Área de la Energía, las Industrias y los Recursos Naturales No Renovables

CARRERA DE INGENIERÍA EN SISTEMAS

"Herramienta de Diseño para la Elaboración de Modelos de Datos del Diseño Conceptual"

Tesis previa a la Obtención del Título de Ingeniera en Sistemas.

Autora:

- Rocío Elizabeth Tene Plaza

Director:

- Ing. Alex Vinicio Padilla Encalada, Mg. Sc.

LOJA-ECUADOR

2015

CERTIFICACIÓN DEL DIRECTOR

Ing. Alex Vinicio Padilla Encalada, Mg. Sc.

DOCENTE DE LA CARRERA DE INGENIERIA EN SISTEMAS, DEL ÀREA DE LA ENERGÍA, LAS INDUSTRIAS Y LOS RECURSOS NATURALES NO RENOVABLES DE LA UNIVERSIDAD NACIONAL DE LOJA.

Certifica:

Que la egresada Rocío Elizabeth Tene Plaza responsable del presente trabajo de titulación denominado, "Herramienta de Diseño para la Elaboración de Modelos de Datos del Diseño Conceptual", ha sido dirigido, asesorado, supervisado y realizado bajo mi dirección durante su desarrollo, ajustándose a los requerimientos establecidos por la Universidad Nacional de Loja por lo que se autoriza su presentación.

A U T O R Í A

Yo, **R O C I O E L I Z A B E T H T E N E P L A Z A**, declaro ser autor del presente trabajo de tesis y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos o acciones legales por el contenido de la misma.

Adicionalmente acepto y autorizo a la Universidad Nacional de Loja, la publicación de mi tesis en el Repositorio Institucional – Biblioteca Virtual.

Firma: 

Cédula: 1104528599

Fecha: 06 de Agosto 2015

CARTA DE AUTORIZACIÓN DE TESIS POR PARTE DE LA AUTORA, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO

Yo, **ROCIO ELIZABETH TENE PLAZA,** declaro ser autor de tesis titulada: **"HERRAMIENTA DE DISEÑO PARA LA ELABORACIÓN DE MODELOS DE DATOS DEL DISEÑO CONCEPTUAL",** como requisito para optar al grado de **INGENIERA EN SISTEMAS;** autorizo al Sistema Bibliotecario de la Universidad Nacional de Loja para que con fines académicos, muestre al mundo la producción intelectual de la Universidad, a través de la visibilidad de su contenido de la siguiente manera en el Repositorio Digital Institucional.

Los usuarios pueden consultar el contenido de este trabajo en el RDI, en las redes de información del país y del exterior, con las cuales tenga convenio la Universidad.

La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia de la tesis que realice un tercero.

Para constancia de esta autorización, en la ciudad de Loja a los seis días del mes de Agosto del dos mil quince.

Firma: 

Autor: Rocío Elizabeth Tene Plaza

Cédula: 1104528599

Fecha: 06 Agosto 2015

Dirección: Loja, (El Pedestal)

Correo Electrónico: eliz.plaza@ yahoo.es

Teléfono: 2560917 **Celular:** 0999547081

DATOS COMPLEMENTARIOS

Director de Tesis: Ing. Alex Vinicio Padilla, Mg. Sc.

Tribunal de Grado: Ing. Walter Rodrigo Tene Ríos, Mg. Sc.

Ing. Mario Andrés Palma Jaramillo, Mg. Sc.

Ing. Jorge Iván Tocto, Mg. Sc

D E D IC A T O R IA

D e d ic o e l p r e s e n t e t r a b a j o d e t it u la c ió n p r in c ip a lm e n t e a m i fa m ilia p o r s e r e l p ila r fu n d a m e n t a l y d e m o s t r a r m e s u a p o y o in c o n d ic io n a l, q u ié n s u p o g u ia r m e p o r e l b u e n c a m in o , d a r m e fu e r z a s p a r a s e g u ir a d e la n t e y n o d e s m a y a r e n lo s p r o b le m a s q u e s e p r e s e n t a b a n , e n s e ñ á n d o m e a e n c a r a r la s a d v e r s id a d e s n i d e s fa lle c e r e n e l in t e n t o , fo r m á n d o m e c o m o p e r s o n a p a r a c o n s e g u ir m is o b je t iv o s .

R o c ío E liz a b e t h T e n e P la z a

A G R A D E C IM IE N T O

M i m ás sincero gesto de gratitud a las personas que hicieron posible la realización del presente trabajo de titulación y en particular a mi fam ilia por su apoyo incondicional durante el desarrollo del mismo.

A Dios por darme la fortaleza al iniciar, desarrollar y cumplir este meta en mi formación profesional.

Especial agradecimiento a mi Director de Trabajo de Titulación Ing. Alex Padilla por su asesoría y constancia.

A la Universidad Nacional de Loja, porque en sus aulas, recibimos el conocimiento intelectual y humano de cada uno de los docentes de la carrera de Ingeniería en Sistemas.

Rocío Elizabeth Tene Plaza

a. Titulo

"Herramienta de Diseño para la Elaboración de Modelos de Datos del Diseño Conceptual".

b. Resumen

El presente trabajo de titulación referida al desarrollo de una herramienta para la elaboración del diagrama entidad-relación, busca mediante la aplicación de los conceptos, proporcionar una metodología que facilite al usuario encontrar la forma de modelar un caso del mundo real a través del proceso de diseño conceptual de base de datos y que así se lo lleve de forma estructurada e integrado en un ambiente web que modele una problemática.

Así mismo para la construcción de los artefactos que intervienen en el diagrama entidad-relación, se optó por emplear los estándares definidos por W3C aplicados a sus tecnologías HTML, PHP, JavaScript y XML, recursos muy utilizados a la hora de crear estos elementos en un entorno web.

De igual manera como parte del proceso de desarrollo de la aplicación se empleó la metodología de desarrollo ágil RAD, debido a que se requirió de una iteración constante y por las actividades que se llevaron a cabo para su implementación.

En los resultados se dan a conocer los procesos utilizados para el desarrollo de la aplicación en base a los objetivos y donde se realizó interpretaciones específicas de los datos mismos que guiaron el curso del trabajo de titulación.

Finalmente con la implementación se pudo establecer la integración de los datos a través del diagrama entidad-relación generando el diccionario de datos, todo ello con el empleo de herramientas de software libre durante su desarrollo, lo cual lo hace clasificable para mejoras futuras que optimicen su operatividad integrando a la vez nuevos requerimientos.

Summary

The following research work of qualification referring to the development of a tool for the development of entity-relationship diagram, looking through the application of concepts, provide a methodology that facilitates the user to find a way to model a real-world case through the process conceptual design of database and thus take him in a structured and integrated into a web environment that models a problem.

Also for the construction of the artifacts involved in the entity-relationship diagram, we chose to use the standards defined by W3C applied to their HTML technologies, PHP, JavaScript and XML, resources widely used when creating these elements in a web environment.

Similarly as part of the development process of implementing the RAD agile development methodology was used, because it required a constant iteration and the activities carried out for implementation.

The results are disclosed processes used for application development based on the objectives and where specific interpretations of the same data that guided the course of work of titration were performed.

Finally the implementation it was established the integration of data through the entity-relationship diagram generating the data dictionary, all with the use of free software tools for its development, making it classifiable for future improvements that optimize its operability while integrating new requirements.

Certificado por el Dr. Freddy Castillo Hoyos (Ver Anexo 5. Certificación de Traducción)

Índice de Contenidos

Índice de Figuras

Índice de Tablas

c. Introducción

El propósito de estudio del presente trabajo de titulación se centra en la creciente aceptación de las bases de datos y principalmente en la etapa de diseño conceptual, ya que es la etapa inicial de este proceso constituyendo así la base para el resto de etapas, generalmente es representada por modelos de datos y particularmente a través del Modelo Entidad Relación cuya implementación gráfica se la lleva a cabo mediante el diagrama entidad-relación, ya que el diseño de bases de datos ha pasado a constituir parte de la formación general de los informáticos, buscando mediante la aplicación de los conceptos del diseño conceptual de base de datos encontrar la forma de facilitar que el proceso de diseño se lleve de forma estructurada y que se lo considere como un proceso estable.

Es así que se pretende proveer al estudiante de una metodología para el diseño conceptual de base de datos que facilite su comprensión, teniendo como base las técnicas de diseño que modele un caso del mundo real, además de proponer un diseño consistente a través de una herramienta web como parte del proceso tecnológico que implica su implementación.

Por lo anterior esta investigación centra su atención y como objetivo principal en el diseñar e implementar una herramienta de diseño web para la elaboración de modelos de datos del diseño conceptual en el que se represente la estructura global lógica de la base de datos así como el comportamiento relacional del denominado diagrama entidad-relación y que a la vez genere el diccionario de datos que almacena la información donde contiene las características lógicas de los datos que se van a utilizar en el sistema.

De esta manera la estructura general del informe inicia con la introducción, en ella se pretende contextualizar en general el desarrollo del tema tratando de hacer que el lector tenga una idea sobre el contenido, seguido de ello tenemos la revisión de la literatura donde se presenta los planteamientos teóricos de la realidad sobre el tema investigado, además de hacer conocer los materiales, métodos y metodología de desarrollo de software para finalmente presentar los resultados obtenidos, discusión, conclusiones y recomendaciones

d. Revisión de Literatura

1. Introducción a los Sistemas de Bases de Datos

1.1. Sistemas de Base de Datos

Un sistema de gestión de bases de datos SGBD consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a esos datos. La colección de datos, normalmente denominada base de datos, contiene información acerca de una empresa determinada.[**1**]

El término bases de datos corresponde a un conjunto auto descriptivo de registros integrados. Es auto descriptivo, porque además de los datos fuentes del usuario, contiene una descripción de su propia estructura; tal descripción es conocida como diccionario de datos. La Base de Datos es un depósito único de datos para toda la organización, por lo que debe ser capaz de integrar los distintos sistemas y aplicaciones, atendiendo a las necesidades de los usuarios en los niveles: operativo, táctico y estratégico.

En general, un SGBD es un software de Base de Datos que centraliza los datos en un único lugar lógico al que acceden todos los usuarios y aplicaciones, es utilizable por múltiples usuarios y aplicaciones concurrentemente, ofrece visiones parciales del conjunto total de información, según las necesidades de un usuario en particular y posee herramientas para asegurar:

- **Independencia** de datos: a varios niveles, permitiendo la modificación de las definiciones de datos sin afectar a las aplicaciones o esquemas que no utilizan esos datos.

- **Integridad** de los datos: que los datos sean correctos en todo momento, de acuerdo con las especificaciones o reglas impuestas al sistema

- **Seguridad** de los datos: que sólo las personas autorizadas puedan acceder a determinados datos y que sólo puedan efectuar las operaciones para las que han sido autorizados.

Es necesario recordar que una base de datos no puede optimizar la recuperación de los datos para una aplicación en especial, ya que deberá compartirse con numerosos usuarios y con varias aplicaciones. Además, se puede llegar a requerir de cierto software adicional para la SGBD. El enfoque de la base de datos es un concepto que se vuelve cada vez más relevante. El uso de base de datos relaciónales en computadoras personales indica el grado de difusión que este concepto ha alcanzado entre los usuarios. Con este enfoque, los usuarios toman una parte de la base de datos central y cargan sus computadoras personales. Luego estas pequeñas bases de datos se utilizan para emitir reportes o contestar consultas específicas del usuario final.

1.2. Organización de Sistemas de Base de datos

Una base de datos, a diferencia de un archivo, es compartida por muchos usuarios, y naturalmente cada usuario verá los datos de manera diferente. Estas presentaciones se examinan en el modelo lógico global de la base de datos, que eventualmente deberá desarrollarse. Finalmente, el modelo lógico de la base de datos debe transformarse en el correspondiente diseño físico de la base de datos. El diseño físico considera la forma del almacenamiento de los datos y de sus interrelaciones, así como la mecánica del acceso.

El objetivo primordial de un SGBD es el de proporcionar un entorno que sea a la vez conveniente y eficiente para ser utilizado al extraer y almacenar información de la base de datos.

Los sistemas de bases de datos están diseñados para gestionar grandes bloques de información. La gestión de datos implica tanto la definición de estructuras para el almacenamiento de información como la provisión de mecanismos para la gestión de la información. Además, los sistemas de bases de datos deben mantener la seguridad de la información almacenada, pese a caídas del sistema o intentos de accesos no autorizados. Si los datos van a ser compartidos por varios usuarios, el sistema debe evitar posibles resultados anómalos.

La importancia de la información en la mayoría de las organizaciones, y por tanto el valor de la base de datos, ha llevado al desarrollo de una gran cantidad de conceptos y técnicas para la gestión eficiente de los datos.

1.3. Fases del diseño de Base de Datos

Con este enfoque, primero se diseña la base de datos, luego las aplicaciones que las usan. Este método se desarrolló en la década de 1970, con el establecimiento de la tecnología de bases de datos. La siguiente figura muestra las etapas previstas en un Enfoque Orientado a los Datos[**2**]:

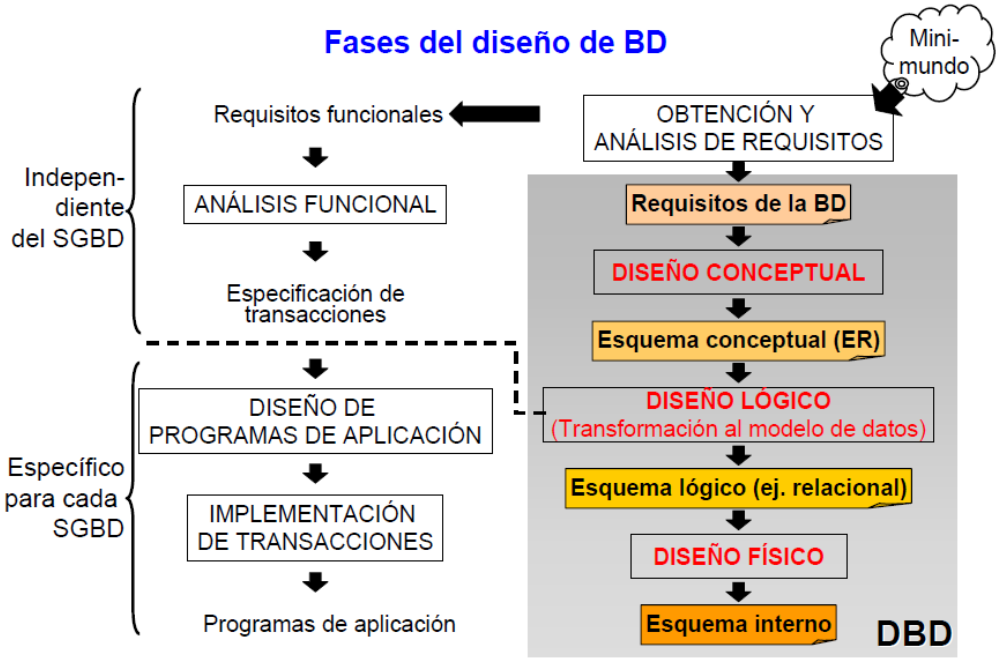


Figura 1. Fases del Diseño de Base de Datos

Los objetivos de cada etapa son las siguientes:

- **Diseño conceptual**. El propósito es describir el contenido de la información de la base de datos, más que las estructuras de almacenamiento que se necesitan para manejar esta información. Esta fase parte de la especificación de requerimientos y su resultado es un esquema conceptual. El esquema conceptual es una descripción de alto nivel de la estructura de la base de datos, independiente del software del SGBD que se use para manipularla. Es importante recalcar que los esquemas conceptuales se describen usando el modelo conceptual.

- **Diseño lógico**. Tiene como fin obtener el esquema lógico, que es una descripción de la estructura de la base de datos que puede procesar el software del SGBD.

  Para especificar el esquema lógico se usa el modelo lógico (jerárquico, de redes, o relacional, que es actualmente el más usado). El diseño lógico depende del

modelo de datos usado por el SGBD y no del SGBD utilizado (el diseño lógico se realiza de la misma forma para todos los SGBD relacionales porque todos utilizan el modelo relacional).

- **Diseño físico**. El objetivo es obtener el esquema físico, el cual es una descripción de la implantación de una base de datos en la memoria secundaria, describe las estructuras de almacenamiento y los métodos usados para tener acceso efectivo a los datos. Hay una retroalimentación entre el diseño físico y lógico, porque las decisiones tomadas durante el diseño físico para mejorar el rendimiento, pueden afectar la estructura del esquema lógico.

1.4. Componentes de los sistemas de bases de datos

Un sistema de base de datos se encuentra dividido en módulos cada uno de los cuales controla una parte de la responsabilidad total de sistema. En la mayoría de los casos, el sistema operativo proporciona únicamente los servicios más básicos y el sistema de la base de datos debe partir de esa base y controlar además el manejo correcto de los datos. Así el diseño de un sistema de base de datos debe incluir la interfaz entre el sistema de base de datos y el sistema operativo. [**3**]

Los componentes funcionales de un sistema de base de datos, son:

1. **Gestor de archivos:** Gestiona la asignación de espacio en la memoria del disco y de las estructuras de datos usadas para representar información.

2. **Manejador de base de datos:** Sirve de interfaz entre los datos y los programas de aplicación.

3. **Procesador de consultas:** Traduce las proposiciones en lenguajes de consulta a instrucciones de bajo nivel. Además convierte la solicitud del usuario en una forma más eficiente.

4. **Compilador de DDL:** Convierte las proposiciones DDL en un conjunto de tablas que contienen metadatos, estas se almacenan en el diccionario de datos.

5. **Archivo de datos:** En él se encuentran almacenados físicamente los datos de una organización.

6. **Diccionario de datos:** Contiene la información referente a la estructura de la base de datos.

7. **Índices:** Permiten un rápido acceso a registros que contienen valores específicos.

1.5. Abstracción de Datos

En un sistema de base de datos se proporciona a los usuarios una visión abstracta de los datos, es decir, el sistema esconde ciertos detalles de cómo se almacenan y mantienen los datos. La unión de todos los datos y sus relaciones forman el llamado esquema conceptual. Mientras que el esquema físico representa el almacenamiento de los datos y sus formas de acceso. El SGBD es el encargado de realizar las traducciones para pasar del esquema o nivel conceptual al físico.

Existen tres niveles de abstracción en las bases de datos para simplificar la interacción de los usuarios con el sistema, de acuerdo con el siguiente esquema:

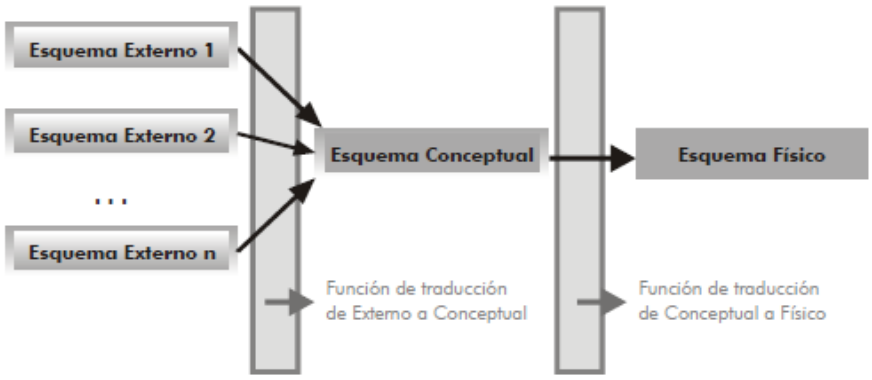


Figura 2. Niveles de Abstracción de Base de Datos

1.5.1. Nivel físico.

Es la representación del nivel más bajo de abstracción, en éste se describe en detalle la forma en cómo se almacenan los datos en los dispositivos de almacenamiento (por ejemplo, mediante señaladores o índices para el acceso aleatorio a los datos).

1.5.2. Nivel conceptual.

El siguiente nivel más alto de abstracción, describe que datos son almacenados realmente en la base de datos y las relaciones que existen entre los mismos, describe la base de datos completa en términos de su estructura de diseño. El nivel conceptual de abstracción lo usan los administradores de bases de datos, quienes deben decidir qué información se va a guardar en la base de datos.

Consta de las siguientes definiciones:

1. **Definición de los datos:** Se describen el tipo de datos y la longitud de campo todos los elementos direccionables en la base. Los elementos por definir incluyen artículos elementales (atributos), totales de datos y registros conceptuales (entidades).

2. **Relaciones entre datos:** Se definen las relaciones entre datos para enlazar tipos de registros relacionados para el procesamiento de archivos múltiples.

En el nivel conceptual la base de datos aparece como una colección de registros lógicos, sin descriptores de almacenamiento. En realidad los archivos conceptuales no existen físicamente. La transformación de registros conceptuales a registros físicos para el almacenamiento se lleva a cabo por el sistema y es transparente al usuario.

1.5.3. Nivel de visión.

Nivel más alto de abstracción, es lo que el usuario final puede visualizar del sistema terminado, describe sólo una parte de la base de datos al usuario acreditado para verla. El sistema puede proporcionar muchas visiones para la misma base de datos.

2. Modelos de Datos

2.1. Definición

Los modelos se utilizan en todo tipo de ciencias. Su finalidad es la de simbolizar una parte del mundo real de forma que sea más fácilmente manipulable. En definitiva es un esquema mental o conceptual en el que se intentan reproducir las características de una realidad específica.

En el caso de los modelos de datos es una colección integrada de conceptos, para describir y manipular datos, relaciones, significado y sus restricciones de consistencia todo ello dentro de una organización[**4**]. Los modelos de datos se describen, por lo general, mediante representaciones lingüísticas y graficas; es decir puede definirse una sintaxis y puede desarrollarse mediante una notación gráfica.

2.2. Características

Entre sus principales características se destacan las siguientes:

- Registrar los requerimientos de datos de un proceso de negocio.
- Se constituye como el proceso de analizar los aspectos de interés para una organización y la relación que tienen unos con otros.
- Resulta en el descubrimiento y documentación de los recursos de datos del negocio.
- El modelado hace la pregunta "¿Qué?" en lugar de "¿Cómo?", ésta última orientada al procesamiento de los datos.
- Permite observar patrones de datos y Usos potenciales de los datos

2.3. Clasificación de Modelos Conceptuales

2.3.1. Modelos Basados en Objetos.

Se usan para describir datos en los niveles conceptual y de visión, es decir, con este modelo representamos los datos de tal forma como nosotros los captamos en el mundo real, tienen una capacidad de estructuración bastante flexible y permiten

especificar restricciones de datos explícitamente. Existen diferentes modelos de este tipo:[**5**]

- **Modelo entidad relación:** el más utilizado por su sencillez y eficiencia, tienen gran aceptación en el diseño de base de datos y se usa ampliamente en la práctica. Este modelo representa a la realidad a través de entidades, atributos y relaciones.

- **Modelo orientado a objetos**: incluye muchos de los conceptos del modelo entidad-relación, pero presenta tanto código ejecutable como datos. Los objetos encapsulan atributos (forman el estado) y métodos (servicios que brinda) lo que le da una cierta funcionalidad. Los objetos coordinan sus actividades a través del llamado mutuo de métodos.

2.3.2. Modelos Basados en Registros.

Se utilizan para describir datos en los niveles conceptual y físico. Estos modelos utilizan registros e instancias para representar la realidad, así como las relaciones que existen entre estos registros (ligas) o apuntadores. A diferencia de los modelos de datos basados en objetos, se usan para especificar la estructura lógica global de la base de datos y para proporcionar una descripción a nivel más alto de la implementación.

Loa 3 modelos más aceptados son[**6**]:

- **Modelo Relacional:** El modelo relacional para la gestión de una base de datos es un modelo de datos basado en la lógica de predicado y en la teoría de conjuntos. Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). De manera simple, una relación representa una tabla, en que cada fila representa una colección de valores que describen una entidad del mundo real. Cada fila se denomina tupla o registro y cada columna campo.

- **Modelo de Red:** Este modelo representa los datos mediante colecciones de registros y sus relaciones se representan por medio de ligas o enlaces, los cuales

pueden verse como punteros. Los registros se organizan en un conjunto de gráficas arbitrarias.

- **Modelo Jerárquico:** fue el pionero en los sistemas de bases de datos, allá por comienzos de los años 60. En el modelo jerárquico sólo se pueden modelar relaciones 1:N, aunque esto se podía arreglar mediante el uso de los llamados "vínculos virtuales" que posibilitan las relaciones N:M.

Al igual que con el modelo de red, el principal problema de los sistemas de bases de datos jerárquicos es el de la poca independencia de los programas respecto a cómo están almacenados los datos, lo que dificulta además la programación de software de acceso a estos sistemas. Es similar al modelo de red en cuanto a las relaciones y datos, ya que estos se representan por medio de registros y sus ligas. La diferencia radica en que están organizados por conjuntos de árboles en lugar de gráficas arbitrarias.

2.4. Modelos de Datos como instrumento de Diseño

Todo modelo de datos tiene la siguiente estructura[**7**]:

1. Estática

- Conjunto de objetos
- Conjunto de asociaciones entre ellos
- Conjunto de restricciones

2. Dinámica

- Recuperación
- Actualización

3. Restricciones: Limitaciones impuestas a la estructura del modelo a los datos que invalidan ciertas ocurrencias de la BD

- Inherentes (propias del modelo): limitaciones impuestas a la estructura del modelo
- Semánticas (propias del usuario): limitaciones impuestas a los valores de los atributos o a las características de las interrelaciones

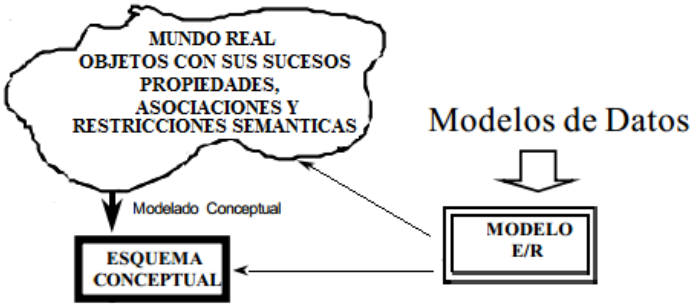


Figura 3. Metodología para el desarrollo de Bases de Datos

3. Diseño Conceptual de Base de Datos

El diseño conceptual se hace independiente al sistema gestor de base de datos (SGBD) que utilice el usuario para la implementación de esta. Para modelar conceptualmente es posible utilizar varios Modelos de Datos, un modelo práctico para ilustrar el diseño conceptual es el modelo entidad relación.

El objetivo es la construcción de un esquema E/R a partir de los requisitos del usuario. El proceso de construcción es incremental: el esquema conceptual se refina y enriquece durante una serie de transformaciones y correcciones. [**8**]

3.1. Etapas del Diseño Conceptual

El Diseño Conceptual es la primera fase del desarrollo de Base de Datos. Se subdivide en dos etapas:[**9**]

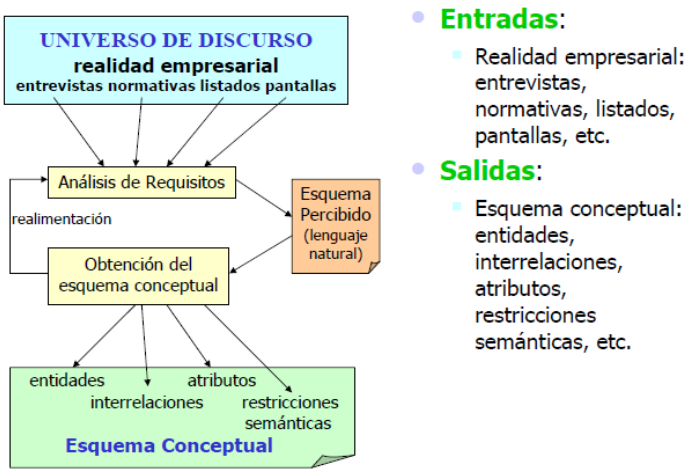


Figura 4. Etapas del Diseño Conceptual

3.1.1. Análisis de Requisitos

Esta primera etapa, en general común para datos y procesos, es de percepción, identificación y descripción de los fenómenos del mundo real a analizar. En el análisis de requisitos se ha de responder a la pregunta: "¿Qué representar?". Mediante el estudio de las reglas de una empresa (que proveen el marco para el análisis del sistema) y de entrevistas a los usuarios de los diferentes niveles de la organización (que proveen los detalles sobre los datos) se llega a elaborar un esquema descriptivo de la realidad. El esquema descriptivo se representa utilizando el lenguaje natural, con ello se ayuda a que el problema de comunicación usuarios/analistas se reduzca.

3.1.2. Generación del esquema conceptual (Conceptualización):

En esta segunda etapa se transforma el esquema descriptivo, refinándolo y estructurándolo adecuadamente.

Esta etapa responde a la pregunta:"¿Cómo representar?", se habrá de buscar una representación normalizada que se apoye en un modelo de datos que cumpla determinadas propiedades (coherencia, plenitud, no redundancia, simplicidad, fidelidad, etc.), para llegar así al denominado esquema conceptual.

Para la representación del esquema conceptual, usaremos el Modelo E-R, además de una serie de fichas o plantillas que sirven de complemento documental al diagrama entidad-relación.

Para generar el esquema conceptual es preciso interpretar las frases del lenguaje natural en el que está descrito el esquema percibido, convirtiéndolas en elementos del modelo entidad-relación.

Si bien no existen reglas deterministas que digan qué elemento va a ser una entidad o cuál otro una interrelación, sí es posible enunciar unos principios generales que, junto al buen criterio del diseñador, puedan ayudar a elaborar un primer esquema conceptual, que será sometido después a un proceso de refinamientos sucesivos. Así, una preposición o frase preposicional entre dos nombres suele ser un tipo de interrelación, o también puede establecer la asociación entre una entidad y sus atributos.

3.2. Criterios generales para la representación de datos

Las siguientes son directrices para traducir las especificaciones informales en constructores del modelo E/R.

- Si un concepto tiene propiedades significativas y/o describe clases de objetos con una existencia autónoma, es apropiado representarlo con una entidad. Por ejemplo: instructor, porque tiene propiedades como nombre, edad y ciudad de nacimiento, y porque su existencia es independiente de otros conceptos.

- Si un concepto tiene una estructura simple, y no tiene propiedades relevantes asociadas, es conveniente representarlo por un atributo de otro concepto al que se refiera. Por ejemplo: edad, ciudad, si bien es cierto tienen propiedades significativas, pero no son relevantes para nosotros, así que lo establece como atributo.

- Si los requisitos contienen un concepto que proporciona un vínculo lógico entre dos (o más) entidades, este concepto se puede representar por una relación. Por ejemplo: Asistir a un curso como relación entre las entidades que representan a los alumnos y a las ediciones de los cursos. Un concepto se debe representar como entidad si el concepto representa una acción que puede repetirse.

- Si uno o más conceptos son casos particulares de otro concepto, es conveniente representarlos mediante generalización.

4. Modelo Entidad-Relación

4.1. Concepto

El modelo entidad-relación ER es un modelo de datos que permite representar cualquier abstracción, percepción y conocimiento en un sistema de información formado por un conjunto de objetos denominados entidades y relaciones, incorporando una representación visual conocida como diagrama entidad-relación.[**10**]

El modelo de datos entidad-relación (E-R) se basa en la percepción de un mundo real, se desarrolló para facilitar el diseño de bases de datos permitiendo la especificación de un esquema empresarial. Este esquema representa la estructura lógica global de la base de datos. Es el modelo de datos más ampliamente usado para el diseño conceptual de bases de datos. El modelo entidad-relación fue propuesto inicialmente

por Peter Chen en 1976 y ha sido estudiado por varios autores. El MER se ve principalmente como una herramienta de diseño.

4.2. Elementos del Modelo Entidad Relación

4.2.1. Entidades

La entidad es cualquier clase de objeto o conjunto de elementos presentes o no, en un contexto determinado dado por el sistema de información o las funciones y procesos que se definen en un plan de automatización[**11**]. Dicho de otra forma, las entidades las constituyen las tablas de la base de datos que permiten el almacenamiento de los ejemplares o registros del sistema, quedando recogidos bajo la denominación o título de la tabla o entidad. Por ejemplo, la entidad usuarios guarda los datos personales de los usuarios de la biblioteca, la entidad catalogo registra todos los libros catalogados, la entidad circulación todos los libros prestados y devueltos y así sucesivamente con todos los casos.

En el modelo entidad-relación las entidades se representan con un rectángulo dentro del cual se escribe el nombre de la entidad:

Figura 5. Representación de una Entidad

Existen dos tipos de entidades:

- **Entidades fuertes**. Lo constituyen las tablas principales de la base de datos que contienen los registros principales del sistema de información y que requieren de entidades o tablas auxiliares para completar su descripción o información. Por ejemplo la tabla usuario es una entidad fuerte en relación a la tabla tipos de usuarios, que es una entidad débil dada su condición auxiliar para clasificar a los usuarios registrados en la biblioteca.
- **Entidades débiles**. Son entidades débiles a las tablas auxiliares de una tabla principal a la que completan o complementan con la información de sus registros relacionados. Por ejemplo también son consideradas entidades débiles las tablas intermedias que sirven para compartir información de varias tablas principales

Figura 6. Representación de Entidad Débil

4.2.2. Relaciones

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.[**8**]

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.

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

Figura 7. Representación de una Relación

4.2.3. Atributos

Son las características, rasgos y propiedades de una entidad, que toman como valor una instancia particular. Es decir, los atributos de una entidad son en realidad sus campos descriptivos, el predicado que permite definir lo que decimos de un determinado sujeto. Por ejemplo de una entidad o tabla catálogo, se pueden determinar los atributos: *título, subtítulo, título paralelo, otras formas del título, autor*

principal, otras menciones de responsabilidad, edición, mención de edición, editorial, lugar de publicación, fecha de publicación,..

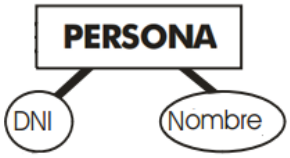


Figura 8. Representación de un atributo

4.2.4. Cardinalidad en Relaciones

La cardinalidad se representan en un diagrama ER como una etiqueta que se ubica en ambos extremos de la línea de relación de las entidades y que puede contener diversos valores entre los que destacan comúnmente el 1 y el *, obteniendo los siguientes tipos:

- **Relación 1:1**. La relación uno a uno, define que una única entidad le corresponde como máximo una ocurrencia de la otra entidad relacionada.

Figura 9. Esquema de Relación 1 a 1

- **Relación 1:N**. La relación de uno a varios, define que a cada ocurrencia de la entidad A le pueden corresponder varias de la entidad B.

Figura 10. Esquema de Relación 1 a muchos

- **Relación N:N**. La relación de varios a varios, define que cada ocurrencia de una entidad puede contener varias de la otra entidad relacionada y viceversa.

Figura 11 Esquema de Relación muchos a muchos

4.2.5. Clave.

Es el campo o atributo de una entidad o tabla que tiene como objetivo distinguir cada registro del conjunto, sirviendo sus valores como datos vinculantes de una relación entre registros de varias tablas.

- **Superclave**: Es la combinación de campos clave que identifican unívocamente un registro en una tabla o entidad.
- **Clave principal primaria**: Permiten identificar unívocamente cada registro de una tabla. Por ejemplo campo auto-numérico interno ID.
- **Clave candidata:** Campos que cumplen las condiciones de identificación única de registros, pero que no fueron definidos como principales por el diseñador.
- **Clave externa**: Campo clave conformado por el valor de una clave principal primaria de otra tabla.

4.3. Diccionario de Datos

A continuación se describen los conceptos y utilidades que ofrece la construcción del diccionario de datos[**12**].

4.4. Definición

Un diccionario de datos es un conjunto de metadatos que contiene las características lógicas y puntuales de los datos que se van a utilizar en el sistema que se programe, incluyendo nombre, descripción, alias, contenido y organización.

El diccionario se desarrolla durante el análisis de flujo de datos y auxilia a los analistas que participan en la determinación de los requerimientos de sistemas. Su objetivo es dar precisión sobre los datos que se manejan en un sistema, evitando así malas interpretaciones o ambigüedades.

También se lo considera como un catálogo, un depósito, de los elementos en un sistema. Como su nombre lo sugiere, estos elementos se centran alrededor de los datos y la forma en que están estructurados para satisfacer los requerimientos de los usuarios y las necesidades de la organización. Aquí se encuentra la lista de todos los elementos que forman parte del flujo de datos en todo el sistema. Los elementos más importantes son flujos de datos, almacenes de datos y procesos.

En el diccionario de datos de define con precisión los datos de entrada, salida, componentes de almacenes, flujos, detalles de las relaciones entre almacenes, etc., siendo un buen complemento al diagrama entidad-relación.

4.5. Uso del diccionario de datos

Entre las razones más comunes tenemos las siguientes:

- Para manejar los detalles en sistemas muy grandes, ya que tienen enormes cantidades de datos, aun en los sistemas más chicos hay gran cantidad de datos.
- Los sistemas al sufrir cambios continuos, es muy difícil manejar todos los detalles, por eso se registra la información, ya sea sobre hoja de papel o usando procesadores de texto. Los analistas más organizados usan el diccionario de datos automatizados diseñados específicamente para el análisis y diseño de software.
- Para asignarle un solo significado a cada uno de los elementos y actividades del sistema.
- Los elementos más importantes son flujos de datos, almacenes de datos y procesos. El diccionario de datos guarda los detalles y descripción de todos estos elementos.

5. Fundamentos de la Web

5.1. Protocolo Http

El protocolo HTTP (Hyper Text Transfer Protocol) es el protocolo base de la WWW. Es un protocolo simple, orientado a conexión y sin estado [**13**].Es un protocolo Orientado a conexión ya que emplea para su funcionamiento un protocolo de comunicaciones(TCP, Transport Control Protocol) de modo conectado, un Protocolo que establece un canal de comunicaciones de extremo a extremo (entre el cliente y el servidor)por el que pasa el flujo de bytes que constituyen los datos a transferir, en contraposición a los protocolos de datagrama o no orientados a conexión que dividen los datos en pequeños paquetes (datagramas) y los envían, pudiendo llegar por vías diferentes del servidor al cliente. El protocolo no mantiene estado, es decir, cada transferencia de datos es una conexión independiente de la anterior, no manteniendo ninguna relación entre ellas. Esto es así hasta el punto de que para transferir una

página Web debemos enviar el código HTML del texto así como las imágenes que la componen.

5.2. El Lenguaje HTML

El otro puntal del éxito del WWW es el lenguaje HTML (Hyper Text Mark-up Language). Este es un lenguaje de marcas (se utiliza insertando marcas en el interior del texto) que nos permite representar de forma rica el contenido, así como referenciar otros recursos (imágenes textos, imágenes, fotografías, audio, sonido, animaciones, video), enlaces a otros documentos (la característica más destacada del WWW), mostrar formularios para luego procesarlos, etc.

El lenguaje HTML actualmente se encuentra en la versión 5.0, las novedades más destacables de HTML 5 serán la inclusión de API's para realizar dibujos en dos dimensiones, controlar la reproducción de audio y vídeo, editar documentos de forma interactiva en el navegador, y mantener datos de forma persistente en la parte cliente de la comunicación para acceder más tarde a ellos.

5.3. XML

XML (Extensible Markup Language) es un lenguaje de etiquetas que ofrece un formato para la descripción de datos estructurados. Esto facilita la declaración de contenido más precisos y unos resultados de búsquedas más significativos en varias plataformas, es decir se trata de un metalenguaje que separa el contenido de la presentación.

Según la especificación, los objetivos de diseñar XML fueron los siguientes[**14**]:

- Es directamente utilizable en Internet
- Soportar una amplia variedad de aplicaciones
- Debería se sencillo escribir programas que procesaran documentos XML
- Los documentos XML deberían ser legibles por las personas y razonablemente claros
- El diseño de XML debe ser rápido
- XML debería ser simple, pero perfectamente normalizado
- Los documentos XML deben ser de fácil creación

Los esquemas (schemas) son estructuras más potentes y expresivas que las DTDs(Declaración de Tipo de Documento), ya que permiten especificar el contenido de los documentos en función del tipo de datos empleado. Con ellos, se pretende definir la estructura, contenidos y semántica de los documentos.

Los esquemas indican tipos de datos, número mínimo y máximo de ocurrencias y otras características más específicas, a continuación un ejemplo de Esquema (XML Schema):

Ejemplo de en formato XML:

```
<Datos-Nacimiento>
  <Persona>
    <Nombre>Mateo</Nombre>
    <Fecha>15.10.2012</Fecha>
    <Ciudad>Madrid</Ciudad>
    <Peso>3.1Kg</Peso>
    <Estatura>45cm</Estatura>
  </Persona>
  <Persona>
    <Nombre>Maribel</Nombre>
    <Fecha>11.09.2012</Fecha>
    <Ciudad>Sevilla</Ciudad>
    <Peso>3Kg</Peso>
    <Estatura>40cm</Estatura>
  </Persona>
</Datos-Nacimiento>
```

5.4. Lenguaje PHP

Es un lenguaje de programación enfocado principalmente a la programación de scripts del lado del servidor, originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que

puede ser usada en aplicaciones independientes. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

Entre sus principalmente características se encuentran las siguientes[**15**]:

- **Scripts del lado del servidor**: Este es el campo más tradicional y el foco principal. Son necesarias tres cosas para que esto funcione. El analizador de PHP (módulo CGI o servidor), un servidor web y un navegador web. Es necesario ejecutar el servidor con una instalación de PHP conectada. Se puede acceder al resultado del programa de PHP con un navegador, viendo la página de PHP a través del servidor.
- **Scripts desde la línea de comandos**: Se puede crear un script de PHP y ejecutarlo sin necesidad de un servidor o navegador. Solamente es necesario el analizador de PHP para utilizarlo de esta manera. Este tipo de uso es ideal para scripts que se ejecuten con regularidad empleando linux o el planificador de tareas (en Windows). Estos scripts también pueden usarse para tareas simples de procesamiento de texto.
- **Escribir aplicaciones de escritorio**: Probablemente PHP no sea el lenguaje más apropiado para crear aplicaciones de escritorio con una interfaz gráfica de usuario, pero si se conoce bien PHP, y se quisiera utilizar algunas características avanzadas de PHP en aplicaciones del lado del cliente, se puede utilizar PHP-GTK para escribir dichos programas. También es posible de esta manera escribir aplicaciones independientes de una plataforma. PHP-GTK es una extensión de PHP, no disponible en la distribución principal.
- **Soporte para bases de datos**: realiza esta tarea utilizando una de las extensiones específicas de bases de datos (p.ej., para mysql, etc), o utilizar una capa de abstracción como PDO, o conectarse a cualquier base de datos que admita el estándar de Conexión Abierta a Bases de Datos por medio de la extensión ODBC.
- Soporte para la instalación de objetos de Java y emplearlos de forma transparente como objetos de PHP.

De modo que con PHP, se tiene la libertad de elegir el sistema operativo y el servidor web. Además, se tiene la posibilidad de utilizar programación por procedimientos o programación orientada a objetos (POO), o una mezcla de ambas. No se está limitado

a generar HTML. También se puede generar fácilmente cualquier tipo de texto, como XHTML y cualquier otro tipo de fichero XML. PHP puede autogenerar estos ficheros y guardarlos en el sistema de ficheros en vez de imprimirlos en pantalla, creando una caché en el lado del servidor para contenido dinámico.

Se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos hasta el día de hoy, lo que ha atraído el interés de múltiples sitios con gran demanda de tráfico. Actualmente el lenguaje sigue siendo desarrollado con nuevas funciones por el grupo PHP. Este lenguaje forma parte del software libre publicado bajo la licencia PHP.

5.5. Javascript

Es un lenguaje de programación que surgió con el objetivo inicial de programar ciertos comportamientos sobre las páginas web, respondiendo a la interacción del usuario y la realización de automatismos sencillos[**16**]. En ese contexto podríamos decir que nació como un "lenguaje de scripting" del lado del cliente, sin embargo, hoy Javascript es mucho más. Las necesidades de las aplicaciones web modernas y el HTML5 ha provocado que el uso de Javascript que encontramos hoy haya llegado a unos niveles de complejidad y prestaciones tan grandes como otros lenguajes de primer nivel.

Pero además, en los últimos años Javascript se está convirtiendo también en el lenguaje "integrador". Lo encontramos en muchos ámbitos, ya no solo en Internet y la Web, también es nativo en sistemas operativos para ordenadores y dispositivos, del lado del servidor y del cliente.

En el contexto de un sitio web, con Javascript se puede hacer todo tipo de acciones e interacción. Antes se utilizaba para validar formularios, mostrar cajas de diálogo y poco más. Hoy es el motor de las aplicaciones más conocidas en el ámbito de Internet: Google, Facebook, Twitter, Outlook... etc, tienen su núcleo realizado en Javascript. La Web 2.0 se basa en el uso de Javascript para implementar aplicaciones enriquecidas que son capaces de realizar todo tipo de efectos, interfaces de usuario y comunicación asíncrona con el servidor por medio de Ajax.

5.6. Canvas

Es un elemento HTML el cual puede ser usado para dibujar gráficos usando scripts (normalmente JavaScript). Este puede, por instancia, ser usado para dibujar gráficos, realizar composición de fotos o simples (y no tan simples) animaciones.

Fue introducido primero por Apple para el Mac OS X Dashboard y después implementado en Safari y Google Chrome. Navegadores basados en Gecko 1.8, tal como Firefox 1.5, que también soportan este elemento. El <canvas> es un elemento que parte de las especificaciones de la WhatWG Web applications 1.0 mejor conocida como HTML5.

5.6.1. Eje de coordenadas de Canvas

Para posicionar elementos en el canvas se debe tener en tener en cuenta su eje de coordenadas en dos dimensiones, que comienza en la esquina superior izquierda del lienzo. El lienzo producido por canvas tendrá unas dimensiones indicadas con los atributos width y height en la etiqueta CANVAS. Por tanto, la esquina superior izquierda será el punto (0,0) y la esquina inferior derecha el punto definido por (width-1, height-1), es decir, el punto máximo de coordenadas marcado por su anchura y altura.

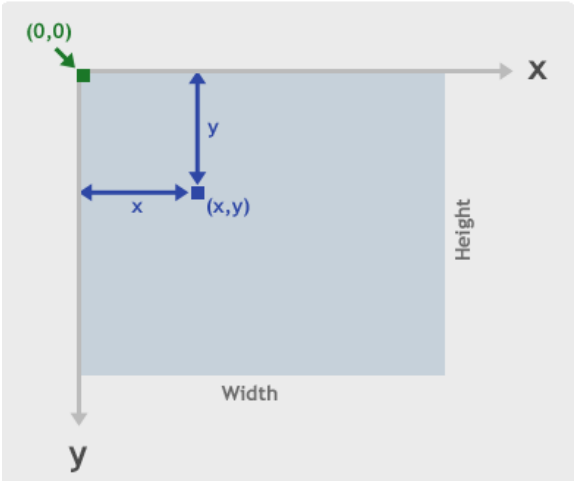


Figura 12. Eje de Coordenadas de Canvas

Cualquier punto dentro del canvas se calcula con la coordenada (x,y), siendo que la x crece según los pixel a la derecha y la y con los pixel hacia abajo.

El tamaño por defecto del canvas es 300px * 150px [ancho (width) * alto (height)]. Pero se puede personalizar el tamaño usando las propiedades height y width de CSS. Con el fin de dibujar gráficos en el lienzo <canvas> se utiliza un objeto de contexto de JavaScript que crea gráficos sobre la marcha.

Para dibujar cualquier tipo de forma en el canvas necesitaremos posicionarla con respecto a las coordenadas.

5.6.2. Formas Básicas en Canvas

Entre ellas se definen las siguientes [**17**]:

- **Rectángulos**

A diferencia de SVG,<canvas> sólo admite una forma primitiva: rectángulos. Todas las demás formas se deben crear mediante la combinación de scripts, ya sea líneas, círculos, etc. Para ello existe una variedad de funciones de trayectoria de dibujo que permiten crear formas muy complejas.

Hay tres funciones que dibujan rectángulos en el lienzo:

fillRect(x, y, ancho, alto): Dibuja un rectángulo relleno.

strokeRect(x, y, ancho, alto): Dibuja un contorno rectangular.

clearRect(x, y, ancho, alto): Borra el área rectangular especificada, por lo que es totalmente transparente.

Cada una de estas tres funciones toma los mismos parámetros x, e y para especificar la posición en el lienzo (en relación con el origen) de la esquina superior izquierda del rectángulo, los parámetros anchura y altura proporcionar el tamaño del rectángulo.

- **Líneas**

Para dibujar líneas rectas, se emplea el método lineTo.

lineTo(x, y): Dibuja una línea desde la posición de dibujo actual hasta la posición especificada por x y y .

Este método tiene dos argumentos, x e y, que son las coordenadas del punto final de la línea. El punto de partida es dependiente de caminos dibujados previamente, en el que el punto final de la trayectoria anterior es el punto de partida para la siguiente, etc. El punto de partida también se puede cambiar utilizando el moveTo () método.

- **Círculo**

El método en canvas que nos permite dibujar círculos y en su defecto arcos, es el denominado cxt.arc, su funcionamiento es de la siguiente manera:

cxt.arc(x,y,radio,0,Math.PI*2,true): Esta función permite crear un arco y sirve para hacer el círculo, respecto a los parámetros, x e y son los puntos de inicio, radio, que es el punto que creamos con las coordenadas X, Y, luego empezamos un círculo desde 0 hasta el final del arco expresado en radianes ya que no soporta grados, (1 vuelta completa de círculo es igual a 2 PI Radianes), y se determina PI de la librería Math, entonces se obtiene Math.PI*2 y por ultimo true para ver la parte positiva del arco es decir se hace el recorrido de acuerdo a como giran las manecillas del reloj.

5.7. SQLite

SQLite es una pequeña librería programada en lenguaje C que implementa un completo motor de base de datos multiplataforma que no precisa configuración. Se distribuye bajo licencia de dominio público. Es muy rápido y la ventaja fundamental es que permite utilizar un amplio subconjunto del lenguaje estándar SQL. SQLite destaca también por su versatilidad. El motor de PHP 5 incluye soporte interno para SQLite.

Combina el motor y el interfaz de la base de datos en una única biblioteca, y almacena los datos en un único archivo de texto plano. Esto hace que cada usuario pueda crear tantas bases de datos como desee sin la necesidad de la intervención de un administrador de bases de datos que gestione los espacios de trabajo, usuarios y permisos de acceso. El hecho de almacenar toda la base de datos en un único archivo, facilita la portabilidad de los datos, y solamente tiene la restricción del espacio de disco asignado al usuario en el servidor.

Su potencia se basa fundamentalmente en la simplicidad, lo que hace que no sea una buena solución en entornos de tráfico muy elevado y/o alto acceso concurrente a datos.

Entre sus características principales se encuentran [**18**]:

- **Tamaño:** SQLite tiene una pequeña memoria y una única biblioteca es necesaria para acceder a bases de datos, lo que lo hace ideal para aplicaciones de bases de datos incorporadas.

- **Rendimiento de base de datos:** SQLite realiza operaciones de manera eficiente y es más rápido que MySQL y PostgreSQL.

- **Portabilidad:** se ejecuta en muchas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración.

- **Estabilidad:** SQLite es compatible con ACID, reunión de los cuatro criterios de Atomicidad, Consistencia, Aislamiento y Durabilidad.

- **Interfaces:** cuenta con diferentes interfaces del API, las cuales permiten trabajar con C++, PHP, Perl, Python, Ruby, Tcl, Groovy, Qt, etc.

- **Costo:** SQLite es de dominio público, y por tanto, es libre de utilizar para cualquier propósito sin costo y se puede redistribuir libremente.
- En su versión 3, SQLite permite bases de datos de hasta 2 Terabytes de tamaño.

6. Metodologías de Desarrollo de Software

6.1. Metodologías de desarrollo ágil

Las metodologías ágiles buscan modelar y organizar con eficacia el desarrollo de software. En pocas palabras, representan un conjunto de valores, principios y prácticas para crear productos software de manera más eficaz, flexible y 'ligera' que las metodologías 'pesadas' o tradicionales, al menos en contextos de trabajo donde hay bastante incertidumbre [**19**].

Las metodologías ágiles son en realidad una familia de modelos o técnicas, todas ellas compartiendo la característica de interpretar el desarrollo de software como una actividad en la que siempre hay un cierto grado de incertidumbre. Incertidumbre que hace necesario poner el énfasis en las personas, dejar que se auto-organicen y que interactúen buscando siempre satisfacer los requisitos del cliente, y planificando

iteración a iteración, adaptándose con flexibilidad a los cambios que se producirán de seguro durante la vida del proyecto.

Las metodologías ágiles presentan diversas ventajas como[**20**]:

- Rápida respuesta a cambios de requisitos a lo largo del desarrollo.
- Entrega continua y en plazos cortos de software funcional.
- Trabajo conjunto entre el cliente y el equipo de desarrollo.
- Minimiza los costos frente a cambios.
- Importancia de la simplicidad, al eliminar el trabajo innecesario.
- Atención continua a la excelencia técnica y al buen diseño.
- Mejora continua de los procesos y el equipo de desarrollo.
- Evita malentendidos de requerimientos entre el cliente y el equipo.
- El equipo de desarrollo no malgasta el tiempo y dinero del cliente desarrollando soluciones innecesariamente generales y complejas que en realidad no son un requisito del cliente.
- Cada componente del producto final ha sido probado y satisface los requerimientos.

6.1.1. Programación Extrema (XP)

XP es una metodología que sigue la filosofía de las metodologías ágiles, cuyo objetivo es conseguir la máxima satisfacción del cliente en forma rápida y eficiente ante los cambios de requisitos. XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo[**21**]. Entre sus características están:

- Realimentación continua entre el cliente y el equipo de desarrollo
- Comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios.
- Propone realizar diseños simples, códigos simples y proporcionar rápida respuesta de lo requerido y lograr un cliente contento.
- Se sustituye la documentación escrita por la comunicación directa entre clientes y desarrolladores o entre los propios desarrolladores.

- Propone un desarrollo iterativo a través de cuatro pasos, planificación, diseño, codificación y prueba. En cada iteración se añaden nuevas funcionalidades al software. Es especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

Se basa en una serie de prácticas y principios que se han ido gestando a lo largo de toda la historia de la ingeniería del software, son de sentido común pero llevadas al extremo.

6.1.2. SCRUM

Se centra principalmente a nivel de las personas y del equipo de desarrollo que constituye el producto. Su objetivo es que los miembros del equipo trabajen juntos y de forma eficiente optimizando productos complejos y sofisticados.

Esta metodología define un marco para la gestión de proyectos, está especialmente indicada para proyectos con un rápido cambio de requisitos.

Las actividades que se llevan a cabo en Scrum son las siguientes[**22**]:

- **Planificación de la iteración**: El primer día de la iteración se realiza la reunión de planificación de la iteración. Tiene dos partes:

1. **Selección de requisitos** (4 horas máximo). El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto. El equipo pregunta al cliente las dudas que surgen y selecciona los requisitos más prioritarios
2. **Planificación de la iteración** (4 horas máximo). El equipo elabora la lista de tareas de la iteración necesarias para desarrollar los requisitos a que se ha comprometido. La estimación de esfuerzo se hace de manera conjunta y los miembros del equipo se auto asignan las tareas.

- **Ejecución de la iteración**: Cada día el equipo realiza una reunión de sincronización (15 minutos máximos). Cada miembro del equipo inspecciona el trabajo que el resto está realizando (dependencias entre tareas, progreso hacia el objetivo de la iteración, obstáculos que pueden impedir este objetivo) para poder hacer las adaptaciones necesarias que permitan cumplir con el compromiso adquirido.

- **Inspección y adaptación:** El último día de la iteración se realiza la reunión de revisión de la iteración. Tiene dos partes:

1. **Demostración** (4 horas máximo). El equipo presenta al cliente los requisitos completados en la iteración, en forma de incremento de producto preparado para ser entregado con el mínimo esfuerzo. En función de los resultados mostrados y de los cambios que haya habido en el contexto del proyecto, el cliente realiza las adaptaciones necesarias de manera objetiva, ya desde la primera iteración, replanificando el proyecto.
2. **Retrospectiva** (4 horas máximo). El equipo analiza cómo ha sido su manera de trabajar y cuáles son los problemas que podrían impedirle progresar adecuadamente, mejorando de manera continua su productividad. El Facilitador se encargará de ir eliminando los obstáculos identificados.

6.1.3. Rapid Aplication Development(RAD)

El "Desarrollo Rápido de Aplicaciones" (RAD) es una metodología que permite a las organizaciones desarrollar sistemas estratégicamente importantes, de manera más rápida reduciendo a la vez los costos de desarrollo y manteniendo la calidad. La gran mayoría de las herramientas gráficas orientadas a objetos tienen interiorizadas el concepto general de RAD. Además, con la creación bien planificada de objetos, la programación de nuevos módulos se vuelve cada vez más simplificada, reutilizando los objetos creados anteriormente.

Algunas de las características principales de esta metodología son:

- Ciclo de desarrollo iterativo e incremental
- Equipos Híbridos: gente motivada y muy versátil capaz de desempeñar diferentes roles durante el desarrollo del proyecto
- Reutilización de componentes
- Utiliza el prototipo como modelo de aproximación a la solución final
- Prototipado rápido con el objetivo de obtener en el menor tiempo posible el análisis, diseño e implementación con la ayuda de herramientas CASE.

6.1.4. Diferencias entre Metodologías Ágiles

En la TABLA I, se presenta una comparativa en base a las principales características que definen cada metodología.

TABLA I. DIFERENCIA ENTRE METODOLOGÍAS AGÍLES.

RAD	SCRUM	XP
Los ciclos de desarrollo son más pequeños ya que se emplea herramientas que facilitan la generación de código.	Las Iteraciones De entrega son de dos a cuatro semanas	Las iteraciones de entrega son de una a tres semanas
Los encargados del proyecto involucran a los usuarios del sistema	Los miembros del equipo trabajan en forma individual	Los miembros del equipo trabajan en parejas
Las fases realizadas pueden ser modificadas durante el transcurso del proyecto	Las tareas que se hayan realizado y que el propietario del producto haya mostrado su conformidad ya que no se retocan, si funciona y está bien se aparta para avanzar	Las tareas se van terminando aunque son susceptibles de ser modificadas durante el transcurso del proyecto
El orden de prioridades de las tareas puede ser modificado mostrando mayor flexibilidad.	Trata de seguir el orden de prioridades de tareas pero se puede cambiar si es mejor para el desarrollo de las tareas	Siguen estrictamente el orden de prioridad de las tareas definidas por el cliente.
Es una metodología de desarrollo ágil basada en la construcción de prototipos	Es una metodología de desarrollo ágil basada más en la administración del proyecto.	Se centra en la programación o en la creación del proyecto.

e. Materiales y Métodos

Respecto a los métodos empleados para el desarrollo del trabajo de titulación lo cual involucra el análisis, diseño, desarrollo e implementación así como la elección de la metodología de software más conveniente a utilizar se ha estimado conveniente emplear los siguientes:

1. Métodos

- **Método Analítico**: Este método se empleó en la elaboración de la situación problemática así como también en la obtención de un estudio y análisis de los requerimientos que conlleva la realización del diseño conceptual de Base de Datos. De igual manera para establecer las definiciones que conformaron la documentación del trabajo de titulación.

- **Método Científico:** Permitió efectuar un estudio sistemático del proceso de Diseño de Base de Datos a través de las técnicas de observación, ideas sobre la experimentación planificada lo cual contribuyo al desarrollo de la aplicación así como la manera de presentar los resultados donde en base a ello poder establecer las conclusiones finales.

- **Método Deductivo:** Con este método se logró en base a una problemática general ya conocida y definida poder concluir en consecuencias particulares de forma que se proporcionó alternativas de solución para inferir en el mejor desarrollo de una metodología de diseño de base de datos.

1.1. Metodología de Desarrollo

Considerando las diversas metodologías de desarrollo de software se ha creído factible emplear la metodología de desarrollo ágil RAD(Desarrollo Rápido de Aplicaciones),debido a que su filosofía busca adaptarse fácilmente a las circunstancias reales, teniendo como base sus cuatro etapas y un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. Esta metodología comprende el desarrollo interactivo y tiende a englobar también la usabilidad, utilidad y la rapidez de ejecución.

Es así que para su aplicación nos fundamentamos en las siguientes fases de la metodología, tales como:

- **Fase de Planeación de Requerimientos**

Esta etapa requirió que usuarios con conocimiento del proceso de diseño de bases de datos determinen cuales fueron las funciones del sistema, estructurando los elementos de análisis que establecían las restricciones y los requerimientos del sistema.

- **Fase de Diseño de Usuario**

Durante esta fase, se realizó énfasis en la creación rápida de prototipos que representan a todos los procesos del sistema, entradas y salidas, este fue un proceso interactivo continuo que permitió al usuario entender, modificar, y eventualmente aprobar un modelo de trabajo del sistema que se ajuste a las necesidades de diseño.

- **Fase de Construcción**

En esta fase tomando en cuenta los prototipos iniciales mismos que incluyeron algunas pero no todas las características que tendría el sistema final y con un entorno de configuraciones ya establecidas se llevaron a cabo las tareas de desarrollo y codificación de la aplicación, cuyo prototipos estuvieron sujetos a modificaciones, donde al culminar se realizaron las pruebas del sistema.

- **Fase de Cierre**

Es esta última fase se puso en marcha la aplicación como resultado de las actividades efectuadas en cada una de las anteriores fases, por lo que la aplicación finalmente estuvo operativa.

f. Resultados

En este apartado se darán a conocer los resultados y análisis del trabajo de titulación en base al desarrollo de los objetivos, en el que se incorporan procesos ordenados sistemáticamente de acuerdo a la metodología de desarrollo, los cuales permitieron realizar interpretaciones específicas de datos, partiendo como eje primordial de las bases teóricas que guiaron el curso del estudio de la problemática planteada. De esta manera los resultados obtenidos son los siguientes:

1. Determinación de la Factibilidad

1.1. Fase de Análisis de Base Teórica

Respecto a la creación de una base de datos es necesario seguir una serie de pasos para llegar a un buen diseño, proceso que se lleva a cabo en cuatro fases:

- **Recolección y análisis de requerimientos**: implica la construcción de un conjunto de requerimientos suficiente por parte del diseñador, acerca del problema o situación que da origen al desarrollo de la base de datos.
- **Diseño conceptual:** con el análisis se hace el diseño conceptual, expresado en un diagrama relacional o en un diagrama entidad relación, que actualmente son los más utilizados, mismos que proporcionan una abstracción de la propuesta de solución.
- **Diseño lógico:** aquí se transforma el diseño conceptual en un diseño lógico, relacionado con la estructura interna de las tablas.
- **Diseño físico**: finalmente, se hace el diseño físico en donde la base de datos encaja en la infraestructura de hardware y software destinada a soportarla.

Centrándonos en la representación conceptual sobre el mini universo que abarca una base de datos, el modelo de datos tiene una función muy importante en donde dicho proceso de abstracción desempeña una función prioritaria.

El modelo de datos, como abstracción del universo de discurso, es el enfoque utilizado para la representación de las entidades y sus características dentro de la base de datos que determina la estructura lógica y de manera fundamental establece el modo

de almacenar, organizar y manipular los datos, y puede ser dividido en 2 grandes tipos:

1. **Modelos lógicos basados en objetos**: los dos más extendidos son el modelo entidad-relación y el orientado a objetos. El modelo entidad-relación (E-R) se basa en una percepción del mundo compuesta por objetos, llamados entidades, y relaciones entre ellos. Las entidades se diferencian unas de otras a través de atributos. El orientado a objetos también se basa en objetos, los cuales contienen valores y métodos, entendidos como órdenes que actúan sobre los valores, en niveles de anidamiento. Los objetos se agrupan en clases, relacionándose mediante el envío de mensajes. Algunos autores definen estos modelos como "modelos semánticos".
2. **Modelos lógicos basados en registros**: el más extendido es el relacional, mientras que los otros dos existentes, jerárquico y de red, se encuentran en retroceso. Estos modelos se usan para especificar la estructura lógica global de la base de datos, estructurándola en registros de formato fijo de varios tipos. El modelo relacional representa los datos y sus relaciones mediante tablas bidimensionales, que contienen datos tomados de los dominios correspondientes. El modelo de red está formado por colecciones de registros, relacionados mediante punteros o ligas en grafos arbitrarios. El modelo jerárquico es similar al de red, pero los registros se organizan como colecciones de árboles. Algunos autores definen estos modelos como "modelos de datos clásicos".

De acuerdo a ello se determina que los modelos de datos tienen como finalidad:

- **Formalizar:** es decir definir formalmente las estructuras permitidas y las restricciones a fin de representar los datos de un Sistema de Información.
- **Diseñar:** el modelo resultante es un elemento básico para el desarrollo de la metodología de diseño de la base de datos.

Por la antes expuesto vemos que existen distintos modelos de datos los cuales permiten que la información pueda ser almacenada y relacionada entre sí. Es así que el origen del trabajo de titulación abordó el proceso de diseño de una Base de Datos introduciendo conceptos cotidianos que sean fáciles de entender por cualquier usuario inexperto, proponiendo la utilización del Modelo Entidad Relación como una representación conceptual del esquema de una base de datos de información y que

genere un Diccionario de Datos. Esta metodología es un esquema que permite una identificación inequívoca de las entidades, relaciones y atributos, de manera que puedan ser satisfechas todas las operaciones asociadas a la utilización del sistema y las reglas para su funcionamiento eficiente.

Como parte complementaria en la realización del determinado modelo de datos se ha establecido generar un diccionario de datos que contenga las características lógicas de los datos que se van a utilizar en el sistema que se diseña, incluyendo nombres, descripción, operaciones, contenido. Este diccionario se desarrolla durante el análisis de flujo de datos y su contenido guarda los detalles y descripción de todos estos elementos.

Así mismo parte importante para llevar a efecto el diseño de un diagrama entidad relación son la herramientas, mismas que no siempre están al alcance y en su mayoría por licenciamiento y es aquí donde las aplicaciones web gratuitas tienen su gran punto a favor para representar gráficamente una idea o un proyecto que requiera el modelado de datos, por lo que se ha determinado factible la realización del tema, "Herramienta de Diseño para la Elaboración de Modelos de Datos del Diseño Conceptual".

1.2. Software

A continuación se describen cada una de las herramientas empleadas para el desarrollo de la aplicación.

1.2.1. Sistema Operativo

Una de las decisiones más importantes a tomar a la hora de desarrollar la aplicación fue la elección del sistema operativo, ya que se optó por emplear Linux debido a su filosofía de software libre y donde la distribución instalada fue la de Debian7.0, esto por su entorno robusto, estable y sencillo.

La elección de este Sistema Operativo reside en dos aspectos la primera radica en que es libre, hecho que elimina completamente los costes de adquisición y renovación de licencias de uso, además de incluir un completo conjunto de aplicaciones que cubren prácticamente todas las necesidades de los usuarios y el segundo aspecto el consumo reducido de recursos para su operatividad.

1.2.2. Lenguaje de Programación

Para el desarrollo de la codificación de la aplicación y teniendo en cuenta que es una aplicación con entorno web se emplearon los siguientes:

- **PHP:** se lo utilizo para la creación del sitio web. PHP es un lenguaje de script interpretado y ejecutado en el lado del servidor, embebidas en páginas HTML, no necesita ser compilado para ejecutarse además de ser un lenguaje multiplataforma con capacidad de conexión con la mayoría de los manejadores de base de datos en este caso SQLite 3.

- **HTML5:** actualmente es el lenguaje hipermedia más aceptado, por tanto es uno de los lenguajes de programación web más importante y uno de los más usados. El Hyper Text Markup Language (HTML) permitió estructurar los documentos y mostrarlos en forma de hipertexto, brindando información correspondiente relacionada con el contenido a mostrar, organizando los elementos lógicos, tales como: cabecera, cuerpo, pie, etc., definiendo las funciones que deben ejecutar sobre dichos elementos.

- **Javascript:** Este es un lenguaje interpretado, no requiere compilación y se lo integro dentro de las páginas web.

- **XML:** a través de ello se pudo representar los artefactos del diagrama entidad-relación en la web, estableciendo un esquema, de modo que esta información pueda ser almacenada, transmitida, procesada, visualizada e impresa, con la finalidad de poder generar el Diccionario de datos de determinado diagrama.

- **Canvas:** El elemento canvas puede definirse como un entorno para crear imágenes dinámicas, su empleo es simple en donde se tiene que especificar sus dimensiones y cada acción, todo ello en combinación con JavaScript para facilitar la creación de cada elemento del diagrama entidad-relación.

2. Diseño e Implementación del Sistema

2.1. Planificación de Requisitos

En esta fase se realizó el análisis de requisitos del sistema con el objetivo de identificar y documentar las necesidades funcionales que deberán ser soportadas por el sistema a desarrollar. Para ello, se identificaron los requisitos que han de satisfacer el nuevo sistema mediante la experiencia propia como estudiante de la carrera de Ingeniería en Sistemas y de la observación directa. Al identificar los requisitos se procederá a transformarlos en diseño y el diseño implementado en código, lo cual proporciona un punto de referencia para validar el sistema final que compruebe que se ajusta a las necesidades del usuario.

2.1.1. Requerimientos Funcionales

El sistema permitirá al usuario:

TABLA II. REQUERIMIENTOS FUNCIONALES

Código	Descripción	Categoría
RF001	Ingresar al sistema a través de un usuario y contraseña	Oculto
RF002	Crear, modificar y eliminar un diagrama	Evidente
RF003	Seleccionar del panel de elementos los artefactos para crear el diagrama.	Evidente
RF004	Definir los atributos de una colección de entidades	Evidente
RF005	Definir la dinámica de las relaciones entre entidades de acuerdo el tipo de relación que le corresponde.	Evidente
RF006	Determinar las interrelaciones (cardinalidad).	Evidente
RF007	Realizar búsquedas de un diagrama determinado.	Evidente
RF008	Exportar el diseño del diagrama a formato pdf, png entre otros.	Evidente
RF009	Usar formas personalizadas en sentencias XML.	Evidente
RF010	Refinar y validar el diseño del diagrama antes de su ejecución.	Oculto
RF011	Generar un diccionario de datos que almacene la información del diagrama entidad-relación.	Evidente

2.1.2. Requerimientos No Funcionales

El sistema permitirá:

TABLA III. REQUERIMIENTOS NO FUNCIONALES

Código	Descripción	Categoría
RFN001	Contar con una interfaz gráfica de administración y de operación, sencilla y amigable.	Evidente
RFN002	Proporcionar ayuda acerca de la utilización de la aplicación	Evidente
RFN003	Utilizar el estándar HTML para garantizar que pueda ser interpretado por cualquiera de los navegadores existentes.	Evidente
RFN004	Acceder a través de un ambiente Web.	Evidente

2.1.3. Lista de Entidades a desarrollar

Tabla IV. LISTA DE ENTIDADES

Entidad	Descripción
Usuario	Entidad que hará uso del sistema a desarrollar.
Editor	Entidad que permite crear, modificar y eliminar los artefactos del diagrama entidad-relación, en base al panel gráfico.
Diagrama	Permite establecer la representación gráfica en base a sus atributos, relaciones y cardinalidad, es decir de acuerdo a las restricciones que conforman el diagrama entidad-relación.
DiccionarioDatos	Permite leer y almacenar las características lógicas del diagrama entidad-relación a través de una estructura definida en un archivo XML.

2.1.4. Diagramas de Actividad.

2.1.4.1. Descripción del diagrama de actividad

A continuación se describe el diagrama de actividad que se emplea para:

- **Login de usuario**

1. El usuario ingresa correo y clave para poder acceder al sistema.
2. El sistema valida si los datos ingresados son correctos, de lo contrario debe elegir Registrarse para ingresar la información y crearse el usuario de ingreso.
3. El usuario accede al sistema.

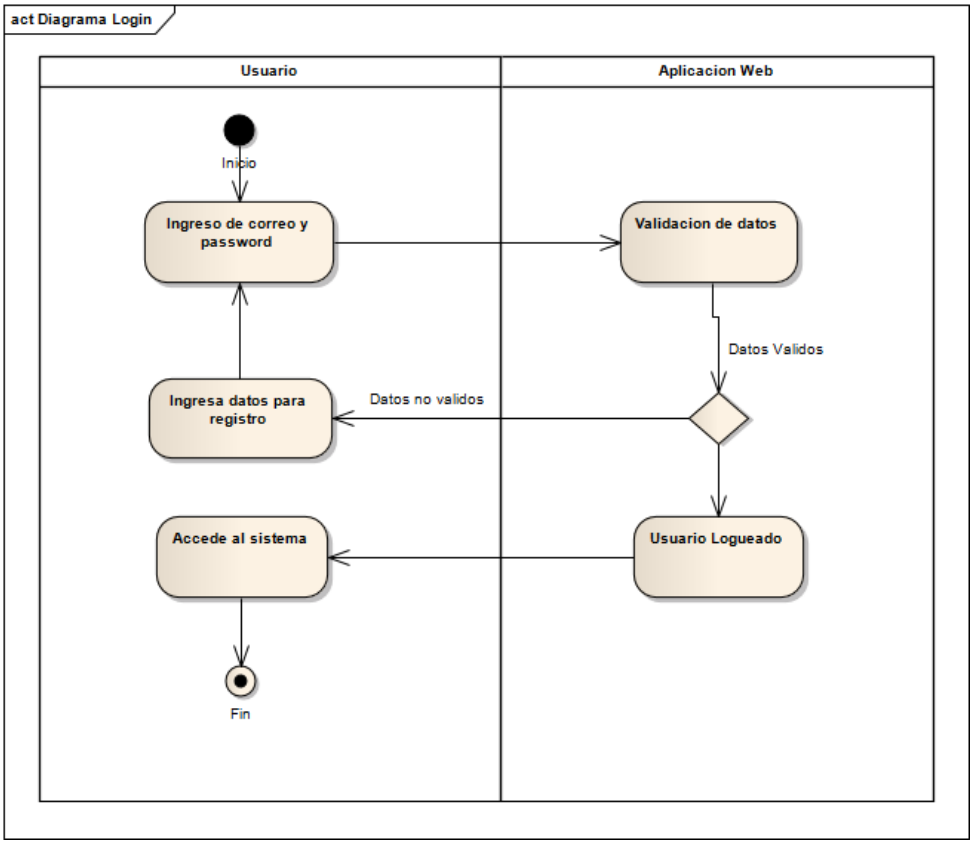


Figura 13. Diagrama de actividades Login de Usuario

- **Modelar Diagrama Entidad-relación**

1. El usuario selecciona nuevo diagrama donde se le asigna un id de diagrama para identificarlo.
2. Se selecciona del panel de artefactos y se le asigna una etiqueta o nombre.
3. El sistema identifica el tipo de figura para de acuerdo a ello crear el correspondiente arreglo de determinada artefacto.
4. El usuario define las relaciones con su respectiva cardinalidad.
5. Una vez finalizado el diagrama el usuario lo guarda, donde automáticamente se crea un archivo xml, mismo que posteriormente será utilizado para generar el diccionario de datos.

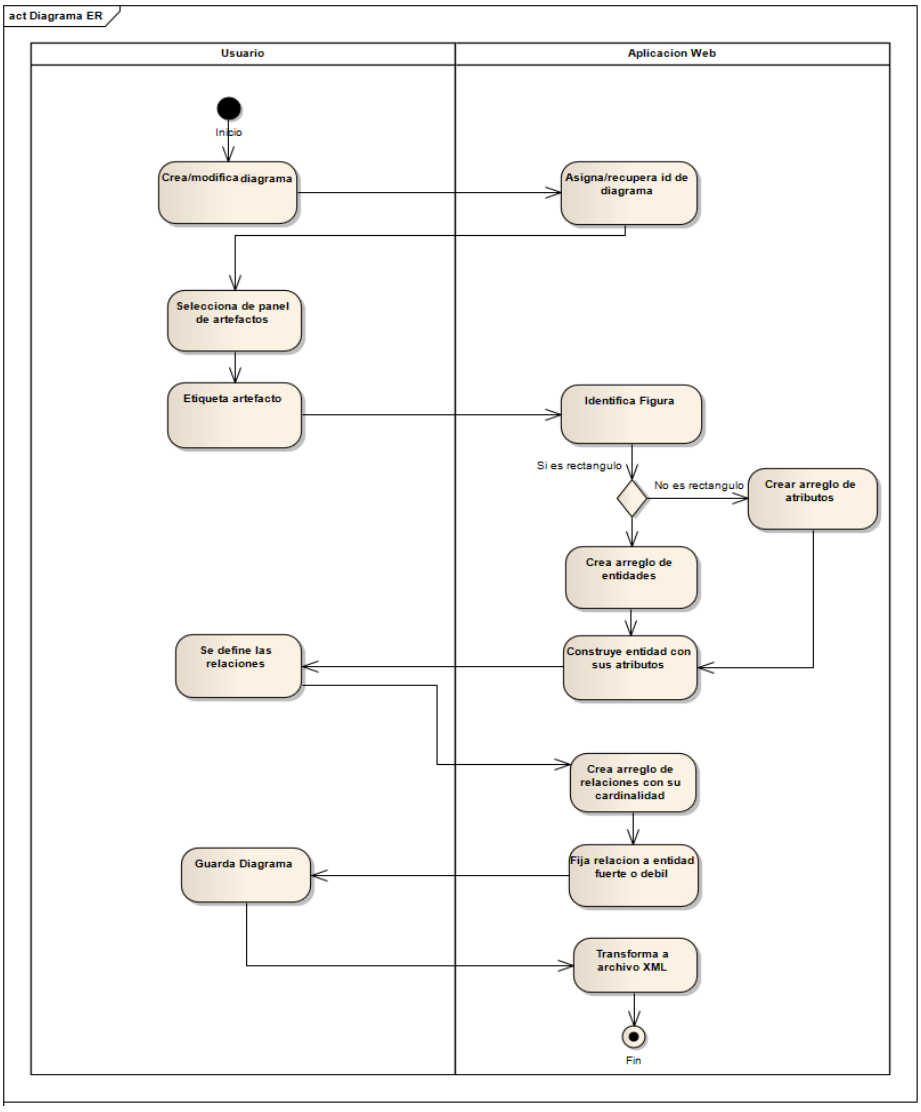


Figura 14. Modelar Diagrama Entidad Relación

- **Generar Diccionario de Datos**

1. El usuario revisa si ha finalizado diseño del diagrama y selecciona guardar.
2. El usuario selecciona Generar Diccionario de Datos, donde se obtiene el id del diagrama y el archivo XML correspondiente, internamente estos datos se fijan en tablas para su presentación.
3. El usuario obtiene el diccionario de datos en formato pdf. Donde se muestran todas las entidades, atributos y relaciones de un determinado diagrama.

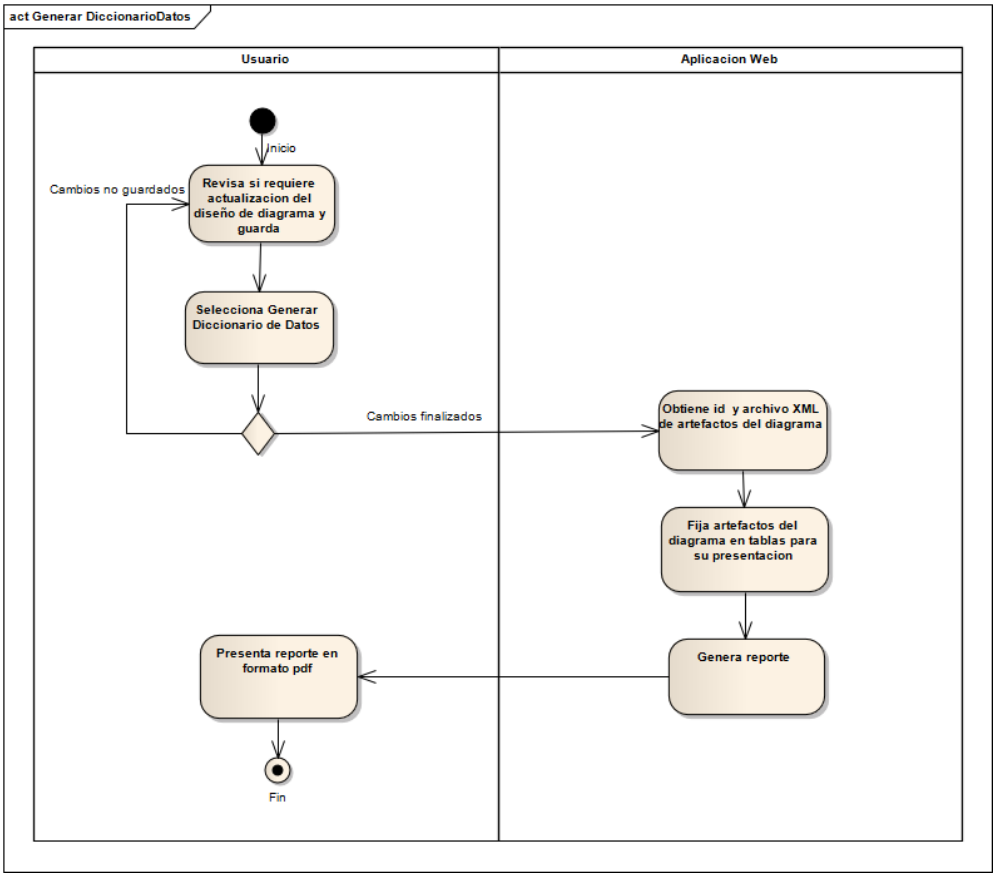


Figura 15. Generar Diccionario de Datos

2.1.5. Flujo de Ventanas principales del sistema

Ya con las lista de requerimientos se procedió a realizar los prototipos rápidos de pantalla iniciales. Para el desarrollo de las interfaces del sistema se hizo necesario el empleo de tres hilos de ejecución de forma paralela. Así tenemos los siguientes:

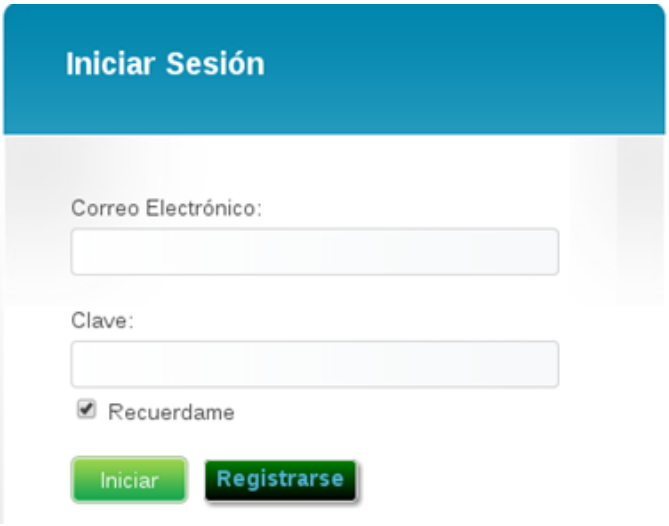


Figura 16. Acceso al Sistema

- El primer hilo se lo empleó para el control de acceso al sistema donde se controlan todos los flujos alternos ocasionados debido a los intentos de acceso, ya que se hace necesario identificar al usuario para que pueda hacer uso del editor e iniciar con el diseño del diagrama entidad-relación.

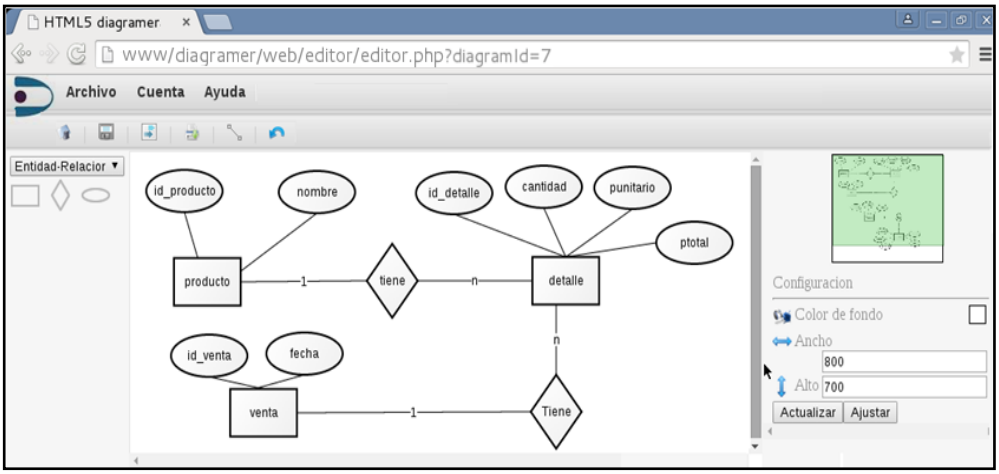


Figura 17. Editor de Diagramer

- La ejecución de segundo hilo, es el encargado de llevar a cabo la construcción de un diagrama entidad-relación, en este proceso intervienen los métodos Listar Entidades, Agregar Entidad, Listar Propiedades, Agregar Relación, Listar Relaciones, Fijar Relación y el método Validar Datos, cabe recalcar que para la mayoría de métodos se empleó javascript, que es donde se identifica de qué tipo de figura se trata.

- En cuanto a la ejecución del tercer hilo una vez ya establecido el diseño del diagrama entidad-relación se procedió a generar el diccionario de datos, donde para dicha implementación se hizo a través de un archivo XML, en el que se definió la estructura que conformará el diccionario de datos además se estableció como consideración particular que para poderlo generarlo se debe primeramente guardar el diagrama, como resultado de ello se obtienen los datos y descripción por cada uno de los elementos que componen el diagrama.

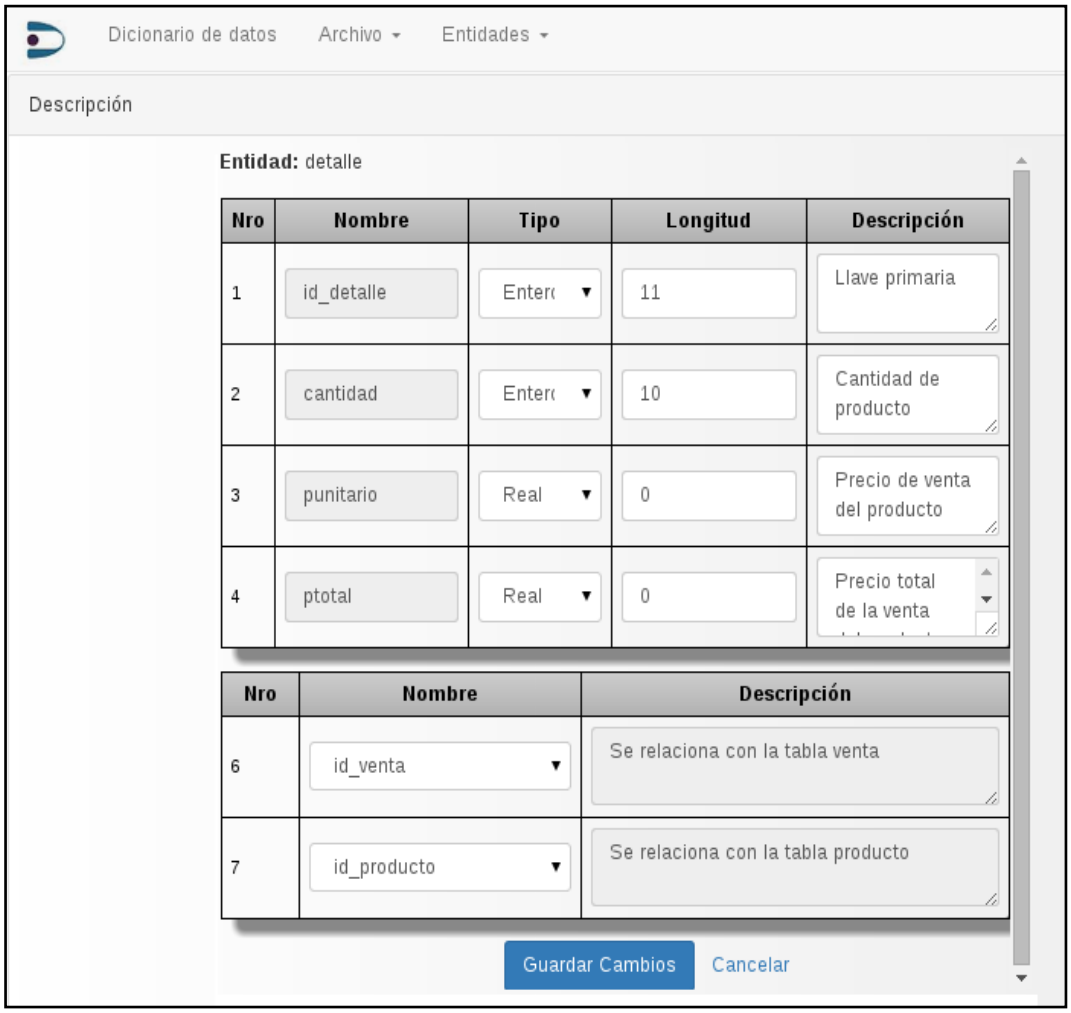
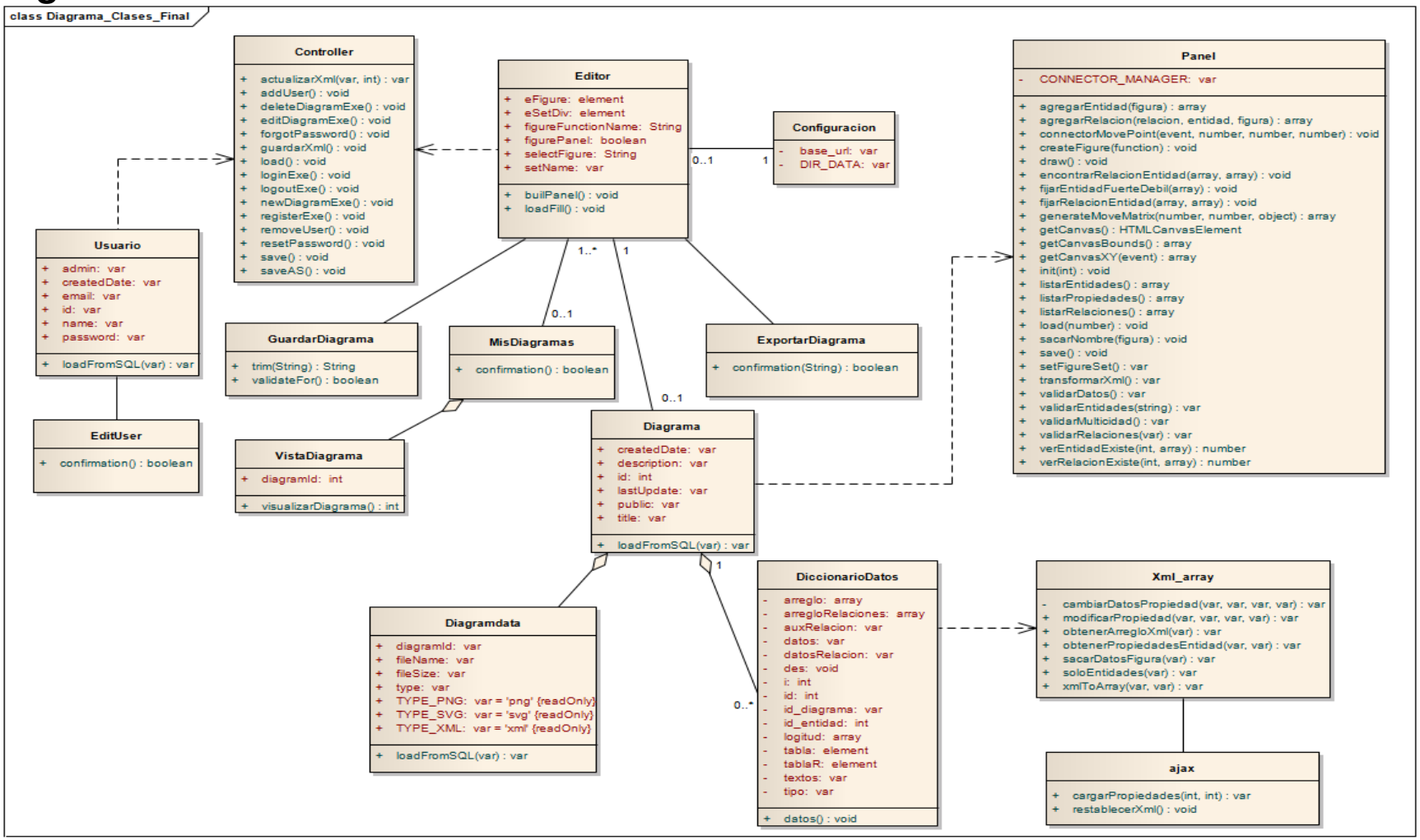


Figura 18. Diccionario de Datos

22 Diseño del Usuario
221 Diagrama de Clases Final



3. Construcción Rápida

En esta fase de la metodología se realizó la implementación del código para la creación de los artefactos del diagrama entidad-relación, cabe recalcar que para la mayoría de los métodos se empleó el archivo javascript CONNECTOR_MANAGER, que es donde se identifica de qué tipo de figura se trata, a continuación se describen los procedimientos empleados.

3.1. Construcción del método Listar Entidades

Permite listar todas entidades del diagrama, este proceso se realiza teniendo en cuenta que solamente las figuras en forma de rectángulo son una entidad.

TABLA V. IMPLEMENTACIÓN DEL MÉTODO LISTAR ENTIDADES

Implementación de Método Listar Entidades

```
functionlistarEntidades() {
var conectores = CONNECTOR_MANAGER.imprimir();
var entidades = [];
var cont = 0;
   for (var i = 0; i <conectores.length; i++){
        var con = CONNECTOR_MANAGER.connectorGetById(conectores[i].id);
        var figP = STACK.figureGetAsFirstFigureForConnector(con.id);
        var figU = STACK.figureGetAsSecondFigureForConnector(con.id);
if (figP.name == 'Rectangle') {
        var indice = verEntidadExiste(entidades, figP.id);
if (indice == -1) {
        var datos = agregarEntidad(figP);
         entidades[cont] = datos;
        cont++;
         }
if (figU.name == 'Rectangle') {
        var indice = verEntidadExiste(entidades, figU.id);
```

```
if (indice == -1) {
        var datos = agregarEntidad(figU);
        entidades[cont] = datos;
        cont++;
      }
   }
return entidades;
}
```

3.2. Construcción del método Agregar Entidad

En este método se crea un arreglo de entidades, teniendo en cuenta que recibe un parámetro de tipo figura a la cual la se la trabaja en un arreglo de cuatro campos; un id de figura, el nombre de la entidad, propiedades y relaciones.

TABLA VI. IMPLEMENTACIÓN DEL MÉTODO AGREGAR ENTIDAD

Implementación del método Agregar Entidad

```
/**
 * Genera las propiedades de las entidades
 * @ param {type} figura Figura
 * @ returns {Array} lista de propiedades
 */
functionagregarEntidad(figura) {
var datos = [];
   datos[0] = figura.id;//para el id
   datos[1] = sacarNombre(figura);//para el nombre de la entidad
   datos[2] = "";//para las propiedades
   datos[3] = "";//para las relaciones
return datos;
}
```

3.3. Construcción del método Listar Propiedades

Permite listar las entidades con sus respectivas propiedades, recibiendo como parámetro un arreglo de entidades

TABLA VII. IMPLEMENTACIÓN DEL MÉTODO PROPIEDADES

Implementación del método Listar Propiedades

```
functionlistarPropiedades(entidades) {
var conectores = CONNECTOR_MANAGER.imprimir();
for (var i = 0; i <conectores.length; i++) {
        var con = CONNECTOR_MANAGER.connectorGetById(conectores[i].id);
        var figP = STACK.figureGetAsFirstFigureForConnector(con.id);
        var figU = STACK.figureGetAsSecondFigureForConnector(con.id);
        var indice = -1;
        var tipo = "";
             //alert("aquiesta 1");
if (figP.name == 'Rectangle') {
        indice = verEntidadExiste(entidades, figP.id);
        tipo = "U";//Uno
     } else if (figU.name == 'Rectangle') {
        indice = verEntidadExiste(entidades, figU.id);
        tipo = "D";//dos
     }
if (tipo == "U") {
        if (figU.name == 'Ellipse') {
                var datosAux = entidades[indice];
                var propiedades = datosAux[2];
         propiedades += sacraNombre(figU) + ";" + figU.id + ":";
                datosAux[2] = propiedades;
               entidades[indice] = datosAux;
        }
     }
```

```
      else if (tipo == "D"){
         if (figP.name == 'Ellipse') {
         var datosAux = entidades[indice];
         var propiedades = datosAux[2];
         propiedades += sacraNombre(figP) + ";" + figP.id + ":";
         datosAux[2] = propiedades;
         entidades[indice] = datosAux;
}
      }
   }
return entidades;
}
```

3.4. Construcción del método Agregar Relación

En este método se crea una relación, y se agrega a una determinada entidad de acuerdo a su id.

Tabla VIII. IMPLEMENTACIÓN DEL MÉTODOAGREGAR RELACIÓN

Implementación del método Agregar Relación

```
/**
 * Crea una relacion
 * @param {type} figura La relacion
 * @param {type} entidad La entidad
 * @param {type} relacion El tipo de relacion
 * @returns {Array} El arreglo de relacion
 */
functionagregarRelacion(figura, entidad, relacion) {
var datos = [];
   datos[0] = figura.id;//para el id
   datos[1] = sacraNombre(entidad) + ":" + entidad.id;//para el nombre de la entidad
   datos[2] = relacion;//para las propiedades
```

```
    datos[3] = "";//para la segunda relacion
    datos[4] = "";//para la segunda relacion
    datos[5] = "";//entidad debil
return datos;
}
```

3.5. Construcción del método Listar Relaciones

Permite crear un arreglo de todas las relaciones para posteriormente fijarlas a una determinada figura.

TABLA IX. IMPLEMENTACIÓN DEL MÉTODOLISTAR RELACIONES

Implementación del método Listar Relaciones

```
/**
 * Construye una lista con relaciones
 * @ returns {Array} lista de relaciones
 */
functionlistarRelaciones() {
var conectores = CONNECTOR_MANAGER.imprimir();
var listaRelaciones = [];
var cont = 0;
   for (var i = 0; i <conectores.length; i++){
         var con = CONNECTOR_MANAGER.connectorGetById(conectores[i].id);
         var figPR = STACK.figureGetAsFirstFigureForConnector(con.id);
         var figUR = STACK.figureGetAsSecondFigureForConnector(con.id);
         var indice = -1;
         var tipo = "";
         var index = -1;
         var linea = conectores[i];
      if (figPR.name == 'Diamond') {
         index = verRelacionExiste(listaRelaciones, figPR.id);
```

```
        if (index == -1) {
                listaRelaciones[cont] = agregarRelacion(figPR, figUR,
                linea.middleText.str.trim() + ":" + linea.middleText.underlined);
                cont++;
        } else {
       var relacion = listaRelaciones[index];
     //datos[1]=sacraNombre(entidad)+":"+entidad.id;//para el nombre de la entidad
       //datos[2]=relacion;//para las propiedades
       relacion[3] = sacraNombre(figUR) + ":" + figUR.id;//para la segunda relacion
        relacion[4] = linea.middleText.str.trim() + ":" +
        linea.middleText.underlined;//para la segunda relacion
        listaRelaciones[index] = relacion;
        }
      } elseif (figUR.name == 'Diamond') {
       index = verRelacionExiste(listaRelaciones, figUR.id);
        if (index == -1) {
        listaRelaciones[cont] = agregarRelacion(figUR, figPR,
        linea.middleText.str.trim() + ":" + linea.middleText.underlined);
        cont++;
        } else {
        var relacion = listaRelaciones[index];
        //datos[1]=sacraNombre(entidad)+":"+entidad.id;//para el nombre de la entidad
        //datos[2]=relacion;//para las propiedades
        relacion[3] = sacraNombre(figUR) + ":" + figUR.id;//para la segunda relacion
        relacion[4] = linea.middleText.str.trim() + ":" +
        linea.middleText.underlined;//para la segunda relacion
        listaRelaciones[index] = relacion;
        }
      }
   }
returnlistaRelaciones;
}
```

3.6. Construcción del método Fijar Relación

Tabla X. IMPLEMENTACIÓN DEL MÉTODOFIJAR RELACION-ENTIDAD

Implementación del método Fijar Relación-Entidad

```
/**
 * Fija las relaciones dentro de las entidades
 * @param {type} listaRelaciones Lista de relaciones ya generadas
 * @param {type} listaEntidad La lista de entidades
 * @returns {@var;entidad} La lista de entidades ya actualizadas
 */
functionfijarRelacionEntidad(listaRelaciones, listaEntidad) {
for (var i = 0; i < listaEntidad.length; i++) {
        var datos = listaEntidad[i];
        var entidad = encontrarRelacionEntidad(listaRelaciones, datos);
        listaEntidad[i] = entidad;
   }
returnlistaEntidad;
   //datos[3];//relaciones
}
```

3.7. Construcción del método Validar Datos

Este método permite realizar una validación de todo el diagrama teniendo en cuenta la lógica de su modelamiento.

TABLA XI. IMPLEMENTACIÓN DEL MÉTODOVALIDAR DATOS

Implementación del método Validar Datos

```
functionvalidarDatos() {
var conectores = CONNECTOR_MANAGER.imprimir();
var resultado = "OK";
var figuras = STACK.figures;
   if (conectores.length> 0) {
      if ((figuras.length - 1) == conectores.length) {
resultado = validarEntidades(conectores);//validar entidades
if(resultado=='OK'){
         resultado = validarRelaciones(conectores);  //validar relaciones
if(resultado=='OK'){
          resultado = validarMulticidad(); //validar multicidad
            }
         }
      } else {
         resultado = "Todos los elementos deben estar conectados de acuerdo a su
función";
      }
   } else {
      resultado = "Los elementos deben ir relacionados";
}
return resultado;
}
functionvalidarEntidades(conectores) {
var resultado = "OK";
for (var i = 0; i <conectores.length; i++) {
        var con = CONNECTOR_MANAGER.connectorGetById(conectores[i].id);
        var figP = null;
        var figU = null;
try {
        figP = STACK.figureGetAsFirstFigureForConnector(con.id);
        figU = STACK.figureGetAsSecondFigureForConnector(con.id);
} catch (e) {
        resultado = "La conexiones deben estar bien relacionadas entre elementos "
+ e.name;
break;
}
      if (figP != null) {
            if (figU != null) {
```

```
            if (figP.name == 'Rectangle' && figU.name == 'Rectangle') {
             resultado = "Las conexiones deben ser entre una entidad y un atributo o
             una entidad y una relación";
            break;
            }
         } else {
            resultado = "Las relaciones deben ir acompañadas otro elemento";
            break;
         }
      } else {
         resultado = "Las relaciones deben ir acompañadas otro elemento";
         break;
      }
   }
return resultado;
}
```

3.8. Construcción del archivo XML

Para la implementación del diccionario de datos se hace uso de un archivo XML, se lo definió de la siguiente manera.

TABLA XII. DISEÑO DE ARCHIVO XML

Diseño del archivo XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<diccionario>
        <entidad>
                <id>0</id>
                <nombre>Text</nombre>
                <propiedad>
                        <nombre>Text;1</nombre>
                        <dato>caracter</dato>
                        <longitud>0</longitud>
                        <descripcion>N</descripcion>
                </propiedad>
                <relaciones></relaciones>
        </entidad>
</diccionario>
```

TABLA XIII. IMPLEMENTACIÓN DEL MÉTODOTRANSFORMAR A XML

Implementación del Método Transformar a XML

```
function _transformarXml() {
var lista = listarEntidades();
lista = listarPropiedades(lista);
var listaRelaciones = listarRelaciones();
listaRelaciones = fijarEntidadFuerteDebil(listaRelaciones);
lista = fijarRelacionEntidad(listaRelaciones, lista);
var xml = '<?xml version="1.0" encoding="UTF-8" ?>\n';
xml += "<diccionario>\n";
for (var i = 0; i < lista.length; i++) {
        var datos = lista[i];
        xml += "<entidad>\n";
        xml += "<id>" + datos[0] + "</id>\n";
        xml += "<nombre>" + datos[1] + "</nombre>\n";
        var propiedad = datos[2];
        varlistaPropiedad = propiedad.split(":");
for (var j = 0; j < listaPropiedad.length - 1; j++) {
        xml += "<propiedad>\n";
        xml += "<nombre>" + listaPropiedad[j] + "</nombre>\n";
        xml += "<dato>caracter</dato>\n";
        xml += "<longitud>0</longitud>\n";
        xml += "<descripcion>N</descripcion>\n";
        xml += "</propiedad>\n";
     }
xml += '<relaciones>' + datos[3] + '</relaciones>\n';
xml += "</entidad>\n";
   }
xml += "</diccionario>";
returnxml;
}
```

4. Pruebas

Para la verificación del comportamiento de la aplicación así como también para mantener un orden en la programación se emplearon las siguientes herramientas:

- **JMETER:** es una herramienta de código abierto desarrollada en Java para realizar pruebas de carga y de rendimiento, principalmente de aplicaciones web aunque permite probar otras como SOAP o Mail, también es utilizado para probar el rendimiento de lenguajes dinámicos Web - PHP, Java, consultas, Servidores FTP, etc., simulando una carga pesada en el servidor, grupo de servidores donde analiza el rendimiento general bajo diferentes tipos de carga pesada concurrente.

4.1. Pruebas de Rendimiento

Para ejecutar las pruebas de carga sobre la aplicación web www.diagramer.com.ec,se llevó a cabo la ejecución de JMeter desde el directorio bin, que contiene las siguientes opciones:

- jmeter: Corresponde a los ejecutables de la interface principal para plataformas *nix.
- jmeter-server: Representan los ejecutables para el emulador de Servidor JMeter para plataformas Linux.
- jmeter.properties: Contiene propiedades de arranque para JMeter que son utilizadas por cualquiera de sus ejecutables.
- jmeter.log: Representa los registros ("logs") generados al ejecutar JMeter.
- users.xml: Un archivo XML empleado para definir características de usuarios que serán simulados por JMeter.
- ApacheJMeter.jar: Archivo JAR que contiene las principales clases de JMeter.

Sobre el Plan de Pruebas se añadió un Grupo de Hilos, este representa el número de usuarios que ejecuta el plan de pruebas. Para la primera prueba se lo estableció en 50, de igual manera se efectuaron peticiones HTTP, este proceso se describe a continuación:

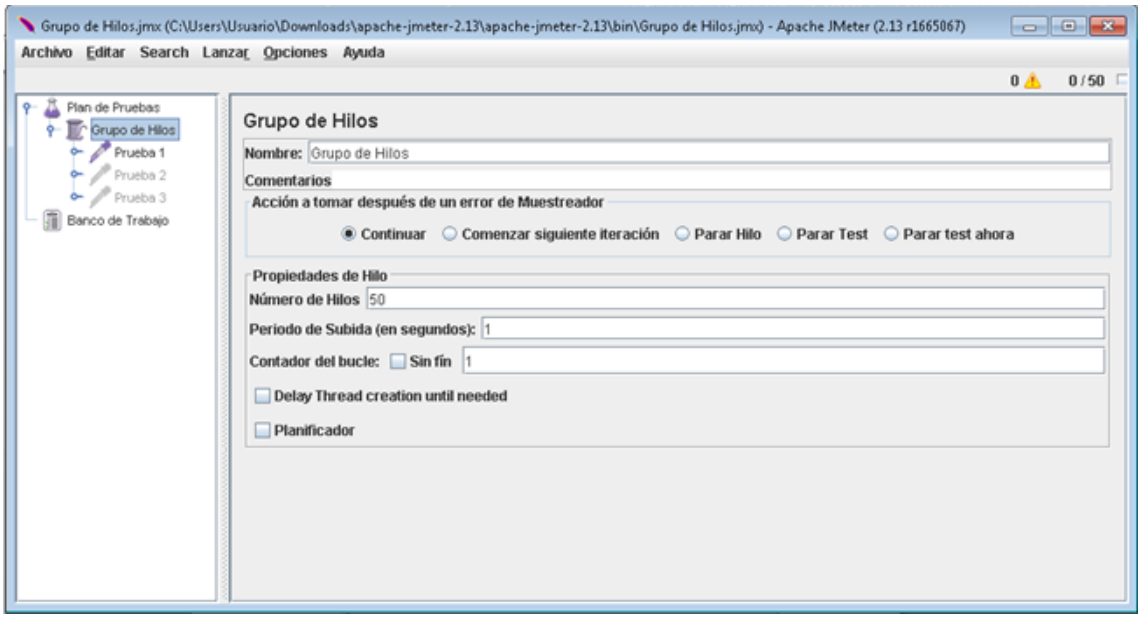


Figura 19. Peticiones HTTP 50 Iteraciones

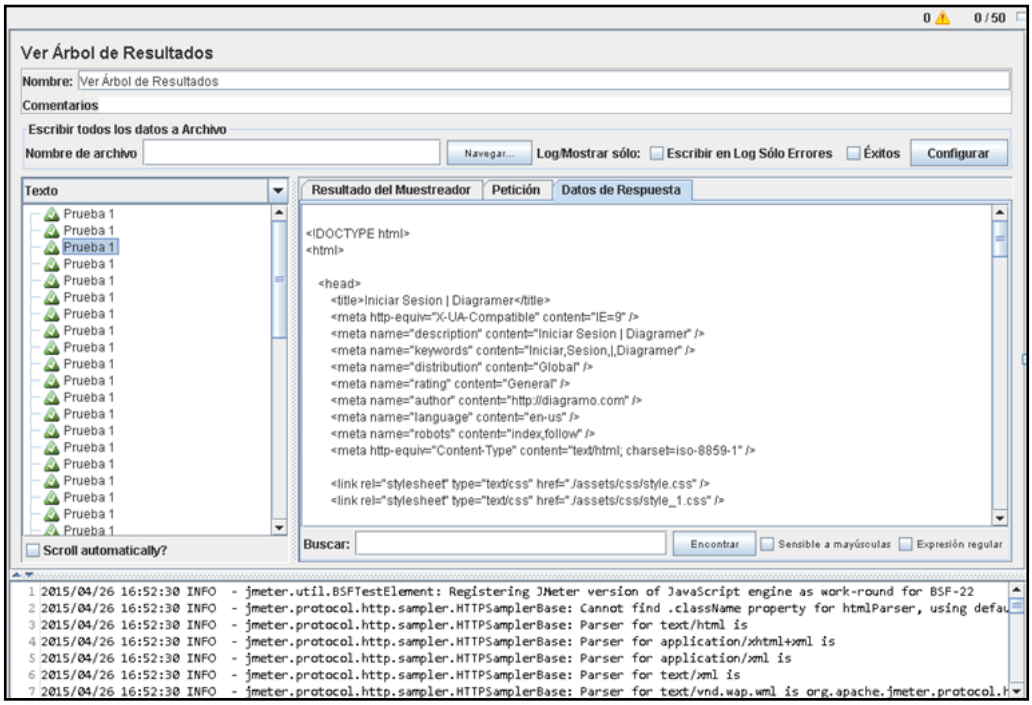


Figura 20. Árbol de Resultados

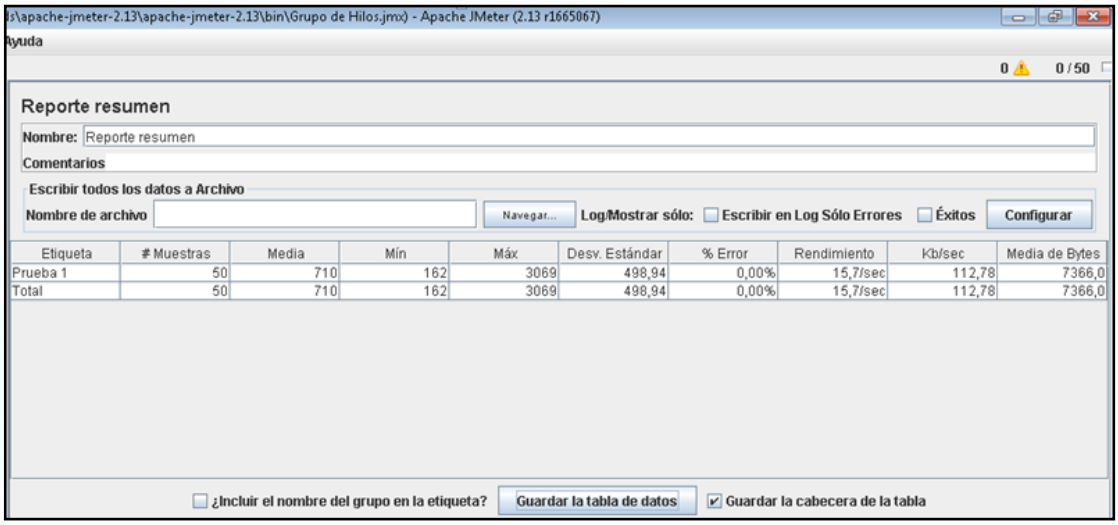


Figura 21. Datos obtenidos por Simulación JMeter

De acuerdo a las Figuras 16 a 18, las pruebas de carga donde se muestra la conducta funcional con una iteración de 50 hilos no muestra porcentaje de error al soportar múltiples solicitudes concurrentes, así como el tiempo de arranque de cada uno de los hilos y duración del test en segundos, además se observa el porcentaje de error en 0%, por lo que a continuación se realiza una prueba de carga con 5000 iteraciones, para seguir comprobando su rendimiento.

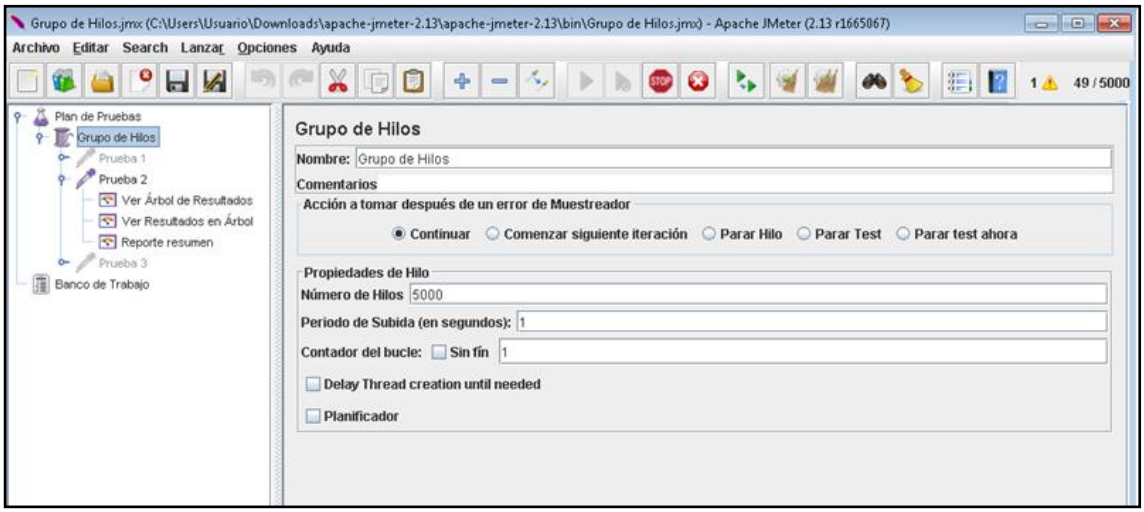
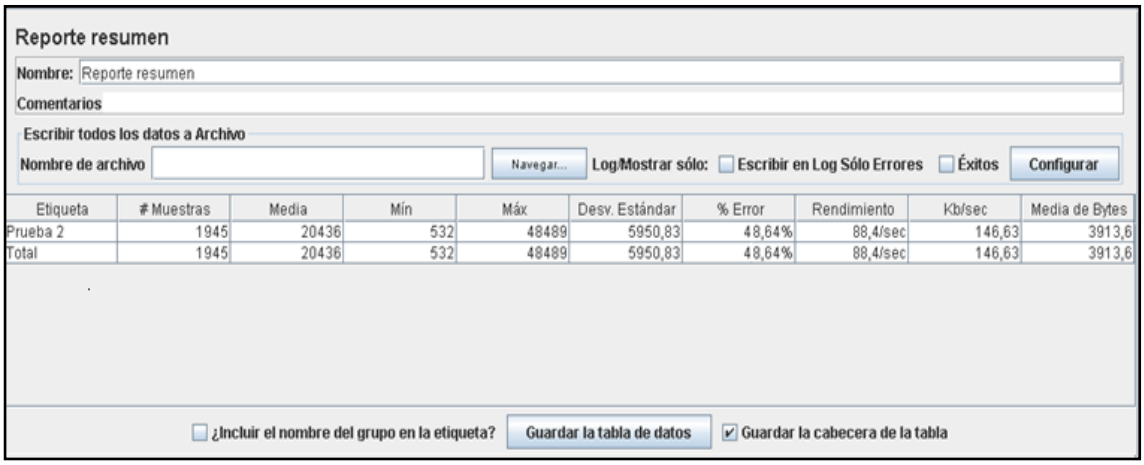


Figura 22. Peticiones HTTP 5000 Iteraciones

En Figura 19. prueba realizada con 5000 iteraciones se puede observar el índice de error que es de 48,64%, así como también el porcentaje de rendimiento que en este caso es de 88,4%, obteniendo estos resultados se puede determinar que son aceptables teniendo en cuenta el número de iteraciones cargadas sobre el servidor, considerándose aceptable en relación a la cantidad de iteraciones efectuadas sobre el mismo y con un tiempo de respuesta estable en estas situaciones de carga.



Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Media de Bytes
Prueba 2	1945	20436	532	48489	5950,83	48,64%	88,4/sec	146,63	3913,6
Total	1945	20436	532	48489	5950,83	48,64%	88,4/sec	146,63	3913,6

Figura 23. Datos Obtenidos con 5000 iteraciones

4.2. Pruebas de Caja Blanca

A través de esta fase se realizaron pruebas individuales del software con el objetivo de encontrar individualidades que la aplicación pueda tener de acuerdo a las diferentes entradas de manera que soporte el ingreso de datos erróneos o inesperados para garantizar la calidad y rendimiento del mismo.

4.2.1. Validación del documento XML utilizando DTD's

Este tipo de prueba requiere de la declaración en el Documento XML, aquí el parámetro !DOCTYPE siempre es utilizado para declarar DTD's, mientras que el elemento que le sigue corresponde al nodo raíz del documento, en este caso corresponde a diccionario, mientras que validador_XML.dtd es el DTD a emplearse; la ubicación del DTD se realizó en el mismo directorio de trabajo que el archivo XML, aunque también es posible indicar

otro directorio de residencia utilizando /(*/slashes/*). A continuación se describe su declaración:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE diccionario SYSTEM "validador_XML.dtd">
```

Hasta este punto si es procesado el documento, la única validación que es llevada a cabo es la de "Your document is correct" y que el DTD exista en el directorio indicado, para que sea llevada a cabo una validación completa "DTD Checked: No syntax error" es necesario pasar unos parámetros de configuración al parser.

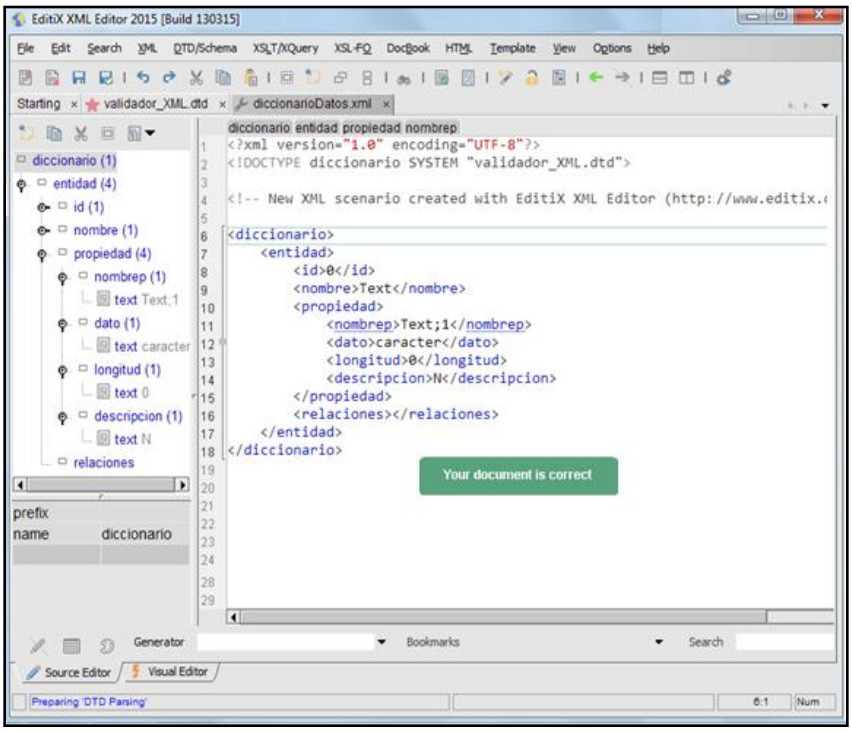


Figura 24. Validación de Archivo XML

En esta figura se evalúan las expresiones y la sintaxis correcta del archivo, obteniendo como resultado el mensaje documento correcto.

Finalmente debe ser definido en la parte superior del archivo XML el DTD que será empleado para poder ejecutar la opción de Check e internamente se realiza un parser del archivo, a continuación se presentan los resultados de la prueba efectuada:

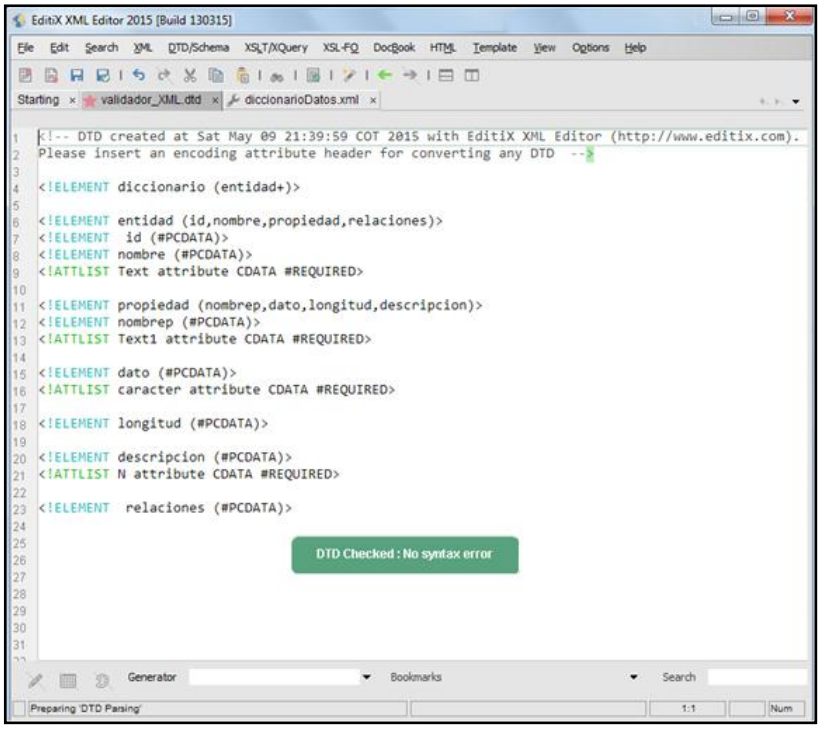


Figura 25. Validación DTC

Estas pruebas fueron realizadas con el propósito de encontrar los posibles fallos de implementación, calidad o accesibilidad, probando el comportamiento del mismo.

g. Discusión

1. Desarrollo de la Propuesta alternativa

En este apartado se analiza y discute los resultados obtenidos de la sección anterior, con el propósito de evaluar el cumplimiento de los objetivos planteados del presente trabajo de titulación, entre los cuales tenemos los siguientes:

- **Objetivo Especifico 1:** Analizar los distintos modelos de datos del diseño conceptual que permita describir el comportamiento relacional al diseñar una base de datos.

Con este objetivo y de acuerdo a la revisión literaria se ha podido establecer que los modelos de datos del diseño conceptual están orientados a la descripción de estructuras de datos y restricciones de integridad, usándose fundamentalmente durante la etapa de análisis de un problema dado, mismos que están encaminados a representar los elementos que intervienen así como también definir sus relaciones. De acuerdo a ello para el modelado de datos se optó por seleccionar el Modelo Entidad-Relación que involucra todos los conceptos de una problemática.

- **Objetivo Especifico 2:** Analizar las entidades y relaciones para definir sus atributos, identificadores y cardinalidad.

Este objetivo permitió identificarlos elementos que intervienen en el diagrama Entidad-Relación, así tenemos, las entidades, donde comúnmente es un sustantivo y se establecen de dos tipos, entidad débil y entidad fuerte, para en base a ello definir las relaciones y su cardinalidad en cuyo aspecto se han establecido de uno a uno y de uno a varios, de la misma forma determinar sus atributos los cuales brindan información acerca de una entidad, estos son de tipo texto, numéricos, lógicos, etc., así como también su correspondiente clave primaria que identifica inequívocamente un solo atributo no permitiendo que se repita en una misma entidad. Todo ello permitió entender la estructura lógica de los datos y como se relacionan entre ellos.

- **Objetivo Especifico 3:** Construir un panel de elementos gráfico que permita representar los elementos de un diagrama conceptual o lógico.

Para alcanzar este objetivo se trabajó con archivos javascript, el lenguaje php y canvas, donde se desarrollaron las clases y métodos principales que permitieron crear todos los artefactos que representen el diagrama entidad-relación, todo en conjunto con un entorno web.

- **Objetivo Especifico 4:** Crear un diccionario de datos para almacenar la información del diagrama entidad-relación que contenga las características lógicas de los datos que se van a utilizar en el sistema.

El cumplimiento de este objetivo contribuyó a entender que significa cada término de datos del diagrama entidad-relación, en base a la descripción de todos los elementos que forman parte del flujo de datos de todo el sistema, constituido en un documento de referencia acerca de los datos manejados, por lo que proporcionar información y es un punto inicial que detalla también la lógica del diagrama.

2. Valoración técnica económica ambiental

2.1. Valoración Técnica – Económica

2.1.1. Recursos Humanos

El trabajo de titulación fue establecido para que sea desarrollado para un único investigador y bajo la asesoría de un docente, donde a continuación se detalla:

Tabla XIV. RECURSOS HUMANOS

RECURSOS HUMANOS			
Descripción	**Valor Hora**	**Número Horas**	**Costo**
Desarrolladores: ▪ Rocío Tene	$ 4.00	1000	$ 4000.00
Director de Tesis: ▪ Ing. Alex Padilla	$ 0.00	----------	$ 0.00
Sub-Total :			**$ 4000.00**

2.1.2. Recursos Materiales

Para el desarrollo del trabajo de titulación se emplearon diversos suministros de oficina, descritos a continuación:

Tabla XV. RECURSOS MATERIALES

Descripción	Cantidad	Precio Unitario	Precio Total
▪ Impresiones	600	$ 0.15	$ 90.00
▪ Copias	5	$ 0.05	$ 5.00
▪ Anillados	6	$ 1.30	$ 7.80
▪ Empastado	3	$ 7.00	$ 21.00
▪ CD´S	5	$ 0.50	$ 2.50
▪ Flash Memory	1	$ 10.00	$ 10.00
Sub-Total :			**$ 136.30**

2.1.3. Recursos Técnicos

Tabla XVI. RECURSOS TÉCNICOS

Descripción	Cantidad	Precio Unitario	Costo
HARDWARE			
▪ Laptop	1	$ 600.00	$ 500.00
▪ Impresora (Copiadora)	1	$ 100.00	$ 100.00
SOFTWARE			
▪ Javascript	----	$ 0.00	$ 0.00
▪ PHP	----	$ 0.00	$ 0.00
▪ Linux (Debian)	----	$ 0.00	$ 0.00
COMUNICACIONES			
▪ Internet	14(meses)	$ 20.05	$ 280.70
Sub-Total :			**$ 880.70**

2.1.4. Servicios Básicos

Tabla XVII. SERVICIOS BÁSICOS

Descripción	Precio Unitario	Costo
▪ Teléfono	$ 8.00	$ 112.00
▪ Transporte	$	$ 100.00
Sub-Total :		**$ 212.00**

El presupuesto total que se empleó para el desarrollo de esta investigación se detalla a continuación:

Tabla XVIII. PRESUPUESTO TOTAL

Descripción		Total
▪ Recursos Humanos		$ 4000.00
▪ Recursos Materiales		$ 136.30
▪ Recursos Técnicos		$ 880.70
▪ Servicios Básicos		$ 212.00
	Sub-Total :	$ 5229.00
	Imprevistos 5%	$ 261.45
	Total	$ 5490.45

2.2. Valoración Ambiental

El trabajo de titulación denominado "Herramienta de Diseño para la Elaboración de Modelos de Datos del Diseño Conceptual", pretende desarrollar y proveer al estudiante de una metodología para el diseño conceptual de base de datos que facilite su comprensión, teniendo como base las técnicas de diseño para llegar a una representación que modele un caso del mundo real, además de proponer un diseño consistente a través de una herramienta web como parte del proceso tecnológico que implica su implementación, buscando mediante la aplicación de los conceptos y avances científicos respecto a las

nuevas tecnologías encontrar la forma de facilitar que el proceso de diseño conceptual de base de datos se lleve de forma estructurada y que se lo considere como un proceso estable con métodos y técnicas propios.

h. Conclusiones

Al finalizar el proceso de desarrollo del sistema denominado "Herramienta de Diseño para la Elaboración de Modelos de Datos del Diseño Conceptual", se concluye lo siguiente:

- En la construcción del diagrama entidad-relación fue fundamental definir la estructura del diccionario de datos mediante un archivo XML, donde se especificó las diferentes etiquetas en base a los artefactos que componen el diagrama, para así poder generar el diccionario de datos y almacenar su información.

- El diccionario de datos permitió presentar las características lógicas del diagrama entidad-relación donde también se almacena dicha información, estableciendo así una estructura general a nivel conceptual y de visión de determinado diagrama.

- Al presentar la estructura lógica en el diccionario de datos fue necesario que el esquema XML sea validado a través de pruebas DTD (Definición de Tipo de Documento) ya que esta es una técnica que valida la sintaxis del esquema evitando errores durante su ejecución.

- La aplicación permitió modelar de una manera rápida y sencilla una base de datos propuesta por el usuario, por medio de artefactos gráficos entre los cuales se encuentran las entidades, atributos, campos claves, relaciones y cardinalidad definidos para la elaboración del diagrama entidad-relación.

- El uso de metodología RAD en el desarrollo del proyecto fue de gran ayuda puesto que permitió obtener una idea clara desde el inicio del proyecto con las actividades a cumplir durante su desarrollo hasta su finalización.

i. Recomendaciones

La ejecución del trabajo de titulación abre nuevas líneas de investigación relacionadas con mejoras en la aplicación, por lo que se recomienda:

- Implementar la siguiente fase del diseño de base de datos la cual consiste en transformar el modelo de datos al diseño lógico.

- Hacer uso del modelo relacional y de las técnicas de normalización como parte del proceso de transición al diseño lógico, ya que este modelo es actualmente el más empleado debido a se obtiene un esquema muy cercano a la realidad.

- Añadir la funcionalidad de Compartir Diagrama para trabajar de forma colaborativa en el mismo diagrama, asignándole los permisos respectivos para modificarlo.

- Realizar un análisis para determinar qué herramientas son las adecuadas para generar los artefactos o elementos del diagrama entidad-relación.

j. Bibliografía

[1] Merchan Oswaldo. (2004, Marzo) Fundamentos de Bases de Datos. [Online]. http://www.uazuay.edu.ec/isi/Fundamentos%20de%20Bases%20de%20Datos.pdf

[2] Caselli Gismondi Hugo. (2009) Base de Datos VII Ciclo. [Online]. http://biblioteca.uns.edu.pe/saladocentes/archivoz/publicacionez/001_manual_base_de_datos___h._caselli_g___v7.1.pdf

[3] Alvarez Castorela Victor. (2009) Elementos de un Sistema Manejador de Bases de Datos. [Online]. http://www.sites.upiicsa.ipn.mx/polilibros/portal/Polilibros/P_proceso/SISTEMAS_MANEJADORES_DE_BASES_DE_DATOS_Victor_Alvarez_Castorela/

[4] Proal Aguilar Carlos. (2010) Modelado de Datos. [Online]. http://ict.udlap.mx/people/carlos/is341/bases02.html

[5] Gil Juan Camilo Machado Adianes. (2007) Bases de Datos. [Online]. https://basesdedatos.wordpress.com/modelos-logicos-basados-en-objetos/

[6] Machado Adianes. (2007) Base de Datos. [Online]. https://basesdedatos.wordpress.com/modelos-logicos-basados-en-registros/

[7] Universidad Carlos III de Madrid. (2008) Diseño y Administración de Bases de Datos. [Online]. http://ocw.uc3m.es/ingenieria-informatica/diseno-y-administracion-de-bases-de-datos/teoria/Tema1(UnaMetodologiaDesarrolloBD).pdf

[8] Sanchez Jorge. (2004) Diseño Conceptual de Bases de Datos. [Online]. http://www.jorgesanchez.net/bd/disenoBD.pdf

[9] Coral Calero Marcela. Diseño Conceptual, Lógico y Físico. [Online]. http://alarcos.inf-cr.uclm.es/doc/bda/doc/teo/ant/BDa-t5.pdf

[10] Blázquez Ochando Manuel. (2014) Fundamentos y Diseño de Bases de Datos. [Online]. http://ccdoc-basesdedatos.blogspot.com/2013/02/modelo-entidad-relacion-er.html

[11] Cardoso Cabrera Humberto. (2014) Modelo Entidad Relacion. [Online]. http://www.ecured.cu/index.php/Modelo_Entidad_Relaci%C3%B3n

[12] Diana Gabriela Higuera Robles. (2014) Base de Datos 2. [Online]. http://katyygaby.blogspot.com/p/diccionario-de-datos.html

[13] Universidad Francisco Gavidia. (2014, Diciembre) Fundamentos de la Web. [Online]. http://www.isis.ufg.edu.sv/www.isis/documentos/TE/005.74-A594d/005.74-A594d-Capitulo%20II.pdf

[14] Leon Paulina. (2014) Lenguajes y Entornos. [Online]. http://aprenderaprogramar.com/index.php?option=com_content&view=article&id=102:i que-es-y-para-que-sirve-el-lenguaje-de-etiquetas-xml-extensible-markup-language&catid=46:lenguajes-y-entornos&Itemid=163

[15] The PHP Group. (2014, Enero) PHP, Conceptos Basicos. [Online]. http://php.net/manual/es/getting-started.php

[16] Alvarez Miguel Angel. (2014, Septiembre) Javascript a fondo. [Online]. http://www.desarrolloweb.com/javascript/

[17] (2014) Drawing shapes with canvas. [Online]. https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API/Tutorial/Drawing_shapes

[18] Educalab. (2014) SQLite. [Online]. http://recursostic.educacion.es/usuarios/web/preguntas-frecuentes/45-sqlite-

[19] Wikiversidad. Metodologías ágiles de desarrollo software. [Online]. http://es.wikiversity.org/wiki/Metodolog%C3%ADas_%C3%A1giles_de_desarrollo_software

[20] Agurto Ronald. (2014) Metodologias Agiles de Desarrollo. [Online]. http://ronald-sistemas.blogspot.com/2009/10/metodologias-agiles-de-desarrollo.html

[21] Gimson Loraine. Metodologías ágiles y desarrollo basado en conocimiento. [Online].http://sedici.unlp.edu.ar/bitstream/handle/10915/24942/Documento_completo__.pdf?sequence=1

[22] Proyectos Agiles.org. (2014) [Online]. http://www.proyectosagiles.org/como-funciona-scrum

[23] Vasquez Cristian. (2002) Los Lenguajes de Metadatos. [Online]. http://users.dcc.uchile.cl/~cvasquez/meta/lenguajes.html

[24] Carbajo Fernando. (2010) Modelizacion de datos. [Online]. http://jroliva.com/fernando/An%C3%A1lisis/Teoria/Tema4.pdf

[25] Cordero Valle Juan Manuel. (2005) Modelos de Datos. [Online]. http://www.lsi.us.es/docencia/get.php?id=1456

[26] Duque Mendez Nestor. (2013) Bases de Datos. [Online]. http://www.virtual.unal.edu.co/cursos/sedes/manizales/4060029/lecciones/cap2-1.html

[27] Tabares Martha Silvia. (2000) Diseño Conceptual. [Online]. http://www.unalmed.edu.co/~mstabare/diseno_conceptual.htm

[28] Universidad de Oviedo. (2014) Diccionario de Datos. [Online]. http://www.docstoc.com/docs/271656/diccionario-de-datos

[29] Garcia Gonzalez Victor. Metadatos en la web. [Online]. http://www.denibol.com/metadatos_xml_rdf/?id=metadatos_web

[30] Sanchez Jorge. (2004) Diseño Conceptual de Bases de Datos. [Online]. http://www.jorgesanchez.net/bd/disenoBD.pdf

[31] Apache Software Foundation. (2014) Apache JMeter. [Online]. http://jmeter.apache.org/

k. Anexos

Anexo 1. Prototipos de Pantallas Iniciales

En cuanto a los prototipos se estableció de manera clara la estructura y flujo de navegación, donde cada pantalla posee las funcionalidades del proyecto.

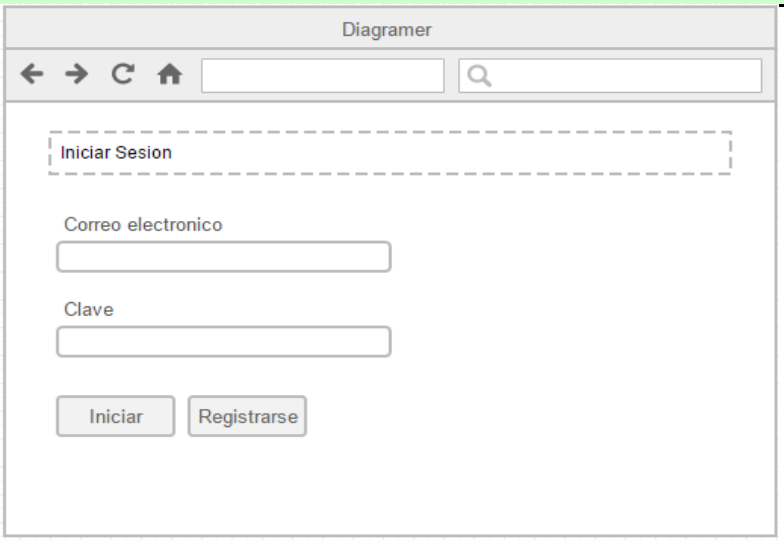
Pantalla	Funcionalidad
	Esta pantalla permite al usuario registrarse e iniciar sesión para de esa manera poder acceder a la aplicación.

Figura 26. Prototipo Pantalla Iniciar Sesión

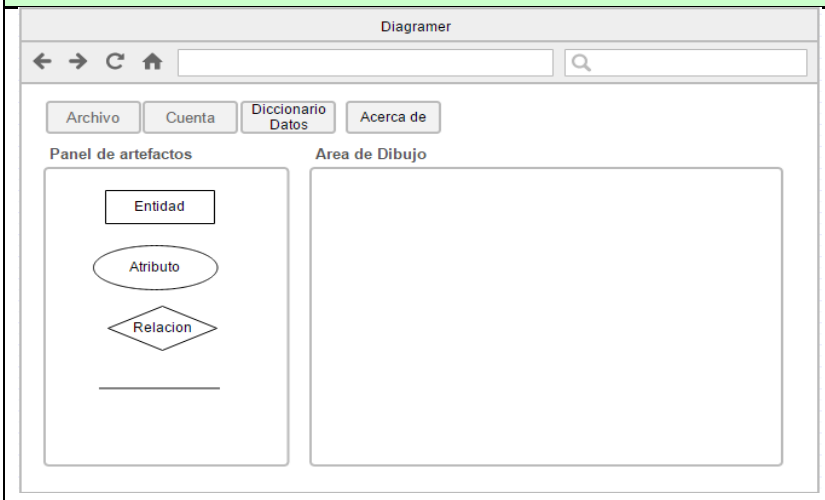
Pantalla	Funcionalidad
	Esta pantalla permite realizar el diagrama entidad-relación en base a los artefactos establecidos.

Figura 27. Prototipo Pantalla Área de Diagrama

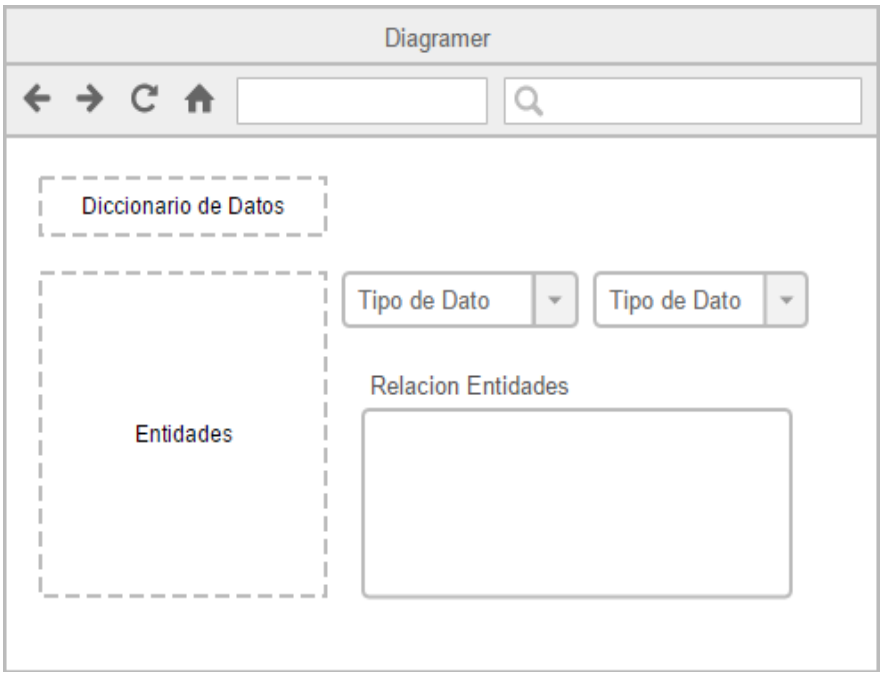
Pantalla	Funcionalidad
	Esta pantalla muestra el diccionario de datos en base al diagrama entidad-relación estableciendo sus tipos de datos, relaciones, etc.

Figura 28. Prototipo Pantalla Diccionario Datos

Anexo 2. Instalación y Configuración de Servidor Web

Para el desarrollo de la aplicación se optó por emplear las siguientes herramientas, seguidamente se describe el proceso de instalación:

- **Instalación de Apache**.

El servidor web **Apache** es actualmente el más empleado de los servidores web en el mundo. Para realizar la instalación ejecutamos los siguientes comandos en la terminal:

```
apt-get install apache2
```

Para confirmar si está levantado dicho servidor basta con digitar en el navegador lo siguiente: http://localhost/.

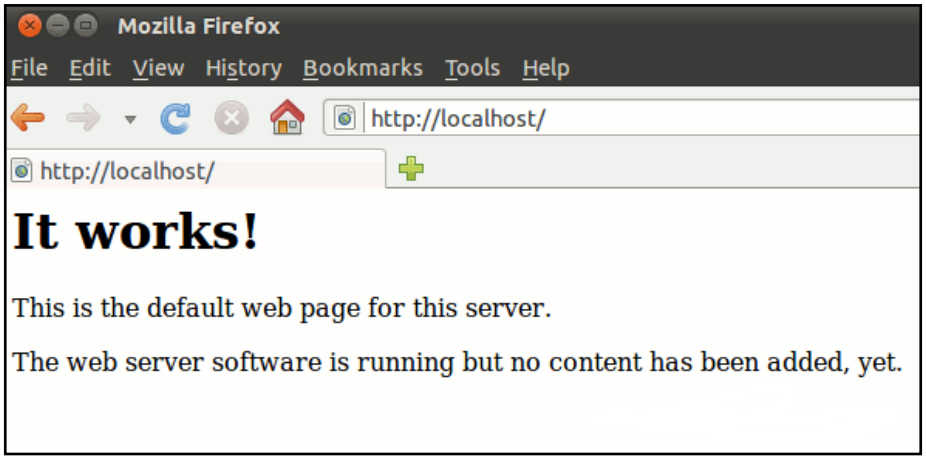


Figura 29. Pantalla de Servicio Iniciado

- **Activar directorios de usuario en Apache - public_html**

Para disponer de esta función debemos activar el módulo de Apache userdir. Creamos el directorio public_html en nuestra cuenta de usuario. En el alojamos los archivos que se verán a través del navegador. Desde un terminal ejecutamos las siguientes órdenes:

```
# mkdir ~/public_html
# su a2enmod userdir
# su /etc/init.d/apache2 restart
```

Ocurre que los script en php en vez de ejecutarse se descargan. Como no queremos esto, debemos editar un archivo de configuración. Desde el terminal abrimos para editar:

```
# nano /etc/apache2/mods-enabled/php5.conf
```

Buscamos en el archivo hasta ver algo así:

```
<IfModulemod_userdir.c>
<Directory /home/*/public_html>
php_admin_value engine Off
</Directory>
</IfModule>
```

Como nos dice en las líneas comentadas, las que llevan una almohadilla, debemos comentar desde <IfModule> hasta </IfModule> para que quede así:

Figura 30. Pantalla Activación de Usuario en Apache

Podríamos simplemente cambiar php_admin_valueengine Off por php_admin_valueengineOn, pero esto se impondría sobre .htaccess donde a veces podemos desear indicar que no puedan ejecutarse scripts php.

Y reiniciamos el servidor Apache para que los cambios tengan efecto:

```
# services apache2 restart
```

Ahora podemos ejecutar nuestros scripts en php en nuestro directorio de usuario de Apache.

- **Adquisición del dominio**

El registro del dominio se lo realizo vía online a través de la pg. Nic.ec que es quien administra el Registro de Nombres de Dominio, a continuación se observa su proceso:

Figura 31. Pantalla Adquisión de Dominio

- **Configuración del dominio en servidor web**

Una vez registrado el dominio seleccionado e procedió a rediccionarlo de la siguiente manera con el siguiente comando:

nano/etc/apache2/sites-enabled/000-defaut

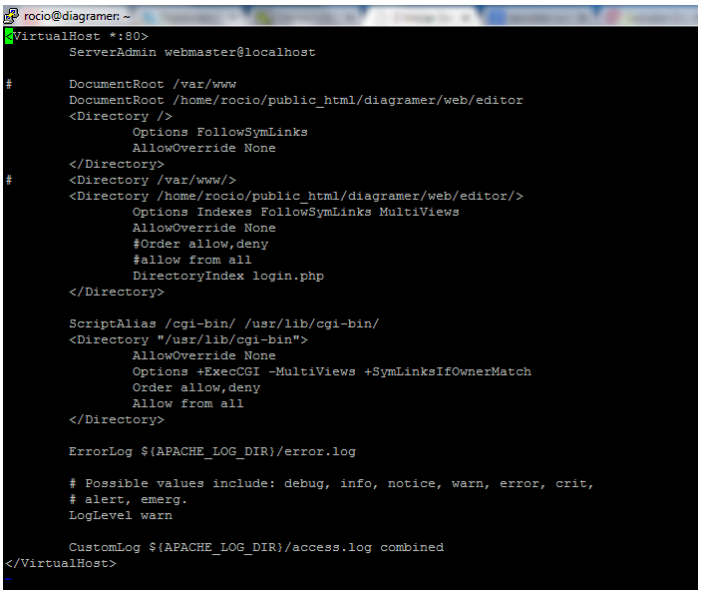


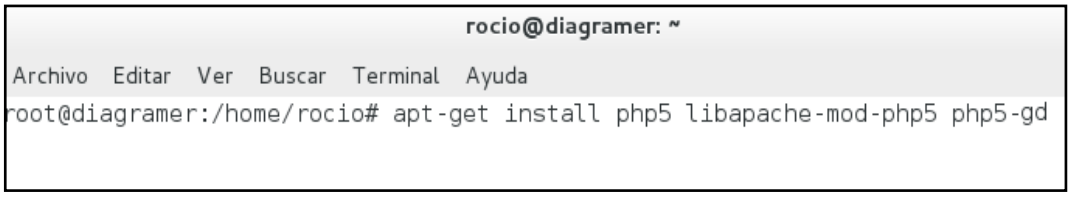
Figura 32. Fichero de Configuración

Finalmente reiniciamos el servidor

```
# servicesapache2 restart
```

- **Instalación de PHP**

Como lenguaje de programación se empleó php que es de uso general y que fue originalmente diseñado para el desarrollo web de contenido dinámico, para ello en una terminal digitamos lo siguiente:



Una vez que se hayan descargado e instalado los paquetes de PHP, se debe reiniciar el servidor Apache para que el servicio esté disponible. Para hacerlo digitamos nuevamente el siguiente comando:

```
# servicesapache2 restart
```

- **Instalación de sqlite3**

```
# apt-get install sqlite3
```

Anexo 3. Licencias aplicadas al Sistema.

El trabajo de titulación consta de dos tipos de licencias, es decir una para la documentación y otra para el código fuente del sistema.

- **Licencia General Public License**

La Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General Public License, es la licencia más ampliamente usada en el mundo del software y garantiza a los usuarios finales la libertad de usar, estudiar, compartir (copiar) y modificar el software. Se optó por la elección de esta licencia con el propósito es establecer que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios. De esta manera el código del proyecto denominado "Herramienta de Diseño para la Elaboración de Modelos de Datos del Diseño Conceptual", se encuentra bajo la licencia GNU GPL versión 3.

- **Licencia Creative Commons**

Las Licencias de derechos de autor Creative Commons no significa que no tengan copyright, este tipo de licencias ofrecen algunos derechos bajo ciertas condiciones. Estas licencias estandarizadas, que en lugar de prohibir el uso lo autoriza bajo algunas condiciones.

De acuerdo a ello se optó por elegir el siguiente tipo de licencia para el proyecto de titulación en lo que se refiere a la documentación:

Reconocimiento – No Comercial – Compartir Igual (by-nc-sa): en la que no se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

Anexo 4. Anteproyecto.

a. Tema

"Herramienta de diseño para la elaboración de modelos de datos del diseño conceptual."

b. Problemática

1. Situación Problemática

Desde un inicio la información ha ocupado un lugar predominante hasta llegar a ser un recurso fundamental para la actividad del hombre. El uso y la generación de información han crecido a gran ritmo, debido al aumento significativo de las instituciones dedicadas a la investigación y al desarrollo de nuevos horizontes tecnológicos, es así que dentro del contexto organizacional se requiere manejar de forma óptima y organizada dicha información, la cual según su importancia es necesario almacenarla para su uso posterior, este proceso de almacenamiento de información se lleva a cabo a través de la denominada base de datos, proceso que además requiere de un adecuado diseño.

El diseño de una base de datos se desarrolla en cuatro fases: recolección y análisis de requerimientos, diseño conceptual, diseño lógico y diseño físico. La primera fase supone la construcción de un conjunto de requerimientos suficiente por parte del diseñador, acerca del problema o situación que da origen al desarrollo de la base de datos; con este análisis se hace el diseño conceptual, expresado en un diagrama relacional o en un diagrama entidad relación extendido, que actualmente son los más utilizados mismos que proporcionan una abstracción de la propuesta de solución; este diseño conceptual se transforma en un diseño lógico, relacionado con la estructura interna de las tablas; finalmente, se hace el diseño físico en donde la base de datos encaja en la infraestructura de hardware y software destinada a soportarla.

Sin embargo en la actualidad, al evaluar herramientas de diseño de una Base de Datos tales como Lucidchart, Dia entre otros, se pudo observar que su dinámica de trabajo permite diagramar una problemática dada pero sin ofrecer una representación que integre en su totalidad el proceso del diseño conceptual, es decir que para obtener el modelo

conceptual representado por el diagrama Entidad Relación y el modelo lógico, representado por el diagrama relacional se lo debe llevar de manera individual.

Por lo tanto el origen del proyecto de investigación abordó el problema del proceso de diseño de una Base de Datos anteriormente mencionado.

De acuerdo a la problemática expuesta surge la necesidad de desarrollar una herramienta de diseño web que ofrezca una integración del diseño conceptual, que permita construir y depurar dichos diagramas.

2. Problema de Investigación

Debido a la creciente aceptación de las bases de datos por parte de la industria y a una variedad de aplicaciones científicas y técnicas, el diseño de bases de datos desempeña un papel central en el empleo de los recursos de información en la mayoría de las organizaciones.

Desafortunadamente, las metodologías de diseño de bases de datos no son muy populares, la mayoría de las organizaciones y de los diseñadores individuales confía muy poco en las metodologías para llevar a cabo el diseño y esto se considera, con frecuencia, una de las principales causas de fracaso en el desarrollo de los sistemas de información.

El diseño de bases de datos ha pasado a constituir parte de la formación general de los informáticos, en el mismo nivel que la capacidad de elaborar algoritmos usando un lenguaje de programación convencional.

Es así que el problema al que se pretende dar una alternativa de solución es el siguiente:

"El diseño de una Base de Datos se ve afectado al tratar de generar los diagramas del diseño conceptual debido a que las herramientas que modelan dicho proceso no permiten una integración de sus modelos de datos.**"**

c. Justificación

El presente trabajo de fin de carrera (TFC) referida al desarrollo de una herramienta web para la elaboración de diagramas busca mediante la aplicación de los conceptos y avances científicos respecto a las nuevas tecnologías encontrar la forma de facilitar que el proceso de diseño conceptual de base de datos se lleve de forma estructurada y que se lo considere como un proceso estable con métodos y técnicas propios.

Respecto a la formación académica a la cual nos debemos, el TFC hará posible poner en práctica los conocimientos adquiridos durante el transcurso de la vida estudiantil, donde más allá de cumplir con los requisitos que demanda la carrera, somos conscientes de la gran experiencia que implica estar en contacto con la realidad en la que vivimos contribuyendo a formarnos como personas críticas, creativas y propositivas.

Es así que el TFC pretende desarrollar y proveer al estudiante de una metodología para el diseño conceptual de base de datos que facilite su comprensión, teniendo como base las técnicas de diseño para llegar a una representación que modele un caso del mundo real, además de proponer un diseño consistente a través de una herramienta web como parte del proceso tecnológico que implica su implementación.

Desde el punto de vista económico el presente proyecto de tesis demanda costos moderados en relación al beneficio que se obtendrá al implementar una herramienta de diseño que permita obtener un diseño conceptual sustentable que represente una abstracción de la propuesta de solución a una problemática dada. Así mismo cabe señalar que el costo que implica su elaboración va por cuenta de su desarrollador.

Por todo lo antes mencionado y respecto al interés investigativo, el esfuerzo personal, ético y profesional de indagar y proponer posibles soluciones sobre la problemática planteada, se puede determinar que es un trabajo de investigación totalmente viable.

d. Objetivos

1. Objetivo General

- Diseñar e implementar una herramienta de diseño web para la elaboración de modelos de datos del diseño conceptual.

2. Objetivos Específicos

- Analizar los distintos modelos de datos del diseño conceptual que permita describir el comportamiento relacional al diseñar una base de datos.

- Analizar las entidades y relaciones para definir sus atributos, identificadores y cardinalidad.

- Construir un panel de elementos gráfico que permita representar los elementos de un diagrama conceptual o lógico.

- Crear un diccionario de datos para almacenar la información del diagrama entidad-relación que contenga las características lógicas de los datos que se van a utilizar en el sistema.

Anexo 5. Certificación de Traducción

Yo, Freddy Castillo Hoyos, profesor del Instituto Washington;

Certifico:

Que tengo el conocimiento y dominio de los idiomas español e inglés y que las traducciones de los siguientes:

RESUMEN del tema:

"Herramienta de Diseño para la Elaboración de Modelos de Datos del Diseño Conceptual".

para: TENE PLAZA ROCIO ELIZABETH

es verdadero y correcto a mi mejor saber y entender.

Firmado en Loja a los ocho días del mes de abril de 2015.

WASHINGTON ENGLISH
DIRECCION ACADEMICA
Loja - Ecuador