Benemérita Universidad Autónoma de Puebla

FACULTAD DE CIENCIAS DE LA COMPUTACION

BASE DE DATOS
COMO HERRAMIENTA DE CONSULTA
EN LA EDUCACION BASICA

TESIS

PARA OBTENER EL GRADO DE
LICENCIADO EN CIENCIAS DE LA COMPUTACION

PRESENTA:
OJILVIE AVILA CORTES

ASESORA:
M. C. BEATRIZ BERNABE LORANCA

Puebla, Puebla. Otoño de 2007.

Nehhuatl, Ojilvie Avila Cortés, ni mexihca quemen nohpallitl, icanon, nic pia huei mahuiliznelhuayotl inpampa to huehcacoltzintzin tolteca huan azteca, axcan nic nequi nic tlazohcamatiz ninque mahuiliztlacatzintzin aquinque o nech macahquen no nemiliz, no mahuiliz huan matiliz, tlen o chihquen nin tlacatl nopampa. Nican nan mech tlahpalohua ican nin nahuatlahtolli tlen o nic zalo itech ineca teotlahtzintli itech metztli mayo huan xihuitl macuilli cempoalli huan ce ican no temachtianitzin, axcan no teopanquixtihtzin.

Tlenica:

No cochmaihnitzin:	*Karina García (ni mitz tlazohtlaz… cenca)*
No nantzin:	*María Luisa Cortés (man xic zalo to nemiliz)*
No tahtzin:	*Ojilvie Avila (axto),*
To cihuateopanquixtihtzin:	*Elodia Popoca (ahmo man xi yolpatzmiqui),*
To teopanquixtihtzin:	*Genaro Medina (cenca yolpaqui),*

Miac tlazohcamati…

Axto ni mo tlazohcamati nanmo ixpantzinco, nehhuatl niquin yolehua nochtin mexihca tlazalozquen nin cuacualtzin huan nelmelahuac tlahtolli (nozo ohze itech nin mexicatlalpan) tlen qui temaca nelyolmelahualiztli tlen tech cotona itech papalotl nemiliztli… huan, icuac tic pian nahuatlahtolli, tehhuan ti cualtizquen tlanonotzazquen ica to teco Teotzin, tlalticpac, huan nochi cemanahuac…

Yo, Ojilvie Avila Cortés, soy mexicano como el nopal, por eso, tengo una grande raíz cultural a causa de nuestros honorables ancestros toltecas y aztecas, hoy quiero agradecer a estas cultas personas quienes me dieron mi vida, mi honor y conocimiento, que hicieron este hombre en mí. De aquí les saludo con esta lengua nahuatl que aprendí desde aquella tarde del mes de mayo del año 2001 con mi culto maestro, hoy mi venerado padrino.

Para:

Mi venerada esposa:	*Karina García (te amaré… siempre),*
Mi respetable madre:	*María Luisa Cortés (aprende nuestra historia),*
Mi respetable padre:	*Ojilvie Avila (primero),*
Nuestra culta madrina:	*Elodia Popoca (ya no esté triste),*
Nuestro culto padrino:	*Genaro Medina (siempre contento),*

Muchas gracias…

Antes de que me despida de su honorable presencia, yo quiero invitar a todos los mexicanos que aprendan esta hermosa y verdadera lengua (u otra de esta tierra mexicana) que da un corazón honesto lejos de la mentira de la historia… y, cuando tengan la lengua nahuatl, ustedes podrán conversar con nuestro amo Dios, la naturaleza, y todo el universo…

Indice general.

Introducción.

El presente proyecto involucra el desarrollo de una herramienta de apoyo a la educación básica, considerando el nivel primaria y posiblemente secundaria, aunque no se descarta el uso del mismo en general. Originalmente se plantea el uso de cinco materias: español, matemáticas, historia, ciencias naturales y civismo, sin embargo, se construye con el propósito de que pueda funcionar para más o menos materias, dependiendo de las necesidades de uso. Para todo lo anterior, se determinó construir un software que tenga como punto principal un diseño de almacenamiento robusto y de procesos de consulta eficientes. Este diseño servirá como patrón de referencia para que información relacionada con el enfoque que ofrecen los términos de educación básica sirva en proyectos de enseñanza. La propuesta adicional de este proyecto es el desarrollo de un prototipo de un motor de búsqueda basado en SQL. El usuario tendrá a su disposición la posibilidad de inspeccionar la información de interés, además de la búsqueda tradicional por temas de recorrido en pantalla.

Actualmente no se tiene pensado una implementación en campo, es decir, que sea probado directamente por usuarios (niños de entre 6 y 12 años) hacia quienes este sistema está orientado, no obstante, no se descarta el uso para una retroalimentación.

Así es como se ha nombrado el proyecto:

Base de datos como herramienta de consulta en la educación básica.

Ahora bien, los objetivos a perseguir con este sistema son los siguientes:

- En el presente proyecto se pretende que la herramienta de software a desarrollar contribuya y apoye la enseñanza a nivel básico.
- El sistema puede ser aplicado a cualquier área requerida, dado que posee un programa motor de búsqueda para ejecutarse sobre una base de datos que tenga la información a consultarse.
- Se aplicará teoría de bases de datos en el diseño y construcción de la base de datos a ser utilizada por el sistema.
- Se aplicará metodología propia de ingeniería de software en la programación del mismo.
- Se especificará el propósito de contar con este trabajo como una alternativa de la multimedia así como también el SGBD elegido para la construcción del proyecto.
- Las conclusiones y perspectivas deben integrarse en la documentación de este proyecto así como la ingeniería de software asociada a ello.
- En consecuencia se espera que este sistema funcione como una excelente herramienta de trabajo para la consulta de diferentes temas de las materias en los niveles ya mencionados.

- Se deberá investigar y estudiar herramientas de motores de búsqueda para ser implantados o adecuados al sistema.
- Se pondrá énfasis en la construcción del diseño para que éste sea aprovechado en temas relacionados a la educación básica

Para finalizar esta introducción, se describe brevemente que este documento se divide en cuatro capítulos y cuatro anexos. En el primer capítulo se incluyen detalladamente, conceptos, técnicas y herramientas de diseño de ingeniería de software, bases de datos y programación orientada a objetos. Para el segundo, se plantea y analiza el sistema desde el punto de vista de las herramientas descritas en el capitulo uno para lograr una solución al problema en cuestión. En el tercero abarca el diseño del sistema y normalización de la base de datos. El cuarto contiene la programación y pruebas de un prototipo totalmente funcional del sistema. El anexo A describe las operaciones del álgebra relacional con ejemplos, que no fueron incluidos en el primer capitulo. En el anexo B, se dan instrucciones básicas de la configuración del controlador de la base de datos que utiliza el sistema bajo Windows. El anexo C es el listado del código fuente del prototipo del sistema, que incluye todas las rutinas (o “formas”) utilizadas. Y por último, el anexo D, presenta en detalle las entidades, relaciones, el diccionario de datos y un diagrama entidad-relación y un esquema relacional de la base de datos pero que incluye todas las relaciones y entidades usadas por el sistema.

Capítulo 1. Herramientas teóricas de análisis y diseño del sistema.

A continuación se describen las metodologías necesarias para llevar a buen término un sistema de software en general. También se incluyen definiciones propias del tipo de base de datos que este sistema utilizará, así como de motores de búsqueda y su funcionamiento, herramientas de análisis, diseño y programación de una base de datos. Todo lo anterior con el fin de utilizarse en capítulos subsecuentes para el desarrollo del sistema a implementarse. Como nota, se debe aclarar que las fuentes citadas en la sección de bibliografía son más que las citadas en este marco teórico, sin embargo, fueron consultadas por el autor para el estudio y realización de este proyecto

1.1. Introducción a la ingeniería de software.

1.1.1. Introducción.

La complejidad creciente de los sistemas de cómputo y su penetración en la sociedad hace cada vez más evidente la necesidad de enfoques sistemáticos para el desarrollo de software así como para su mantenimiento. En las últimas décadas se ha hecho especial énfasis en la importancia de contar con una tecnología para el desarrollo del software. La ingeniería de software es el campo de estudio relacionado con esta tecnología.

1.1.1.1. Definiciones básicas

De acuerdo con Boehm, (Boehm, 1), la ingeniería de software incluye "la aplicación práctica del conocimiento científico en el diseño y construcción de programas para computadoras y la documentación asociada requerida para desarrollarlos, operarlos y mantenerlos".

Las metas primordiales de esta disciplina tecnológica son mejorar la calidad de los productos de programación y aumentar la productividad y satisfacción profesional de los ingenieros de esta disciplina.

La calidad de los programas es una preocupación primordial de los ingenieros de programación, las características importantes de la calidad dependerán del producto en particular. Sin embargo, existen algunas características de la misma que son fundamentales en todo producto de programación: la utilidad, la claridad, la confiabilidad, la eficiencia y la economía.

El factor más importante de la calidad de un producto es su utilidad, es decir, que el producto de programación satisfaga las necesidades del usuario. Muchos paquetes entregados a los usuarios con frecuencia no desempeñan las funciones esperadas; este problema es síntoma de la pobre comunicación existente entre el cliente, los usuarios y los ingenieros de programación. La

planeación cuidadosa, el análisis y la participación del cliente son indispensables para el desarrollo de productos de programación útiles.

La confiabilidad del producto esta definida como la "capacidad de un programa para desempeñar una función requerida bajo ciertas condiciones durante un tiempo especifico" (IEEE, 2). El grado de confiabilidad deseado en un producto particular puede ser expresado en términos del costo de la falla del producto. La cantidad de esfuerzo empleado en obtener confiabilidad debe ser función del costo de las imperfecciones del producto; sin embargo, en cualquier caso, existe un nivel mínimo de confiabilidad que todo producto debe poseer.

Los productos de programación deben estar escritos con claridad y ser fáciles de entender. Las pruebas y actividades de mantenimiento consumen gran cantidad del presupuesto del proyecto. La clave para realizar un sistema fácil de probar y mantener radica en hacerlo comprensible; los productos de programación que se presenten a los usuarios deben tener una integridad conceptual y ser claros en el propósito del proyecto.

Un producto de programación deberá ser eficiente, tanto como la aplicación particular lo amerite. En los primeros días de las computadoras digitales el equipo electrónico era muy costoso y lento, si se consideran las normas actuales; por esto se hacía hincapié en aprovechar al máximo el ciclo de operación de la memoria, de modo que la característica mas importante del producto era su eficiencia. Conforme los productos se hacen más complejos y grandes, los atributos de utilidad, confiabilidad y claridad van cobrando prioridad en ellos.

Por otro lado, existen productos que están limitados críticamente por el tamaño de memoria y la velocidad de ejecución (sistemas de tiempo real), en estos casos la eficiencia permanece como el principal atributo.

Finalmente, un producto debe ser costeable en su desarrollo, mantenimiento y uso. Un producto de programación debe desempeñar en su empleo diario una tarea específica usando menos tiempo o menos recursos humanos o industriales que los que se requerían antes de tenerlo.

Los grandes sistemas de software requieren un tiempo considerable para su desarrollo y permanecen en uso durante un tiempo aun mayor. En estos periodos de desarrollo y uso pueden identificarse varias etapas que constituyen lo que se llama el Ciclo de Vida del Software.

1.1.1.2. Ciclos de vida.

Un "macro modelo" del ciclo de vida es el siguiente (evolución de software de acuerdo a Lehman, 3):

1. Análisis y definición de necesidades. Los servicios, restricciones y objetivos del sistema, se establecen consultando con los usuarios. Una vez acordados, deben definirse de una manera comprensible, tanto para los usuarios como para el personal de desarrollo. En esta primera fase podemos distinguir las actividades de: Planeación y Especificación de Requisitos.

2. Diseño del sistema. El diseño de un sistema de software es el proceso de representar las funciones de este a fin de poderlo transformar con facilidad en uno o más programas.

3. Instrumentación y pruebas de unidades. Durante esta etapa, el diseño del software se instrumenta como un conjunto de programas o unidades de programa escritos en algún lenguaje de programación. Las pruebas de unidades implican comprobar que cada unidad cumple con su especificación.

4. Pruebas del sistema. Las unidades de programa se integran y se prueban como un sistema completo para asegurar que se cubren las necesidades del software.

Después de las pruebas, el sistema se envía al cliente para su aceptación.

5. Operación y mantenimiento. Esta fase suele ser la mas larga del ciclo de vida. Se instala el sistema y se pone en uso. La actividad de mantenimiento implica corregir errores que no se descubrieron en las primeras etapas del ciclo de vida, mejorar la aplicación de las unidades del sistema y aumentar los servicios de este a medida que se detectan nuevas necesidades.

1.1.2. Planeación.

El principal propósito de esta fase es aclarar las metas del proyecto, las necesidades del cliente y las restricciones del producto. El proceso de planeación puede analizarse paso a paso. En general, los tres aspectos a cubrir son: definición del problema, desarrollo de una estrategia de solución y planeación del proceso de desarrollo. Se describirán las principales actividades que es necesario realizar en cada uno de estos aspectos.

1.1.2.1. Desarrollar un enunciado definitivo del problema por resolver.

En este enunciado debe incluirse una descripción de la situación actual, de las restricciones del problema y de las metas que se espera lograr. Es necesario que este enunciado se realice empleando la terminología del cliente, ya que será el primer acercamiento a la detección de las necesidades de éste.

Para el caso específico de esta tesis, es indudable que los manejadores de bases de datos son uno de los productos de software que desde que aparecieron han servido como una herramienta muy útil para la organización de la información,

y no sólo eso, sino que las aplicaciones que de ellas derivan se han multiplicado gracias a los avances que los SGBD nos brindan para la gran facilidad en su utilización.

El común denominador de todas las operaciones de consulta es el diseño y construcción de un motor de búsqueda asociado a éste lo cual permitirá un rendimiento mayor en el acceso a la información.

1.1.2.2. Justificar una estrategia de solución computarizada para el problema.

Una estrategia de solución debe considerar todos los factores externos que son visibles para el usuario del producto, y debe redactarse de tal manera que permita caminos alternos para el diseño del producto.

La estrategia seleccionada proporciona un marco de referencia para el diseño y la instrumentación del producto de programación. Una estrategia de solución es factible si las metas y requisitos del proyecto se pueden satisfacer dentro de las restricciones de tiempo disponible, recursos y tecnología por medio de esa estrategia.

Las técnicas para determinar la factibilidad de una estrategia de solución comprenden el estudio de casos, análisis del peor caso, simulación y construcción de prototipos. Una estrategia de solución debe incluir una lista con prioridades de las características del producto, ya que en algún momento posterior en el ciclo de desarrollo puede necesitarse posponer o eliminar algunas de las capacidades del sistema debido a inconsistencias en los requisitos, excesos en tiempo o costo.

1.1.2.3. Identificación de consideraciones varias.

Hay varias partes que deben ser tomadas en cuenta antes de cualquier desarrollo para poder así, aceptar como viable un proyecto de programación, y estas pueden ser: la identificación de las funciones por realizar, las restricciones, el subsistema de equipo de cómputo, el subsistema de productos de programación, y el del personal involucrado.

Un sistema de programación está formado por los subsistemas de personal, equipo y de productos de programación, así como de las interrelaciones entre estos.

El primer subsistema incluye operadores, personal de mantenimiento y usuarios finales. El segundo comprende el equipo de cómputo y todo tipo de dispositivos. El tercer subsistema contiene programas que deben desarrollarse, más los programas que ya existen y que pueden utilizarse.

Es necesario identificar las funciones que cada subsistema principal debe realizar, establecer las interacciones entre subsistemas y determinar las restricciones en el desarrollo y operación para cada subsistema principal. Las limitaciones especifican número y tipo de equipo, cantidad y habilidades del personal, y características del producto de programación como funcionamiento, precisión y nivel de confiabilidad.

1.1.2.4. Determinar metas y requisitos preliminares.

Las metas sirven para establecer el marco de referencia para el proyecto de desarrollo del producto de programación. Las metas pueden ser cualitativas y cuantitativas.

Metas generales: Todo producto de programación debe ser útil, confiable, comprensible y eficiente en costos. Todo proceso de desarrollo debe producir productos finales a tiempo y dentro de los estimados de costo, así como permitir que el personal del proyecto desarrolle nuevas habilidades.

Los requisitos especifican las capacidades que debe tener un sistema para la solución de un problema. Se establecen requisitos de funcionalidad, de rendimiento de equipo, de programación, de interfaces a usuario y de control de calidad.

Los requisitos se utilizan como base para la aceptación del producto final. Cada módulo debe incluir el método que se empleará para su verificación.

En el caso de este proyecto se necesita que el sistema sea robusto, flexible y funcional.

1.1.3. Definición de requisitos.

A continuación se describen las necesidades que deben cumplirse antes de comenzar la planeación de un sistema de software.

1.1.3.1. Especificación de requisitos.

Es un documento que sirve como fundamento para la ingeniería, hardware, software, base de datos, e ingeniería humana. Describe la función y rendimiento de un sistema basado en computadoras y las dificultades que estarán presentes durante su desarrollo. Las especificaciones de los requisitos del software se producen en la terminación de la tarea del análisis.

Para formalizar la especificación de los requerimientos en el contexto del presente proyecto, se propone una técnica de especificación del tipo operacional, donde describimos los requerimientos sobre el sistema a través de detallar

comportamiento deseado (modelando el sistema), de acuerdo al tipo de modelo que corresponda, como será descrito en los siguientes párrafos.

Los requerimientos vinculados a la funcionalidad de un sistema, podrán ser especificados mediante diagramas de flujo de datos (Data Flow Diagrams, o DFD).

Los requerimientos vinculados a relaciones entre datos de un sistema serán especificados mediante el modelo entidad relación (Entity-Relationship Model MER).

Colaboración mutua: es conveniente durante el proceso de desarrollo, prestar especial atención a la designación de responsables de ambas partes (cliente-desarrollador) que ajusten los métodos para acordar los requisitos y la forma de aprobar cambios, las acciones de prevención de no conformidades, el registro y aprobación de resultados, y la validación de especificaciones como actividad previa al proceso de realización.

En el caso del presente proyecto, la metodología del modelado y los diagramas ya mencionados serán descritos con mayor detalle más adelante en este capítulo.

1.1.4. Diseño.

Un buen diseño implica la división del todo en partes que sean fáciles de manejar al momento de la programación, así como técnicas del mismo, así como los correspondientes diagramas que ayudan al desarrollo del sistema.

1.1.4.1. Modularidad.

Esta es la división mencionada del todo en partes, lo que facilita su programación ya sea por uno o más programadores.

Criterios de modularidad

Un método de diseño que merezca ser llamado *modular*, debería satisfacer cinco requisitos fundamentales (fuente de Internet, 15):

- Descomposición modular. Ayuda en la tarea de descomponer el problema en un número de sub-problemas menos complejos, interconectados.
- Composición modular. Favorece la producción de elementos de software que se puedan combinar libremente unos con otros para producir nuevos sistemas.
- Comprensibilidad modular. Ayuda a producir software en el cual un lector humano puede entender cada módulo sin tener que conocer los otros (o examinando sólo unos pocos).

- Continuidad modular. Se satisface si un pequeño cambio en la especificación de un problema provoca sólo cambios en un solo módulo o en un número pequeño de módulos.
- Protección modular. Si produce arquitecturas en las cuales el efecto de una situación anormal ocurrida en un módulo durante la ejecución, queda confinado a ese módulo (o se propaga sólo a unos pocos vecinos).

Principios de modularidad

De los criterios anteriores, podemos resumir que los principios de modularidad son (fuente de Internet, 16):

- Un sistema complejo debe ser dividido en partes.
- Permite aplicar el principio anterior en dos fases:
    - detalles de cada parte sin tomar en cuenta las restantes
    - relaciones entre partes sin tomar en cuenta los detalles
- No sólo aplica a los aspectos estructurales, sino a todo el proceso de desarrollo.
- Se basa en: descomposición, composición y comprensión.
- Cohesión y acoplamiento.

Metodologías de diseño de software

Existen (de acuerdo a fuente de Internet 18) muchas metodologías de diseño. Describiremos brevemente sólo una clase de ellas, llamado métodos de descomposición. El diseño de un sistema de software implica la descomposición del sistema en partes de menor tamaño, cada una de las cuales puede refinarse en forma independiente. Dentro de los métodos de descomposición, mencionaremos los siguientes dos:

- Descomposición algorítmica: Corresponde al proceso de dividir el sistema en partes, cada una de las cuales representa un pequeño paso de un proceso más grande. La aplicación de métodos de diseño estructurado llevan a una descomposición algorítmica, cuyo foco está puesto en el control de flujo del sistema.
- Descomposición orientada a objetos: Corresponde al proceso de dividir el sistema en partes, cada una de las cuales representa una clase u objeto del dominio del problema. La aplicación de métodos orientados a objetos llevan a descomposición orientada a objetos, en la cual se observa al mundo como una colección de objetos que cooperan entre ellos para obtener la funcionalidad deseada.

Diseño detallado

Dado que el presente proyecto se planea como orientado a objetos, su diseño considera el proceso de descomposición orientada a objetos, así como una

notación para representar los modelos lógico y físico del sistema en diseño. También se incluyen los modelos estático y dinámico del sistema. Específicamente, la notación incluye diagramas de clases, diagramas de objetos, diagramas de módulos y diagramas de procesos.

Notaciones para el diseño

Todos los diagramas que en este proyecto se presentan son de acuerdo al tipo de especificación que sea aplicada. En este caso, los diagramas de flujo de datos y modelos entidad-relación, con su respectiva notación. Si existiere algún otro diagrama o notación que deba ser incluida en este trabajo, será incluida en el anexo correspondiente.

Principios generales de diseño

Para evaluar la calidad de la representación del diseño, es necesario establecer criterios técnicos para un buen diseño. A continuación se presenta una lista de criterios que pueden utilizarse (fuente de Internet, 18).

• Un diseño debe contener una organización jerárquica que haga un uso inteligente del control entre los elementos del software.
• Un diseño debe ser modular. En otras palabras, el sistema debe estar particionado lógicamente en elementos que realizan funciones y subfunciones específicas.
• Un diseño debe contener abstracciones de datos y abstracciones procedurales.
• Un diseño debe conducir a módulos (esto es, subrutinas o procedimientos), que muestren características funcionales independientes.
• Un diseño debe considerar interfaces que reduzcan la complejidad de conexiones entre módulos y con el ambiente externo.
• Un diseño debiese ser construido usando un método repetible, guiado por la información obtenida durante la fase de requisitos de software.

1.1.5. Programación.

El lenguaje de programación escogido afecta significativamente los costos, confiabilidad y rendimiento del sistema. Ningún lenguaje es ideal para todas las aplicaciones. La elección del lenguaje debe estar basada en la naturaleza de las aplicaciones (tiempo-real, incrustada, procesamiento batch, basado en Web, etc.) y en la importancia de algunos indicadores de calidad (rendimiento versus confiabilidad). La base de datos también debe ser confiable y proteger privacidad e información confidencial de usuarios no autorizados.

Los estilos de codificación incluyen los nombres de variables, la forma de hacer comentarios en el código fuente, el diseño y escritura de rutinas y módulos, la creación de tipos de datos, la selección y control de estructuras y la organización de bloques de instrucciones. A veces, algunos de estos factores son

determinados por la sintaxis y el paradigma de programación utilizados, existiendo estándares para lo anterior. Típicamente, un grupo de programadores, o una empresa, utiliza el mismo estándar con el propósito de que todos los programadores puedan acceder fácilmente a todo el código.

1.1.5.1. Estilo de programación.

Existen diferentes metodologías para realizar las actividades de programación. Sin embargo, todas ellas muestran un patrón común. Este patrón consiste en la exploración de herramientas y lenguajes, la determinación del estilo de programación, el desarrollo de herramientas utilitarias y rutinas comunes para administrar la entrada, salida y errores, la codificación y depuración del sistema, la escritura de la documentación técnica, y para todo el equipo de programadores, realizar revisiones personales periódicas y reuniones.

En la exploración de las herramientas y lenguajes, se debe considerar el tipo de aplicación y su naturaleza, con el fin de determinar el lenguaje apropiado. Para realizar esta selección, se debe considerar la metodología utilizada en las actividades de diseño así como el paradigma de programación del lenguaje.

El desarrollo de las herramientas utilitarias y rutinas comunes debe realizarse antes de la entrega del documento que contiene el diseño arquitectónico del sistema. Estos utilitarios ayudan al programador a realizar la codificación más rápida debido a que puede focalizarse sólo en la codificación de los factores especificados en el documento de diseño en vez de ocupar tiempo en la codificación de otras rutinas que son necesarias en todo programa. En estas funciones utilitarias, el programador debe construir algunas rutinas que considere necesarias, dependiendo del tipo de aplicación. Algunas herramientas de desarrollo han integrado herramientas que contienen rutinas comunes para cada programa, así como rutinas para ambientes específicos. Este factor debe ser considerado en la selección del lenguaje y herramientas de desarrollo (de acuerdo a fuente de Internet 18).

Un factor muy importante en la construcción de un sistema es el paradigma de programación utilizado. Uno de los paradigmas muy utilizados es el orientado a objetos, el cual tiene varias ventajas sobre otras metodologías de programación, por tanto, es el método de programación que será utilizado, dado las características que tiene (herencia, encapsulamiento, etc.).

1.1.5.2. Herramientas automatizadas.

Existen muchas herramientas para ayudar al programador a desarrollar el sistema. Éstas consisten principalmente en ambientes de desarrollo integrados adaptados para un lenguaje de programación. Estos ambientes pueden compilar y ejecutar un programa usando comandos para ello. La mayoría de estos ambientes

también proveen herramientas para depurar el programa, y algunos otros, herramientas de “temporización” (scheduling).

1.1.6. Pruebas del producto de programación.

Las pruebas implican utilizar una metodología que en forma sistemática, organizada y estructurada, que permita detectar y corregir, los errores y defectos introducidos en el proceso de desarrollo del sistema. Típicamente, se utilizan dos técnicas para ello: la prueba de la caja blanca y la prueba de la caja negra, descritas en el párrafo 1.1.6.1. Las pruebas y sus correspondientes reportes de validación del sistema serán mostrados con más detalle en el capítulo de implementación del sistema.

1.1.6.1. Pruebas.

Ahora se detallarán los métodos de prueba mencionadas anteriormente.

Según referencia (Internet, 18), las pruebas de caja blanca corresponden a un método de diseño de casos de pruebas que utilizan la estructura de control del diseño procedural para derivar los casos de pruebas. Usando este método, se puede derivar casos de pruebas para lo siguiente:

- Asegurarse que todas las trayectorias independientes en un módulo han sido visitadas al menos una vez.
- Ejercitar todas las decisiones lógicas en sus lados verdadero y falso.
- Ejecutar todos los ciclos (loops) en sus límites y sobre sus límites operacionales.
- Ejercitar estructuras de datos internas para asegurar su validez.

Por otro lado, las pruebas de caja negra se focalizan en los requisitos de usuario del sistema. De esa forma, este tipo de pruebas permite que los probadores generen conjuntos de datos de entrada que ejercitarán completamente los requisitos del sistema. Las pruebas de caja negra no son una alternativa a las pruebas de caja blanca. Más aún, corresponde a un enfoque complementario que posiblemente descubra una clase diferente de errores. Las pruebas de caja negra tratan de encontrar errores en las siguientes categorías:

- Funciones incorrectas o faltantes.
- Errores de interfaces.
- Errores en estructuras de datos o acceso a bases de datos externas.
- Errores de rendimiento del sistema y errores de inicialización y terminación.

Las pruebas de caja blanca son realizadas tempranamente en el ciclo de vida del desarrollo. Las pruebas de caja negra se utilizan en etapas más tardías del ciclo. Debido a que la metodología de caja negra ignora las estructuras de control del programa, la atención se focaliza en el dominio de la información.

1.2. Introducción a las bases de datos.

1.2.1. Introducción a las bases de datos.

En este apartado se verán los tipos de bases de datos que se han definido a través de la evolución de las mismas. De igual manera podremos ver la historia y otros conceptos básicos que son necesarios para el desarrollo de esta tesis.

1.2.1.1. Tipos de bases de datos según el modelo de datos.

Tradicionalmente, las bases de datos se han clasificado en tres grupos: jerárquicas, en red y relacionales. Sin embargo, la aparición de nuevos entornos y aplicaciones ha introducido un modelo de bases de datos diferente: las bases de datos orientadas a objetos.

Las bases de datos jerárquicas fueron las primeras en aparecer. La información se representa en forma de árbol. El problema con este tipo de estructura es que no todas las bases de datos se adaptaban a la estructura en árbol. En un intento de eliminar la rigidez de las bases de datos jerárquicas, se desarrolló la estructura en red, en la cual se permitían todo tipo de relaciones. Cualquier conjunto de información es representable mediante una base de datos en red. Por último, aparecieron las bases de datos relacionales que pretendían obtener mayor flexibilidad y rigor en el tratamiento de datos. Nacieron en la década de los años 70 de la mano de E. F. Codd, quien planteó una alternativa a las bases de datos jerárquicas y en red existentes hasta el momento.

La reciente aparición de las bases de datos orientadas a objetos surge como consecuencia de las deficiencias observadas en las bases de datos relacionales en cuanto al modelado de datos complejos como pueden ser objetos CAD o documentos. Se pueden ver como una extensión de las bases de datos en red con facilidades para el modelado de datos. Las bases de datos orientadas a objetos soportan encapsulación de objetos, fuerte tipificación de objetos, herencia y reusabilidad.

De todos los modelos, la aproximación relacional es la más utilizada en el diseño y gestión de bases de datos existiendo una amplia gama de productos disponibles en el mercado. Los modelos jerárquico y en red, implantados en muchas bases de datos antiguas, han quedado obsoletos y prácticamente no se utilizan en los nuevos sistemas. Sin embargo, las bases de datos orientadas a objetos están ganando aceptación en los últimos años debido, fundamentalmente, a la aparición de nuevas aplicaciones y necesidades de los usuarios que no pueden ser cubiertas por los modelos existentes. El mercado de bases de datos orientadas a objetos ha experimentado un gran crecimiento y, actualmente, existen bastantes productos disponibles, especialmente para entornos de estaciones de trabajo.

1.2.1.2. Sistemas de bases de datos.

El por qué utilizar sistemas que contengan una gran cantidad de información y sus definiciones básicas.

1.2.1.2.1. Evolución.

Los sistemas orientados a los datos se caracterizan porque los datos no son de una aplicación sino de una organización entera que los va a utilizar. Se tiende a integrar las aplicaciones, evitando aplicaciones aisladas. Se diferencian las estructuras lógicas y físicas, de manera que el usuario final solo se vincule con las estructuras lógicas. La descripción de la estructura lógica se separa de los lenguajes de programación. El concepto de relación cobra importancia, de modo que se requiere de herramientas que permitan definirlas y almacenarlas.

Originalmente las aplicaciones que se desarrollaban para las organizaciones estaban orientadas a cubrir necesidades muy específicas de procesamiento, por lo tanto, los lenguajes de programación como las estructuras de datos se centraban de manera más eficiente una tarea específica.

Así, las bases de datos buscan resolver principalmente el problema de repetición innecesaria de datos y, así, evitar las inconsistencias que se producían por la utilización de los mismos datos lógicos desde distintas plataformas físicas (archivos) a través de procesos independientes.

El análisis entonces, comienza por formular la lógica de los datos organizacionales como un todo, para después vincular aquellos con los procesos que los utilizan. Es en este análisis en que las bases de datos como una unidad tanto teórica como conceptual y física cobran importancia. Y el mecanismo sobre el cual esto se articula es el de disminuir la redundancia a través del establecimiento de relaciones entre los datos de una organización.

1.2.1.2.2. Concepto de base de datos.

La idea de base de datos surge como una necesidad de mantener datos relacionados. Veamos un ejemplo.

Supongamos 3 archivos: artículos, clientes y pedidos. Un cliente puede tener varios pedidos, pueden existir varios pedidos para un mismo artículo, diferentes pedidos en diferentes fechas, etc.

Si los requerimientos son:

- Los pedidos del cliente XXX, una solución es generar una lista por cada cliente, generando una relación entre los archivos clientes y pedidos.

- Todos los pedidos de un artículo, se tendrá una lista por cada artículo, generando una relación entre pedidos y artículos.

Las relaciones anteriores son complejas, por lo que se hace necesario contar con alguna herramienta que facilite estos requerimientos. Se debe considerar que los requerimientos hacen uso de los mismos datos.

Las definiciones de base de datos son numerosas. Todas coinciden en que es un conjunto de datos almacenados en un soporte de acceso directo. Los datos están interrelacionados y estructurados de acuerdo a un modelo que sea capaz de recoger el máximo contenido semántico.

Definición 1. “Colección de datos interrelacionados almacenados en un conjunto sin redundancias perjudiciales o innecesarias; su finalidad es servir a una o más aplicaciones de la mejor forma posible; los datos se almacenan de modo que resulten independientes de los programas que los usan; se emplean métodos bien determinados para incluir nuevos datos y para modificar o extraer los datos almacenados”. Martín, 1975.

Definición 2. “Colección integrada y generalizada de datos, estructurada atendiendo a las relaciones naturales de modo que suministre todos los caminos de acceso necesarios a cada unidad de datos con objeto de poder atender todas las necesidades de los diferentes usuarios”. Deen, 1985.

Definición 3. “Colección de datos integrados, con redundancia controlada y con una estructura que refleje las interrelaciones y restricciones existentes en el mundo real; los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de éstas, y su definición y descripción, únicas para cada tipo de datos, han de estar almacenadas junto con los mismos. Los procedimientos de actualización y recuperación, comunes y bien determinados, habrán de ser capaces de conservar integridad, seguridad y confidencialidad del conjunto de los datos”. A. de Miguel, 1993.

Definición 4. “La base de datos consiste en alguna colección de datos persistentes e independientes usados por una organización determinada”. Date, 1995.

1.2.1.2.3. Característica del dato:

- No efímero, en el sentido que permanece en el tiempo.
- Estructurado, para que facilite el compartirlos por aquellos que lo necesiten.
- Operacional transaccional (OLTP), manipularlos aplicando operadores para obtener resultados.
- Sentido semántico.
- Integro, en el sentido que refleja una realidad existente.

De hecho los datos que contiene una base de datos tienen una característica especial, se les reconoce como datos de operación.

Datos de operación: los datos de una base de datos se consideran datos de operación, distinguiéndose de los datos de entrada y de salida. Una base de datos, es un conjunto de datos de operación almacenados y utilizados por los sistemas de aplicación de una organización específica. Cualquier organización necesita disponer de una gran cantidad de datos acerca de su funcionamiento. Estos constituyen sus datos de operación.

1.2.2. Diseño de bases de datos.

Se comienza este apartado con la teoría necesaria que incluye definiciones básicas que nos auxilien en el diseño de la base de datos del sistema en cuestión. Veremos también las formas normales como herramientas para hacer más eficiente el diseño que contendrá la información del paquete (fuente de Internet 22).

1.2.2.1. Conceptos para el diseño de bases de datos.

Dependencias funcionales. Sea R(A1...An) y sean x, y subconjuntos de {A1...An}, decimos que x determina funcionalmente a y (o que y depende funcionalmente de x), se denota $x \Rightarrow y$, si para cualquier relación r de R no es posible que r tenga 2 tuplas que coincidan en x pero difieran en y.

Llave. Un candidato K a llave es un conjunto de atributos (posiblemente un solo atributo) de una relación R si cumple con las siguientes propiedades:

Identificación única. Cada tupla de R puede ser identificada de manera única por el valor de K.

No redundancia. Ningún atributo que pertenezca a K puede ser eliminado sin que se destruya la propiedad (i).

Dependencia funcional. Cada atributo de R es funcionalmente dependiente de la llave.

Se dice que Z es superllave si contiene a alguna llave candidata de R, es decir, $X \subseteq Z$ y X es llave candidata.

Dada una relación pueden existir diferentes candidatos a llave y una de las llaves candidatas es arbitrariamente designada como la llave primaria.

Atributo primo. Se dice que un atributo es ‘primo’ en R, si es un atributo de R y forma parte de al menos una llave candidata de R y todos los atributos de R son llamados ‘no primos’.

1.2.2.2. Formas normales.

Un número de diferentes propiedades o formas normales para los esquemas relacionales con dependencias han sido definidas. El más significativo de ellos es la llamada ‘tercera forma normal’ y la forma ‘normal de Boyce-Codd’. Estas formas normales garantizan que muchos de los problemas de redundancia y anomalías no ocurran (de acuerdo a la fuente de Internet 22).

Primera forma normal. Se da ésta cuando todos los atributos ‘no primos’ de una relación R son definidos funcionalmente por la llave primaria. Cuando esto no ocurre se crean lo que se llaman ‘dependencias incompletas’. Al aplicar la primera forma normal se eliminan estas dependencias. Se denota también por 1FN.

Segunda forma normal. Se dice que una relación R está en segunda forma normal (2FN) cuando está en primera forma normal y, además, todos los atributos ‘no primos’ de R están en dependencia funcional completa con cada candidato la llave de R. Dicho de otra manera, una relación R viola la 2FN si algún atributo no primo de R es funcionalmente dependiente de un subconjunto de una llave. Esto se conoce como ‘dependencias parciales’.

Tercera forma normal. Ahora, si la relación R está en segunda forma normal y todos los atributos no primos de R no son dependientemente funcionales de algún otro atributo no primo, se dice que la relación R está en tercera forma normal o 3FN. A la definición funcional de atributos no primos por atributos no primos, también, se le llama ‘dependencias transitivas’. La 3FN elimina estas dependencias.

Cuarta forma normal. También conocida como forma normal de Boyce-Codd o 4FN. Se cumple cuando toda dependencia funcional de una relación R, está definida por una superllave de R.

1.2.2.3. Algebra relacional.

Las operaciones de álgebra relacional manipulan relaciones. Esto significa que estas operaciones usan una o dos relaciones existentes para crear una nueva relación. Esta nueva relación puede entonces usarse como entrada para una nueva operación. Este poderoso concepto - la creación de una nueva relación a partir de relaciones existentes 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 nueve operaciones (según documentos 33 y 34 de fuentes de Internet):

- 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. Pueden ser vistas en detalle en el anexo A.

1.3. Motores de búsqueda.

1.3.1. Definiciones básicas.

A continuación describiremos las definiciones básicas y operación de un motor de búsqueda para la Internet. En este caso, interesa un motor de búsqueda no tan sofisticado y complejo como los descritos abajo, sin embargo es necesario que haga búsquedas sobre cualquier atributo de texto.

1.3.1.1 ¿Qué es un motor de búsqueda?

La fuente de Internet 23, da la siguiente definición: “Durante cuatro mil años, el género humano ha organizado la información de manera que fuera fluida y accesible fácil e inmediatamente. Internet y en particular el web en estos últimos años resultan ser el medio principal para publicar y poner la información a disposición de todos. Los motores de búsqueda representan el medio para recuperar la información, pero a la luz de sus dimensiones resulta cada vez más difícil encontrar la información que uno va buscando y por lo tanto más difícil para ellos catalogarla. Por esta razón, durante los últimos años los motores de búsqueda han aprendido a proporcionar una valuación sobre la importancia de las páginas para evitar de ofrecer al navegante resultados inútiles. Puede parecer una paradoja, pero tener muchísima información equivale a no poseer ninguna. Consecuentemente, últimamente se ha dado importancia a la calidad de la presencia en los motores de búsqueda y más en general en el web, y por esta razón los motores han modificado sus algoritmos o se han convertido en portales donde uno paga para tener un resalto mayor. Los motores de búsqueda son, sin duda alguna, el punto de partida de un navegante que esté buscando algo. Todos sabemos que los motores contestan a una búsqueda que el navegante hace en el formulario de búsqueda y restituyen un listado de sitios que es la elaboración de las palabras solicitadas en el formulario. Estas palabras están sometidas a un algoritmo, que cambia al cambiar el motor. El listado puede ser ordenado según la importancia que el motor le da a los sitios web. Esta fase se define del rango y es la llave del éxito de un motor de búsqueda. El motor hoy más popular y utilizado es Google, que gracias a su sofisticado mecanismo de análisis de las páginas (ranking) se ha convertido en uno de los más importantes sujetos del web”.

En el Internet, un motor de búsqueda es un conjunto coordinado de programas que incluye:

- Un “spider” (también llamado un "crawler" o un "bot") que va a cada paginación o a las paginaciones del representante en cada sitio web que desee ser investigable y la lea, usando conexiones de hypertexto en cada paginación para descubrir y para leer otras paginaciones de un sitio.
- Un programa que crea un índice enorme (a veces llamado un "catálogo") de las paginaciones que se han leído.

- Un programa que recibe su petición de la búsqueda, la compara a las entradas en el índice, y le devuelve resultados

Todos los motores de la búsqueda hacen búsquedas de palabra clave contra una base de datos, pero los varios factores influencian los resultados de cada uno. La talla de la base de datos, frecuencia de la actualización, capacidad y diseño de la búsqueda, y la velocidad puede conducir a resultados asombroso diversos. La adición reciente del nuevo contenido, el reajuste y los cambios de la sociedad han satisfecho la misión del nuevo nombre: portales. Este nombre implica un punto de partida y una localización central para todas las aplicaciones del Web.

1.3.1.2 ¿Por qué un motor de búsqueda?

En el caso específico de este proyecto, lo que se pretende no es un buscador como se ha descrito anteriormente, sino una utilidad que encuentre información específica sobre una base de datos relacional, lo que puede ser considerado como tal, dada la extensión de información que podría llegar a abarcar esta base de datos.

1.3.1.3. Algoritmos.

El método de búsqueda depende de la información y de cómo esté almacenada. En este caso, la información estará en una base de datos, y por tanto las búsquedas serán hechas en ella. Esto lleva al término no precisamente de búsqueda sino de consulta, y dado que, el lenguaje de consulta prácticamente universal sobre una base de datos relacional es el SQL, entonces se planea utilizarlo como el buscador o motor de búsqueda de este sistema, efectuando consultas como se verá más adelante.

1.4. Herramientas de análisis, diseño y programación.

En esta sección serán descritas las metodologías a ser usadas para la realización del presente proyecto.

1.4.1. Análisis de flujo de datos.

La estrategia del flujo de datos muestra el empleo de éstos en forma gráfica. Las herramientas usadas para seguir esta estrategia muestran todas las características esenciales del sistema y la forma en que se ajustan entre sí (fuente de Internet 27).

Puede ser difícil comprender en su totalidad un proceso de la empresa si se emplea para ello solo una descripción verbal; las herramientas para el flujo de datos ayudan a ilustrar los componentes esenciales de un sistema junto con sus interacciones.

El análisis de flujo de datos usa las siguientes herramientas:

- Diagrama de flujo de datos.
- Diccionario de datos.
- Diagrama de estructura de datos (diagrama de E-R, ver sección 1.4.3)
- Gráfica de estructura.

1.4.1.1. Diagrama de flujo de datos.

Son una de las cuatro herramientas del análisis estructurado. Es una herramienta gráfica que se emplea para describir y analizar el movimiento de los datos a través de un sistema, ya sea este manual o automatizado, incluyendo procesos, lugares para almacenar datos y retrasos en el sistema. Los DFD, como se les conoce popularmente son la herramienta más importante y la base sobre la cual se desarrollan otros componentes. La transformación de datos de entrada en salida por medio de procesos puede describirse en forma lógica e independiente de los componentes físicos (computadoras, gabinetes de archivos, y procesadores de texto) asociados con el sistema (fuente de Internet 27).

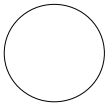
Notación: los DFD se pueden dibujar con solo cuatro notaciones sencillas, a saber:

Flujo de datos: movimiento de datos en determinada dirección, desde un origen hasta un destino en forma de documentos, cartas, llamadas telefónicas o virtualmente cualquier otro medio. El flujo de datos es un “paquete de datos”.

Representación (fuentes de Internet 27 y 28):

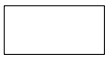
Procesos: personas procedimientos o dispositivos que usan o producen (transforman) datos.

Representación:



Fuente o destino de datos: fuentes o destinos externos de datos, que pueden ser personas, programas, organizaciones u otras entidades que interactúan con el sistema pero que se encuentran fuera de sus fronteras. La diferencia fundamental con los procesos es que las fuentes o destinos no transforman información, al menos no dentro de las fronteras del sistema que se está modelando.

Representación:



Almacenamiento de datos: es el lugar donde se guardan los datos o al que referencían los procesos en el sistema. El almacenamiento de datos puede representar dispositivos tanto computarizados como no computarizados.

Representación:

Los DFD se concentran en el movimiento de los datos a través del sistema, no en los dispositivos o el equipo. Los analistas identifican y describen, desde el inicio hasta del final proceso, para comprender un área de aplicación o los datos que fluyen por todo el sistema y entonces explican por qué los datos entran o salen y cuál es el procesamiento que se realiza con ellos. Es muy importante determinar cuándo entran los datos al área de aplicación y cuándo salen de ésta.

A medida que los analistas reúnen hechos y detalles, comprenden mejor el proceso; esto los conduce a formular preguntas relacionadas con aspectos específicos del mismo y los lleva a una investigación adicional. La investigación se divide en detalles que tienen cada vez un nivel menor hasta que se comprenden todos los componentes esenciales junto con sus interrelaciones.

Lo que se quiere dar a entender con esto, es que una investigación de sistemas produce muchos conjuntos de DFD, algunos (los primeros) brindan panoramas de procesos importantes, mientras que otros (los que se obtienen de los primeros) nos muestran con bastante detalle elementos dato, almacenes de

datos y pasos de procesamiento para componentes específicos de un sistema grande.

A los primeros diagramas obtenidos se les conoce como diagramas de alto nivel, mientras que a los resultantes de estos se les conoce como diagramas de bajo nivel.

En este sentido el primer diagrama que se obtiene se le conoce con el nombre de diagrama de contexto, es un diagrama de nivel muy general (alto nivel); es también conocido como diagrama de nivel 0. Contiene un solo proceso pero juega un papel muy importante en el estudio del sistema en uso; ya que define fronteras. Todo lo que no se encuentre dentro de las fronteras identificadas en el diagrama no forman parte del estudio de sistemas. La forma en que funcionen otras organizaciones o elementos externos (las fuentes y destinos) está fuera de nuestro control y no será estudiado con detalle.

Cada flujo de datos (cada flecha) emplea una etiqueta que describe que datos emplea. Cuando los datos se mueven de un lugar a otro el flujo de datos apunta hacia el lugar donde se dirige el flujo.

Ejemplo (fuente de Internet 27):

- Un sistema está formado por varias actividades o procesos, cada uno de los cuales contiene varios subprocesos con marcadas interrelaciones entre ellos. Por ejemplo un proceso de cuentas por pagar puede estar integrado por tres subprocesos que podrían llamarse: autorización de la factura, revisión del adeudo en la cuenta y elaboración del cheque.

- A su vez cada subproceso se divide en subprocesos más específicos.

- Los nombres dados a los procesos especifican acciones y procedimientos de control que realizan.

- Cada proceso se etiqueta además con un número que identifica de donde proviene (excepto el diagrama de contexto que solo se identifica con un nivel 0 más el nombre que se le proporcione).

- En términos generales todo componente de los DFD se etiqueta con un nombre que sea representativo.

Primer nivel del DFD

En el primer nivel, es muy importante identificar los principales procesos, y flujos que dan en forma conjunta sentido operacional al sistema que se está modelando.

Algunos analistas consideran ventajoso trabajar primero con todos los flujos de datos y asignar, como ya se dijo nombres que sean significativos y descriptivos. Se identifican todos los procesos, como ya se mencionó pero no se les da nombre hasta que sean bien entendidos todos los flujos de datos. Después cuando se les ha asignado nombre a los procesos, si el analista tiene dificultas para ligar los flujos de datos con los nombres apropiados entonces esta situación indica que es necesario dividir aun más el proceso.

Expansión de los procesos a diagramas de mayor nivel

Una vez que se ha desarrollado el sistema como está descrito en el diagrama de primer nivel, es indudable que el analista formule preguntas en relación con la forma que se lleven a cabo los procesos. En general se debe estar seguro de:

- Todos los flujos de datos que explican el proceso en el diagrama previo deben incluirse en el diagrama del siguiente nivel inferior.

- Los flujos y almacenes de datos nuevo se añaden si son usados internamente por el proceso para eslabonar otros procesos introducidos por primera vez en la expansión de este nivel. Se deben mostrar los flujos y almacenes de datos originados en el proceso dentro en este nivel.

- Ninguna entrada debe contradecir las descripciones de los DFD de niveles más altos (si lo hacen uno o ambos son incorrectos y deben introducirse cambios).

En general la expansión de niveles depende de la naturaleza y complejidad del sistema que se modele; no es posible especificar un número de niveles, en general se debe continuar con el proceso de expansión todo lo que sea necesario para comprender los detalles del sistema y la forma en que trabaja, teniendo cuidado de verificar todos los aspectos con usuarios que conocen el sistema, en general, se debe expandir todo aquel proceso que incluyen varias tareas para las que es necesario, el flujo de datos entre diferentes personas o localidades. Por otra parte no requieren expansión aquellas tareas que son realizadas por una persona o en un escritorio, donde no existe flujo de datos.

Reglas adicionales para el dibujo de DFD.

Ya se han identificado la mayor parte de los lineamientos que se siguen para el dibujo de los DFD, he aquí algunas más:

- Cualquier flujo de datos que abandone un proceso debe estar basado en los datos que entran al proceso.

- Todos los flujos de datos tienen un nombre que refleja los datos que fluyen entre procesos, almacenes de datos, fuentes o destinos.

- Solo deben entrar al proceso, los datos necesarios para llevarlo a cabo.

- Un proceso no debe saber nada de ningún otro en el sistema, es decir debe ser independiente, la única dependencia que debe existir es aquella basada en sus propios datos de entrada y salida.

- Los procesos siempre están en continua ejecución, no se inician ni tampoco se detienen. Los analistas siempre deben suponer que un proceso está listo para ejecutar su trabajo.

- La salida de los procesos puede tomar una de las siguientes formas.

- Flujo de datos con información añadida por el proceso.

- Una respuesta o cambio en la forma de los datos.

- Un cambio de condición.

- Cambio de contenido.

- Cambios en la organización.

- La norma común es definir cada nivel inferior en términos de 3 a 7 procesos para cada proceso de nivel superior, si son necesarios más detalles se puede hacer en el siguiente nivel.

- Los almacenes y flujos de datos que son relevantes solo para el interior del proceso, son ocultados hasta que el proceso se extiende con mayor detalle.

- Los datos que fluyen hacia los procesos experimentan cambios. Por consiguiente, el flujo de datos de salida tiene un nombre diferente al de la entrada; si no se efectúa algún cambio en el flujo de datos, entonces ¿cuál es la finalidad del proceso?

- En cuanto a los nombres de los procesos lo más apropiado es escoger un verbo y un sujeto que reciba la acción y no nombres generales que no digan nada. Si un nombre de proceso es vago o complejo tal vez se deba subdividir el proceso aún más.

Por otra parte no se ha mencionado nada aún sobre controles en los DFD, ni nada al respecto sobre como manejar errores o excepciones, por ejemplo, el procesamiento de facturas incorrectas. Aunque esta información es necesaria para el análisis final, no es importante identificar todos los flujos de datos (los

errores o excepciones son también flujos de datos). Los diagramas secundarios (por debajo del segundo o tercer nivel), deben mostrar el manejo de errores y excepciones del proceso.

Aun así ciertos detalles físicos como el día de la semana que se debe hacer un pago u otros controles de este tipo son innecesarios en los DFD, puesto que no tienen nada que ver con los aspectos lógicos y de datos de la determinación de requerimientos. Los elementos importantes para comprender un proceso durante el análisis lógico de flujo de datos, no son el número de copias que se requieren de un documento sino las descripciones de los datos necesarios para llevar a cabo el proceso.

1.4.1.2. Diccionario de datos.

La fuente de Internet 27, nos dice que “un diccionario de datos es un catálogo, un depósito, de los elementos de un sistema. Estos elementos se centran alrededor de los datos y la forma en que están estructurados para satisfacer los requerimientos y las necesidades de la organización. En él se encuentran la lista de todos los elementos que forman parte del flujo de datos en todo el sistema”.

Importancia del diccionario:

Los analistas usan los diccionarios de datos por cinco razones principales:

- Manejar los detalles en sistemas grandes.
- Comunicar un significado común para todos los elementos del sistema.
- Documentar las características del sistema.
- Facilitar el análisis de los detalles con la finalidad de evaluar las características y determinar donde efectuar cambios en el sistema.
- Localizar errores y omisiones en el sistema.

Contenido de un registro del diccionario:

Campos: es el nivel más importante de datos; ninguna unidad más pequeña tiene significado para los analistas. La descripción de los datos debe ir acompañada por los siguientes elementos:

Estructuras de datos: son un grupo de datos elementales que están relacionados con otros y que en conjunto describen un componente del sistema. Los flujos de datos, o los almacenes de datos son ejemplo de estructuras de datos. Dicho de otra forma si las estructuras están en movimiento reciben el nombre de flujos y si son estéticas son almacenes de datos. Se construyen sobre cuatro relaciones de componentes; que bien pueden ser datos o estructuras de datos también. Se pueden usar las siguientes combinaciones ya sea en forma individual o en conjunción con alguna otra:

- Relación secuencial
- Relación de selección
- Relación de iteración
- Relación opcional

Notación empleada en el Diccionario de datos:

Se usa símbolos especiales con la finalidad de limitar la cantidad de texto necesario empleado para describir las relaciones entre los datos y al mismo tiempo mostrar con claridad las relaciones estructurales.

La simbología empleada se describe a continuación (fuentes 27 y 28 de Internet):

Símbolo	Significado	Explicación	Uso
=	Es equivalente a	Alias	Denota sinónimos
+	Y	Concatenación, componentes que siempre están incluidos en una estructura	Denota una relación de secuencia
[|]	Uno u otro	Define opciones entre los componentes de una estructura	Denota una relación de selección
{ }	Iteraciones de	Define la repetición de un componente de la estructura	Denota una relación de iteración
( )	Opcional	Define componentes de la estructura que puede o no estar presente una sola vez	Denota una relación opcional.
*** ***		Comentarios	Delimitación de comentarios.
@	Clave	Identificador	Hace la distinción entre los registros de la tabla.

Registro de las descripciones de datos en el diccionario:

Flujos de datos:

- Nombre del flujo de datos
- Descripción
- Proviene de los procesos
- Para los procesos
- Estructuras de datos

Almacenes de datos:

- Nombre del almacén
- Descripción
- Flujos de datos recibidos
- Flujos de datos proporcionados
- Descripción de los datos (mención a los datos o estructuras que contiene)
- Volumen
- Acceso

Estructuras de datos (es aquí donde se emplea la notación descrita en la tabla anterior):

- Nombre de la estructura
- Descripción
- Contenido
- Volumen

Elementos datos:

- Nombre del dato
- Descripción
- Tipo
- Longitud
- Alias
- Rango de valores
- Lista de valores específicos (en caso que existan)
- Otros detalles de edición

Procesos:

- Nombre del proceso
- Descripción
- Flujos que entran
- Flujos que salen
- Resumen de la lógica

1.4.2. Consideraciones de POO.

Ingeniería de software es la producción de software con calidad. Calidad implica dos tipos de factores: internos y externos. Los factores externos son cualidades que son "detectadas" por los usuarios, por ejemplo: velocidad y facilidad de uso. Los factores internos son cualidades perceptibles por profesionales del área de la computación, con acceso al código fuente, por ejemplo: modularidad y legibilidad.

1.4.2.1. Factores externos.

Según fuente de Internet 15:

Correctitud: Capacidad para realizar con exactitud las tareas definidas en las especificaciones.

Robustez: Capacidad de reaccionar apropiadamente ante condiciones excepcionales.
Extensibilidad: Facilidad de adaptar los productos de software a los cambios en la especificación.
Reutilización: Capacidad de los elementos de software de servir para la construcción de muchas aplicaciones diferentes.
Compatibilidad: Facilidad de combinar unos elementos de software con otros.
Eficiencia: Capacidad para exigir la menor cantidad posible de recursos (tiempo de procesador, espacio de memoria, ancho de banda, etc.).
Portabilidad: Facilidad de transferir los productos de software a diferentes entornos de hardware y software.
Facilidad de uso: Cubre la facilidad de instalación, de operación y de supervisión.
Funcionalidad: Conjunto de posibilidades que proporciona un sistema.

El mantenimiento no se menciona como un factor, pero se estima que el 70% del costo del software se dedica al mismo. Lientz identifica ocho categorías en que se puede desglosar el mantenimiento:

- Cambios en los requisitos del usuario (41.8%)
- Cambios en los formatos de los datos (17.6%)
- Cambios de emergencia (12.4%)
- Arreglo de rutinas (9%)
- Cambios en el hardware (6.2%)
- Documentación (5.5%)
- Mejoras en la eficiencia (4%)
- Otros (3.5%)

A partir de los objetivos de extensibilidad y reutilización, dos de los factores de calidad más importantes, se desprende la necesidad de tener arquitecturas de sistemas flexibles, hechas con componentes autónomos de software. Esto se logra con una adecuada modularidad.

Un método de diseño que merezca ser llamado modular, debería satisfacer cinco requisitos fundamentales (fuente de Internet 15):

1. Descomposición modular. Ayuda en la tarea de descomponer el problema en un número de sub-problemas menos complejos, interconectados.
2. Composición modular. Favorece la producción de elementos de software que se puedan combinar libremente unos con otros para producir nuevos sistemas.
3. Comprensibilidad modular. Ayuda a producir software en el cual un lector humano puede entender cada módulo sin tener que conocer los otros (o examinando sólo unos pocos).
4. Continuidad modular. Se satisface si un pequeño cambio en la especificación de un problema provoca sólo cambios en un solo módulo o en un número pequeño de módulos.

5. Protección modular. Si produce arquitecturas en las cuales el efecto de una situación anormal ocurrida en un módulo durante la ejecución, queda confinado a ese módulo (o se propaga sólo a unos pocos vecinos).

1.4.2.2. Reglas derivadas.

De los criterios anteriores para asegurar la modularidad, se derivan cinco reglas (fuente de Internet 15):

1. Correspondencia directa. Conexión de un sistema de software con los sistemas externos con que está relacionado. La estructura modular obtenida, debe seguir siendo compatible con cualquier otra estructura modular en el dominio del problema.
2. Pocas interfaces. Cada módulo debe comunicarse con el menor número de módulos posible.
3. Interfaces pequeñas (acoplamiento débil). Si dos módulos se comunican, deben intercambiar la menor información posible.
4. Interfaces explícitas. Siempre que dos módulos A y B se comuniquen, esto debe ser obvio a partir del texto de A, del de B o de ambos.
5. Ocultar información, sea, mostrar sólo lo necesario.

1.4.3. Modelos Funcional y Entidad-Relación.

Funcional.

En el modelo funcional, el objetivo de los analistas es el de identificar y analizar las áreas funcionales, funciones y actividades que se realizan para llevar a cabo los objetivos de la empresa. Este tipo de información constituye el punto de partida para la agrupación adecuada de la información que se empleará para construcción de los modelos de datos (fuente 43).

Las tareas principales para llevar a cabo el análisis del modelo funcional son:

Identificar y analizar las áreas funcionales.
Identificar y analizar las funciones de cada área funcional.
Identificar y analizar las actividades de cada función.
Revisar y analizar el modelo funcional con los ejecutivos de alto nivel.

El modelo funcional es representado por un documento analítico que muestra el conjunto de áreas funcionales de la organización con sus funciones y actividades el cual se puede representar de formas. La primera forma consta de una tabla que contiene el nombre del área funcional con sus funciones y su respectiva actividad por función. La forma de representar un Modelo Funcional se puede llevar a cabo mediante un organigrama jerárquico, en el que el punto más alto describe el propósito fundamental del negocio, y debajo se colocan las

actividades relevantes (funciones) y cada una deberá empezar por un verbo, descendiendo a funciones más simples.

Entidad-Relación.

El modelo entidad-relación es el modelo conceptual más utilizado para el diseño conceptual de bases de datos. Fue introducido por Peter Chen en 1976. El modelo entidad-relación está formado por un conjunto de conceptos que permiten describir la realidad mediante un conjunto de representaciones gráficas y lingüísticas (fuente de Internet 24).

Originalmente, el modelo entidad-relación sólo incluía los conceptos de entidad, relación y atributo. Más tarde, se añadieron otros conceptos, como los atributos compuestos y las jerarquías de generalización, en lo que se ha denominado modelo entidad-relación extendido, que no se utilizará ni se verá en este proyecto.

1.4.3.1. Elementos básicos.

Entidad

Cualquier tipo de objeto o concepto sobre el que se recoge información: cosa, persona, concepto abstracto o suceso. Por ejemplo: coches, casas, empleados, clientes, empresas, oficios, diseños de productos, conciertos, excursiones, etc. Las entidades se representan gráficamente mediante rectángulos y su nombre aparece en el interior. Un nombre de entidad sólo puede aparecer una vez en el esquema conceptual.

Hay dos tipos de entidades: fuertes y débiles. Una entidad débil es una entidad cuya existencia depende de la existencia de otra entidad. Una entidad fuerte es una entidad que no es débil (fuente de Internet 24).

Relación (interrelación)

Es una correspondencia o asociación entre dos o más entidades. Cada relación tiene un nombre que describe su función. Las relaciones se representan gráficamente mediante rombos y su nombre aparece en el interior (fuente de Internet 24).

Las entidades que están involucradas en una determinada relación se denominan entidades participantes. El número de participantes en una relación es lo que se denomina grado de la relación. Por lo tanto, una relación en la que participan dos entidades es una relación binaria; si son tres las entidades participantes, la relación es ternaria; etc.

Una relación recursiva es una relación donde la misma entidad participa más de una vez en la relación con distintos papeles. El nombre de estos papeles es importante para determinar la función de cada participación.

A veces, surgen problemas cuando se está diseñado un esquema conceptual. Estos problemas, denominados trampas, suelen producirse a causa de una mala interpretación en el significado de alguna relación, por lo que es importante comprobar que el esquema conceptual carece de dichas trampas. En general, para encontrar las trampas, hay que asegurarse de que se entiende completamente el significado de cada relación. Si no se entienden las relaciones, se puede crear un esquema que no represente fielmente la realidad.

Una de las trampas que pueden encontrarse ocurre cuando el esquema representa una relación entre entidades, pero el camino entre algunas de sus ocurrencias es ambiguo. El modo de resolverla es reestructurando el esquema para representar la asociación entre las entidades correctamente.

Otra de las trampas sucede cuando un esquema sugiere la existencia de una relación entre entidades, pero el camino entre una y otra no existe para algunas de sus ocurrencias. En este caso, se produce una pérdida de información que se puede subsanar introduciendo la relación que sugería el esquema y que no estaba representada.

Atributo

Es una característica de interés o un hecho sobre una entidad o sobre una relación. Los atributos representan las propiedades básicas de las entidades y de las relaciones. Toda la información extensiva es portada por los atributos. Gráficamente, se representan mediante bolitas que cuelgan de las entidades o relaciones a las que pertenecen (fuente de Internet 24).

Cada atributo tiene un conjunto de valores asociados denominado dominio. El dominio define todos los valores posibles que puede tomar un atributo. Puede haber varios atributos definidos sobre un mismo dominio.

Los atributos pueden ser simples o compuestos. Un atributo simple es un atributo que tiene un solo componente, que no se puede dividir en partes más pequeñas que tengan un significado propio. Un atributo compuesto es un atributo con varios componentes, cada uno con un significado por sí mismo. Un grupo de atributos se representa mediante un atributo compuesto cuando tienen afinidad en cuanto a su significado, o en cuanto a su uso. Un atributo compuesto se representa gráficamente mediante un óvalo.

Por último, los atributos pueden ser derivados. Un atributo derivado es aquel que representa un valor que se puede obtener a partir del valor de uno o varios atributos, que no necesariamente deben pertenecer a la misma entidad o relación.

Identificador

Un identificador de una entidad es un atributo o conjunto de atributos que determina de modo único cada ocurrencia de esa entidad. Un identificador de una entidad debe cumplir dos condiciones (fuente de Internet 24):

- No pueden existir dos ocurrencias de la entidad con el mismo valor del identificador.
- Si se omite cualquier atributo del identificador, la condición anterior deja de cumplirse.

Toda entidad tiene al menos un identificador y puede tener varios identificadores alternativos. Las relaciones no tienen identificadores.

Jerarquía de generalización

Una entidad E es una generalización de un grupo de entidades E^1, E^2, ... E^n, si cada ocurrencia de cada una de esas entidades es también una ocurrencia de E. Todas las propiedades de la entidad genérica E son heredadas por las subentidades.

Cada jerarquía es total o parcial, y exclusiva o superpuesta. Una jerarquía es total si cada ocurrencia de la entidad genérica corresponde al menos con una ocurrencia de alguna subentidad. Es parcial si existe alguna ocurrencia de la entidad genérica que no corresponde con ninguna ocurrencia de ninguna subentidad. Una jerarquía es exclusiva si cada ocurrencia de la entidad genérica corresponde, como mucho, con una ocurrencia de una sola de las subentidades. Es superpuesta si existe alguna ocurrencia de la entidad genérica que corresponde a ocurrencias de dos o más subentidades diferentes.

Un subconjunto es un caso particular de generalización con una sola entidad como subentidad. Un subconjunto siempre es una jerarquía parcial y exclusiva.

1.4.3.2. Cardinalidad.

La cardinalidad con la que una entidad participa en una relación especifica el número mínimo y el número máximo de correspondencias en las que puede tomar parte cada ocurrencia de dicha entidad. La participación de una entidad en una relación es obligatoria (total) si la existencia de cada una de sus ocurrencias requiere la existencia de, al menos, una ocurrencia de la otra entidad participante. Si no, la participación es opcional (parcial). Las reglas que definen la cardinalidad de las relaciones son las reglas de negocio (fuente de Internet 24).

Los atributos también pueden clasificarse en monovalentes o polivalentes. Un atributo monovalente es aquel que tiene un solo valor para cada ocurrencia de la entidad o relación a la que pertenece. Un atributo polivalente es aquel que tiene varios valores para cada ocurrencia de la entidad o relación a la que pertenece. A estos atributos también se les denomina multivaluados, y pueden tener un número máximo y un número mínimo de valores. La cardinalidad de un atributo indica el

número mínimo y el número máximo de valores que puede tomar para cada ocurrencia de la entidad o relación a la que pertenece. El valor por omisión es (1,1). Por defecto, si las interrelaciones son exclusivas, es decir, si sólo es aplicable una entidad y una relación a otra entidad, la cardinalidad será de (0,1) o (0,n), dependiendo del tipo de relación asignado.

1.4.4. SQL.

SQL es un lenguaje para organizar, gestionar y recuperar datos almacenados en una base de datos informática.

El nombre "SQL" es una abreviatura de Structured Query Languaje (Lenguaje de consultas estructurado). Como su propio nombre indica, SQL es un lenguaje informático que se puede utilizar para interaccionar con una base de datos y más concretamente con un tipo especifico llamado base de datos relacional (fuente de Internet 30).

SQL es a la vez un lenguaje fácil de aprender y una herramienta completa para gestionar datos. Las peticiones sobre los datos se expresan mediante sentencias, que deben escribirse de acuerdo con unas reglas sintácticas y semánticas de este lenguaje.

Su aprendizaje no solo sirve para esta aplicación sino, también, para todas las existentes en el mercado que soporten este lenguaje ya que es un lenguaje estándar por haberse visto consolidado por el Instituto Americano de Normas (ANSI) y por la Organización de Estándares Internacional (ISO).

1.4.4.1. Sentencias de selección o consultas.

Las consultas son el corazón del lenguaje SQL. La sentencia SELECT, que se utiliza para expresar consultas en SQL, es la más potente y compleja de las sentencias SQL.

La sentencia SELECT recupera datos de una base de datos y los devuelve en forma de resultados de la consulta. Consta de seis cláusulas: las dos primeras (SELECT y FROM) obligatorias y las otras cuatro opcionales (fuente de Internet 30).

La forma de la sentencia SELECT es:

```
SELECT [DISTINCT] {* | expresión_columna, ...}
FROM nombretabla [alias_tabla] ...
[WHERE expresión1 operador expresion2]
[GROUP BY {expresión_columna, ...} ]
[HAVING {condición} ]
[UNION [ALL] (SELECT ...)]
```

[ORDER BY {expresión_orden [DESC | ASC], ...]

Cláusulas.

Las cláusulas son condiciones de modificación utilizadas para definir los datos que desea seleccionar o manipular (fuente de Internet 30):

FROM Utilizada para especificar la tabla de la cual se van a seleccionar los registros.
WHERE Utilizada para especificar las condiciones que deben reunir los registros que se van a seleccionar.
GROUP BY Utilizada para separar los registros seleccionados en grupos específicos.
HAVING Utilizada para expresar la condición que debe satisfacer cada grupo.
ORDER BY Utilizada para ordenar los registros seleccionados de acuerdo con un orden específico.

Operadores Lógicos.
AND Es el "y" lógico. Evalúa dos condiciones y devuelve un valor de verdad sólo si ambas son ciertas.
OR Es el "o" lógico. Evalúa dos condiciones y devuelve un valor de verdad si alguna de las dos es cierta.
NOT Negación lógica. Devuelve el valor contrario de la expresión.

Operadores de Comparación.
< Menor que.
> Mayor que.
<> Distinto de.
<= Menor ó Igual que.
>= Mayor ó Igual que.
= Igual que.
BETWEEN Utilizado para especificar un intervalo de valores.
LIKE Utilizado en la comparación de un modelo.
In Utilizado para especificar registros de una base de datos.

Funciones de Agregado.
AVG Utilizada para calcular el promedio de los valores de un campo determinado.
COUNT Utilizada para devolver el número de registros de la selección.
SUM Utilizada para devolver la suma de todos los valores de un campo determinado.
MAX Utilizada para devolver el valor más alto de un campo especificado.
MIN Utilizada para devolver el valor más bajo de un campo especificado.

Capítulo 2. Análisis del sistema.

En el capítulo anterior se trataron de manera general las herramientas que serán utilizadas en este capítulo y los subsecuentes para el desarrollo del sistema de una manera metódica. En lo posible, se hace referencia a los temas del marco teórico dentro del siguiente análisis.

2.1. Ingeniería de software como ayudante en el análisis.

2.1.1. Análisis.

Previamente se ha enfocado a dar anticipadamente una senda por la que será analizado, diseñado e implementado el sistema en cuestión. Ahora, será aplicado todo lo correspondiente al análisis con la ingeniería de software como herramienta primaria.

2.1.1.1. Enunciado definitivo del problema por resolver.

Es requerido un sistema de cómputo que utilice tecnología basada en bases de datos que sea auxiliar en la enseñanza de cualquier materia de la educación básica:

Bases de datos como herramienta de consulta en la educación básica.

Habiendo hecho el enunciado, se comenta que esta aplicación debe tener almacenada información de consulta de diversos géneros. En muchas ocasiones no es tan factible tener a la mano una enciclopedia específica que nos proporcione la información deseada. Una alternativa que nos permita obtener fácilmente el contenido de algún tema en específico sin contar con la enciclopedia es precisamente una base de datos. Esta puede contener la información necesaria para satisfacer lo que se requiere en un momento dado, en este caso, como ya ha sido mencionado, cualquier materia de nivel básico. Esto suena interesante debido a que la información que contendrá este proyecto es variable y puede ser mayor o menor, lo importante es que el sistema en cuestión tenga un diseño robusto para que pueda ser aplicada la reingeniería y control de calidad de forma adecuada.

El común denominador de todas las operaciones de consulta es el diseño y construcción de un motor de búsqueda asociado a ésta lo cual permitirá un rendimiento mayor en el acceso a la información.

2.1.1.2. Estrategia de solución computarizada para el problema.

Se ha pensado que el sistema contenga una base de datos con la información auxiliar y relevante de la materia seleccionada, y que contenga, además de la información textual, un ayudante visual (imágenes) que resultan atractivas para los niños y que patrocinan una mejor aprensión de conocimientos.

En este caso, los niños que utilicen este sistema serán los usuarios finales, aunque no exclusivos, dado que los profesores también podrán consultar esta información, teniendo incluso, el privilegio de actualizarla, modificarla o corregirla.

Existen numerosas plataformas sobre la cual podría ser implementada el sistema, sin embargo, es bien sabido que la plataforma que más es utilizada por usuarios comunes es Microsoft Windows en varios de sus versiones de sistemas operativos. Y aprovechando la ventaja de ser un ambiente gráfico, resulta bastante atractivo el basar este sistema sobre esta plataforma, programado en alguno de los lenguajes visuales orientados a objetos que existen bajo la misma.

Lo anterior nos da grandes ventajas, como la facilidad de uso que tendrán los usuarios finales, calculando que el tiempo de programación no es muy cuantioso y que es completamente factible.

Además que el desarrollo de aplicaciones Web ha venido creciendo en gran medida durante los últimos años, hace considerable también, una posibilidad de ampliar este sistema para ser consultado desde Internet.

En este caso, además, la implementación de métodos de búsqueda estratégica para la educación básica además de ofrecer un diseño de bases de datos robusto para aplicaciones similares.

El desarrollo del sistema, que sea probado y cumpla con la función para la cual fue creado y aplicado al aprendizaje de niños de la educación básica.

El sistema deberá ser descompuesto en módulos. Cada módulo debe incluir el método que se empleará para su verificación.

En el caso de este proyecto se necesita que el sistema sea robusto, flexible y funcional.

2.1.2. Definición de requisitos.

Aquí son descritas las consideraciones que deben ser cumplidas, al menos, para que el sistema sea considerado exitoso.

2.1.2.1. Especificación de requisitos.

Para este sistema en particular, a continuación se describen los requisitos que deberá cumplir:

Requisitos funcionales.

Por medio del lenguaje natural definiremos lo siguiente:

1 El sistema se compondrá de una base de datos con información que ayuden y complementen la enseñanza de la educación a nivel básico, de cinco materias.
2 El sistema tendrá un menú principal donde se podrá elegir entre las materias (ya mencionadas) que se dispondrán en la base de datos.
3 Actualmente se plantean las cinco materias de educación básica: matemáticas, español, historia, civismo y ciencias naturales.
4 Cada materia será consultada por aparte de las demás.
5 Para cada materia se debe: tener desplegado principal, poder efectuar búsqueda, tener diccionario de términos, un desplegado de objetos, evaluación, imprimir la información mostrada y ayuda correspondiente.
6 Del punto anterior se deriva que las tablas de cada materia tengan la misma estructura para que los módulos a programarse manipulen la información de cualquier materia, y no sean programados para una en específico.
7 El desplegado principal mostrará a la vez el concepto con su significado o información respectiva y, a lo más, un objeto asociado.
8 Las búsquedas se aplicarán sólo sobre la materia elegida y deberán ser sobre los atributos de concepto, o bien, sobre el atributo de significado o información correspondiente.
9 El diccionario de términos será propiamente un vocabulario de la terminología utilizada en la materia dada.
10 Deben poder ser vistos únicamente los objetos que están asociados a los conceptos.
11 La evaluación tiene un juego de preguntas sobre los temas aprendidos en la materia elegida.
12 Puede enviarse a impresora el dato desplegado en pantalla en el momento.
13 Se plantea que en un futuro sea posible la actualización o agregar otras materias, además de las ya incluidas.

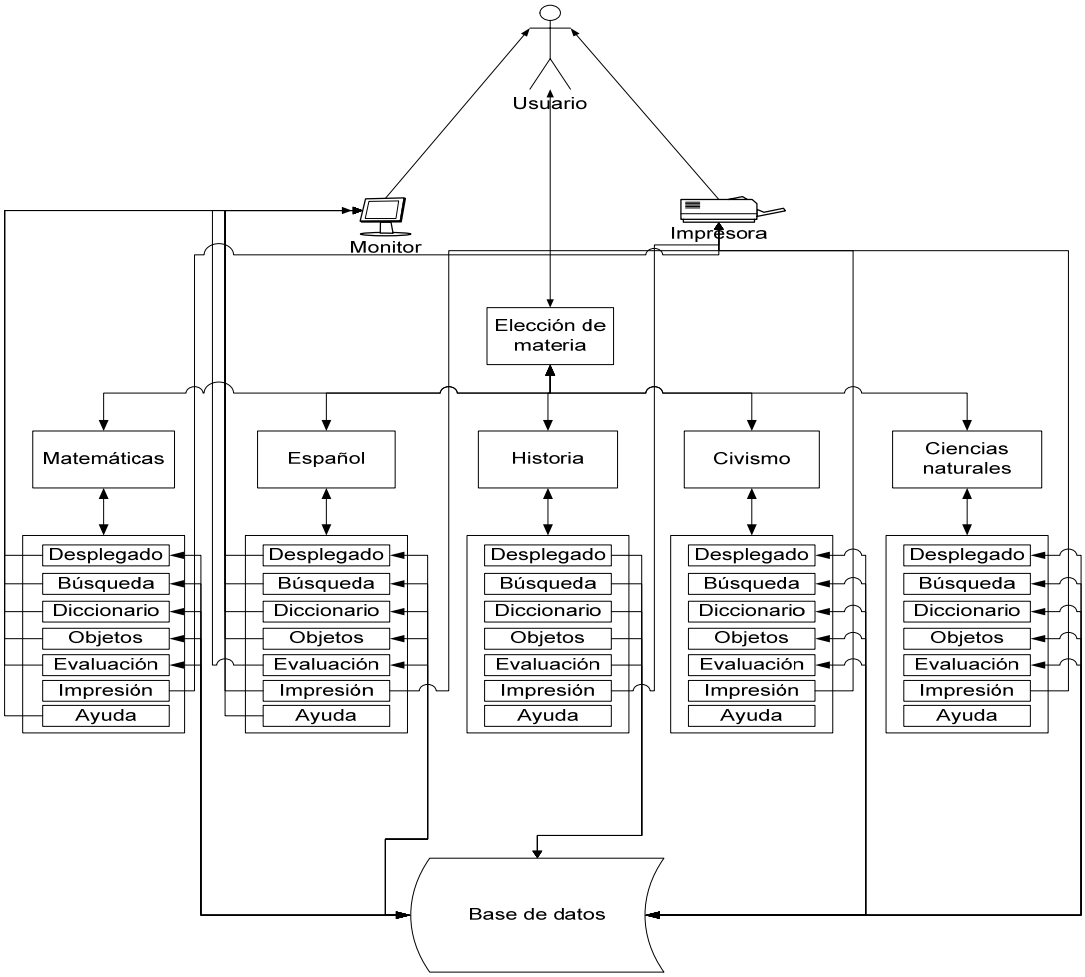
Requisitos no funcionales.

Se prevé que la base de datos sea local al sistema para un mejor desempeño. Esto implica que se encuentre en la misma computadora y en la misma ruta. Aunque no se descarta a futuro una aplicación Web para que esta base de datos pueda llegar a ser consultada por Internet.

Por el momento no se han previsto más requisitos no funcionales que el anterior, pero es posible que sean agregados más para el mejor desempeño del sistema.

2.1.2.2. Bosquejo de la arquitectura del sistema.

Dados los requisitos anteriores, a continuación se presenta un diagrama que permitirá visualizar al sistema en cuestión.



Nótese que todas las materias hacen uso de los mismos módulos como se mencionó en el punto 6 de la especificación de requisitos, aunque para su mejor comprensión, en el anterior diagrama aparezcan separados.

2.1.3. Diseño.

Una vez establecido un punto de referencia como es el diagrama de la arquitectura del sistema, es posible comenzar a trabajar en las partes que lo compondrán.

2.1.3.1. Modularidad.

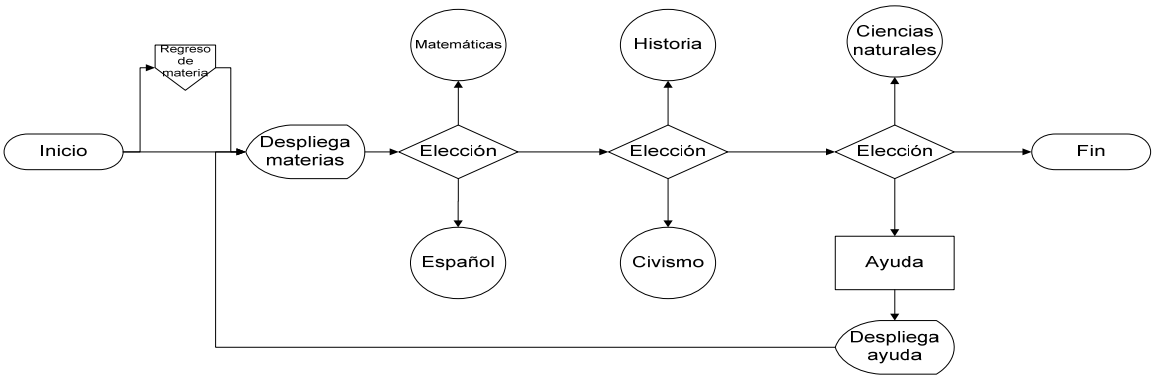
Dada la extensión del presente proyecto se tomarán en cuenta los principios de modularidad (mencionados en el capítulo anterior), aplicados a todo el sistema. Con ellos, se analizará el sistema durante la especificación de requisitos, y serán mejor detallados en el siguiente capítulo del presente documento.

2.1.3.2. Metodología a aplicarse en el diseño del sistema.

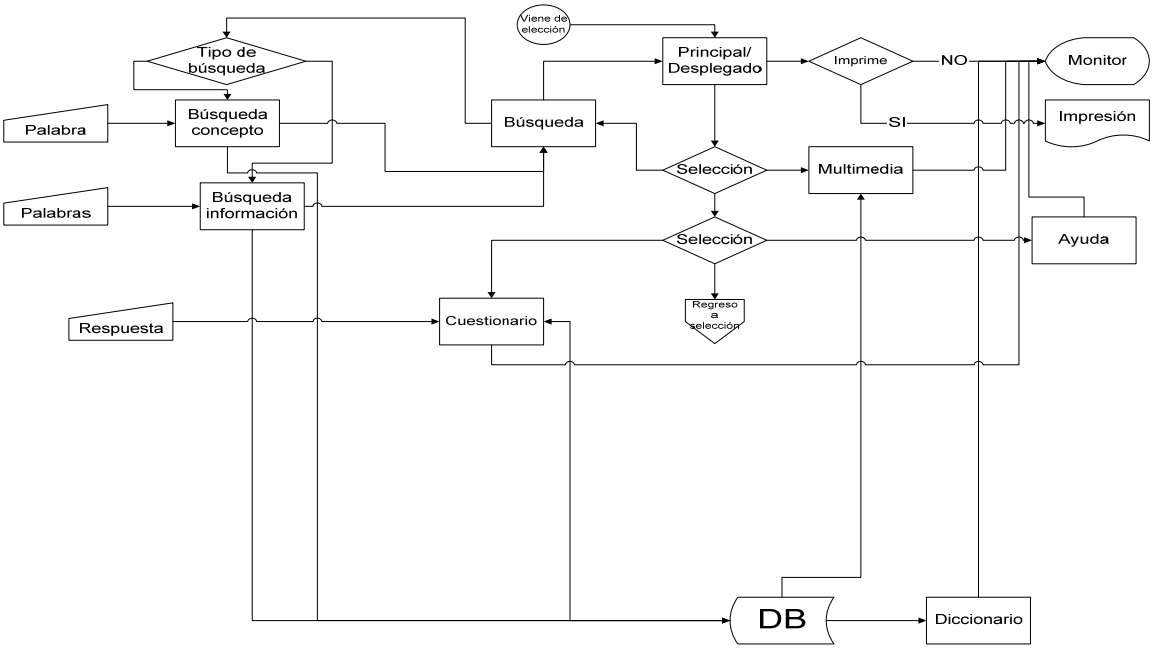
Como ya fue mencionado en el capítulo anterior, las dos descomposiciones que serán aplicadas al proyecto son la algorítmica y orientada a objetos. En el caso de orientado a objetos deben ser descritos durante la programación por el paradigma que incluye, mientras que la algorítmica se elaborará un diagrama de flujo que la contiene y que podrá ser observado en el párrafo siguiente.

2.1.3.3. Diagrama de flujo del sistema.

El siguiente algoritmo corresponde al menú principal, sólo para seleccionar con qué materia se trabajará.



Ahora, una vez electa la materia, es el mismo algoritmo para todas y cada una, dadas las especificaciones ya mencionadas.



Con las siguientes funciones:

- Búsqueda: se puede elegir sobre qué atributos se efectuará:
    - Búsqueda concepto: pide palabra a buscar sobre el tema en el atributo concepto en la tabla de la materia que corresponda.
    - Búsqueda información: busca la palabra o palabras, pero ahora en el atributo de información en la materia respectiva.
- Impresión. Manda a impresora el dato que se muestra en principal.
- Desplegado/principal: muestra el contenido uno a uno de todos los temas que tenga la tabla y en él se puede elegir búsqueda, diccionario, cuestionario, impresión, ayuda, desplegar objetos, en pantalla
- Multimedia: muestra los objetos multimedia que tiene la base de datos de la materia elegida.
- Cuestionario: efectúa la evaluación.
- Diccionario: muestra el diccionario o vocabulario de la materia actual, y es posible hacer búsqueda sobre los atributos:
    - Buscar palabra o concepto.
    - Buscar significado.
    - Desplegar todo el diccionario o vocabulario.

Nuevamente se hace hincapié en que las tablas de cada materia deberán tener la misma estructura, dado que compartirán el programa que las manipulará.

2.1.4. Consideraciones de programación.

En esta sección, se definen las herramientas a utilizarse en la programación, así como el lenguaje a utilizarse.

2.1.4.1. Lenguaje de programación

Como el factor más importante en la construcción de un sistema es el paradigma de programación utilizado, y, en este caso es orientado a objetos, el cual tiene varias ventajas sobre otras metodologías de programación, por tanto, es el método de programación que será utilizado, dado las características que tiene (herencia, encapsulamiento, etc.), y se ha pensado que el lenguaje a utilizar para el desarrollo de este sistema sea Delphi versión 5 o posteriores, que es un lenguaje de programación orientado a objetos basado en el lenguaje procedural Pascal, el cual ofrece muchas ventajas, de las cuales, nos interesa su manejo de bases de datos, ser un lenguaje visual y bastante robusto.

2.1.4.2. Herramientas automatizadas a utilizarse.

En este caso, serán utilizadas las herramientas como el depurador que incluye el entorno del Delphi, por su flexibilidad y la facilidad de programación, aunque no por eso deja de ser robusto. Además del SGBD, que en este caso será Microsoft Access 2003, y algún editor de imágenes.

2.2. Definición de la base de datos.

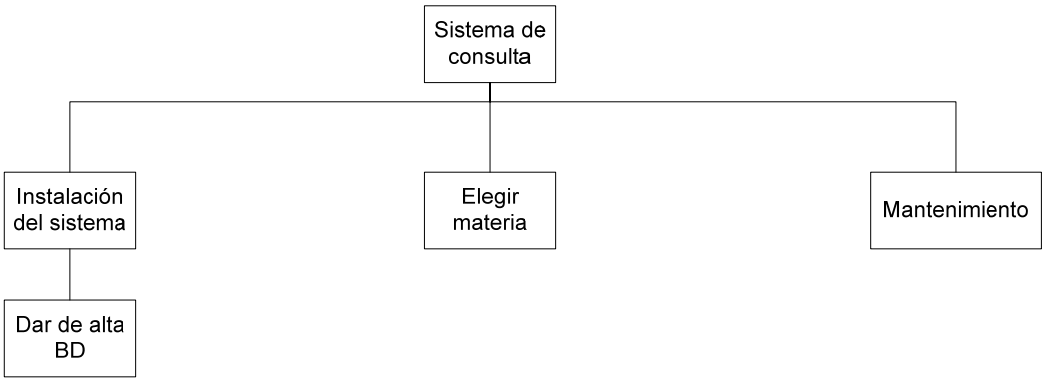
2.2.1. Metodología de bases de datos.

Los métodos utilizados en este proyecto son el modelo funcional y el modelo entidad-relación para el diseño de la base de datos, ya explicados en el capítulo anterior. Ahora, se procederá a aplicar cada uno de ellos en el sistema a desarrollar.

2.2.1.1. Modelo funcional.

A partir del “borrador” de la arquitectura del sistema, obtenido de los requisitos funcionales, es posible ahora descomponerlo funcionalmente como se plantea a continuación:

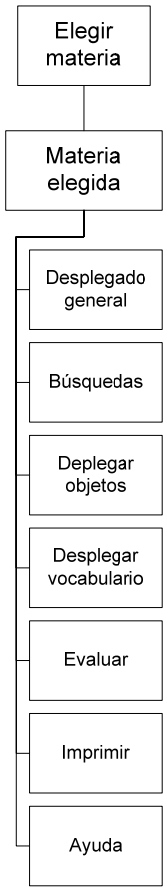
Se propone el siguiente esquema para explicar funcionalmente el sistema en cuestión en el nivel más alto:



Descripción:

1. Instalación del sistema. Este proceso es el primero que se debe hacer para poder utilizar el paquete. Se compone de una función:
   a. “Dar de alta BD”. Esto es agregar la base de datos en el DSN del sistema para que el paquete pueda hacer uso de la misma.
2. Elegir materia. Una vez instalada la aplicación y su correspondiente juego de datos, se podrá hacer la elección de la materia a consultarse.
3. Mantenimiento. Sólo se dará en caso que el sistema o su base de datos presenten fallos.

Ahora, se decompondrá la función “Elegir materia”:



Descripción:

2.1 Desplegado general. Una vez que se ha elegido la materia a usarse, se desplegará el dato de la tupla actual de la base de datos correspondiente, así como un menú de opciones. También se podrá pedir la tupla siguiente o anterior, así como la primera y la última.
2.2 Búsquedas. Se presenta la búsqueda de información sobre la base de datos principal de la materia elegida, y solicita el dato a encontrar. (se subdivide más adelante)
2.3 Desplegar objetos. Presenta la imagen asociada al dato que sea presentado en el desplegado general de la materia elegida. Puede navegarse en esta tabla, pudiendo solicitar el siguiente, el anterior, el primero y el último objeto.
2.4 Desplegar vocabulario. Muestra el contenido del vocabulario asociado a la materia, y solicita datos para ser encontrados en éste. (se subdivide más adelante)
2.5 Evaluar. Presentará una serie de preguntas (de acuerdo a la materia elegida) y solicitará una respuesta de entre tres posibles.

2.6 Imprimir. Toma la tupla que esté en el desplegado principal y solicitará la autorización del usuario para mandarla a una impresora.
2.7 Ayuda. Dará una orientación sobre el uso del sistema en general, sin llegar a sustituir los manuales de usuario.

Todas las funciones anteriores son las mismas (compartidas) para todas y cada una de las materias disponibles.

Del esquema anterior, ahora se procederá a hacer la descomposición de la función “Búsquedas” y de “Desplegar vocabulario”:

Descripción:

2.2.1 Buscar tema. El tema a ser encontrado será solicitado al usuario y buscará sobre el atributo tema de la tabla.
2.2.2 Buscar información. También se podrá buscar en el atributo información y se pedirá el dato a ser encontrado.
2.4.1 Desplegar todo. Por defecto desplegará una tupla (a la vez) de la tabla de vocabulario, y se podrá pedir la anterior, la siguiente, la primera y la última de la tabla, de acuerdo al orden en que se encuentre.
2.4.2. Buscar concepto. Se solicitará éste para ser buscado en el mismo atributo, y efectuará la localización del dato solicitado.
2.4.3. Buscar significado. De igual manera que el anterior, pero en el atributo que almacenará el significado.

Todas estas funciones con la descripción parecen ser complejas, sin embargo se podrán observar en el capítulo 4, en la sección módulos (4.2.2).

2.2.1.2. Modelo Entidad-Relación.

Para complementar el Modelo Funcional, es necesario ahora, hacer el modelado con MER, para lo cual se debe comenzar identificando las entidades que almacenarán información.

Entidades. De acuerdo a la especificación de requisitos se comienza la enumeración de las entidades:

Entidad 1.
Nombre: MATERIAS
Descripción: Listado de las materias disponibles, así como sus claves.

Entidad 2.
Nombre: MATERIA
Descripción: Temas, subtemas y sus explicaciones relacionados con la materia que va a ser tratada.

Entidad 3.
Nombre: IMAGENES
Descripción: Ilustraciones de la materia respectiva.

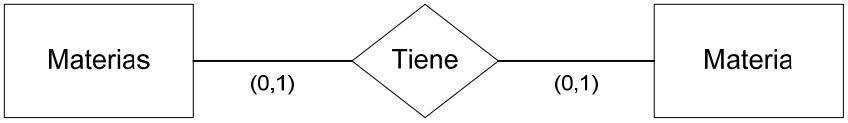
Entidad 4.
Nombre: VOCABULARIO
Descripción: Diccionario alusivo a la materia utilizada.

Entidad 5.
Nombre: CUESTIONARIO
Descripción: Preguntas y respuestas a temas de la materia elegida.

Relaciones. Una vez establecidas las entidades, se procede a formar las relaciones entre éstas:

Nombre: Tiene

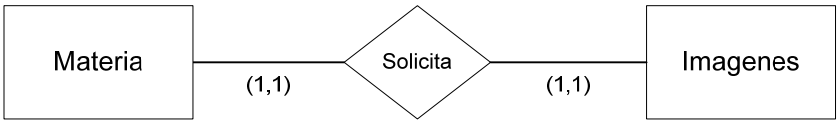
Entidades relacionadas:	Materias	-	Materia
Cardinalidad respectiva	(0,1)		(0,1)



Nota: la cardinalidad anterior es (0,1), dado que a la vez sólo puede estar activa una sola materia con un mismo tipo de relación entre la entidad “materia” y cada una de ellas, por tanto, son exclusivas una de las demás, como se mencionó en el apartado 1.4.3.2.

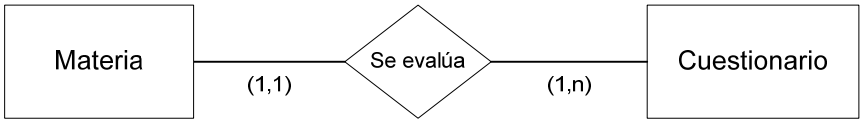
Nombre: Solicita

Entidades relacionadas:	Materia	-	Imagenes
Cardinalidad respectiva:	(1,1)		(1,1)



Nombre: Se comprende
Entidades relacionadas: Materia - Vocabulario
Cardinalidad respectiva: (1,1) (1,n)

Nombre: Se evalua
Entidades relacionadas: Materia - Cuestionario
Cardinalidad respectiva: (1,1) (1,n)



Atributos y dominios. Es necesario también, enumerar los atributos que componen cada una de las entidades, con sus correspondientes valores que pueden ser tomados, es decir, los dominios. De igual manera es necesario hacer hincapié en el atributo que dará la identificación única en cada una de las entidades, y serán denotados por el subrayado:

Nombre de entidad	**Atributos**	**Dominio de los atributos**
Materias	Clave_principal	Número entero mayor que 0
	Materia	[Español], [Matemáticas], [Ciencias Naturales], [Civismo], [Historia]. (Es posible agregar más, en caso de que el sistema se amplíe).

Nombre de entidad	**Atributos**	**Dominio de los atributos**
Materia	Clave_M	Número entero mayor que 0
	Tema	Nombre de tema
	Subtema	Nombre de subtema
	Info	Información asociada al subtema
	Objeto	Nombre del objeto
	Materia	[Español], [Matemáticas], [Ciencias Naturales], [Civismo], [Historia]. (Es posible agregar más, en caso de que el sistema se amplíe).

Nombre de entidad	**Atributos**	**Dominio de los atributos**
Imagenes	Objeto	Nombre del objeto
	Descripcion	Describe al objeto
	Imagen	Archivo tipo imagen

Nombre de entidad	Atributos	Dominio de los atributos
Vocabulario	Clave_V	Número entero mayor que 0
	Tema	Nombre de tema
	Palabra	Palabra que se busca
	Tipo	Depende de la materia
	Significado	Qué significa la palabra

Nombre de entidad	Atributos	Dominio de los atributos
Cuestionario	Clave_C	Número entero mayor que 0
	Subtema	Nombre de subtema
	R1	Posible respuesta 1
	R2	Posible respuesta 2
	R3	Posible respuesta 3
	Rc	Número entero mayor o igual que 0 y menor o igual que 2

Diagrama entidad-relación. Una vez completados la identificación de entidades, relaciones y atributos, se procede a esquematizar todo lo anterior en el siguiente diagrama:

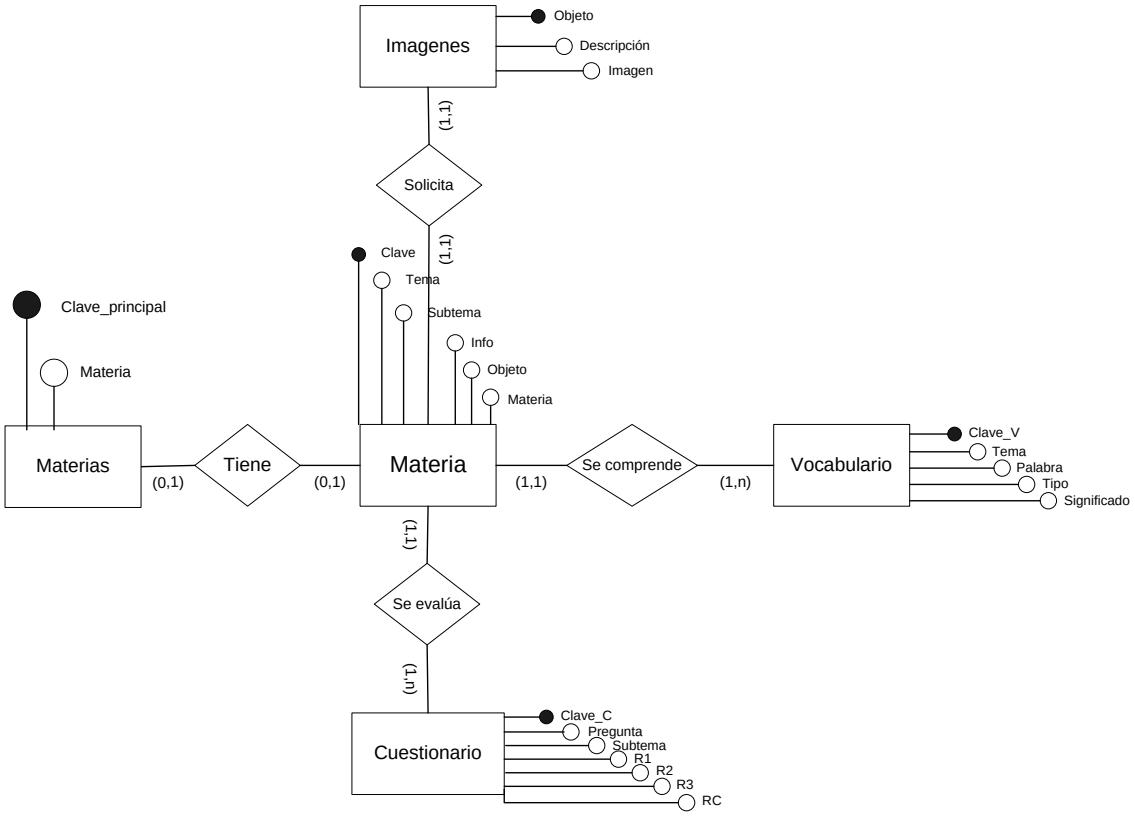


Ilustración 1

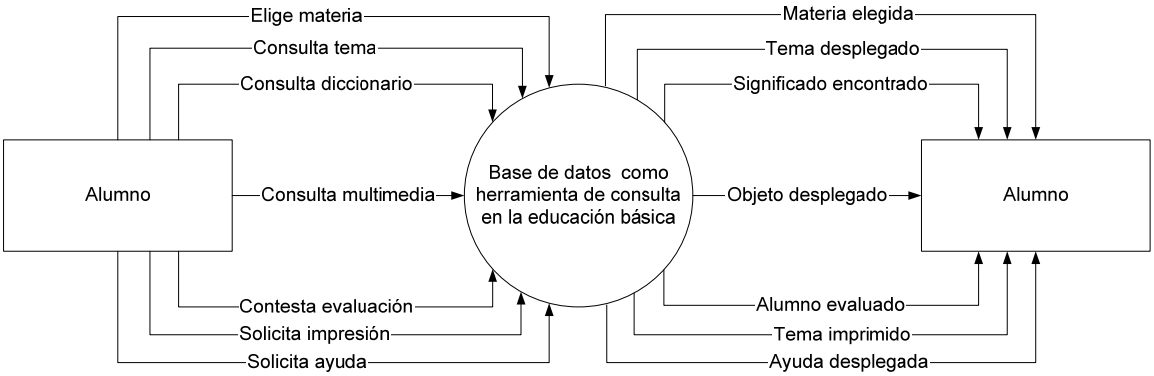
Se debe notar que en el esquema anterior (ilustración 1) no se incluyen todas las materias, dado que es un diagrama entidad-relación, y será exactamente igual para todas y cada una de las entidades "Materia" que se relacionen con la

entidad “Materias”, pudiéndose ampliar a una figura más desarrollada que se incluye en el anexo D, la cual corresponde a las entidades y relaciones que tendrá este sistema en particular, de acuerdo a la modelación de este apartado.

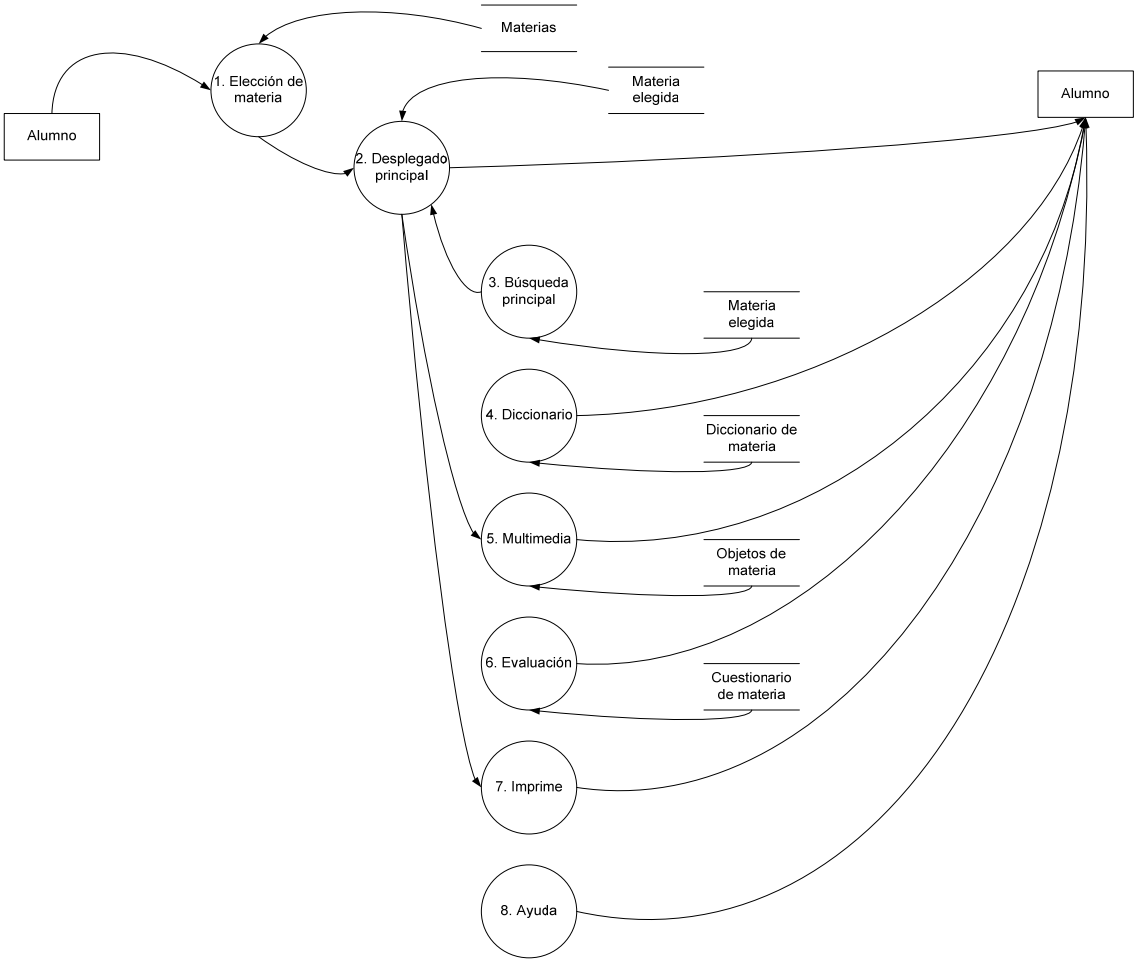
2.2.1.3. Diagrama de flujo de datos.

Los niveles del diagrama que tiene el sistema se exponen a continuación, de acuerdo al análisis ya iniciado y se irá desde el nivel 0 hasta el nivel 2. Cabe hacerse notar que una vez elegida la materia los diagramas son iguales para cualquiera, aunque se haga uso de diferente información.

Nivel 0:

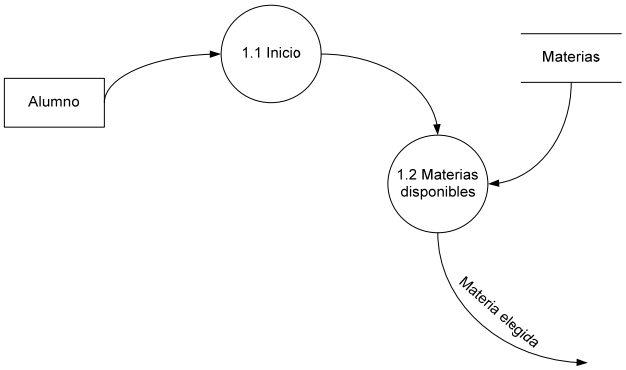


Nivel 1:

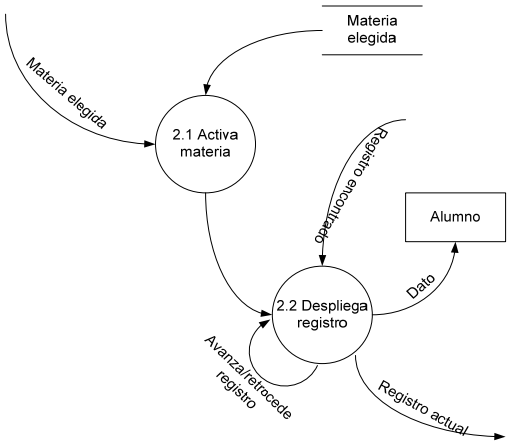


De la anterior numeración de los procesos, se da la siguiente descomposición:

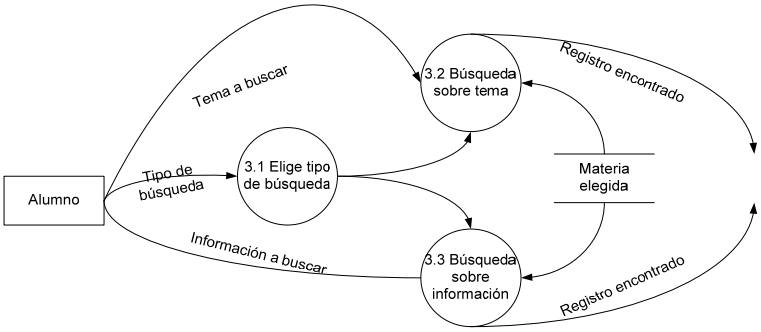
1. Nivel 2 para la elección de la materia:



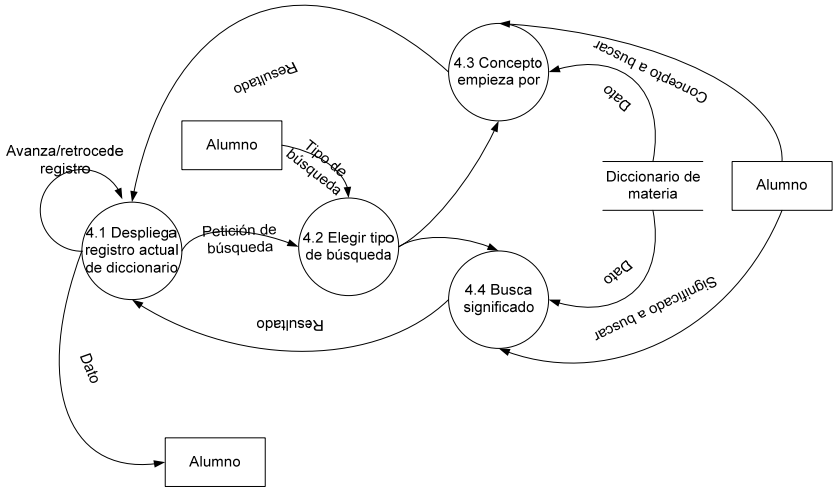
2. Nivel 2 para el desplegado principal:



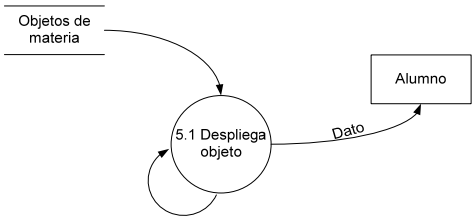
3. Nivel 2 para la búsqueda principal:



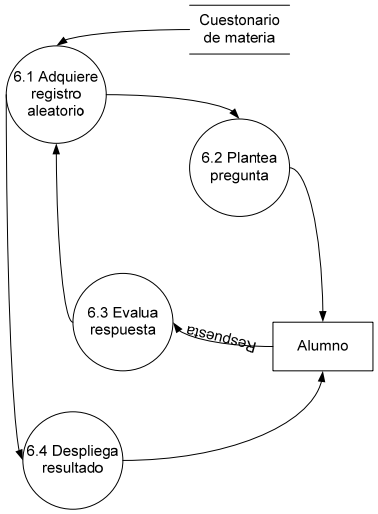
4. Nivel 2 para la búsqueda en el diccionario:



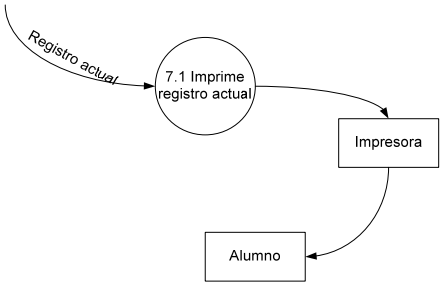
5. Nivel 2 para el desplegado multimedia:



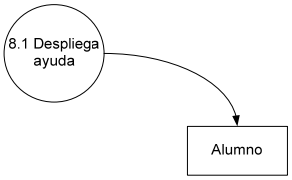
6. Nivel 2 para la evaluación:



7. Nivel 2 para la impresión:



8. Nivel 2 para la ayuda:



Nota: Los procesos anteriores hacen uso de cada materia a la vez.

A continuación se hace una especificación de procesos a partir del DFD nivel 1 y del Diagrama de flujo que está en la sección 2.1.3.3:

1. Elección de materia.
COMIENZA
    Elección = Acción de usuario
    Caso Elección de
        Caso Elección=”Español”
            Abrir tablas de materia *Español*
            Desplegado principal
        Caso Elección=”Matematicas”
            Abrir tablas de materia *Matemáticas*
            Desplegado principal
        Caso Elección=”Naturales”
            Abrir tablas de materia *Ciencias Naturales*
            Desplegado principal
        Caso Elección=”Civismo”
            Abrir tablas de materia *Civismo*
            Desplegado principal
        Caso Elección=”Historia”
            Abrir tablas de materia *Historia*
            Desplegado principal
    FIN_Caso
TERMINA

Nota: Para este proyecto sólo se incluyen las materias que fueron anticipadas, sin embargo, es posible ampliar los casos de elección de acuerdo a las necesidades de uso del sistema.

2. Desplegado principal
EMPIEZA
    Despliega registro actual de tabla principal de la materia elegida
    Registro_actual = Acción de usuario
    Caso Registro_actual=”Primero”
        Ve a REGISTRO_inicial
    Caso Registro_actual=”Atrás”
        Ve a REGISTRO_anterior
    Caso Registro_actual=”Siguiente”
        Ve a REGISTRO_siguiente
    Caso Registro_actual=”Ultimo”
        Ve a REGISTRO_final
    Despliega registro actual de tabla principal de la materia elegida
    Elección = Acción de usuario
    Caso Elección de
        Caso Elección=”Búsqueda principal”

Desplegado *búsqueda principal*
Caso Elección=”Diccionario”
Desplegado *diccionario*
Caso Elección=”Multimedia”
Desplegado *Imágenes*
Caso Elección=”Evaluación”
Desplegado *Cuestionario*
Caso Elección=”Imprime”
Desplegado *Imprime*
Caso Elección=”Ayuda”
Desplegado *Ayuda*
Caso Elección=”Regresar”
Regresa a Elección de materia (1)
FIN_Caso
TERMINA

3. Búsqueda principal
EMPIEZA
Elección = Acción de usuario
Caso Elección de
Caso Elección=”Búsqueda sobre campo concepto”
ENCONTRAR “cadena” en campo *concepto*
Si “cadena” ENCONTRADA THEN
Regresa *registro actual* a desplegado principal (2)
Si NOT
Desplegar “No encontrado, busque nuevamente”
Caso Elección=”Búsqueda sobre campo información”
ENCONTRAR “cadena” en campo *info*
Si “cadena” ENCONTRADA THEN
Regresa *registro actual* a desplegado principal (2)
Si NOT
Desplegar “No encontrado, busque nuevamente”
FIN_Caso
TERMINA

4. Diccionario
COMIENZA
Despliega registro actual de tabla diccionario de la materia elegida
Registro_actual = Acción de usuario
Caso Registro_actual=”Primero”
Ve a REGISTRO_inicial
Caso Registro_actual=”Atrás”
Ve a REGISTRO_anterior
Caso Registro_actual=”Siguiente”
Ve a REGISTRO_siguiente
Caso Registro_actual=”Ultimo”

```
            Ve a REGISTRO_final
        Despliega registro actual de tabla diccionario de la materia elegida
        Elección = Acción de usuario
        Caso Elección de
            Caso Elección=”Búsqueda sobre campo palabra”
                ENCONTRAR “cadena” en campo palabra
                Si “cadena” ENCONTRADA THEN
                    Despliega registro actual de tabla diccionario
                Si NOT
                    Desplegar “No encontrado, busque nuevamente”
            Caso Elección=”Búsqueda sobre campo significado”
                ENCONTRAR “cadena” en campo significado
                Si “cadena” ENCONTRADA THEN
                    Despliega registro actual de tabla diccionario
                Si NOT
                    Desplegar “No encontrado, busque nuevamente”
        FIN_Caso
TERMINA

5. Multimedia
COMIENZA
        Despliega registro actual de tabla objetos de la materia elegida
        Registro_actual = Acción de usuario
        Caso Registro_actual=”Primero”
            Ve a REGISTRO_inicial
        Caso Registro_actual=”Atrás”
            Ve a REGISTRO_anterior
        Caso Registro_actual=”Siguiente”
            Ve a REGISTRO_siguiente
        Caso Registro_actual=”Ultimo”
            Ve a REGISTRO_final
        Despliega registro actual de tabla objetos de la materia elegida
TERMINA

6. Evaluación
COMIENZA
        Contador = 1
        Calificación = 0
        REPETIR
            GENERA número_aleatorio
            Despliega campos Pregunta, R1, R2 y R3 del registro de_
                número_aleatorio
            Elección = Acción de usuario (0 <= Elección <= 2)
            Si Elección = campo Rc THEN
                INCREMENTA Calificación
            INCREMENTA Contador
```

HASTA que contador = 10
Desplegar Calificación
TERMINA

7. Imprime
COMIENZA
Despliega forma de impresión con registro actual de tabla de materia
Elección = Acción de usuario
Si Elección = VERDADERO THEN
Imprime forma y registro actual
TERMINA

8. Ayuda
COMIENZA
Despliega página de ayuda general.
TERMINA

2.2.1.4. Diccionario de datos.

Una vez establecidas las entidades (sección 2.2.1.2), e identificados los almacenes de datos (sección 2.2.1.3), se puede ahora proceder a establecer el diccionario de datos correspondiente al sistema.

Materias = {Materias}
Materias = @Clave_principal + Materia
Materia elegida = [Materia_E | Materia_M | Materia_N | Materia_C | Materia_H]
Materia = @Clave_M + Tema + Subtema + Info + Objeto + Materia
Diccionario de materia = [Vocabulario]
Vocabulario = @Clave_V + Tema + Palabra + Tipo + Significado
Objetos de materia = [Imagenes]
Imagenes = @Objeto + Descripcion + Imagen
Cuestionario de materia = [Cuestionario]
Cuestionario = @Clave_C + Pregunta + Subtema + R1 + R2 + R3 + Rc
Tiene = Materia.Materias + Materia.Materia
asociación exclusiva Materias y Materia a elegir
Solicita = Objeto.Materia + @Objeto.Imagenes
asociación 1 a 1 de Materia y Imagenes
Se comprende = Tema.Materia + Tema.Vocabulario
asociación 1 a n de Materia y Vocabulario
Se evalúa = Subtema.Materia + Subtema.Cuestionario
asociación 1 a n de Materia y Cuestionario
Clave_principal = 1 {número} 4
Materia = 1 {texto} 20
Clave_M = 1 {número} 4
Tema = 1 {texto} 30

Subtema = 1 {texto} 30
Info = 1 {texto} 200
Objeto = 1 {texto} 15
Clave_V = 1 {número} 4
Palabra = 1 {texto} 30
Tipo = 1 {texto} 30
Significado = 1 {texto} 100
Clave_C = 1 {número} 4
Pregunta = 1 {texto} 50
R1 = 1 {texto} 30
R2 = 1 {texto} 30
R3 = 1 {texto} 30
Rc = [1|2|3]
Descripción = 1 {texto} 20
Imagen = mapa de bits

Nota: El diccionario de datos completo (es decir, para todas las materias) que será utilizado por este sistema se encuentra en el Anexo D.

2.2.1.5. Sistema gestor de bases de datos a utilizarse.

Actualmente se considera al Access versión 2000 o posterior, dado que abarca al menos todos los requerimientos de la base de datos, ya definida, como el manejo de archivos multimedia como objetos.

2.3. Desarrollo del motor de búsqueda.

Debido a que Delphi tiene sus propios procedimientos de búsqueda, es posible aprovechar, en la medida de lo posible, esta fortaleza, sin embargo, haciendo hincapié en que será utilizada una base de datos, lo más recomendable es que sean ejecutadas consultas sobre la misma.

2.3.1. Tipos de buscadores a implementar.

Debido a que la base de datos tendrá dos campos sobre los cual se ha de buscar información, ya sea una o más palabras, es posible que sean desarrollados buscadores distintos, aunque en esencia iguales. Éstos serán descritos detalladamente a continuación.

2.3.1.1. Buscador sobre el campo “concepto”.

El tipo de buscador que se ha de programar para la búsqueda de información en la base de datos sobre el campo de “concepto” con una consulta SQL en la base de datos. Esto hace posible la localización eficiente y de manera rápida de la información, dando menos posibles respuestas.

2.3.1.2. Buscador sobre el campo de “información”.

Debido a que el campo de información tendrá todos los significados de los conceptos que ha de contener la base de datos, no es posible un buscador binario o de algún tipo de algoritmo ya establecido y deberán efectuarse, por tanto, consultas también, hasta encontrar la similitud requerida en el campo contra la información solicitada por el usuario, desplegando las coincidencias encontradas. El modo de búsqueda que se ha pensado para este módulo es una consulta con el lenguaje SQL. Se prevé que este tipo de búsqueda sea más inefectiva porque toda la información que contenga este campo puede ser mucho más extensa y variada, y es posible que se encuentren bastantes ocurrencias que cumplan con lo solicitado.

2.3.1.3. Buscador sobre el diccionario.

Dado que también está incluido el vocabulario en este sistema, es indispensable un buscador sobre el mismo, el diccionario incluido con este sistema debe proveer también una consulta solicitada para encontrar las palabras más cercanas a lo solicitado se y ejecutará la búsqueda sobre la “palabra” o su “significado”.

Capítulo 3. Diseño del sistema.

Una vez analizado el sistema en cuestión, se procederá a diseñar tanto la base de datos como el prototipo de software para el manejo de la misma. Se obtendrá el esquema relacional, el cual será normalizado y se hará la separación en módulos de los procesos que ejecutará el sistema.

3.1 Diseño de la base de datos.

3.1.1. Diseño lógico.

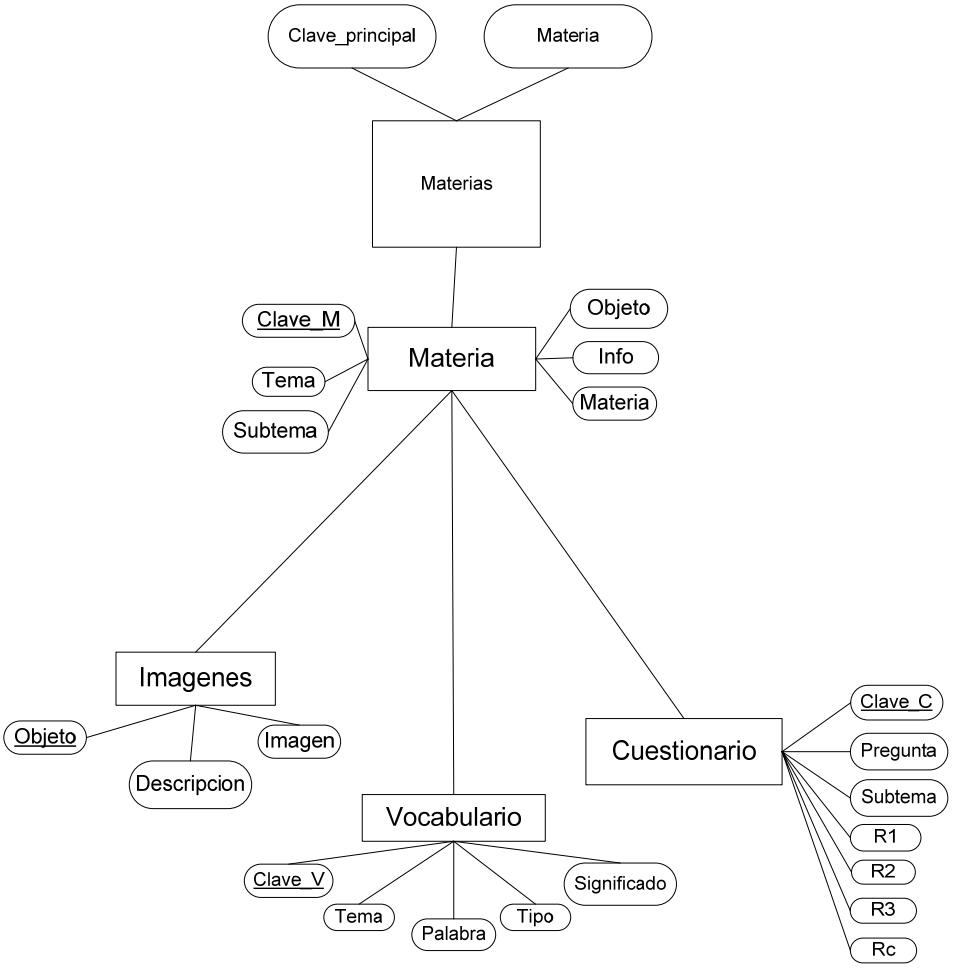
Ahora se hará una traslación del modelo entidad relación al modelo relacional, por medio de este diseño lógico. A continuación Se empezarán a integrar las relaciones (tablas llamadas de igual manera que las entidades) y sus respectivas llaves (también llamadas igual que las claves):

Entidad	**Llave primaria**
Materias	Clave_principal
Materia	Clave_M
Imagenes	Objeto
Vocabulario	Clave_V
Cuestionario	Clave_C

De lo anterior y del modelo conceptual podemos entonces definir las siguientes relaciones:

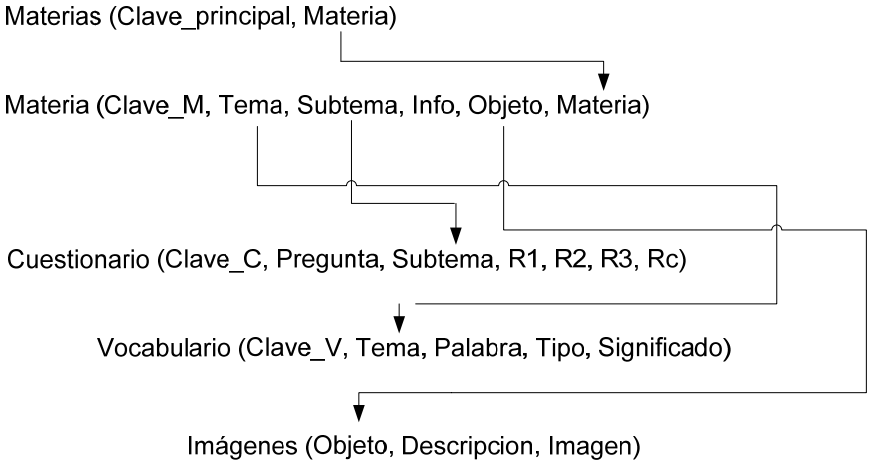
Materias (Clave_principal, Materia)
Materia (Clave_M, Tema, Subtema, Info, Objeto, Materia)
Imagenes (Objeto, Descripcion, Imagen)
Vocabulario (Clave_V, Tema, Palabra, Tipo, Significado)
Cuestionario (Clave_C, Pregunta, Subtema, R1, R2, R3, Rc)

Se obtiene también el siguiente esquema relacional:



Nota: Se obtuvo del modelo entidad-relación del capítulo anterior. El esquema completo para el sistema puede verse en el anexo D.

Y con las siguientes relaciones:



3.2. Estructura modular.

3.2.1. Contabilización e identificación de los módulos.

Como es posible observar en el diagrama de flujo de datos (apartado 2.2.1.3), existen un total de 8 procesos, que en este caso serán considerados como módulos, y sobre éstos se efectuará el diseño.

Módulo de elección de la materia. Se presentan todas las posibles opciones del área a consultar.

Módulo desplegado. En éste está el menú que llevará a los siguientes 7 módulos. Además, desplegará la base de datos (tablas INFORMACION E IMAGENES). Este módulo envía su actual registro al módulo de impresión.

Módulo de búsqueda. Aquí se hará la selección de una búsqueda sobre el atributo “concepto”, o bien, una consulta sobre el atributo “info” de la tabla INFORMACION, y devolverá un resultado, dependiendo del caso, el cual será desplegado por el módulo principal.

Módulo multimedia. Presenta en pantalla únicamente la tabla IMAGENES.

Módulo cuestionario. Ejecuta la evaluación, tomando los datos de la tabla CUESTIONARIO.

Módulo diccionario. Tiene dos secciones: la primera es la búsqueda (o consulta) sobre el atributo palabra; y la segunda sobre el atributo significado, ambos de la tabla DICCIONARIO.

Módulo ayuda. Sólo presentará la ayuda del sistema en pantalla.

Módulo impresión. Una vez vista una pantalla de vista previa del dato que actualmente despliegue el módulo principal, se puede presionar un botón que enviará a la impresora dicho dato.

De los módulos anteriores todos, menos “ayuda”, obtendrán información de la base de datos.

3.2.1.1. Módulo “elección de materia”.

En éste se mostrarán todas las materias disponibles para ser manipuladas en cada uno de los siguientes módulos, y que serán iguales para todas éstas.

3.2.1.2. Módulo “desplegado principal”.

Aquí se incluye el menú principal, con esto, se puede acceder a cualquiera de los otros 5 módulos y regresar a él y la salida del programa. Incluye también la interfase en la que se despliegan los atributos “concepto” y “info” de la tabla INFORMACION y los atributos “descripción” y “imagen” de la tabla IMAGENES. Por defecto, el sistema leerá la primera tupla de la tabla información con su respectiva relación de la tabla del anexo multimedia. Se planea agregar botones para la navegación dentro de la base de datos para avanzar y retroceder una

tupla, ir al principio y al final de la tabla. Este orden dependerá del orden que lleve el índice sobre el atributo “concepto” de la tabla INFORMACION (orden ascendente alfabético). El acceso aleatorio a la misma se deja dentro del módulo de búsqueda. El dato que actualmente se esté presentando en pantalla es el que podrá ser imprimido y éste será un atributo que será pasado al módulo de impresión.

3.2.1.3. Módulo de “búsqueda principal”.

Está compuesto por dos funciones que son prácticamente iguales. Ambas funciones están ligadas a la búsqueda que se ejecutará sobre la base de datos. Como ya se ha mencionado en la sección 2.3 del capítulo anterior, se pensó en efectuar la búsqueda en dos atributos:

A. La primera de ellas que será sobre el atributo de “concepto” corresponde en este caso a la función que aceptará una o más palabra.
B. De manera análoga, la que corresponde a la búsqueda sobre el atributo de “informacion” tiene la función que aceptará una o más palabras.

Este módulo consiste en una función que recibirá como atributos el campo sobre el cuál ejecutará la búsqueda, y la palabra, o bien, palabras que ha de encontrar en la base de datos. Como ya se ha mencionado en la sección 2.3 del capítulo 2, se pensó hacer dos procedimientos para cada esta función, que son descritos en los siguientes párrafos.

El procedimiento de búsqueda sobre ambos campos, corresponde a una consulta con SQL sobre el atributo correspondiente de la manera siguiente:

```
SELECT CONCEPTO, INFORMACION
FROM TABLA_INFORMACION
WHERE CONCEPTO LIKE “%cadena_a_buscar%”
```

O bien:

```
SELECT CONCEPTO, INFORMACION
FROM TABLA_INFORMACION
WHERE INFORMACION LIKE “%cadena_a_buscar%”
```

Si no es encontrado el dato buscado, es decir que la consulta sea igual al vacío, entonces se desplegará un cuadro de diálogo con la leyenda “no encontrado”, en caso contrario, los resultados de cualquiera de las dos maneras de buscar ya descritas, serán pasados al módulo principal para su correspondiente desplegado en pantalla y posible impresión.

3.2.1.4. Módulo de “diccionario”.

De la misma manera que el módulo principal, éste presentará a la vez una tupla de la tabla DICCIONARIO, con posibilidades de avanzar y retroceder una tupla, e ir al principio y final de la tabla. Además, tendrá dos tipos de búsqueda: sobre el atributo “palabra” o sobre el atributo “significado” de la tabla mencionada.

Las búsquedas (cualquiera de las dos), serán hechas bajo el principio de consulta ya descrito en la sección 2.3 del módulo de búsqueda, es decir, con consultas en SQL.

```
SELECT PALABRA, TIPO, SIGNIFICADO
FROM TABLA_DICCIONARIO
WHERE PALABRA LIKE “%cadena_a_buscar%”
```

Para buscar en palabra, o bien:

```
SELECT PALABRA, TIPO, SIGNIFICADO
FROM TABLA_DICCIONARIO
WHERE SIGNIFICADO LIKE “%cadena_a_buscar%”
```

Para buscar en significado.

Ningún dato o atributo será devuelto de este módulo hacia algún otro.

3.2.1.5. Módulo multimedia.

Para tener una apreciación visual de las diferentes imágenes con que cuenta la base de datos, en este módulo se podrá navegar solamente en la tabla de ANEXMULT, presentando la tupla actualmente seleccionada (por defecto, al principio desplegará la primera tupla de la tabla), con la posibilidad de avanzar y retroceder una tupla, e ir al principio y final de la tabla. Este módulo no regresará dato o atributo a cualquier otro.

3.2.1.6. Módulo de cuestionario.

Presentará un procedimiento que hará preguntas de respuesta múltiple para evaluar lo aprendido. Este módulo utilizará la tabla CUESTIONARIO, que contendrá la pregunta y las posibles respuestas (3) como atributos en una tupla. Hará la comparación del número de la respuesta seleccionada contra el número de respuesta correcta de la tabla y, después de 10 preguntas, mostrará una forma que dé el número de aciertos.

Ningún atributo o valor será devuelto desde este módulo hacia el principal.

3.2.1.7. Módulo de impresión.

Mostrará una pantalla de vista previa del dato e imprimirá los mismos atributos de las tablas que se muestren actualmente en el módulo principal.

No devuelve datos o atributos a otro módulo.

3.2.1.8. Módulo de ayuda.

Sólo será desplegada la ayuda rápida del sistema. Un documento que incluya una explicación detallada del manejo del sistema será el correspondiente manual de usuario.

Tampoco devolverá datos o atributos a algún otro módulo del sistema.

3.3. Normalización de la base de datos.

3.3.1. Atributos que forman la base de datos.

Como se ha podido ver en los diagramas de la sección 3.1.1, los atributos que componen la base de datos son los siguientes:

Tabla Materias.

- Clave_principal. Identificador de las tuplas.
- Materia. Nombre de la materia que puede ser elegida.

Tabla Materia.

- Clave. Identificador de esta materia.
- Tema. Nombre del por ser visto de esta materia.
- Subtema. Nombre de la palabra o concepto a consultarse.
- Info. Explicación del subtema.
- Objeto. Nombre del objeto relacionado con el subtema y su explicación.
- Materia. Nombre de la materia que está siendo consultada.

Tabla Cuestionario.

- Clave_C. Identificador de la tabla.
- Pregunta. Enunciado de la pregunta a formularse.
- Subtema. Concepto de la tabla principal de la materia que relaciona la pregunta.
- R1. Posible respuesta 1.
- R2. Posible respuesta 2.
- R3. Posible respuesta 3.
- Rc. Número de respuesta correcta.

Tabla Vocabulario.

- Clave_V. Identificador de la tabla.
- Tema. Atributo de la tabla principal de la materia con que se relaciona la palabra.
- Palabra. Concepto del que se dará la explicación.
- Tipo: Hace alusión a la naturaleza de la palabra, como es verbo, sustantivo o adjetivo.
- Significado. La explicación de la palabra.

Tabla Imagenes.

- Objeto. Nombre identificador de la tabla.
- Descripción. Breve explicación y origen de la imagen a mostrar.
- Imagen. Atributo que contiene la imagen a ser desplegada.

3.3.1.1. Definición funcional de atributos.

Como podemos deducir de los detalles de los atributos anteriores, tenemos que las llaves candidatas son:
“Clave_principal” de la tabla MATERIAS.
“Clave_M” de las tablas de la materias.
“Objeto” de las tablas de imágenes.
“Clave_C” de las tablas de cuestionario.
“Clave_V” de las tablas de vocabulario.

De hecho, éstas serán las llaves primarias de las tablas correspondientes, dado que no existen más llaves candidatas.

Ahora, todos son atributos primos ya que forman parte de una llave candidata, y por tanto ahora serán vistas las dependencias funcionales que se crean:

Tabla MATERIAS:
Llave candidata: Clave_principal
Atributos primos: Clave_principal
Llave primaria: Clave_principal

Clave_principal $\Rightarrow$ Materia
Clave_principal cumple con la definición de superllave.

Tablas de cada materia:
Llave candidata: Clave_M
Atributos primos: Clave_M
Llave primaria: Clave_M

Clave $\Rightarrow$ Tema, Subtema, Info, Objeto, Materia
Clave cumple con la definición de superllave.

Tablas de IMAGENES:
Llave candidata: Objeto
Atributos primos: Objeto
Llave primaria: Objeto

Objeto $\Rightarrow$ Descripción, Imagen
Objeto cumple con la definición de superllave.

Tablas de CUESTIONARIO:
Llave candidata: Clave_C
Atributos primos: Clave_C
Llave primaria: Clave_C

Clave_C ⇒ Pregunta, Subtema, R1, R2, R3, Rc
Clave_C cumple con la definición de superllave.

Tablas de VOCABULARIO:
Llave candidata: Clave_V
Atributos primos: Clave_V
Llave primaria: Clave_V

Clave_V ⇒ Tema, Palabra, Tipo, Significado
Clave_V cumple con la definición de superllave.

Ahora se tendrán las formas normales.

3.3.1.2. Primera forma normal.

En esta fase de normalización deben ser eliminadas las dependencias incompletas.

- Tabla Materias.

La relación se encuentra con las dependencias funcionales ya mencionadas y está de la siguiente manera:

Materias (Clave_principal, Materia)

Todos los atributos dependen funcionalmente de la llave primaria. √

Por lo tanto, la relación Materias se encuentra en la primera forma normal.

- Tablas de cada materia.

Materia (Clave_M, Tema, Subtema, Info, Objeto, Materia)

Todos los atributos dependen funcionalmente de la llave primaria. √

Por tanto, la relación Materia está en la primera forma normal.

- Tablas de imágenes.

Imagenes (Objeto, Descripcion, Imagen)

Todos los atributos dependen funcionalmente de la llave primaria. √

Por tanto, la relación Imagenes está en la primera forma normal.

- Tablas de cuestionario.

Cuestionario (Clave_C, Pregunta, Subtema, R1, R2, R3, Rc)

Todos los atributos dependen funcionalmente de la llave primaria. √

Por tanto, la relación Cuestionario está en la primera forma normal.

- Tablas de vocabulario.

Vocabulario (Clave_V, Tema, Palabra, Tipo, Significado)

Todos los atributos dependen funcionalmente de la llave primaria. √

Por tanto, la relación Vocabulario está en la primera forma normal.

3.3.1.3. Segunda forma normal.

En esta etapa deben ser eliminadas las dependencias parciales:

- Tabla Materias.

Materias (Clave_principal, Materia)

Véase que no existen dependencias parciales. Todos los atributos no primos dependen funcionalmente de la llave candidata que contiene a los atributos primos de la relación. √

Se observa también que está en la primera forma normal. √

Por tanto, la relación Materias está en la segunda forma normal.

- Tablas de materias.

Materia (Clave_M, Tema, Subtema, Info, Objeto, Materia)

No hay dependencias parciales. Todos los atributos no primos dependen funcionalmente de la llave candidata (con los atributos primos). √

Materia ya estaba en primera forma normal. √
Por tanto, la relación Materia está en la segunda forma normal.

- Tablas de imágenes.

Imagenes (Objeto, Descripcion, Imagen)

No hay dependencias parciales. Todos los atributos no primos dependen funcionalmente de la llave candidata (con los atributos primos). √

Imagenes ya estaba en primera forma normal. √

Por tanto, la relación Imagenes está en la segunda forma normal.

- Tablas de cuestionario.

Cuestionario (Clave_C, Pregunta, Subtema, R1, R2, R3, Rc)

Todos los atributos no primos dependen funcionalmente de la llave candidata que a su vez contiene a los atributos primos. √

Cuestionario está en primera forma normal. √

Por tanto, la relación Cuestionario está en la segunda forma normal.

- Tablas de vocabulario.

Vocabulario (Clave_V, Tema, Palabra, Tipo, Significado)

Todos los atributos no primos dependen funcionalmente de la llave candidata que contiene a los atributos no primos. √

Vocabulario ya está en primera forma normal. √

Por tanto, la relación Vocabulario está en la segunda forma normal.

3.3.1.4. Tercera forma normal.

Ahora se deben eliminar las dependencias transitivas.

- Tabla Materias.

Materias (Clave_principal, Materia)

Se puede observar que no existen dependencias parciales. Todos los atributos no primos NO dependen funcionalmente de otros atributos no primos de la relación. √

Podemos observar también que está en la segunda forma normal. √

Por tanto, la relación Materias está en la tercera forma normal.

- Tablas de materias.

Materia (Clave_M, Tema, Subtema, Info, Objeto, Materia)

No hay dependencias transitivas. Todos los atributos no primos NO dependen funcionalmente de otros atributos no primos. √

Materia ya estaba en segunda forma normal. √

Por tanto, la relación Materia está en la tercera forma normal.

- Tablas de imágenes.

Imagenes (Objeto, Descripcion, Imagen)

No hay dependencias transitivas. Todos los atributos no primos NO dependen funcionalmente de otros atributos no primos. √

Imagenes ya estaba en segunda forma normal. √

Por tanto, la relación Imagenes está en la tercera forma normal.

- Tablas de cuestionario.

Cuestionario (Clave_C, Pregunta, Subtema, R1, R2, R3, Rc)

Todos los atributos no primos NO dependen funcionalmente de otros atributos no primos. √

Cuestionario está en segunda forma normal. √

Por tanto, la relación Cuestionario está en la tercera forma normal.

- Tablas de vocabulario.

Vocabulario (Clave_V, Tema, Palabra, Tipo, Significado)

Todos los atributos no primos NO dependen funcionalmente de otros atributos no primos. √

Vocabulario ya está en segunda forma normal. √

Por tanto, la relación Vocabulario está en la tercera forma normal.

3.3.1.5. Cuarta forma normal.

Se deben tener todas las dependencias funcionales definidas por una superllave.

- Tabla MATERIAS.

Materias (Clave_principal, Materia)

Véase ahora que todas las dependencias funcionales están definidas por la llave candidata Clave_principal.

Lo anterior implica que todos los atributos no primos están definidos por una superllave. √

Por tanto, la relación Materias está en la cuarta forma normal.

- Tablas de materias.

Materia (Clave_M, Tema, Subtema, Info, Objeto, Materia)

Todas las dependencias funcionales están definidas por la llave candidata Objeto, que a su vez cumple con ser superllave. √

Por tanto, la relación Materia está en la cuarta forma normal.

- Tablas de imágenes.

Imagenes (Objeto, Descripcion, Imagen)

Todas las dependencias funcionales están definidas por la llave candidata Objeto, que a su vez cumple con ser superllave. √

Por tanto, la relación Imagenes está en la cuarta forma normal.

- Tablas de cuestionario.

Cuestionario (Clave_C, Pregunta, Subtema, R1, R2, R3, Rc)

Todas las dependencias funcionales están definidas por la llave candidata Objeto, que a su vez cumple con ser superllave. √

Por tanto, la relación Cuestionario está en la cuarta forma normal.

- Tablas de vocabulario.

Vocabulario (Clave_V, Tema, Palabra, Tipo, Significado)

Todas las dependencias funcionales están definidas por la llave candidata Objeto, que a su vez cumple con ser superllave. √

Por tanto, la relación Vocabulario está en la cuarta forma normal.

3.3.1.6. Nuevo diseño conceptual.

Se observa que el diagrama relacional que se había contemplado en el apartado 3.1.1 no cambió, aún dado que las formas normales se aplicaron, no afectaron dicho diseño.

Ahora bien, es necesario definir el tipo de los atributos de las tablas, de acuerdo a los dominios definidos en la sección 2.2.1.2 y al SGBD elegido en la sección 2.1.4.2.

Tabla Materias

Nombre del campo:	Clave_principal
Tipo de datos:	Autonumérico
(Hacerlo “Clave principal”)	

Nombre del campo:	Materia
Tipo de datos:	Texto
Tamaño del campo:	20
Requerido:	Sí

Tabla Materia

Nombre del campo:	Clave_M
Tipo de datos:	Autonumérico
(Hacerlo “Clave principal”)	

Nombre del campo:	Tema
Tipo de datos:	Texto
Tamaño del campo:	30
Requerido:	Sí

Nombre del campo:	Subtema
Tipo de datos:	Texto
Tamaño del campo:	30
Requerido:	Sí

Nombre del campo:	Info
Tipo de datos:	Texto

Tamaño del campo: 200
Requerido: No

Nombre del campo: Objeto
Tipo de datos: Texto
Tamaño del campo: 15
Requerido: Sí

Nombre del campo: Materia
Tipo de datos: Texto
Tamaño del campo: 20
Requerido: Sí

Tabla Vocabulario

Nombre del campo: Clave_V
Tipo de datos: Autonumérico
(Hacerlo “Clave principal”)

Nombre del campo: Tema
Tipo de datos: Texto
Tamaño del campo: 30
Requerido: Sí

Nombre del campo: Palabra
Tipo de datos: Texto
Tamaño del campo: 30

Nombre del campo: Tipo
Tipo de datos: Texto
Tamaño del campo: 30

Nombre del campo: Significado
Tipo de datos: Texto
Tamaño del campo: 100

Tabla Cuestionario

Nombre del campo: Clave_C
Tipo de datos: Autonumérico
(Hacerlo “Clave principal”)

Nombre del campo: Pregunta
Tipo de datos: Texto
Tamaño del campo: 50

Nombre del campo: Subtema
Tipo de datos: Texto
Tamaño del campo: 30
Requerido: Sí

Nombre del campo: R1
Tipo de datos: Texto
Tamaño del campo: 30

Nombre del campo: R2
Tipo de datos: Texto
Tamaño del campo: 30

Nombre del campo: R3
Tipo de datos: Texto
Tamaño del campo: 30

Nombre del campo: Rc
Tipo de datos: Número
Tamaño del campo: Entero largo
Regla de validación: >=1 y <=3

Tabla Imagenes

Nombre del campo: Objeto
Tipo de datos: Texto
Tamaño del campo: 15
(Hacerlo “Clave principal”)

Nombre del campo: Descripcion
Tipo de datos: Texto
Tamaño del campo: 200

Nombre del campo: Imagen
Tipo de datos: Objeto OLE

Nota: todas las otras tablas de las materias, tienen exactamente la misma estructura, motivo por el cual no es necesario repetir sus correspondientes diseños conceptuales.

Capítulo 4. Programación del sistema.

Teóricamente el sistema ya está terminado del todo, lo que resta es la implementación del mismo a través del lenguaje de programación Delphi. Debe hacerse la aclaración que la base de datos fue creada en access 2000/2003 y que se definió de acuerdo a lo tratado en el capítulo anterior, por ello, aquí sólo se tratará la codificación, depuración y pruebas del programa que hará uso de la base de datos mencionada.

4.1. Lenguaje de programación.

4.1.1. Lenguaje Delphi.

Dadas las características que ofrece esta poderosa herramienta, como se describe en los apartados siguientes, se ha planteado desde el Capítulo 1, que el lenguaje de programación a utilizar debería ser orientado a objetos, y los beneficios que Delphi tiene, son razón suficiente para desarrollar la aplicación sobre este paradigma y con este lenguaje.

4.1.1.1. Introducción.

Con el auge que ha tenido en los últimos años el sistema operativo Windows de Microsoft, se ha convertido en el estándar de la mayoría de las computadoras personales en el mundo. Debido a esto, las empresas desarrolladoras de software y, más por necesidad que por convencimiento, han enfocado sus esfuerzos a producir aplicaciones que puedan convivir con el ambiente de este sistema operativo.

Anteriormente las aplicaciones Windows eran desarrolladas en el lenguaje C, el cual tiene un ambiente basado en DOS, esto generaba una gran cantidad de código y requería de un gran esfuerzo de los programadores. Con la llegada de los lenguajes visuales como Visual Basic, Visual C++ y Delphi, entre otros, el desarrollo de aplicaciones para el ambiente Windows se hizo más sencillo y menos arduo.

La programación de los eventos en estos lenguajes se realiza precisamente utilizando la filosofía de las ventanas de que tiene Windows, lo que permite de manera fácil localizar la ubicación del código, esto es lo que los hizo más populares a la hora de seleccionar el lenguaje de desarrollo.
Entre estos lenguajes visuales, el que tiene una forma mas amigable y sencilla de crear las aplicaciones utilizando el paradigma de la programación orientada a objetos, es Delphi, ya que posee una estructura semejante al idioma inglés por lo que se facilita la lectura y escritura del código de programación.

4.1.1.2. Reseña histórica de Delphi.

Delphi tiene todas las características de turbo pascal, el cual durante la época de los 80 fue uno de los lenguajes mas populares en el desarrollo de aplicaciones para las computadoras personales debido a su facilidad de aprendizaje.

A principios de 1995, la empresa Borland International lanzaba Delphi como una opción mas en el desarrollo de aplicaciones visuales basadas en el entorno Windows, que hasta el momento dominaba Visual Basic de Microsoft y al cual no tardó en quitarle la hegemonía.

La característica principal que los desarrolladores advirtieron en Delphi, y que Visual Basic no ofrecía, era que generaba un verdadero código ejecutable ya que se trataba de un compilador y, por lo tanto las aplicaciones funcionaban mas rápido. Además que con su biblioteca de controles cubría todo lo necesario para la programación en Windows 3.1 y crear controles propios para agregarlos a la librería.

El Object Oriented Pascal es el lenguaje que Delphi utiliza para crear las aplicaciones orientadas a objetos que hoy en día es el paradigma mas utilizado. Debido a que Delphi pertenece a la empresa Borland, la potencia de éste puede comparase con el compilador de C++.

Actualmente existen varias versiones fundamentales de Delphi: Delphi 1 para Windows 3.1 y Windows 3.11, y Delphi 2, junto a Delphi 3, para Windows 95-Windows NT. En la aparición de la versión 4, solo se agregó el soporte de la arquitectura Cliente/Servidor.

En cuanto a información sobre técnicas de programación en Delphi, existe un vasto de opciones a elegir, tales como miles de páginas Web, muchos foros de debate, sitios FTP que contienen una enorme cantidad de librerías, News, Chats sobre el tema y mucha mas información que puede ser obtenida a través de Internet, aunado a la multitud de libros de diversos autores que los principiantes y avanzados pueden consultar.

4.1.1.3. Delphi: un lenguaje orientado a eventos.

Un suceso o evento es alguna acción que se presenta cuando ocurre algo en el ambiente que lo haga reaccionar. Por ejemplo, un foco de una habitación se enciende cuando se acciona el interruptor, ya que la corriente fluye hasta llegar a él. Otra reacción del foco se presenta cuando el interruptor cambia de posición cortando el flujo de corriente eléctrica haciendo que el foco se apague. En este caso, el evento es la acción de activar el interruptor y la reacción o respuesta se nota al encenderse o apagarse el foco.

Análogamente, una aplicación construida para ser utilizada en una computadora, puede tener en su entorno eventos cuyas reacciones provoquen la ejecución de procesos que realicen tareas específicas.

Muchas aplicaciones diseñadas para el sistema operativo DOS, son programadas para que respondan a eventos del teclado o incluso del ratón. En ellas, la presión de una tecla de función, tal como F1, F2, etc., invoca a una rutina que hace que se presente una pantalla de ayuda o puede mostrar un catálogo de ítems para elegir uno de ellos en una captura de datos. De la misma forma es posible programar combinaciones de teclas tales como CTRL+A que, cuando sean presionadas se active un procedimiento dentro del sistema.

Para realizar este tipo de programación, es necesario conocer el código numérico de las teclas y mediante la verificación de éste, en el momento que el programador decida, la rutina asociada se ejecuta.

4.1.1.4. Elementos de Delphi.

El lenguaje cuenta con un variado acervo de componentes que permiten modelar la aplicación lo más cercano a la realidad.

4.1.1.4.1. Tipos.

Se tiene en primer lugar los tipos de datos primitivos, propios del lenguaje, con los cuales es posible abstraer algunos elementos de la aplicación, tales como las variables o constantes utilizadas en ella. Así como en los otros lenguajes de programación, los tipos de datos determina el rango de valores que puede tomar una variable y las operaciones posibles que se pueden realizar con ellos.

Delphi, al igual que Pascal, es un lenguaje fuertemente tipificado, lo que significa que una variable, procedimiento o función deberá ser declarado antes de ser utilizado. Podemos decir que esta es una desventaja y no da libertad al programador, como en el caso del lenguaje C, sin embargo esta desventaja se compensa cuando se crea el programa ejecutable, el cual es más veloz que cualquier otro generado con un lenguaje diferente para el ambiente Windows.
Los tipos de datos que Delphi tiene disponible para el programador son:

- Enteros: Integer, ShortInt, LongInt, Byte, Word, SmallInt, LongWord, Int64.
- Reales: Real, Single, Double, Extended, Comp.
- Lógicos: Boolean, ByteBool, WordBol, LongBool.
- Caracteres: Char, AnsiChar, WideChar, String, AnsiString, WideString.
- Moneda: Currency.
- Puntero: Pointer, PChar.
- Cualquier tipo: Variant.

Delphi también permite crear nuevos tipos de datos utilizando los primitivos, gracias a ello es posible que el desarrollador pueda aplicar la POO.

4.1.1.4.2. Estructuras de control.

Las estructuras de control permiten al programador cambiar el flujo del programa, desviándolo hacia otras rutinas o realizando iteraciones sobre él. En Delphi se cuenta con los siguientes:

Estructura If

Permite desviar el flujo con la finalidad de que se ejecuten unas u otras sentencias de acuerdo al resultado de una expresión lógica.

La sintáxis es la siguiente:

```
If <expresión lógica> Then
  Begin
    Sentencias
      ........
  End
Else
  Begin
    Sentencias
      ..........
End;
```

Estructura Case

El uso de esta estructura permite, que de acuerdo al resultado de una expresión ordinal, se realice un determinado bloque de sentencias que se encuentran definidas explícitamente o se haga llamar a un procedimiento donde se encuentre dicho bloque. En el Case se listan los posibles valores legales que puede tomar la expresión para que ejecute alguna de las acciones programadas.

También es posible incluir el código que se ejecutara en caso de que el resultado de la expresión no se encuentre en la lista de valores.

```
Case <expresión ordinal>
  Lista1: Bloque de sentencias;
  Lista2: Bloque de sentencias;
  .........
  ListaN: Bloque de sentencias;
Else
  Bloque de sentencias;
End;
```

Estructura For

Mediante ella es posible realizar repeticiones de trozos de código un número determinado de veces.

```
For Contador:=ValorInicial To ValorFinal Do
  Begin
    Sentencias;
    .............
  End;
```

Estructura While

Tiene la misma función que For, sólo que en vez de un contador utiliza una expresión lógica, que al ser falsa el ciclo finaliza. La expresión lógica es evaluada antes de iniciar el ciclo.

```
While <expresión lógica> Do
  Begin
    Sentencias;
    .............
  End;
```

Estructura Repeat

Establece un ciclo semejante al de While, la diferencia radica en que en Repeat la expresión se evalúa al final del ciclo y termina cuando ésta es verdadera. El uso de esta estructura asegura la ejecución de al menos una vez el bloque de sentencias.

```
Repeat
  Sentencias;
  .............
Until <Expresión lógica>;
```

4.2. Codificación.

4.2.1. Programación.

Como ya se ha mencionado antes, el lenguaje de programación a utilizar es el Delphi, que es un lenguaje orientado a objetos basado en el lenguaje procedural Pascal, integrando la propiedad visual bajo Windows, y por tanto, tiene ya definidos muchos objetos con métodos y varias herramientas de programación, tales como el manejo de bases de datos, que sirve en el desarrollo de este sistema. Las especificaciones que conllevó el diseño se muestran a continuación.

4.2.1.1. Miniespecificaciones del sistema.

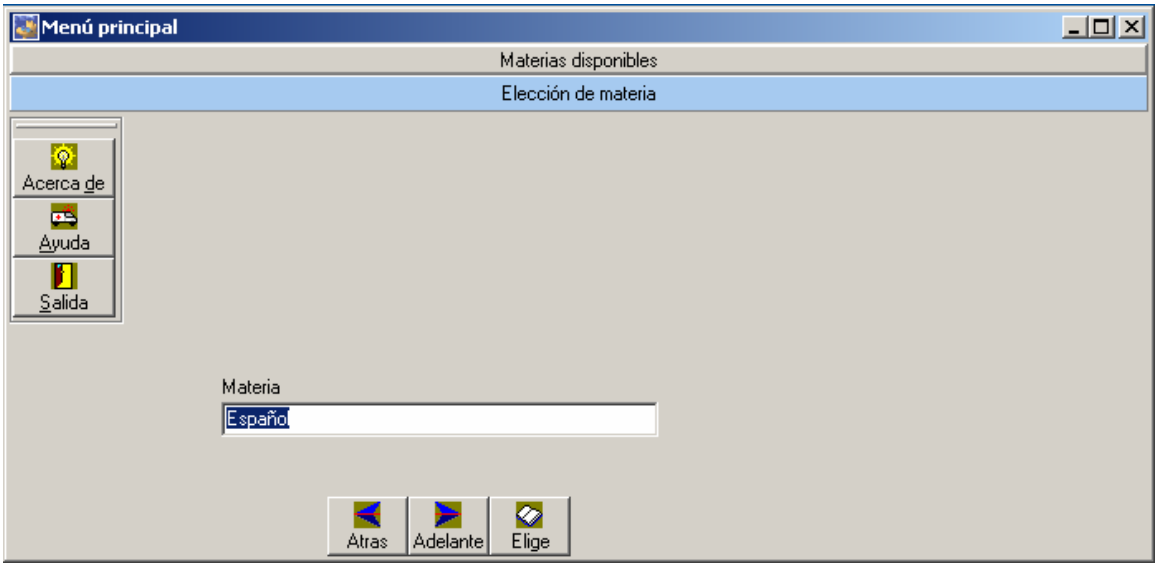
A continuación se muestran los nombres de los módulos (8) que componen el sistema, y que son por la especificación de requisitos (sección 2.1.2) y enumeración de módulos o procesos (sección 3.1.1):

1. Elección de materia.
2. Desplegado principal.
3. Búsqueda principal.
4. Desplegado de objetos multimedia.
5. Cuestionario o evaluación.
6. Diccionario o vocabulario.
7. Ayuda.
8. Impresión.

4.2.2. Módulos.

Por cada uno de los módulos, excepto “Ayuda”, se tiene una sola pantalla y se describen con sus respectivos diccionarios de datos, textos, iconos y/o imágenes y objetos.

4.2.2.1. Módulo de elección de materia.



Descripción: En esta pantalla se elige la materia con la que se trabajará posteriormente.

Nombre de la forma: FormPrincipal.
Texto: “Menú principal”
Icono:
Componentes:

Panel1:
Posición: Parte superior extremo de la ventana.
Texto: “Materias disponibles”.
Panel2:
Posición: Parte superior de la ventana bajo el Panel1.
Texto: “Elección de materia”.
Menú1:
Posición: Lado izquierdo de la pantalla bajo el Panel2.
Número de opciones: 3
Número de botones: 3
“Acerca de”: Muestra una forma con datos del sistema.
Tecla rápida: C
Icono:
“Ayuda”: Despliega la ayuda correspondiente.
Tecla rápida: A
Icono:
“Salida”: Da por terminado el programa.
Tecla rápida: S
Icono:
Texto1:
Posición: Central de la ventana.
Texto: “Materia”.

DBEdit1:

Posición: Central de la ventana, bajo Texto1.

Contenido: El correspondiente al campo “materia” de la tabla “materias”.

Menú2:

Posición: Parte inferior de la forma, al centro.

Número de opciones: 3

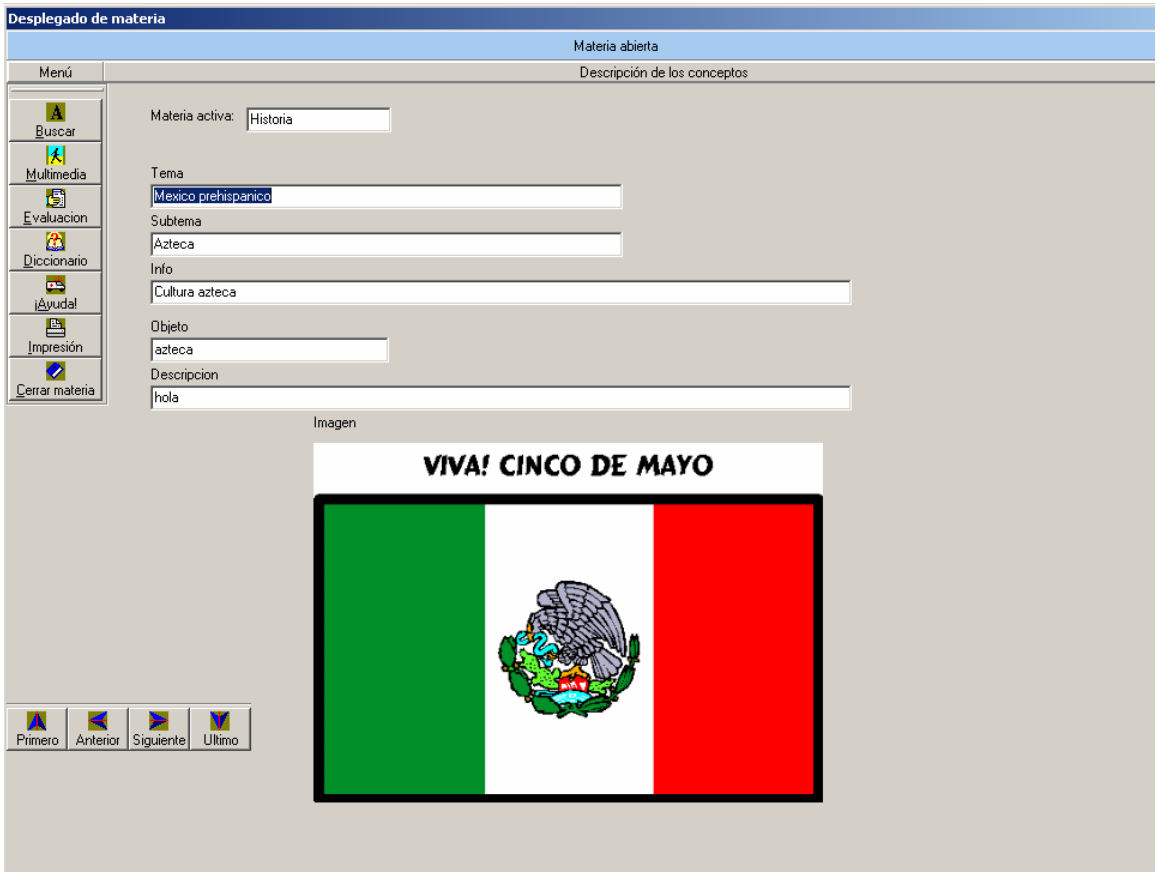
Número de botones: 3

“Atrás”: Busca la materia anterior.
Icono: 

“Adelante”: Busca la materia siguiente.
Icono: 

“Elige”: Abre módulo desplegado con la materia mostrada en DBEdit1.
Icono: 

4.2.2.2. Módulo desplegado.



Descripción: Ahora se muestra de la materia elegida la tabla de información e imágenes, pudiendo hacer llamados a las opciones del menú.

Nombre de la forma: FormDesplegado.

Texto: “Desplegado de materia”.
Icono: 
Componentes:

Panel1:
Posición: Parte superior extremo de la ventana.
Texto: “Materia abierta”.
Panel2:
Posición: Parte superior de la ventana bajo el Panel1.
Componentes:
PanelA:
Posición: Lado izquierdo del Panel2.
Texto: “Menú”.
PanelB:
Posición: Derecha del PanelA.
Texto: “Descripción de los conceptos”.
Menú1:
Posición: Lado izquierdo de la pantalla bajo el Panel2.
Número de opciones: 7
Número de botones: 7
“Buscar”: Abre el módulo buscar.
Tecla rápida: B
Icono: 
“Multimedia”: Abre el módulo multimedia.
Tecla rápida: M
Icono: 
“Evaluación”: Abre el módulo evaluación.
Tecla rápida: E
Icono: 
“Diccionario”: Abre el módulo diccionario.
Tecla rápida: D
Icono: 
“Ayuda”: Despliega la ayuda correspondiente.
Tecla rápida: A
Icono: 
“Impresión”: Abre el módulo impresión.
Tecla rápida: I
Icono: 
“Cerrar materia”: Regresa al módulo elección de materia.
Tecla rápida: C
Icono: 
Texto1:
Posición: Bajo Panel2 y a la derecha del Menú1.
Texto: “Materia activa”.
DBEdit1:
Posición: Derecha de Texto1.

Contenido: Correspondiente a la materia activa de la tabla “materias”.

Texto2:

Posición: Bajo Texto1.

Texto: “Tema”.

DBEdit2:

Posición: Bajo Texto2.

Contenido: Correspondiente al campo “tema” de la tabla activa de la materia actual.

Texto3:

Posición: Bajo DBEdit2.

Texto: “Subtema”.

DBEdit3:

Posición: Bajo Texto3.

Contenido: Correspondiente al campo ”subtema” de la tabla activa de la materia actual.

Texto4:

Posición: Bajo DBEdit3.

Texto: “Información”.

DBEdit4:

Posición: Bajo Texto4.

Contenido: Correspondiente al campo “info” de la tabla activa de la materia actual.

Texto5:

Posición: Bajo DBEdit4.

Texto: “Objeto”.

DBEdit5:

Posición: Bajo Texto5.

Contenido: Correspondiente al campo “objeto” de la tabla activa de la materia actual.

Texto6:

Posición: Bajo DBEdit5.

Texto: “Descripción”.

DBEdit6:

Posición: Bajo Texto6.

Contenido: Correspondiente al campo “descripcion” de la tabla activa de las imágenes de la materia actual.

Texto7:

Posición: Bajo DBEdit 6 hacia la derecha.

Texto: “Imagen”.

Imagen1:

Posición: Bajo Texto7.

Contenido: Correspondiente al campo “imagen” de la tabla activa de las imágenes de la materia actual.

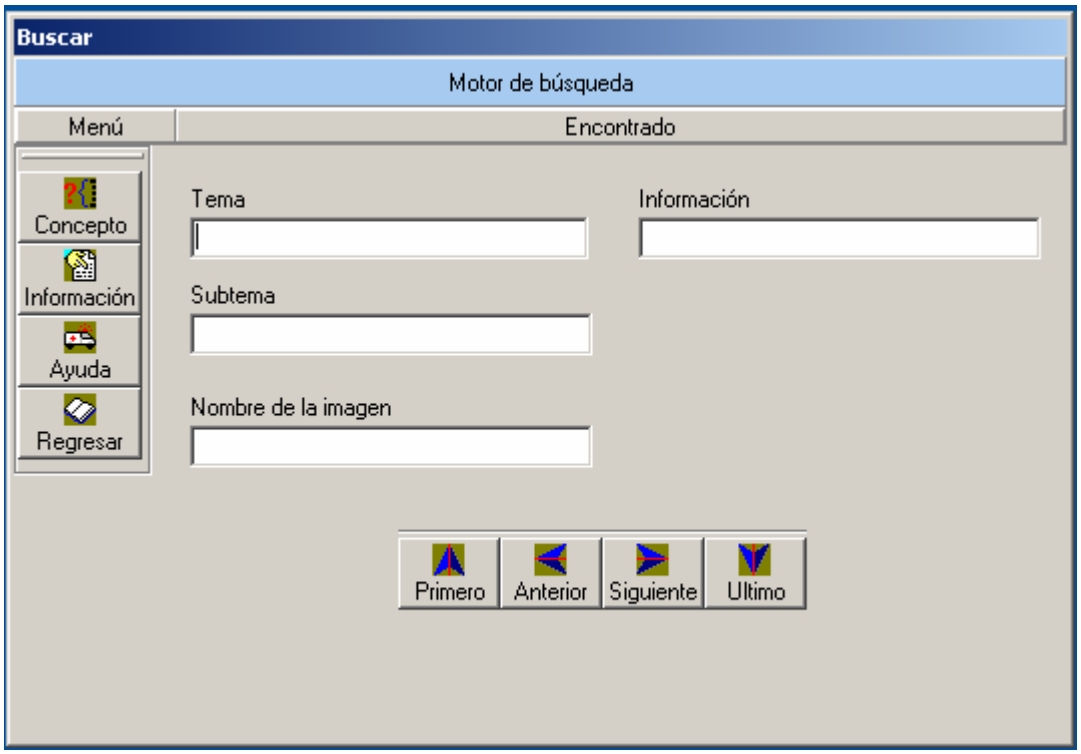
Menú2.

Posición: Lado izquierdo en la parte inferior de la forma.

Número de opciones: 4.
Número de botones: 4.

“Primero”:	Va directamente al primer registro de la tabla de la materia activa y su relación con la materia de imágenes. Icono:
“Anterior”:	Retrocede un registro de la tabla de la materia activa y su relación con la materia de imágenes. Icono:
“Siguiente”:	Avanza un registro de la tabla de la materia activa y su relación con la materia de imágenes. Icono:
“Ultimo”:	Va directamente al último registro de la tabla de la materia activa y su relación con la materia de imágenes. Icono:

4.2.2.3. Módulo buscar.



Descripción: En este módulo se ejecuta la búsqueda sobre los campos “subtema” e “info” sobre la tabla “materias” de la materia elegida desde el módulo principal.

Nombre de la forma: FormBuscar.

Texto: “Buscar”.
Icono: 
Componentes:

- Panel1:
  - Posición: Parte superior extremo de la ventana.
  - Texto: “Motor de búsqueda”.
- Panel2:
  - Posición: Parte superior de la ventana bajo el Panel1.
  - Componentes:
    - PanelA:
      - Posición: Lado izquierdo del Panel2.
      - Texto: “Menú”.
    - PanelB:
      - Posición: Derecha del PanelA.
      - Texto: “Encontrado”.
- Menú1:
  - Posición: Lado izquierdo de la pantalla bajo el Panel2.
  - Número de opciones: 4
  - Número de botones: 4
    - “Concepto”: Busca sobre el campo “subtema” de la materia abierta.
      Tecla rápida: C
      Icono: 
    - “Información: Busca sobre el campo “info” de la materia abierta.
      Tecla rápida: I
      Icono: 
    - “Ayuda”: Despliega la ayuda correspondiente.
      Tecla rápida: A
      Icono: 
    - “Regresar”: Regresa al módulo desplegado de materia.
      Tecla rápida: R
      Icono: 
- Texto1:
  - Posición: Bajo Panel2 y a la derecha del Menú1.
  - Texto: “Tema”.
- DBEdit1:
  - Posición: Bajo Texto1.
  - Contenido: Correspondiente al campo “tema” de la consulta hecha de la tabla “materias”.
- Texto2:
  - Posición: Bajo Texto1.
  - Texto: “Subtema”.
- DBEdit2:

Posición: Bajo Texto2.
Contenido: Correspondiente al campo “subtema” de la consulta hecha de la tabla “materias”.

Texto3:
Posición: Bajo DBEdit2.
Texto: “Nombre de la imagen”.

DBEdit3:
Posición: Bajo Texto3.
Contenido: Correspondiente al campo “objeto” de la consulta hecha de la tabla “materias”.

Texto4:
Posición: Derecha de Texto1.
Texto: “Información”.

DBEdit4:
Posición: Bajo Texto4.
Contenido: Correspondiente al campo “info” de la consulta hecha de la tabla “materias”.

Menú2.
Posición: Centrado en la parte inferior de la forma.
Número de opciones: 4.
Número de botones: 4.

“Primero”: Va directamente al primer registro de la consulta si no es vacía.
Icono: 

“Anterior”: Retrocede un registro de la consulta si no es vacía.
Icono: 

“Siguiente”: Avanza un registro de la consulta si no es vacía.
Icono: 

“Ultimo”: Va directamente al último registro de la consulta si no es vacía.
Icono: 

4.2.2.4. Módulo multimedia.

Descripción: Ahora se muestra de la materia elegida la tabla de información e imágenes, pudiendo hacer llamados a las opciones del menú.

Nombre de la forma: FormMultimedia.
Texto: “Multimedia”.
Icono:
Componentes:

Panel1:
Posición: Parte superior extremo de la ventana.
Texto: “Anexo multimedia”.
Panel2:

Posición: Parte superior de la ventana bajo el Panel1.
Componentes:
PanelA:
Posición: Lado izquierdo del Panel2.
Texto: “Menú”.
PanelB:
Posición: Derecha del PanelA.
Texto: “Imagen”.

Menú1:
Posición: Lado izquierdo de la pantalla bajo el Panel2.
Número de opciones: 2
Número de botones: 2
“Ayuda”: Despliega la ayuda correspondiente.
Tecla rápida: A
Icono: 
“Regresar”: Regresa al módulo desplegado de materia.
Tecla rápida: R
Icono: 

Texto1:
Posición: Bajo Panel2 y a la derecha del Menú1.
Texto: “Objeto”.

DBEdit1:
Posición: Derecha de Texto1.
Contenido: Correspondiente al campo “objeto” de la tabla imagenes de la materia elegida.

Texto2:
Posición: Bajo Texto1.
Texto: “Descripción”.

DBEdit2:
Posición: Bajo Texto2.
Contenido: Correspondiente al campo “descripcion” de la tabla imagenes de la materia elegida.

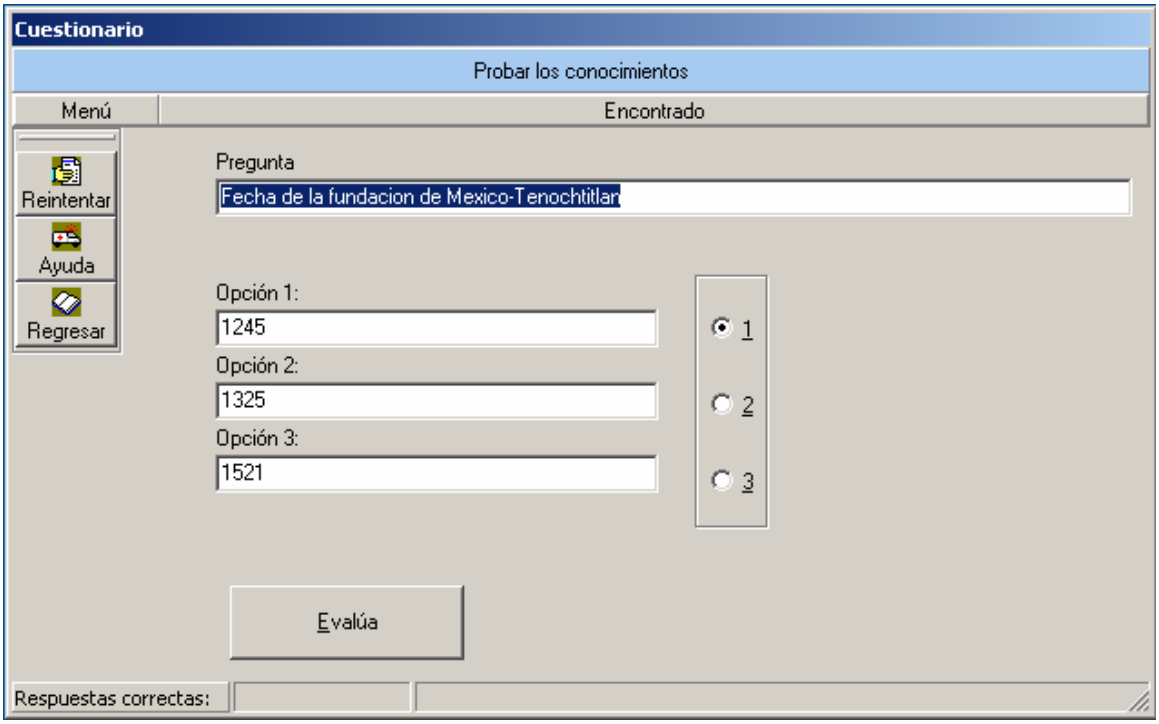
Texto3:
Posición: Centrado bajo DBEdit2.
Texto: “Imagen”.

Imagen1:
Posición: Bajo Texto3.
Contenido: Correspondiente al campo “imagen” de la tabla imagenes de la materia elegida.

Menú2.
Posición: Lado izquierdo en la parte inferior de la forma.
Número de opciones: 4.
Número de botones: 4.

“Primero”:	Va directamente al primer registro de la tabla imagenes de la materia activa. Icono:
“Anterior”:	Retrocede un registro de la tabla imagenes de la materia activa. Icono:
“Siguiente”:	Avanza un registro de la tabla de la imagenes de la materia activa. Icono:
“Ultimo”:	Va directamente al último registro de la tabla imagenes de la materia activa. Icono:

4.2.2.5. Módulo evaluación.



Descripción: Para el módulo de evaluación se tiene la forma “cuestionario” en la que se da una pregunta y tres posibles respuestas, y después de 10 preguntas se da la correspondiente calificación.

Nombre de la forma: FormCuestionario.
Texto: “Cuestionario”.
Icono:
Componentes:

Panel1:

Posición: Parte superior extremo de la ventana.
Texto: “Probar los conocimientos”.

Panel2:
Posición: Parte superior de la ventana bajo el Panel1.
Componentes:
PanelA:
Posición: Lado izquierdo del Panel2.
Texto: “Menú”.
PanelB:
Posición: Derecha del PanelA.
Texto: “Pregunta con respuestas”.

Menú1:
Posición: Lado izquierdo de la pantalla bajo el Panel2.
Número de opciones: 3
Número de botones: 3
“Reintentar”: Reinicia la forma para evaluar de nuevo.
Tecla rápida: I
Icono: 
“Ayuda”: Despliega la ayuda correspondiente.
Tecla rápida: A
Icono: 
“Regresar”: Regresa al módulo desplegado de materia.
Tecla rápida: R
Icono: 

Texto1:
Posición: Bajo Panel2 y a la derecha del Menú1.
Texto: “Pregunta”.

DBEdit1:
Posición: Bajo Texto1.
Contenido: Correspondiente al campo “pregunta” de la tabla “cuestionario” de la materia activa.

Texto2:
Posición: Bajo DBEdit1.
Texto: “Opción 1”.

DBEdit2:
Posición: Bajo Texto2.
Contenido: Correspondiente al campo “r1” de la tabla “cuestionario” de la materia activa.

Texto3:
Posición: Bajo DBEdit2.
Texto: “Opción 2”.

DBEdit3:
Posición: Bajo Texto3.
Contenido: Correspondiente al campo “r2” de la tabla “cuestionario” de la materia activa.

Texto4:
- Posición: Bajo DBEdit3.
- Texto: “Opción 3”.

DBEdit4:
- Posición: Bajo Texto4.
- Contenido: Correspondiente al campo “r3” de la tabla “cuestionario” de la materia activa.

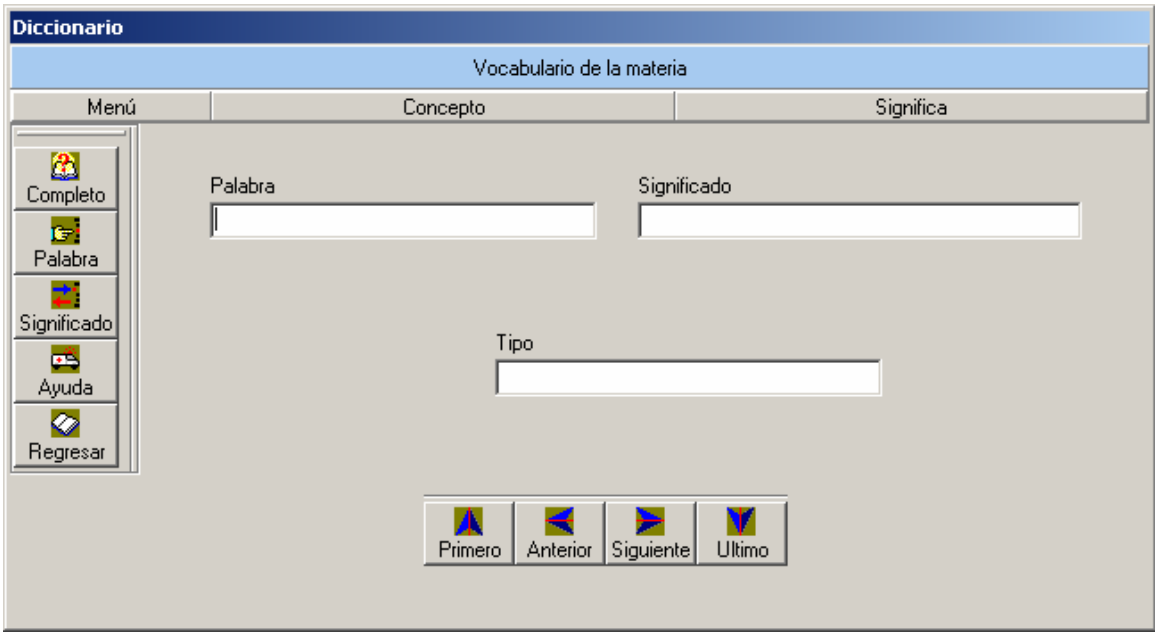
RadioGroup1:
- Posición: Derecha de Texto1, Texto2 y Texto3.
- Elección: Se compara contra el valor que tenga el campo “rc” de la tabla “cuestionario” de la materia activa.

Botón1:
- Posición: Bajo DBEdit4.
- Texto: “Evalúa”.

StatusBar1:
- Posición: Parte inferior de la forma.
- Componentes:
  - PanelA:
    - Posición: Parte izquierda de StatusBar1.
    - Texto: “Respuestas correctas:”
  - PanelB:
    - Posición: Derecha de PanelA.
    - Texto: Número de respuestas correctas + “/10”
  - PanelC:
    - Posición: Derecha de PanelB.
    - Texto “Mal, inténtalo de nuevo” o “Bien, pero puedes mejorar” o “Excelente, aprendiste bien tu lección”

  Nota: PanelB y PanelC sólo serán visibles cuando se hayan evaluado diez preguntas.

4.2.2.6. Módulo diccionario.



Descripción: Esta forma presenta el vocabulario correspondiente a la materia activa. Tiene dos opciones de búsqueda (sobre la palabra y sobre el significado) o bien, puede ser mostrado todo el contenido del diccionario.

Nombre de la forma: FormDiccionario.
Texto: “Diccionario”.
Icono:
Componentes:

Panel1:
Posición: Parte superior extremo de la ventana.
Texto: “Vocabulario de la materia”.
Panel2:
Posición: Parte superior de la ventana bajo el Panel1.
Componentes:
PanelA:
Posición: Lado izquierdo del Panel2.
Texto: “Menú”.
PanelB:
Posición: Derecha del PanelA.
Texto: “Concepto”.
PanelC:
Posición: Derecha del PanelC.
Texto: “Significa”.
Menú1:
Posición: Lado izquierdo de la pantalla bajo el Panel2.

Número de opciones: 5
Número de botones: 5

"Completo": Abre todo el vocabulario de la materia.
Tecla rápida: C
Icono: 

"Palabra": Busca sobre el campo "palabra" de la tabla "diccionario" de la materia activa.
Tecla rápida: P
Icono: 

"Significado": Busca sobre el campo "significado" de la tabla "diccionario" de la materia activa.
Tecla rápida: S
Icono: 

"Ayuda": Despliega la ayuda correspondiente.
Tecla rápida: A
Icono: 

"Regresar": Regresa al módulo desplegado de materia.
Tecla rápida: R
Icono: 

Texto1:
- Posición: Bajo Panel2 y a la derecha del Menú1.
- Texto: "Palabra".

DBEdit1:
- Posición: Bajo Texto1.
- Contenido: Correspondiente al campo "palabra" de la tabla "diccionario" de la materia activa.

Texto2:
- Posición: Bajo Panel2 y a la derecha de Texto1.
- Texto: "Significado".

DBEdit2:
- Posición: Bajo Texto2.
- Contenido: Correspondiente al campo "significado" de la tabla "diccionario" de la materia activa.

Texto3:
- Posición: Bajo DBEdit1 y DBEdit2 al centro.
- Texto: "Tipo".

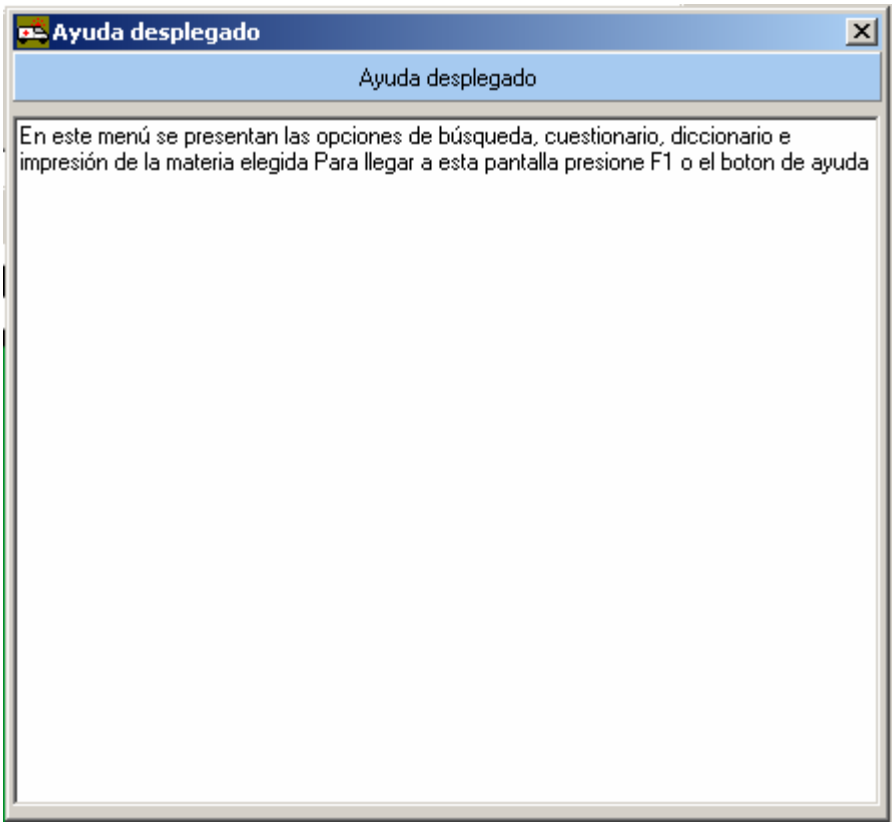
DBEdit3:
- Posición: Bajo Texto3.
- Contenido: Correspondiente al campo "tipo" de la tabla "diccionario" de la materia activa..

Menú2.
- Posición: Lado izquierdo en la parte inferior de la forma.
- Número de opciones: 4.
- Número de botones: 4.

"Primero": Va directamente al primer registro de la

	consulta si no es vacía. Icono:
“Anterior”:	Retrocede un registro de la consulta si no es vacía. Icono:
“Siguiente”:	Avanza un registro de la consulta si no es vacía. Icono:
“Ultimo”:	Va directamente al último registro de la consulta si no es vacía. Icono:

4.2.2.7. Módulo ayuda.



Descripción: Cada módulo tiene su propia forma de ayuda, a la cual se accede con la tecla <F1> o con la opción “Ayuda” de cualquier módulo.

Nombre de la forma: Puede ser cualquiera de los siguientes:

“FormAyudaPrincipal”
“FormAyudaDesplegado”
“FormAyudaBuscar”
“FormAyudaImagenes”
“FormAyudaCuestionario”

"FormAyudaDiccionario"
"FormAyudaImprime"

Texto: Puede ser cualquiera de los siguientes:
"Ayuda principal"
"Ayuda desplegado"
"Ayuda buscar"
"Ayuda imágenes"
"Ayuda cuestionario"
"Ayuda diccionario"
"Ayuda imprime"

Icono:  (para todas las formas)

Componentes:

Panel1:

Posición: Parte superior extremo de la ventana.

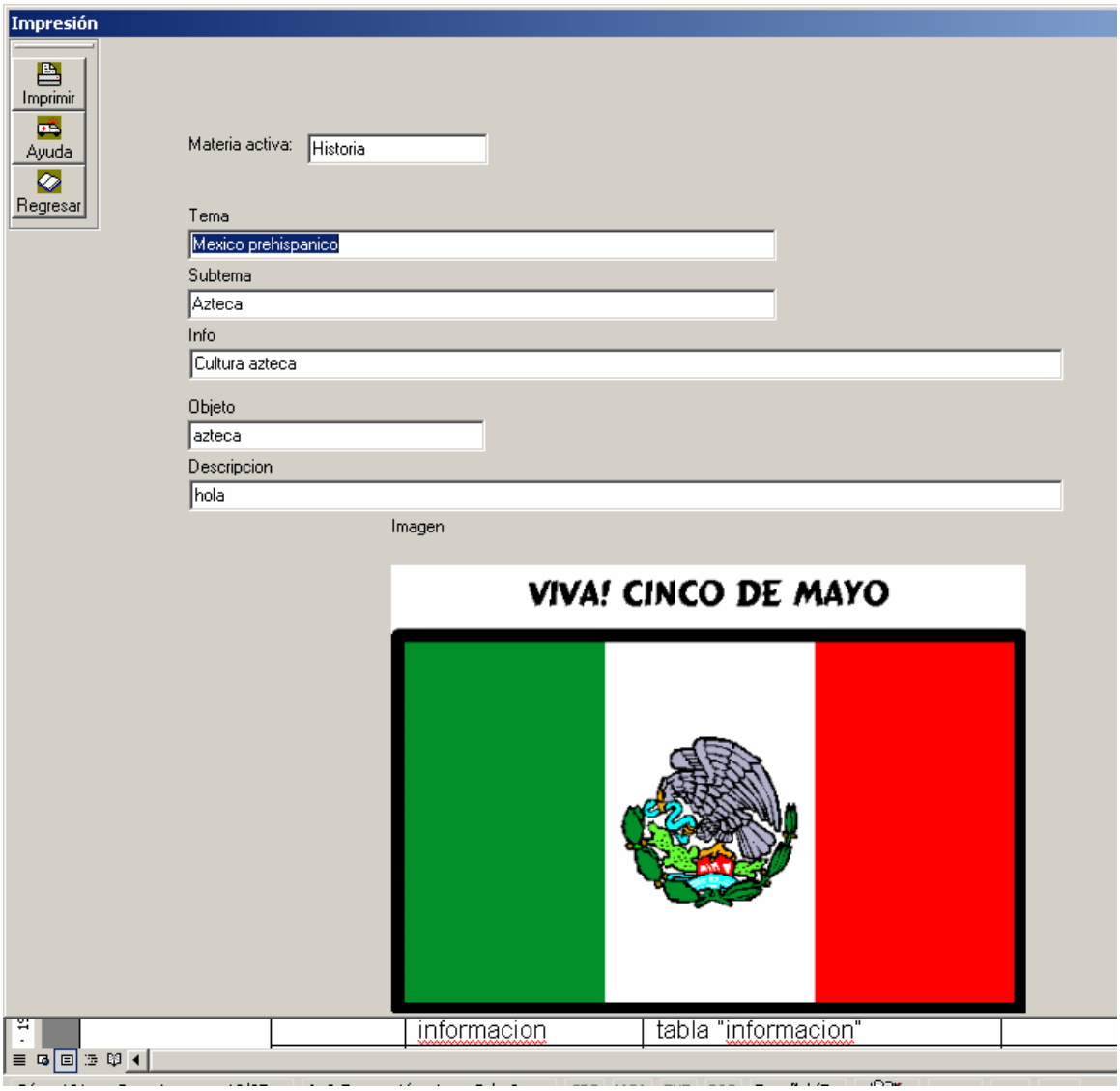
Texto: Puede ser cualquiera de los siguientes:
"Ayuda principal"
"Ayuda desplegado"
"Ayuda buscar"
"Ayuda imágenes"
"Ayuda cuestionario"
"Ayuda diccionario"
"Ayuda imprime"

Texto1:

Posición: Central de la ventana.

Texto: Depende de la ayuda correspondiente al módulo desde el cual fue llamado el módulo de ayuda.

4.2.2.8. Módulo impresión.



Descripción: Cuando se quiera mandar a imprimir el dato que se muestra en el módulo desplegado, se llama al módulo de impresión para que el mismo sea imprimido.

Nombre de la forma: FormImprime.
Texto: “Impresión”.
Icono:
Componentes:

Menú1:

Posición: Lado izquierdo de la pantalla parte superior.
Número de opciones: 3
Número de botones: 3

“Impresión”: Abre el diálogo de impresión.

Tecla rápida: I
Icono: 

“Ayuda”: Despliega la ayuda correspondiente.
Tecla rápida: A
Icono: 

“Regresar”: Regresa al módulo desplegado de materia.
Tecla rápida: R
Icono: 

Texto1:
- Posición: Parte superior y a la derecha del Menú1.
- Texto: “Materia activa”.

DBEdit1:
- Posición: Derecha de Texto1.
- Contenido: Correspondiente a la materia activa de la tabla “materias”.

Texto2:
- Posición: Bajo Texto1.
- Texto: “Tema”.

DBEdit2:
- Posición: Bajo Texto2.
- Contenido: Correspondiente al campo “tema” de la tabla activa de la materia actual.

Texto3:
- Posición: Bajo DBEdit2.
- Texto: “Subtema”.

DBEdit3:
- Posición: Bajo Texto3.
- Contenido: Correspondiente al campo ”subtema” de la tabla activa de la materia actual.

Texto4:
- Posición: Bajo DBEdit3.
- Texto: “Información”.

DBEdit4:
- Posición: Bajo Texto4.
- Contenido: Correspondiente al campo “info” de la tabla activa de la materia actual.

Texto5:
- Posición: Bajo DBEdit4.
- Texto: “Objeto”.

DBEdit5:
- Posición: Bajo Texto5.
- Contenido: Correspondiente al campo “objeto” de la tabla activa de la materia actual.

Texto6:

Posición: Bajo DBEdit5.
Texto: “Descripción”.

DBEdit6:
Posición: Bajo Texto6.
Contenido: Correspondiente al campo “descripcion” de la tabla activa de las imágenes de la materia actual.

Texto7:
Posición: Bajo DBEdit 6 hacia la derecha.
Texto: “Imagen”.

Imagen1:
Posición: Bajo Texto7.
Contenido: Correspondiente al campo “imagen” de la tabla activa de las imágenes de la materia actual.

4.3. Pruebas.

4.3.1. Pruebas de caja negra.

Una vez hecha la integración de módulos con sus respectivos objetos, se efectúa la serie de pruebas. En este caso, dado que el sistema ha sido programado en un lenguaje visual, se han destinado en gran parte las pruebas de caja negra. Esto es, probar el sistema sin que importen las estructuras de control del programa.

Durante este tipo de pruebas no se encontraron errores, dado que todos los módulos acceden a los otros sin problemas y no existieron “callejones sin salida” o sin regreso.

Dado lo anterior, ahora se procederá a reportar las pruebas de caja blanca sobre los módulos.

4.3.1.1. Pruebas de la estructura lógica del código.

En este tipo de pruebas se encontraron errores de los siguientes tipos:

Los eventos que efectuaban alguna acción sobre la base de datos no siempre eran los correctos, y en algunas veces las tablas quedaban abiertas.
Algunos métodos y propiedades de los objetos no siempre eran las mejores y más adecuadas para el desempeño del sistema.

4.3.1.2. Pruebas de ciclos.

Los únicos ciclos que se implementaron fueron en el módulo del cuestionario. Un ciclo para generar números aleatorios y otro para contabilizar hasta 10 el número de preguntas. Ambos ciclos funcionaron de acuerdo a lo deseado.

4.3.1.3. Pruebas de condiciones.

Las condiciones programadas en varios de los módulos como son las de tipo “no encontrado”, rupturas de ciclos y comparaciones se han probado y no han presentado fallas, excepto las comparaciones, las cuales en un principio tuvieron algún error posteriormente corregido y han funcionado de acuerdo a lo deseado.

4.3.1.4. Pruebas de vinculación de datos.

La vinculación de datos se da desde la asignación de las tablas a usarse en el módulo principal hacia el módulo desplegado y posteriormente entre el módulo desplegado y los módulos multimedia, cuestionario e impresión. Los módulos de

búsqueda y diccionario hacen consultas sobre las tablas que correspondan a la elección de la materia.

Se presentaron errores cuando no se hacía la relación entre el campo “objeto” de la tabla correspondiente a la materia con el campo del mismo nombre de la tabla de “imagenes” lo que se solucionó al hacer ésta con una línea de código cuando la forma fuera abierta.

4.4. Depuración.

4.4.1. Detección y corrección de errores.

Los prototipos del sistema en cuestión han provocado errores como en cualquier sistema grande o pequeño. Sin embargo fueron resueltos cuando fueron localizados. A continuación se describen los tipos de éstos surgidos durante la programación.

4.4.1.1 Detección y corrección de errores semánticos.

Se encontraron algunos errores en comparaciones como OR por AND, lo que ocasiona el mal funcionamiento del sistema. Este tipo fueron de los más comunes. Además algunos de los métodos de los objetos a veces eran extraños a lo deseado, y como ya se ha mencionado en la sección anterior, también eran inadecuadas ciertas instrucciones en eventos dados, por ejemplo el evento “create” (crear) no es lo mismo que el evento “show” (mostrar). Todos estos errores se han corregido en versiones sucesivas del prototipo.

4.4.1.2 Detección y corrección de errores sintácticos.

Los errores de tipo sintáctico simplemente no pudieron haber dejado avanzar el desarrollo del sistema, dado que el programa nunca pudo haber corrido con ellos, fueron corregidos en el instante antes de la ejecución.

Los más comunes errores de este tipo fueron la falta de puntos y comas (“semicolon”) y algunos paréntesis en ciertas instrucciones (“operator”).

4.5. Implementación.

4.5.1. Integración del sistema.

Ya terminada la fase de pruebas y depuración del sistema debe ahora hacerse la versión de distribución, libre (en lo posible) de errores, empaquetada e instalable con su respectiva base de datos.

4.5.1.1 Configuración de equipo.

Para esta sección no es necesario que se tenga una configuración ajena a la normal, es decir el sistema funcionará sobre cualquier plataforma Windows de 32 bits que tenga pantalla SVGA (con resolución de 1024x768 pixeles) y teclado y ratón instalados.

4.5.1.2 Instalación del software.

En este apartado sólo se mencionará que es necesaria la lectura del anexo B para la correcta instalación del controlador de la base de datos y el programa cliente.

4.6. Mantenimiento y versiones sucesivas.

4.6.1. Anticipación al cambio.

Este sistema originalmente fue concebido para operar una sola materia, pudiendo posteriormente ser aumentado a cinco o las que sean requeridas cuando sea necesario, siempre y cuando también sean agregadas dichas materias a la base de datos.

4.6.1.1 Detección de módulos susceptibles a actualizar.

Dado que puedan ser aumentadas al sistema otras materias, el primer módulo en actualizarse sería el principal, posteriormente búsqueda y diccionario, para que pudieran ser abiertas las tablas correspondientes a las nuevas materias agregadas. Pudiendo dejar los demás módulos intactos porque la lógica del programa en sí no lo requeriría.

La programación de los módulos ya mencionados es relativamente sencilla, sólo habría que agregar los casos que fueran necesarios.

4.6.2. Futuras versiones.

Como se mencionó en el apartado anterior, las futuras versiones incluirían el manejo de más materias, pero también es posible la programación de más opciones de búsqueda e incluso el manejo directo de objetos como películas o audios además de las imágenes, y poder hacer un sistema más completo aún.

4.6.2.1 Posible consulta por Internet.

También, considerando el manejo de bases de datos a través de Web, puede ser implementado un sistema que efectúe las mismas funciones que éste pero en manera remota a través de páginas de Internet en una red de área local o incluso desde cualquier lugar autorizado en el mundo.

Conclusiones y tesis.

Conclusiones.

Una vez habiendo finalizado todo el diseño, desarrollo e implementación requerida para el término del sistema de este proyecto, se pueden enunciar las siguientes conclusiones:

- Las bases de datos la mayor parte del tiempo han sido considerados como herramientas administrativas. En este caso, pueden ser aplicadas a un fin educativo.
- El lenguaje estructurado de consulta (SQL) sigue siendo la mejor manera de efectuar consultas en bases de datos relacionales.
- Dado lo anterior, es posible implementar con SQL motores de búsqueda sobre información que contengan estas bases de datos.
- Es recomendable hacer uso de lenguajes visuales para hacer aplicaciones gráficas.
- El paradigma de programación orientada a objetos es muy útil para el desarrollo de aplicaciones de diversas índoles.
- Tomando en cuenta el pasado enunciado, en el caso particular del presente sistema, el manejo de bases de datos.
- En específico, el Delphi es una herramienta muy poderosa como ya se ha mostrado, por ser visual, orientado a objetos y en el manejo de bases de datos.

Tesis.

Los objetivos planteados en el principio del presente documento han sido alcanzados, y, por tanto, se presenta esta tesis:

Este sistema cumple con las diversas metodologías que conllevó su diseño, cumpliendo los objetivos y siendo una excelente herramienta propuesta auxiliar al sistema educativo mexicano en el nivel básico.

Bibliografía y fuentes.

Fuentes bibliográficas.

1. Boehm, B. “Software Engineering Education: Some Industry Needs”, en software Engineering Education: Needs and Objectives, editado por P. Freeman and A. Wasserman, Springer-Verlag, Berlin, 1976.
2. IEEE Standard Glosarry of Software engineering Terminology, IEEE Standard 729-1983.
3. Lehman, M. M., “Programs, Life cycles and the laws of software evolution”, Proc. IEEE, 68 (9), 1060-76, 1980.
4. Fairley, Richard, Ingeniería de software”, McGraw Hill, México 1988.
5. Somerville, Ian, “Ingeniería de software”, Addison Wesley Iberoamericana, México, 1988.
6. Burch, John G., Grundnitski Garyj, “Diseño de sistemas de información”, Grupo Noriega Editores, México 1992.
7. Meyer, Bertrand, “Object Oriented Software Construction”, Prentice Hall International, Series in computer science, UK, 1988.
8. Hopcroft, John E., Ullman, Jeffrey D., “Introducción a la teoría de autómatas, lenguajes y computación”, CECSA, México, 1993.
9. Vázquez López Sergio, Chanez Chávez Ma. Alejandra, “hacedor, sistema asistido por computadora para generar automáticamente el código de estructuras que integran programas”, tesis profesional, Colegio de Computación, BUAP, Puebla, 1994.
10. Date, C. J. “An introduction to database system”, Vol. 1 5th edition, Addison-Wesley Publishing Co. 1986.
11. Codd, E. F. “The relational Model for database management”, Addison-Wesley, 1990.
12. Batini, Ceri, Navathe, “diseño conceptual de bases de datos”, Addison-Wesley, 1994.
13. Adoración de Miguel y Piattine, “Concepción y diseño de bases de datos”. Addison-Wesley, 1993.
14. Korth and Silberschatz, “Fundamentos de bases de datos” 2nda edición. MacGRaw-Hill.

Fuentes de Internet.

Generales.

15. http://www.dcc.uchile.cl/~luguerre/cc51h/clase2.html
16. http://www.fceia.unr.edu.ar/asist/unidad13-4.pdf
17. http://www.monografias.com/trabajos/anaydisesis/anaydisesis.shtml
18. http://fuller.ing.puc.cl/docencia/Taller2004a/temas/tema03/Cap04%20v1.30%2020030213.pdf
19. http://www.inf.udec.cl/revista/ediciones/edicion1/lmonsalve.PDF
20. http://rinconprog.metropoliglobal.com/CursosProg/Entornos/Delphi/BD_Delphi/index.php?cap=1

21. http://www.tutorialbox.com/tutors/delphi/Tut.htm
22. http://cs.uns.edu.ar/~mfalappa/research/f-cacic2004.pdf
23. http://usuarios.lycos.es/motoresdebusqueda/home.htm

Modelo Entidad-Relación.
24. http://www3.uji.es/~mmarques/f47/apun/node83.html
25. http://isym.bwl.uni-mainz.de/publikationen/wi-ab53/wi-ab53.htm
26. http://www.utexas.edu/its/windows/database/datamodeling/dm/overview.html

Diagramas de Flujo de Datos.
27. http://html.rincondelvago.com/analisis-de-sistemas.html
28. http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/proyectoinformatico/libro/c4/c4.htm
29. http://members.fortunecity.es/2y2/Cuando.los.analitas.comienzan.a.trabajar.sobre.un.proyecto.de.suistemas.de.informacion.a.menudo.tienen.que.profundizar.en.un.area.vilcar.htm

Structured Query Language.
30. http://www.lania.mx/biblioteca/seminarios/basedatos/sql.html
31. http://www.cs.us.es/cursos/bd-2003/SQL/sql.htm
32. http://www.accefyn.org.co/bioinfo/sql.htm

Algebra relacional.
33. http://www.programacion.com/bbdd/tutorial/modrel/
34. http://es.tldp.org/Postgresql-es/web/navegable/tutorial/x574.html

OLE en Delphi.
35. http://www.efg2.com/Lab/Library/Delphi/ADO/Northwind/index.html

Historia de Delphi.
36. http://www.itlp.edu.mx/posgrado/lengprog/delphi.htm

Programación en Delphi.
37. http://www.manualesgratis.com/manuales/redir.asp?id=631
38. http://www.manualesgratis.com/manuales/redir.asp?id=623
39. http://www.solotutoriales.com/visitar.asp?id=2140
40. http://www.hackerdude.com/courses/spanish/delphi/indice.html
41. http://www.arrakis.es/~ppriego/delphi/delphi.htm
42. http://personal.redestb.es/revueltaroche/ccind.htm

Modelo funcional.
43. www.ilustrados.com/publicaciones/EpVAlypFkprHwkZnwE.php

Indice temático.