Universidad de las Ciencias Informáticas
Facultad 6

Título: Diseño e implementación de la Base de Datos del Sistema Informático de los Tribunales Militares de Región

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Licet Cardero Tasé

Daliagna Bicet Zayas

Tutor: M.Sc. David Ardite Martínez

Junio, 2007
Año 49 de la Revolución

DECLARACIÓN DE AUTORÍA

Declaramos que somos las únicas autoras de este trabajo y autorizamos a la dirección de los Tribunales Militares y al grupo UCI-MINFAR 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 _____.

_____ _____

Daliagna Bicet Zayas
Autor

Licet Cardero Tasé
Autor

_____

David Ardite Martínez
Tutor

"La preservación de la independencia de este país depende hoy, fundamentalmente, de la ciencia y de la técnica".

Fidel Castro Ruz

AGRADECIMIENTOS

A nuestros padres por confiar en nosotras

A Alain por estar siempre que lo necesitamos

A nuestro tutor David Ardite por su necesaria ayuda

Al colectivo de directivos y trabajadores de los Tribunales Militares.

A nuestros amigos y compañeros por la preocupación

A Duniel y Arodis por estar siempre dispuestos a ayudarnos

A estudiantes y profesores del proyecto UCI-MINFAR

A Fidel y a la Revolución

DEDICATORIA

Licet Cardero Tasé

A mis padres.

A mi mamá, por querer lo mejor para mi

A mi papá, por ser mi ejemplo y guía

A mi hermana, por el apoyo incondicional

A mi abuela por estar siempre conmigo

A mis familiares y a la memoria de los que ya no están

Daliagna Bicet Zayas

A mis padres, por confiar y esperar lo mejor de mí

A mi hermana, por su gran ayuda

A mis familiares queridos

A Fidel y a nuestra Revolución

RESUMEN

Los Tribunales Militares de la República de Cuba están estructurados en los niveles inferior, intermedio y superior. Actualmente existe un software que sirve de apoyo para la toma de decisiones en estas instituciones, que ha sido muy importante para el almacenamiento de la información correspondiente a los procesos jurídicos, pero debe ser remplazado por el motivo de que no acepta todas las funcionalidades requeridas por los trabajadores. El mismo fue creado en 1993 en MS-DOS, con un lenguaje de programación y un Sistema Gestor de Bases de Datos que hoy en día han quedado obsoletos. La Base de Datos que el sistema utiliza en la actualidad presenta una serie de problemas en su diseño que imposibilitan a los usuarios tener un buen control y manipulación de la información, por lo que se hizo necesario realizar un nuevo diseño, cumpliendo con los requisitos planteados por los clientes. Con la nueva estructura se garantiza que cada nivel tenga una Base de Datos propia y se aumenta el grado de seguridad e integridad en éstas. Este trabajo presenta una descripción de los pasos necesarios y las herramientas empleadas para la confección de la nueva Base de Datos del Sistema Informático de los Tribunales Militares de Región, que corresponden al nivel inferior, así como los métodos utilizados para la validación teórica y funcional del diseño propuesto.

PALABRAS CLAVE

SITMR: Sistema Informático de los Tribunales Militares de Región.

BD: Base de Datos.

SGBD: Sistema Gestor de Base de Datos.

TABLA DE CONTENIDO

INTRODUCCIÓN

El avance tecnológico de la computación ha alterado la vida cotidiana de la sociedad y cambiado los modos de proceder e inclusive de vivir. Las computadoras se han convertido en una herramienta indispensable para la realización de tareas y la solución de problemas, tanto así, que sin ellas en la actualidad muchas de las actividades colapsarían.

La informática ha penetrado en todos los ámbitos de la vida humana, estando presente en tiendas comerciales, universidades, tribunales, etc. La Informática Jurídica es una parte de ella aplicada sobre el Derecho, abordando el tratamiento lógico y automático de la información legal. Es necesario resaltar que la informatización de los órganos de justicia tiene una influencia positiva en los mismos, permitiendo aumentar la celeridad en los procesos judiciales y además, agilizar los asuntos de carácter jurídico-administrativos.

En el contexto internacional se prevé que en el futuro, como resultado de la introducción de las nuevas tecnologías en los Tribunales, se podrá permitir el conocimiento de los juicios a través de las redes informáticas. De esta manera se evitará, en ciertos aspectos, la sobrepoblación en los mismos.

La informatización del Poder Judicial es necesaria porque sirve de instrumento auxiliar de los jueces y del proceso. Es un elemento útil para las partes en el juicio, porque cada Tribunal tendría sus terminales y computadoras, donde el personal autorizado se dirigiría directamente marcando el número del expediente o la causa y obtendría la información necesaria. Por estas razones, es indispensable en cualquier país la puesta en marcha de esta tecnología, debido a que a través de la informática jurídica se puede lograr una mayor eficacia en la vigencia y respeto a los derechos humanos.

Cuba no ha estado ajena a la evolución de la informática y las comunicaciones que se ha llevado a cabo desde la segunda mitad del pasado siglo. En nuestro país las tareas del progreso científico-técnico se realizan de la manera más integral posible, no sólo creando instituciones de investigación, sino desarrollando también actividades como la información científica, la normalización y control de calidad, la organización científica del trabajo, las patentes y licencias, y la proyección industrial, incluidas las tareas vinculadas con la transferencia de tecnología y su asimilación.

En la actualidad la informática cubana se va caracterizando por una mayor asimilación de las aplicaciones relacionadas con el código abierto, teniendo como objetivo deshacerse de las patentes de Microsoft y propiciar un mayor desarrollo interno en el campo de la producción de software.

La utilización de la informática en el sistema judicial cubano va cobrando cada vez más fuerza. Con la instalación de equipos, redes y programas digitales, las instituciones jurídicas acometen un proceso de automatización para mejorar la calidad y rapidez de su labor. La informatización del sector judicial incluye hoy la interconexión digital lograda entre el Tribunal Supremo Popular y los tribunales provinciales. Se labora en la inclusión de los de nivel municipal, lo cual mejorará la eficiencia, agilidad y calidad del trabajo del sistema.

La igualdad de los ciudadanos ante la ley y su aplicación uniforme por los jueces, hace del sistema cubano una fórmula superior a la de aquellos países donde las fuerzas económicas provocan desbalances, conducen al soborno y a la desigualdad en el momento que se imparte justicia.

En 1976 fue promulgado el primer texto legal que creaba los Tribunales Militares, regulaba su estructura, organización y funcionamiento de forma colegiada en los actos de impartir justicia, así comenzaron a formar parte del Sistema de Tribunales de la República de Cuba.

Los Tribunales Militares tienen como objetivos específicos prevenir y reprimir, a través de sus pronunciamientos en los actos de justicia, todo hecho delictivo que afecte o pueda afectar la seguridad del Estado, la capacidad y disposición combativa de las instituciones armadas, la disciplina militar o el orden reglamentario establecido para el cumplimiento del servicio militar, así como las demás infracciones de la ley penal. Deben cumplir y hacer cumplir la legalidad socialista y contribuir a la educación de los militares en la observancia estricta de las leyes, reglamentos, disposiciones y órdenes militares.

Estas organizaciones se dividen en tres niveles: en el nivel inferior se encuentran los Tribunales Militares de Región, en el intermedio los Tribunales Militares Territoriales y en el superior la Sala de lo Militar del Tribunal Supremo Popular. En este trabajo sólo se abordan los procesos jurídicos correspondientes a los Tribunales Militares de Región. En cada provincia existe un tribunal de este tipo, excepto en La Habana, que cuenta con los Tribunales Militares Ciudad Habana, Región Oeste y Región Este. En La Isla de La Juventud también existe un Tribunal especial para el desarrollo de sus procesos judiciales.

Figura 1. Niveles de los Tribunales Militares.

En los Tribunales Militares de Región se encuentra en explotación un sistema informático, elaborado en el año 1993 en el sistema operativo MS-DOS y con los recursos de programación propios del momento. Este sistema ha jugado un importante papel en apoyo del trabajo judicial, pero se hace necesario la confección de una nueva versión en un lenguaje de programación y en un entorno actualizados, que permita perfeccionar su estructura, mejorar la interacción de los usuarios con el mismo e integrarle nuevas opciones de trabajo de acuerdo con las experiencias acumuladas en su explotación, de forma que no juegue solo un rol informativo, sino que tenga también un papel más activo en apoyo a la dirección de la actividad judicial en estas instituciones.

Las deficiencias que el sistema presenta, por las cuales no satisface las necesidades actuales de sus usuarios son las siguientes:

- No permite consultar la información por criterios de selección.
- No tiene implementada una documentación de referencia (Manual de Usuario) para facilitar el adiestramiento a los trabajadores sin experiencia.
- Deficiente normalización de la Base de Datos (BD).
- Han surgido nuevas necesidades informativas que no se recogen en el sistema, y por tanto en su BD.

- Mucha información aparece duplicada y a veces triplicada en el sistema.

Para resolver la situación relacionada con la Base de Datos se llegó a la conclusión de que se debía dar respuesta al **problema** siguiente: ¿Cómo erradicar las deficiencias existentes en la BD del Sistema Informático de los Tribunales Militares de Región?

Se definieron como **objeto de estudio** los procesos informativos de los Tribunales Militares de Región y como **campo de acción** el diseño y procesamiento de la Base de Datos del sistema.

Para la solución del problema se planteó como **objetivo general*:*** Diseñar e implementar la Base de Datos del nuevo Sistema Informático de los Tribunales Militares de Región.

A partir del análisis del objetivo general se derivaron los siguientes **objetivos específicos:**

- Diseñar la Base de Datos.
- Validar el diseño de la Base de Datos.
  - Validación teórica del diseño.
  - Validación funcional.
- Implementar la Base de Datos.
- Garantizar la seguridad de los datos.

Las tareas desarrolladas para dar cumplimiento a los **objetivos específicos** fueron las siguientes:

- Estudio de los requisitos funcionales y no funcionales del sistema.
- Interpretación de la propuesta de arquitectura seleccionada por los arquitectos.
- Estudio del diagrama de clases persistentes obtenido a partir del diagrama de clases del diseño.
- Diseño del Diagrama Entidad Relación de la Base de Datos.
- Normalización de la Base de Datos.
- Análisis de la seguridad de la Base de Datos.
- Pruebas a la Base de Datos con consultas y herramientas de carga intensiva.

La tesis está estructurada en los capítulos siguientes:

Capítulo1

El capítulo1 incluye un estado del arte sobre el tema de las Bases de Datos en la actualidad. Se proporciona una breve explicación de la evolución de las mismas y se analizan los distintos modelos de datos que más se conocen hoy en día. También se hace referencia a la arquitectura que emplean los Sistemas Gestores de Bases de Datos y a la metodología y herramientas utilizadas para diseñar e implementar la BD del Sistema Informático de los Tribunales Militares de Región.

Capítulo2

En el capitulo2 se describe y analiza la solución del diseño propuesto. Primeramente se argumentan los principales requisitos funcionales y no funcionales del nuevo sistema, extraídos a partir de la labor realizada por el analista del proyecto. Luego se describe la interacción del software con otros sistemas y la propuesta de arquitectura de acuerdo al estudio efectuado por los arquitectos. Para continuar con el trabajo se analiza el diseño de la BD a través del diagrama de clases persistentes, del modelo de datos y de cada una de sus tablas. Por último se hace referencia al tema de optimización de consultas en la BD diseñada.

Capítulo3

En el último de los capítulos se examina la validación teórica y funcional del diseño realizado, a través de la integridad, normalización y seguridad de la Base de Datos. También se tratan otros aspectos como el análisis de redundancia en la información, la trazabilidad de las acciones y las pruebas empleadas para comprobar la consistencia y buena estructuración de la BD.

Capítulo 1
Fundamentación Teórica

1.1 Introducción.

Este capítulo presenta una breve reseña del tema de las Bases de Datos (BD) en la actualidad, así como de las tendencias, técnicas, tecnologías y metodologías utilizadas en el mundo para su tratamiento. También se especifican las herramientas empleadas para diseñar Bases de Datos y principalmente las que se utilizaron en el trabajo para confeccionar, implementar y realizarle pruebas a la BD del Sistema Informático de los Tribunales Militares de Región.

1.2 Introducción a las Bases de Datos.

Los datos constituyen la parte esencial de un sistema de información, y son los que justifican su existencia. Para organizar y gestionar estos datos en el computador, se han desarrollado técnicas cuya evolución ha estado determinada, principalmente, por el desarrollo de la tecnología de los computadores, así como por los requisitos y necesidades planteadas por los usuarios.

En la actualidad, las técnicas de Bases de Datos representan la tecnología informática disponible para la organización y gestión de grandes volúmenes de datos en un computador. Se puede afirmar que el núcleo de todo sistema de información actual es una Base de Datos, y que el diseño y creación de ésta constituyen una etapa importante en la construcción del sistema.

1.2.1 Breve historia de los sistemas de almacenamiento de datos.

Los sistemas de almacenamiento de datos han evolucionado desde los archivos secuenciales hasta los Sistemas de Gestión de Bases de Datos que hoy se conocen. Los primeros brindaban un acceso muy rápido a la información pero tenían un inconveniente, para acceder a una posición se debía hacer de forma secuencial, es decir, se tenía que recorrer el archivo entero. Posteriormente aparecieron los archivos indexados, con los que se podía ir directamente al lugar deseado (de forma aleatoria).

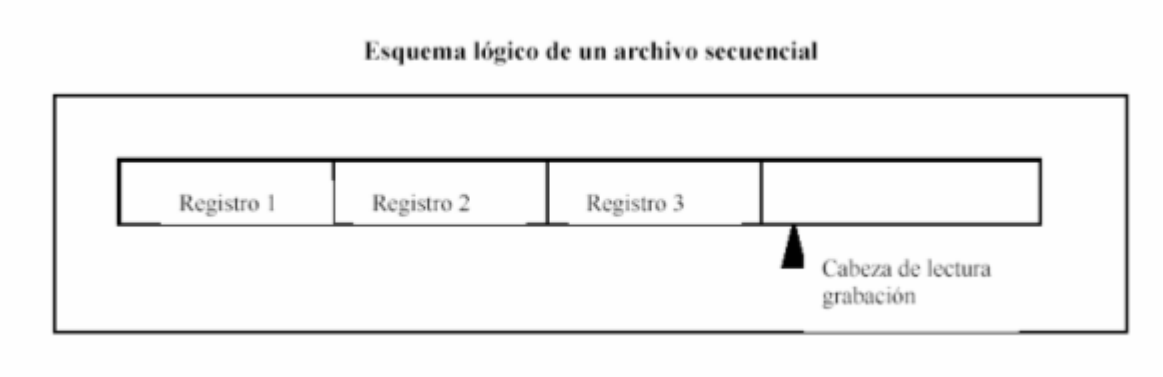


Figura 2. Esquema lógico de un archivo secuencial.

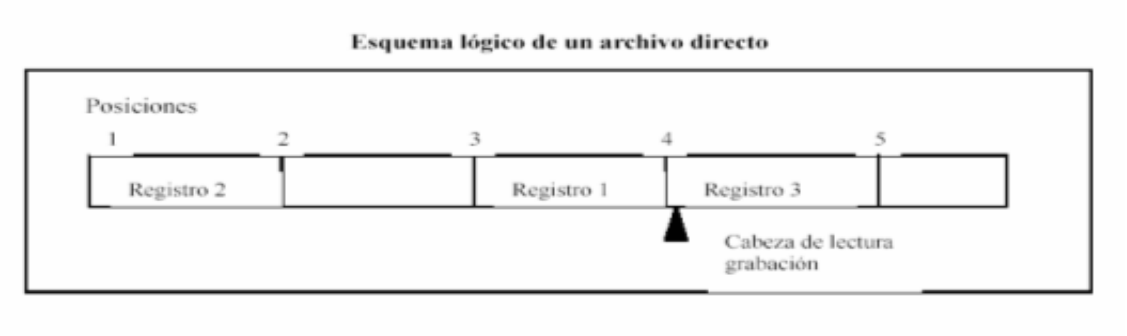


Figura 3. Esquema lógico de un archivo directo.

Por la necesidad de compartir datos entre varias máquinas surgió el NFS (Sistema de Archivos de Red), posibilitando que distintos sistemas conectados a una misma red accedan a ficheros remotos como si fueran locales, y más tarde para evitar fallos en los sistemas de archivos aparecieron los dispositivos RAID, utilizándose para aumentar la integridad de los datos en los discos.

Debido a que los programas eran cada vez más complejos y grandes, se requería almacenar los datos en sistemas que garantizaran un cierto número de condiciones, permitiendo operaciones complejas sin que se violaran estas restricciones y que garantizaran la seguridad de la información. Respondiendo a

estas necesidades surgieron las Bases de Datos Jerárquicas, donde los datos se situaban siguiendo una jerarquía, pero tenían el problema de que los accesos a éstos eran unidireccionales. (Hansen)

Para dar absoluta libertad a las relaciones entre tablas surgieron las Bases de Datos Relacionales en los años 80, y con ellas los Sistemas de Bases de Datos Relacionales (SBDR). Esta nueva forma de almacenamiento de la información aportó dos características muy importantes: las propiedades ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad) y un lenguaje común de acceso a los datos: SQL, facilitando la programación de aplicaciones con Bases de Datos, y consiguiendo que los programas sean independientes de los aspectos físicos de la Base de Datos.

Con el transcurso del tiempo aparecieron, en los años 90, las Bases de Datos Distribuidas. Estas BD no se almacenan completamente en una localidad central, sino que se distribuyen en una red de localidades que pueden estar geográficamente separadas y conectadas por enlaces de comunicaciones. Cada localidad tendrá su propia BD y está capacitada para acceder a los datos de otras localidades.

1.2.2 ¿Qué es una Base de Datos?

Es un conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora. O sea, que puede considerarse una colección de datos variables en el tiempo. (C.J.Date, 2003)

Una BD está compuesta por los siguientes componentes: (C.J.Date, 2003)

1. ***Hardware***

Se refiere a los dispositivos de almacenamiento en donde reside la Base de Datos, así como a los dispositivos periféricos (unidad de control, canales de comunicación, etc.) necesarios para su uso.

2. ***Software***

Está constituido por un conjunto de programas que se conoce como Sistema Gestor de Base de Datos, el cual maneja todas las solicitudes formuladas por los usuarios de la misma.

3. ***Usuarios***

Existen tres clases de usuarios relacionados con una Base de Datos:

- El programador de aplicaciones, quien crea programas de aplicación que utilizan la BD.
- El usuario final, quien accede a la BD por medio de un lenguaje de consulta o de programas de aplicación.

- El Administrador de la BD, quien se encarga del control general del Sistema Gestor de Base de Datos.

1.2.3 Sistema Gestor de Base de Datos (SGBD).

El software que permite la utilización y/o actualización de los datos almacenados en una o varias Bases de Datos, desde diferentes puntos de vista a la vez, por uno o varios usuarios, se denomina Sistema Gestor de Bases de Datos. (C.J.Date, 2003)

El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular, en términos abstractos y de una forma práctica y eficiente, los datos, de manera que no le sea necesario conocer el modo de almacenamiento de la información en la computadora, ni el método de acceso empleado. Se compone de un lenguaje de definición de datos (DDL), de un lenguaje de manipulación de datos (DML) y de un lenguaje de consulta (SQL).

1.3 Modelos de Bases de Datos.

Un modelo de datos es básicamente una “descripción” de algo conocido como contenedor de datos (algo en donde se guarda la información), así como de los métodos para almacenar y recuperar información de esos contenedores. (C.J.Date, 2003)

Los modelos de Bases de Datos más extendidos son el modelo relacional, el modelo de red, el modelo jerárquico. El modelo orientado a objetos es también muy popular, pero no existe un modelo estándar orientado a objetos. Con el desarrollo de la tecnología ha aparecido también el modelo de datos objeto relacional y el modelo multidimensional. (mai)

1.3.1 Bases de Datos Jerárquicas.

Las Bases de Datos Jerárquicas surgieron a mediados de 1960 y son Bases de Datos que, como su nombre indica, 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 se le conoce como raíz, y a los nodos que no tienen hijos se les conoce

como hojas. Una de las principales limitaciones de este modelo, es su incapacidad de brindar eficientemente una solución a la redundancia de datos. (Hansen)

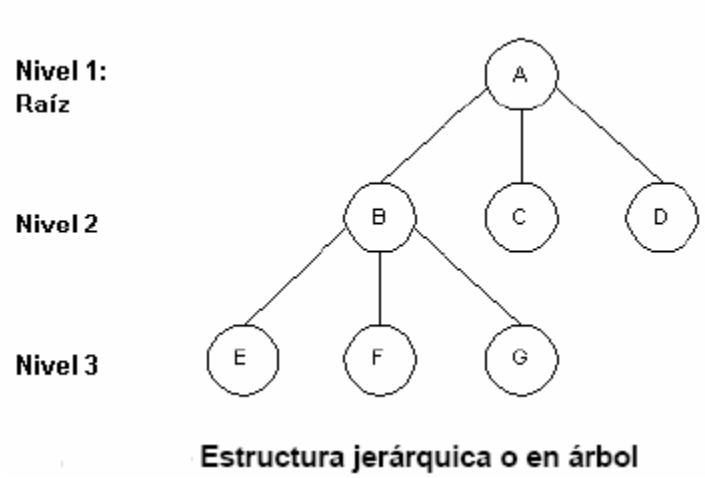


Figura 4. Estructura de una Base de Datos Jerárquica.

Los productos jerárquicos, exactamente el IMS y el DL/I de IBM como máximos exponentes de estos sistemas, consiguieron altas cuotas de mercado, aunque la actual difusión de la tecnología relacional los ha llevado a convertirse en sistemas superados. Lo cual no quiere decir que no persistan todavía importantes aplicaciones soportadas en estos productos que están trabajando, por su eficiente respuesta, a satisfacción de sus usuarios. Las aplicaciones desarrolladas sobre ellos se mantienen sin apenas cambios.

1.3.2 Bases de Datos de Red.

Es un modelo ligeramente distinto del jerárquico, y surge a finales de 1960. Su diferencia fundamental es la modificación del concepto de un nodo, permitiendo que un mismo nodo tenga varios padres (algo no permitido en el modelo jerárquico) (Hansen).

Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos. Pero aún así, la dificultad que implica administrar la información en

una Base de Datos de Red, ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales. Un ejemplo de Base de Datos de Red es el sistema denominado IDMS/R.

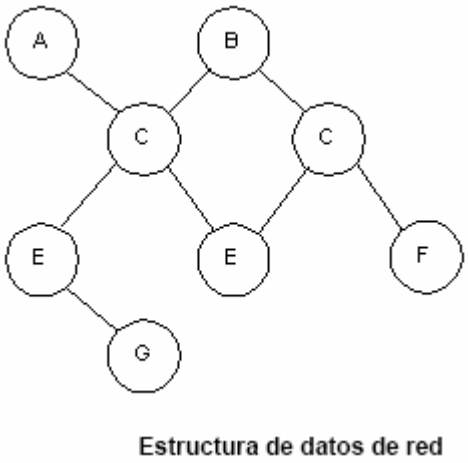


Figura 5. Estructura de una Base de Datos de Red.

1.3.3 Bases de Datos Relacionales.

Es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Su idea fundamental es el uso de entidades (tablas), compuestas por registros (las filas de una tabla), que representarían las tuplas, y campos (las columnas de una tabla). (Hansen)

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). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario casual de la Base de Datos. La información puede ser recuperada o almacenada por medio de “consultas” que ofrecen una amplia flexibilidad y poder para administrarla. Ejemplos de Bases de Datos Relacionales son: Oracle, Informix, MySql, etc.

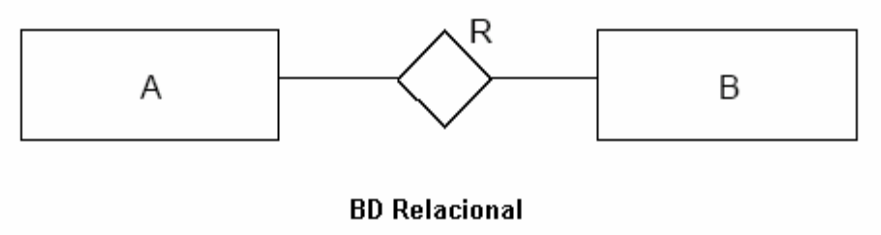


Figura 6. Estructura de una Base de Datos Relacional.

1.3.4 Bases de Datos Orientadas a Objetos.

A finales de los 80's aparecieron las primeras BDOO (Bases de Datos Orientadas a Objetos). Están diseñadas para ser eficaces, desde el punto de vista físico, para almacenar objetos complejos y métodos. (C.J.Date, 2003)

Este modelo, bastante reciente, y propio de los modelos informáticos orientados a objetos, trata de almacenar en la Base de Datos los objetos completos (estado y comportamiento). Combinan las mejores cualidades de los archivos planos, las Bases de Datos Jerárquicas y Relacionales. Las BDOO representan el siguiente paso en la evolución de las Bases de Datos para soportar análisis, diseño y programación orientada a objetos. Ofrecen un mejor rendimiento de la máquina que las Bases de Datos Relacionales, para aplicaciones o clases con estructuras complejas de datos. (C.J.Date, 2003)

Entre los Gestores de Bases de Datos Orientados a Objetos más conocidos se encuentran los siguientes: (Hansen)

Gemstone: Proporciona control de concurrencias y recuperación de la información almacenada, así como gestión de almacenamiento secundario. Soporta acceso concurrente y métodos para mantener la seguridad y la integridad de las BD.

Vbase: Surge en 1987 y enfatiza algunas características de la orientación a objetos.

También existen Orion, PDM, Iris y O2, entre otros.

1.3.5 Bases de Datos Objeto _ Relacionales.

Las Bases de Datos Objeto_Relacionales son Bases de Datos que desde el modelo relacional evolucionan hacia Bases de Datos más extensas y complejas incorporando, para obtener este fin,

conceptos del modelo orientado a objetos. Se puede afirmar que un Sistema de Gestión Objeto-Relacional (SGBDOR) contiene dos tecnologías; la tecnología relacional y la tecnología de objetos. (C.J.Date, 2003)

Con las Bases de Datos Objeto-Relacional, se pueden crear nuevos tipos de datos, que permiten gestionar aplicaciones más complejas con una gran riqueza de dominios. Tienen la posibilidad de incluir el chequeo de las reglas de integridad referencial a través de los disparadores, entre otras características. Uno de los gestores de Bases de Datos Objeto Relacional que más se utiliza en la actualidad es PostgreSQL.

1.3.6 Bases de Datos Multidimensionales.

Un enfoque que ha cobrado actualmente fuerza es el análisis analítico en línea (en inglés, denominado On-Line Analytical Processing u OLAP). Su nombre se deriva del contraste con el procesamiento de transacciones en línea (On-Line Transaction Processing, OLTP). Mientras que el OLTP depende de Bases de Batos Relacionales, el OLAP ha desarrollado una tecnología de Bases de Datos Multidimensionales. Estas Bases de Datos fundan los cimientos para el desarrollo de los cálculos y análisis multidimensionales que requiere la inteligencia empresarial.

Las Bases de Datos Multidimensionales son BD estructuradas como un hipercubo, con un eje por dimensión de estructura basada en dimensiones orientadas a consultas complejas y alto rendimiento. Estos modelos tienen gran aplicación en la bioinformática, debido al almacenamiento de grandes cantidades de información biológica. (Gascón, 1999-2004)

A continuación se muestra una representación espacial de una variable multidimensional con una, dos y tres dimensiones. En esta figura los cubitos representan valores de dimensión, y las esferas son datos.

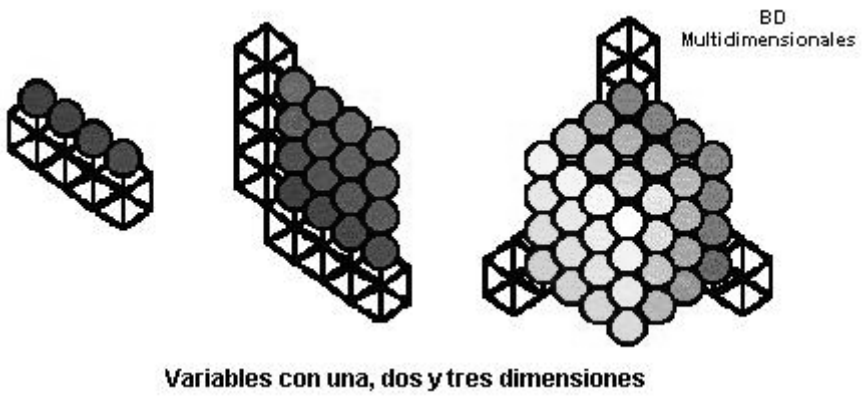


Figura 7. Representación espacial de una variable multidimensional.

Una variable unidimensional podría ser el cambio del euro con el dólar, que sólo varía en la dimensión <tiempo>. Los cubitos serían, por ejemplo, los días del año y las esferas serían los valores numéricos correspondientes al cambio monetario en cada momento. Un ejemplo de variable de dos dimensiones es el número de habitantes, que se mueve por las dimensiones <Geografía> y <tiempo>. Finalmente, los ingresos de una organización podrían almacenarse mediante una variable de tres dimensiones: <producto>, <Geografía> y <tiempo>.

1.4 Tendencias actuales y futuras.

La utilización de Bases de Datos como plataforma para el desarrollo de Sistemas de Aplicación en las organizaciones se ha incrementado notablemente en nuestros días. Actualmente las BD ofrecen numerosas funcionalidades de gran importancia para el almacenamiento y la manipulación de la información en las instituciones, pero siguen creciendo las demandas de nuevas operaciones que faciliten una mayor eficiencia en el manejo de los datos. Esto trae consigo el desarrollo de técnicas y herramientas que brinden una solución eficiente a las solicitudes planteadas por los usuarios.

Algunas de las tendencias actuales y futuras de Bases de Datos son citadas a continuación.

- La explotación efectiva de la información brinda ventajas competitivas a las organizaciones.

- El uso de las Bases de Datos Distribuidas y Multidimensionales se incrementa de manera considerable en la medida en que la tecnología de comunicación de datos brinde más facilidades para ello.
- El uso de las Bases de Datos facilita y soporta en gran medida a los sistemas de información para la toma de decisiones.
- Los lenguajes de consulta (SQL) permitirán el uso del lenguaje natural para solicitar información de la Base de Datos, haciendo más rápido y fácil su manejo.

En los últimos años se ha visto un gran crecimiento en la capacidad de generación y almacenamiento de información, debido a la creciente automatización de procesos. Desgraciadamente no ha existido un desarrollo equivalente en las técnicas de análisis de información, por lo que existe la necesidad de una nueva generación de técnicas y herramientas computacionales con la capacidad de asistir a usuarios en el análisis automático e inteligente de datos. El procesar automáticamente grandes cantidades de datos para encontrar conocimiento útil para un usuario y satisfacerle sus metas, es el objetivo principal del área de Descubrimiento de Conocimiento en Bases de Datos o KDD (Knowledge Discovery from Data Base). Este es el campo que está evolucionando para proporcionar soluciones al análisis automatizado de datos.

1.5 Arquitectura de los Sistemas Gestores de Bases de Datos.

Las arquitecturas de Bases de Datos han evolucionado mucho desde sus comienzos, aunque la considerada estándar hoy en día es la descrita, a finales de los años 70, por el comité ANSI/X3/SPARC (Standard Planning and Requirements Committee of the American National Standards Institute on Computers and Information Processing) La misma se emplea como ayuda para conseguir la separación entre los programas de aplicación y los datos, el manejo de múltiples vistas por parte de los usuarios y el uso de un catálogo para almacenar el esquema de la Base de Datos.

Este comité propuso una arquitectura general basada en tres niveles o esquemas: el nivel físico, o de máquina, el nivel externo, o de usuario, y el nivel conceptual. Así mismo describió las interacciones entre estos tres niveles y todos los elementos que conforman cada uno de ellos.

- Nivel interno: Describe la estructura física de almacenamiento de Base de Datos. Emplea un modelo físico de datos y los únicos datos que existen están realmente en este nivel.
- Nivel conceptual: Describe la estructura de toda la Base de Datos para una comunidad de usuarios. Oculta los detalles físicos de almacenamiento y trabaja con elementos lógicos como entidades, atributos y relaciones.
- Nivel externo o de vistas: tiene varios esquemas externos o vistas de usuario. Cada esquema describe la visión que tiene de la Base de Datos un grupo de usuarios, ocultando el resto.

Las posibles proyecciones de datos al utilizar la arquitectura de tres niveles quedan resumidas en la gráfica:

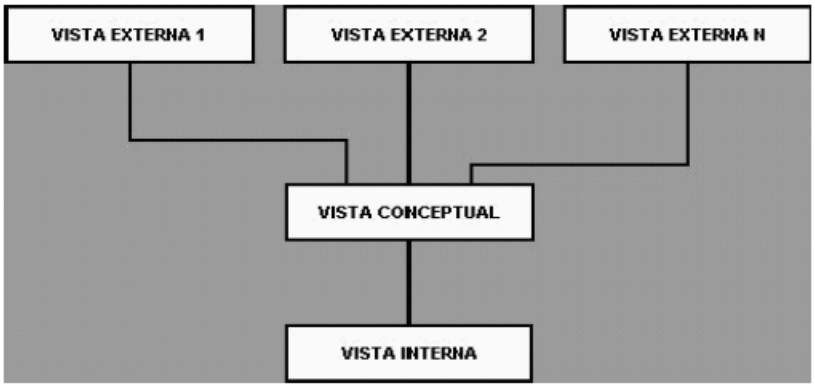


Figura 8. Posibles proyecciones de datos al utilizar la arquitectura de tres niveles.

1.6 RUP (Proceso Unificado de Desarrollo).

De acuerdo con los lineamientos, principios y normativas para el desarrollo del software en las FAR se establece que todas sus aplicaciones informáticas utilicen RUP como Proceso de desarrollo de software en la creación de sus proyectos. La Base de Datos del Sistema Informático de los Tribunales Militares de Región se diseñó cumpliendo con lo reglamentado en el documento que plantea estas normas.

RUP (Rational Unified Process) más que un simple proceso para el desarrollo de aplicaciones, es un marco de trabajo genérico, que puede especificarse para una gran variedad de aplicaciones informáticas, para diferentes áreas de aplicación, diferentes niveles de aptitud y diferentes tamaños de proyectos. Garantiza la elaboración de todas las fases de un producto de software orientado a objetos, desde la etapa de ingeniería de requerimientos hasta la de prueba, lo que resulta muy importante en la elaboración del diseño de la BD, ya que depende del trabajo desarrollado en otras etapas del proyecto para la realización del mismo.

Este proceso es dirigido por casos de uso para describir lo que se espera del software, está muy orientado a la arquitectura del sistema, mostrando la visión común del sistema completo y por tanto describiendo los elementos del modelo que son más importantes para su construcción. Se apoya en UML (Unified Modeling Language) como lenguaje expresivo, claro y uniforme, que no garantiza el éxito de los proyectos pero sí mejora sustancialmente el desarrollo de los mismos, al permitir una nueva y fuerte integración entre las herramientas, los procesos y los dominios.

Según RUP el Diseñador Principal de Bases de Datos es el responsable de definir los detalles del diseño de la Base de Datos y para ello crea el Modelo de Datos, garantizando su integridad y consistencia. Este artefacto describe las representaciones lógicas y físicas de los datos persistentes usados por la aplicación.

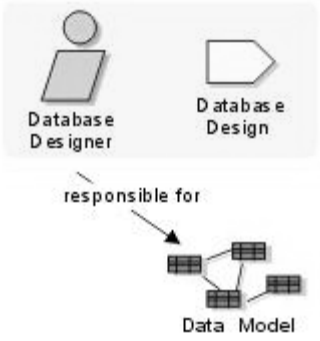


Figura 9. Artefacto construido por el diseñador de la Base de Datos.

1.7 Herramientas.

Para confeccionar una Base de Datos se necesitan herramientas que posibiliten garantizar la seguridad, consistencia e integridad de la información, así como lograr que todo el proceso de su creación sea lo más sencillo y rápido posible.

Existe una gran variedad de herramientas que se encargan del diseño de las Bases de Datos, entre ellas se encuentran:

ERwin

Brinda productividad en diseño, generación, y mantenimiento de aplicaciones. Desde un modelo lógico de los requerimientos de información, hasta el modelo físico perfeccionado para las características específicas de la Base de Datos diseñada, ERwin permite visualizar la estructura, los elementos importantes, y optimizar el diseño de la Base de Datos. Genera automáticamente las tablas y miles de líneas de procedimientos almacenados y disparadores para los principales tipos de Base de Datos.

Soporta principalmente bases de datos relacionales SQL y Bases de Datos que incluyen Oracle, Microsoft SQL Server, Sybase, DB2, e Informix.

EasyCASE

EasyCASE permite capturar los detalles de diseño de un sistema y comunicar las ideas gráficamente, para que sean fáciles de ver y entender. Para un diseño legítimo y modelación de datos, procesos y eventos, permite crear y mantener diagramas de flujo de datos, diagramas de entidad-relación, mapas de estructura y más. Posee herramientas de corrección avanzadas que permiten revisiones generales en minutos, en lugar de horas o días. Permite re-usar diagramas o partes de diagramas para economizar el diseño de un proyecto.

Soporta Oracle, Paradox, Postgress, SQLBase, SQL Server, Sybase, Watcom SQL, Access, ANSI SQL, Clipper, dBASE III, IV, VDB2, FoxPro, Informix, otras más.

También existen, entre otras herramientas, DB Designer, que integra diseño, modelación, creación y mantenimiento de bases de datos de forma simple y sencilla y Case Studio, que fue la escogida para el diseño de la BD del Sistema Informático de los Tribunales Militares de Región, por su facilidad en la creación de diagramas y que permite trabajar con PosgreSQL de forma sencilla y rápida.

Al igual que hay variedad de herramientas para el diseño de BD también existen una gran cantidad de Sistemas Gestores de Bases de Datos, que permiten la manipulación y actualización de los datos. Entre los más conocidos se encuentran:

MySQL

MySQL es un Sistema de Gestión de Bases de Datos Relacional. Es, probablemente, el más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación se debe, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración. Soporta gran cantidad de tipos de datos para las columnas; tiene gran portabilidad entre sistemas y gestiona usuarios y contraseñas, manteniendo un buen nivel de seguridad en los datos.

Oracle

Oracle es un sistema de administración de base de datos, fabricado por Oracle Corporation. Es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hacen que sólo se vea en empresas muy grandes y multinacionales.

Es sin duda uno de los mejores SGBD que existen en el mercado. Es robusto y tiene muchas características que garantizan la seguridad e integridad de los datos; que las transacciones se ejecuten de forma correcta, sin causar inconsistencias. Ayuda a administrar y almacenar grandes volúmenes de datos; tiene estabilidad, escalabilidad y es multiplataforma.

Otros Sistemas Gestores de Bases de Datos que tienen gran popularidad son: DB2 , SQL Server, Sybase ASE, Firebird y PostgreSQL.

1.7.1 Visual Paradigm.

El diagrama de clases persistentes obtenido a partir del diagrama de clases del diseño se generó utilizando Visual Paradigm (versión 5.3), herramienta CASE que ofrece un entorno de creación de diagramas para UML. El diseño es centrado en casos de uso y enfocado al negocio, lo que permite generar softwares de gran calidad. Esta herramienta usa un lenguaje estándar común para todo el equipo de desarrollo que facilita la comunicación. Tiene capacidad para la ingeniería directa e inversa en

Java, C++, PHP, entre otros lenguajes y disponibilidad de múltiples versiones para cada necesidad. Es multiplataforma y muy útil para la generación de código fuente en PHP. 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. Incorpora el soporte para trabajo en equipo, proporcionando que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros de equipo. (Equipo de Softonic, 1997-2007)

1.7.2 Case Studio.

Para realizar el diagrama entidad-relación se utilizó el Case Studio (versión 2.18), una herramienta profesional con la que se diseñan Bases de Datos, facilitando la creación de diagramas de relación, modelado de datos y gestión de estructuras. Tiene soporte para trabajar con una amplia variedad de formatos de Base de Datos (Oracle, SQL, MySQL, PostgreSQL, Access, etc.) y permite además generar scripts SQL, aplicar procesos de ingeniería inversa a las Bases de Datos, usar plantillas de diseño personalizables y crear detallados informes en HTML y RTF. Posibilita la creación de procedimientos almacenados, vistas, disparadores, dominios, tipos de datos definidos por el usuario, entre otras funcionalidades.

1.7.3 PostgreSQL.

Al igual que RUP el uso de PostgreSQL está establecido por las normativas de las FAR. En el proyecto se utilizó la versión 8.0.

PostgreSQL es uno de los Sistemas Gestores de Bases de Datos más utilizados por la comunidad de software libre por las razones siguientes:

Cumple con las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) y soporta el lenguaje común de acceso a los datos: SQL. Es multiplataforma y posee buenas interfaces de instalación y administración. Aproxima los datos a un modelo Objeto-Relacional, y es capaz de manejar completas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multiversión, soporte multiusuario, transacciones y optimización de consultas.

Está basado en el proyecto POSTGRES, de la universidad de Berkeley. Es una derivación libre (OpenSource) de este proyecto, y utiliza el lenguaje SQL92/SQL99. Fue el pionero en muchos de los

conceptos existentes en el Sistema Objeto-Relacional actual, incluido, más tarde en otros sistemas de gestión comerciales. Incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores e integridad transaccional.

Lleva más de una década de desarrollo, siendo hoy en día un sistema bastante avanzado, que tiene soporte nativo para los lenguajes de programación: C, C++, Java, Python, PHP y muchos más. Se encuentra bajo la licencia BSD (Berkeley Software Distribution).

Entre otras características que presenta se encuentran las siguientes:

1. Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP...), cadenas de bits, etc. Además permite la creación de tipos de datos propios.
2. Incorpora una estructura de datos array.
3. Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etc.
4. Permite la declaración de funciones propias, así como la definición de disparadores.
5. Soporta el uso de índices, reglas y vistas.
6. Incluye herencia entre tablas.
7. Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

1.7.4 PGManager.

PGManager es una herramienta gráfica fácil de utilizar para la administración de PostgreSQL. Trabaja con cualquiera de sus versiones hasta la 8.1 y apoya sus últimas características incluyendo los tablespaces. Es la más completa hasta ahora. Permite obtener la documentación sobre el diseño de la BD y tiene un grupo de herramientas importantes para la importación y exportación de datos. Posibilita asignar los derechos de usuarios de una forma sencilla y rápida. (1999-2007) La versión del PGManager que se utilizó fue la 3.3.0.1.

1.8 Conclusiones.

Los Sistemas Gestores de Bases de Datos más utilizados a nivel mundial son los relacionales, pero se ha evolucionado mucho en este tema. Hoy en día existen otros modelos de datos como las Bases de Datos Orientadas a Objetos y las Objeto Relacionales, que implementan características de la orientación a objeto. Las Bases de Datos Distribuidas y Multidimensionales también se han ido expandiendo en la medida que se va desarrollando la tecnología.

En el mundo existen diversas herramientas para el diseño y procesamiento de las BD, las que se van ampliando y perfeccionando con el tiempo. Muchas veces se hace complejo escoger entre las distintas técnicas desarrolladas para crear una BD, pero se ha llegado a la conclusión de que las seleccionadas fueron de gran ayuda para lograr un buen diseño, implementación y validación de la Base de Datos del Sistema Informático de los Tribunales Militares de Región, destacándose sobre todas PostgreSQL, que es el SGBD donde reside la BD en estos momentos, por su gran capacidad para garantizar la seguridad e integridad de los datos.

Capítulo 2
Descripción y Análisis

2.1 Introducción.

En este capítulo se definen las actividades más importantes que posibilitaron la construcción de la Base de Datos, teniendo como resultado final el diseño de la misma.

Primeramente se seleccionaron y argumentaron los requisitos funcionales y no funcionales del sistema propuesto, que permitieron reconocer las entidades y atributos de la BD. Luego se describió y fundamentó la arquitectura a utilizar en el proyecto para una buena comprensión del tema del acceso a datos desde la BD. Un paso clave que se tuvo en cuenta en este capítulo fue el estudio del diagrama de clases persistente obtenido a partir del diagrama de clases del diseño por su importancia para la confección del modelo de datos. Por último se confeccionó el diagrama entidad relación de la BD y se describieron sus tablas.

2.2 Selección y argumentación de los requisitos funcionales y no funcionales del sistema propuesto.

Para que un sistema de software funcione correctamente se debe lograr una comunicación efectiva entre los usuarios y el equipo de proyecto con el objetivo de llegar a un entendimiento de lo que hay que hacer, lo que constituye la clave del éxito en la producción de un software.

Aquí radica la importancia que en los últimos años se le ha dado a la identificación de los requerimientos como parte del proceso de desarrollo del software. Los requisitos se pueden clasificar en: funcionales y no funcionales.

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir. Los no funcionales son propiedades o cualidades que el producto debe tener, permitiendo que el mismo sea atractivo, usable, rápido y confiable.

2.2.1 Requisitos Funcionales.

En esta sección se presentan los requisitos funcionales fundamentales del sistema, que constituyeron un paso esencial para el diseño de las entidades de la Base de Datos propuesta.

1. Registrar, buscar y modificar Expedientes de Fase Preparatoria (EFP).

Un EFP es un expediente que se elabora por la fiscalía, durante el proceso de instrucción contra un acusado de cometer uno o varios delitos previstos en la ley, con el objetivo de almacenar los datos que se aportan como pruebas en la investigación.

El sistema debe permitir almacenar el número del EFP, la fiscalía militar de procedencia, la fecha en que ocurrieron los hechos delictivos, la fecha de denuncia, de entrada, y la cantidad de acusados relacionados con el expediente. Además se generará un número de registro cada vez que se introduzca un EFP en el sistema y el usuario tendrá la posibilidad de ir desde la interfaz del EFP a la de los acusados y las piezas de convicción.

Se dará la posibilidad de realizar búsquedas de los expedientes por los números de registros o por el número de expediente conjuntamente con la fiscalía de procedencia y la fecha de entrada. De igual manera los usuarios podrán modificar los datos de los expedientes.

2. Registrar, buscar, modificar y eliminar datos de los acusados.

Se permitirá registrar el carné de identidad de los acusados, el nombre, los apellidos, el sexo, la provincia y municipio de nacimiento, el color de la piel, los nombres de la madre y del padre y la nacionalidad.

El sistema brindará la posibilidad de realizar búsquedas de los acusados por sus respectivos números de carné de identidad o nombres, al igual que permitirá la modificación y eliminación de sus datos.

3. Registrar, buscar, modificar y eliminar los datos de los acusados en un determinado EFP.

Un acusado puede estar relacionado con diferentes expedientes, al igual que un expediente puede contener varios acusados. Por tal motivo el sistema debe permitir realizar diferentes operaciones sobre un acusado en un EFP determinado.

Se deben registrar el identificador del acusado y del expediente con el que se relaciona, el número correspondiente del acusado en ese EFP, su situación legal y en el caso que se encuentre en prisión preventiva se guarda el lugar en que cumple esta y la fecha en que comenzó a cumplirla. También se almacenan el tipo de efectivo que es (militar, civil o civil de la defensa) y su nivel escolar, así como su estado civil, su integración política, la cantidad de hijos que tiene y otros datos de interés.

El sistema debe posibilitar también buscar los acusados en los EFP por sus respectivos identificadores, realizar modificaciones a los datos y eliminarlos en caso que sea necesario.

4. Registrar, buscar, modificar y eliminar datos de los acusados militares en un determinado EFP.

El sistema permitirá insertar los datos correspondientes a los acusados militares en un determinado EFP, los cuales son: identificador del acusado, del expediente al que pertenece, el grado y el cargo, la unidad a la que pertenece, la especialidad que ocupa y el tiempo de servicio en la institución armada a la que pertenece el acusado.

De igual forma posibilitará la búsqueda de estos acusados, la modificación de sus datos y la eliminación de ellos.

5. Registrar, buscar, modificar y eliminar datos de los acusados civiles en un determinado EFP.

El sistema permitirá insertar los datos correspondientes a los acusados civiles en un EFP, los cuales son: identificador del acusado y del expediente al que pertenece, la profesión que ejerce y el centro de trabajo.

De igual forma posibilitará la búsqueda de estos acusados, la modificación de sus datos y la eliminación de ellos.

6. Registrar, buscar, modificar y eliminar datos de los acusados civiles de la defensa en un determinado EFP.

El sistema permitirá insertar los datos correspondientes a los acusados civiles de la defensa en un determinado EFP, los cuales son: identificador del acusado y del expediente al que pertenece, la profesión que ejerce, la unidad militar a la que pertenece y su categoría ocupacional.

De igual forma posibilitará la búsqueda de estos acusados, la modificación de sus datos y la eliminación de ellos.

7. Registrar, buscar, modificar y eliminar los delitos correspondientes a un acusado en un determinado EFP.

Un acusado en un determinado expediente puede tener varios delitos, por lo que el sistema debe permitir realizar operaciones sobre los delitos de un acusado en el EFP correspondiente.

Se deben registrar los identificadores del acusado y del expediente con el que se relaciona, así como sus delitos. También se permitirá la búsqueda por delitos y la modificación o eliminación según corresponda.

8. Registrar, buscar, modificar y eliminar piezas de convicción.

Las piezas de convicción son los instrumentos con los que se cometen los delitos.

Se permitirá el almacenamiento de las descripciones de las piezas de convicción, así como se generará un número de registro por cada entrada al sistema de las piezas. El usuario debe tener la posibilidad de buscar las piezas de convicción por los números de registros y modificarlos o eliminarlos según corresponda.

9. Registrar, buscar, modificar y eliminar los datos correspondientes a la realización de los estudios de los expedientes.

Cuando se realizan los estudios de los EFP se deben almacenar el identificador del expediente, el juez que efectuó el estudio, la fecha inicial y final del mismo, la decisión del juez con respecto al expediente, y el destino y la fecha de remisión de acuerdo a la decisión tomada.

También se posibilitará realizar búsquedas a los expedientes estudiados, modificarlos y eliminarlos en caso necesario.

10. Registrar, buscar, modificar y eliminar los datos de las sesiones dispositivas.

Las sesiones dispositivas son parecidas a un juicio, pero se realizan cuando la fiscalía no está de acuerdo con la decisión del tribunal derivada del estudio del EFP. Se deben almacenar el identificador de la sesión, la fecha de celebración, los datos que identifican al expediente estudiado, el juez que participa y el fiscal, así como la decisión del tribunal y si ésta fue impugnada o no.

También se posibilitarán realizar búsquedas, modificaciones y eliminaciones de las sesiones dispositivas.

11. Registrar, buscar, modificar y eliminar los datos correspondientes a las impugnaciones.

Cuando se realiza una sesión dispositiva si la decisión que se tomó en la misma fue impugnada, el expediente se remite a un tribunal superior que se encarga de procesarlo y se deben registrar los siguientes datos: identificador de la sesión dispositiva en que se impugnó la decisión, la fecha de impugnación, el tribunal superior al que fue remitido, la fecha de salida hacia el mismo y el fiscal que impugnó la decisión del tribunal.

De igual forma el sistema debe permitir buscar los datos correspondientes a las impugnaciones, si es necesario modificarlos y eliminarlos.

12. Registrar, buscar o modificar las causas.

Una causa es un Expediente de Fase Preparatoria que después de estudiado se definió que estaba acto para constituirse como tal y continuar el proceso legal con ella, hacia la celebración del juicio oral.

El sistema debe permitir registrar el número de causa, el identificador del expediente del que proviene, la cantidad de acusados, la fecha en que fue radicada, el juez de radicación, la institución a la que pertenece, el tipo de hecho relacionado con los delitos y el tipo de procedimiento que se va a seguir en la causa . Además se debe tener en cuenta la fecha en que se va a realizar el juicio oral y la fecha en que los acusados conocen los motivos por los que se les acusa.

Los usuarios podrán efectuar búsquedas de las causas por su número o por el identificador del EFP del que provienen, al igual que modificarlas.

13. Registrar, buscar, modificar o eliminar datos de los acusados en determinadas causas.

Cuando se radica una causa se almacenan datos de los acusados que no se recogieron en el expediente.

El sistema debe darle la posibilidad al usuario de insertar el identificador del acusado y de la causa con la que se relaciona, su estado civil, antecedentes penales, situación legal y la fecha y el lugar, en caso de que se encuentre en privación de libertad, donde se encuentra cumpliendo su sentencia.

De la misma forma se deben permitir realizar búsquedas de los acusados y modificaciones o eliminaciones según corresponda.

14. Registrar, buscar, modificar y eliminar los datos de los juicios.

El sistema posibilitará registrar los datos correspondientes a todos los juicios, se hayan cebrado suspendido o resuelto a través de otras vías, los cuales son: identificador del juicio, el número de la causa de la que proviene, la fecha de celebración y el juez asignado a éste.

También se dará la posibilidad de realizar búsquedas de juicios por sus identificadores, modificaciones y eliminaciones de los datos de los juicios.

15. Registrar, buscar y modificar los datos de los juicios suspendidos.

Cuando se suspende un juicio el sistema debe permitir registrar el identificador del mismo, la fecha de suspensión, el motivo, las medidas tomadas y la fecha en que se reanuda éste.

Se debe dar la posibilidad hacer búsquedas a los juicios suspendidos y modificaciones en caso de que sean necesarias.

16. Registrar, buscar y modificar los datos de los juicios realizados.

Cuando se realiza un juicio el sistema debe permitir registrar el identificador del mismo, la modalidad, el número de sesiones y la fecha de culminación.

Se debe dar la posibilidad hacer búsquedas a los juicios realizados y modificaciones en caso de que sean necesarias.

17. Registrar, buscar y modificar los datos de los juicios resueltos por otras vías.

Cuando se realiza un juicio y se resuelve el mismo por otras vías, el sistema debe permitir registrar el identificador del mismo, la fecha de resolución, el motivo, el destino y la fecha de remisión.

Se debe dar la posibilidad hacer búsquedas a los juicios resueltos por otras vías y modificaciones en caso de que sean necesarias.

2.2.2 Requisitos No Funcionales.

Los requisitos no funcionales mencionados a continuación son los que tienen correspondencia con el desarrollo de la Base de Datos. (Dirigirse a los anexos para ver todos los requisitos no funcionales del sistema).

1. Rendimiento.

La aplicación debe estar concebida para el consumo mínimo de recursos y el sistema debe ser capaz de formular la respuesta lo más rápido posible.

2. Soporte.

Se requiere que esté instalado un Gestor de Base de Datos que soporte grandes volúmenes de datos y velocidad de procesamiento.

3. Portabilidad.

El sistema deberá ser compatible con el sistema operativo Linux y con Windows (Versiones como 2000 y XP), siendo además accesible principalmente con el navegador Mozilla.

4. Hardware.

Para el servidor de Base de Datos se requiere tarjeta de red, que tenga al menos 256MB de RAM y 1 GB de disco duro; y que el procesador tenga 1.2 GHz de velocidad como mínimo.

5. Software.

Los servidores, incluido el de la Base de Datos, deben tener instalados el sistema operativo Windows 2000 o superior, o Linux preferencialmente. La Base de Datos será implementada en el Gestor de Bases de datos PostgreSQL.

6. Seguridad.

El sistema debe comunicarse usando un protocolo seguro, (https). Los datos no pueden viajar de forma transparente por la red, deben ser encriptados. Se debe mantener la integridad de la información, es decir que no se pierda durante su almacenamiento o transporte. También se deben realizar auditoría a los principales eventos dentro del sistema, registrando al usuario y los eventos efectuados.

7. Confiabilidad.

La información manejada por el sistema deberá estar protegida de acceso no autorizado y divulgación.

8. Integridad.

La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción.

9. Fiabilidad.

Debe garantizarse el resguardo de la información, así como la grabación periódica de la Base de Datos, de forma tal que se posibilite la reinstalación del sistema y los datos, en caso de algún problema presentado en la explotación del mismo.

10. Legales.

El sistema se debe basar en el documento que expone las normativas para el desarrollo de softwares en las FAR. La mayoría de las herramientas de desarrollo serán libres y del resto, las licencias deben estar avaladas. El sistema tendrá en cuenta lo establecido por la ¨ Ley de procedimiento penal Militar¨ y la ¨ Ley número 97 de los tribunales Militares¨ en todo lo referido al desarrollo del trabajo judicial.

2.3 Estrategia de integración de la solución con otros módulos o sistemas.

Dado que la Fiscalía Militar Principal comenzará la elaboración de un proyecto para el control informático de los Expedientes de Fase Preparatoria, dicho sistema pudiera brindar como salida un flujo informativo que incluyera los datos que se recogen como entrada en el registro de los expedientes del Sistema Informático de los Tribunales Militares de Región.

Según el documento donde se plantean los lineamientos, principios y normativas para el desarrollo del software en las FAR, las BD del Sistema de estructuras serán la base normativa y de consulta para el resto de las aplicaciones, por lo que la Base de Datos del Sistema Informático de los Tribunales Militares de Región se nutrirá de ellas para actualizar sus nomencladores de unidades militares.

2.4 Descripción de la arquitectura y fundamentación.

El diseño del sistema de los Tribunales Militares está basado en una arquitectura orientada a objetos y servicios, para que pueda ampliarse con facilidad en caso de que las necesidades aumenten. Se utiliza la arquitectura en capas, facilitando el desarrollo y posibilitando que si se produce algún cambio sólo se ataca al nivel requerido sin tener que revisar entre código mezclado.

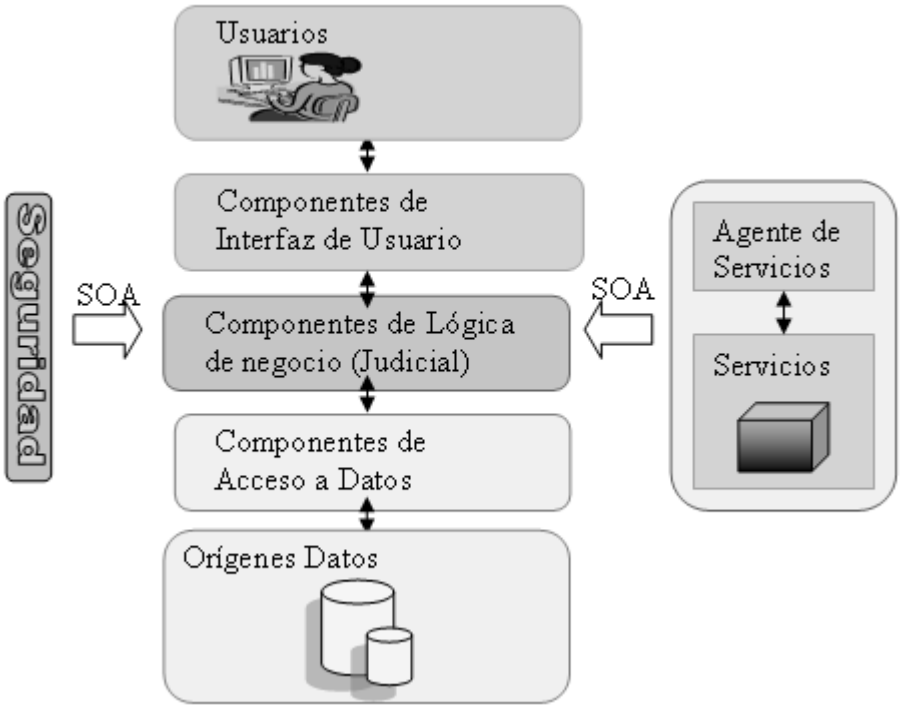


Figura 10. Arquitectura en capas. Tipos de componentes utilizados.

Los tipos de componentes identificados en el escenario de diseño son los siguientes:

1 **Componentes de interfaz de usuario (IU)**. Las interfaces de usuario se implementaron utilizando formularios, controles y funciones que permiten procesar y dar formato a los datos de los usuarios, así como adquirir y validar los datos entrantes procedentes de éstos.

2 **Componentes de lógica de negocio**. Independientemente de si el proceso consta de un único paso o de un flujo de trabajo organizado, la aplicación requiere el uso de componentes que implementen reglas y realicen tareas.

3 **Componentes de acceso a datos**. La aplicación y los servicios necesitan obtener acceso a un almacén de datos en un momento determinado del proceso judicial. Por ejemplo, la aplicación de los tribunales necesita recuperar los datos de un acusado de la Base de Datos para mostrar al usuario los detalles de los mismos, así como insertar dicha información en la misma cuando un usuario realiza una revisión. Por tanto, es razonable abstraer la lógica necesaria para obtener acceso a los datos en una capa independiente de componentes lógicos de acceso a datos, ya que de este modo se centraliza la funcionalidad de acceso a datos y se facilita la configuración y el mantenimiento de la misma.

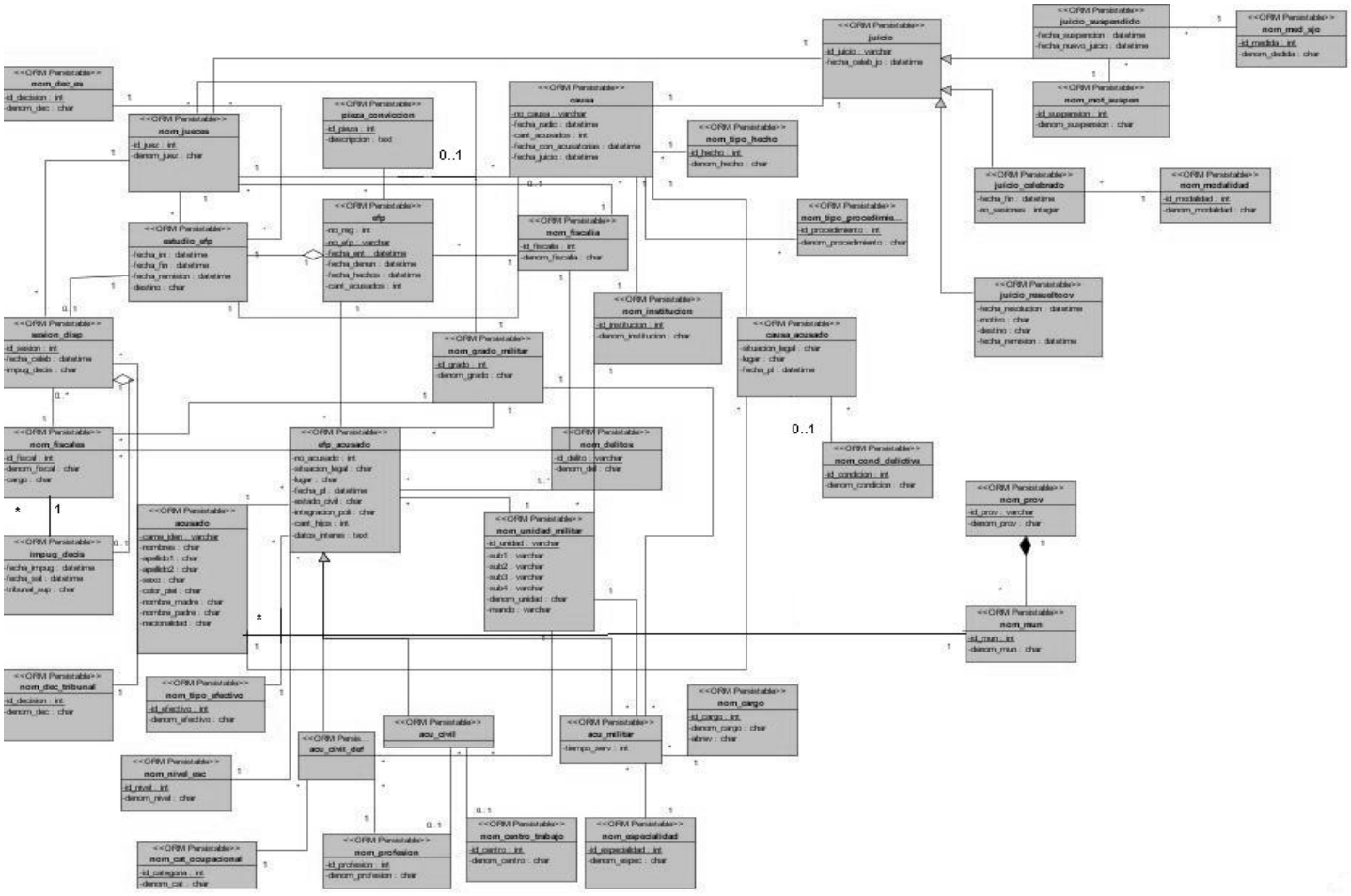
4 **Servicios**. Cuando un componente judicial requiere el uso de la funcionalidad proporcionada por un servicio externo, tal vez sea necesario hacer uso de código para administrar la semántica de la comunicación con dicho servicio. Los agentes de servicios permiten aislar las idiosincrasias de las llamadas a varios servicios desde la aplicación y pueden proporcionar servicios adicionales, como la asignación básica del formato de los datos que expone el servicio al formato que requiere la aplicación.

5 **Componentes de seguridad:** La directiva de seguridad se ocupa de la autenticación, autorización, comunicación segura, auditoría y administración de perfiles.

2.5 Diagrama de clases persistentes obtenido a partir del diagrama de clases del diseño.

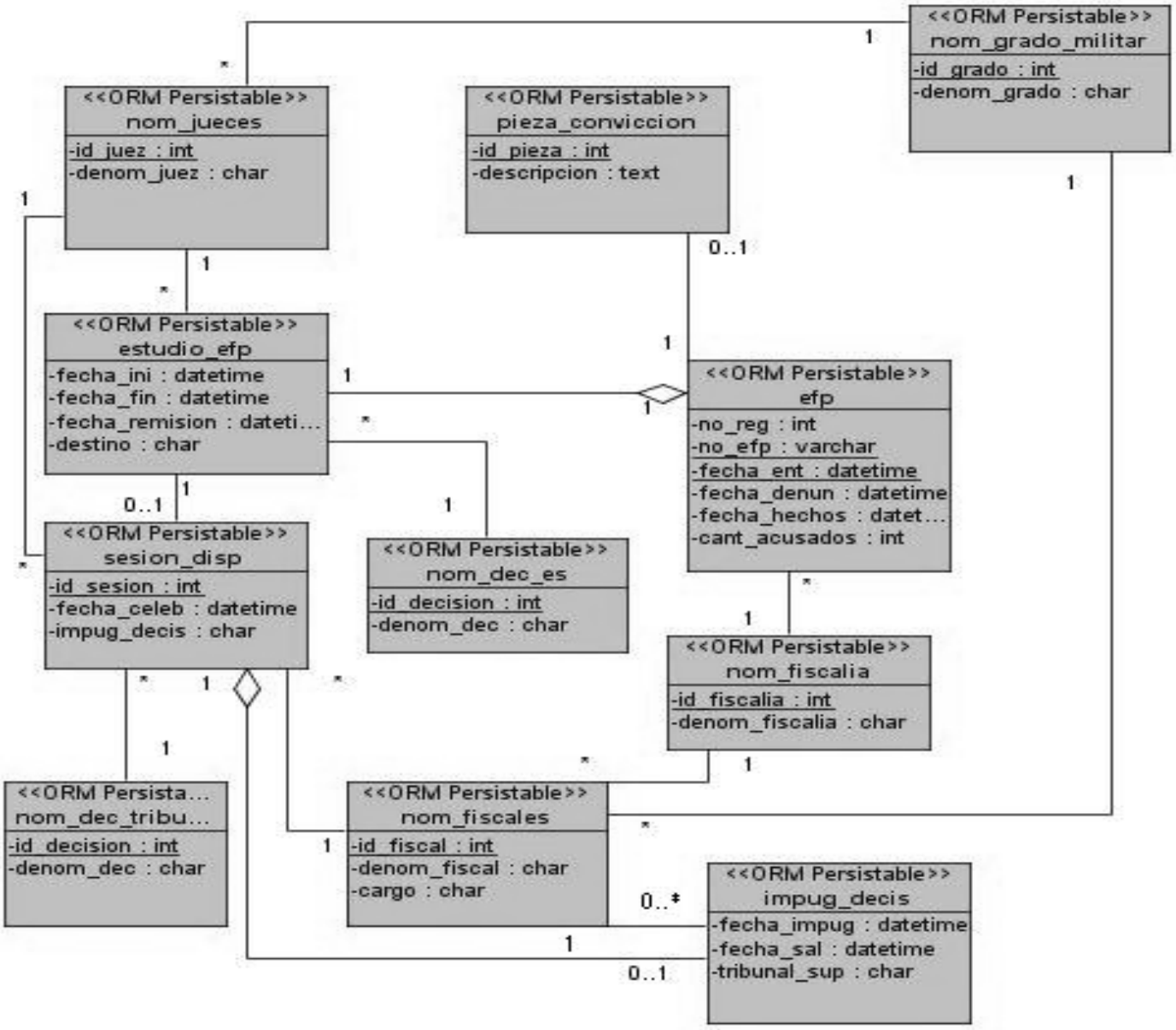
A continuación se muestra el diagrama de clases persistentes obtenido a partir del diagrama de clases del diseño, el cual ha sido fraccionado en varios submodelos para una mejor comprensión.

Diagrama general

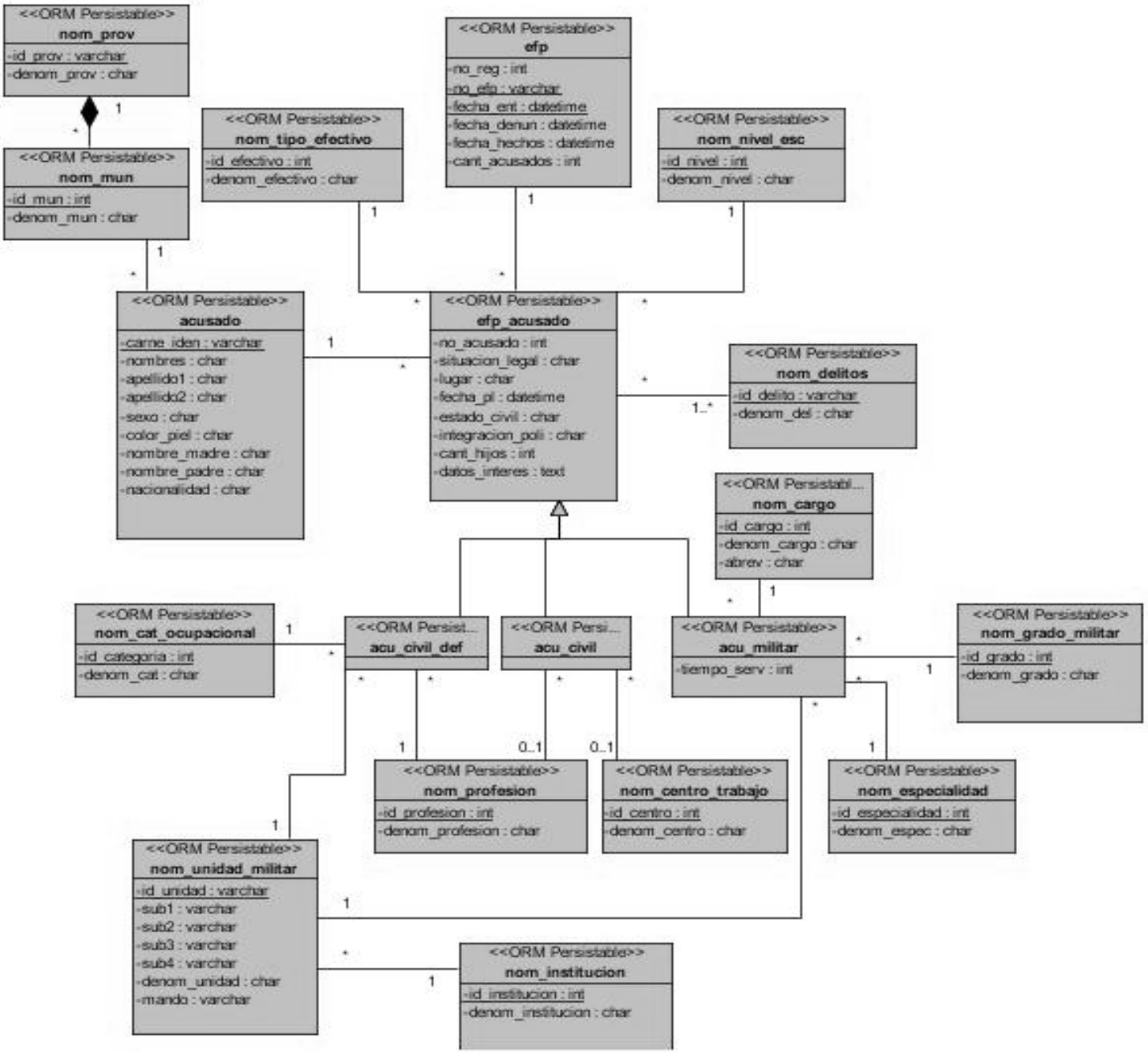


.

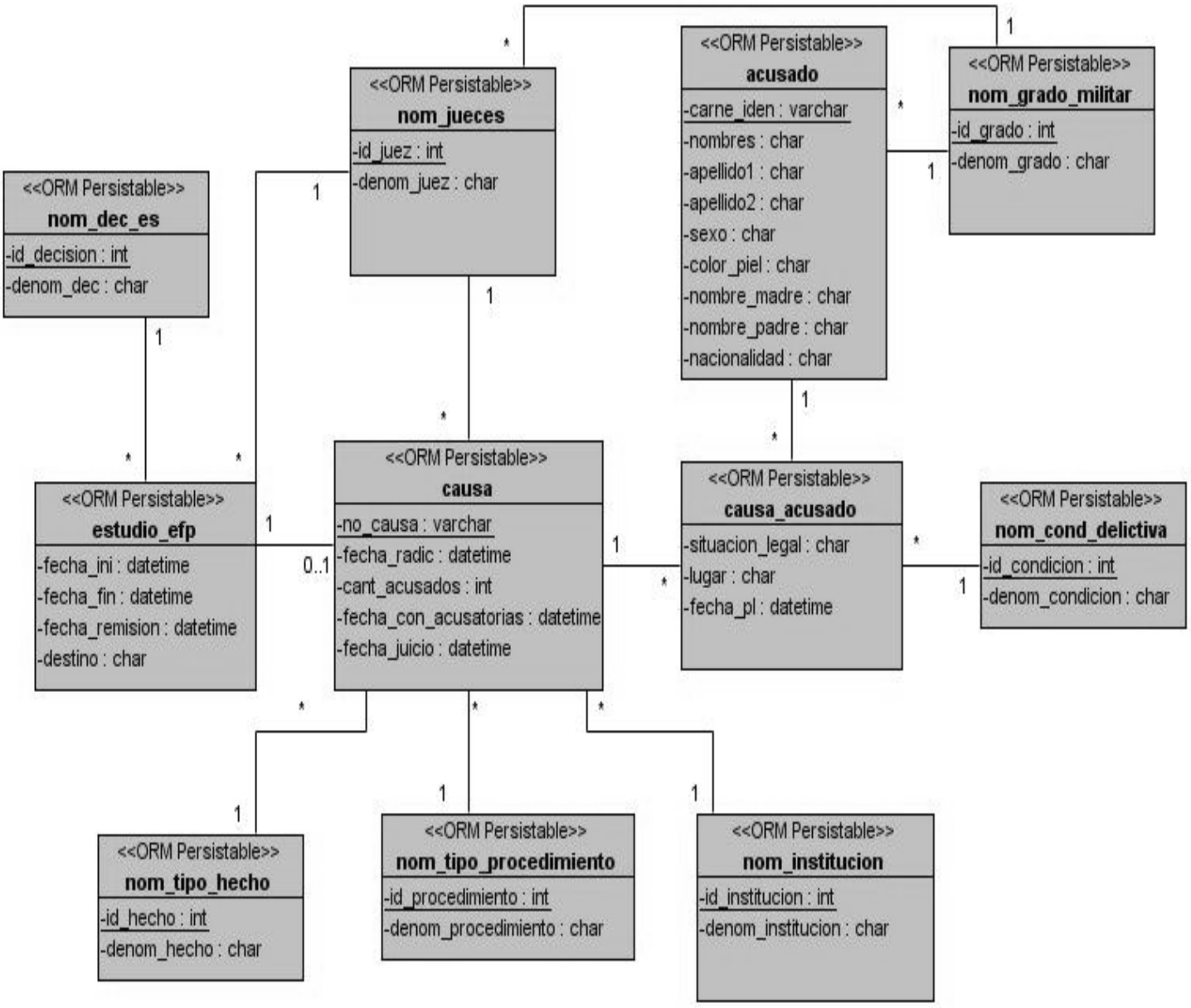
Submodelo1


<<ORM Persistable>>
nom_grado_militar
-id_grado : int
-denom_grado : char
<<ORM Persistable>>
nom_jueces
-id_juez : int
-denom_juez : char
<<ORM Persistable>>
pieza_conviccion
-id_pieza : int
-descripcion : text
<<ORM Persistable>>
estudio_efp
-fecha_ini : datetime
-fecha_fin : datetime
-fecha_remision : dateti...
-destino : char
<<ORM Persistable>>
efp
-no_reg : int
-no_efp : varchar
-fecha_ent : datetime
-fecha_denun : datetime
-fecha_hechos : datet...
-cant_acusados : int
<<ORM Persistable>>
sesion_disp
-id_sesion : int
-fecha_celeb : datetime
-impug_decis : char
<<ORM Persistable>>
nom_dec_es
-id_decision : int
-denom_dec : char
<<ORM Persistable>>
nom_fiscalia
-id_fiscalia : int
-denom_fiscalia : char
<<ORM Persista...
nom_dec_tribu...
-id_decision : int
-denom_dec : char
<<ORM Persistable>>
nom_fiscales
-id_fiscal : int
-denom_fiscal : char
-cargo : char
<<ORM Persistable>>
impug_decis
-fecha_impug : datetime
-fecha_sal : datetime
-tribunal_sup : char

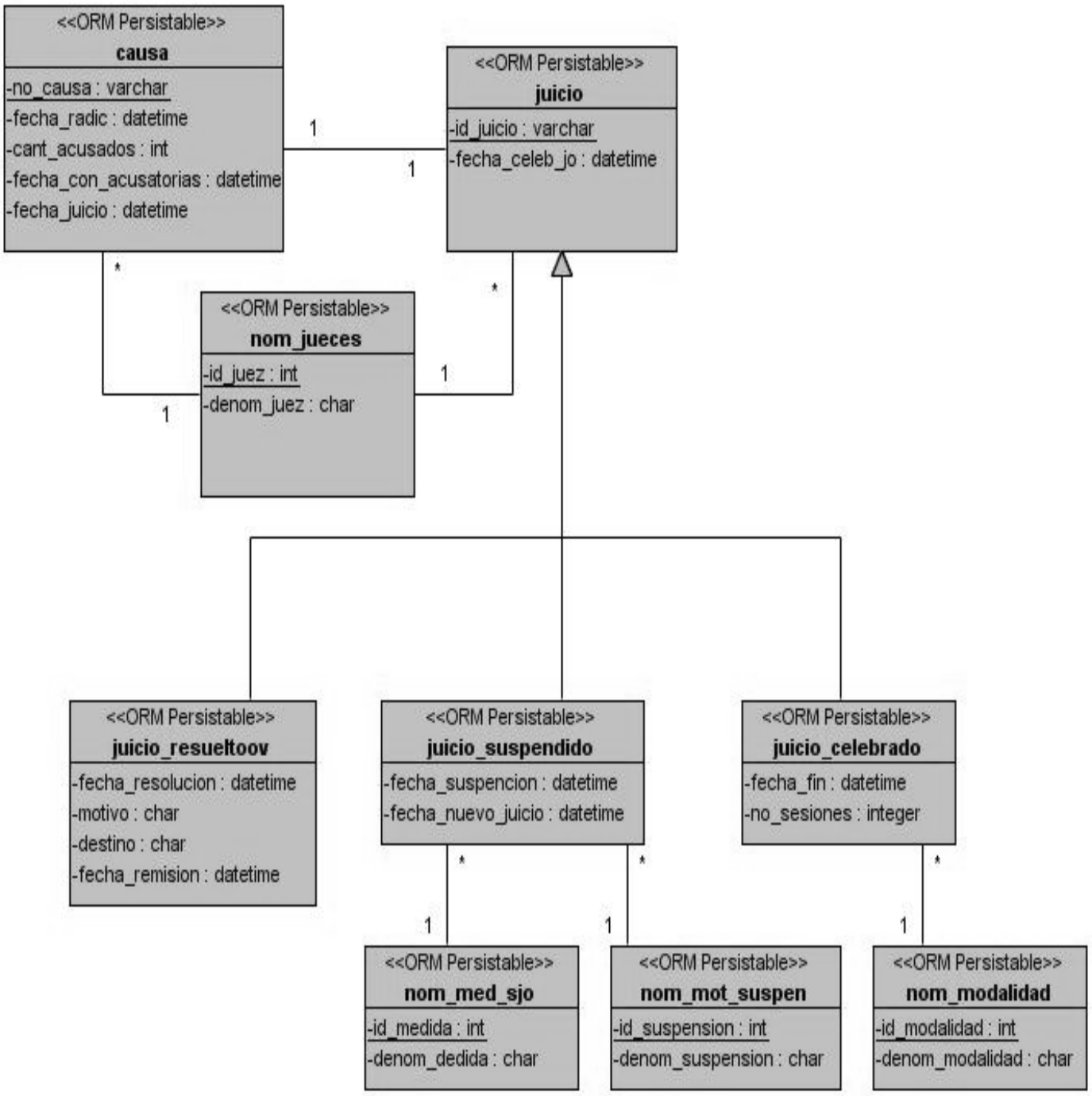
Submodelo2


<<ORM Persistable>>
nom_prov
-id_prov : varchar
-denom_prov : char
<<ORM Persistable>>
nom_mun
-id_mun : int
-denom_mun : char
<<ORM Persistable>>
nom_tipo_efectivo
-id_efectivo : int
-denom_efectivo : char
<<ORM Persistable>>
efp
-no_reg : int
-no_efp : varchar
-fecha_ent : datetime
-fecha_denun : datetime
-fecha_hechos : datetime
-cant_acusados : int
<<ORM Persistable>>
nom_nivel_esc
-id_nivel : int
-denom_nivel : char
<<ORM Persistable>>
acusado
-carne_iden : varchar
-nombres : char
-apellido1 : char
-apellido2 : char
-sexo : char
-color_piel : char
-nombre_madre : char
-nombre_padre : char
-nacionalidad : char
<<ORM Persistable>>
efp_acusado
-no_acusado : int
-situacion_legal : char
-lugar : char
-fecha_pl : datetime
-estado_civil : char
-integracion_poli : char
-cant_hijos : int
-datos_interes : text
<<ORM Persistable>>
nom_delitos
-id_delito : varchar
-denom_del : char
<<ORM Persistabl...
nom_cargo
-id_cargo : int
-denom_cargo : char
-abrev : char
<<ORM Persistable>>
nom_cat_ocupacional
-id_categoria : int
-denom_cat : char
<<ORM Persist...
acu_civil_def
<<ORM Persi...
acu_civil
<<ORM Persistable>>
acu_militar
-tiempo_serv : int
<<ORM Persistable>>
nom_grado_militar
-id_grado : int
-denom_grado : char
<<ORM Persistable>>
nom_profesion
-id_profesion : int
-denom_profesion : char
<<ORM Persistable>>
nom_centro_trabajo
-id_centro : int
-denom_centro : char
<<ORM Persistable>>
nom_especialidad
-id_especialidad : int
-denom_espec : char
<<ORM Persistable>>
nom_unidad_militar
-id_unidad : varchar
-sub1 : varchar
-sub2 : varchar
-sub3 : varchar
-sub4 : varchar
-denom_unidad : char
-mando : varchar
<<ORM Persistable>>
nom_institucion
-id_institucion : int
-denom_institucion : char

Submodelo3


<<ORM Persistable>>
nom_jueces
-id_juez : int
-denom_juez : char
<<ORM Persistable>>
acusado
-carne_iden : varchar
-nombres : char
-apellido1 : char
-apellido2 : char
-sexo : char
-color_piel : char
-nombre_madre : char
-nombre_padre : char
-nacionalidad : char
<<ORM Persistable>>
nom_grado_militar
-id_grado : int
-denom_grado : char
<<ORM Persistable>>
nom_dec_es
-id_decision : int
-denom_dec : char
<<ORM Persistable>>
estudio_efp
-fecha_ini : datetime
-fecha_fin : datetime
-fecha_remision : datetime
-destino : char
<<ORM Persistable>>
causa
-no_causa : varchar
-fecha_radic : datetime
-cant_acusados : int
-fecha_con_acusatorias : datetime
-fecha_juicio : datetime
<<ORM Persistable>>
causa_acusado
-situacion_legal : char
-lugar : char
-fecha_pl : datetime
<<ORM Persistable>>
nom_cond_delictiva
-id_condicion : int
-denom_condicion : char
<<ORM Persistable>>
nom_tipo_hecho
-id_hecho : int
-denom_hecho : char
<<ORM Persistable>>
nom_tipo_procedimiento
-id_procedimiento : int
-denom_procedimiento : char
<<ORM Persistable>>
nom_institucion
-id_institucion : int
-denom_institucion : char

Submodelo 4


<<ORM Persistable>>
causa
-no_causa : varchar
-fecha_radic : datetime
-cant_acusados : int
-fecha_con_acusatorias : datetime
-fecha_juicio : datetime
<<ORM Persistable>>
juicio
-id_juicio : varchar
-fecha_celeb_jo : datetime
<<ORM Persistable>>
nom_jueces
-id_juez : int
-denom_juez : char
<<ORM Persistable>>
juicio_resueltoov
-fecha_resolucion : datetime
-motivo : char
-destino : char
-fecha_remision : datetime
<<ORM Persistable>>
juicio_suspendido
-fecha_suspencion : datetime
-fecha_nuevo_juicio : datetime
<<ORM Persistable>>
juicio_celebrado
-fecha_fin : datetime
-no_sesiones : integer
<<ORM Persistable>>
nom_med_sjo
-id_medida : int
-denom_dedida : char
<<ORM Persistable>>
nom_mot_suspen
-id_suspension : int
-denom_suspension : char
<<ORM Persistable>>
nom_modalidad
-id_modalidad : int
-denom_modalidad : char

2.5.1 Representación de las clases persistentes.

A continuación se representan las clases persistentes con sus atributos y tipos de datos correspondientes.

Nombre: efp	
Tipo de clase: entidad	
Atributo	**Tipo**
no_reg	int
no_efp	varchar
fecha_ent	datetime
fecha_hechos	datetime
fecha_denun	datetime
cant_acusados	int

Nombre: pieza_conviccion	
Tipo de clase: entidad	
Atributo	**Tipo**
id_pieza	int
descripcion	text

Nombre: acusado	
Tipo de clase: entidad	
Atributo	**Tipo**
carne_iden	varchar
sexo	char
nombres	char
apellido1	char
apellido2	char
color_piel	char
nombre_madre	char
nombre_padre	char
nacionalidad	char

Nombre: efp_acusado	
Tipo de clase: entidad	
Atributo	**Tipo**
no_acusado	int
situacion_legal	char
lugar	char
fecha_pl	datetime
estado_civil	char
integracion_poli	char
cant_hijos	integer
datos_interes	text

Nombre: acusado_militar	
Tipo de clase: entidad	
Atributo	**Tipo**
tiempo_servicio	int

Nombre: estudio_efp	
Tipo de clase: entidad	
Atributo	**Tipo**
fecha_ini	datetime
fecha_fin	datetime
fecha_remision	datetime
destino	char

Nombre: sesion_disp	
Tipo de clase: entidad	
Atributo	**Tipo**
id_sesion	int
fecha_celeb	datetime
impug_decis	char

Nombre: impug_decis	
Tipo de clase: entidad	
Atributo	**Tipo**
fecha_impug	datetime
fecha_sal	datetime
tribunal_sup	char

Nombre: causa	
Tipo de clase: entidad	
Atributo	**Tipo**
no_causa	varchar
fecha_radic	datetime
cant_acusados	int
fecha_con_acusatorias	datetime
fecha_juicio_oral	datetime

Nombre: causa_acusado	
Tipo de clase: entidad	
Atributo	**Tipo**
situacion_legal	char
lugar	char
fecha_pl	datetime

Nombre: juicio	
Tipo de clase: entidad	
Atributo	**Tipo**
id_juicio	int
fecha_celeb_jo	datetime

Nombre: juicio_suspendido	
Tipo de clase: entidad	
Atributo	**Tipo**
fecha_suspencion	datetime
fecha_nuevo_juicio	datetime

Nombre: juicio_celebrado	
Tipo de clase: entidad	
Atributo	**Tipo**
fecha_fin	datetime
no_sesiones	int

Nombre: juicio_resueltoov	
Tipo de clase: entidad	
Atributo	**Tipo**
fecha_resolucion	datetime
motivo	char
destino	char
fecha_nuevo_juicio	datetime

2.6 Diseño de la BD.

Las Bases de Datos necesitan de una definición de su estructura que le permita almacenar datos, reconocer el contenido, y recuperar la información. La estructura tiene que ser desarrollada para satisfacer las necesidades de las aplicaciones que la usarán.

La puesta en práctica de la Base de Datos es el paso final en el desarrollo de aplicaciones de soporte del negocio. Se conforman con los requisitos del proceso del negocio, que son la primera abstracción de la vista de la Base de Datos.

Para el diseño de la Base de Datos se realizó el diagrama de clases persistentes, mostrado anteriormente, y el diagrama entidad relación en el Case Studio.

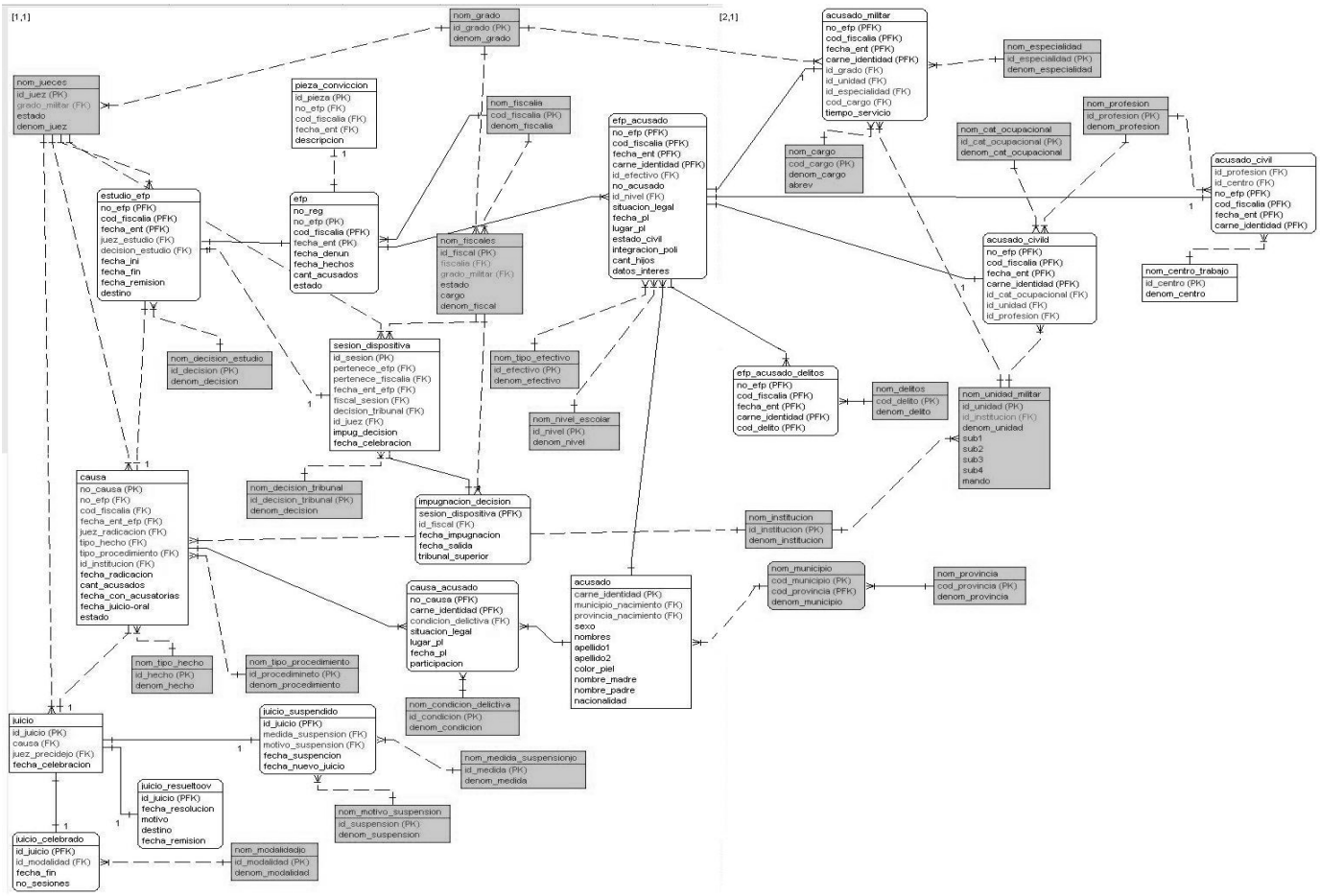
Una clase persistente (persistent) es una clase entidad que tiene la capacidad de mantener su valor en el espacio y en el tiempo. Lo contrario son las clases temporales (transient) que son manejadas y almacenadas por el sistema en tiempo de ejecución por lo que dejan de existir cuando termina el programa. (Cutiño, 2006)

En el diagrama entidad relación se especificaron los detalles físicos de la Base de Datos.

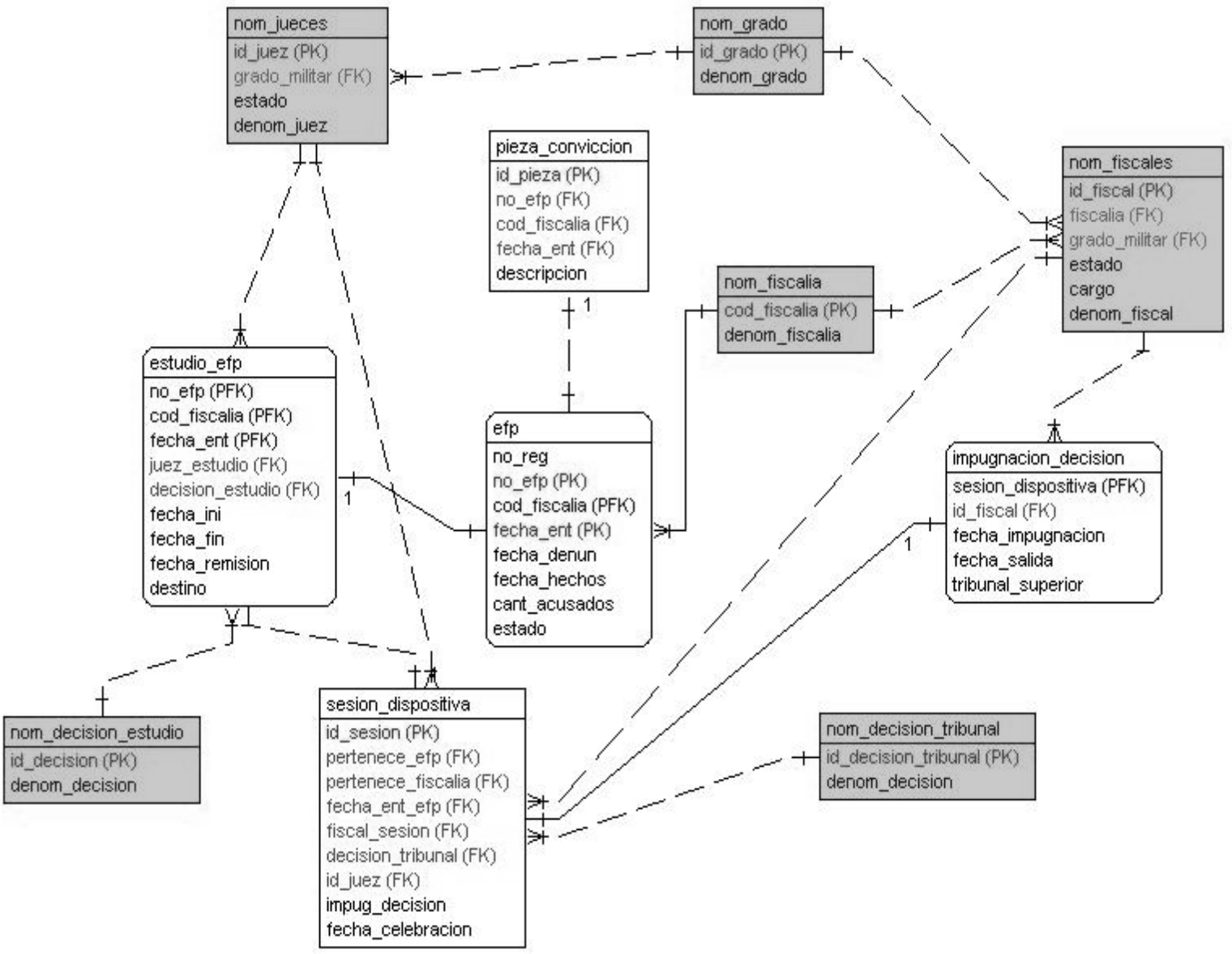
2.6.1 Diagrama Entidad Relación de la BD.

La siguiente figura representa el diagrama entidad relación correspondiente a la Base de Datos del Sistema Informático de los tribunales Militares de Región. A continuación se representan y describen los diferentes submodelos en los que se ha fragmentado la misma para una mejor comprensión.

Diagrama general



Submodelo1

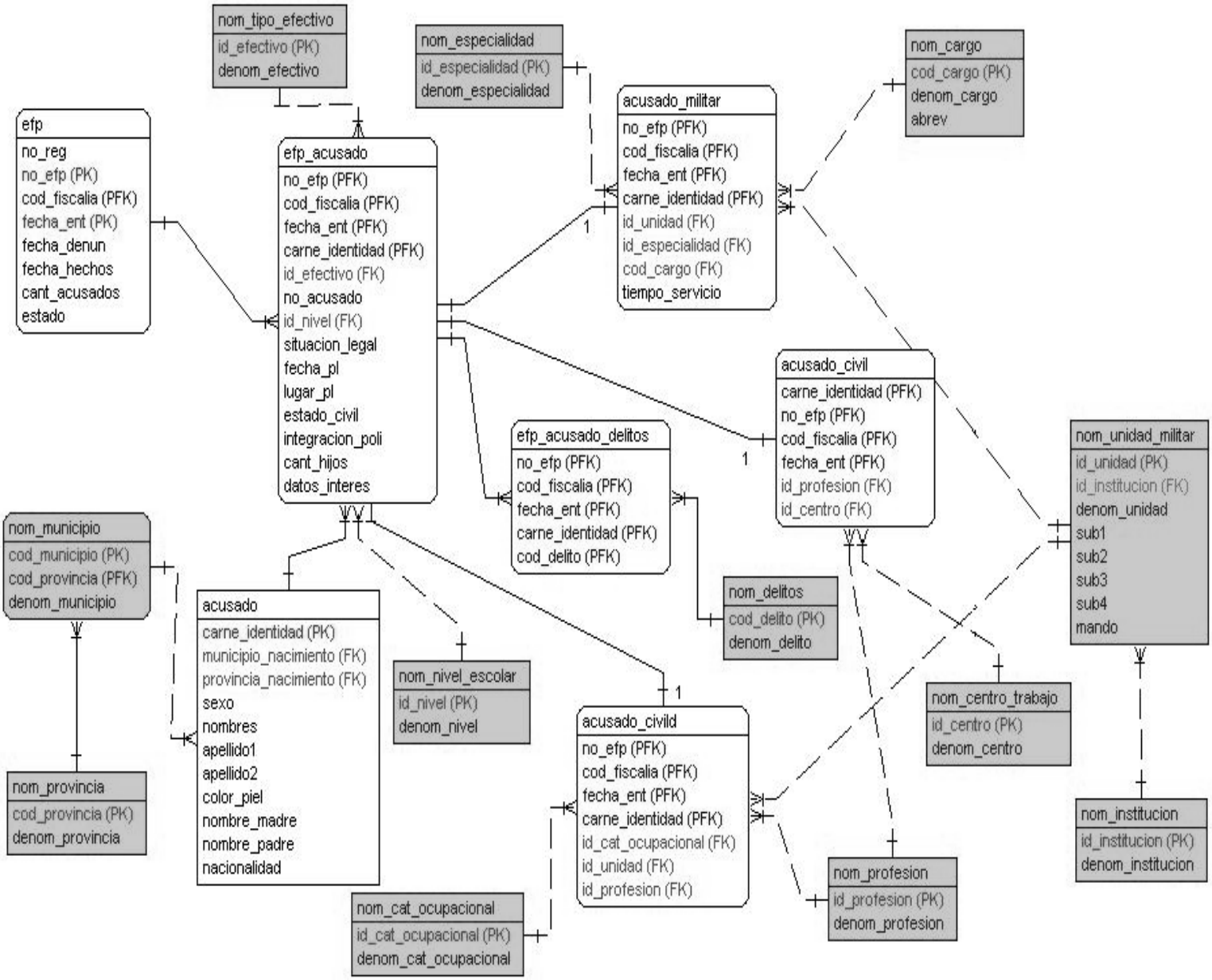

nom_jueces
id_juez (PK)
grado_militar (FK)
estado
denom_juez
nom_grado
id_grado (PK)
denom_grado
pieza_conviccion
id_pieza (PK)
no_efp (FK)
cod_fiscalia (FK)
fecha_ent (FK)
descripcion
nom_fiscales
id_fiscal (PK)
fiscalia (FK)
grado_militar (FK)
estado
cargo
denom_fiscal
nom_fiscalia
cod_fiscalia (PK)
denom_fiscalia
estudio_efp
no_efp (PFK)
cod_fiscalia (PFK)
fecha_ent (PFK)
juez_estudio (FK)
decision_estudio (FK)
fecha_ini
fecha_fin
fecha_remision
destino
efp
no_reg
no_efp (PK)
cod_fiscalia (PFK)
fecha_ent (PK)
fecha_denun
fecha_hechos
cant_acusados
estado
impugnacion_decision
sesion_dispositiva (PFK)
id_fiscal (FK)
fecha_impugnacion
fecha_salida
tribunal_superior
nom_decision_estudio
id_decision (PK)
denom_decision
sesion_dispositiva
id_sesion (PK)
pertenece_efp (FK)
pertenece_fiscalia (FK)
fecha_ent_efp (FK)
fiscal_sesion (FK)
decision_tribunal (FK)
id_juez (FK)
impug_decision
fecha_celebracion
nom_decision_tribunal
id_decision_tribunal (PK)
denom_decision
1
1
1

Primeramente se introducen los datos correspondientes al expediente de fase preparatoria, registrándose la fiscalía a la que pertenece. Si este expediente trae asociadas las piezas de convicción con las que se produjeron los delitos, entonces se describen todas en un mismo registro. Luego se almacenan los datos del estudio del EFP, que va a poseer el identificador del juez que realizó el análisis, el cual tiene un grado militar, y la decisión que éste tomó a partir del estudio efectuado.

Si la decisión derivada del análisis del EFP consistió en que el mismo debía ir a sesión dispositiva, se registran los datos correspondientes a ésta, entre los que se encuentran el identificador del juez que participó en la misma; el fiscal, que pertenece a una fiscalía y cuenta con un grado militar y la decisión que tomó el tribunal sobre los aspectos tratados en esta sesión.

Si se decide impugnar la decisión tomada en la sesión dispositiva, se almacena la información correspondiente a ésta y el fiscal que realizó la impugnación de la misma.

Submodelo2

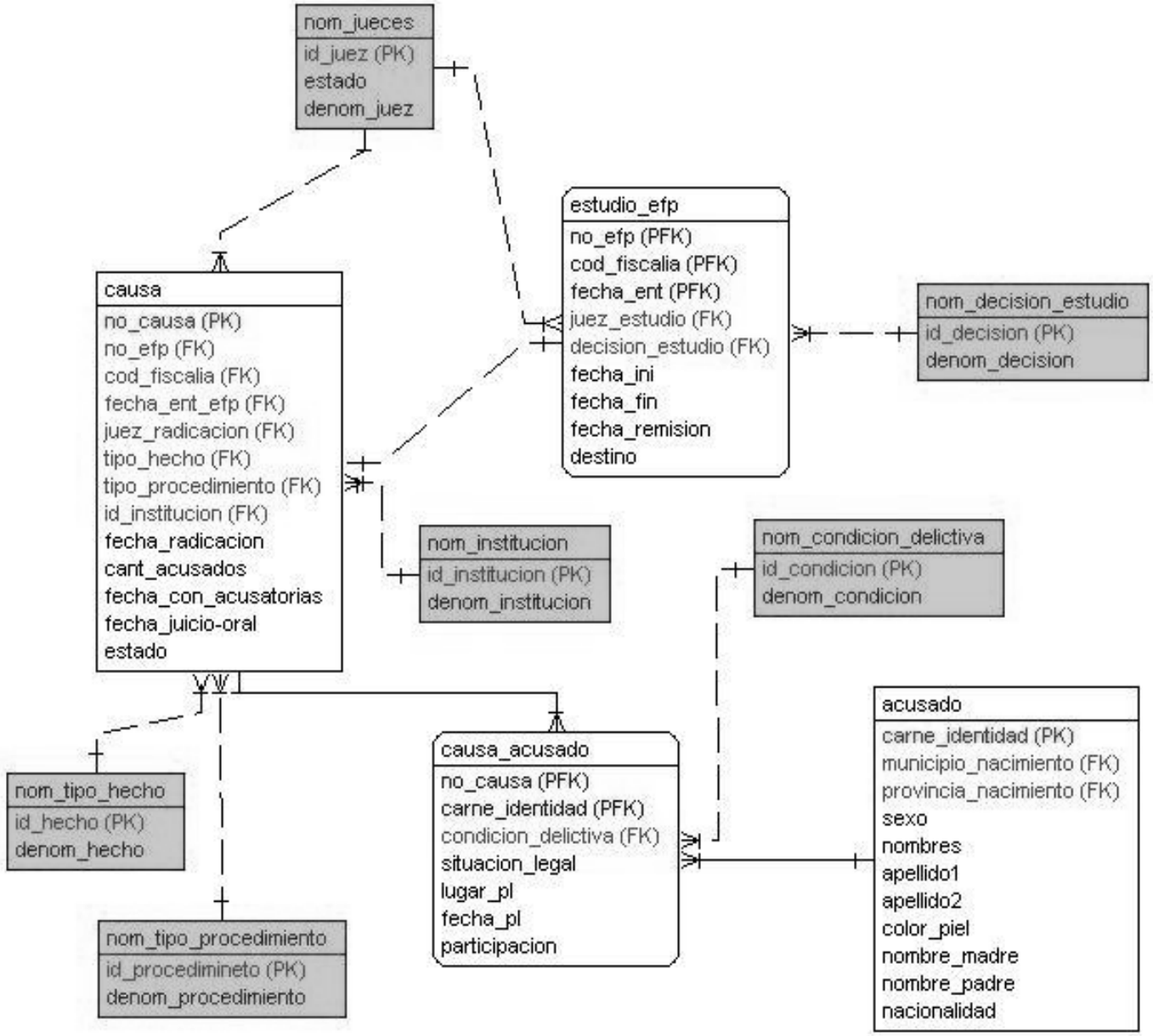

nom_tipo_efectivo
id_efectivo (PK)
denom_efectivo
nom_especialidad
id_especialidad (PK)
denom_especialidad
nom_cargo
cod_cargo (PK)
denom_cargo
abrev
acusado_militar
no_efp (PFK)
cod_fiscalia (PFK)
fecha_ent (PFK)
carne_identidad (PFK)
id_unidad (FK)
id_especialidad (FK)
cod_cargo (FK)
tiempo_servicio
efp
no_reg
no_efp (PK)
cod_fiscalia (PFK)
fecha_ent (PK)
fecha_denun
fecha_hechos
cant_acusados
estado
efp_acusado
no_efp (PFK)
cod_fiscalia (PFK)
fecha_ent (PFK)
carne_identidad (PFK)
id_efectivo (FK)
no_acusado
id_nivel (FK)
situacion_legal
fecha_pl
lugar_pl
estado_civil
integracion_poli
cant_hijos
datos_interes
1
1
acusado_civil
carne_identidad (PFK)
no_efp (PFK)
cod_fiscalia (PFK)
fecha_ent (PFK)
id_profesion (FK)
id_centro (FK)
efp_acusado_delitos
no_efp (PFK)
cod_fiscalia (PFK)
fecha_ent (PFK)
carne_identidad (PFK)
cod_delito (PFK)
nom_unidad_militar
id_unidad (PK)
id_institucion (FK)
denom_unidad
sub1
sub2
sub3
sub4
mando
nom_municipio
cod_municipio (PK)
cod_provincia (PFK)
denom_municipio
acusado
carne_identidad (PK)
municipio_nacimiento (FK)
provincia_nacimiento (FK)
sexo
nombres
apellido1
apellido2
color_piel
nombre_madre
nombre_padre
nacionalidad
nom_delitos
cod_delito (PK)
denom_delito
nom_nivel_escolar
id_nivel (PK)
denom_nivel
1
acusado_civild
no_efp (PFK)
cod_fiscalia (PFK)
fecha_ent (PFK)
carne_identidad (PFK)
id_cat_ocupacional (FK)
id_unidad (FK)
id_profesion (FK)
nom_centro_trabajo
id_centro (PK)
denom_centro
nom_provincia
cod_provincia (PK)
denom_provincia
nom_institucion
id_institucion (PK)
denom_institucion
nom_profesion
id_profesion (PK)
denom_profesion
nom_cat_ocupacional
id_cat_ocupacional (PK)
denom_cat_ocupacional

Cuando se introduce el EFP en el sistema, se realiza una búsqueda de los acusados que vienen asociados en el mismo, si no se encuentran en la tabla “acusados” entonces se almacenan los datos correspondientes a ellos, entre los que se encuentra el código de su provincia y municipio de residencia De la relación entre estas dos tablas surge una entidad nueva, denominada “efp_acusado”, en la que se registra la información de cada acusado en el expediente con que se relaciona. Esta tabla va a tener, además, el identificador del tipo de efectivo que es el acusado y el nivel escolar que posee.

A partir de la relación entre las tablas “efp_acusado” y la de los delitos se crea la entidad “efp_acusado_delitos”, en la que residen los delitos pertenecientes a los acusados en un expediente determinado.

Por el hecho de que los acusados civiles, civiles de las FAR y los militares tienen algunos atributos diferentes, correspondientes a los EFP con que se relacionan, la información de cada tipo de acusado se almacena en una tabla distinta. Los datos de los acusados civiles van a coexistir en la entidad “acusado_civil”, guardándose en ella el centro de trabajo y la profesión, si los tienen. Los datos de los acusados civiles de las FAR se van a encontrar en “acusado_civild”, y entre ellos la profesión, categoría ocupacional que ocupan y la unidad militar a la que pertenecen. La información de los acusados militares va a estar almacenada en “acusado _ militar”, este tipo de acusado va a tener un grado, un cargo, una especialidad y una unidad militar asociada.

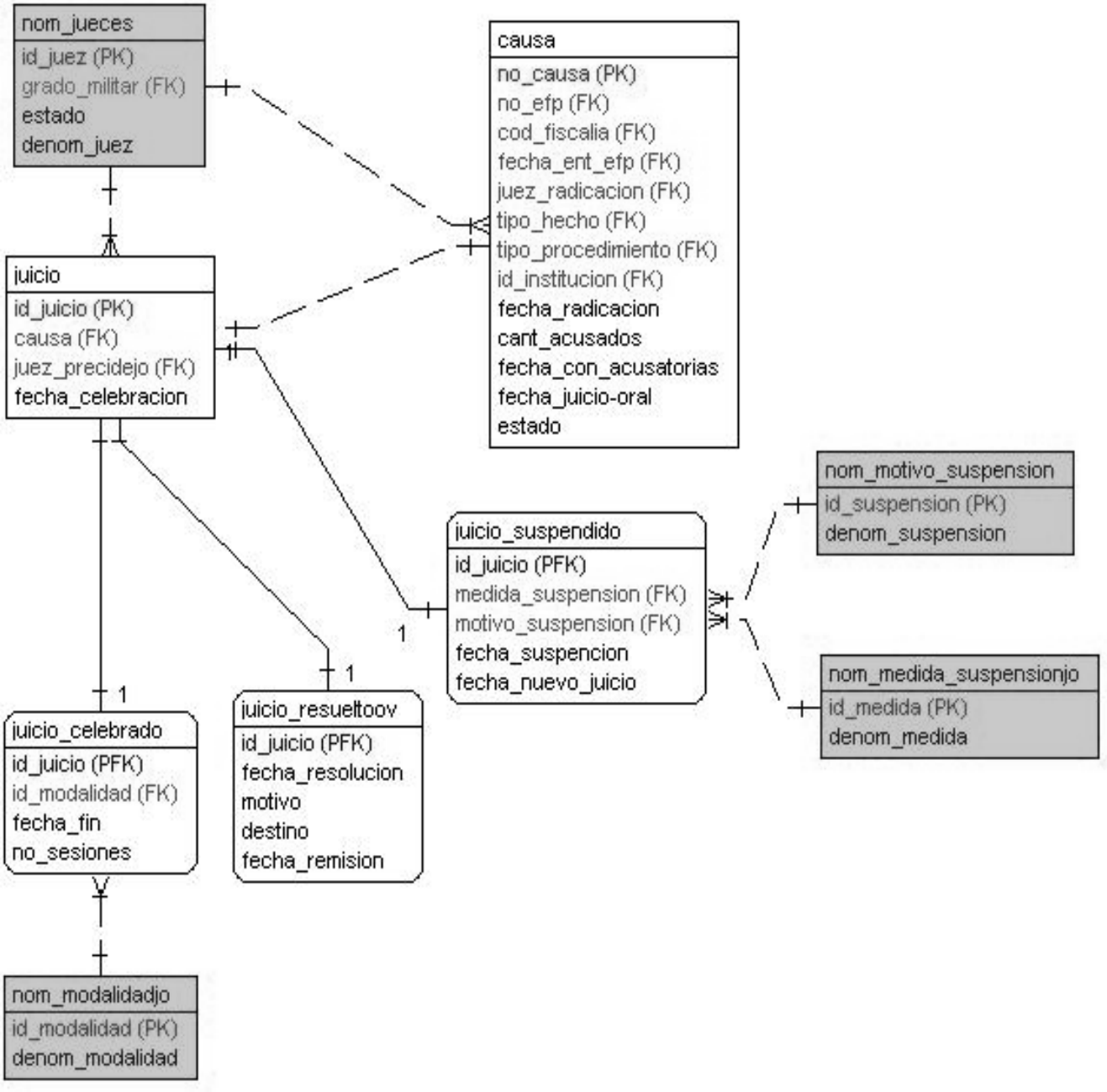
Submodelo3


nom_jueces
id_juez (PK)
estado
denom_juez
estudio_efp
no_efp (PFK)
cod_fiscalia (PFK)
fecha_ent (PFK)
juez_estudio (FK)
decision_estudio (FK)
fecha_ini
fecha_fin
fecha_remision
destino
nom_decision_estudio
id_decision (PK)
denom_decision
causa
no_causa (PK)
no_efp (FK)
cod_fiscalia (FK)
fecha_ent_efp (FK)
juez_radicacion (FK)
tipo_hecho (FK)
tipo_procedimiento (FK)
id_institucion (FK)
fecha_radicacion
cant_acusados
fecha_con_acusatorias
fecha_juicio-oral
estado
nom_institucion
id_institucion (PK)
denom_institucion
nom_condicion_delictiva
id_condicion (PK)
denom_condicion
nom_tipo_hecho
id_hecho (PK)
denom_hecho
causa_acusado
no_causa (PFK)
carne_identidad (PFK)
condicion_delictiva (FK)
situacion_legal
lugar_pl
fecha_pl
participacion
acusado
carne_identidad (PK)
municipio_nacimiento (FK)
provincia_nacimiento (FK)
sexo
nombres
apellido1
apellido2
color_piel
nombre_madre
nombre_padre
nacionalidad
nom_tipo_procedimiento
id_procedimineto (PK)
denom_procedimiento

Cuando se realiza el estudio del expediente y se toma la decisión de radicar el mismo para formar causa, se almacenan los datos correspondientes a la misma, entre ellos se encuentran el juez que efectuó la radicación, la institución relacionada con ésta, el tipo de procedimiento que se será llevado a cabo y el tipo de hecho con el que se relacionan los delitos.

De la relación de la causa con los acusados se crea una entidad nueva denominada "causa_acusado", que contiene información correspondiente a cada acusado en una causa determinada, y en las que se registra la condición delictiva del mismo.

Submodelo4


nom_jueces
id_juez (PK)
grado_militar (FK)
estado
denom_juez
causa
no_causa (PK)
no_efp (FK)
cod_fiscalia (FK)
fecha_ent_efp (FK)
juez_radicacion (FK)
tipo_hecho (FK)
tipo_procedimiento (FK)
id_institucion (FK)
fecha_radicacion
cant_acusados
fecha_con_acusatorias
fecha_juicio-oral
estado
juicio
id_juicio (PK)
causa (FK)
juez_precidejo (FK)
fecha_celebracion
nom_motivo_suspension
id_suspension (PK)
denom_suspension
juicio_suspendido
id_juicio (PFK)
medida_suspension (FK)
motivo_suspension (FK)
fecha_suspencion
fecha_nuevo_juicio
nom_medida_suspensionjo
id_medida (PK)
denom_medida
1
1
1
juicio_celebrado
id_juicio (PFK)
id_modalidad (FK)
fecha_fin
no_sesiones
juicio_resueltoov
id_juicio (PFK)
fecha_resolucion
motivo
destino
fecha_remision
nom_modalidadjo
id_modalidad (PK)
denom_modalidad

Cuando se forma la causa se procede a realizar el juicio oral, que tiene entre sus atributos el juez que efectuó el mismo. De la tabla "juicio" se derivan otras tres correspondientes a los juicios realizados, los resueltos por otras vías y los juicios que han sido suspendidos. Los juicios que se han celebrado tienen en su información la modalidad llevada a cabo en ellos, y los suspendidos el motivo y la medida de suspensión.

2.6.2 Descripción de las tablas.

En esta sección se describen de forma general los datos que se almacenan en cada tabla del diagrama entidad relación de la Base de Datos del Sistema Informático de los Tribunales Militares de Región, al igual que se explica brevemente lo que representan todos sus atributos.

Nombre: efp		
Descripción: Almacena los datos correspondientes a los expedientes de fase preparatoria que entran a los Tribunales Militares de Región. Un mismo expediente correspondiente a una fiscalía puede entrar al sistema varias veces, distinguiéndose precisamente por su fecha de entrada.		
Atributo	**Tipo**	**Descripción**
no_reg	serial	Número consecutivo que va ascendiendo a medida que se de entrada a un nuevo expediente.
no_efp	varchar(8)	Representa el número del expediente correspondiente a una fiscalía seguido de un / y luego el número del año.
fecha_ent	date	Es la fecha de entrada de ese expediente en el sistema.
cod_fiscalia	integer	Representa a la fiscalía a la que corresponde ese expediente.
fecha_hechos	date	Es la fecha en que ocurrieron los hechos delictivos.
fecha_denun	date	Fecha de denuncia de los delitos cometidos que dieron lugar a que se abriera un expediente.
cant_acusados	integer	Cantidad de acusados asociados a ese expediente.
estado	char(2)	Estado en que se encuentra el expediente. Si se sigue trabajando con él o ya no es necesaria su utilización por el momento.

Nombre: pieza_conviccion		
Descripción: Se describen las piezas que se utilizaron en la realización de los delitos.		
Atributo	**Tipo**	**Descripción**
id_pieza	serial	Número consecutivo ascendente que hace referencia a las piezas utilizadas por los acusados en los delitos.
descripcion	text	Se describen las piezas de convicción.
no_efp	varchar(8)	Hace referencia al expediente al que corresponde dicha pieza.
fecha_ent	date	Fecha de entrada del expediente al sistema.
cod_fiscalia	integer	Fiscalía a la que corresponde el expediente.

Nombre: acusado		
Descripción: Almacena los datos de todos los acusados que existen en ese Tribunal.		
Atributo	**Tipo**	**Descripción**
carne_identidad	varchar(11)	Se almacena el número del carné de identidad correspondiente a cada acusado. En el caso del acusado militar el carné tiene 6 dígitos
nombres	char(20)	Nombre de cada acusado.
apellido1	char(20)	Primer apellido del acusado.
apellido2	char(20)	Segundo apellido del acusado.
provincia_nacimiento	varchar(2)	Identificador de la provincia de nacimiento.
municipio_nacimiento	varchar(2)	Identificador del municipio de nacimiento.
sexo	char(1)	Sexo del acusado (F o M).
color_piel	char(1)	Color del acusado.
nombre_madre	char(15)	Nombre de la madre del acusado.
nombre_padre	char(15)	Nombre del padre del acusado.
nacionalidad	char(15)	Nacionalidad del acusado.

Nombre: efp_acusado		
Descripción: Se almacenan los datos de cada acusado pero con relación a un expediente determinado.		
Atributo	**Tipo**	**Descripción**
carne_identidad	varchar(11)	Identificación del acusado.
no_efp	varchar(8)	Expediente con el que se relaciona el acusado.
fecha_ent	date	Fecha de entrada del expediente al sistema.
cod_fiscalia	integer	Fiscalía a la que corresponde dicho expediente.
no_acusado	integer	Número correspondiente a un acusado en un expediente.
situación_legal	char(1)	Situación legal del acusado en el momento que está siendo procesado.
lugar_pl	char(2)	Si se encuentra en prisión preventiva, el lugar donde cumple la misma..
fecha_pl	date	Si está en prisión preventiva, fecha en la que pasó a este estado.
estado_civil	char(1)	Estado civil en el que se encuentra el acusado.
integración_poli	char(3)	Organización política a la que pertenece el acusado.
cant_hijos	integer	Cantidad de hijos que tiene el acusado.
id_efectivo	integer	Tipo de efectivo que es el acusado (militar, civil, civil de la defensa).
id_nivel	integer	Nivel escolar del acusado (identificador).
datos_interes	text	Otros datos que serán de interés en el proceso.

Nombre: efp_acusado_delitos		
Descripción: Representa los delitos correspondientes a un determinado acusado en un expediente.		
Atributo	**Tipo**	**Descripción**
carne_identidad	varchar(11)	Identificador del acusado.
no_efp	varchar(8)	Número del expediente al que pertenece.
cod_fiscalia	integer	Fiscalía a la que corresponde el expediente.
fecha_ent	date	Fecha de entrada de ese expediente al sistema.
cod_delito	varchar(6)	Identificador del delito cometido por el acusado.

Nombre: acusado_militar		
Descripción: Se almacenan los datos correspondientes a los acusados militares en un determinado expediente.		
Atributo	**Tipo**	**Descripción**
carne_identidad	varchar(11)	Identificador del acusado.
no_efp	varchar(8)	Número del expediente al que pertenece.
cod_fiscalia	integer	Fiscalía a la que corresponde el expediente.
fecha_ent	date	Fecha de entrada de ese expediente al sistema.
id_grado	integer	Grado del acusado.
cod_cargo	integer	Cargo del acusado.
id_unidad	varchar(4)	Unidad militar a la que corresponde el acusado militar.
id_especialidad	integer	Especialización del acusado.
tiempo_servicio	integer	Tiempo de servicio en la institución armada a la que pertenece el acusado militar.

Nombre: acusado_civild		
Descripción: Se almacenan los datos correspondientes a los acusados civiles de la defensa en un determinado expediente.		
Atributo	**Tipo**	**Descripción**
carne_identidad	varchar(11)	Identificador del acusado.
no_efp	varchar(8)	Número del expediente al que pertenece.
cod_fiscalia	integer	Fiscalía a la que corresponde el expediente.
fecha_ent	date	Fecha de entrada de ese expediente al sistema.
id_unidad	varchar(4)	Unidad militar a la que corresponde el acusado civil de la defensa.
id_profesion	integer	Profesión que ejerce el acusado.
id_cat_ocupacional	integer	Categoría ocupacional que posee el acusado.

Nombre: acusado_civil		
Descripción: Se almacenan los datos correspondientes a los acusados civiles en un determinado expediente.		
Atributo	**Tipo**	**Descripción**
carne_identidad	varchar(11)	Identificador del acusado.
no_efp	varchar(8)	Número del expediente al que pertenece.
cod_fiscalia	integer	Fiscalía a la que corresponde el expediente.
fecha_ent	date	Fecha de entrada de ese expediente al sistema.
id_profesion	integer	Profesión que ejerce el acusado.
id_centro	integer	Representa al centro de trabajo del acusado.

Nombre: estudio_efp		
Descripción: Se van a almacenar los datos correspondientes al proceso de estudio de los Expedientes de Fase Preparatoria.		
Atributo	**Tipo**	**Descripción**
no_efp	varchar(8)	Número del expediente estudiado.
fecha_ent	date	Fecha de entrada de ese expediente en el sistema.
cod_fiscalia	integer	Fiscalía a la que pertenece el expediente.
fecha_ini	date	Fecha de inicio del estudio del expediente.
fecha_fin	date	Fecha en que finaliza el estudio.
juez_estudio	integer	Identificador del juez que realizó el estudio.
decision_estudio	integer	Decisión que se tomó de acuerdo al estudio del expediente.
destino	varchar(25)	De acuerdo al estudio del expediente el destino que se le decidió dar al mismo.
fecha_remision	date	Fecha de remisión del expediente de acuerdo a su destino.

Nombre: sesion_dispositiva		
Descripción: Si se decide en el estudio que el expediente tiene que ir a Sesión Dispositiva entonces se almacenan los datos correspondientes a ésta.		
Atributo	**Tipo**	**Descripción**
id_sesion	serial	Número consecutivo ascendente que corresponde a cada una de las sesiones realizadas.
fecha_celebracion	date	Fecha de celebración de las Sesiones Dispositivas.
pertenece_efp	varchar(8)	Número del expediente al que pertenece esa Sesión Dispositiva.
pertenece_fiscalia	integer	Fiscalía a la que pertenece el expediente.
fecha_ent_efp	date	Fecha de entrada del Expediente al sistema.
fiscal_sesion	integer	Identificación del fiscal que participa en la Sesión Dispositiva.
id_juez	integer	Identificación del juez que participa en la Sesión Dispositiva.
decision_tribunal	integer	Decisión que toma el Tribunal de acuerdo a los asuntos tratados en la Sesión Dispositiva.
impug_decision	char(1)	Se emplea para saber si la decisión que se tomó por el Tribunal en la Sesión dispositiva, se impugna o no.

Nombre: impugnacion_decision		
Descripción: Si la Fiscalía decide impugnar la decisión de la Sesión dispositiva, se almacenan en esta tabla los datos correspondientes a esa impugnación.		
Atributo	**Tipo**	**Descripción**
sesion_dispositiva	integer	Identificador de la sesión dispositiva a la que pertenece la impugnación.
fecha_impugnacion	date	Fecha en que se impugnó la decisión de la sesión dispositiva.
fecha_salida	date	Fecha de salida del expediente para el tribunal superior que decidirá sobre la misma.
tribunal_superior	char(15)	Tribunal superior al que se remiten los documentos para su posterior análisis.
id_fiscal	integer	Fiscal que impugnó la decisión del estudio.

Nombre: causa		
Descripción: Se almacenan los datos de los Expedientes que forman causa, es decir, que ya han sido estudiados y se radicaron como causas.		
Atributo	**Tipo**	**Descripción**
no_causa	varchar(8)	Identificador de las causas, compuesto por un número de tres posiciones, un / y luego el número del año.
fecha_radicacion	date	Fecha en que fue radicada la causa.
cant_acusados	integer	Cantidad de acusados que tienen relación con una determinada causa.
no_efp	varchar(8)	Número del Expediente del que proviene la causa.
fecha_ent_efp	date	Fecha de entrada de ese Expediente al sistema.
cod_fiscalia	integer	Fiscalía a la que pertenece el Expediente.
juez_radicacion	integer	Juez que radicó la causa.
tipo_procedimiento	integer	Tipo de procedimiento que se va a llevar en la causa.
tipo_hecho	integer	Es el hecho que corresponde a la realización de los delitos.
id_institucion	integer	Institución a la que pertenece la causa.
fecha_con_acusatorias	date	Fecha en la que los acusados conocen las conclusiones acusatorias.
fecha_juicio_oral	date	Fecha en que se acuerda realizar el juicio oral.
cant_acusados	integer	Cantidad de acusados que trae la causa.
estado	char(1)	Estado en que se encuentra la causa.

Nombre: causa_acusado		
Descripción: Se almacenan los datos de los acusados correspondientes a la causa donde están involucrados.		
Atributo	**Tipo**	**Descripción**
no_causa	varchar(8)	Identificador de la causa con la que tiene relación el acusado.
carne_identidad	varchar(11)	Identificador del acusado.
condicion_delictiva	integer	La condición que tiene el acusado en el momento que se está procesando.
situacion_legal	char(20)	Situación legal que tiene el acusado en este momento.
lugar_pl	char(20)	Si está en prisión preventiva lugar donde está cumpliendo la misma.
fecha_pl	date	Fecha en que entró el acusado en prisión preventiva.
participacion	char(1)	Como fue la participación del acusado en los hechos (si fue autor de los hechos o cómplice).

Nombre: juicio		
Descripción: Se almacenan los datos correspondientes a los juicios.		
Atributo	**Tipo**	**Descripción**
id_juicio	varchar(8)	Número que representa a un determinado juicio.
causa	varchar(8)	Identificador de la causa de la que proviene el juicio.
fecha_celebracion	date	Fecha en que se da comienzo al juicio oral.
juez_precidejo	integer	Identificador del juez que participa en el juicio oral.

Nombre: juicio_suspendido		
Descripción: Se almacenan los datos correspondientes a los juicios suspendidos.		
Atributo	**Tipo**	**Descripción**
fecha_suspension	date	Fecha en que se suspende el juicio.
fecha_nuevo_juicio	date	Fecha determinada para que se reanude el juicio.
id_juicio	varchar(8)	Identificador del juicio al que corresponde la suspensión.
medida_suspension	integer	Motivo de la suspensión del juicio.
motivo_medida	integer	Medida impuesta por motivo de la sus pensión del juicio oral.

Nombre: juicio_celebrado		
Descripción: Se almacenan los datos correspondientes a los juicios celebrados.		
Atributo	**Tipo**	**Descripción**
id_juicio	varchar(8)	Identificador del juicio.
id_modalidad	integer	Identificador del tipo de juicio y el lugar en que se celebró.
fecha_fin	date	Fecha en que finalizó el juicio.
no_sesiones	integer	Número de sesiones que tuvo el juicio.

Nombre: juicio_resueltoov		
Descripción: Se almacenan los juicios que han sido resueltos por otras vías.		
Atributo	**Tipo**	**Descripción**
id_juicio	varchar(8)	Identificador del juicio.
fecha_resolucion	date	Fecha en que se resolvió el juicio.
motivo	char(20)	Motivo por el cual fue resuelto por otras vías.
destino	char(20)	Lugar destinado para la realización del juicio.
fecha_remision	date	Fecha en que se remite el juicio para que sea resuelto por otras vías.

2.7 Análisis de optimización de consultas.

Para analizar el tema de optimización de consultas en la BD propuesta se definieron determinados parámetros que son de alta importancia para que el diseño sea el más adecuado posible.

1. Diseño de las tablas.

- Todas las tablas de la Base de Datos están normalizadas hasta la tercera forma normal, lo que evita la duplicidad en los datos y permite que se aproveche al máximo el almacenamiento en las mismas.
- Los primeros campos de cada tabla de la Base de Datos son los más importantes, dentro de ellos están situados primero los de longitud fija y luego los de longitud variable. De esta manera a la hora de realizar una consulta utilizando estos atributos se optimizaría el tiempo de duración de la misma.
- El tamaño de los campos de la tabla está ajustado al máximo para evitar desperdiciar espacio.

2. Gestión y elección de los índices.

Los índices son campos elegidos arbitrariamente por el constructor de la base de datos para permitir la búsqueda a partir de dicho campo a una velocidad notablemente superior. Sin embargo, esta ventaja se ve contrarrestada por el hecho de ocupar mucha más memoria (el doble más o menos) y de requerir para su inserción y actualización un tiempo de proceso superior.

En las tablas de la Base de Datos del Sistema Informático de los Tribunales Militares de Región están indexados los campos que son llaves y los que contienen valores únicos, que son los índices que PostgreSQL admite por defecto. No es necesario crear más índices porque precisamente por medio de estos campos se van a realizar la mayoría de las consultas.

3. Realización de las consultas.

Las consultas de la aplicación no se van a realizar en el mismo gestor de datos, con motivo de que si ocurre un cambio de sistema no se tenga que implementar toda esta parte de nuevo para otro gestor. Se recomienda que a la hora de confeccionar las mismas se tengan en cuenta los siguientes pasos.

1. Campos a Seleccionar.

A la hora de realizar una consulta utilizando varias tablas se debe especificar a qué tabla pertenece cada campo, así se le ahorrará tiempo al gestor de localizar donde se encuentra ubicado cada atributo.

Ejemplo:

En lugar de SELECT no_efp, id_fiscalia, no_causa from dat_efp_acusado, dat_causa_acusado where id_acusado=id_encausado es conveniente usar: SELECT dat_efp_acusado.no_efp, dat_efp_acusado.id_fiscalia, dat_causa_acusado.no_causa from dat_efp_acusado, dat_causa_acusado where dat_efp_acusado.id_acusado= dat_causa_acusado.id_encausado.

2. Campos de Filtro.

Se debe procurar elegir en la cláusula WHERE aquellos campos que formen parte de la clave del fichero por el cual se interroga. Además se deben especificar en el mismo orden en el que estén definidos en la clave.

Se recomienda interrogar siempre por campos que sean clave.

3. Orden de las Tablas.

A la hora de utilizar varias tablas dentro de una misma consulta se debe tener cuidado con el orden empleado en la cláusula FROM.

Ejemplo:

Si se desea saber cuántos acusados de un determinado Expediente de Fase Preparatoria han pasado a ser encausados y se escribe: SELECT sum (dat_causa.cant_acusados) from dat_causa, dat_efp where dat_causa.no_efp=dat_efp.no_efp and dat_efp.no_efp=001/2007 and dat_efp.id_fiscalia=1 el gestor recorrerá todas las causas para sumar la cantidad de acusados y devolver el resultado. Si por el contrario se escribe SELECT sum(dat_causa.cant_acusados) from dat_efp, dat_causa where dat_efp.no_efp=001/2007 and dat_efp.id_fiscalia=1 and dat_efp.no_efp=dat_causa.no_efp el gestor filtra primero los expedientes, busca las causas y suma la cantidad de acusados, disminuyendo el tiempo de la consulta.

PostgreSQL cuenta con un comando denominado VACUUM, que limpia y analiza una Base de Datos. El VACUUM da la posibilidad de imprimir un reporte detallado de la actividad de análisis para cada tabla y de actualizar las estadísticas de columnas usadas por el optimizador para determinar la manera más eficiente de ejecutar una consulta. Las estadísticas representan la dispersión de los datos en cada columna.

La ejecución de VACUUM periódicamente aumenta la velocidad de la base de datos al procesar las consultas del usuario. La consulta VACUUM puede ser ejecutada en cualquier momento, particularmente, después de copiar una clase grande en PostgreSQL o después de borrar un gran número de registros. Esto actualizará los catálogos del sistema con todos los cambios recientes, y permitirá al organizador de consultas de PostgreSQL tomar las mejores decisiones al planear las consultas de los usuarios.

El administrador de la BD debe realizar un VACUUM periódicamente para actualizar los cambios y aumentar la velocidad de las consultas.

Otro comando que tiene PostgreSQL es el EXPLAIN. Este comando muestra el plan de ejecución que el planificador de PostgreSQL genera para la consulta dada. El plan de ejecución muestra la manera en que serán escaneadas las tablas referenciadas. La parte más crítica de la presentación es el costo estimado de ejecución de la consulta, que es la suposición del planificador sobre el tiempo que tomará correr la consulta (medido en unidades de captura de páginas de disco).

Esto es de mucha ayuda a la hora de construir una consulta, ya que analiza la manera más óptima de ejecutarla, brindando al usuario la posibilidad de escoger entre las de menos costo. Es de gran importancia para determinar cuáles serán las mejores formas de crear las consultas que se utilizarán para acceder a la información de la BD del Sistema Informático de los Tribunales Militares de Región, que por supuesto, serán las de menos costo.

2.8 Conclusiones.

El diseño propuesto de la Base de Datos del Sistema Informático de los Tribunales Militares de Región recoge cada uno de los requisitos funcionales planteados por los usuarios. Además, las tablas han sido construidas de un modo óptimo para eliminar el tiempo de duración de las consultas. Se ha reducido la redundancia en la información debido a que ya no existen datos repetidos innecesariamente, con lo cual también se aumenta la integridad de los mismos. Las tres fases empleadas en la normalización han sido utilizadas correctamente, evitando que se creen anomalías en las consultas realizadas para extraer e insertar datos en el sistema.

Capítulo 3
Validación del Diseño

3.1 Introducción.

El presente capítulo trata de la validación teórica y funcional del diseño de la BD realizado. En la primera se recogen aspectos como la integridad, seguridad y normalización de la BD, así como del análisis de la redundancia de la información y la trazabilidad de las acciones. La segunda se basa en las pruebas realizadas a la BD para comprobar su funcionamiento.

3.2 Validación teórica del diseño.

No basta solamente con realizar el diseño de una Base de Datos, se tienen que tener en cuenta también los aspectos que garanticen la consistencia, integridad y seguridad de la misma. Los Sistemas Gestores de Bases de Datos deben cumplir estos objetivos para facilitar la manipulación y confidencialidad de los datos.

3.2.1 Integridad.

La integridad de los datos se contempla en diferentes niveles. Las restricciones de dominio, transacciones y entidades definen las reglas para el mantenimiento de la integridad de las relaciones individuales. Las relaciones de integridad referencial aseguran que se mantienen las asociaciones necesarias entre las relaciones. Las restricciones de integridad de la Base de Datos gobiernan la BD como un todo y las restricciones de integridad de transacciones controlan la forma en que se manipulan los datos, dentro de una o entre múltiples Bases de Datos. (Riordan)

Integridad de Dominio.

Una restricción de integridad de dominio es una regla que define valores válidos para los atributos de las diferentes tablas de una BD. Puede ser necesario definir más de una restricción de dominio para describir por completo un dominio.

El primer paso es la elección de un tipo de datos lógico (un tipo de datos lógico puede ser una fecha, cadena, número). Si un valor de una cadena es de no más de 30 caracteres se debe usar un char (30). Por ejemplo, en el nomenclador de delitos la denominación no admite números, por lo que es de tipo caracter (char), y no sobrepasa las 40 letras. Su dominio sería char (40). De igual forma sucede con los restantes datos, las fechas, como la fecha de entrada en los expedientes de fase preparatoria, son de tipo date. Otro tipo de dato es varchar, donde se admiten números y letras. Los números de los expedientes, de las causas y de los juicios son varchar (8) porque aceptan los dos tipos de datos y su tamaño no sobrepasa la cantidad máxima entre 8 números y caracteres.

El siguiente aspecto a considerar sobre Integridad de dominio es si al dominio se le permite contemplar valores desconocidos o inexistentes, sabiendo que no es lo mismo desconocido, que inexistente. Esto se puede explicar de la siguiente forma; es posible que en la tabla dat_efp el campo fecha_hechos sea desconocido, pero si efectivamente se han realizado los hechos, tiene que haber una fecha en que ocurrieron los mismos y por tanto no puede ser inexistente, es decir, no puede ser nulo. Otro ejemplo es el de la situación legal del acusado en la tabla efp_acusado. Si el acusado se encuentra en privación de libertad se debe introducir el lugar donde cumple su sentencia, pero si no está en esta condición el lugar sería inexistente y el campo admitiría valores nulos.

El último aspecto de integridad de dominio es que se deberá definir el conjunto de los valores representados por un dominio lo más específicamente que se pueda. En el nomenclador de los jueces se encuentra un campo denominado estado, donde se especifica si un juez está ejerciendo o no. El dominio para este campo admite solamente el conjunto de valores siguientes: A, para especificar que el juez está activo e I para cuando esté inactivo, evitando que el usuario introduzca un dato no adecuado que afecte la integridad del sistema.

Integridad de Transacciones.

Define los estados por los que una tupla puede pasar válidamente, como son introducido, pendiente, seleccionado, enviado, cancelado y terminado. Está encargada de asegurar que el estado de una determinada tupla, no pase de un estado inicial a uno final, sin haber pasado por los estados intermedios. En el caso de la BD del Sistema Informático de los Tribunales Militares de Región no se encuentran restricciones de este tipo.

Integridad de Entidades.

Las restricciones de entidades aseguran la integridad de las entidades que son modeladas por el sistema.

Un caso de este tipo de restricción es el de las llaves primarias para cada tabla. En el caso de la BD que se está analizando cada entidad tiene definida su llave primaria, que no puede ser nula ni se puede repetir. Otra forma de esta integridad es la definida en el expediente de fase preparatoria, donde se chequea que la fecha de los hechos sea menor o igual que la de denuncia, al igual que ésta sea menor que la fecha de entrada del expediente al sistema.

Integridad Referencial.

Otra de las reglas, es la llamada restricción de integridad referencial que se especifica entre dos relaciones y sirve para mantener la consistencia entre tuplas de las dos relaciones. Establece que una tupla en una relación que haga referencia a otra, deberá referirse a un valor existente en esa relación. Un ejemplo de aplicación de este concepto en el modelo de datos es el siguiente: se asegura que los atributos no_causa y no_acusado de la tabla causa_acusado sean llaves foráneas, debido a que provienen y son llaves primarias de las entidades causa y acusado respectivamente. Además cumplen con las restricciones siguientes:

1. Los números de la causa y del acusado en la tabla causa_acusado tienen los mismos dominios que en sus entidades de origen.

2. Al introducir los números del acusado y de la causa en la tabla causa_acusado, éstos tienen que encontrarse en su entidad de origen, es decir, causa_acusado no puede tener un número de causa y de acusado que no estén presentes en la tabla causa y acusado respectivamente.

Por lo planteado anteriormente se comprueba que en el modelo de datos se cumple con la regla de integridad referencial, que es una de las más importantes dentro de la integridad de la base de datos.

Integridad entre varias entidades de una Base de Datos.

Otra forma de integridad, que fue chequeada en la BD diseñada, es la que relaciona a varias entidades de la Base de Datos. Por ejemplo, existe un campo en la tabla expediente donde se almacena

la cantidad de acusados que se vieron implicados en un hecho determinado. Luego, en la tabla que registra los datos de los acusados en un determinado expediente, se inserta un número de acusado por cada EFP. Para prevenir que el usuario, por un error o por maldad, introdujera un número mayor que el definido en la cantidad de acusados, se implementó un disparador (trigger) que validara que el número del acusado estuviera entre 1 y la cantidad de acusados que existe en el expediente.

Tomando en cuenta cada una de las restricciones definidas en este epígrafe, se chequeó la integridad de la Base de Datos del Sistema Informático de los Tribunales Militares de Región, dependiendo de las características brindadas por el Sistema Gestor de Bases de Datos: PostgreSQL.

Otro medio de mantener la integridad es el control de concurrencia en la Base de Datos. A diferencia de la mayoría de otros sistemas de Bases de Datos que usan bloqueos para el control de concurrencia, PostgreSQL mantiene la consistencia de los datos en un modelo multiversión. Esto significa que mientras se consulta una base de datos, cada transacción ve una imagen de los datos (una versión de la base de datos) como si fuera tiempo atrás, sin tener en cuenta el estado actual de los datos que hay por debajo. Esto evita que la transacción vea datos inconsistentes que pueden ser causados por la actualización de otra transacción concurrente en la misma fila de datos, proporcionando aislamiento transaccional para cada sesión de la base de datos.

Existen cuatro niveles de aislamiento transaccional en función de tres hechos que deben ser tenidos en cuenta entre transacciones concurrentes. Estos hechos no deseados son:

- Lecturas sucias.

Una transacción lee datos escritos por una transacción no esperada, no cursada.

- Lecturas no repetibles.

Una transacción vuelve a leer datos que previamente había leído y encuentra que han sido modificados por una transacción cursada.

- Lecturas fantasmas.

Una transacción vuelve a ejecutar una consulta, devolviendo un conjunto de filas que satisfacen una condición de búsqueda y encuentra que otras filas que satisfacen la condición han sido insertadas por otra transacción cursada.

Los cuatro niveles de aislamiento transaccional se describen en la tabla en la siguiente.

	Lectura "sucia"	Lectura no repetible	Lectura "fantasma"
Lectura no cursada	Posible	Posible	Posible
Lectura cursada	No posible	Posible	Posible
Lectura repetible	No posible	No posible	Posible
Serializable	No posible	No posible	No posible

PostgreSQL ofrece lectura cursada y niveles de aislamiento serializables.

La Lectura cursada es el nivel de aislamiento por defecto en PostgreSQL. Cuando una transacción se ejecuta en este nivel, la consulta sólo ve datos cursados antes de que se ejecutara y nunca ve ni datos "sucios" ni los cambios en transacciones concurrentes cursados durante la ejecución de la consulta.

La serialización proporciona el nivel más alto de aislamiento transaccional. Cuando una transacción está en el nivel serializable, la consulta sólo ve los datos cursados antes de que la transacción comience y nunca ve ni datos sucios ni los cambios de transacciones concurrentes cursados durante la ejecución de la transacción. Por lo tanto, este nivel emula la ejecución de transacciones en serie, como si las transacciones fueran ejecutadas una detrás de otra, en serie, en lugar de concurrentemente.

Para controlar la concurrencia en la Base de Datos del Sistema Informático de los Tribunales Militares de Región se empleará el modo serializable, para evitar que cuando haya más de un usuario trabajando concurrentemente en la BD se produzcan hechos no deseados entre las transacciones, que puedan afectar la integridad de los datos en el sistema.

3.2.2 Normalización de la Base de Datos.

La teoría de la normalización se ha desarrollado para obtener estructuras de datos eficientes que eviten las anomalías de actualización. Es la expresión formal del modo de realizar un buen diseño y provee los medios necesarios para describir la estructura lógica de los datos en un sistema de

información. Evita anomalías en la actualización y mejora la independencia de los datos, permitiendo realizar extensiones de la BD, afectando muy poco, o nada, a los programas de aplicación existentes que accesan la base de datos.

Para normalizar una BD se deben realizar varias fases en orden. (García, 2005)

- Primera Fase Normal (1FN).
- Segunda Fase Normal (2FN).
- Tercera Fase Normal (3FN).

También existen la Forma Normal de Boyce-Codd (FNBC), la cuarta y la quinta formas normales.

La BD del Sistema Informático de los Tribunales Militares de Región ha sido normalizada hasta la tercera forma normal, debido a que de esta manera se resuelven los problemas de inconsistencia y redundancia en sus tablas.

- Primera Forma Normal:

Una entidad está en 1FN si cumple la propiedad de que sus dominios no tienen elementos que, a su vez, sean conjuntos.

La BD propuesta cumple en su totalidad con esta restricción. Ninguno de los atributos de sus tablas toma en algún momento un conjunto de valores.

- Segunda Forma Normal:

Se dice que una entidad está en 2FN si está en 1FN y si sus atributos no llaves (ni primarias ni candidatas) son funcional y completamente dependientes de la llave primaria. Lo relacionado con lo de la dependencia funcional completa se aplica solo a entidades con llaves compuestas.

La BD que se está analizando cumple con la 2FN. Todas sus tablas se encuentran en 1FN y las que tienen llaves compuestas cumplen con la dependencia funcional completa. Ejemplo:

Todos los atributos de los expedientes de fase preparatoria dependen de su llave primaria (no_efp, id_fiscalia, fecha _ent) y no lo hacen de alguno de sus atributos.

no_efp, id_fiscalia, fecha_ent → no_registro, fecha_denun, fecha_hechos, cant_acusados

- Tercera Forma Normal:

Una entidad está en 3FN si está en 2FN y si todos sus atributos no llaves son independientes de cualquier otro atributo no llave primaria.

En la BD del SITMR los atributos de cada tabla dependen exclusivamente de sus respectivas llaves primarias.

3.2.3 Análisis de redundancia de información.

Con el diseño de la nueva BD para el SITMR se ha disminuido considerablemente la redundancia en los datos que existía en el sistema anterior. Un punto muy importante para lograr lo anterior fue la normalización de la BD, posibilitando que en cada tabla estén presentes los datos que se necesiten realmente. A parte de lo anterior se disminuyeron los datos repetidos innecesariamente en las mismas.

En la tabla efp_acusado existen datos que se repiten en causa_acusado, ellos son los correspondientes a la situación legal del acusado, y en caso de que se encuentre en privación de libertad el lugar donde esté cumpliendo sentencia y la fecha de inicio de la misma. Esta información es necesaria tenerla en ambos lados debido a que la situación podría variar desde el momento que se crea el expediente hasta que se radique la causa, por lo que se justifica la redundancia. En otras tablas no se encontrarán datos duplicados.

3.2.4 Análisis de la seguridad de la Base de Datos.

La seguridad garantiza el acceso autorizado a los datos, para irrumpir cualquier intento de acceso no autorizado, ya sea por error del usuario o por mala intención.

Para proteger la BD del SITMR se decidió crear un usuario como administrador que va a ser el único que tendrá todos los privilegios para acceder al SGBD y darle mantenimiento a la BD. Desde el sitio podrán insertar y modificar información solamente las secretarias, los otros usuarios solo podrán realizar consultas.

La información debe ser recuperada en caso de que ocurra algún error en el servidor de la Base de Datos o alguna falla de energía. Deben realizarse copias de seguridad de las Bases de Datos regularmente. Dado que PostgreSQL gestiona sus propios ficheros en el sistema, no se recomienda confiar en los sistemas de copia de seguridad del sistema para las copias de respaldo de las BD; no hay

garantía de que los ficheros estén en un estado consistente que permita su uso después de la restauración.

Para la realización de copias a la BD PostgreSQL cuenta con un comando denominado pg_dump que se encarga de hacerle un backup a la BD en el momento que el administrador decida, luego la información se recupera por medio de otro comando: pg_restore. Además, este SGBD se puede ejecutar en dos modos: fsync y no fsync. Si se ejecuta en el primer modo el gestor sería más lento, pero garantizaría que si el sistema operativo se bloquea o se produce una pérdida de energía, todos sus datos estarían almacenados y sin daños en el disco. El modo de ejecución de PostgreSQL en el servidor de BD de la aplicación será fsync, para evitar pérdida de la información en caso de que ocurra algún error inevitable en el sistema.

Las contraseñas de los usuarios, tanto los de la aplicación como el administrador de la BD estarán encriptadas, para evitar que intrusos se apoderen de ellas con malos fines.

3.2.5 Trazabilidad de la acciones.

La auditoría es uno de los pasos fundamentales para asegurar una Base de Datos. De ella dependen que se registren los sucesos que ocurren en el sistema, la fecha, la hora, quién los realizó y otros datos de interés.

Esto resulta muy importante porque se pueden detectar personas que intenten ejecutar acciones en la Base de Datos, que no les estén permitidas, y detenerlas antes de que corrompan los datos o realicen alguna operación en contra de la integridad de la BD.

Para garantizar la trazabilidad de las acciones en la BD del Sistema Informático de los Tribunales Militares de Región se creó una tabla denominada auditoria, donde se almacenan las operaciones realizadas sobre las demás tablas, es decir, se almacena el nombre de la tabla en la que ocurrió algún cambio, la fecha, la hora, el usuario y el evento (si insertó, modificó o eliminó). De esta forma se recogen los sucesos más importantes que ocurren en la BD, para en caso de algún cambio no deseado, saber a quién acudir para que explique la razón por la que ejecutó la acción.

3.3 Validación funcional.

Para garantizar que la Base de Datos cumpla con los requisitos de los usuarios y opere sin problemas se necesitan buenos procedimientos de pruebas. Las pruebas de validación normalmente se centran en las operaciones siguientes:

- La carga de la Base de Datos se realiza sin violar la integridad de los datos.
- La correcta interfaz de las aplicaciones con la Base de Datos.
- El rendimiento del sistema satisface las necesidades para las que se adquirió el SGBD.

El diseño de la Base de Datos del Sistema Informático de los Tribunales Militares de Región incluye en su estructura tablas que contienen información que no varían frecuentemente, y que se encuentran en formato DBF. Para la carga de estos ficheros se utilizó la funcionalidad para importación de datos que tiene PostgreSQL por defecto, lo que permite disminuir el tiempo de llenado de las tablas y evitar equivocaciones en el momento de insertar los datos existentes en la aplicación anterior.

Para la realización de pruebas de un llenado voluminoso e inteligente de la Base de Datos, sin violar la integridad de la misma, se empleó PostgreSQL Data Generator. Esta herramienta es muy poderosa en la generación de datos de pruebas para Bases de Datos construidas en PostgreSQL. Permite seleccionar tablas y columnas para generar datos, definir rangos de valores, generar columnas de caracteres por máscara. También brinda la posibilidad de generar los datos a través de consultas SQL o de escogerlos por medio de listas de valores. Una de las ventajas que tiene es que genera los datos que provienen de otras tablas para evitar errores de integridad referencial. El empleo de esta herramienta posibilitó chequear la integridad de la BD, obteniéndose muy buenos resultados.

Las tablas fueron llenadas con un rango de datos equivalente al que tendrá la Base de Datos en un año aproximadamente.

Otra forma de probar que la Base de Datos funciona correctamente es a través de consultas. De acuerdo a las operaciones que se espera se realicen una vez implantada la BD se ejecutaron, entre otras, las siguientes consultas:

- Obtener el nombre, código de las fiscalías y cantidad de causas de las mismas, que se radicaron en el mes de abril (4), agrupadas por el código y nombre y ordenadas de acuerdo a los nombres de las fiscalías.

CREATE VIEW "public"."reporte_04" (codigo, cod_fiscalia, causas_radicadas) AS SELECT DISTINCT nf.denom_fiscalia AS codigo, dc.cod_fiscalia, count (dc.cod_fiscalia) AS causas_radicadas FROM (nom_fiscalia nf JOIN dat_causa dc ON ((nf.cod_fiscalia = dc.cod_fiscalia))) WHERE (date_part('MONTH'::text, dc.fecha_radicacion) = (4)::double precision) GROUP BY nf.denom_fiscalia, dc.cod_fiscalia ORDER BY nf.denom_fiscalia, dc.cod_fiscalia, count(dc.cod_fiscalia);

El resultado obtenido a partir de esta consulta fue de dos filas en un tiempo de 0,03 segundos.

- Obtener los nombres de los acusados relacionados con los expedientes de fase preparatoria, los números de carné de identidad, apellidos, tipo de efectivo e identificadores de los efp.

SELECT ac.nombres, ac.carne_identidad, ac.apellido1, ac.apellido2, ec.no_efp, ec.cod_fiscalia, ec.fecha_ent, te.denom_efectivo as efectivo, ec.situacion_legal from dat_acusado ac join dat_efp_acusado ec on (ac.carne_identidad=ec.carne_identidad) join nom_tipo_efectivo te on (ec.id_efectivo=te.id_efectivo) group by ac.carne_identidad, ac.nombres, ac.apellido1, ac.apellido2, ec.no_efp, ec.cod_fiscalia, ec.fecha_ent, te.denom_efectivo, ec.situacion_legal order by ac.nombres;
Se obtuvo un resultado de 42 filas en un tiempo de 0,06 segundos.

- Obtener los juicios celebrados, la causa y el expediente del que provienen.

CREATE OR REPLACE VIEW "public"."reporte_07" (juicios_realizados, no_causa, no_efp, cod_fiscalia, fecha_entrada) AS SELECT jr.id_juicio AS juicios_realizados, j.no_causa, c.no_efp, c.cod_fiscalia, c.fecha_ent AS fecha_entrada FROM ((dat_juicio_celebrado jr JOIN dat_juicio j ON (((jr.id_juicio)::text = (j.id_juicio)::text))) JOIN dat_causa c ON (((j.no_causa)::text = (c.no_causa)::text)));
El resultado obtenido fue de un máximo de 15 filas en 0,03 segundos.

Las pruebas realizadas a una BD nunca pueden tomarse como resultados reales, aunque dan un aproximado al mismo, dependiendo del grado de acercamiento que se utilice a los procesos de la vida real. Aún así, la implantación y utilización de una BD, está marcada por factores como, cantidad de usuarios a conectarse y el grado de concurrencia de las solicitudes hechas por los mismos, los cuales hay que tener en cuenta siempre. Además, es necesario velar el cumplimiento de todos los requerimientos no funcionales propuestos para la BD, ya que validan el funcionamiento correcto y máximo rendimiento de la misma. Aunque las pruebas realizadas brinden resultados que puedan parecer alentadores, no pueden

ser motivo de confianza, todo lo contrario, 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 buscando mejorar o resolver, posibles errores que se detecten durante su vida útil.

3.4 Valoración de resultados.

Luego de concluir el diseño y la implementación de la Base de Datos del Sistema Informático de los Tribunales Militares de Región se obtuvieron los resultados siguientes:

- La BD cumple con los requisitos de los usuarios.
- Las relaciones entre las tablas fueron construidas adecuadamente.
- Correcta normalización de la BD.
- Las restricciones de integridad garantizan que los datos introducidos o modificados sean los correctos.
- El chequeo de las distintas operaciones realizadas sobre las tablas de la BD proporcionan un nivel de seguridad para el conocimiento de cambios no deseados en las mismas.
- La redundancia en la información se ha reducido en gran medida.

3.5 Conclusiones.

Con la validación teórica del diseño realizado se chequearon parámetros muy importantes a tener en cuenta en el correcto funcionamiento de la Base de Datos. Se definieron restricciones de integridad para garantizar la validez de los datos introducidos o modificados en la misma. Fueron tomados en cuenta aspectos para garantizar la seguridad de los datos en el sistema y llevar a cabo la trazabilidad de las acciones en la BD. Por otra parte, las pruebas realizadas a la BD permitieron chequear la integridad y seguridad de los datos y el correcto funcionamiento de la misma.

CONCLUSIONES

Las Bases de Datos se han venido utilizando desde hace ya varios años. Su marcada importancia hace que muchas empresas las incorporen en sus organizaciones para el procesamiento de la información. Los Sistemas Administradores de Bases de Datos, que facilitan la manipulación y actualización de los datos de una aplicación, han aumentado su potencialidad con el transcurso del tiempo, siendo hoy en día los Sistemas de Bases de Datos Relacionales los más utilizados. Otros gestores de BD que también se van desarrollando son los Orientados a Objetos, los Distribuidos, los Multidimensionales y los Objetos Relacionales, los últimos emplean la estructura relacional pero incorporan conceptos de la orientación a objetos.

Un ejemplo de Sistema Gestor de Base de Datos Objeto Relacional es PostgreSQL, en el cual reside la Base de Datos confeccionada para el Sistema Informático de los Tribunales Militares de Región. Este sistema ha posibilitado garantizar la integridad de la BD por medio de restricciones y disparadores que chequean la validez de los datos y definir aspectos de seguridad de la información.

Se llegó a la conclusión de que la nueva BD del Sistema Informático de los Tribunales Militares de región cumple con el proceso de normalización hasta la tercera forma normal, posibilitando que en las tablas se encuentren datos que realmente pertenecen a ellas y que se disminuya la redundancia en la información. Se han tomado en cuenta, además, aspectos que permitan la reducción del tiempo de ejecución de las consultas de acuerdo a las necesidades de los usuarios.

Las pruebas realizadas a la BD han demostrado que cumple con los requisitos de los clientes y que la misma está acta para su funcionamiento en los Tribunales Militares de Región, de acuerdo a su grado de integridad y seguridad.

REFERENCIAS BIBLIOGRÁFICAS

mailxmail.com. [En línea] MAILXMAIL, S.L. [Citado el: 5 de marzo de 2007.] http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm.

C.J.Date. 2003. Introducción a los Sistemas de Base de Datos. Primera Parte. La Habana: s.n., 2003.

—. 2003. Introducción a los Sistemas de Base de Datos. Tercera Parte. La Habana: s.n., 2003.

Cutiño, Alieski Sarmiento y Elian. 2006. LIMS DE CALIDAD DEL CENTRO DE INGENIERÍA GENÉTICA Y BIOTECNOLOGÍA: ANÁLISIS DEL GRUPO. La Habana: s.n., 2006.

1999-2007. EMS SQL Manager for PostgreSQL. [En línea] 1999-2007. [Citado el: 10 de marzo de 2007.] http://www.sqlmanager.net/products/postgresql/manager.

Garcia, Rosa M. Matos. 2005. Sistema de Base de Datos. 2005.

Gascón, Manuel de la Herrán. 1999-2004. Administración y Optimización de Bases de Datos Oracle. [En línea] 1999-2004. [Citado el: 20 de abril de 2007.] http://www.redcientifica.com/oracle/c0001p0005.html.

Hansen, James W. Diseño y Administracion de base de datos.

1997-2007. Potente utilidad de modelado para varias bases de datos. [En línea] 1997-2007. [Citado el: 15 de abril de 2007.] http://case-studio.softonic.com/.

Riordan, Rebeca M. Diseño de bases de datos relacionales con Acces y SqlServer.

BIBLIOGRAFÍA

mailxmail.com. [En línea] MAILXMAIL, S.L. [Citado el: 5 de marzo de 2007.] http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm.

A., Ernesto Quiñones. Introducción a Postgresql. s.l. : Asociación Peruana de Software Libre (APESOL).

2006. Bases de datos_mysql. [En línea] Intitucionjmr, 2006. [Citado el: 20 de febrero de 2007.] http://blog.iespana.es/institucionjmr.

Build Quality Applications Faster, Better and Cheaper. [En línea] [Citado el: 20 de enero de 2007.] http://www.visual-paradigm.com/product/vpuml/.

C.J.Date. 2003. Introducción a los Sistemas de Base de Datos. Primera Parte. La Habana: s.n., 2003.

—. 2003. Introduccion a los Sistemas de Base de Datos. Segunda Parte. La Habana: s.n., 2003.

—. 2003. Introducción a los Sistemas de Base de Datos. Tercera Parte. La Habana: s.n., 2003.

Cutiño, Alieski Sarmiento y Elian. 2006. LIMS DE CALIDAD DEL CENTRO DE INGENIERÍA GENÉTICA Y BIOTECNOLOGÍA: ANÁLISIS DEL GRUPO. La Habana: s.n., 2006.

1999-2007. EMS SQL Manager for PostgreSQL. [En línea] 1999-2007. [Citado el: 10 de marzo de 2007.] http://www.sqlmanager.net/products/postgresql/manager.

Garcia, Rosa M. Matos. 2005. Sistema de Base de Datos. 2005.

Gascón, M.Herrán. 1999-2004. Administración y Optimización de bases de datos Oracle. 1999-2004.

Gascón, Manuel de la Herrán. 1999-2004. Administración y Optimización de Bases de Datos Oracle. [En línea] 1999-2004. [Citado el: 20 de abril de 2007.] http://www.redcientifica.com/oracle/c0001p0005.html.

Hansen, James W. Diseño y Administracion de base de datos.

J. Ponce, G. Solis y L.Ulfe. Data Mining.

1997-2007. Potente utilidad de modelado para varias bases de datos. [En línea] 1997-2007. [Citado el: 15 de abril de 2007.] http://case-studio.softonic.com/.

Programación en castellano. 1999-2006. Java en Castellano. [En línea] 1999-2006. [Citado el: 25 de febrero de 2007.] http://www.programacion.net/java/noticia/1018/.

Riordan, Rebeca M. Diseño de bases de datos relacionales con Acces y SqlServer. .

ANEXOS

Anexo 1. Descripción de las clases del diagrama de clases persistentes.

Nombre: nom_fiscalia	
Tipo de clase: entidad	
Atributo	**Tipo**
id_fiscalia	int
denom_fiscalia	char

Nombre: nom_delitos	
Tipo de clase: entidad	
Atributo	**Tipo**
id_delito	varchar
denom_del	char

Nombre: nom_institucion	
Tipo de clase: entidad	
Atributo	**Tipo**
id_institucion	int
denom_institucion	char

Nombre: nom_unidad_militar	
Tipo de clase: entidad	
Atributo	**Tipo**
id_unidad	varchar
sub1	varchar
sub2	varchar
sub3	varchar
sub4	varchar
denom_unidad	char
mando	varchar

Nombre: nom_cargo	
Tipo de clase: entidad	
Atributo	**Tipo**
id_cargo	int
denom_cargo	char
abrev	char

Nombre: nom_especialidad	
Tipo de clase: entidad	
Atributo	**Tipo**
id_especialidad	int
denom_espec	char

Nombre: nom_prov	
Tipo de clase: entidad	
Atributo	**Tipo**
id_prov	varchar
denom_prov	char

Nombre: nom_mun	
Tipo de clase: entidad	
Atributo	**Tipo**
id_mun	varchar
denom_mun	char

Nombre: nom_jueces	
Tipo de clase: entidad	
Atributo	**Tipo**
id_juez	int
denom_juez	char

Nombre: nom_grado_militar	
Tipo de clase: entidad	
Atributo	**Tipo**
id_grado	int
denom_grado	char

Nombre: nom_dec_tribunal	
Tipo de clase: entidad	
Atributo	**Tipo**
id_decision	int
denom_dec	char

Nombre: nom_nivel_esc	
Tipo de clase: entidad	
Atributo	**Tipo**
id_nivel	int
denom_nivel	char

Nombre: nom_centro_trabajo	
Tipo de clase: entidad	
Atributo	**Tipo**
id_centro	int
denom_centro	char

Nombre: nom_dec_es	
Tipo de clase: entidad	
Atributo	**Tipo**
id_decision	int
denom_dec	char

Nombre: nom_fiscales	
Tipo de clase: entidad	
Atributo	**Tipo**
id_fiscal	datetime
denom_fiscal	char
cargo	char

Nombre: nom_med_sjo	
Tipo de clase: entidad	
Atributo	**Tipo**
id_medida	int
denom_medida	char

Nombre: nom_modalidadjo	
Tipo de clase: entidad	
Atributo	**Tipo**
id_modalidad	int
denom_modalidad	char

Nombre: nom_mot_suspen	
Tipo de clase: entidad	
Atributo	**Tipo**
id_suspencion	int
denom_suspencion	char

Nombre: nom_profesion	
Tipo de clase: entidad	
Atributo	**Tipo**
id_profesion	int
denom_profesion	char

Nombre: nom_cond_delictiva	
Tipo de clase: entidad	
Atributo	**Tipo**
id_condicion	int
denom_condicion	char

Nombre: nom_tipo_efectivo	
Tipo de clase: entidad	
Atributo	**Tipo**
id_efectivo	int
denom_efectivo	char

Nombre: nom_tipo_hecho	
Tipo de clase: entidad	
Atributo	**Tipo**
id_hecho	int
denom_hecho	char

Nombre: nom_tipo_proc	
Tipo de clase: entidad	
Atributo	**Tipo**
id_procedimiento	int
denom_procedimiento	char

Nombre: nom_cat_ocupacional	
Tipo de clase: entidad	
Atributo	**Tipo**
id_categoria	int
denom_cat	char

Anexo 2. Descripción de las tablas del diagrama entidad relación.

Nombre: nom_fiscalia		
Descripción: Se almacenan los nombres de las fiscalías que existen en el país.		
Atributo	**Tipo**	**Descripción**
cod_fiscalia	integer	Es un número que representa a cada fiscalía militar.
denom_fiscalia	char(20)	Representa los nombres de las fiscalías pertenecientes a un determinado tribunal.

Nombre: nom_delitos		
Descripción: Se almacenan los delitos que existen y que puede cometer una persona.		
Atributo	**Tipo**	**Descripción**
cod_delito	varchar(6)	Código que identifica a cada uno de los delitos.
denom_delito	char(30)	Denominación de cada uno de los delitos.

Nombre: nom_institucion		
Descripción: Nombre de las instituciones militares que existen el país (Ejemplo: FAR, MININT, EJT).		
Atributo	**Tipo**	**Descripción**
id_institucion	serial	Número consecutivo ascendente que representa a cada una de las instituciones existentes.
denom_institucion	char(20)	Denominación de cada una de las instituciones.

Nombre: nom_unidad_militar		
Descripción: Se almacenan los datos de las unidades militares pertenecientes a las diferentes instituciones que existen.		
Atributo	**Tipo**	**Descripción**
id_unidad	varchar(4)	Número que representa la unidad militar y la institución a la que corresponde.
sub1	varchar(4)	Batallón al que pertenece.
sub2	varchar(4)	Brigada a la que pertenece.
sub3	varchar(4)	Sector militar al que pertenece.
sub4	varchar(4)	Región militar a la que pertenece.
denom_unidad	char(60)	Denominación de cada una de las Unidades Militares.
mando	varchar(4)	Mando al que pertenece.
id_institucion	integer	Identificador de la institución a la que corresponde la Unidad Militar.

Nombre: nom_cargo		
Descripción: Se almacenan todos los cargos que existen.		
Atributo	**Tipo**	**Descripción**
id_cargo	serial	Número consecutivo ascendente que representa a cada cargo existente.
denom_cargo	char(30)	Definición de cada uno de los cargos.
abrev	char(12)	Es una abreviatura de la definición del cargo.

Nombre: nom_especialidad		
Descripción: Se almacenan las posibles especialidades que pueden poseer los acusados.		
Atributo	**Tipo**	**Descripción**
id_especialidad	serial	Número consecutivo ascendente que representa a cada una de las especialidades que existen.
denom_especialidad	char (30)	Especialidades existentes.

Nombre: nom_provincia		
Descripción: Se almacenan los nombres de las distintas provincias que existen en el país, con sus respectivos códigos.		
Atributo	**Tipo**	**Descripción**
cod_provincia	varchar(2)	Código que representa a cada provincia.
denom_provincia	char(20)	Nombre de cada una de las provincias.

Nombre: nom_municipio		
Descripción: Se almacenan los nombres de todos los municipios existentes en el país, y la provincia a que pertenecen.		
Atributo	**Tipo**	**Descripción**
cod_municipio	varchar(2)	Número que representa a cada municipio que existe, dentro de la provincia.
denom_municipio	char(20)	Nombre de cada uno de los municipios existentes.
cod_provincia	varchar(2)	Identificador de la provincia a la que corresponde ese municipio.

Nombre: nom_jueces		
Descripción: Se almacenan datos sobre los jueces.		
Atributo	**Tipo**	**Descripción**
id_juez	serial	Número consecutivo ascendente que representa a cada juez.
denom_juez	char(30)	Representa el nombre de los jueces.
estado	char(1)	Se utiliza para determinar si un juez está activo o ya no sigue ejerciendo.
grado_militar	integer	Identificador del grado que tiene el juez.

Nombre: nom_grado		
Descripción: Se almacenan las denominaciones de todos los grados que existen.		
Atributo	**Tipo**	**Descripción**
id_grado	serial	Número consecutivo ascendente correspondiente a un determinado grado.
denom_grado	char(20)	Denominaciones de cada uno de los grados.

Nombre: nom_decision_tribunal		
Descripción: Se almacenan las posibles decisiones que puede tomar un tribunal durante una Sesión Dispositiva.		
Atributo	**Tipo**	**Descripción**
id_decision_tribunal	serial	Número consecutivo ascendente que representa a cada decisión.
denom_decision	char(40)	Decisiones que pueden ser tomadas durante una Sesión Dispositiva.

Nombre: nom_nivel_escolar		
Descripción: Se almacenan las denominaciones de los niveles escolares existentes.		
Atributo	**Tipo**	**Descripción**
id_nivel	serial	Número consecutivo ascendente que representa a cada nivel escolar.
denom_nivel	char(20)	Niveles escolares existentes.

Nombre: nom_centro_trabajo		
Descripción: Se almacenan los posibles centros de trabajo que pueden ocupar los acusados.		
Atributo	**Tipo**	**Descripción**
id_centro	serial	Número consecutivo y ascendente que representa a un determinado centro de trabajo.
denom_centro	char(30)	Denominaciones de los centros de trabajo existentes.

Nombre: nom_decision_estudio		
Descripción: Se almacenan las posibles decisiones que se pueden tomar al estudiar un Expediente de Fase Preparatoria.		
Atributo	**Tipo**	**Descripción**
id_decision	serial	Número consecutivo ascendente que representa a cada una de las posibles decisiones.
denom_decision	Char(30)	Posibles decisiones que se pueden tomar.

Nombre: nom_fiscales		
Descripción: Representa los datos de los fiscales.		
Atributo	**Tipo**	**Descripción**
id_fiscal	serial	Número consecutivo ascendente para identificar a cada uno de los fiscales.
denom_fiscal	char(30)	Nombre de los fiscales.
estado	char(1)	Estado en que se encuentra el fiscal: activo o inactivo.
grado_militar	integer	Grado del fiscal.
fiscalia	integer	Fiscalía a la que pertenece cada fiscal.
cargo	cargo (25)	Cargo del fiscal.

Nombre: nom_medida_suspensionjo		
Descripción: Se almacenan las medidas tomadas cuando se suspenden los juicios orales.		
Atributo	**Tipo**	**Descripción**
id_medida	int	Número consecutivo ascendente que representa las medidas tomadas por la suspensión de los juicios orales.
denom_medida	char(25)	Posibles medidas a tomar por la suspensión de un juicio oral.

Nombre: nom_modalidadjo		
Descripción: Se almacenan las posibles modalidades de juicios que existen.		
Atributo	**Tipo**	**Descripción**
id_modalidad	serial	Número consecutivo ascendente que representa a cada una de las modalidades.
denom_modalidad	char(25)	Denominación de las distintas modalidades de juicio que existen.

Nombre: nom_motivo_suspension		
Descripción: Se almacenan los posibles motivos de suspensión de los juicios orales.		
Atributo	**Tipo**	**Descripción**
id_suspension	serial	Número consecutivo ascendente que representa los motivos por los cuales se puede suspender un juicio oral.
denom_suspension	char(35)	Motivos de suspensión de los juicios orales.

Nombre: nom_profesion		
Descripción: Se almacenan las posibles profesiones que puede poseer un acusado.		
Atributo	**Tipo**	**Descripción**
id_profesion	serial	Número consecutivo ascendente que representa a cada una de las profesiones.
denom_profesion	char(20)	Profesiones existentes.

Nombre: nom_condicion_delictiva		
Descripción: Se almacenan las posibles condiciones delictivas que pueden tener los acusados. Por ejemplo, son repitentes en los delitos cometidos.		
Atributo	**Tipo**	**Descripción**
id_condicion	serial	Número consecutivo ascendente que representa a cada una de las condiciones.
denom_condicion	char(20)	Distintas condiciones existentes.

Nombre: nom_tipo_efectivo		
Descripción: Se almacenan los diferentes tipos de efectivos que existen (civil, militar, civil de la defensa).		
Atributo	**Tipo**	**Descripción**
id_efectivo	serial	Número consecutivo y ascendente que representa a los tipos de efectivos que existen.
denom_efectivo	char(15)	Denominaciones de los efectivos que existen.

Nombre: nom_tipo_hecho		
Descripción: Se almacenan los hechos que tienen relación con los delitos.		
Atributo	**Tipo**	**Descripción**
id_hecho	serial	Número consecutivo ascendente que identifica a cada hecho.
denom_hecho	char(50)	Denominación de cada hecho.

Nombre: nom_tipo_procedimiento		
Descripción: Se almacenan los tipos de procedimientos que se pueden efectuar en las causas.		
Atributo	**Tipo**	**Descripción**
id_procedimiento	serial	Número consecutivo ascendente que representa a cada procedimiento.
denom_procedimiento	char(20)	Denominación de los diferentes tipos de procedimientos.

Nombre: nom_cat_ocupacional		
Descripción: Se guardan las categorías ocupacionales que pueden poseer los acusados.		
Atributo	**Tipo**	**Descripción**
id_cat_ocupacional	serial	Número consecutivo ascendente que representa a cada una de las categorías ocupacionales.
denom_cat_ocupacional	char(20)	Denominaciones de las distintas categorías ocupacionales.

Anexo 3. Requisitos no funcionales.

Apariencia o interfaz externa.

- La interfaz a implementar debe ser sencilla, para que los usuarios que no son personas expertas en la rama de la informática, no necesiten tanto tiempo de adiestramiento.
- Por el uso diario y constante que tendrá el software, la interfaz debe ser agradable, que favorezca el estado de ánimo del cliente y que combine correctamente los colores, tipo de letra y tamaño y que los íconos estén en correspondencia con lo que representan.
- Deben utilizarse plantillas con un mismo estilo.

Usabilidad.

- El sistema debe ser de fácil manejo para los usuarios que tengan niveles básicos sobre la computación o hallan trabajado con la Web.
- Debe tener una opción de ayuda sobre las principales operaciones que se realizan y sus iconos respectivos para lograr un menor tiempo de aprendizaje.
- El sistema simulará tal y como es el proceso judicial para lograr el menor tiempo en cuanto a la comprensión del sistema y del proceso.

Rendimiento.

- La aplicación debe estar concebida para el consumo mínimo de recursos.
- El sistema debe ser capaz de formular la respuesta lo más rápido posible.

Soporte.

Para el servidor de aplicaciones:

- Se requiere que esté instalado un intérprete de ficheros PHP y con las últimas actualizaciones del lenguaje.

Para el servidor de base de datos:

- Se requiere que esté instalado un gestor de base de datos que soporte grandes volúmenes de datos y velocidad de procesamiento.

Para el cliente:

- Se requiere esté instalado un navegador que interprete Javascript y versiones HTML 3.0 o superior.

Portabilidad.

- El sistema deberá ser compatible con el sistema operativo UNIX (Linux).
- El sistema deberá ser compatible con el sistema operativo Windows (Versiones como 2000 y XP), siendo además accesible principalmente en la Intranet con el navegador Mozilla.

Hardware.

Para las computadoras del cliente:

- Se requiere tengan tarjeta de red.
- Se requiere tengan al menos 64 MB de memoria RAM.
- Se requiere al menos 100MB de disco duro.
- Procesador 512 MHz como mínimo.

Para los servidores:

- Se requiere tarjeta de red.
- Se requiere tenga la menos 256MB de RAM.
- Se requiere al menos 1GB de disco duro.
- Procesador 1.2 GHz como mínimo.

Software.

- El sistema se desarrollará con tecnología PHP versión 5.0 o superior.
- Se utilizará un servidor con el sistema operativo instalado Windows 2000 o superior, o con sistema operativo UNIX (Linux) preferencialmente.
- Se utilizará tecnología Apache versión 2.0 o superior para el servidor Web.
- El sistema utilizará una base datos implementada en PostgreSQL versión 8.0.
- En las computadoras de los clientes se garantizará versiones de Windows 2000 o superior, así como Linux y sus correspondientes distribuciones.
- En las computadoras de los clientes solo se requiere de un navegador (Internet Explorer versión 4.0 o superior, Mozilla Firefox versión 1.5 o superior.
- En caso de que el usuario no contara con los recursos suficientes para que la aplicación funcione con la arquitectura descrita, entonces la computadora tiene que tener instalados todos los programas antes mencionados.

Seguridad.

- El sistema debe comunicarse usando un protocolo seguro, (https).
- Chequear si el usuario que está accediendo al sistema está autenticado y brindarle servicio de autenticación.
- Mantener la integridad de la información, es decir que no se pierda durante su almacenamiento o transporte.
- Permitir que cuando se borre cualquier documento o información pueda existir una opción de advertencia antes realizar la acción.
- Realizar auditoría a los principales eventos dentro del sistema, registrando al usuario y los eventos efectuados.

Confiabilidad.

- La información manejada por el sistema estará protegida de acceso no autorizado y divulgación.

Integridad.

- La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción.

Fiabilidad.

- Debe garantizarse el resguardo de la información, de modo que haya varias copias, de forma tal que se posibilite la reinstalación del sistema y los datos, en caso de algún problema presentado en la explotación del mismo.

Legales.

- El sistema debe basarse en el documento que expone las normativas para el desarrollo de softwares en las FAR.
- La mayoría de las herramientas de desarrollo son libres y del resto, las licencias están avaladas.
- El sistema tendrá en cuenta lo establecido por la ¨ Ley de procedimiento penal militar¨ y en la ¨ Ley número 97, de los tribunales militares, en todo lo referido al desarrollo del trabajo judicial que se lleve al nuevo sistema.

GLOSARIO DE TÉRMINOS

Microsoft (acrónico de Microcomputer Software)

Es una empresa de Estados Unidos, fundada por Bill Gates y Paul Allen. Dueña y productora de los sistemas operativos: Microsoft DOS y Microsoft Windows, que se utilizan en la mayoría de las computadoras del planeta.

Sistema operativo (SO)

Es un conjunto de programas destinados a permitir la comunicación del usuario con un computador y gestionar sus recursos de una forma eficaz.

Lenguaje de programación

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, respectivamente.

Archivos secuenciales

Un archivo organizado secuencialmente es un conjunto de registros lógicamente relacionados cuya secuencia de acceso está determinada por su ordenamiento. Los registros deben ser grabados consecutivamente cuando el archivo es creado, y deben ser leídos de la misma manera cuando es usado posteriormente como entrada.

Archivos indexados

Es la aplicación de incluir índices en el almacenamiento de los archivos; de esta forma será más fácil buscar algún registro sin necesidad de ver todo el archivo.

Un índice en un archivo consiste en un listado de los valores del campo clave que ocurren en el archivo, junto con la posición de registro correspondiente en el almacenamiento masivo.

RAID (Redundant Array Of Independent/Inexpensive Disks).
Es un término que hace referencia a un conjunto de discos redundantes e independientes. Se utiliza para mejorar el rendimiento, la tolerancia, fallos y errores en los discos, así como también mejora la integridad de los datos.

ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad)

- Atomicidad (Indivisible): es la propiedad que asegura que la operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias.
- Consistencia: es la propiedad que asegura que sólo se empieza aquello que se puede acabar. Por lo tanto se ejecutan aquellas operaciones que no van a romper la reglas y directrices de integridad de la Base de Datos.
- Aislamiento: es la propiedad que asegura que una operación no puede afectar a otras. Esto asegura que dos transacciones sobre la misma información nunca generarán ningún tipo de error.
- Durabilidad: es la propiedad que asegura que una vez realizada la operación, ésta persistirá y no se podrá deshacer aunque falle el sistema.

SQL (Structured Query Language)
Es un lenguaje de acceso a las Bases de Datos, permite especificar todas las operaciones sobre la base de datos como por ejemplo: Inserción, Borrado, Actualización. Utiliza características de álgebra y cálculo relacional permitiendo de esta forma realizar consultas a la base de datos de forma sencilla.

DDL (Lenguaje de definición de datos)
Es el lenguaje proporcionado por el Sistema Gestor de Base de Datos para llevar a cabo tareas de definición de estructuras de la Base de Datos.

DML (Lenguaje de manipulación de datos)
Permite a los usuarios llevar a cabo consultas y manipulación de los datos.

IBM (International Business Machines Corporation)

Conocida coloquialmente como el Gigante Azul, es una empresa que fabrica y comercializa hardware, software y servicios relacionados con la informática.

Archivos planos

Un archivo plano es un archivo secuencial puro, o txt.
Conjunto de caracteres ANSI organizados de tal forma que permiten ser almacenados y recuperados. Se utilizan para la transferencia de datos.

Disparadores (trigger)

Un disparador es una regla en la que se especifica una acción que el SGBD debe ejecutar como respuesta a la ocurrencia de un evento, siempre que se cumpla la condición especificada.

Hipercubo

Un hipercubo se define como un cubo desfasado en el tiempo, es decir, cada instante de tiempo por el cual se movió pero todos ellos juntos.

Vistas

Una vista es el resultado dinámico de una o varias operaciones relacionales realizadas sobre las relaciones base. El contenido de una vista está definido como una consulta sobre una o varias relaciones base.

Catálogo (Diccionario de datos)

Es una estructura propia de la base de datos en la que se definen los elementos que forman parte de la misma.

Artefacto

Un artefacto es una información que es utilizada o producida mediante un proceso de desarrollo de software. Pueden ser artefactos un modelo, una descripción o un software. Los artefactos de UML se especifican en forma de diagramas, éstos, junto con la documentación sobre el sistema constituyen los artefactos principales que el modelador puede observar.

Herramienta CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador)

Las Herramientas CASE son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.

Multiplataforma

Es un término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de software, que puedan funcionar en diversas plataformas.

Una plataforma es una combinación de hardware y software usada para ejecutar aplicaciones; en su forma más simple consiste únicamente de un sistema operativo, una arquitectura, o una combinación de ambos.

Control de concurrencia multiversión

Consiste en que mientras se consulta una Base de Datos, cada transacción ve una imagen de los datos (una versión de la BD) como si fuera tiempo atrás, sin tener en cuenta el estado actual de los datos que hay por debajo. Esto evita que la transacción vea datos inconsistentes que pueden ser causados por la actualización de otra transacción concurrente en la misma fila de datos, proporcionando aislamiento transaccional para cada sesión de la Base de Datos.

N procesos son concurrentes cuando se ejecutan (acción por acción) en el mismo intervalo de tiempo.

Transacciones

Una transacción es un conjunto de procesos que se ejecutan uno después del otro. Si algún subproceso falla, lo realizado anteriormente debe reversarse para que los datos no se alteren, a este comportamiento se lo denomina todo o nada.

Tablespaces

Un tablespace es una unidad lógica de almacenamiento de datos representada físicamente por uno o más archivos de datos.

Servidores

Un servidor, en informática o computación, es el ordenador en el que se ejecuta un programa que realiza alguna tarea en beneficio de otras aplicaciones llamadas clientes.

Protocolo

Un protocolo de comunicación es la manera de comunicarse que tiene una computadora con otra, cuando se están transmitiendo datos entre sí.