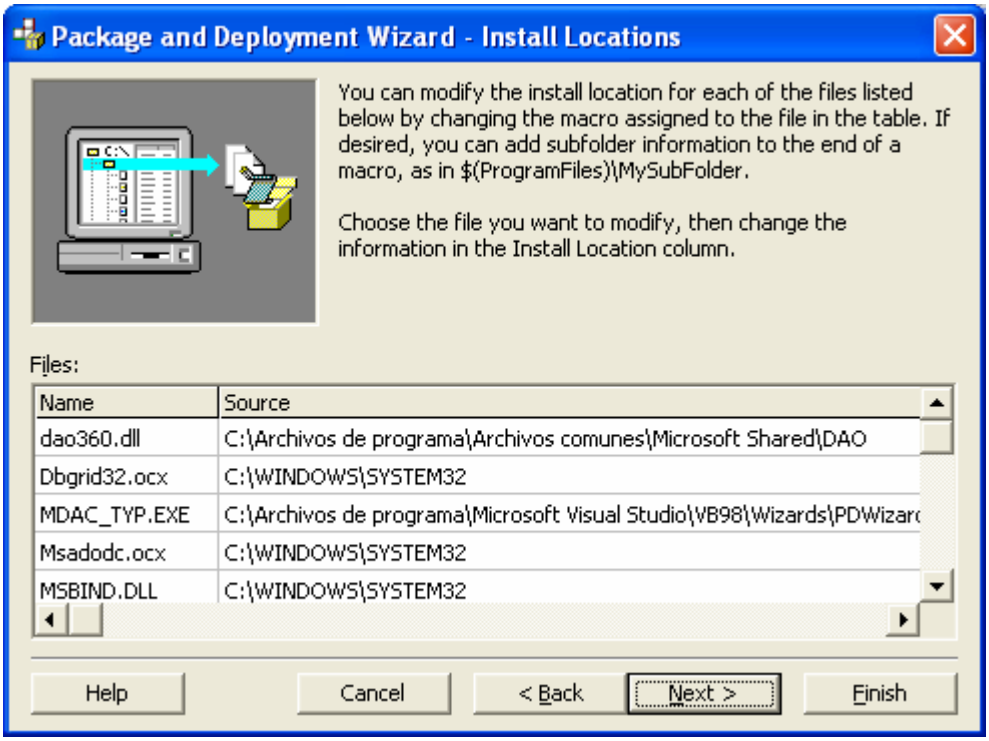


Figura 11. Ubicación de los archivos de instalación.

De nuevo “Next”.

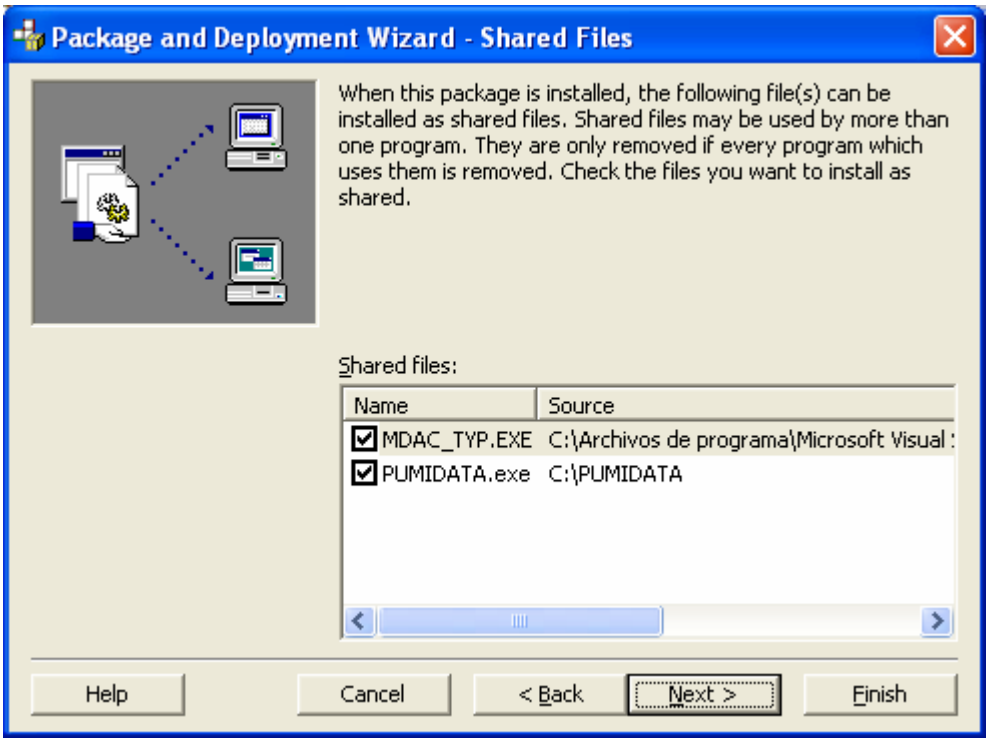


Figura 12. Archivos compartidos.

Por último “Finish”.

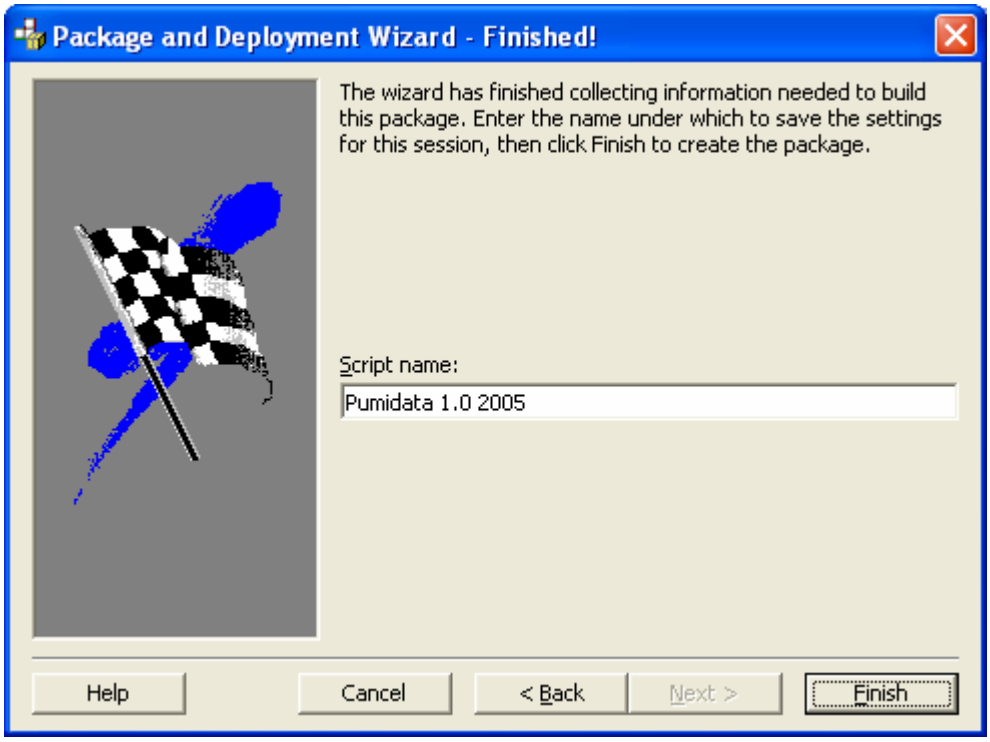


Figura 13. Fin del asistente.

Como resultado de ejecutar el asistente del paquete de instalación y distribución, se crea la carpeta Pumidata 1.0 en la unidad C, en donde aparecen los siguientes archivos:

Figura 14. Archivos que conforman el instalador.

El archivo PUMIDATA posee el nombre de la carpeta donde estará el ejecutable del sistema.

Ya se tiene listo el instalador con un tamaño de 43 MB disponible para ser distribuido.

En el equipo de cómputo donde se instalará el sistema se descarga la carpeta Pumidata 1.0 y se da doble click en “Setup”, a partir de este momento se procede como una instalación normal.

Se da click en OK.

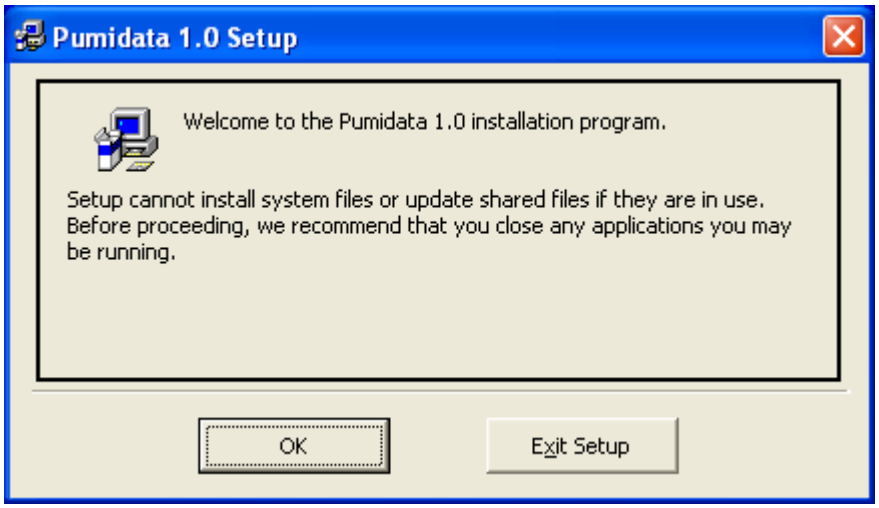


Figura 15. Inicio de la instalación de Pumidata 1.0.

Se elige la carpeta de instalación y se inicia la instalación.

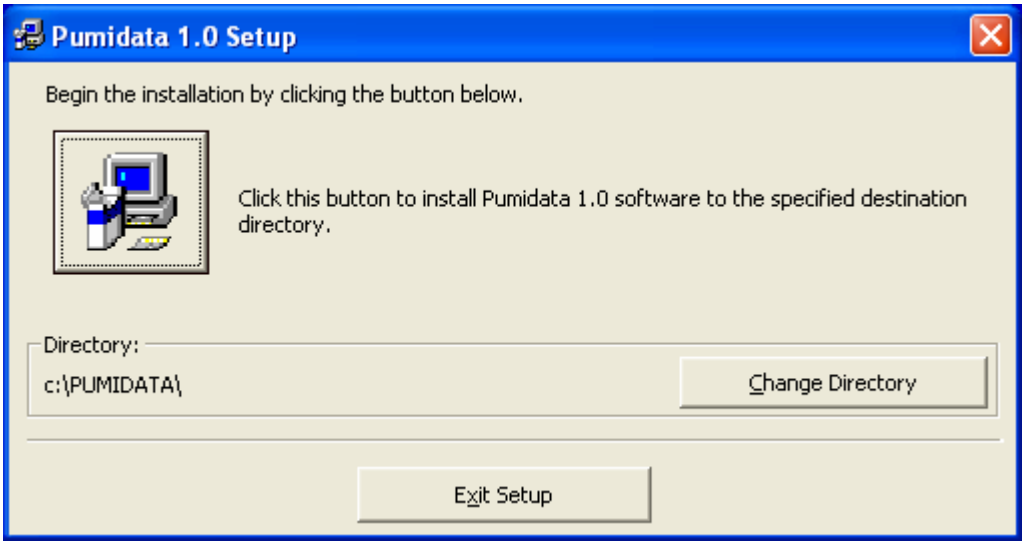


Figura 16. Elegir directorio de instalación.

Se escoge el grupo de programa.

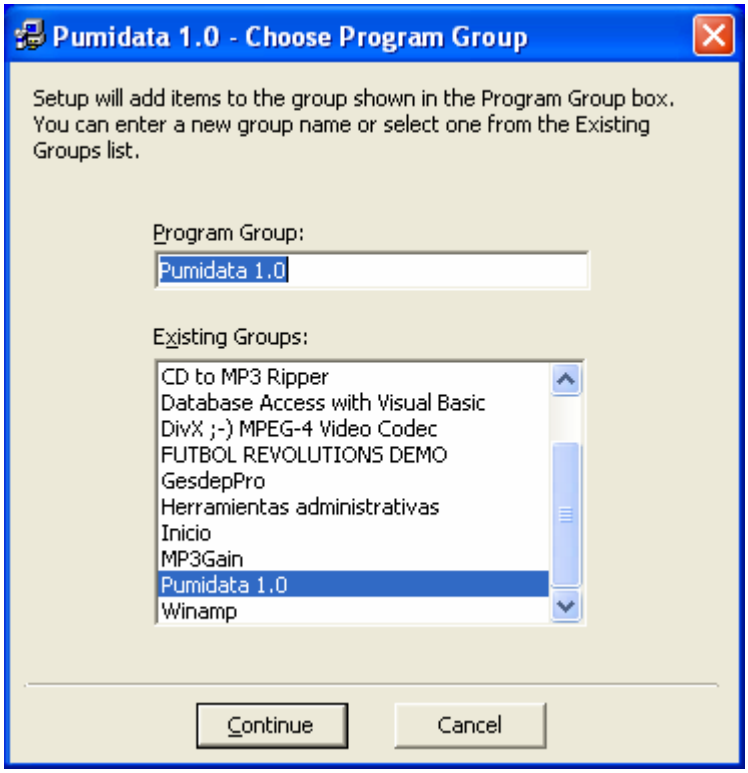


Figura 17. Elegir grupo de programa.

Y el proceso de instalación termina automáticamente.

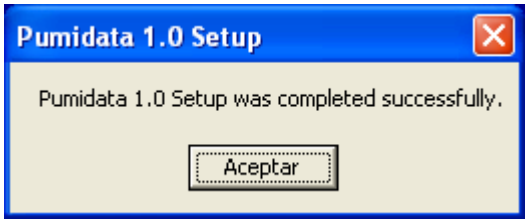


Figura 18. Instalación concluida.

El inicio del programa se realiza desde:

>Inicio
>>Todos los programas
>>>Pumidata 1.0
>>>>Pumidata 1.0

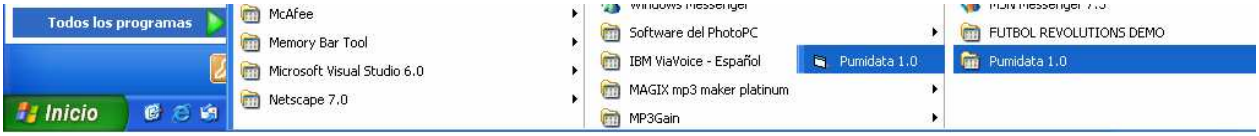


Figura 19. Abriendo Pumidata 1.0.

En la carpeta PUMIDATA se tienen los siguientes archivos después de la instalación:

Figura 20. Ejecutable y notas de la instalación.

A la cual se le agregan los siguientes archivos y carpetas:

Carpetas:

- ASISTENCIA: contiene los archivos en Excel de la asistencia anual de los equipos.
- Equipos: contiene las imágenes con los escudos de los equipos.
- FotoEquipo: contiene las fotos de los equipos.
- Fotos infantiles: contiene las fotos de los integrantes de la Organización Pumitas.
- Respaldo: designada como default para almacenar el respaldo de la base de datos.

Archivos:

- Pumitas2.mdb: base de datos.
- ASISTENCIA ANUAL EN BLANCO: archivo de Excel con el formato para registrar la asistencia de un equipo.
- ASISTENCIA ACONDICIONAMIENTO: archivo de Excel con el formato para registrar la

asistencia del taller de Acondicionamiento Físico.

- ASISTENCIA MOTRICIDAD: archivo de Excel con el formato para registrar la asistencia del taller de Motricidad.
- ASISTENCIA MULTILATERALIDAD: archivo de Excel con el formato para registrar la asistencia del taller de Multilateralidad.
- ASISTENCIA PORTEROS: archivo de Excel con el formato para registrar la asistencia del taller de Porteros.

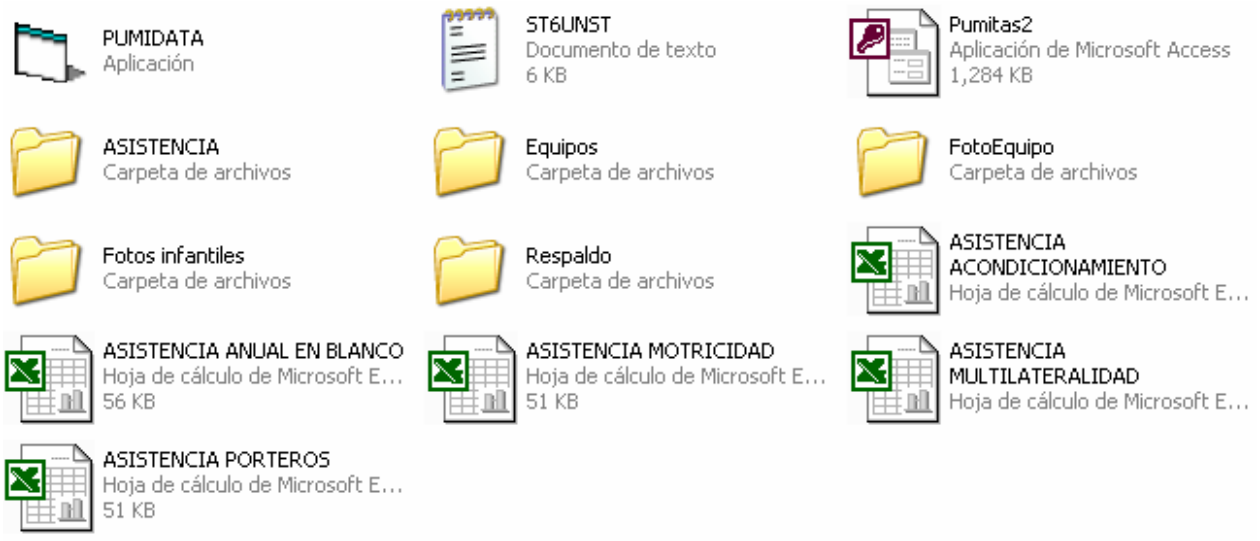


Figura 21. Archivos y carpetas necesarios para el sistema.

APÉNDICE B. Manual de usuario.

Pumidata 1.0 es un sistema de computación diseñado para el Seguimiento, Control y manejo de información de los integrantes de la Institución Educativa-Deportiva Pumitas C.U. Fútbol, A.C.

Lo primero que debemos de hacer es identificar la ruta de acceso al programa. Para dar inicio a una sesión de trabajo, será necesario que localizar el programa Pumidata 1.0, desde Inicio y Programas.

Una vez que se ha activado el programa, Pumidata esperará la entrada de la clave de usuario y contraseña para poder acceder a las opciones del sistema.

Figura 1. Acceso a Pumidata 1.0

Si la clave y contraseña han sido incorrectas el sistema mostrará un mensaje de error y solicitará de nuevo que introduzca los datos para acceder al sistema.

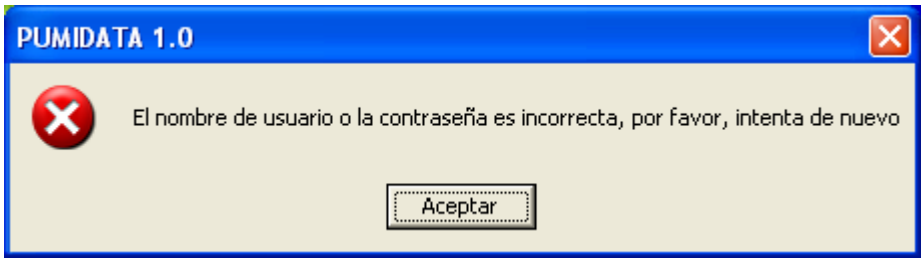


Figura 2. Mensaje de error. Nombre de usuario o contraseña incorrecto.

Si la clave y contraseña han sido correctas, ingresará al menú principal.

Para mostrar todas las opciones disponibles de Pumidata 1.0 se utiliza la cuenta del administrador del sistema, la cual cuenta con todos los permisos para realizar ajustes en todas las pantallas de captura.

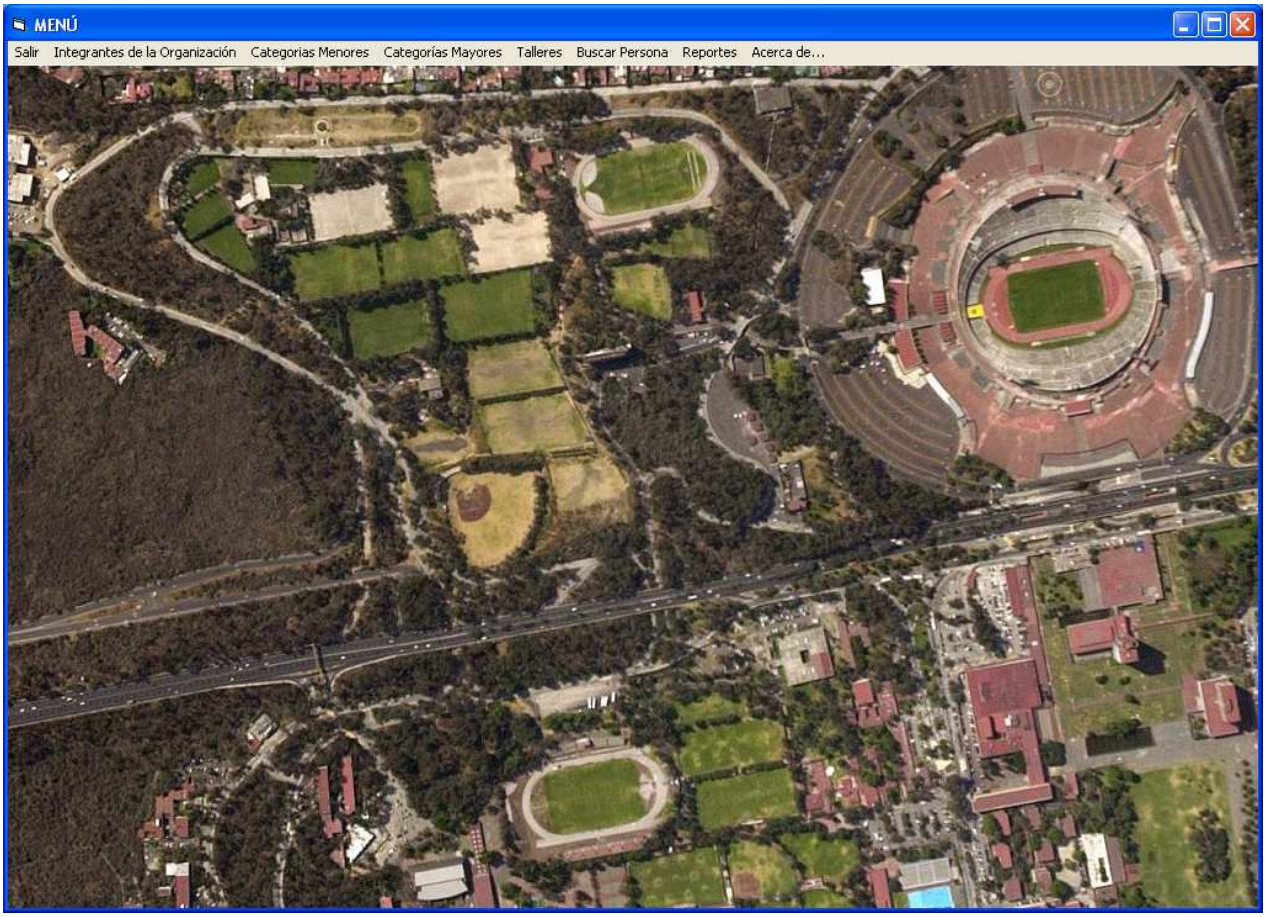


Figura 3. Menú principal.

En el menú principal, se observa los siguientes accesos:

- Integrantes de la Organización,
- Categorías Menores,
- Categorías Mayores,
- Talleres,
- Búsqueda y
- Reportes.

En el menú Integrantes de la Organización tenemos la información correspondiente a los miembros de Pumitas, así, por ejemplo, en el menú de Coordinadores Administrativos se cuenta con las siguientes opciones:

Figura 4. Coordinadores Administrativos.

En el menú de Coordinadores Técnicos se cuenta con las siguientes opciones:

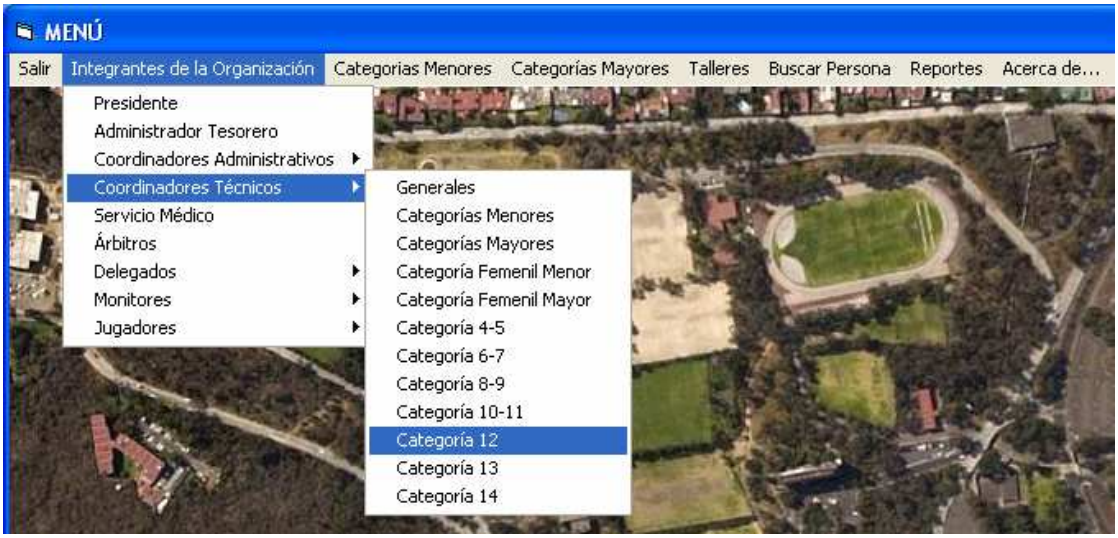


Figura 5. Coordinadores Técnicos.

Accesando en el menú de la Figura 5, se puede visualizar los Coordinadores Técnicos de la Categoría 12 años, en donde se tiene la siguiente información:

- Datos generales.
- Antecedentes.
- Relación de los equipos asignados a cada coordinador dependiendo del bloque asignado.
- Enlace a datos médicos.

Figura 6. Coordinadores de la Categoría de 12 años.

En el menú Delegados se cuenta con las siguientes opciones:

Figura 7. Delegados.

De igual forma, en el menú Monitores se tiene las siguientes opciones:

Figura 8. Monitores.

Lo mismo sucede con Jugadores:

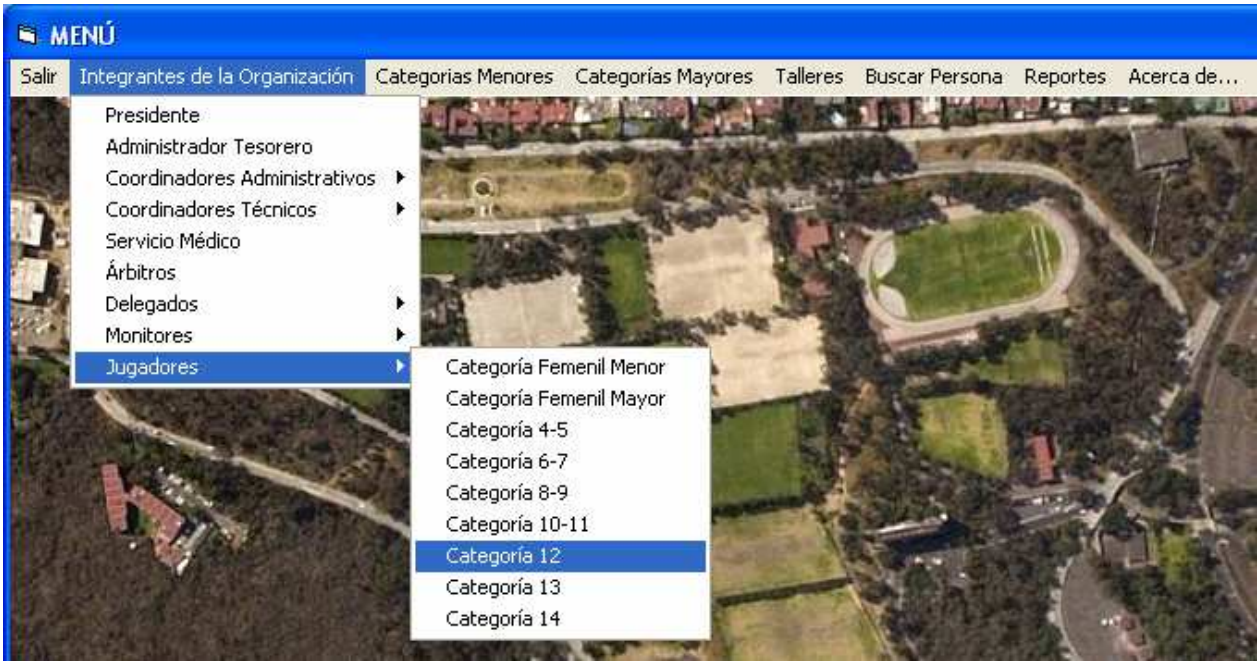


Figura 9. Jugadores.

Otra forma de acceder a la información se presenta utilizando el menú Categoría Menores:

Figura 10. Categorías Menores.

O bien, utilizando el menú Categorías Mayores:

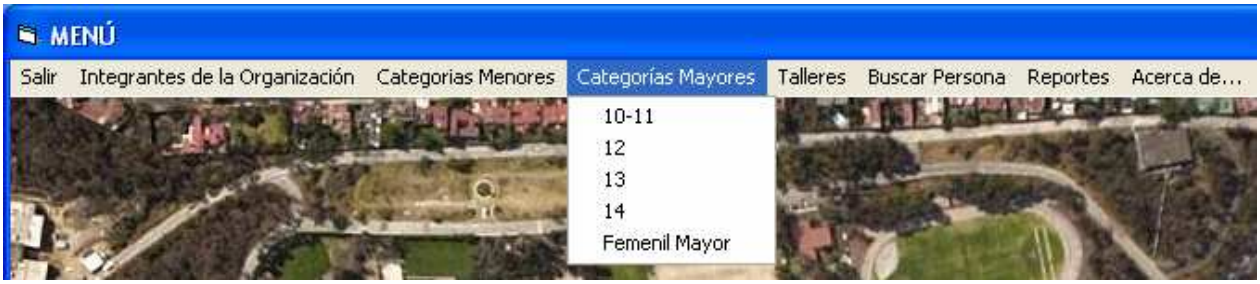


Figura 11. Categorías Mayores.

Igualmente, el seguimiento adecuado de los jugadores se continua en el menú Talleres, donde se tendrá actualizada la información de la asistencia de los jugadores a cada taller (Acondicionamiento Físico, Motricidad, Multilateralidad o Porteros) por categoría, para tener un seguimiento del trabajo desarrollado a lo largo de la temporada.

Figura 12. Talleres.

Pumidata cuenta con la presentación de Reportes con las siguientes modalidades:

Figura 13. Reportes.

La ficha informativa del sistema se muestra en el menú Acerca de:

Figura 14. Acerca de Pumidata 1.0.

Al elegir el menú opción Salir, de la pantalla principal o de cualquier otra pantalla que nos muestre esta opción, el sistema aprovecha para brindarle la oportunidad al usuario de realizar una copia de respaldo de la base de datos, como estrategia de seguridad, para evitar la pérdida de información:

Figura 15. Respaldo.

Al aceptar, le da la opción al usuario de seleccionar la ubicación de destino donde desea tener el respaldo, así se tiene:

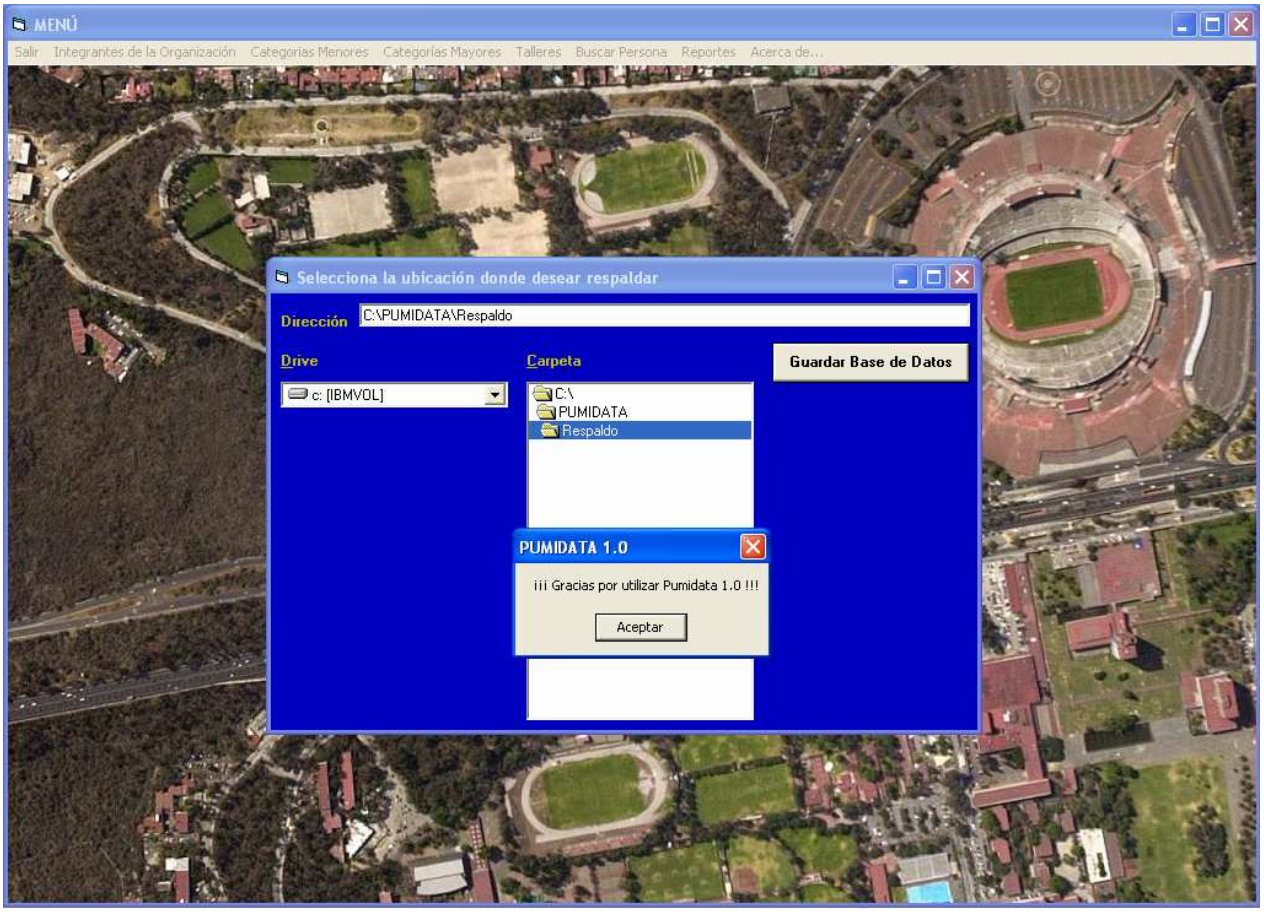


Figura 16. Guardar respaldo y salir.

El manejo de la información generada por las distintas categorías presenta el mismo esquema, así que echaremos un vistazo del funcionamiento del sistema, teniendo como punto de referencia la Categoría de 12 años. La información que se presenta es:

- Datos generales.
- Enlace a datos técnicos.
- Enlace a datos médicos.

Figura 17. Datos Generales de los Jugadores de la Categoría 12 años.

Altas. Si algún jugador no se encuentra dentro de la lista, podemos agregarlo y confirmar que queremos guardar el registro nuevo, no sin antes validar los datos, es decir, evitar mensajes de error por inconsistencia de datos ya que existen campos que sólo permiten cierto tipo de datos, por ejemplo, si un campo es numérico, sólo permitirá números, y si tratamos de ingresar algún otro tipo de dato, el sistema nos mostrará un mensaje de error.

El sistema también nos ayuda insertar de forma rápida textos predefinidos, a través de combos desplegables. Una vez que se ha realizado la captura de los datos, se oprime guardar, así, la información se guardará en la base de datos. Pero si se oprime cancelar no guardará nada.

Bajas. Cuando se elige eliminar algún registro de la base de datos, Pumidata 1.0 nos pide que se les confirme si en verdad queremos eliminar ese registro, ya que después de borrado un registro es imposible recuperarlo.

En caso de agregar un registro nuevo es necesario agregar la foto del jugador, y en caso de modificación, se permite cambiar la foto del jugador del registro activo. Para esto, se oprime en el botón Asignar foto y Pumidata 1.0 nos abrirá una nueva pantalla en la que nos pedirá el tipo de archivo, la ubicación de la foto, y nos mostrará la vista previa de la imagen seleccionada.

Tal imagen debió haberse obtenido previamente por un proceso de digitalización, ya sea por un escáner o directamente por una cámara digital conectada al equipo de cómputo. Las imágenes con formato “jpg” son las más recomendadas debido a que su tamaño es pequeño, es fácil de transportarlas, y hace más eficiente el proceso de cargarlas a las pantallas correspondientes.

Figura 18. Almacenar foto del jugador.

Podemos revisar los Datos Técnicos del jugador.

- Historial deportivo del avance por las distintas categorías y equipos por los que ha incursionado en Pumitas.
- Evaluación del jugador.
- Asignación a talleres.

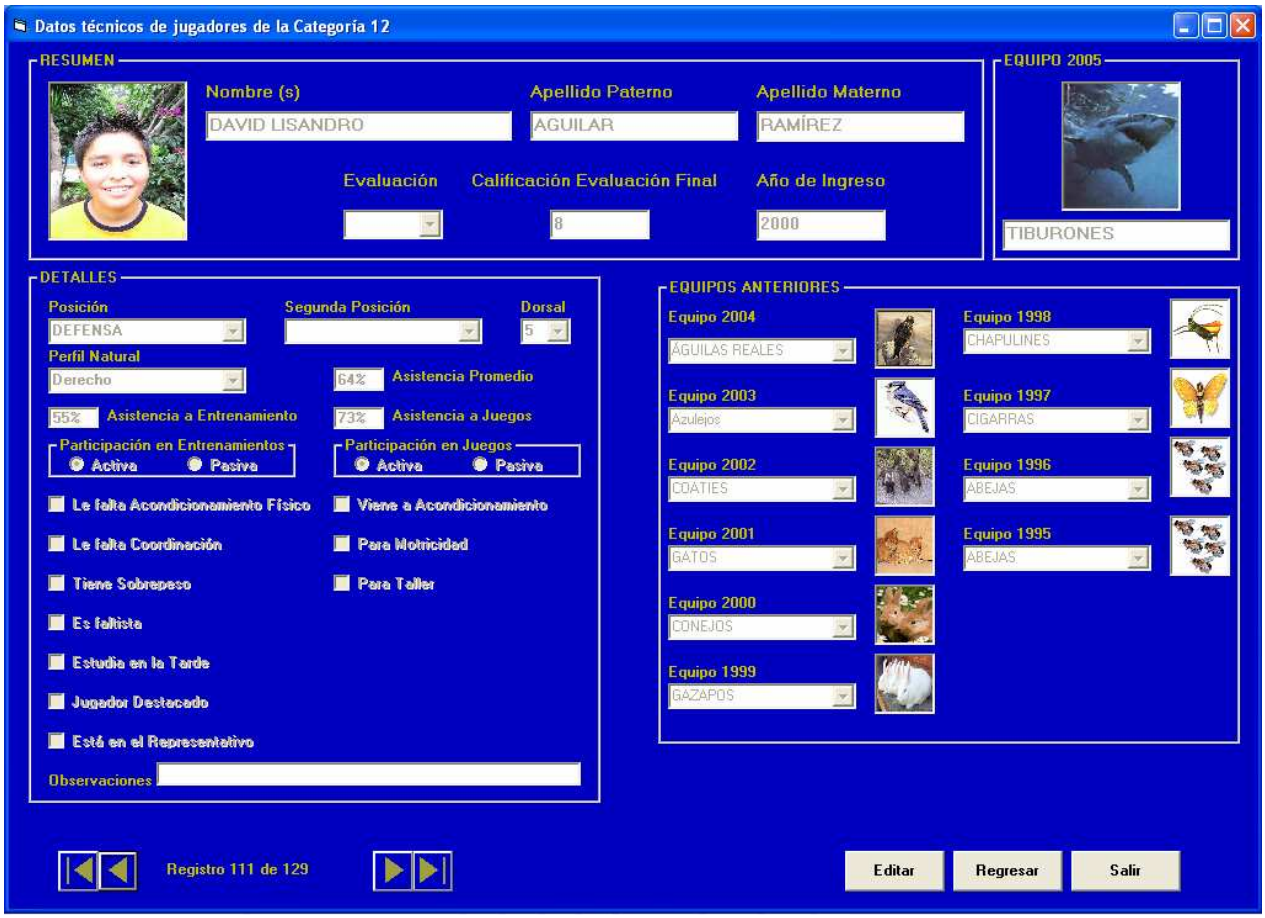


Figura 19. Datos Técnicos e Historial Deportivo de los jugadores de la Categoría 12 años.

En la sección Detalles, se pueden activar algunas de las casillas, como son: Acondicionamiento Físico, Motricidad o Taller, al activarlas, el sistema agrega al jugador del registro activo en lista de asistencia de los talleres activados y al consultar los talleres de su categoría aparecerá en la lista de los talleres a los cuales se le asignó al jugador.

También se puede acceder a sus datos médicos:

Figura 20. Datos y antecedentes Médicos del jugador.

Relación de los jugadores por equipo, dependiendo de la categoría.

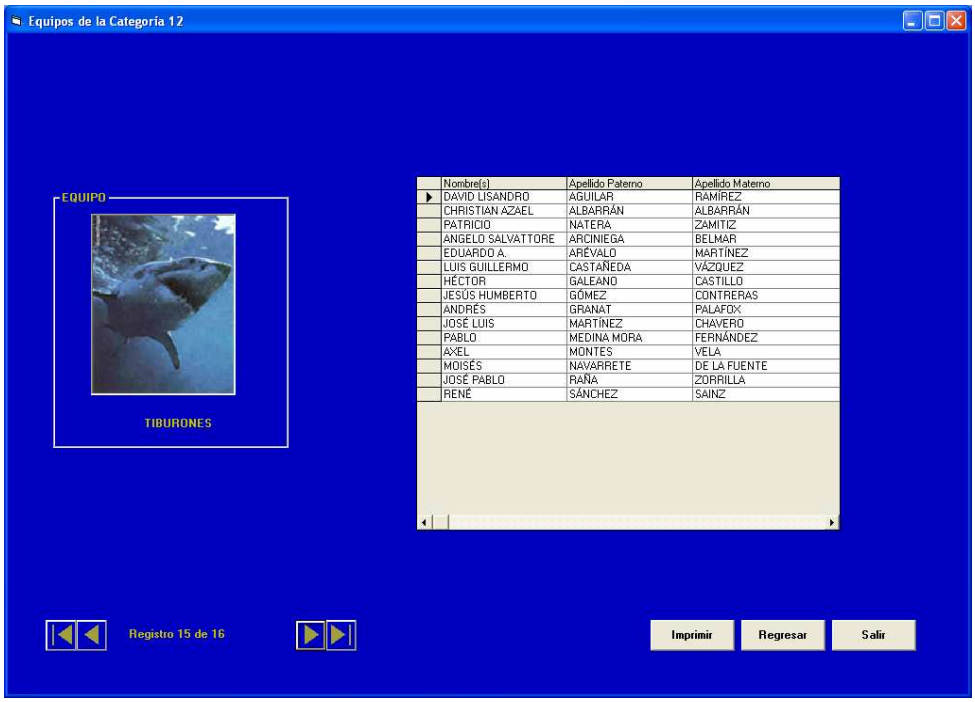


Figura 21. Jugadores por equipo de la Categoría 12 años.

Coordinadores de Categoría.

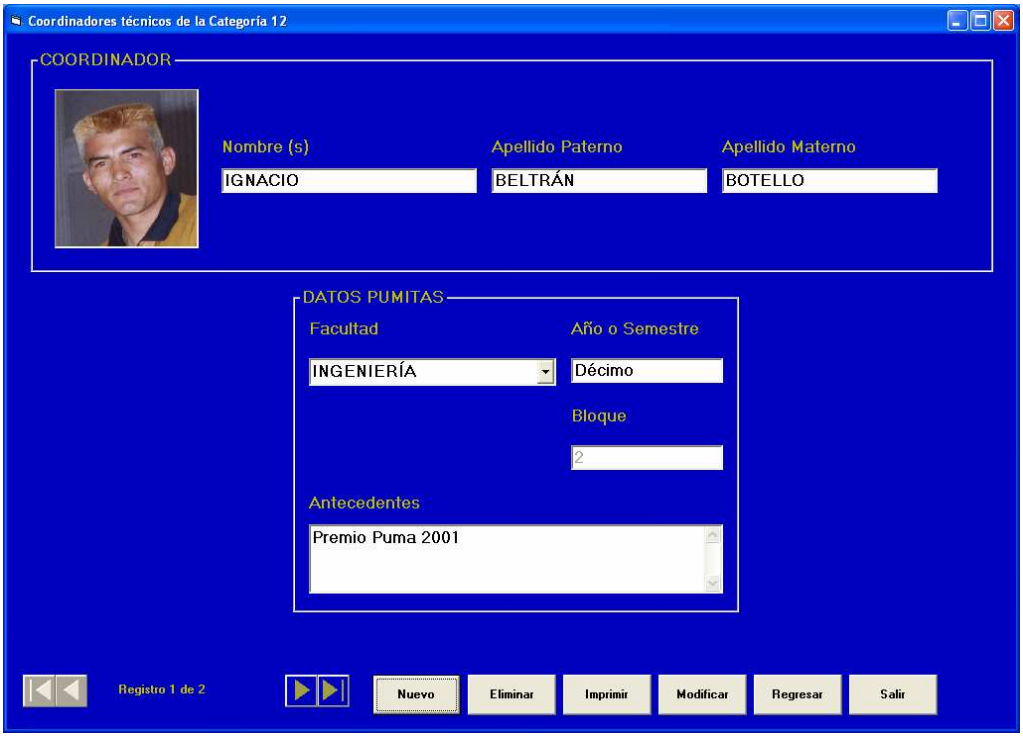


Figura 22. Datos de los Coordinadores Técnicos de la Categoría 12 años.

Monitores.

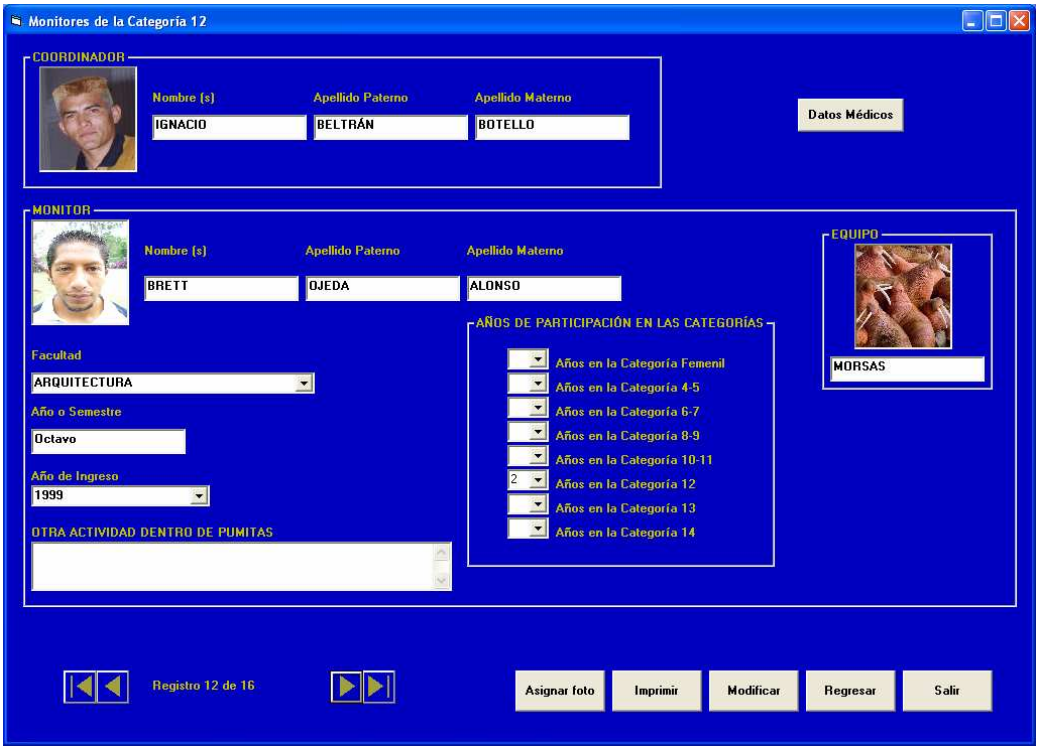


Figura 23. Datos de los Monitores de la Categoría 12 años.

En la pantalla anterior se cuenta con la siguiente información:

- Historial de participaciones en las distintas categorías, es decir, el número de años que ha participado en cada una de ellas, y el equipo con el que actualmente se encuentra.
- Datos escolares.
- Datos generales.

Equipos:

- Características.
- Monitor.
- Coordinador.
- Enlace a la lista de asistencia de los integrantes del equipo.

Figura 24. Datos de los Equipos de la Categoría 12 años.

Cargar Escudo del Equipo:

Figura 25. Escoger Logotipo del equipo.

Cargar Foto del Equipo:

Figura 26. Elegir Foto del Equipo.

Asistencia. En este punto, Pumidata permite tener enlace con Excel para facilitar el uso de la información de los monitores, ya que, ellos tendrán acceso a las planillas generadas en Excel por Pumidata 1.0, las cuales irán llenando durante el transcurso de la temporada y serán regresadas mes con mes para actualizar la información y poder tener un sustento necesario para determinar la necesidad o no de ayudar al equipo con más ingresos de jugadores.

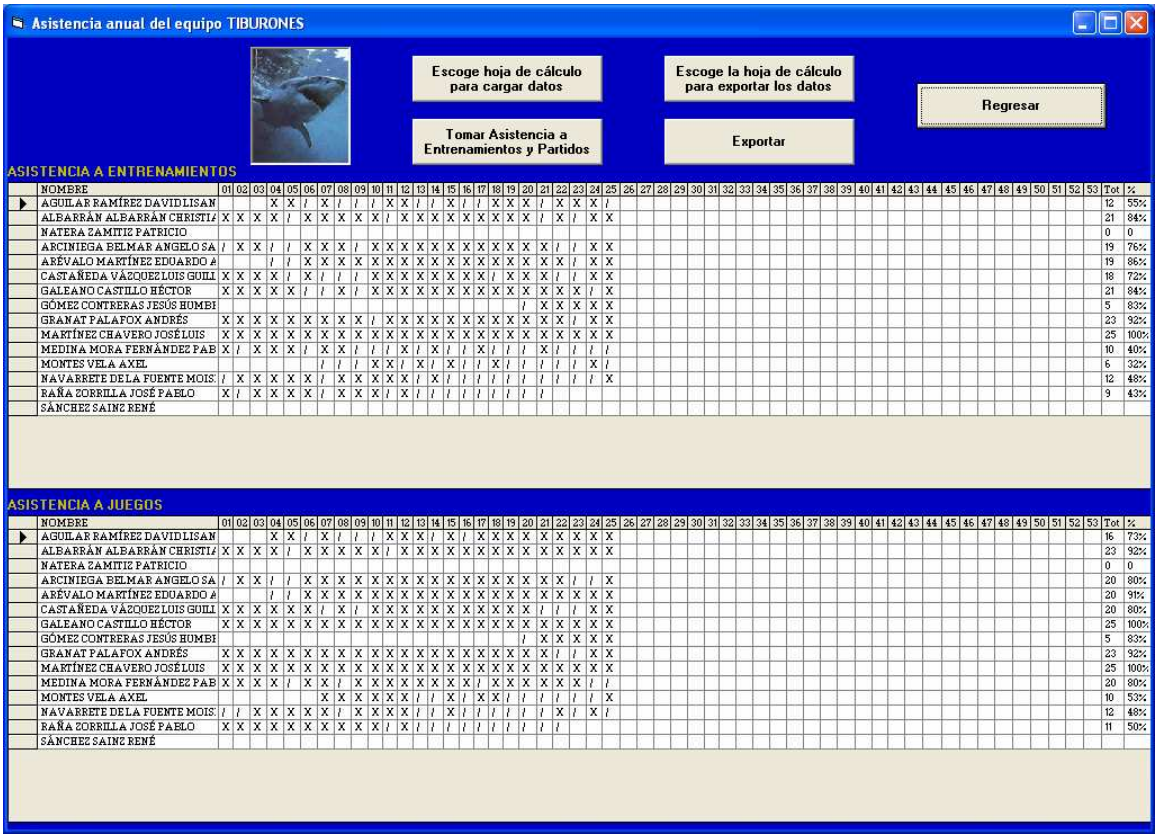


Figura 27. Asistencia Anual del Equipo Tiburones de la Categoría 12 años.

Selecciona hoja de cálculo:

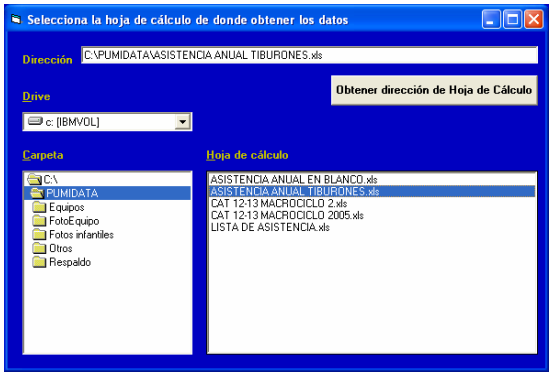


Figura 28. Selección de la Hoja de Cálculo en Excel de donde se van a extraer los datos.

El archivo de Excel debe cargarse del equipo de cómputo de Pumitas, previo almacenamiento por medio de disco que entregarán los monitores con la debida asistencia de su equipo.

Selecciona para exportar:

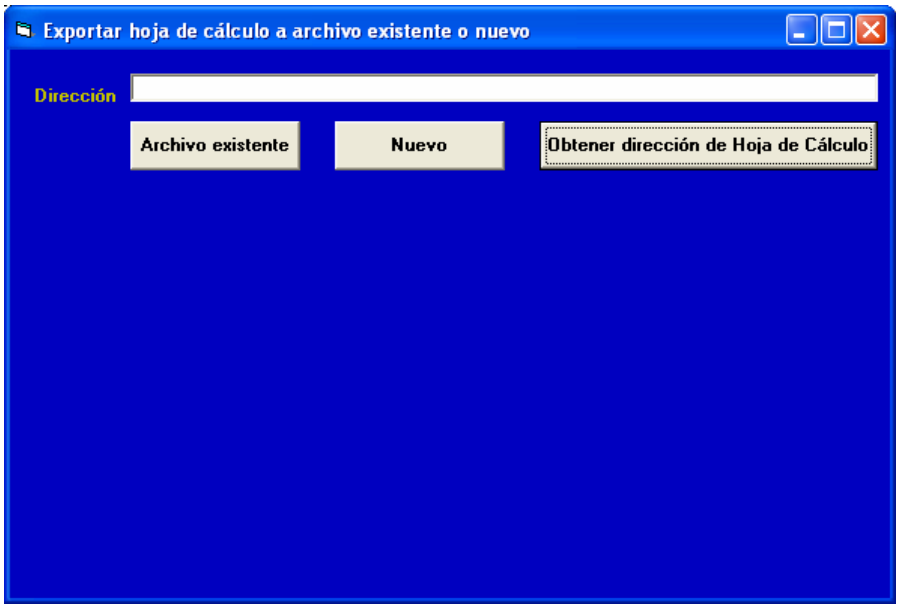


Figura 29. Elegir archivo de Excel para exportar datos.

Si es un archivo existente:

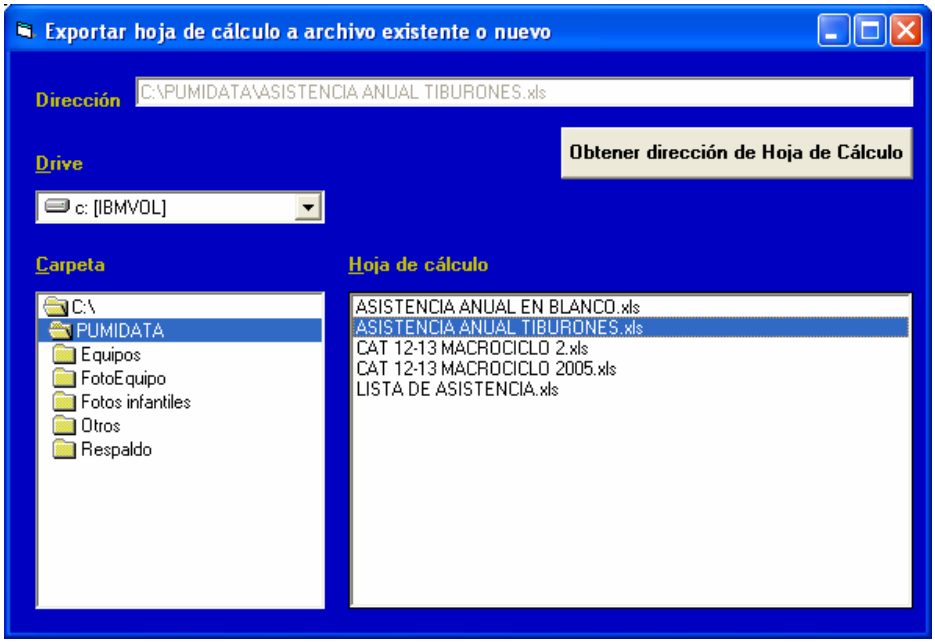


Figura 30. Elegir archivo existente de Excel para exportar datos.

O si se crea un archivo nuevo:

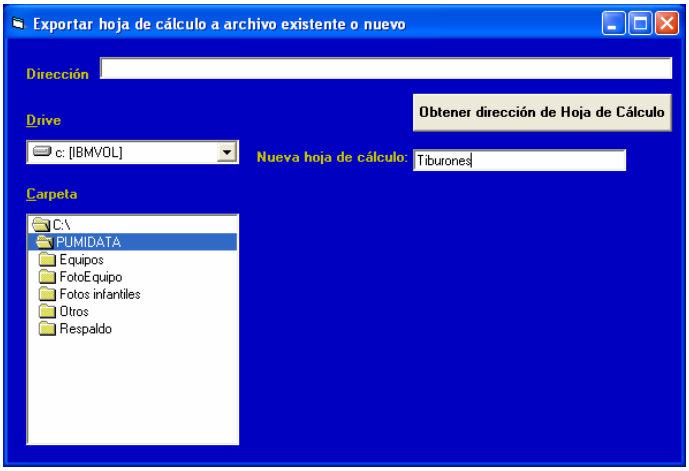


Figura 31. Elegir ubicación y escribir nombre del nuevo archivo en Excel para exportar los datos.

El resultado de exportar la asistencia del equipo genera el siguiente archivo de Excel, el cual puede ser fácilmente manipulado por el monitor, teniendo la posibilidad de entregárselo en disco o enviarlo por correo electrónico.

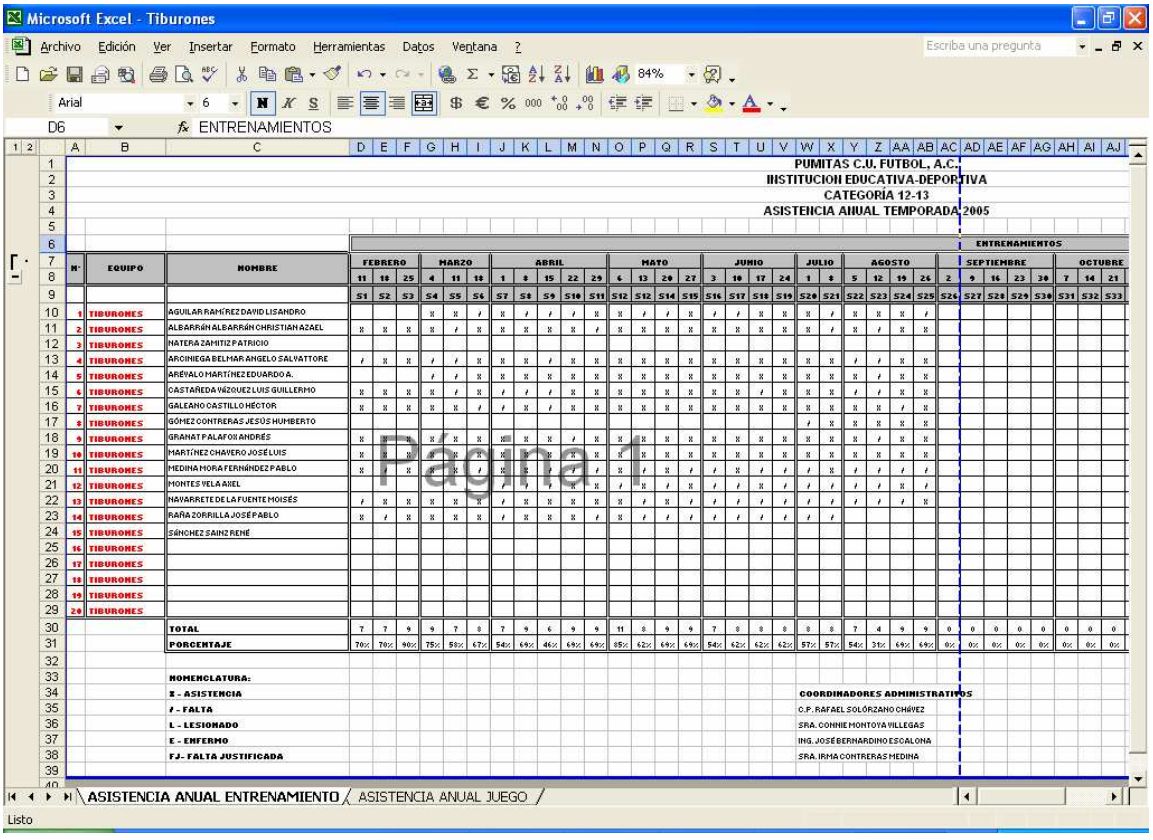


Figura 32. Archivo generado después de exportar los datos.

Búsqueda. Cuando deseamos buscar a alguien en especial, anotamos la cadena de texto (se puede utilizar caracteres comodines como ? y *) que buscamos pudiendo buscar ya sea por apellido paterno, materno o nombre, y obteniendo el equipo y categoría en donde se encuentra el jugador o la persona buscada.

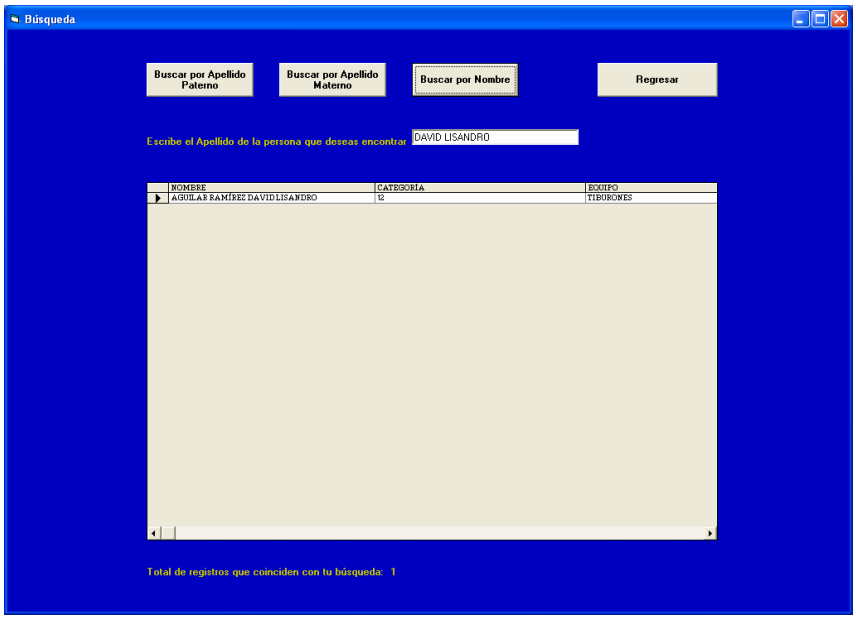


Figura 33. Búsqueda de personas dentro de Pumitas.

Búsqueda con caracteres comodines:

Figura 34. Búsqueda utilizando caracteres comodines.

Ejemplo de Generación de Reportes:

EQUIPO	APELLIDO PATERNO	APELLIDO MATERNO	NOMBRE (S)	POSICIÓN
BALLENAS	BARRAGÁN	MORALES	ENRIQUE	MEDIO
BALLENAS	DEL CANTO	SÁNCHEZ	ROBERTO DE JESÚS	MEDIO
BALLENAS	GONZÁLEZ	MARTÍNEZ	IVÁN YAHIR	PORTERO
BALLENAS	MARTÍNEZ	ALVARADO	RICARDO	MEDIO
BALLENAS	OCARANZA	VALDOVINOS	PABLO ALEJANDRO	DEFENSA
BALLENAS	ROMERO	MOTA	PATRICIO	DEFENSA
BALLENAS	ZAMORA	CAMACHO	DOMINGO	DEFENSA
BARRACUDAS	CISNEROS	PARTIDA	ANDRÉS ARAM	DEFENSA
BARRACUDAS	IBARRA	ACOSTA	PEDRO	DELANTERO
BARRACUDAS	RIVERO	ÁVILA	JUAN LUIS	DELANTERO
BARRACUDAS	RODRÍGUEZ	SÁMANO	MIGUEL	DEFENSA
CALAMARES	BAEZ	PEDRAJO	ALONSO	MEDIO
CALAMARES	RAMÍREZ	VELÁZQUEZ	MIGUEL A.	DEFENSA
CAMARONES	ESCANDÓN	BETAN	HÉCTOR MISAEL	MEDIO
CAMARONES	VALDÉS	MEDINA	ADRIÁN	DEFENSA
CANGREJOS	MÉNDEZ	RODRÍGUEZ	PEDRO	PORTERO
CANGREJOS	REYES	SOLLEIRO	ADOLFO	PORTERO
CAREYES	RAYGOZA	SOLÍS	RODOLFO	DEFENSA
CAREYES	RÍOS	MINOR	OSCAR ULISES	PORTERO
DELFINES	GUTIÉRREZ	TORRES	JUAN EDUARDO	DEFENSA
DELFINES	VALDERRAMA	GIL	J. EDUARDO	DEFENSA
HIPOCAMPOS	PIREZ	TORRES	DANIEL ALFONSO	MEDIO

Figura 35. Reporte de los Integrantes de los equipos de la Categoría 12 años.

Exportar el Reporte:

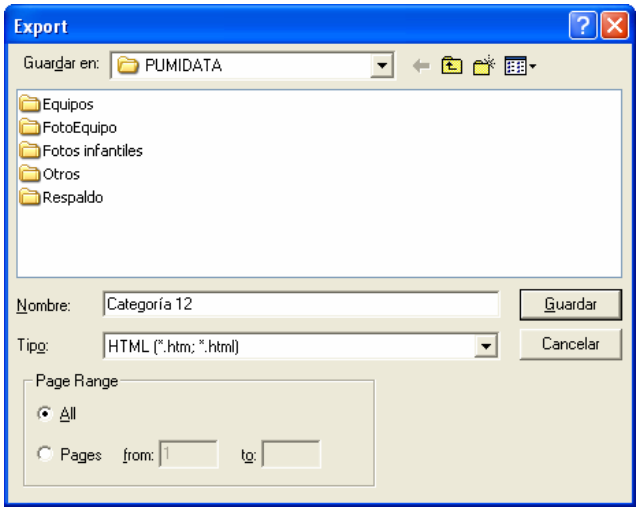


Figura 36. Exportar el reporte como archivo html o como archivo de texto.

PUMIDATA 1.0

BIBLIOGRAFÍA.

Bases de datos con Visual Basic 6.
Jeffrey P. McManus.
Editorial Prentice Hall.

Programming Microsoft Visual Basic 6.
Francesco Balena.
Microsoft Press.

Microsoft Visual Basic 6.
Lenguaje Referente.
Microsoft Press.

Curso de programación de Visual Basic 6.
Francisco Javier Ceballos.
Alfaomega.

Manual Práctico de Organización Deportiva.
Óscar Martín Andrés.
Gymnos Editorial.

Manual para la Organización y el Entrenamiento en las Escuelas de Fútbol.
Ález Sans Torrelles y César Frattarola Alcaraz.
Editorial Paidotribo

PÁGINAS WEB.

Pumitas C.U. Fútbol, A.C.:
http://www.pumitasfutbol.unam.mx

Apuntes de Ficheros y Bases de Datos:
http://www3.uji.es/~mmarques/f47/apun/apun.html

Modelo relacional:
http://www.galeon.com/apuntesbd/enlaces793753.html

Modelo relacional:
http://www.programacion.com/tutorial/modrel/5.html

Administración de proyectos de desarrollo de sistemas de información:
http://www.gestiopolis.com/recursos2/documentos/fulldocs/ger/adproysisinf.htm