PRIMICIA, Plataforma de Televisión Informativa.
Rol de Diseñador de Bases de Datos

UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

2008-2009

Autor: Karen Mercedes Zerquera Rosa

Tutor: Yunior Montaner Hernández

Las estrellas brillan para que cada uno pueda un día encontrar la suya.

Antoine de Saint-Exsupery, El pequeño Príncipe.

A mis padres por ser mi razón de ser.

A mi familia, sin cuyas raíces no hubiera sido posible la voluntad y fortaleza para llegar hasta aquí.

A los que me estiman y me quieren.

AGRADECIMIENTOS

Mami y papi, los amo, por ustedes estoy aquí, y para ustedes van todos mis logros.

A mis abuelos, les agradezco su infinito afecto, cariño y preocupación.

A mis tíos y primos, son especiales, gracias por su granito de arena.

A todos mis profesores que durante estos años han aportado sus conocimientos para mi formación.

A Yunior Montaner, tutor de mi trabajo, por ser un amigo.

A mis compañeros de aula y a todos los que han colaborado para que estos cinco años pasaran como una hermosa experiencia.

Gracias Albert, cuya ayuda y esfuerzo han contribuido a tornar palpable mi sueño. Alegna, Yadira, Elizabeth, Juan Carlos, Darluin, Yaro, Yoa, Arturo y Jean Michel gracias por todos los momentos que pasamos juntos.

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Facultad 9 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Karen Mercedes Zerquera Rosa

_____

Yunior Montaner Hernández

_____

DATOS DEL CONTACTO

Tutor: Ing. Yunior Montaner Hernández.

- ✓ Ingeniero en Ciencias Informáticas, Universidad de Ciencias Informáticas, 2007.
- ✓ Vinculado al Proyecto Productivo UCITeVe durante los 5 años de estudios universitarios trabajando en el área de Desarrollo de Aplicaciones Informáticas para la Televisión.
- ✓ Profesor del Departamento de Técnicas de Programación, Facultad 9.

E-mail: ymontaner@uci.cu

RESUMEN

La información es poder y hoy por hoy está en todas partes. Las estrategias de comunicación no pueden basarse en una sola vía. La efectividad de los mensajes dependerá cada vez más del uso complementario de medios para hacerlo llegar, lo ideal es llegar a las audiencias a través de un sistema multimedia, en el cual se combine el uso de medios directos con medios de comunicación colectiva.

Los potenciales receptores de los mensajes han cambiado, las tecnologías de la información han transformado los medios de comunicación y han aportado nuevas opciones, ante tales retos la Universidad de Ciencias Informáticas labora en el desarrollo de sistemas que van orientados a la transmisión automatizada de televisión informativa.

Por el desarrollo que han tenido versiones de este tipo de sistema, ha hecho que el mismo se conceptualice como producto, decidiéndose desarrollar PRIMICIA, la plataforma de Televisión Informativa, como respuesta a la automatización de la transmisión de televisión informativa, donde se velará principalmente por la manipulación de la información, que constituye el eslabón más fuerte en este tipo de sistema, donde una pérdida de la misma ocasionaría que no llegase de manera eficiente, constante, e inmediata, que es la funcionalidad principal de este producto.

Este trabajo centra su objetivo en dotar a PRIMICIA de una estructura eficiente donde almacenar la información referente a las noticias y la administración y gestión de las mismas.

En este documento se plasma todo el proceso de diseño, implementación y prueba de la base de datos de PRIMICIA.

Palabras claves: información, televisión informativa, PRIMICIA, almacenar, base de datos.

TABLA DE CONTENIDO

TABLA DE ILUSTRACIONES.

Introducción

La arqueología es la ciencia que ha seguido la evolución de la conducta cultural humana a través del estudio de sus huellas, huellas que constituyen declaraciones de los diferentes cambios físicos y estructuras sociales del hombre prehistórico, génesis del hombre contemporáneo.

Durante toda su vida, este hombre recibía las profundas impresiones visuales y la dinámica que comportaba su peligrosa existencia diaria, de aquí derivó su extraordinario conocimiento, relaciones y comunicación con sus semejantes.

La demanda de realizar acciones conjuntas como el trabajo, baile y otros, jugaron un papel importante en la comunicación que comenzó con un comportamiento básico, basado en respuestas instintivas. Con el desarrollo del cerebro humano, que hizo posible el pensamiento abstracto y la expresión a través del lenguaje se abre una etapa en la comunicación más sofisticada. Los pueblos antiguos buscaron los medios para representar y enviar sus mensajes. Pintaban en las paredes de las cuevas y utilizaban signos y símbolos para designar tribus y pertenencias.

Con el desarrollo de las civilizaciones y de las lenguas surgió también la necesidad de comunicarse a distancia de forma regular, con el fin de facilitar el comercio entre naciones e imperios.

De los diferentes tipos de servicios de comunicación de la antigüedad, el más notable fue el sistema de relevos del Imperio persa. Jinetes a caballo transportaban mensajes escritos de una estación de relevos a otra. Basándose en este sistema, los romanos desarrollaron su propio sistema de postas (del latín positus, 'puesto'), de donde procede el término "servicio postal".

La creación de los sistemas postales modernos siguió creciendo con la aparición del ferrocarril, los vehículos de motor, los aviones y otros medios de transporte. Sin embargo, a lo largo de los siglos siempre se busco medios de comunicación a larga distancia que fueran más rápidos que los convencionales y que hicieran llegar la información con cierta inmediatez.

El mundo avanza imparable y la comunicación, proceso de transmisión y recepción de ideas, información y mensajes, sigue constituyendo una necesidad imperiosa.

Cada vez más empresas y otros centros necesitan hacer llegar mensajes con cierta inmediatez a grandes audiencias, si esto no se presenta de una forma adecuada se torna

difícil la comunicación, de aquí, se deriva la importancia de desarrollar sistemas de comunicación versátiles.

Las tecnologías de la información han transformado los medios de comunicación y han aportado nuevas opciones.

La evolución de las formas de comunicar es una necesidad impostergable. La Universidad de Ciencias Informáticas (UCI), consecuente con esta realidad labora en el desarrollo de sistemas que van orientados a la transmisión automatizada de televisión informativa, la cual está destinada a la emisión constante de información en diferentes formatos.

Actualmente las versiones Señal 3, el canal informativo de la UCI y Señal ACN, el canal de teletexto de la Plataforma Satelital Cubana, que va destinado a los cooperantes cubanos que cumplen misión internacionalista y a los habitantes de las zonas de silencio del país como sustitución de la prensa plana, son una muestra del progreso obtenido para dar solución a la problemática de la comunicación masiva.

El desarrollo de estas soluciones ha posibilitado que el sistema se conceptualice como producto informático, decidiéndose desarrollar PRIMICIA, Plataforma de Televisión Informativa, como una respuesta a la automatización de la transmisión de televisión informativa.

Al definirse como producto, hace que el mismo sea más genérico, la implementación deberá ser aún más robusta que la de las versiones anteriores, velará principalmente por la manipulación de la información, que constituye el eslabón más fuerte en este tipo de sistema, y una pérdida de la misma ocasionaría que esta no llegase de manera eficiente, constante, e inmediata, que es la funcionalidad principal de este producto.

Por la necesidad de desarrollar este producto y que cumpla con los requerimientos básicos que demanda el mismo surge el siguiente **problema**:

"La no existencia de una estructura eficiente para el correcto almacenamiento de la información correspondiente a las noticias y la administración y gestión de las mismas, hace que el funcionamiento y desarrollo de PRIMICIA no sea el más correcto y eficiente".

Se tomará como **objeto de estudio**:

"La Plataforma de Televisión Informativa PRIMICIA".

El **campo de acción** se delimitó en:

"Las actividades del rol Diseñador de Bases de Datos durante el ciclo de desarrollo del software PRIMICIA".

A partir del problema se definió como **objetivo general**:

"Diseñar la base de datos para PRIMICIA" y como **objetivos específicos**:

- Modelar la Base de Datos.
- Implementar la Base de Datos.
- Garantizar la seguridad de los datos.

Para dar cumplimiento al objetivo anteriormente planteado se dará cumplimiento a las siguientes **tareas**:

- Evaluar el negocio y requisitos del sistema y herramientas para el correcto diseño de la base de datos.
- Evaluar la integración y comunicación con los módulos del sistema.
- Evaluar replicas de la base de datos y datos persistentes.
- Elaborar la propuesta de diseño de la base de datos a utilizar en el desarrollo y despliegue del sistema.
- Validar teóricamente la propuesta de diseño de la base de datos.
- Evaluar herramientas para el análisis y pruebas de carga intensiva.
- Validar funcionalmente la propuesta de diseño de la base de datos.
- Modelar los artefactos correspondientes al rol Diseñador de Bases de Datos.

Los **métodos científicos** utilizados para darle solución al problema son los siguientes:

Métodos Teóricos:

- **Análisis Histórico-Lógico**: Utilizado para describir las tendencias actuales de creación de bases de datos y de los sistemas enfocados a la transmisión de Televisión Informativa a nivel mundial, nacional y en la UCI, de manera que se puede deducir cual es la más idónea para PRIMICIA.
- **Analítico-Sintético**: Permitirá extraer lo esencial de la bibliografía consultada.

- **Inductivo-Deductivo:** Facilita el análisis de los elementos generales a elementos más particulares, o sea, deducir cual es la mejor estrategia de diseño de la base datos de PRIMICIA partiendo del resultado de otras ya implementadas.
- **Modelación**: Utilizado para obtener el modelo lógico, el modelo de datos y el diagrama de clases persistentes.

Capítulo 1 Fundamentación Teórica

1.1 Introducción.

En el presente capítulo aporta una visión general de la concepción de PRIMICA, y otros relacionados con el manejo de la información más específicamente en las bases de datos. Para ello se da una descripción de los principales conceptos asociados al dominio del problema.

Con esto se realiza un primer acercamiento a las tendencias y tecnologías sobre las que se apoyará la propuesta final.

1.2 PRIMICIA

PRIMICIA se ha conceptualizado para ser desarrollado y soportado completamente en software libre, contribuyendo de esta manera a la soberanía tecnológica del país.

Se encuentra estructurado en dos subsistemas que se relacionan entre sí y actúan como un todo para brindar un resultado final eficiente y acorde a las necesidades de los usuarios.

En el Subsistema de Administración, se realiza la administración del canal y toda la gestión de las informaciones y recursos multimedia que se transmiten, y el Subsistema de Transmisión es el encargado de la visualización de las noticias y materiales publicados.

Su arquitectura física puede ser adaptada en dependencia del tipo de red de televisión que se desee utilizar para la transmisión del canal y las condiciones del entorno donde se instale, las que pueden ser más sofisticadas o en algún caso estar restringidos por un bajo presupuesto.

El canal informativo que soporta la plataforma muestra de forma automática ciclos de noticias constantes y repetitivos condicionados por las informaciones publicadas para determinados períodos de tiempo, logrando integrar en sus transmisiones textos combinados con imágenes y video.

Durante las emisiones permite la reproducción de un fondo musical que puede ser personalizado según la noticia que se muestra, además es posible la utilización de cintillos informativos o infocintas que permitan el adelanto de breves asuntos de carácter relevante o promocional.

Como elemento adicional es posible mostrar la fecha y hora, tiempo restante de la noticia o pantalla y titular de la próxima.

El sistema ofrece además la transmisión de señales de video externas, tales como filmaciones en vivo o transmisiones de otras televisoras.

Es importante señalar que el desarrollo de PRIMICIA se ha enfocado hacia la obtención de funcionalidades genéricas fácilmente escalables, que no dependan de un entorno dado y no atadas a un diseño gráfico específico, pudiendo acordar con los clientes la personalización o eliminación específica de alguna funcionalidad.

1.3 Introducción a las Bases de datos.

Sobre las Bases de datos existen criterios diversos:

- Una base de datos es un conjunto de datos almacenados entre los que existen relaciones lógicas y ha sido diseñada para satisfacer los requerimientos de información de una empresa u organización. En una base de datos, además de los datos, también se almacena su descripción(2).
- Un sistema de bases de datos es básicamente un sistema computarizado cuya finalidad general es almacenar información y permitir a los usuarios recuperar y actualizar esa información con base en peticiones(20).
- Colección o depósito de datos integrados, almacenados en soporte secundario (no volátil) y con redundancia controlada. Los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de ellos, y su definición –estructura de la base de datos– única y almacenada junto con los datos, se ha de apoyar en un modelo de datos, el cual ha de permitir captar las interrelaciones y restricciones existentes en el mundo real. Los procedimientos de actualización y recuperación, comunes y bien determinados, facilitarán la seguridad del conjunto de los datos(21).

A modo general una base de datos (BD) es un conjunto de información relacionada con un tema o actividad específica, que está formado por un conjunto de datos contenidos en dispositivos de almacenamiento masivo de información (disco), que permiten el acceso directo a esos datos, y por un grupo de programas que los manipulan.

Las BD representan algún aspecto del mundo real, llamado universo de discurso o mini mundo. Constituyen un conjunto lógicamente coherente, por lo que una colección aleatoria de datos no puede considerarse propiamente una BD.

Una BD se diseña, construye y puebla con datos para un objetivo específico, ya que debe estar dirigida a un grupo de usuarios y tener aplicaciones preconcebidas.

1.4 Diseño de una base de datos.

A medida que han ido avanzando las tecnologías relacionadas con las BD han evolucionado también las metodologías y técnicas de diseño.

Se ha alcanzado un entendimiento común en cuanto al proceso de diseño de bases de datos en fases, basándose en el los principales objetivos de cada etapa y sobre las técnicas para conseguir los mismos (1).

El diseño de una BD se encuentra dividido en tres fases: el diseño conceptual, el diseño lógico y el diseño físico.

Fig. 1 Fases o Etapas del diseño de una base de datos.

En el **diseño conceptual** se debe construir un esquema de la información del entorno o el sistema donde se implantará la BD, independientemente de cualquier consideración física, a este se le denomina esquema conceptual. El objetivo del mismo es comprender el punto de vista que cada usuario tiene de los datos, su naturaleza y su uso a través de las áreas de aplicación. El mismo se construye utilizando la información que se encuentra en la especificación de los requisitos de usuarios. El diseño conceptual es completamente independiente de los aspectos de implementación, como puede ser el Sistema Gestor de Base de Datos (SGBD). Este esquema es una fuente de información para el diseño lógico de la BD.

El **diseño lógico** es el proceso de construir un esquema de información pero basándose en un modelo de BD específico, independiente del SGBD concreto que se vaya a utilizar o de cualquier otra consideración física. En esta etapa, se transforma el esquema conceptual en un esquema lógico que utilizará las estructuras de datos del modelo de BD en el que se basa el SGBD que se vaya a utilizar, como puede ser el modelo relacional, el modelo de red, el modelo jerárquico o el modelo orientado a objetos. El esquema lógico es una fuente de información para el diseño físico. Además, juega un papel importante durante la etapa de mantenimiento del sistema, ya que permite que los futuros cambios que se realicen sobre los programas de aplicación o sobre los datos, se representen correctamente en la BD.

El **diseño físico** es el proceso de producir la descripción de la implementación de la BD en estructuras de almacenamiento y métodos de acceso que garanticen un acceso eficiente a los datos. Para dar comienzo a esta etapa, se debe haber decidido cuál es el SGBD que se va a utilizar, ya que el esquema físico se adapta a él. Entre el diseño físico y el diseño lógico hay una realimentación, ya que algunas de las decisiones que se tomen durante el diseño físico para mejorar las prestaciones, pueden afectar a la estructura del esquema lógico. (3)

1.5 Tipos de bases de datos.

Las bases de datos pueden clasificarse de varias formas, de acuerdo al criterio elegido para su clasificación, que puede ser según la variabilidad de los datos almacenados o según el contenido.

1.5.1 Según la variabilidad de los datos almacenados.

Esta clasificación depende de cómo los datos cambian o varían dentro de la BD, o si se mantienen estáticos sin sufrir modificaciones o cambios. Dentro de esta clasificación se tiene lo siguiente:

Bases de datos estática:

Éstas son bases de datos de sólo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones.

Bases de datos dinámicas:

Éstas son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta. Un ejemplo de esto puede ser la BD utilizada en un sistema de información de una tienda, una farmacia, un videoclub, etc. (4)

1.5.2 Según el contenido.

Esta clasificación basa su enfoque en el contenido o lo que almacena la BD. Ejemplos de esta clasificación son:

Bases de datos bibliográficas:

Solo contienen un "representante" de la fuente primaria, que permite localizarla. Un registro típico de una BD bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, etc. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo, porque sino se estaría en presencia de una BD a texto completo (o de fuentes primarias). Como su nombre lo indica, el contenido son cifras o números.

Bases de datos de texto completo:

Almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas.

Directorios:

Un ejemplo son las guías telefónicas en formato electrónico.

Bases de datos de información biológica:

Son bases de datos que almacenan diferentes tipos de información proveniente de las ciencias de la vida o médicas. (2)

Por las características que presenta la concepción del sistema PRIMICIA y por la necesidad que presenta el mismo de modificar información con el tiempo, o sea de actualizar y adicionar datos, así como de utilizar las operaciones fundamentales de consulta, el tipo de BD seleccionada para el desarrollo de este trabajo ha sido una **Base de datos Dinámica**.

1.6 Modelo de bases de datos:

Además de la clasificación por la función de la BD, también se le puede clasificar de acuerdo a su modelo de administración de datos.

Un modelo de datos es una descripción del contenedor de datos, así como de los métodos que utiliza para almacenar y recuperar la información de esos contenedores. Los mismos no constituyen elementos físicos, son abstracciones que permiten la implementación de un sistema eficiente de base de datos.

Algunas de los modelos utilizados con frecuencia son:

Bases de datos jerárquicas:

Estas bases de datos almacenan su información en una estructura jerárquica.

En este modelo los datos se organizan en una forma similar a un árbol, en donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres es llamado raíz, y a los nodos que no tienen hijos se los conoce como hojas.

Las bases de datos jerárquicas son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo crear estructuras estables y de gran rendimiento.

Una de las principales limitaciones de este modelo es su incapacidad de representar eficientemente la redundancia de datos.

Bases de datos de red:

Éste es un modelo ligeramente distinto del jerárquico; su diferencia fundamental es la modificación del concepto de nodo: se permite que un mismo nodo tenga varios padres (posibilidad no permitida en el modelo jerárquico).

Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos; pero, aun así, la dificultad que significa administrar la información en una BD de red ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales.

Bases de datos relacionales:

Es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. No tardó en consolidarse como un nuevo paradigma en los modelos de BD. Su idea fundamental es el uso de "relaciones". Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "tuplas".

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), lo cual tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos.

La información puede ser recuperada o almacenada mediante "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información. El lenguaje más habitual para construir las consultas en bases de datos relacionales es SQL, un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales. Durante su diseño, una BD relacional pasa por un proceso al que se le conoce como normalización de una BD.

Bases de datos Orientadas a objetos:

Este modelo, bastante reciente, y propio de los modelos informáticos orientados a objetos, trata de almacenar en la BD los objetos completos (estado y comportamiento).

Una BD orientada a objetos es una base de datos que incorpora todos los conceptos importantes del paradigma de objetos: encapsulación (propiedad que permite ocultar la información al resto de los objetos, impidiendo así accesos incorrectos o conflictos), herencia (propiedad a través de la cual los objetos heredan comportamiento dentro de una jerarquía de clases) y polimorfismo (propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos). En las bases de datos orientadas a objetos, los usuarios pueden definir operaciones sobre los datos como parte de la definición de la base de datos.

Bases de datos documentales:

Permiten la indexación a texto completo, y en líneas generales realizar búsquedas más potentes.

Bases de datos deductivas:

Un sistema de base de datos deductivo, es un sistema de base de datos pero con la diferencia de que permite hacer deducciones a través de inferencias. Se basa principalmente en reglas y hechos que son almacenados en la base de datos. También las bases de datos deductivas son llamadas base de datos lógica, a raíz de que se basan en lógica matemática. (5)

El modelo de Base de datos a utilizar para PRIMICIA será el **modelo de Base de datos relacional** ya que con este permite evitar la duplicidad de registros, además favorece la normalización por ser más comprensible al trabajar, y es un modelo donde el orden de inserción de los datos no es relevante, beneficiando esto el trabajo en la base de datos.

1.7 Sistemas Gestores de Bases de datos:

Los Sistemas Gestores de Bases de datos (SGBD) constituyen un conjunto de programas no visibles al usuario final que se encargan de la privacidad, la integridad, la seguridad de los datos y la interacción con el sistema operativo.

Proporciona una interfaz entre los datos, los programas que los manejan y los usuarios finales.
Cualquier operación que el usuario hace contra la base de datos está controlada por el gestor.

El objetivo principal de los sistemas gestores de bases de datos es proporcionar una forma de almacenar y recuperar la información de una base de datos de manera que sea tanto práctica como eficiente. (6)

Los SGBD dan la posibilidad de eliminar las inconsistencias en los datos, permiten compartir los mismos datos entre diferentes aplicaciones con distintas necesidades, mejoran la seguridad de los datos, pues normalmente incorporan mecanismos de seguridad en el propio SGBD, y permiten la creación de entornos de alta disponibilidad, pues con algunos SGBD es posible llegar a disponer de aplicaciones funcionando ininterrumpidamente.

Pero a pesar de disponer de tantas ventajas presentan dificultades. Una que requieren de una gran cantidad de espacio en disco y de memoria para trabajar de forma eficiente y dos que son vulnerables a los fallos ya que el hecho de que todo esté centralizado en el SGBD hace que el sistema sea más vulnerable ante los fallos que puedan producirse

Existen varios SGBD, ejemplo de ellos son:

1.7.1 Microsoft SQL Server:

Constituye un SGBD relacionar, basa su lenguaje en Transact-SQL, es capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. Como ventajas presenta que provee a la base de datos de escalabilidad, estabilidad y seguridad, que soporta procedimientos almacenados, que incluye un potente entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente y que además permite administrar información de otros servidores de datos. Pero presenta como desventajas que tiene valores muy altos sus licencias de uso y que solo puede ser utilizado sobre los Sistemas Operativos Windows. (7)

1.7.2 Oracle.

Es un SGBD relacionar desarrollado por Oracle Corporation.

Se considera como uno de los SGBD más completos por su estabilidad, seguridad, soporte de transacciones, escalabilidad y soporte multiplataforma.

Sus principales ventajas son que este sistema puede ser usado tanto para gigantescas bibliotecas multimedia como para el manejo de información personal, soporta la mayoría de los idiomas computacionales, corre en más de 80 arquitecturas de hardware y software distintos sin tener la necesidad de realizar modificaciones, soporta datos alfanuméricos, así como textos sin estructura, imágenes, audio y video y está disponible en múltiples plataformas como Windows, Linux, y todas las versiones de Unix.

Pero este sistema presenta desventajas; constituye un producto donde sus costos por soporte técnico son elevadísimos, así como por constituir un software propietario y su licencia ser de las más caras en el mundo del software solo se ve en empresas muy grandes y multinacionales, por lo general. (8)

1.7.3 PostgreSQL

PostgreSQL es un servidor de base de datos relacional de software libre. Es una alternativa de otros sistemas de bases de datos de código abierto, como MySQL, Firebird y MaxDB, así como de sistemas como Oracle o DB2. (9)

Es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional.

Proporciona un gran número de características que normalmente sólo se encontraban en las bases de datos comerciales de alto calibre tales como Oracle.

Cumple la prueba ACID (Atomicity/Atomicidad, Consistency/Consistencia, Integrity/Integridad, Durability/Durabilidad).

PostgreSQL está disponible bajo una Licencia BSD que permite tanto el uso comercial como el no comercial.

Posee interfaces nativas para lenguajes como ODBC, JDBC, C, C++, PHP, PERL, TCL, ECPG; PYTHON y RUBY, además de traer soporte para la herencia y la seguridad de la capa de dispositivo de transportación de datos (SSL, Secure Sockets Layer).

Constituye el gestor de base de datos más utilizado actualmente en la comunidad de software libre después de MySQL aunque presenta una estructura superior a este y brinda muchas más funcionalidad. (10)

Por lo antes expuesto nos decimos a utilizar **PostgreSQL** como el gestor de base de datos idóneo para desarrollar PRIMICIA.

1.8 Herramientas de modelado visual.

Existen herramientas que permiten representar un software en su totalidad antes de codificarlo, mediante diagramas que permiten modelar cada etapa del ciclo de vida del software, y que logran que se utilice un lenguaje común en el equipo de trabajo.

1.8.1 ERwin.

PLATINUM ERwin es una CASE(Computer-Aided Software Engineering: Ingeniería de Software Asistida por Computadoras) empleada en el diseño de base de datos, con la cual se puede hacer un trabajo productivo por las facilidades que brinda para la generación y mantenimiento de aplicaciones de forma sencilla para el diseñador.

ERwin permite realizar el diseño del modelo lógico de los requerimientos de información, así como el diseño del modelo físico, ya con nivel mayor, refinando las características de la base de datos diseñada.

Con esta herramienta es posible visualizar la estructura de la base de datos diseñada, lo cual tiene como ventaja que el diseñador pueda observar en su totalidad el trabajo realizado, para realizar un análisis y si fuera necesario hacer cambios en busca de optimizar el diseño final. Además permite generar, de forma automática, las tablas y el código referente a los stored procedure (procedimientos almacenados) y triggers para los principales tipos de bases de datos.

La migración automática garantiza la integridad referencial de la base de datos, es decir, evita la pérdida de información durante este proceso. ERwin puede hacer una conexión entre dos bases de datos, una diseñada y otra sin diseño, estableciendo una transferencia entre ellas y la aplicación de ingeniería reversa. Mediante esta conexión puede automáticamente generar índices, vistas, tablas, reglas de integridad referencial (llaves foráneas, llaves primarias), restricciones de dominios y campos y valores por defecto. ERwin básicamente soporta bases de datos del tipo relacionales SQL, y es compatible con otras como: Oracle, Microsoft SQL Server, Sybase, Microsoft Access, dBase, FoxPro, Informix, SQL Base y SQL Anyware, entre otras. Con un mismo modelo es posible generar múltiples bases de datos o hacer una conversión de una aplicación de una base de datos a otra. Es compatible con los sistemas operativos de la familia Windows, como Windows 95, Windows 98, Windows XP y Windows NT.

1.8.2 Visual Paradigm.

Constituye una herramienta CASE que utiliza UML, como lenguaje de modelado. Está disponible en varias ediciones: Enterprise, Professional, Community, Standard, Modeler y Personal, que permite escoger la más idónea para el usuario según su necesidad.

Soporta UML 2.1, permite realizar ingeniería tanto directa como inversa, tiene la capacidad de crear el esquema de clases a partir de una Base de Datos y crear la de Base de Datos a partir del esquema de clases.

Es una herramienta colaborativa, soporta múltiples usuarios trabajando sobre el mismo proyecto y brinda la posibilidad de que los mismos vean los cambios hechos en tiempo real; genera la documentación del proyecto automáticamente en varios formatos como Web o Pdf, y permite el control de versiones.

Es multiplataforma y muy útil para la generación de código fuente en PHP.

Por las múltiples ventajas con que cuenta, que es muy fácil de instalar y actualizar, que cuenta con una comunidad que brinda soporte técnico, y que es una herramienta de software libre, se selecciona Visual Paradigm con la herramienta para realizar el modelado visual del presente trabajo.

1.9 Herramientas de desarrollo.

1.9.1 EMS SQL Manager para PostgreSQL.

El EMS SQL Manager es una herramienta gráfica fácil de utilizar para la administración de PostgreSQL. Trabaja con cualquiera de sus versiones hasta la 8.1 y soporta todas las últimas características de PostgreSQL, incluyendo espacios de tablas, nombres de argumentos en funciones y más. El programa ofrece muchas herramientas poderosas para usuarios experimentados, como un Diseñador Visual de Bases de Datos y un Constructor Visual de Consultas. Es la más completa hasta la actualidad. Permite obtener la documentación sobre el diseño de la base de datos y tiene un grupo de herramientas importantes para la importación y exportación de datos. (Veliz Díaz, 2008)

Características:

- Soporte completo para PostgreSQL hasta la versión 8.1.
- Administración y navegación rápida de bases de datos.
- Administración fácil de todos los objetos PostgreSQL.
- Herramientas de manipulación avanzada de datos.

- Administración efectiva de seguridad.
- Excelentes herramientas visuales y de texto para la construcción de consultas.
- Capacidades de exportación e importación de datos.
- Poderoso diseñador visual de bases de datos.
- Modo guiado para labores de mantenimiento.
- Interfaz atractiva.

1.9.2 EMS Data Generator for PostgreSQL

EMS Data Generator for PostgreSQL constituye una poderosa herramienta para generar datos de prueba a tablas de bases de datos PostgreSQL de una vez.

Permite definir las tablas y campos a las que se le desea generar datos, configurar valores de rangos, obtener listas de valores desde consultas SQL.

También te provee una aplicación consola, que te permite generar datos de un "clic" usando plantillas de generación. La interfaz del asistente es muy fácil de usar. Provee de diferentes tipos de generación de datos por cada campo, lista, azar (random), generación incremental de datos y más. Control automático sobre integridad referencial para la generación de datos a tablas vinculadas. Capacidad para configurar valores nulos para ciertos casos. (16)

Esta herramienta será utilizada durante la etapa de validación del diseño de la BD, realizando un llenado voluminoso de la misma, y generando datos para futuras pruebas.

1.9.3 EMS SQL Query for PostgreSQL

Constituye una herramienta que permite rápida y sencilla construcción de consultas SQL a bases de datos PostgreSQL.

Presenta una interfaz gráfica fácil de usar, que permite conectarse a bases de datos PostgreSQL, seleccionar tablas y campos para una consulta al servidor PostgreSQL, y establecer los criterios de selección.

Permite trabajar con varios PostgreSQL a la vez, editar el texto de consulta en el editor, ver los resultados y el tiempo que demoró en realizar la misma.

Soporta datos UTF8, permite trabajar simultáneamente con varias consultas en ventanas separadas, múltiples conexiones con bases de datos, y presenta un historial de las consultas realizadas. (17)

Por la posibilidad que brinda para la realización de consultas, la información que provee al realizarlas y las múltiples ventajas que presenta para su manejo, ha sido seleccionada para la realizar las prueba a la base de datos, utilizando las consultas más frecuentes.

1.10 Replicación de datos:

PRIMICIA se encuentra estructurada en dos subsistemas, el Subsistema de Administración y el Subsistema de Transmisión.

Ambos subsistemas se encuentran en servidores diferentes (aunque en caso que la entidad cliente no cuente con un gran presupuesto es posible utilizar un solo servidor que cumpla con ambas funciones) por lo que una modificación en el Subsistema de Administración debe ser enviado al Subsistema de Transmisión para que este visualice las noticias y los materiales publicados, es por ello que se deberá utilizar la replicación de datos para realizar este proceso.

La réplica de datos es una técnica que permite copiar y distribuir idénticamente las tablas de una base de datos en múltiples bases de datos ubicadas en diferentes nodos de la red. La replicación asegura que los datos correctos estén siempre disponibles en el momento y en el lugar necesario. (11)

La replicación de datos presenta determinadas características:

- *Efectividad*:

  Depende de la forma en la que los datos sean distribuidos y almacenados. A mayor efectividad, mayor será la disponibilidad de datos para ejecutar procesos paralelos.

- *Alta Disponibilidad:*

  Es la razón de tiempo prudente en la que un servicio puede ser accedido. En el mejor de los casos puede ser de un 100%, a pesar de los fallos que se puedan presentar en el servidor, ya que debe existir un servidor adicional que posea alguna técnica de replicación que lo pueda suplantar en caso de ser necesario.

- *Tolerancia a fallos:*

  Garantiza un comportamiento correcto, donde efectivamente pueden existir un número finito de fallos y tipos de fallos.

- *Coordinación*:

  Cada una de las partes que conforman la base de datos, acuerda un consenso para realizar las invocaciones de los servicios a los objetos, que al final de la transacción debe realizarse tal y como fue solicitada, para lo cual debe utilizar algún tipo de ordenamiento. (12)

1.10.1 Tipos de entornos de réplica:

Existen dos tipos de entornos de replica:

Maestro-Esclavo (master-slave): o de solo lectura, que permite a un solo maestro recibir consultas de lectura/escritura, mientras los esclavos solo pueden aceptar consultas de lectura.

Multi-Maestro (multi-master): también llamado par-a-par o la réplica de camino de n, el cual permite múltiples sitios, actuando como pares iguales. Cada sitio, en un ambiente de réplica de multimaestro, es un sitio de maestro, y cada sitio se comunica con otros sitios maestros. Esta capacidad tiene también un severo impacto en el desempeño debido a la necesidad de sincronizar los cambios entre los servidores.

Este tipo de entorno puede ser usado para mantener sitios recuperables ante posibles desastres o caídas, para proveer sistemas con alta disponibilidad y para balancear la carga de consultas a través de las distintas ubicaciones. (13)

1.10.2 Modelos de distribución de datos:

Los modelos de distribución de datos son esencialmente aplicados a cada uno de los entornos antes vistos y estos se clasifican en:

Asincrónico: a menudo llamada almacena-y-reenvía, que captura cualquier cambio local, los almacena en una cola, y, a intervalos regulares, propaga y aplica estos cambios en sitios remotos. Con esta forma de réplica, hay un período de tiempo antes de que todos los sitios alcancen la convergencia de datos.

Sincrónico: también conocida como la réplica en tiempo real, aplica cualquier cambio o ejecuta cualquier procedimiento reproducido en todos los sitios que participan en el ambiente de réplica como parte de una sola transacción. Si el procedimiento falla en cualquier sitio, entonces la transacción entera se anula. La réplica sincrónica asegura la consistencia de datos en todos los sitios en tiempo real.

1.10.3 Técnicas de replicación

Cada modelo de distribución tiene asociada técnicas de replicación las cuales son:

Sincrónica

Confirmación en dos fases (Two Phase Commit)

Permite la sincronización de datos distribuidos. Cada transacción solamente es aceptada si todos los sistemas implicados en la réplica están conectados y listos para recibirla, si al menos uno falla, todo el proceso es anulado.

Asincrónica

Descarga y Recarga (Dump and Reload)

Consiste en hacer un volcado de los datos, copiar la salva para un dispositivo de almacenamiento para luego distribuir la salva por los demás servidores. Esta técnica presenta el inconveniente de que en la mayoría de las ocasiones se consultaban datos que tenían semanas de desactualización, además de que el proceso se realizaba de forma manual.

Instantánea (Snapshot)

Consiste en hacer una instantánea de una tabla en un momento dado y esta “foto” es la que se replica en las demás bases de datos. Aunque mucho más avanzado que la técnica de Descarga y Recarga esta presenta el inconveniente de que las tablas esclavas son tablas de solo lectura. Se recomienda utilizar cuando: la mayoría de los datos no cambian con frecuencia; se replican pequeñas cantidades de datos; los sitios con frecuencia están desconectados y es aceptable un periodo de latencia largo (la cantidad de tiempo que transcurre entre la actualización de los datos en un sitio y en otro).

Transaccional:

En este caso se propaga una instantánea inicial de datos a las bases de datos Esclavas, y después, cuando se efectúan las modificaciones en la base de datos Maestra, las transacciones individuales se propagan a los suscriptores. Almacena las transacciones que afectan a los objetos replicados y propaga esos cambios a los suscriptores de forma continua o a intervalos programados. Al finalizar la propagación de los cambios, todos los suscriptores tendrán los mismos valores que el publicador. Suele utilizarse cuando: se desea que las modificaciones de datos se propaguen a los suscriptores, normalmente pocos segundos después de producirse.

1.10.4 Herramientas de replicación

Existen herramientas destinadas a realizar la réplica de datos, enfocadas al Sistema Gestor de Base de Datos seleccionado: PostgreSQL. Estas aplicaciones se clasifican de acuerdo al entorno de replica donde pueden ser aplicadas.

Para el entorno “Maestro-Esclavo”, la más usada es:

1.10.4.1 Slony I

Slony I constituye un replicador asíncrono donde una única base de datos Maestra, replica a múltiples esclavos, soportando la réplica en cascada y permitiendo a un esclavo ser a su vez maestro para otro servidor.

Slony-I nació de la idea de crear un sistema de replicación que no estuviese vinculado a las versiones específicas de PostgreSQL, que pudiese ser inicializado y detenido en una base de datos existente, sin necesidad de un ciclo dump/reload.

No es un sistema de gestión de red y no tiene ninguna funcionalidad dentro del mismo para detectar un nodo con fallos, ni para promover automáticamente un nodo a un maestro u otro origen de datos.

El mismo incluye requerimientos básicos para replicar grandes bases de datos con un número razonable de sistemas esclavos. “Razonable” en este contexto hace referencia a una docena de servidores. Si el número de servidores crece más allá de eso, el costo prohibitivo de las comunicaciones aumenta, y el incremento de beneficios de tener múltiples servidores se pierde en ese momento.

Utiliza disparadores para determinar las actualizaciones de las tablas, donde un solo "origen" (maestro) se puede replegar a los "suscriptores múltiples" (esclavos) incluyendo suscriptores conectados en cascada.(18)

Slony I realiza una réplica de espejos, exactamente igual al origen de datos, no es posible actualizar los datos a medida que se produce algún cambio en ellos.

Ventajas:

- Funciona en gran cantidad de versiones de PostgreSQL.
- Es fácil de instalar.
- Presenta una alta actividad en la comunidad open source y por ello se encuentra bien documentado.
- Soporta actualización y modificación del esquema en tiempo de ejecución.

Desventajas:

- Soporta pocos nodos.
- No implementa arquitectura multi-maestro.
- No detecta errores ni es capaz de recuperarse de ellos.
- Requiere una red de comunicaciones altamente confiable y de alta tasa de transferencia.

Para el entorno "Multi-maestro", la más usada es:

1.10.4.2 PgCluster

Es un sistema de replicación sincrónico y "multi-maestro" para PostgreSQL

PgCluster consiste en tres tipos de servidores, un servidor para balance de carga, un cluster BD y un servidor de réplica.

Tiene dos funciones principales:

- Compartir carga

La carga de la sesión de las demandas es distribuida. Es efectivo en aplicaciones Web donde existe gran demanda por el número de peticiones.

- Alta disponibilidad

Cuando ocurre un fallo en el Cluster BD, el servidor de balance de carga y el de replicación separan el fallo del sistema, y continúa el servicio que usa el BD restante.

El Cluster BD cuando es reparado puede restaurarse dinámicamente a un sistema, sin detener el servicio. (19)

Los datos son copiados automáticamente a la BD restaurada o añadidos desde otra BD.

Sincroniza el árbol de directorio del Pgsql, cuando un servidor cae y se vuelve a poner en funcionamiento automáticamente inicia la sincronización de datos.

Por la estructura que presenta PRIMICIA se puede concluir que el sistema presentará un entorno de réplica Maestro-Esclavo, ya que el servidor que soportará el Subsistema de Administración, que es donde se realizan todas las modificaciones, replicará hacia el Servidor de Transmisión la información con solo privilegios de lectura, además el Modelo de Distribución será Asíncrono, ya que los cambios se almacenan en una cola, y en intervalos regulares se actualizará los nuevos datos en la base de datos montada sobre el Servidor de Transmisión, además el modelo de replicación que se utilizará será el Transaccional, donde se replicará desde el maestro(base datos de origen) hacia los esclavos, cuando ocurra algún evento definido con anterioridad. La herramienta que se utilizará para realizar la replicación será el Slony I, porque cumple con entorno de réplica y el modelo de distribución que demanda el sistema en cuestión.

1.11 Análisis de otras soluciones existentes:

Existen soluciones que han sido implementadas para que cumplan con funcionalidades muy parecidas a las que tiene conceptualizadas PRIMICIA.

El Canal Informativo Señal3 de la Universidad de las Ciencias Informáticas (UCI), es una de estas soluciones, el cual está orientado a informar a la comunidad universitarias las 24 horas a través de un sistema de teletexto adaptado a las características del centro el cual es transmitido por la red de televisión por cable.

Otra solución es el canal de teletexto de la Plataforma Satelital Cubana, Señal ACN, que va destinado a los cooperantes cubanos que cumple misión internacionalista y a los habitantes de las zonas de silencio del país como sustitución de la prensa plana.

Ambos sistemas centran su desarrollo completamente sobre software propietario, utilizando para el diseño e implementación de su base de datos a Microsoft SQL Server, herramienta potente pero que solo puede ser utilizada por los Sistemas Operativos Windows, esta característica hace que se descarte completamente la estructura de estas dos soluciones, ya que uno de los elementos principales de los que ostenta PRIMICIA es que constituye un sistema implementado sobre software libre para contribuir a la soberanía tecnológica del país.

1.12 Conclusiones

Luego de realizar el primer acercamiento al objeto de estudio, PRIMICIA, y estudiar a fondo las características y elementos con que debe cumplir una base de datos, para este tipo de producto, se arrojaron las siguientes conclusiones:

- Se usará una base de datos de tipo Dinámica, por la posibilidad que provee la misma para modificar y actualizar los datos.
- Se seleccionó un modelo de base de datos relacional, porque favorece la normalización y no es importante el orden de inserción de los datos.
- Se definió PostgreSQL como el SGBD a utilizar, por las múltiples ventajas presentadas con anterioridad.
- Se utilizará un entorno de replicación Maestro-Esclavo con un Modelo de Distribución Asíncrono y con una técnica de replicación basada en disparadores (Triggers).
- Se empleará para la replicación de datos la herramienta Slony I.
- Se utilizará la herramienta CASE Visual Paradigm, para el modelado y diseño de la base de datos.
- Para la validación funcional de la base de datos se utilizará EMS Data Generator for PostgreSQL y EMS SQL Query 2005 for PostgreSQL.

Capítulo 2. Descripción y análisis de la solución propuesta.

2.1 Introducción.

Este capítulo abarca los procesos principales que se llevaron a cabo para el diseñar y desarrollar la base de datos de PRIMICIA. Durante el mismo se muestra una descripción de la integración con otros módulos o sistemas, se realiza una fundamentación de la arquitectura propuesta, se exponen los requisitos funcionales y no funcionales del sistema en cuestión, se describen las clases persistentes, y se muestra el diagrama Entidad-Relación con la descripción de las tablas de la base de datos diseñada.

2.2 Integración de la solución con otros módulos o sistemas.

PRIMICIA se encuentra estructurada por siete módulos, acoplados en dos subsistemas.

Los módulos de Seguridad, Redacción, Editorial, Gestión de Archivos, Reportes y Gestión de Señal, se encuentran en el Subsistema de Administración, y el módulo Transmisión de Señal pertenece al Subsistema de Transmisión.

Todos los módulos se encuentran interrelacionados, y acceden a la base de datos física de manera indirecta ya que lo realizan a través de clases de acceso a datos que provee el Framework Symfony, posibilitando separar el acceso a la base de datos desde la aplicación en sí.

2.3 Descripción de la arquitectura y fundamentación.

La arquitectura de software constituye la base fundamental en el desarrollo de un software, esta favorece a que todos los programadores, diseñadores, y analistas implicados en la creación del software lleguen a un entendimiento común, y todo se desarrolle siguiendo estándares y patrones ya definidos.

Algunas arquitecturas son más favorables utilizar con una determinada tecnología, en el caso de PRIMIICIA, se utilizará el Framework de PHP Symfony, el cual emplea la arquitectura Modelo-Vista-Controlador (MVC), para su implementación.

La arquitectura Modelo-Vista-Controlador, divide su estructura en tres componentes diferentes:

Vista: enmarca la interfaz, la página HTML, el código que provee datos dinámicos a la página; aprovechar la separación de código. Las páginas web suelen contener elementos que se visualizan de igual forma durante toda la navegación: cabeceras de la página, el layout genérico, el pie de página. Generalmente sólo cambia el interior de la página es por ello que se separará la vista en un layout y en una plantilla.

Controlador: responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista. Entre las tareas comunes que realiza este componente se encuentran el manejo de las peticiones del usuario, el manejo de la seguridad, y cargar la configuración de la aplicación.

Modelo: aquí se ubica el Sistema Gestor de Base de Datos y la Lógica de negocios. Es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones.

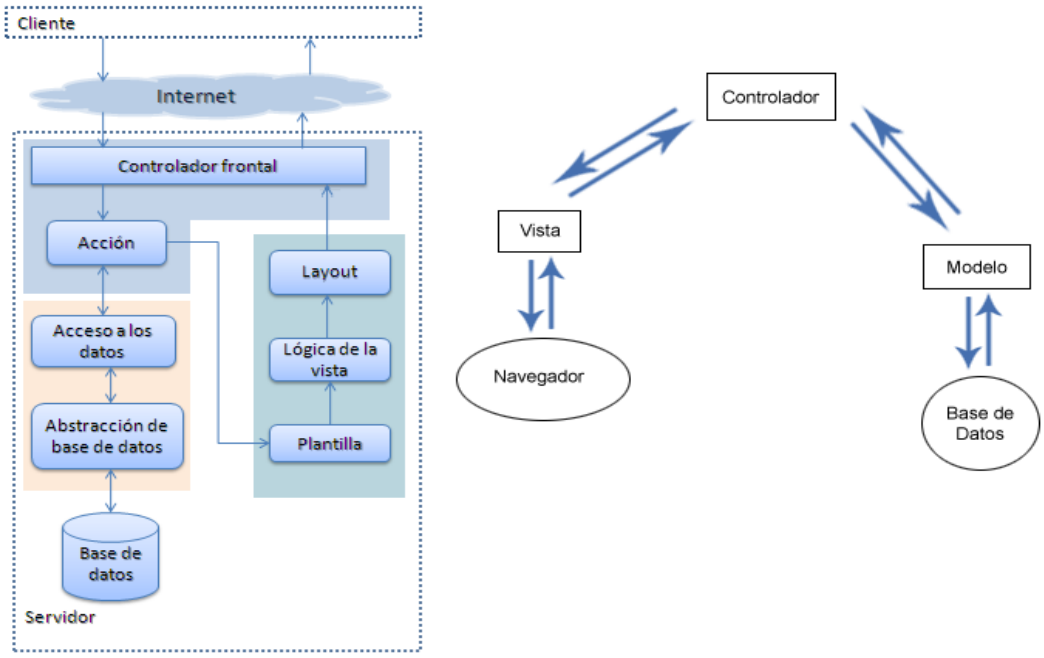

Fig. 2 Proceso de relación entre los componentes en una Arquitectura MVC.

2.4 Análisis de optimización de Querys.

El utilizar una arquitectura MVC posibilita que en la capa del modelo se pueda dividir en la capa de acceso a los datos y en la capa de abstracción de la base de datos. De esta forma, las funciones que acceden a los datos no utilizan sentencias ni consultas que dependen de

una base de datos, sino que utilizan otras funciones para realizar las consultas. Así, si se cambia de sistema gestor de bases de datos, solamente es necesario actualizar la capa de abstracción de la base de datos.

2.5 Requisitos funcionales y no funcionales del sistema.

2.5.1 Requisitos funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir, dicho de otra forma, especifican acciones que el sistema debe ser capaz de realizar.

R1. Autenticación de usuarios:

El usuario para su acceso al sistema deberá autenticarse, lo cual garantizará según su rol las credenciales para el trabajo con el sistema.

R2. Gestionar usuarios:

Se debe dar la posibilidad de registrar nuevos usuarios estableciendo los roles de los mismos, modificar los permisos de determinado usuario y eliminar usuarios del sistema.

R3. Gestionar secciones temáticas:

El sistema debe permitir que se establezcan el orden de las secciones para su visualización, horario en que serán mostradas las informaciones en el canal, además se podrá habilitar o deshabilitar la visualización de las secciones.

R4. Redactar noticias:

Se redactarán las noticias teniendo en cuenta los parámetros establecidos: título, sección temática, música de fondo y pantallas por las que se compondrá la noticia según los distintos formatos definidos (texto, texto-imagen, imagen y video). Para las pantallas se especificarán distintas informaciones requeridas en dependencia del su tipo.

R5 Mostrar las noticias redactadas:

Se deben listar las noticias redactadas por el usuario autenticado en caso que tenga permisos de redacción y que la noticia no haya sido publicada. Para las noticias mostradas se ofrecerán las opciones de mostrar la noticia, modificarla, eliminarla y visualizar las modificaciones que ha sufrido la noticia.

R6. Publicar noticia:

Se mostrará un listado de las noticias redactadas organizadas por secciones temáticas, para cada una se mostrarán las opciones mostrar la noticia, modificar, publicar, archivar, eliminar y visualizar las modificaciones que ha sufrido la noticia. La publicación se debe realizar teniendo en cuenta fecha de inicio y fin de publicación.

R7. Mostrar las noticias publicadas:

Se mostrará un listado de las noticias publicadas organizadas por secciones temáticas y estado de transmisión (espera, transmisión, vencida), para cada noticia se mostrará las opciones mostrar la noticia, modificar, archivar, eliminar y visualizar las modificaciones que ha sufrido la noticia. La publicación se debe realizar teniendo en cuenta fecha de inicio y fin de publicación.

R8. Mostrar modificaciones de las noticias:

El sistema guardará las modificaciones realizadas a las noticias, almacenando los cambios realizados, el autor y la fecha de modificación. Cuando se muestren las noticias el sistema deberá dar la opción de mostrar estas modificaciones, en este caso se visualizarán por parejas los estados por los que ha transcurrido la noticia (estado x – estado anterior), mostrando la información de ambas noticias, su autor y fecha de modificación.

R9. Administrar archivo de noticias:

El sistema contará con un archivo de noticias donde se almacenarán aquellas noticias que decida el editor del canal. Se deberá brindar funcionalidades para administrar este archivo mostrando un listado de las noticias archivadas organizadas por sección temática, ofreciendo las opciones de mostrar, publicar y eliminar las noticias.

R10. Redactar infocintas:

El sistema deberá posibilitar la redacción de infocintas, estableciendo la sección a la que pertenece, el texto, la fecha y hora del período de publicación y determinar si estará o no activada por defecto.

R11. Administrar infocintas:

Se mostrará un listado de las infocintas publicadas de acuerdo a la sección temática a la que pertenezca, posibilitando establecer un orden de visualización de las mismas, habilitar o deshabilitarlas, modificar la información de la infocinta y eliminarla.

R12 Almacenar recursos multimedia:

Se podrá almacenar imágenes, videos y música a través del sistema, contando de esta manera con un archivo de recursos que serán utilizados para la visualización y/o confección de las noticias. Para la acción anterior se deben tener en cuenta los siguientes parámetros para cada tipo de recurso:

Música: país, género y título.

Videos e imágenes: sección temática, palabras claves, fecha y nombre.

R13. Administrar archivo de recursos multimedia:

Se mostrará un listado de los recursos almacenados filtrados de acuerdo al tipo de recurso, brindando las opciones para modificar la información de los recursos, eliminar recursos y visualizarlos.

R14. Reproducir recursos multimedia:

Al seleccionar la opción visualizar recurso en dependencia del tipo se visualizará en caso que sea una imagen o video, o se escuchará la música.

R15. Generar reportes:

Se deberán brindar funcionalidades para la generación de reportes sobre la actividad de los redactores y de las noticias publicadas en el sistema. Para la generación de reportes de redactores se tendrá en cuenta un período determinado por fechas de inicio y fin, permitiendo conocer cuantas y cuales noticias han redactado los mismos en el período especificado. En cuanto a los reportes de noticias publicadas, se realizará una búsqueda atendiendo distintos criterios (Fecha, Temática, Palabras claves, Titulo), resultando un listado de las noticias publicadas según los criterios señalados. Una vez generados los reportes se ofrecerán las opciones de exportar el reporte formato PDF, así como imprimir los mismos.

R16. Transmitir canal:

Deberá verificar el estado de la señal a transmitir y en dependencia de esto visualizará la señal determinada por el Operador de Transmisiones.

R17.Mostrar cartelera:

El sistema deberá visualizar una cartelera del ciclo de trasmisión como inicio de cada ciclo, mostrando para cada noticia la sección temática y el titular, en el orden que serán mostrados posteriormente.

R18. Visualizar noticias:

Deberá mostrar las noticias publicadas teniendo en cuenta su período de publicación y el formato de sus pantallas. Se mostrará la sección temática a la que pertenece la noticia, el titular y el contenido de la pantalla, en el caso de las pantallas de tipo imagen y video se mostrarán los mismos a pantalla completa, ocultando todo el contenido del canal. Para las pantallas de tipo imagen se mostrará el pie de imagen especificado en la redacción de la noticia.

R19. Mostrar información adicional:

Se mostrará información adicional para la cartelera y pantallas de tipo texto y texto-imagen, mostrándose la fecha en formato abreviado y la hora. En el caso específico de las pantallas de texto y texto-imagen se mostrará además el tiempo restante de visualización de la pantalla y titular de la próxima noticia y sección temática.

R20. Reproducir fondo musical:

Se deberá reproducir un fondo musical para el canal el cual estará dado por la música especificada opcionalmente para cada noticia, en caso que no se haya definido tema musical para la noticia entonces se escuchará un tema escogido aleatoriamente de un grupo genérico. En caso que se muestre un video el fondo musical del canal no se escuchará.

R21. Mostrar infocintas:

Se visualizarán las infocintas publicadas teniendo en cuenta su período de publicación y la sección temática a la que pertenece.

R22. Trasmitir video en vivo:

Deberá ser capaz de transmitir la señal de video externa.

R23. Mostrar patrón del canal:

En caso que el canal pase a fuera de servicio se deberá mostrar una imagen de patrón de canal.

2.5.2 Requisitos no Funcionales:

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido y confiable.

Usabilidad

- El usuario podrá utilizar fácilmente los objetivos con los que debe interactuar.
- El uso del diseño debe ser fácil de entender, atendiendo a la experiencia, conocimiento y habilidades del usuario.

Interfaz

Interfaces de usuario

- Establecer una división en secciones temáticas de manera visual y no por bloques de noticias de forma tal que se puedan alternar noticias de diferentes naturalezas sin que por esto se pierda la referencia temática de las mismas.
- Los tipos de pantallas para las noticias se definen de cuatro (4) maneras: solo texto, texto e imagen, solo imagen y video.
- El formato de salida del canal será 16:9 panorámico.
- Uso de video de presentación y despedida de la transmisión.
- Uso de cortinillas para el encadenamiento instantáneo con televisoras.

Interfaces Software

- Se reutilizan las funcionalidades brindadas por el VLC para la creación y transmisión de los videos y música que se visualizará en el canal.

Interfaces de Comunicación

- Se utilizará la red interna de televisión para la transmisión de la señal del canal.

Seguridad

- La información manejada por el sistema estará protegida de acceso no autorizado mediante una integración del dominio de la entidad.
- Solo se accederá a las opciones autorizadas según el rol que desempeña.
- Solo se accederá al sistema de administración desde las computadoras autorizadas.
- La información será objeto de cuidadosa protección contra la corrupción y estado inconsistente.
- Deberán existir mecanismos de chequeo de integridad.
- Se deberán hacer copias de respaldo que puedan restaurar el sistema en caso de fallo crítico o pérdida total de la información.

2.6 Diagrama de clases persistentes.

El diagrama lógico de datos o diagrama de clases persistentes muestra la capacidad de un objeto de mantener su valor en el espacio y en el tiempo.

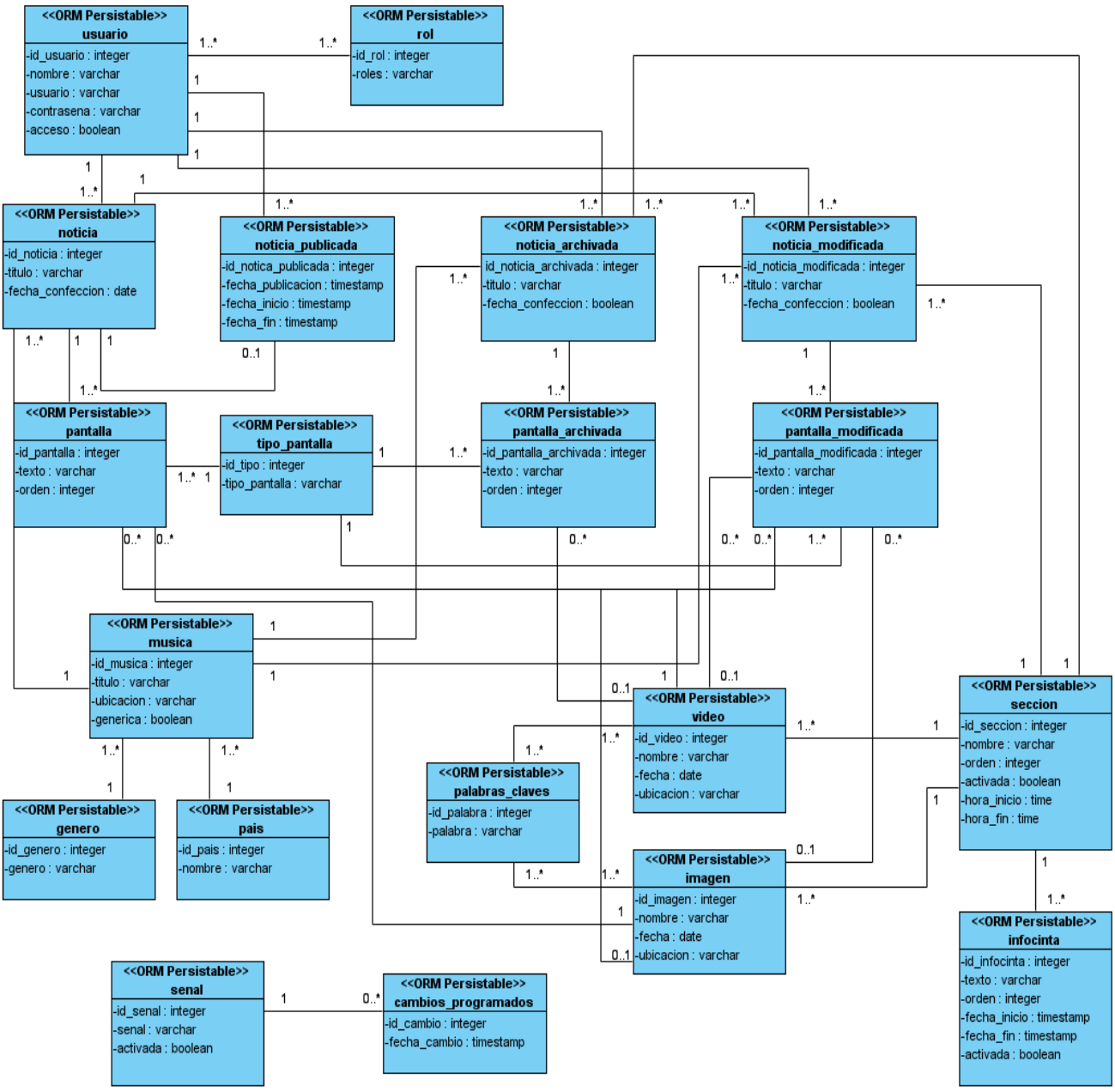

Fig. 3 Diagrama de clases persistentes.

2.7 Descripción de las clases.

La descripción de las clases muestran los atributos con el tipo de dato que representan cada uno para un mejor entendimiento.

Nombre: usuario	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_usuario	Integer
nombre	Varchar
usuario	Varchar
contraseña	Varchar
acceso	Boolean

Nombre: rol	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_rol	Integer
rol	Varchar

Nombre: noticia	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_noticia	Integer
título	Varchar
fecha_confección	Date

Nombre: noticia_publicada	
Tipo de clase: entidad.	
Atributo	**Tipo**
Id_noticia_publicada	Integer
fecha_publicación	Date
fecha_inicio	Date
fecha_fin	Date

Nombre: noticia_archivada	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_noticia_archivada	Integer
título	Varchar
fecha_confección	Date

Nombre: noticia_modificada	
Tipo de clase: entidad.	
Atributo	**Tipo**
Id_noticia_modificada	Integer
título	Varchar
fecha_confección	Date

Nombre: pantalla	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_pantalla	Integer
texto	Varchar
orden	Integer

Nombre: tipo_pantalla	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_pantalla	Integer
tipo_pantalla	Varchar

Nombre: pantalla_archivada	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_pantalla_archivada	Integer
texto	Varchar
orden	Integer

Nombre: pantalla_modificada	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_pantalla_modificada	Integer
texto	Varchar
orden	Integer

Nombre: música	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_musica	Integer
título	Varchar
ubicación	Varchar
genérica	Boolean

Nombre: genero	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_genero	Integer
género	Varchar

Nombre: país	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_país	Integer
nombre	Varchar

Nombre: sección	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_sección	Integer
nombre	Varchar
orden	Integer
activada	Boolean
hora_inicio	Time
hora_fin	Time

Nombre: palabras_claves	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_palabras	Integer
palabra	Varchar

Nombre: video	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_video	Integer
nombre	Varchar
fecha	Date
ubicación	Varchar

Nombre: imagen	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_imagen	Integer
nombre	Varchar
fecha	Date
ubicación	Varchar

Nombre: señal	
Tipo de clase: entidad.	
Atributo	**Tipo**
Id_señal	Integer
señal	Varchar
activada	Boolean

Nombre: cambios_programados	
Tipo de clase: entidad.	
Atributo	**Tipo**
id_cambios	Integer
fecha_cambio	Timestamp

Nombre: infocinta	
Tipo de clase: entidad.	
Atributo	**Tipo**
Id_infocinta	Integer
texto	Varchar
orden	Integer
fecha_inicio	Time
fecha_fin	Time
activada	Boolean

2.8 Diseño de la BD

2.8.1 Diagrama Entidad Relación de la BD.

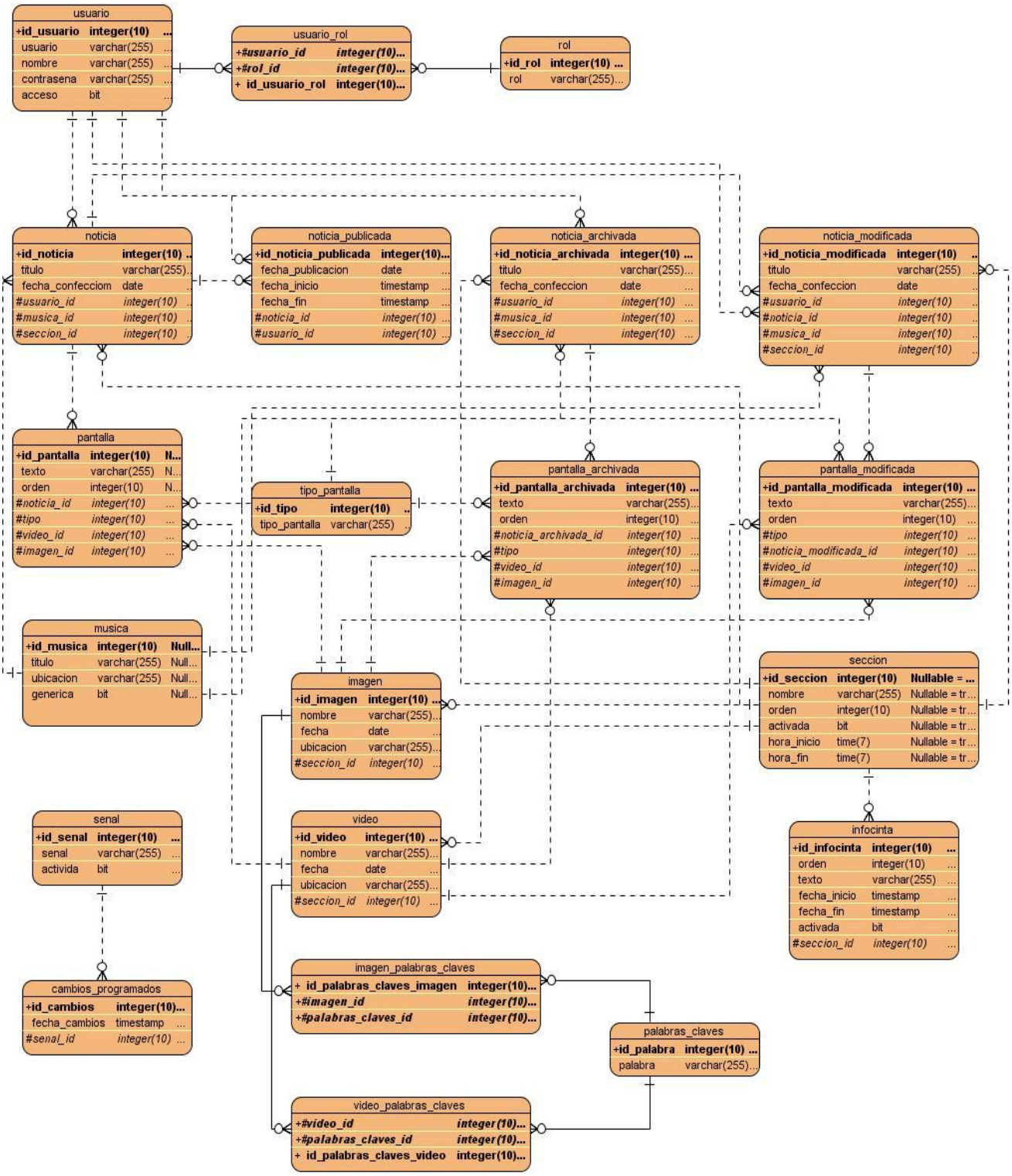

Fig. 4 Diagrama entidad-relación.

2.8.2 Descripción de las tablas.

Nombre: usuario		
Descripción: Esta tabla contiene los usuarios que pueden acceder al sistema		
Atributo	**Tipo**	**Descripción**
id_usuario	integer(10)	Identificador del usuario
nombre	varchar(255)	Nombre del usuario
usuario	varchar(255)	Usuario para acceder al sistema
contraseña	varchar(255)	Contraseña para acceder al sistema
acceso	bit	Determina si tiene permisos para acceder

Nombre: rol		
Descripción: Esta tabla contiene los roles que pueden tener los usuarios		
Atributo	**Tipo**	**Descripción**
id_rol	integer(10)	Identificador del rol
rol	varchar(255)	Nombre del rol

Nombre: usuario_rol		
Descripción: Esta es la tabla que se genera por la relación entre usuario y rol		
Atributo	**Tipo**	**Descripción**
id_usuario_rol	integer(10)	Identificador del usuario rol
usuario_id	integer(10)	Identificador del usuario
rol_id	integer(10)	Identificador del rol

Nombre: música		
Descripción: Esta tabla contiene todos los datos de los temas musicales.		
Atributo	**Tipo**	**Descripción**
id_musica	integer(10)	Identificador de la música
titulo	varchar(255)	Título de la música
ubicación	varchar(255)	Ubicación física de la música
género_id	integer(10)	Identificador del género asociado

Nombre: infocinta		
Descripción: Esta tabla contiene las infocintas que se mostrarán en cada sección		
Atributo	**Tipo**	**Descripción**
id_infocinta	integer(10)	Identificador de la infocinta
orden	integer(10)	Orden en que se mostrarán
texto	varchar(255)	Texto de la infocinta
fecha_inicio	date	Fecha de inicio de publicación de la infocinta
fecha_fin	date	Fecha de fin de publicación de la infocinta
activada	bit	Determina si se mostrará
sección_id	integer(10)	Identificador de la sección asociada

Nombre: noticia		
Descripción: Esta tabla contiene todas las noticias del sistema		
Atributo	**Tipo**	**Descripción**
id_noticia	integer(10)	Identificador de la noticia
titulo	varchar(255)	Título de la noticia
fecha_confección	date	Fecha en que se redactó la noticia
usuario_id	integer(10)	Identificador del usuario asociado
sección_id	integer(10)	Identificador de la sección asociada
música_id	integer(10)	Identificador de la música asociada

Nombre: sección		
Descripción: Esta tabla contiene todas las secciones del sistema		
Atributo	**Tipo**	**Descripción**
id_sección	integer(10)	Identificador de la sección
nombre	varchar(255)	Nombre de la sección
orden	integer(10)	Orden en que se encuentran las secciones
activada	bit	Determina si se mostrará
hora_inicio	time(7)	Hora en que comienza a mostrarse la sección
hora_fin	time(7)	Hora en que termina de mostrarse la sección

Nombre: pantalla		
Descripción: Esta tabla contiene las pantallas de las noticias		
Atributo	**Tipo**	**Descripción**
id_pantalla	integer(10)	Identificador de la pantalla
noticia_id	integer(10)	Identificador de la noticia
texto	varchar(255)	Texto de la pantalla
tipo	integer(10)	Tipo de pantalla
video_id	integer(10)	Identificador del video asociado
imagen_id	integer(10)	Identificador de la imagen asociado
orden	integer(10)	Orden en que se mostrarán las pantallas

Nombre: video		
Descripción: Esta tabla contiene todos los datos de los videos		
Atributo	**Tipo**	**Descripción**
id_video	integer(10)	Identificador del video
nombre	varchar(255)	Nombre del video
fecha	date	Fecha relaciona con el video
ubicación	varchar(255)	Ubicación física del video
palabra_id	integer(10)	Identificador de la palabra clave asociada

Nombre: imagen		
Descripción: Esta tabla contiene todos los datos de las imágenes		
Atributo	**Tipo**	**Descripción**
id_imagen	integer(10)	Identificador de la imagen
nombre	varchar(255)	Nombre de la imagen
fecha	date	Fecha relacionada con la imagen
ubicación	varchar(255)	Ubicación física de la imagen
palabra_id	integer(10)	Identificador de la palabra clave asociada

Nombre: genero		
Descripción: Esta tabla contiene los géneros musicales		
Atributo	**Tipo**	**Descripción**
id_genero	integer(10)	Identificador del género musical
genero	varchar(255)	Nombre del género musical

Nombre: país		
Descripción: Esta tabla contiene los países a los que puede pertenecer una música		
Atributo	**Tipo**	**Descripción**
id_país	integer(10)	Identificador del país
nombre	varchar(255)	Nombre del país

Nombre: palabras_claves		
Descripción: Esta tabla contiene palabras claves asociadas a los videos y las imágenes		
Atributo	**Tipo**	**Descripción**
id_palabras	integer(10)	Identificador de la palabra clave
palabra	varchar(255)	Palabra clave

Nombre: palabras_claves_imagen		
Descripción: Esta es la tabla que se genera por la relación entre imagen y palabras claves		
Atributo	**Tipo**	**Descripción**
id_palabras_imagen	integer(10)	Identificador de palabras claves imagen
imagen_id	integer(10)	Identificador de la imagen asociada
palabras_claves_id	integer(10)	Identificador de palabras claves asociadas

Nombre: palabras_claves_video		
Descripción: Esta es la tabla que se genera por la relación entre video y palabras claves		
Atributo	**Tipo**	**Descripción**
id_palabras_video	integer(10)	Identificador de palabras claves video
video_id	integer(10)	Identificador del video asociado
palabras_claves_id	integer(10)	Identificador de palabras claves asociadas

Nombre: noticias_archivadas		
Descripción: Esta tabla contiene las noticias archivadas		
Atributo	**Tipo**	**Descripción**
id_noticia	integer(10)	Identificador de la noticia archivada
titulo	varchar(255)	Título de la noticia archivada
fecha_confección	date	Fecha en que se confeccionó la noticia
usuario_id	integer(10)	Identificador del usuario asociado
sección_id	integer(10)	Identificador de la sección asociada
música_id	integer(10)	Identificador de la música asociada

Nombre: pantalla_archivada		
Descripción: Esta tabla contiene las pantallas de las noticias archivadas		
Atributo	**Tipo**	**Descripción**
id_pantalla	integer(10)	Identificador de la pantalla archivada
noticia_id	integer(10)	Identificador de la noticia archivada asociada
texto	varchar(255)	Texto de la pantalla
tipo	integer(10)	Tipo de pantalla
video_id	integer(10)	Identificador del video asociado
imagen_id	integer(10)	Identificador de la imagen asociada
orden	integer(10)	Orden en que se mostrarán las pantallas

Nombre: cambios_programados		
Descripción: Esta tabla contiene los cambios de señal		
Atributo	**Tipo**	**Descripción**
id_cambios	integer(10)	Identificador de los cambios programados
fecha_cambio	timestamp	Fecha en que realizará el cambio
señal_id	integer(10)	Identificador de la señal asociada al cambio

Nombre: señal		
Descripción: Esta tabla contiene las señales que se pueden transmitir		
Atributo	**Tipo**	**Descripción**
id_señal	integer(10)	Identificador de la señal
señal	varchar(255)	Nombre de la señal
activada	bit	Determina si se mostrará

Nombre: tipo_pantalla		
Descripción: Esta tabla contiene los tipos de pantallas		
Atributo	**Tipo**	**Descripción**
id_tipo	integer(10)	Identificador del tipo de pantalla
tipo_pantalla	varchar(255)	Nombre del tipo de pantalla

Nombre: pantalla_modificada		
Descripción: Esta tabla contiene las pantallas de las noticias modificadas		
Atributo	**Tipo**	**Descripción**
id_pantalla_modificada	integer(10)	Identificador de la pantalla modificada
texto	varchar(255)	Texto de la pantalla modificada
orden	integer(10)	Orden en que se mostrarán las pantallas
tipo_pantalla_id	integer(10)	Tipo de pantalla asociada
noticia_modificada_id	integer(10)	Identificador de la noticia modificada asociada
video_id	integer(10)	Identificador del video asociado
imagen_id	integer(10)	Identificador de la imagen asociada

Nombre: noticias_publicadas		
Descripción: Esta tabla contiene las noticias publicadas		
Atributo	**Tipo**	**Descripción**
id_noticia_publicada	integer(10)	Identificador de la noticia publicada
noticia_id	integer(10)	Identificador de la noticia
fecha_inicio	timestamp	Fecha de inicio de publicación
fecha_fin	timestamp	Fecha de fin de publicación
fecha_publicacion	timestamp	Fecha en que se publicó la noticia
usuario_id	integer(10)	Identificador del usuario asociado

Nombre: noticias_modificadas		
Descripción: Esta tabla contiene las noticias modificadas		
Atributo	**Tipo**	**Descripción**
id_noticia_modificada	integer(10)	Identificador de la noticia modificada
titulo	varchar(255)	Título de la noticia modificada
fecha_confección	date	Fecha en que se confeccionó la noticia
usuario_id	integer(10)	Identificador del usuario asociado
sección_id	integer(10)	Identificador de la sección asociada
música_id	integer(10)	Identificador de la música asociada

2.9 Conclusiones

Este capítulo engloba el resultado del diseño de la base de datos, que fue arrojado de la necesidad planteada por los requisitos funcionales y no funcionales descritos.

Se presentó el diagrama de clases persistentes, sus interrelaciones y la descripción de cada clase que dieron lugar al diagrama entidad-relación compuesto por las tablas que componen el diseño final de la base de datos. Así como la descripción de la arquitectura sobre la que se implementará el sistema y la interrelación entre los módulos y subsistemas de PRIMICIA.

Capítulo 3. Validación del diseño realizado.

3.1 Introducción

En este capítulo se expone el proceso de validación del diseño de la Base de Datos desarrollada. Se divide en una validación teórica y una funcional. Se miden elementos como la integridad, la normalización de la base de datos, la redundancia de datos y la seguridad de la base de datos, así como el funcionamiento de la base de datos, bajo pruebas de carga intensiva.

3.2 Validación teórica del diseño

El diseño de la base de datos, constituye un proceso primordial, pero luego de este, se garantizan aspectos como la consistencia, integridad y seguridad de los datos, así como detalles importantes en el proceso de desarrollo de una base de datos.

3.2.1 Integridad

El concepto de integridad de la base de datos hace referencia a la validez y coherencia de los datos almacenados.

La integridad se suele expresar en términos de restricciones, que son reglas de coherencia, que la base de datos no puede violar. Las restricciones pueden aplicarse a los elementos de datos contenidos en un único registro o aplicarse a las relaciones entre registros.

La integración permite que el Administrador de la base de datos defina en el SGBD las restricciones de integridad.

Dos pasos importantes en el diseño de las tablas son la identificación de valores válidos para una columna y la determinación de cómo forzar la integridad de los datos en la columna. La integridad de datos pertenece a una de las siguientes categorías:

- Integridad de entidad

La integridad de entidad pretende que cada entidad que se guarda en la base de datos sea identificable de un modo único, es decir, que se evite la información redundante.

Cada entidad almacenada en la Base de Datos, debe tener una clave principal.

Una clave principal constituye un hecho, o grupo de hechos, que distinguen esa entidad del resto de entidades que comparten un conjunto de hechos, o sea cada registro de una tabla debe tener un campo que lo identifique de modo exclusivo respecto al resto presente en esa tabla.

Para elegir una clave principal se debe valorar los actuales y futuros datos que puede presentar, porque puede darse el caso que una columna actualmente sirva como clave principal, y que en un futuro se repitan valores en ese campo, siendo necesario utilizar otra columna como clave principal o crear una clave compuesta.

Definir correctamente el identificador principal para cada registro, es de vital importancia, ya que esto representa la principal herramienta que utiliza el servidor de base de datos para seleccionar la información que se necesite, evitando la duplicidad de registros, y el lanzamiento de excepciones por errores en los registros.

Para velar por la integridad de entidad de los datos de la base de datos, se definió una llave primaria para cada tabla, de tipo SERIAL (autoincremental) y la misma no permite valores nulos.

- Integridad de dominio

Permite controlar la información que se guarda en la base de datos. Dominio se define como un conjunto de normas de negocio que gestionan la disponibilidad de datos en una determinada columna de una tabla.

Existe integridad de dominio básica, como no poder introducir letras en campos donde se va a almacenar números.

Mientras más sean las reglas que se definan mejor será el funcionamiento de la base de datos.

Estas normas o reglas de integridad de dominio pueden indicar que campos son necesarios tener obligatoriamente con valores, para que la base de datos no tenga datos sin conectar en el caso de tener relaciones o dependencias entre tablas.

Algunas herramientas que asegurar la integridad de dominio son: definir los tipos de datos correctamente para cada campo, definir campos no nulos (NOT NULL) para evitar resultados inconsistentes, definir reglas (CHECK), que pueden ser aplicadas a nivel de tabla y columna y que restringen los valores durante las inserciones y modificaciones, etc.

- Integridad referencial

Se define para asegurarse que las filas relacionadas entre tablas, no desaparezca ni varíe, cuando se lleve a cabo una modificación en los datos. Con esta integridad se limita la actividad que puede realiza un usuario sobre la base de datos. Por lo tanto, con la misma, se da a comprender que existen relaciones entre tablas que deben permanecer invariables sea cual sea la actividad sobre ellas.

Para mantener esta integridad en la base de datos se utilizó: las llaves foráneas o externas, las cuales obligan a que los valores introducidos en las columnas marcadas por esta restricción correspondan a valores en la tabla referenciadas, así como permite realizar acciones en caso de actualización o eliminación de los valores a los que se hacen referencia, para esto, se definió que al realizar la actualización o eliminación de datos, las mismas se realizarían en cascada,

3.2.2 Normalización de la Base de datos

La teoría de la normalización se basa en la necesidad de encontrar una representación del conjunto de relaciones, que en el proceso de actualización sea más adecuada.

Al aplicarla se obtienen estructuras de datos eficientes que evitan anomalías de actualización, inserción y eliminación. (15)

Para realizar la normalización de la base de dato, se fue comprobando que cada relación (tabla) cumpliese con un conjunto de reglas basadas en las dependencias funcionales y las claves primarias.

Cada paso ejecutado respondió a una forma normal:

3.2.2.1 Primera Forma Normal

La Primera Forma Normal (1FN), constituye el primer nivel en el proceso de normalización, y representa la más elemental de todas.

Una tabla va estar en 1FN si el valor que contiene un atributo de un registro es único y elemental, o sea en cada atributo solo se puede incluir un dato, aunque sea compuesto, pero no se puede incluir una lista de datos. Por lo tanto en esta etapa se eliminaran los atributos multievaluados, los atributos compuestos y sus combinaciones.

Poner la BD en la 1FN, hace que se eliminen los grupos repetitivos, que son los atributos o conjuntos de atributos que tienen múltiples valores para cada tupla de la relación.

3.2.2.2 Segunda Forma Normal

Una relación está en Segunda Forma Normal (2FN) si, y sólo si, está en 1FN y, además todas las columnas que no son llave son completamente dependientes de la llave primaria (PK), es decir, establece que todas las dependencias parciales se deben eliminar y separar dentro de sus propias tablas. Una dependencia parcial es un término que describe a aquellos datos que no dependen de la llave primaria de la tabla para identificarlos.

Para eliminar las dependencias parciales de las claves primarias, se eliminaron los atributos que son funcionalmente dependientes de cada relación y se colocaron en una nueva relación con una copia de su determinante (los atributos de la clave primaria de los que dependen). La 2FN se aplicó a las relaciones que tenían claves primarias compuestas por dos o más atributos.

Una vez alcanzado el nivel de la Segunda Forma Normal, se controló la mayoría de los problemas de lógica. Se podían insertar registros sin un exceso de datos en la mayoría de las tablas, de esta forma se evitan anomalías a la hora de realizar actualizaciones.

3.2.2.3 Tercera Forma Normal.

Una tabla está en Tercera Forma Normal (3FN) si está en 2FN y si todos sus atributos no llaves son independientes de cualquier otro atributo no llave primaria, es decir, no hay dependencias transitivas. Una dependencia transitiva es aquella en la cual las columnas que no son llave son dependientes de otras columnas que tampoco son llave.

Una tabla está normalizada en esta forma si todas las columnas que no son llave son funcionalmente dependientes por completo de la llave primaria y no hay dependencias transitivas.

Cuando las tablas están en la Tercera Forma Normal se previenen errores de lógica cuando se insertan o eliminan registros. Cada columna en una tabla está identificada de manera única por la llave primaria, y no debe haber datos repetidos. Esto provee un esquema limpio y elegante, que es fácil de trabajar y expandir.

Después de este proceso la Base de Datos de la Plataforma de Televisión Informativa, PRIMICIA, ha quedado normalizada hasta la 3FN,

La Base de Datos está en 1FN puesto que se puede asegurar que cada tupla contiene exactamente un valor para cada atributo de las tablas de la BD. También se cumple que los esquemas de relación están en 2FN, porque se encuentran en 1FN, y además todos los

atributos que no son claves en las tablas, dependen totalmente de la clave primaria. Y finalmente se puede plantear que se encuentra en 3FN, porque está en 2FN, y además no existen dependencias transitivas entre llaves candidatas y atributos no primos.

3.2.3 Análisis de redundancia de información

La redundancia de información, es un término que se utiliza para hacer referencia a la presencia de datos repetidos en la base de datos. Esto debe evitarse, ya que la redundancia dificulta la tarea de modificación de datos, y es el motivo más frecuente de inconsistencia de datos. Además requiere un mayor espacio de almacenamiento, que influye en mayor coste y mayor tiempo de acceso a los datos.

La normalización es una técnica que elimina en gran medida esta dificultad, aunque en base de datos grandes es imposible eliminarla al 100%.

En el caso de la base de datos de PRIMICIA, al normalizarla, se eliminó completamente la presencia de datos repetidos innecesariamente.

3.2.4 Análisis de la seguridad de la base de datos

La seguridad constituye una de las características más importante por la que se debe velar al diseñar una base de datos. Las bases de datos se encuentran en constante riesgo de ser penetradas por la gran información que almacenan y por la importancia de las mismas.

En PRIMICIA, los datos que se almacenan en la BD constituyen la base fundamental del funcionamiento del sistema, ya que una pérdida de los mismos ocasionaría un paro en la transmisión constante de las noticias funcionalidad principal de la Plataforma de Televisión Informativa.

Como primer punto, hay que custodiar que a la base de datos solo tenga acceso el personal autorizado, para ello PostgreSQL, permite realizar configuraciones para controlar los permisos de los usuarios.

En el archivo pg_hba.conf se permite configurar los permisos de conexión para los host y los usuarios de la o las Bases de Datos, definiéndose que direcciones de IP tendrán acceso, a cuál o a cuáles Bases de Datos, y el modo en que se conectarán (conexión sin contraseña, o validando el usuario y la contraseña para conectarse).

En el archivo pg_ident.conf, se permite configurar que usuarios tendrán acceso a determinadas Bases de Datos.

Así como se permite configurar dentro de la Base de Datos los accesos a las tablas, utilizando los comandos GRANT y REVOKE para permitir o denegar los permisos, respectivamente.

Como seguridad definida por el sistema, se definió dar control total de todas las tablas de la Base de Datos solamente a los usuarios que respondieran al Rol de Jefe del Canal, y este usuario será el único que podrán insertar, modificar o eliminar los nuevos usuarios.

Los roles que se definen para cada usuario, protegen la información almacenada, ya que los usuarios registrados tienen los permisos restringidos, es decir solo podrán acceder a las tablas de la base de datos que respondan al rol que desempeñan.

Para que no exista pérdida de información PostgreSQL permite realizar salvas (backups) automáticas o a petición del cliente, para ello utiliza el comando pg_dump el cual indica al SGBD que va a realizar la salva de la Base de Datos. Se le ha de proveer una serie de parámetros, el nombre de la Base de Datos, el usuario y contraseña para conectarse a la misma, el nombre que tomará el archivo de salva y donde se guardará la misma. Las desventajas de este proceso residen en la lentitud que puede alcanzar con Base de Datos de gran volumen y en el consumo de recursos del sistema.

Si se desea restaurar la salvas, se utiliza el comando pg_restore, indicándole la salva que se desea restaurar.

3.3 Validación funcional

Para comprobar que la Base de Datos del sistema, cumple con los requisitos funcionales definidos, es necesario realizarle pruebas, antes de dar por terminado el proceso de diseño y creación de la misma.

El propósito de estas pruebas es simular una carga de producción real y observar cómo se comporta la base de datos bajo cargas intensivas. Esto permite solucionar los problemas de rendimiento, antes de poner la Base de Datos en marcha. El objetivo general es ejecutar las consultas que se supone que sean las más utilizadas y percibir como responde el sistema a ellas.

3.3.1 Descripción de las pruebas realizadas.

Las pruebas de Volumen centran su trabajo en poblar a la base de datos de grandes cantidad de información para determinar si existe algún momento donde la misma alcance sus límites de almacenamiento y cause fallas en el sistema, identificándose así la carga máxima que puede manejar la base de datos en un período dado.

Para el llenado voluminoso de la misma, que aunque no proyectan un comportamiento real, al menos es una aproximación para comprobar la calidad del trabajo realizado, se utilizó la herramienta EMS Data Generator 2005 for PostgreSQL, el mismo permite generar datos para una o varias tablas a la vez, además permite definir el rango de valores admisibles para cada campo, así como brinda la posibilidad de generar los datos a través de consultas SQL o de escogerlos por medio de listas de valores. El EMS Data Generator respeta la integridad referencial, ya que genera los datos que provienen de otras tablas para evitar errores.

Se generó un volumen de 16 500 datos para las tablas cambios_programados, imagen, infocintas, música, noticia, noticia_archivada, noticia_modificada, noticia_publicada, país, palabras_claves, palabras_claves_imagen, palabras_claves_video, pantalla, pantalla_modificada, pantalla_archivada, rol_usuario, usuario y video, en un intervalo de 0,52 segundos.

Luego se procede a comprobar el comportamiento de la base de datos frente a las consultas más frecuentes que se realizarán a la misma, para registrar el tiempo de respuesta a ellas:

- Listar los usuarios registrados en el sistema:

1. **SELECT** COUNT(usuario.id_usuario)

   FROM usuario

   WHERE usuario.acceso=**TRUE**

2. **SELECT** usuario.id_usuario, usuario.nombre, usuario.usuario, usuario.acceso, usuario.contrasena, **UPPER**(usuario.nombre)

   FROM usuario

   WHERE usuario.acceso=**TRUE**

   ORDER BY UPPER(usuario.NOMBRE)

Para un volumen de 434 tuplas se generaron 18 resultados en un tiempo de 31 ms y para un volumen de 20 206 se arrojaron 482 resultados en un tiempo de 376 ms.

- Listar las noticias redactadas del usuario que está logueado en el sistema:

1. **SELECT** noticia.id_noticia, noticia.titulo, noticia.fecha_confeccion, noticia.usuario_id, noticia.musica_id, noticia.seccion_id

   FROM noticia

   WHERE noticia.usuario_id=17

   ORDER BY noticia.id_noticia **ASC**

2. **SELECT** noticia_publicada. id_noticia_publicada, noticia_publicada.fecha_publicación, noticia_publicada.fecha_inicio, noticia_publicada.fecha_fin, noticia_publicada.usuario_id, noticia_publicada.noticia_id

   FROM noticia_publicada

   WHERE noticia_publicada.usuario_id=17

Para un volumen de 434 tuplas se generaron 5 resultados en un tiempo de 156 ms y para un volumen de 20 206 se arrojaron 12 resultados en un tiempo de 407 ms

- Listar las noticias archivadas por secciones.

1. **SELECT** seccion.id_seccion, seccion.nombre, seccion.orden, seccion.activada, seccion.hora_inicio, seccion.hora_fin

   FROM seccion

   ORDER BY seccion.orden **ASC**

3. **SELECT** COUNT(noticia_archivada.id_noticia_archivada)

   FROM noticia_archivada

   WHERE noticia_archivada.seccion_id=0

Para un volumen de 434 tuplas se generaron 17 resultados en un tiempo de 266 ms y para un volumen de 20 206 se arrojaron 17 resultados en un tiempo de 375ms

- Cambiar la señal de transmisión:

1. **SELECT** senal.id_senal, senal.senal, senal.activada

   FROM senal

   WHERE senal.activada=**TRUE**

2. **SELECT** senal.id_senal, senal.senal, senal.activada

   FROM senal

   ORDER BY senal.id_senal **ASC**

Para un volumen de 434 tuplas se generaron 2 resultados en un tiempo de 47 ms y para un volumen de 20 206 se arrojaron 3 resultados en un tiempo de 63ms

- Listar los cambios programados:

1. **SELECT** senal.id_senal, senal.senal, senal.activada

   FROM senal

   ORDER BY senal.id_senal **ASC**

2. **SELECT** COUNT(cambios_programados.id_cambios)

   FROM cambios_programados

3. **SELECT** cambios_programados.id_cambios, cambios_programados.fecha_cambio, cambios_programados.senal_id

   FROM cambios_programados

   ORDER BY cambios_programados.fecha_cambio **ASC**

Para un volumen de 434 tuplas se generaron 2 resultados en un tiempo de 63 ms y para un volumen de 20 206 se arrojaron 1000 resultados en un tiempo de 74

Una vez comprobado el comportamiento de la base de datos, frente a volúmenes elevados de datos, se puede expresar, que la misma, procesa los datos de manera eficiente, en un intervalo de tiempo proporcional a los datos que almacena.

Los resultados son alentadores, pero no deben ser del todo confiables, ya que la mejor prueba que se le puede realizar a cualquier resultado informático lo constituye la interacción directa del usuario, de aquí los procesos de mantenimiento de software y la realización de versiones del mismo buscan mejorar o resolver, posibles errores que se detecten durante su vida útil.

3.4 Conclusiones:

Este capítulo se centró en documentar el proceso de validación funcional y teórica de la base de datos. Se expusieron los mecanismos para velar por la integridad de los datos, así como el proceso de normalización hasta la Tercera Forma Normal, que dio lugar a la eliminación total de la redundancia de información.

Se analizó la seguridad que debía presentar la base de datos, configurando los archivos de PostgreSQL, limitando el acceso por IPs o validando a los usuarios, así como cumpliendo los elementos de seguridad propios de los requisitos funcionales del sistema.

Para la validación funcional se procedió a realizar pruebas de cargas intensiva, realizando un llenando voluminoso a la base de datos y comprobándose el tiempo de respuesta de la base de datos a las consultas más frecuentes.

Conclusiones

Con el desarrollo de este trabajo se ha dotado a PRIMICIA de una estructura para almacenar la información correspondiente a las noticias, y la administración y gestión de las mismas, dando respuesta así al problema que dio pie a este trabajo de diploma.

Se dio cumplimiento a los objetivos generales y específicos: se diseñó la base de datos para PRIMICIA luego de modelarla, implementarla y comprobar que la misma, garantizaba la seguridad de los datos.

Se investigó y se seleccionó el modelo, el tipo de base de datos, y el Sistema Gestor de Base de Datos más idóneos a utilizar en un sistema como PRIMICIA.

Se evaluó la estrategia de replicación que cumpliese con las características de la arquitectura física de PRIMICIA.

Se analizó el funcionamiento de la Base de Datos, y la misma respondió satisfactoriamente a todos los casos de pruebas.

Se puede concluir planteando que este trabajo ha dado lugar a una Base de Datos eficiente que vela por la integridad y la consistencia de los datos y que da cumplimento a los requerimientos funcionales que demanda PRIMICIA.

Bibliografía referenciada:

1. **Dell'Ordine, José Luis.** Historia de la comunicación. [En línea] [Citado el: 18 de Enero de 2009.] http://www.rrppnet.com.ar/hiscomunicacion.htm.

2. **Márques, Mercedes.** *Apuntes de Ficheros y Bases de Datos.* Castellón de la Plana, España: Ingeniería Técnica en Informática de Gestión de la Universidad Jaume I, 2001.

3. **Hansen, Gary W. y Hansen, James V.** *Diseño y Administración de Bases de Datos.* s.l. : Prentice Hall.

4. **Moreno Barajas, Matilde.** GestioPolis. *EL SERVICIO DE CONSULTA DINÁMICA.* [En línea] [Citado el: 23 de Enero de 2009.] www.gestiopolis.com/administracion-estrategia/servicio-de-consulta-olap-bases-de-datos.htm.

5. **Elmasri, Navathe.** *Fundamentals of database systems.* s.l. : Addison-Wesley, 2000.

6. **Ruiz, Marlon.** *Introducción a los Sistemas de Base de Datos.*

7. **Centro de desarrollo de SQL Server.** [En línea] [Citado el: 19 de Enero de 2009.] http://msdn.microsoft.com/es-es/sqlserver/default.aspx.

8. **Rodríguez, Héctor Julio.** *Historia de las Bases de Datos Sistematizadas en Ciencia de la Información.* 2006.

9. **Burbano Proaño, Diego Javier.** *Análisis comparativo de bases de datos de código abierto.* Quito : s.n., 2006.

10. **Group, Postgres Global Development.** *Tutorial de PostgreSQL.* [ed.] Thomas Lockhart.

11. **Castro Morell, Daniel Eduardo y Pérez Váquez, Ramiro.** *Replicación de Datos.* Santa Clara : Universidad Central "Marta Abreu" de las Villas.

12. **García, C.** *Escenario de red para la supervisión de fallas en centrales telefónicas.* Santa Clara : Universidad Central "Marta Abreu" de Las Villa, 1999.

13. **Armengot Iborra, Marcelo J.** *Sistema de Base de Datos cliente-servidor con interfaz gráfica multiplataforma: Perl.* Valencia : s.n., 2004.

14. **Nuñez Camallea, Noel L.** *Gestión de Bases de Datos con Acces XP.* s.l. : Editorial Científico Técnica.

15. **Mato García, Rosa María.** *Diseño de Base de Datos.* 1999.

16. **Database Mangement Solutions**. *EMS Data Generator for PostgreSQL. 2007* [En línea] [Citado el: 19 de Febrero del 2009.] http://www.sqlmanager.net/en/products/postgresql/datagenerator.

17. **Database Mangement Solutions**. *EMS SQL Query for PostgreSQL. 2007* [En línea] [Citado el: 19 de Febrero del 2009.] http://www.sqlmanager.net/en/products/postgresql/query

18. **Mustain, E** Introducing Slony, 2004. [En línea] [Citado el: 19 de Marzo del 2009]

http://www.onlamp.com/pub/a/onlamp/2004/11/18/slony.html

19. **Sitio oficial PgCluster**. *PGCluster is the synchronous replication system of the multi-master composition for PostgreSQL*. [En línea] [Citado el: 20 de Marzo del 2009]. http://pgcluster.projects.postgresql.org/feature.html

20. **Date, Christopher J**. *Introducción a los Sistemas de Bases de Datos.* Ciudad de La Habana: Editorial Félix Varela, 2003.

21. **Miguel Castaño, Adoración De y Piattini Velthuis, Mario G**. *Fundamentos y Modelos de Bases de Datos.* 2da ed. Madrid, España: Editorial RA-MA S.A., 1999. ISBN 9788478973613.

Bibliografía consultada:

- Bandi, Hans Geog. *La Edad de Piedra.* Barcelona. Praxis S.A, 1962.
- Bonilla, Carlos*. Asegure el impacto de su mensaje.* México, 2004, Mundo Ejecutivo.
- Guevara Injoque, Marco A., Flores Nazario, César R. *Conceptos Básicos de Modelamiento Lógico*. [En línea] www.librosdigitales.net
- Charte Ojeda, Francisco. *Bases de Datos con Microsoft Visual Basic.Net* Madrid: Ediciones Anaya Multimedia.[En línea][Consultado el: 10 de Marzo del 2009.], http://bibliodoc.uci.cu/pdf/reg00581.pdf
- Díaz Villanueva, Wladimiro. 2002*. Base de datos objeto-relacionales*. [En línea] http://informatica.uv.es/iiguia/DBD/Practicas/boletin_1.pdf
- Espinosa, Humberto. 2005.. *PostgreSQL: una alternativa de DBMS Open Source.* Links Global Services [En línea] http://www.lgs.com.ve/pres/PresentacionES_PSQL.pdf.