Universidad de las Ciencias Informáticas

Facultad 7

Trabajo de diploma para optar por el Título de Ingeniero en Ciencias Informáticas

Título: Herramienta para el enmascaramiento de datos en el proyecto alas HIS.

Autores: Rey Ernesto Matos González

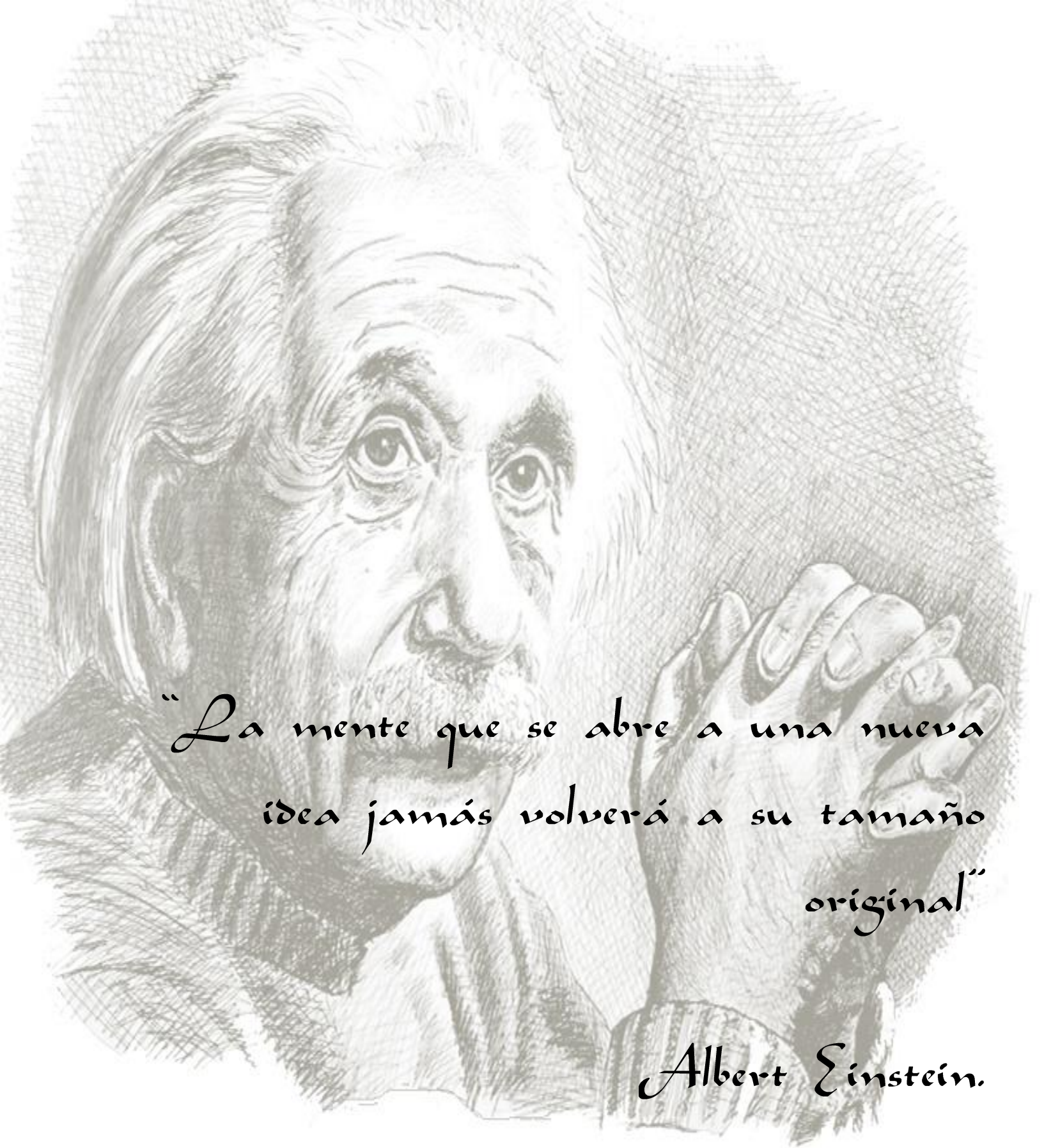
Javier Armando Muguercia Kindelán

Tutor: Ing. Yubismel Perdomo Velázquez

Cotutor: Ing. Yurielkis Matos Pérez

La Habana, junio 2012

"Año 55 de la Revolución"



"La mente que se abre a una nueva idea jamás volverá a su tamaño original"

Albert Einstein.

Ing. Yubismel Perdomo Velázquez: Graduado de Ingeniería Informática en el año 2006, profesor instructor. Ha impartido las asignaturas: Sistemas de Bases de Datos I, Sistemas de Bases de Datos II y Programación II. Ha participado en proyectos desempeñándose como programador y jefe de módulo. Fue jefe de las asignaturas: Sistemas de Bases de Datos I, Sistemas de Bases de Datos II y Programación II.
Correo electrónico: yubismel@uci.cu

Ing. Yurielkis Matos Pérez: Graduado de Ingeniería Informática en la UCI. Es profesor del departamento de IGSW.
Correo electrónico: ymatos@uci.cu

De Rey:

Gracias a mis compañeros de estudio y amigos incondicionales, que durante 5 años hemos compartido en buenos y malos momentos (Alexis, Daniellys, Hanssel, Andres, Yunior, Carlos, Yanisbel, Gloria, Daymer, David, Katy, Nelson y Leo).

A los tutores Yubismel y Yurielkis por sus consejos y su entrega en este trabajo.

A José Rafael, quien sin ninguna queja siempre estuvo ahí cuando lo necesité.

A mis familiares, quienes nunca dudaron y me ayudaron en todo lo que me hizo falta sin esperar nada a cambio solo que fuese la persona que siempre he sido.

A Inés María Quevedo y Aline Fernández Quevedo dos personas a las que quiero mucho y que nunca dudaron de mí.

Gracias a Javier quien ha estado siempre en disposición de trabajar para que todo salga bien.

A nené, tony, yani, mi mamá y mi papá porque sin ellos no habría logrado nada en la vida.

De Javier:

A mis amigas, Adita y Heidi (Maminskys), por los momentos vividos conmigo, cada dia que estuvimos juntos, por estar a mi lado en todos los instantes, en la alegría y la tristeza; a Fernando (El gordo), por ser como mi hermano, por acompañarme desde mucho antes de esta etapa y ser un amigo de confianza; a Danilo (El Perro), mi compañero en muchas aventuras dentro y fuera de aula.

Gracias a Noel Piloto (Noe), por ser siempre amigo y aparecer para salvarme cuando lo necesite, gracias a Saralys, por aguantarme en todo instante y siempre tener un tiempo para ayudarme, gracias a Betty, Jose el chino, Malena, Adrián y Miguel, por su aliento cada semana y los momentos compartidos, a Yeni por ser una amiga que nunca falto a la hora de realizar aclaraciones y por ser el "Artefacto" indispensable en este trabajo, gracias a Reynier (El Infla) por su ayuda incondicional en todo momento, por su amistad, gracias a todos los muchachos de mi grupo y otras amistades de la universidad.

Muchas gracias a mi compañero de tesis Rey, por ser el guía en todo momento, por trabajar juntos y siempre encontrar la manera de salir bien en cada situación, gracia a mis tutores, que han sido de gran ayuda en todo este proceso. Por último y muy especial, gracias a mi hermana Mercedes, que no está conmigo, a ella gracias por ser la mejor hermana del mundo, por todo lo que me ha enseñado, por su cuidado y atenciones y por no olvidarse de mí ni un dia.

De Rey:

A mis padres por su apoyo, por todo lo que han sacrificado para que yo lograra ser alguien en la vida, por su amor incondicional y su comprensión.

A mis hermanos quienes sé que me quieren tanto como yo a ellos.

De Javier:

A mi familia, mis padres, mi abuela, mi tía, mis hermanos y mi novia, por su apoyo en todo momento y porque son la razón por la que he llegado tan lejos, en especial a mi papa.

Resumen

El presente trabajo surge de la necesidad de mantener la integridad y confidencialidad de la información sensible contenida en la base de datos del proyecto alas HIS, para evitar la pérdida o sustracción indebida de la misma, principalmente en los ambientes no productivos (usuarios encargados de la realización de las pruebas y capacitación en los proyectos) manteniendo la estructura original de la información almacenada en la base de datos. Por esta razón se evaluaron sistemas existentes con objetivos similares, así como las técnicas y arquitecturas usadas para su implementación. Como resultado de este análisis se determinó la necesidad de desarrollar una herramienta informática que permita el enmascaramiento de datos para el proyecto alas HIS. A partir de esta decisión se trazaron y ejecutaron las tareas del proceso de desarrollo de dicha herramienta, basado en el Proceso Unificado de Desarrollo, las cuales de forma resumida fueron la gestión de requisitos, el modelado del negocio, el diseño y la implementación.

Como resultado de estas tareas se obtuvieron los artefactos derivados de cada flujo de trabajo y la herramienta desarrollada fue sometida al proceso de pruebas, arribándose a una herramienta acabada y funcional. El uso de la herramienta desarrollada (dataHISMask) constituirá una parte fundamental en la seguridad de la información del proyecto alas HIS garantizando la integridad y confidencialidad de la información almacenada, además proporcionará datos realistas para los ambientes de pruebas y de desarrollo.

Palabras claves: alas HIS, base de datos, enmascaramiento de datos, seguridad.

Índice

Introducción

La Informática es una ciencia que estudia métodos, procesos y técnicas con el fin de almacenar, procesar y transmitir información en formato digital, por lo que ha asumido el protagonismo del tratamiento de información. Debido a esto casi cualquier dato, por no decir todos los que pasan por una organización o institución determinada se almacena en algún soporte informático. El simple hecho de registrar una solicitud, implica que la información que se maneja de los clientes se registre en las aplicaciones informáticas de los distintos organismos a los que ha accedido. Debido a que las instituciones cada vez tienen más clientes y crece el volumen de la información, hacen uso de las bases de datos para almacenar dicha información, de ahí que las bases de datos se conviertan en un blanco para el robo y sustracción de los datos.

La mayoría de las veces, la existencia de rigurosos mecanismos de control de acceso hacen que los niveles de seguridad en los entornos productivos de software sean aceptables. Sin embargo son en los entornos no productivos (usuarios encargados de la realización de las pruebas y capacitación en los proyectos informáticos) donde los datos corren el mayor peligro.

En muchos casos, las bases de datos con información de suma importancia (información sensible) están expuestas a un gran número de personas que participan en los proyectos, cuyo acceso a la totalidad de los datos no es necesario. Estas personas pueden ser integrantes de los entornos no productivos, los cuales hacen copias exactas de los datos sensibles contenidos en las bases de datos para la realización de sus tareas. Cuando en estos entornos no se protegen debidamente los datos es posible que se produzca el robo o substracción de información vulnerando la confidencialidad de los mismos.

Existen diferentes métodos para la protección de los datos:

- ✓ La encriptación: es el proceso mediante el cual cierta información es cifrada de forma tal que el resultado sea ilegible a menos que se conozcan los datos necesarios para su interpretación.
- ✓ La autenticación: garantiza que solamente el personal autorizado tenga acceso a las bases de datos.
- ✓ Los antivirus: aseguran que no se transmitan datos utilizando programas maliciosos.
- ✓ El monitoreo: se ocupa de verificar que no se pueda tener acceso a los datos sensibles de manera innecesaria.

Los métodos estándar para proteger la privacidad en los ambientes de producción hacen que se dificulte el robo de información mientras es transmitida por algún medio, pero no impide el abuso de ella antes de que se realice alguno de estos métodos, además no son efectivos cuando son aplicados a los ambientes no productivos, en los cuales los desarrolladores y probadores requieren acceso a los datos reales.
Por otra parte, el enmascaramiento se refiere a la transformación de los datos de manera que solamente la información necesaria quede expuesta al usuario final, esta técnica no está orientada a reemplazar las tecnologías actuales de seguridad y privacidad, su objetivo es reforzarlas(1).

Aunque se tiende a pensar que el enmascaramiento es equivalente al cifrado, hay una diferencia fundamental entre los dos, la cifrada no necesariamente debe descifrarse, quedando expuestos los datos originales. En cambio el enmascaramiento garantiza que la información original nunca estará disponible para el usuario final, ya que solamente tendrá acceso a la información enmascarada y, hace que los datos de todas maneras permanezcan válidos.

La Universidad de las Ciencias Informáticas (UCI) es una universidad productiva, cuya misión es producir software y servicios informáticos a partir de la vinculación estudio–trabajo como modelo de formación(2). Está constituida por facultades, dentro de las cuales la Facultad 7 posee un Centro de Informática Médica (CESIM), dividido por departamentos, siendo uno de ellos el Departamento de Sistemas de Gestión Hospitalaria (DSGH). Dicho departamento desarrolla el Sistema de Información Hospitalaria alas HIS utilizando la tecnología web, haciendo uso del gestor de base de datos PostgreSQL proporcionando servicios para la creación, el almacenamiento, el procesamiento y consulta de la información almacenada en la base de datos, actuando como un intermediario entre la aplicación y los datos.

El sistema alas HIS almacena datos confidenciales o sensibles los cuales son visualizados en su formato original por los desarrolladores y probadores mientras realizan sus trabajos durante el ciclo de vida del proyecto, lo que trae aparejado que los datos sensibles puedan ser sustraídos y/o utilizados con fines dañinos, violándose de esta manera la confidencialidad e integridad de la información almacenada, conllevando un bajo nivel de seguridad de los datos en el proyecto.

Por lo anteriormente expuesto se ha identificado como **problema a resolver**: ¿Cómo garantizar la integridad y confidencialidad de los datos en el proyecto alas HIS para elevar la seguridad de la información almacenada? Esta investigación tendrá como **objeto de estudio**: Proceso de

enmascaramiento de datos; enmarcándose en el **campo de acción**: Enmascaramiento de datos para bases de datos PostgreSQL.

Para dar solución al problema anteriormente planteado se define como **objetivo general**: Desarrollar una herramienta informática que permita llevar a cabo el proceso de enmascaramiento de datos en el proyecto alas HIS.

Según lo expuesto se plantea la siguiente **idea a defender:** La implementación y puesta en práctica de una herramienta que permita realizar el proceso de enmascaramiento de datos en el proyecto alas HIS contribuirá a la seguridad garantizando la integridad y confidencialidad de la información almacenada.

Como **tareas de la investigación** se proponen las siguientes:

1. Estudio del estado del arte referente al proceso de enmascaramiento de datos a nivel nacional e internacional con el objetivo de verificar la existencia de sistemas que pudieran solucionar la situación que se plantea.
2. Asimilación de la metodología, plataforma, tecnologías, librerías, herramientas y pautas definidas por el Departamento de Sistemas de Gestión Hospitalaria para el desarrollo de sus aplicaciones.
3. Obtención, basado en la metodología, los artefactos resultantes del Modelado de Negocio, la Gestión de Requisitos, el Diseño y la Implementación.
4. Desarrollo de la herramienta aplicando las pautas de diseño definidas.
5. Realización de pruebas para la validación de la herramienta desarrollada.

Entender los pormenores del problema en cuestión puede resultar muy complejo en ocasiones, debido a esto, se hace necesario al inicio de la investigación, concretar los principales métodos científicos que posteriormente se utilizarán. El método científico de investigación es la forma de abordar la realidad, de estudiar la naturaleza, la sociedad y el pensamiento, con el propósito de descubrir su esencia y sus relaciones(3).

Métodos generales: El método hipotético-deductivo para la elaboración de la hipótesis central de la investigación, y para proponer nuevas líneas de trabajo a partir de los resultados parciales; el método histórico-lógico para el estudio crítico de los trabajos anteriores, y para utilizar estos como punto de referencia y comparación de los resultados alcanzados.

Métodos Lógicos: El método analítico-sintético para organizar la información de varias fuentes diferentes.

El desarrollo de la herramienta brindará una serie de beneficios a la sociedad, dígase a probadores, desarrolladores y sectores administrativos:

- ✓ Incremento de la protección y control ante el robo de datos, garantizando mayor seguridad a la información almacenada en el proyecto alas HIS.
- ✓ Prevención de filtraciones de datos confidenciales o sensibles contenidos en los diferentes módulos del proyecto en el intercambio con equipos internos o externos de la red.
- ✓ Facilitación de datos completamente funcionales luego de realizado el proceso de enmascaramiento de la información original confidencial.
- ✓ Realización de pruebas visualizando resultados coherentes o razonables (lo cual puede resultar más difícil si cada dato sensible es encriptado).

La estructura del documento se presenta de la siguiente manera:

Capítulo 1: Fundamentación Teórica: incluye el estudio del estado del arte de las aplicaciones existentes de enmascaramiento de datos. Presenta además una descripción de las tendencias, técnicas, tecnologías, metodologías y herramientas usadas para el desarrollo de la herramienta de enmascaramiento de datos.

Capítulo 2: Características del Sistema: describe el flujo actual de los procesos del negocio y se realiza el análisis para definir las funcionalidades y los requisitos no funcionales del sistema a desarrollar.

Capítulo 3: Análisis y Diseño del Sistema: contiene la modelación y construcción de la estructura de la herramienta.

Capítulo 4: Implementación y Pruebas: describe la implementación de las clases y subsistemas en términos de componentes de la solución propuesta. Presenta las pruebas realizadas a la herramienta en aras de validar su correcto funcionamiento.

Capítulo 1: Fundamentación Teórica.

En el presente capítulo se efectúa el estudio de aplicaciones existentes de enmascaramiento de datos, se investiga acerca de las principales empresas proveedoras de este tipo de aplicación. Además contiene un análisis de las metodologías, herramientas y tecnologías existentes para desarrollar la herramienta de enmascaramiento de datos para el proyecto alas HIS.

El DSGH desarrolla el Sistema de Información Hospitalaria alas HIS, donde son almacenados datos relacionados con la medicina, los que se pueden considerar confidenciales o sensibles, haciendo uso del gestor de base de datos PostgreSQL, el que proporciona servicios para la creación, el procesamiento y consulta de la información almacenada en la base de datos, actuando como un intermediario entre la aplicación y los datos. La información que se trata en el proyecto es visualizada en su formato original por los desarrolladores y probadores mientras realizan sus trabajos durante el ciclo de vida del proyecto, por lo que es posible que los datos sensibles puedan ser sustraídos y/o utilizados con fines dañinos, violándose de esta manera la confidencialidad e integridad de la información almacenada, conllevando un bajo nivel de seguridad de los datos en el proyecto.

Confidencialidad de la información.

La confidencialidad es la cualidad que posee cierta información de mantenerse reservada para el conocimiento de una persona o de algunas, pero que no debe ser expuesta en forma masiva.

Existen normas que expresamente disponen para algunas profesiones el deber de no divulgar las informaciones que reciben, como por ejemplo, los trabajadores de la salud no deben revelar las enfermedades de sus pacientes, salvo a éstos, a sus familiares directos o por requerimiento judicial en ciertos casos(4).

Integridad de la información.

La integridad de datos se refiere a los valores reales que se almacenan y se utilizan en las estructuras de datos de la aplicación. La aplicación debe ejercer un control deliberado sobre todos los procesos que utilicen los datos para garantizar la corrección permanente de la información(5).

1.1 Enmascarado de Datos.

La seguridad de los datos es esencial, ya que la divulgación de la información puede ocurrir a través de publicaciones de los empleados o al dejar a la vista de forma accidental datos confidenciales. El coste de las infracciones de seguridad de datos, en términos monetarios y de credibilidad de las empresas es elevado.

Aunque las empresas toman medidas para la protección de los datos sensibles en las bases de datos, no son suficientes para garantizar la seguridad en los entornos no productivos. En muchas ocasiones las bases de datos con información personal están expuestas a la mirada y/o copia de un conjunto de personas que participan en los distintos proyectos informáticos, por lo que se hace necesario desarrollar aplicaciones que no expongan los datos sensibles, regulando que ni los propietarios tengan acceso a esta información.

El Des-Identificar o Enmascaramiento es una forma de garantizar que la información original nunca estará disponible para el usuario final ya que él solamente tendrá acceso a la información enmascarada. En un ambiente no productivo, el enmascaramiento de datos es un proceso que remueve o transforma sistemáticamente elementos de datos confidenciales que pueden identificar a un individuo, cliente o cuenta bancaria. Las técnicas de enmascaramiento permiten alterar los datos sin perder sus atributos, de tal modo que garantice la confidencialidad de la información enmascarada (6).

El proyecto alas HIS almacena en su base de datos información que en su mayoría es relacionada con la medicina, la cual no debe de ser expuesta a personas que no necesiten conocer estos datos.
Aunque el proyecto cumple con los métodos de protección de datos en los entornos de desarrollo y prueba, la información está presente en la base de datos en su estado original, lo que propicia que usuarios que interactúan con ella frecuentemente visualicen dichos datos, por lo que se viola la integridad y confidencialidad de la información almacenada.

El uso de una herramienta para el enmascaramiento de datos en el proyecto alas HIS, prevé que los datos sensibles no lleguen a terceros, y garantiza la seguridad en un área tan vulnerable (entorno no-productivo) sustituyendo la información original por alguna que es similar.

1.1.1 Antecedentes.

A nivel mundial la técnica del enmascarado de datos ha despertado un gran interés en las grandes empresas en cuanto a seguridad y confidencialidad se trata, favoreciendo al auge de la fabricación de aplicaciones que permitan la puesta en práctica de dicha técnica para los diferentes gestores de bases de datos, tal es el caso de:

Sistemas internacionales.

Informatic Dynamic Data Masking: De la compañía de integración de datos informáticos de Estados Unidos, es un software de enmascaramiento dinámico de datos que proporciona funcionalidades en tiempo real para evitar que los usuarios no autorizados tengan acceso a información confidencial(7).

Este software logra disminuir drásticamente el riesgo de filtraciones de datos durante el enmascaramiento dinámico de datos, es capaz de elegir automáticamente la técnica basándose en políticas como el cifrado de nombres o el ocultamiento de la información sobre tarjetas de crédito y salarios.

Uno de sus principales beneficios es que preserva la apariencia personal, la coherencia y la integridad de los datos enmascarados.

PowerCenter Data Masking Option: De Informática Corporation (NASDAQ: INFA) protege la información confidencial y privada al enmascararla durante su transmisión para producir datos que parecen reales, con lo que se reduce el riesgo de fallos de seguridad y de incumplimiento de normativas.

Las distintas técnicas y algoritmos de enmascaramiento de datos garantizan la asignación al azar, a la vez que mantienen la naturaleza original de los datos y mantienen la integridad referencial. Las reglas y el contenido especializados, incorporados en esta opción, se encargan de los campos confidenciales, como el nombre, la dirección, el número de la Seguridad Social, de la tarjeta de crédito o el teléfono(8).

DgMasker: De Dataguise es un software destinado para simplificar los riesgos potenciales de robo, pérdida o violación externa de los datos, con interfaz visual fácil de usar, soportando enmascarado para Oracle, Microsoft SQL Server y bases de datos DB2 para Linux, Windows y AIX.

Protege la información de identificación personal y otros datos confidenciales o de propiedad con respecto a la divulgación, es decir aborda la necesidad de controlar los datos confidenciales de las empresas en los ambientes de desarrollo, prueba, control de calidad y otros(9).

El principal inconveniente de los software internacionales es que presentan licencias de software propietarios, lo que trae como consecuencia que, además del costo correspondiente al producto, se debe de invertir en el pago de licencias por cada estación de trabajo, pues son software de escritorio. Se tiene además que ninguno realiza el proceso de enmascaramiento para PostgreSQL.

Sistemas nacionales.

Mask-PG: De la UCI, es una herramienta de enmascarado de datos para bases de datos PostgreSQL que evita la pérdida o sustracción inapropiada de los datos tanto en ambientes productivos, como no productivos(10).

Mask-PG es una aplicación de escritorio, que no cumple con las pautas de desarrollo del DSGH y su uso es complejo cuando se trata de una base de datos de gran tamaño, como es el caso de la base de datos del cliente.

Debido a que ninguna de las herramientas antes mencionadas puede realizar el proceso de enmascaramiento en el proyecto alas HIS, surge la necesidad de desarrollar una herramienta informática que permita llevar a cabo esta importante tarea para dicho proyecto.

1.1.2 Métodos o técnicas comunes del enmascaramiento de datos.

Existen diversas técnicas usadas para el enmascaramiento de datos, entre las que se encuentran:

Sustitución: La técnica de sustitución consiste en reemplazar al azar el contenido de una columna de datos con información que es similar pero que no tiene relación ninguna con la real.

La sustitución es muy efectiva en términos de preservar la apariencia actual de los datos. El inconveniente es que en una base de datos bastante grande se necesitará de una base de conocimientos lo suficientemente grande ya que debe existir una sustitución para cada columna de datos.

Barajas: Consiste en remplazar al azar el contenido de una columna de datos con información que es similar pero que no tiene relación ninguna con la real, los datos se derivan de la propia columna, es decir,

los datos de una columna se trasladan al azar entre las filas hasta que no haya ninguna correlación razonable con el resto de la información en la fila.

Variación de números y fechas: Muy útil para fechas y datos numéricos, implica la modificación de cada número o valor de fecha en una columna por algunos al azar. En pocas palabras, el algoritmo trata de modificar cada número o valor de fecha en una columna por un porcentaje aleatorio de su valor real. Esta técnica tiene la ventaja agradable de proporcionar un disfraz razonable para los datos mientras se mantiene el rango y la distribución de los valores en la columna dentro de los límites existentes.

Cifrado: Esta técnica consiste en cifrar los datos con cierta clave ofreciendo la opción de dejar los datos en su lugar y visibles para las personas que conozcan dicha clave.

Anulación de salida o eliminación: Consiste en eliminar una columna de datos mediante su sustitución con valores NULL. Es una eficaz manera de asegurar y controlar la visibilidad excesiva de la información sensible en los entornos de desarrollo y prueba.

Enmascarado de datos de salida: Consiste en la sustitución de determinados campos por un carácter de máscara. Este método oculta el contenido de los datos y conserva el mismo formato en las pantallas de interfaz e informes (11).

Teniendo en cuenta el tipo de información que almacena el proyecto alas HIS, se implementarán durante el desarrollo de la herramienta los métodos **Barajas** y **Variación de números y fechas**, asegurando que en el caso del método Barajas la información provenga de la propia base de datos, manteniendo la similitud de la información. Para el método de Variación de números y fechas se limitará a variar solamente los campos numéricos (begint, numeric, smallin, double precisión, decimal, real e integer), es decir que para la fecha se usará la técnica barajas. La herramienta será la encargada de identificar la técnica a usar en una situación determinada.

1.1.2 Arquitecturas para el enmascaramiento de datos.

Para el diseño de software de enmascaramiento de datos existen dos arquitecturas básicas.

Sobre la Marcha, Servidor a Servidor: En esta arquitectura los datos no existen en la base de datos destino antes del enmascaramiento. Las reglas de anonimato se aplican como parte del proceso de mover

los datos desde el origen al destino. A menudo este tipo de enmascaramiento está integrado en el proceso de clonación que crea la base de datos de destino(11).

Esta arquitectura brinda la ventaja de que los datos nunca están presentes en su forma original en la base de datos destino.

La desventaja que presenta este software es que cualquier error en el proceso interrumpiría la transferencia de los datos.

En el Sitio (In-Situ): En este estilo, el clon de la base de datos a ser enmascarado se crea por otros medios y el software simplemente opera sobre la base de datos clonada. Hay dos tipos de In-Situ de enmascaramiento: reglas de enmascaramiento que son ejecutadas y controladas como una entidad independiente en el destino y las reglas de enmascaramiento de datos que están controladas por un sistema diferente que luego se conecta con el destino y controla la ejecución de las reglas(11).

Con el uso de esta arquitectura se garantiza que se pueda realizar el proceso de enmascaramiento en cualquier momento, las operaciones de enmascaramiento son independientes del proceso de clonación de la base de datos. Aunque también propicia que los datos se presenten en su estado auténtico en la base de datos destino por lo que será necesario durante ese tiempo mantener las medidas de seguridad adecuadas.

Luego de analizadas las arquitecturas de enmascaramiento usadas para el diseño de software de enmascaramiento de datos se decide usar la arquitectura **In-Situ**, ya que separa el proceso de clonación del de enmascarado, brindando la posibilidad de seleccionar lo que se desea enmascarar (módulos, tablas y atributos).

1.2 Tecnologías, técnicas, metodologías y herramientas usadas.

El Departamento de Sistemas Hospitalarios propone un grupo de herramientas y tecnologías para el desarrollo de sus aplicaciones, las cuales son abordadas a continuación.

1.2.1 Metodologías de desarrollo de software.

Se propone como metodología de desarrollo el Proceso Unificado de Desarrollo (RUP) y como lenguaje de modelado: el Lenguaje Unificado de Modelado (UML).

1.2.1.1 Proceso Unificado de Desarrollo (RUP).

RUP es el resultado de varios años de trabajo y uso práctico en el que se han unificado técnicas de desarrollo, a través del UML, y trabajo de muchas metodologías utilizadas por los clientes. En RUP se han agrupado las actividades en grupos lógicos en los que se definen nueve flujos de trabajo principales. Los seis primeros son conocidos como flujos de ingeniería y los tres últimos como flujos de apoyo. El ciclo de vida de RUP se caracteriza por ser dirigido por caso de uso, centrado en la arquitectura, iterativo e incremental(12).

1.2.1.2 Lenguaje Unificado de Modelado (UML).

UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software. Permite la modelación de sistemas con tecnología orientada a objetos. Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como RUP), pero no especifica en sí mismo qué metodología o proceso utilizar.

Este lenguaje de modelado formal permite tener un mayor rigor en la especificación, realizar una verificación y validación del modelo desarrollado, automatizar determinados procesos y generar código a partir de los modelos y a la inversa. Esto último permite que el modelo y el código estén actualizados(13).

1.2.2 Arquitectura Cliente-Servidor.

Esta arquitectura consiste en un programa en el que el cliente (es una aplicación informática que se utiliza para acceder a los servicios que ofrece un servidor) realiza peticiones a otro programa servidor (es una computadora que, formando parte de una red, provee servicios a otras denominadas clientes) que le da respuesta. Aunque esta idea se puede aplicar a programas que se ejecutan sobre una sola computadora es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras (14).

La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un sólo programa.

Cliente: Es el que inicia un requerimiento de servicio. El requerimiento inicial puede convertirse en múltiples requerimientos de trabajo a través de redes LAN o WAN. La ubicación de los datos o de las aplicaciones es totalmente transparente para el cliente(15).

Servidor: Es cualquier recurso de cómputo dedicado a responder a los requerimientos del cliente. Los servidores pueden estar conectados a los clientes a través de redes LAN o WAN, para proveer de múltiples servicios a estos(15).

Entre las características fundamentales de esta arquitectura encontramos que tanto el cliente como el servidor pueden realizar tareas en forma conjunta como separada ya que el cliente también tiene sus propias aplicaciones, archivos y bases de datos y que además, pueden estar en la misma plataforma o en plataformas diferentes. Por otra parte, el servidor puede brindar varios servicios a la vez, tanto al mismo cliente como a clientes múltiples(16).

1.2.2.1 Modelo Vista Controlador (MVC).

MVC es un patrón de diseño de arquitectura de software que separa la parte lógica de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página. El modelo es el sistema de gestión de base de datos y la lógica de negocio, y el controlador es el responsable de recibir los eventos de entrada desde la vista(17).

Modelo: Representa las estructuras de datos. Las clases del modelo de clases contendrán funciones para consultar, insertar y actualizar información de la base de datos.

Vista: Es la información presentada al usuario y para su interacción con él, más conocida como interfaz.

Controlador: Actúa como intermediario entre el Modelo, la Vista y cualquier otro recurso necesario para generar una página web. Es el responsable de recibir los eventos de entrada desde la vista (18).

1.2.3 Capa de la vista.

La capa de presentación es la que presenta el sistema al usuario, le comunica la información y captura la que este introduce en un mínimo de procesos. Esta capa se comunica únicamente con la capa de negocio(19).

1.2.3.1 Java Server Faces (JSF).

Con el uso de JSF como tecnología y ambiente de desarrollo o corrida (framework[1]) para aplicaciones Java basadas en Web se simplifica el desarrollo de interfaces de usuario en la aplicación. JSF usa Java Server Pages (JSP) como la tecnología que permite hacer el despliegue de las páginas, facilita y agiliza el diseño de interfaces de usuario, pues implementa una serie de componentes, estado de los mismos, eventos del lado de servidor, entre otras ventajas(20).

1.2.3.2 RichFaces.

El uso del framework de código abierto RichFaces permite la posibilidad de añadir capacidad Ajax dentro de la aplicación JSF existentes sin recurrir a JavaScript. RichFaces incluye ciclo de vida, validaciones, conversiones y la gestión de recursos estáticos y dinámicos. Los componentes de RichFaces están construidos con soporte Ajax que puede ser fácilmente incorporado dentro de las aplicaciones JSF(21).

1.2.3.3 Ajax4JSF.

Para lograr realizar cambios sobre las páginas sin necesidad de recargarlas totalmente, lo que significa aumentar la interactividad, velocidad y usabilidad de las aplicaciones se hace uso de Ajax, acrónimo de Asynchronous JavaScript And XML (JavaScript[2] asíncrono y XML[3]), es una técnica de desarrollo web para crear aplicaciones interactivas. Esta técnica se ejecuta del lado del cliente, es decir, en el navegador de los usuarios, mientras se mantiene en segundo plano la comunicación asíncrona con el servidor.

Ajax es una técnica válida para múltiples plataformas y utilizable en muchos sistemas operativos y navegadores, dado a que está basada en estándares abiertos como JavaScript y Document Object Model (DOM)(22).

Ajax4jsf es una librería de código abierto que se integra totalmente en la arquitectura de JSF y extiende la funcionalidad de sus etiquetas dotándolas con tecnología Ajax de forma limpia y sin añadir código JavaScript. Mediante este framework se puede variar el ciclo de vida de una petición JSF, recargar

[1]Estructura bien definida que da soporte a un proyecto web, también ayuda a que el proyecto sea organizado y bien desarrollado.

[2]Es un lenguaje de programación interpretado que se utiliza principalmente para crear páginas web dinámicas.

[3]XML no es un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades, de ahí que se le denomine metalenguaje.

determinados componentes de la página sin necesidad de recargarla por completo, realizar peticiones automáticas al servidor y controlar cualquier evento de usuario(23).

1.2.3.4 Facelets.

La utilización de Facelets como framework para plantillas centrado en la tecnología JSF, se logra que JSP y JSF puedan funcionar conjuntamente en una misma aplicación web ya que estos no se complementan naturalmente. JSP procesa los elementos de la página de arriba a abajo, mientras que JSF dicta su propio re-rendering (ya que su ciclo de vida está dividido en fases marcadas). Facelets llena este vacío entre JSP y JSF, siendo una tecnología centrada en crear árboles de componentes y estar relacionada con el complejo ciclo de vida JSF(24).

1.2.3.5 HTML.

El HTML, Hyper Text Markup Language (Lenguaje de marcación de Hipertexto) es el lenguaje de marcas de texto utilizado normalmente en la www (World Wide Web). Fue creado en 1986 por el físico nuclear Tim Berners-Lee; el cual tomó dos herramientas preexistentes: El concepto de Hipertexto (Conocido también como link o ancla) el cual permite conectar dos elementos entre sí y el SGML (Lenguaje Estándar de Marcación General) el cual sirve para colocar etiquetas o marcas en un texto que indique como debe verse. HTML no es propiamente un lenguaje de programación como C++, Visual Basic entre otros, sino un sistema de etiquetas. HTML no presenta ningún compilador, por lo tanto algún error de sintaxis que se presente éste no lo detectará y se visualizará en la forma como éste lo entienda.

El entorno para trabajar HTML es simplemente un procesador de texto, como el que ofrecen los sistemas operativos Windows (Bloc de notas), UNIX (el editor vi o ed) o el que ofrece MS Office (Word). El conjunto de etiquetas que se creen, se deben guardar con la extensión .htm o .html

Estos documentos pueden ser mostrados por los visores o "browsers" de páginas Web en Internet, como Netscape Navigator, Mosaic, Mozilla Firefox, Opera y Microsoft Internet Explorer(25).

1.2.3.6 CSS.

Hojas de Estilo en Cascada (Cascading Style Sheets) es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos.

CSS se utiliza para dar estilo a documentos HTML y XML, separando el contenido de la presentación. Los estilos definen la forma de mostrar los elementos HTML y XML. CSS permite a los desarrolladores web controlar el estilo y el formato de múltiples páginas web al mismo tiempo. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento(26).

1.2.4 Capa del controlador.

La capa de negocio es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para almacenar o recuperar los mismos(19).

1.3.4.1 JBoss Seam 2.1.

El uso de JBoss Seam permite integrar la capa de presentación (JSF) con la capa de negocios y persistencia (EJB), funcionando, según versa su significado en español, como una “costura” entre estos componentes. Seam también se integra perfectamente con otros frameworks como: RichFaces, ICE Faces, MyFaces, Hibernate y Spring(27).

1.2.5 Tecnologías horizontales.

Existen un conjunto de tecnologías que se extienden horizontalmente por todas las capas antes mencionadas y sirven de soporte a las tecnologías que se utilizan en cada una de ellas. Las mismas se describen a continuación.

1.2.5.1 Java Platform Enterprise Edition (JavaEE 5).

Java Platform Enterprise Edition o Java versión 5 es una plataforma de programación (parte de la Plataforma Java) para desarrollar y ejecutar software de aplicaciones en lenguaje de programación Java con arquitectura de N niveles distribuida. Se basa ampliamente en componentes de software modulares y se ejecuta sobre un servidor de aplicaciones(28).

1.2.5.2 JBoss Server 4.2.

Es el servidor de aplicaciones de código abierto más desarrollado del mercado. Por ser una plataforma certificada J2EE, soporta todas las funcionalidades de J2EE 1.4 e incluye servicios adicionales como clustering, caching y persistencia. JBoss es ideal para aplicaciones Java y aplicaciones basadas en la web. También soporta Enterprise Java Beans (EJB) 3.0, lo que hace el desarrollo de las aplicaciones mucho más simple. Además, al ser desarrollado con tecnología Java, es multiplataforma(29).

1.2.5.3 Java Runtime Environment (JRE 6).

JRE es el acrónimo de Java Runtime Environment (entorno en tiempo de ejecución Java) y se corresponde con un conjunto de utilidades que permite la ejecución de programas Java sobre todas las plataformas soportadas. JVM (máquina virtual Java) es una instancia de JRE en tiempo de ejecución. Este interpreta el código Java y está compuesto además por las librerías de clases estándar que implementan el API de Java. Ambas JVM y API deben ser consistentes entre sí, de ahí que sean distribuidas de modo conjunto(30).

1.2.5.4 JBoss Tools 3.3.

Es un conjunto de plugin para el Eclipse que permite el manejo de diferentes frameworks que facilitan el desarrollo de aplicaciones. Está constituido por varios módulos: RichFaces VE, Seam Tools, Hibernate Tools y JBoss AS Tools(31).

1.2.6 Java.

Java es el lenguaje de programación orientado a objetos que define el DSGH, desarrollado por *Sun Microsystems*. Entre algunas de sus características se pueden mencionar: que elimina la complejidad de los lenguajes como "C" y da paso al contexto de los lenguajes modernos orientados a objetos, independiente de plataforma (multiplataforma), corre sobre una máquina virtual (interpretado), robusto, dinámico y seguro, limita lo que se puede hacer o no con los recursos críticos de una computadora(32).

Java se volvió más popular a partir de la aparición de la especificación de Servlets y JSP (Java Server Pages) una tecnología orientada a crear páginas web. Los servlets y las JSPs supusieron un importante avance ya que el API (Interfaz de Programación de Aplicaciones) es muy sencillo, flexible y extensible(33).

1.2.7 Herramientas.

1.2.7.2 Visual Paradigm 6.4.

El uso de la herramienta Visual Paradigm hace posible que se logre el soporte del ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a la construcción de aplicaciones de mejor calidad y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML(34).

1.2.7.3 PostgreSQL Server 8.3.

Para el almacenamiento de los datos en el DSGH se hace uso de PostgreSQL el cual es un Sistema de Gestión de Bases de Datos Objeto-Relacional (Object-Relational Database Management System (ORDBMS)) que no tiene costo asociado por lo que se puede disponer de su código fuente, modificarlo y redistribuirlo libremente. PostgreSQL garantiza concurrencia, para lo cual utiliza la tecnología de Control de Concurrencia Multi-Versión (Multiversion concurrency control (MVCC)), con lo que se logra que ningún lector sea bloqueado por un escritor. Es extensible, soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario.

Tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/PGSQL. Este lenguaje es comparable al lenguaje procedural del sistema de gestión de base de datos relacional Oracle, PL/SQL. En cuanto a sus funciones, poseen bloques de código que se ejecutan en el servidor los cuales pueden ser escritos en varios lenguajes, con la potencia que cada uno de ellos brinda. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que se intente conectar a la base de datos. Tiene una adecuada documentación, además de contar con una comunidad de usuarios y desarrolladores a los que acudir en caso de tener problemas(35).

1.2.7.4 Eclipse 3.4.

Para el desarrollo de la aplicación se hace uso del Eclipse, el cual es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma

típicamente ha sido usada para desarrollar entornos de desarrollo integrados (IDE), como el IDE de Java llamado Java Development Toolkit (JDT) y el compilador (ECJ) que se entrega como parte del Eclipse.

El IDE de Eclipse emplea módulos (plugin) para proporcionar toda su funcionalidad al frente de la plataforma de los llamados clientes enriquecidos. Esto lo diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software, que adicionalmente permite al Eclipse extenderse, usando otros lenguajes de programación como C/C++, Python y Java(36).

Conclusiones Parciales.

En este capítulo se realizó un estudio del enmascarado de datos y las diferentes herramientas existentes en el mundo que realizan dicho proceso, arrojando como resultado: no se encontró ninguna herramienta que permita realizar el enmascarado de datos en el proyecto alas HIS. Además se hizo énfasis en las arquitecturas básicas donde se seleccionó para el diseño de la aplicación la arquitectura **In-Situ**. Se abordaron las técnicas de enmascarado de datos, de las cuales **Baraja** y V**ariación de números** serán las implementadas en la herramienta para el enmascarado de datos, apoyados por la metodología de desarrollo RUP, haciendo uso del lenguaje de programación Java con la ayuda del IDE de desarrollo Eclipse.

Capítulo 2: Características del sistema.

Una parte imprescindible en el desarrollo de cualquier herramienta, es el profundo análisis de las características de la misma y constituye un eslabón fundamental la comprensión de los procesos existentes en el dominio del problema a resolver. En este capítulo se aborda la estructura de la herramienta. Se especifican los requerimientos funcionales que debe cumplir, así como elementos fundamentales para su diseño y la arquitectura para su implementación.

2.1 Flujo actual del proceso.

Los entornos no productivos posibilitan que se visualice y adquiera información sensible inapropiadamente de las Bases de Datos (BD). Las instituciones bajo este entorno muchas veces contienen en sus BD información auténtica, la cual se convierte en víctima de expropiación indebida, por ejemplo, si un miembro de un proyecto necesita realizar una tarea en la cual deba acceder a información contenida en la BD, hará uso de esta en su estado auténtico, favoreciendo el uso indebido de estos datos. El proyecto alas HIS se encuentra en una situación similar a la antes expuesta, lo que trae como consecuencia que no se aseguren principios de la seguridad informática tales como la confidencialidad y la integridad de los datos.

2.2. Modelo de dominio.

El modelo de dominio puede ser tomado como el punto de partida para el diseño del sistema. Este modelo permite mostrar de manera visual los principales conceptos u objetos del mundo real que se manejan, ayudando a los usuarios, desarrolladores e interesados a utilizar un vocabulario común, para poder comprender el contexto en que se desarrolla el sistema. Contribuye a identificar personas, eventos, transacciones y objetos involucrados en el sistema.

A continuación en la figura 2.1 se muestra el diagrama del Modelo de dominio para lograr un mayor entendimiento de la actualidad del negocio.

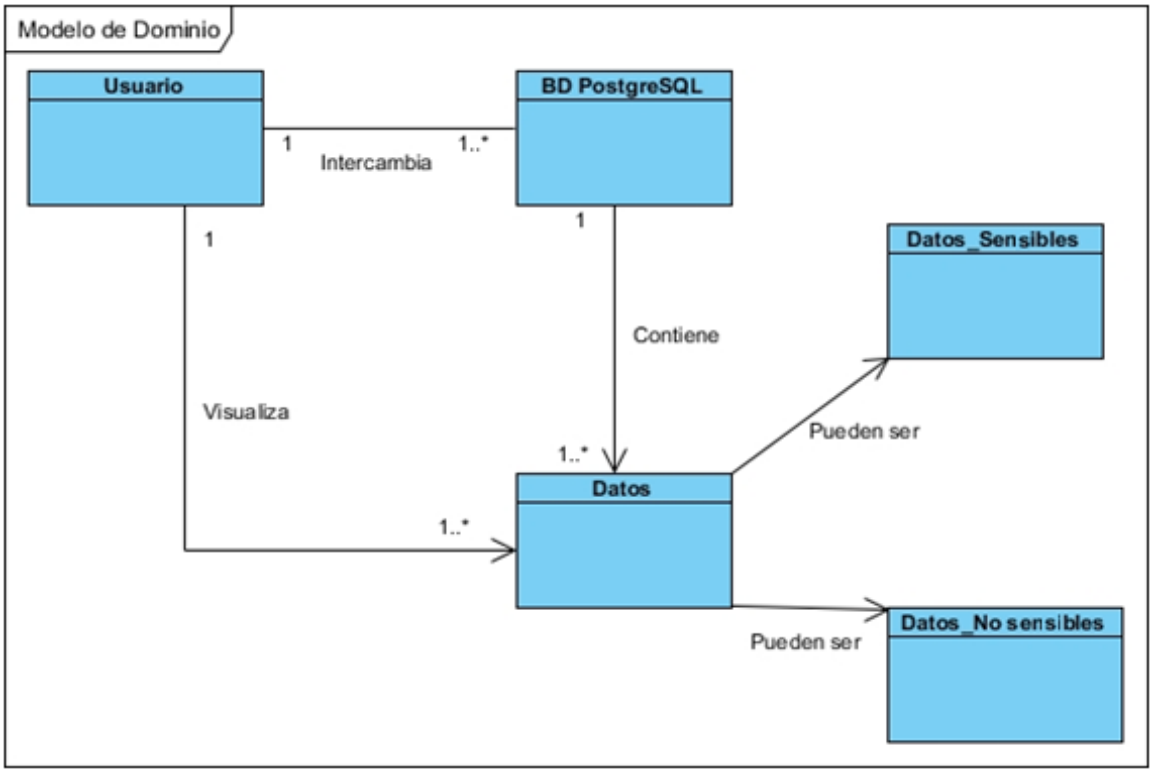


Figura 2.1: Diagrama Modelo de Dominio.

Como se muestra en el modelo de Dominio, un usuario es cualquier personal con acceso a la base de datos y que interactúa con esta, la base de datos contiene un volumen de información que puede ser en ocasiones datos de vital importancia para el desarrollo del proyecto, los cuales el usuario puede visualizar, utilizar e incluso modificar sin previo permiso, violando los principios de la integridad y confidencialidad de los datos. Para tener un mayor entendimiento a continuación se muestra la tabla 2.1 con las descripciones de los conceptos u objetos fundamentales.

Conceptos u Objetos	Descripción
Usuario	Usuario es cualquier integrante del proyecto que accede a la base de datos para hacer uso de la información guardada en sus labores o tareas.
BD PostgreSQL	Es la BD del proyecto, donde se almacena la información, bajo el uso del gestor de base de datos PostgreSQL.
Datos	Información contenida en la BD PostgreSQL del proyecto, pueden ser visualizados por los usuarios.
Datos Sensibles	Información contenida en la BD PostgreSQL del proyecto, con un grado crítico para el funcionamiento del mismo, información sumamente importante.
Datos no Sensibles	Información contenida en la BD PostgreSQL del proyecto, son datos comunes, no tienen gran impacto en el funcionamiento del proyecto.

Tabla 2.1: Descripciones de los conceptos u objetos fundamentales.

2.3 Propuesta del sistema.

Para solucionar el problema existente en el proyecto alas HIS se plantea desarrollar una herramienta que garantice la confidencialidad e integridad de la información almacenada en la BD PostgreSQL del proyecto, mediante el uso del enmascaramiento de los datos. Esta herramienta prevé que en caso de pérdida o sustracción indebida de datos no puedan ser usados con fines dañinos, además permitirá la interacción de los miembros del proyecto con la BD para así obtener un mayor resultado en el desempeño de sus labores.

2.3.1 Objetivos de la herramienta.

La herramienta tiene como objetivos permitir al usuario conectarse a la base de datos auténtica, realizar la clonación de la base de datos original (BDO), seleccionar los datos que desea enmascarar y realizar el enmascarado de la información a la base de datos clonada (BDC) o base de datos destino (BDD), mostrando finalmente los resultados del proceso realizado.

En esencia el fin esperado es un mayor resguardo de la información contenida en la BDO, evitando todo peligro que pueda afectar el funcionamiento del proyecto, garantizando el acceso a la información verdadera solo a aquel usuario con permiso previo a su utilización. A continuación se muestra en la imagen qué se quiere lograr con el desarrollo de esta herramienta de enmascarado de datos.

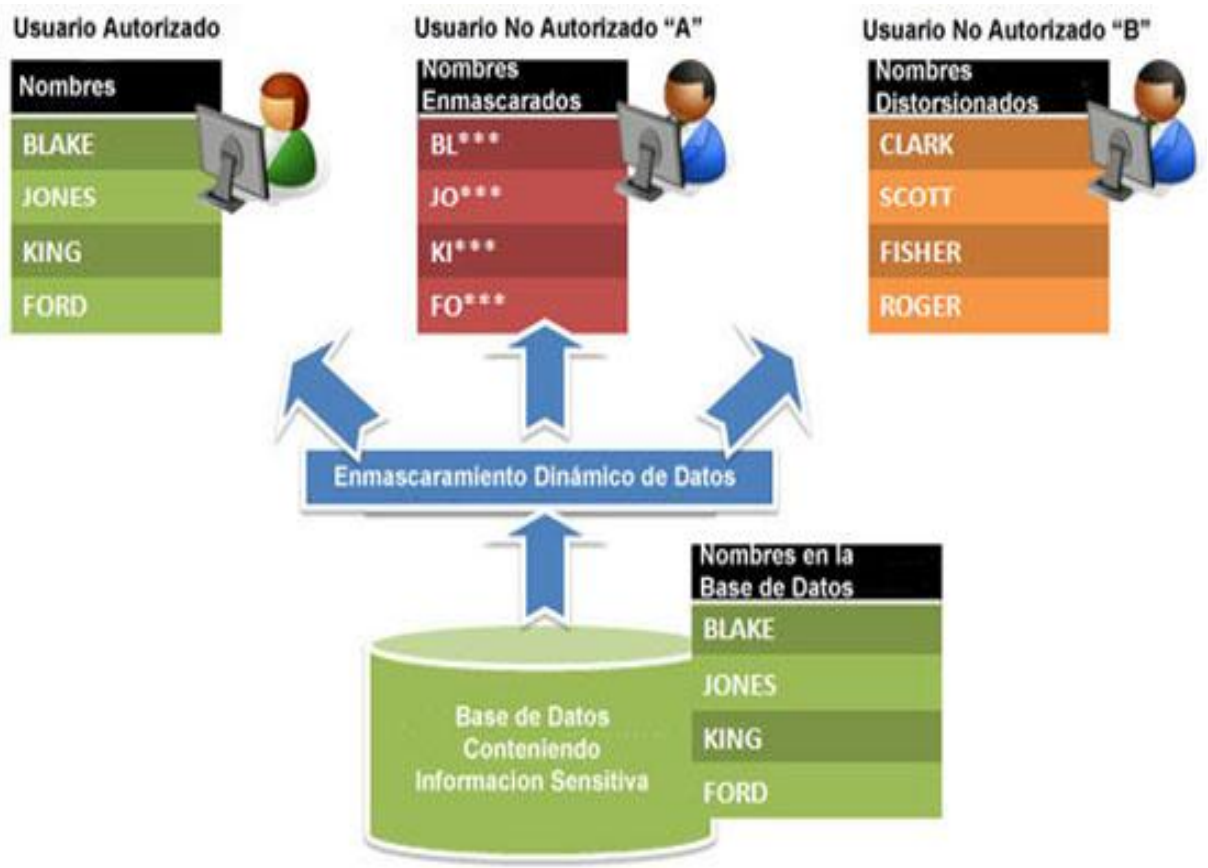


Figura 2.2: Resultados de la aplicación del enmascaramiento de datos.

Como se puede observar en la figura 2.2, una vez realizada las funciones de enmascaramiento en la base de datos solo el usuario autorizado a visualizar la información original podrá hacerlo, los demás usuarios solo tendrán en su poder información enmascarada, o sea datos similares y con la misma estructura que los originales, pero no verdaderos, dándole seguridad a los datos sensibles contenidos en la BD.

2.4 Definición de los requisitos.

La definición de los requisitos de un software es una tarea de suma importancia, estos permiten definir una interfaz de usuario para el sistema, enfocada en las necesidades y metas del usuario, además provee a los desarrolladores un mejor entendimiento de los requisitos del sistema. Ofrece claves específicas sobre qué se debe cumplir para dar satisfacción a lo deseado por el cliente, sirve de guía en la identificación de qué y cómo debe de funcionar cada segmento de la herramienta en desarrollo y constituye una fase fundamental para un avance con alto nivel de calidad.

2.4.1 Definición de los requisitos funcionales.

Los requisitos funcionales son las funcionalidades que el sistema debe incorporar para dar solución al problema identificado, estas son especificadas mediante los requisitos funcionales del sistema que son capacidades o condiciones que este debe cumplir y que definen el comportamiento interno del software. Se utilizan para describir los servicios que se espera que el sistema cumpla para satisfacer las necesidades del usuario. Además aporta una visión más detallada de lo que se va a implementar.

A continuación se muestran los requisitos funcionales que se determinaron:
RF1: Establecer conexión con la BD.
RF1.1: Realizar clonación de BD.
RF1.2: Mostrar las tablas de la BD.
RF1.3: Cargar registro del enmascaramiento.
RF2: Seleccionar atributos a los que se le va a aplicar el proceso de enmascarado.
RF3: Enmascarar datos en la BD clon.
RF3.1: Visualizar resultados del proceso de enmascarado.
RF3.2: Guardar registro de la operación.

2.4.2 Definición de los requisitos no funcionales.

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Estos solo describen los atributos con que debe contar el sistema y son las propiedades que hacen al producto atractivo, usable, rápido y confiable.

Estos pueden ser clasificados en diferentes tipos, entre los que podemos hallar: Software, Hardware, Seguridad, Soporte, entre otros más. Teniendo en cuenta lo estudiado anteriormente, se definen los siguientes requisitos no funcionales:

Seguridad.

Usabilidad.

RNF1: El sistema estará diseñado de manera que los usuarios adquieran las habilidades necesarias para explotarlo en un tiempo reducido:

- Usuarios normales: 5 días.
- Usuarios avanzados: 2 días.

Confiabilidad.

RNF2: Se mantendrá seguridad y control a nivel de base de datos, garantizando acceso a estas solo al usuario que conozca la contraseña del gestor de base de datos.

RNF3: Se creará una BDD a partir de la BDO la cual será enmascarada logrando tener datos similares a los de la BDO pero no reales, garantizando la integridad y confidencialidad de los datos originales.

RNF4: La BDO no será eliminada del gestor de base de datos.

Eficiencia.

RNF5: La aplicación adoptará buenas prácticas de programación para incrementar el rendimiento en operaciones costosas para la máquina virtual como la creación de objetos.

Soporte.

RNF6: Las notificaciones de las deficiencias detectadas en la aplicación desplegada deberán realizarse por escrito.

RNF7: Una vez notificada la deficiencia detectada en la aplicación desplegada, el equipo de desarrollo deberá solucionarla en un período de 7 días.

Restricciones de diseño.

RNF8: La capa de la vista contendrá todas las vistas y la lógica de la presentación.

RNF9: La capa de acceso a datos contendrá las entidades y los objetos de acceso a datos correspondientes a las mismas.

RNF10: Las interfaces se realizarán siguiendo el estándar definido por el proyecto.

Interfaz.

RNF11: Las ventanas del sistema contendrán los datos claros y bien estructurados para facilitar la interpretación correcta de la información.

RNF12: La entrada de datos incorrecta será detectada claramente e informada al usuario.

Requerimientos de hardware.

RNF13: Las estaciones de trabajo deben tener como mínimo 512 Mb de memoria RAM y un microprocesador de 2.0 GHz.

RNF14: Los servidores de base de datos deberán tener: Procesador Intel®Xeon® 5140 Dual-Core 4GB de memoria y 2x72GB de disco y sistema operativo Linux.

RNF15: Los servidores de aplicaciones deberán tener: Procesador Intel® Xeon® 5140 Dual - Core 4GB de memoria y 2x72GB de disco y sistema operativo Linux.

Requerimientos de software.

RNF16: La aplicación debe correr en sistemas operativos Windows XP, Unix y Linux Ubuntu 10.4, utilizando la plataforma JAVA (Java Virtual Machine, JBoss AS y PostgreSQL).

RNF17: La aplicación deberá disponer de un navegador web, estos pueden ser IE7, Opera 9, Google chrome 1 y Firefox 6.

2.4.3 Modelo de casos de uso del sistema.

Los requisitos o capacidades internas de la herramienta a implementar han sido expresados como casos de uso, teniendo en cuenta que son adecuados para describir actividades atómicas que ofrecen un resultado definido para cada acción que realice el usuario.

El modelo de caso de uso del sistema documenta el comportamiento de la herramienta desde el punto de vista del usuario, permitiendo representar funciones que se desea que realice (Casos de uso), el entorno del sistema (actores) y las relaciones entre ellos. Aunque la parte más visible de este modelo es el diagrama de caso de uso, es acompañado de una descripción textual de cada uno de los casos de uso.

Los actores del sistema no forman parte del mismo, sino que representan elementos que interactúan con él. Estos elementos son nombrados roles que pueden ser desempeñados por una o varias personas, un equipo o un sistema automatizado. Un actor puede introducir o recibir información del sistema.

El actor identificado en el desarrollo de la herramienta se muestra en la tabla 2.2:

Actor	Descripción
Administrador de BD	Es la persona que interactúa con la aplicación, a través de la cual realiza el proceso de enmascaramiento. Es el encargado de crear la conexión a la BD mediante la especificación del puerto de conexión y nombre de la BD, una vez creada la conexión, selecciona lo que va a enmascarar.

Tabla 2.2: Descripción de actores del sistema.

2.4.4 Diagrama de caso de uso del sistema.

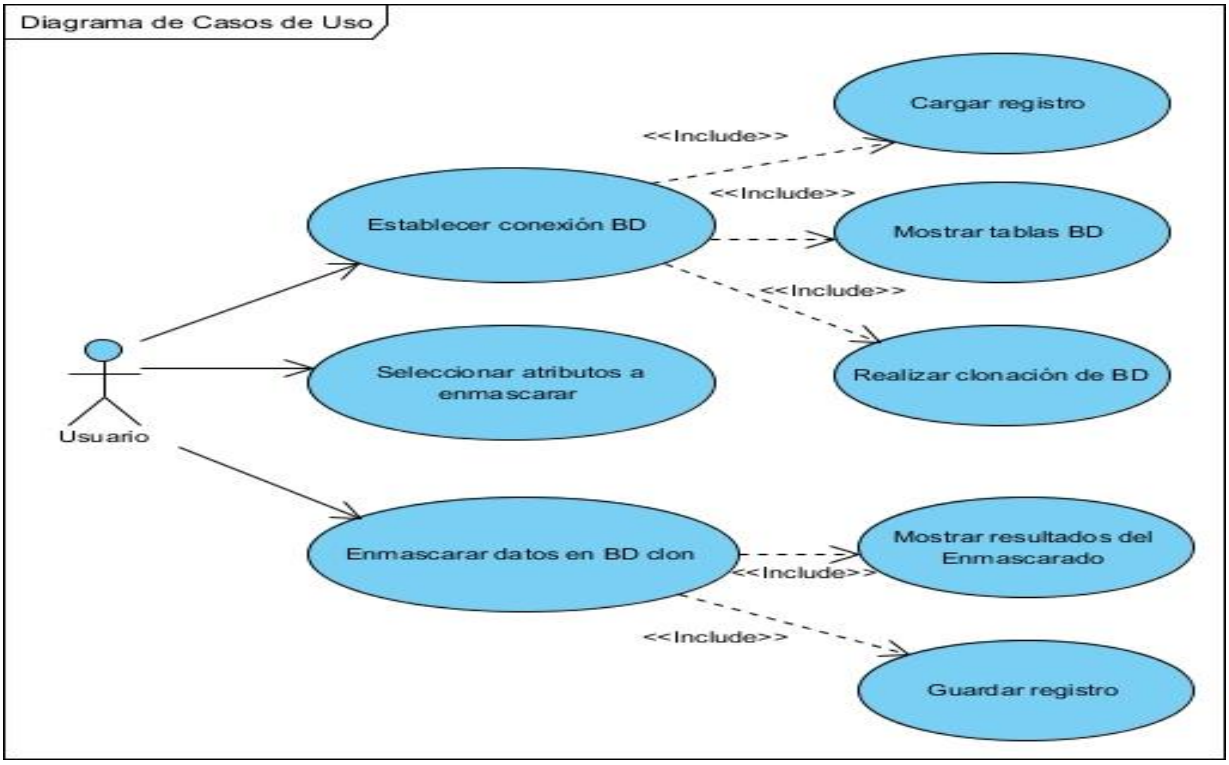


Figura 2.3: Diagrama de caso de uso del sistema.

2.4.5 Descripción textual de los casos de uso.

La descripción textual de los casos de uso es un artefacto fundamental en el desarrollo de todo sistema informático, brinda un mayor entendimiento del flujo de evento que es realizado en la interacción actor-sistema y contiene una descripción textual de todas las maneras que los actores previstos podrían trabajar con el software.

A continuación en las tablas 2.3, 2.4 y 2.5 se muestra la descripción textual de los casos de uso identificados:

Caso de uso	
CU 1	Establecer conexión BD.
Propósito	Establecer conexión BD.
Actores	Administrador de BD.
Resumen	El caso de uso inicia cuando el actor introduce el nombre del servidor, contraseña, usuario y el puerto por el cual conectara con la BD. El caso de uso culmina luego de creada la conexión mostrando una lista con los esquemas y tablas contenidas en la BD.
Referencias	RF1

Tabla 2.3: Descripción del CU Establecer conexión BD.

Caso de uso	
CU 2	Seleccionar atributos a enmascarar.
Propósito	Seleccionar los atributos que se desean enmascarar.
Actores	Administrador de BD.
Resumen	El actor selecciona los atributos que no desea enmascarar y que se encuentran en la lista a ser enmascarada y los elimina de la misma.
Referencias	RF2

Tabla 2.4: Descripción del CU Seleccionar atributos a enmascarar.

Caso de uso	
CU 3	Enmascarar datos en BDC.
Propósito	Enmascarar la BD resultante de clonar la BDO.
Actores	Administrador de BD.
Resumen	El caso de uso inicia cuando el usuario selecciona la opción enmascarar datos y se enmascara la información en la BD clon.
Referencias	RF6

Tabla 2.5: Descripción del CU Enmascarar datos en BD clon.

Conclusiones Parciales.

En este capítulo se elaboró una descripción detallada del modelo de dominio para mostrar los objetos y conceptos fundamentales de la herramienta a desarrollar. Con el apoyo de diagramas se elaboró una representación detallada de cada proceso, lo que posibilita un mejor entendimiento del problema y una rápida identificación de las funcionalidades y propiedades del sistema, a partir de las cuales se definió el modelo de caso de uso del sistema con sus diagramas y especificaciones de casos de uso, permitiendo dar paso al capítulo 3, fase de análisis y diseño de la herramienta.

Capítulo 3: Análisis y diseño del sistema.

En el presente capítulo se profundiza en el análisis y diseño del sistema. Se tienen como objetivos fundamentales: transformar los requerimientos definidos a una propuesta de diseño que será la guía a seguir para la implementación de la herramienta; evolucionar hacia una arquitectura del software robusta y flexible ante la aplicación de nuevos cambios.

3.1 Descripción de la arquitectura.

La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema. Consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software.

El sistema que se propone presenta una arquitectura basada en el patrón arquitectónico Modelo Vista Controlador (MVC), abordado en el capítulo 1.

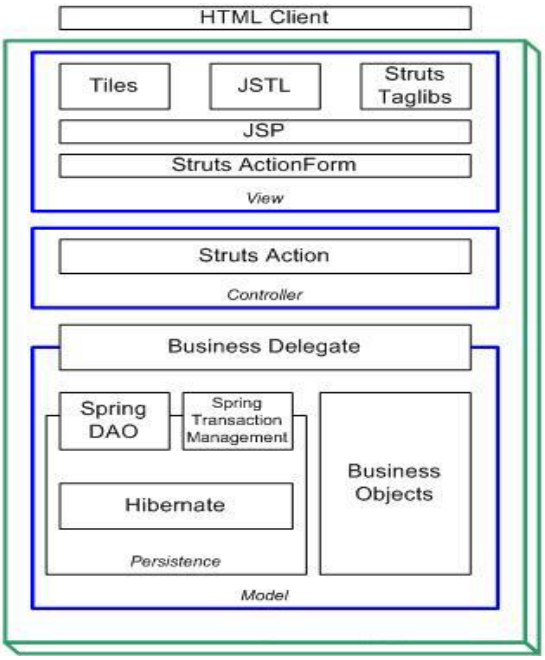


Figura 3.1 Comportamiento del patrón Modelo-Vista-Controlador con JSF.

Fuente: http://oness.sourceforge.net/docbook/exploracion.html

El uso del framework JSF permite contar con la implementación de dicho patrón, brindándole a la aplicación la posibilidad de tener una separación clara entre cómo se muestra la información al usuario, cómo se manejan las acciones que el usuario desea hacer sobre el sistema y cómo se realizan estas acciones como se muestra en la figura 3.1.

3.2 Modelo del diseño.

Mediante el modelo de diseño se hace un refinamiento del proceso de análisis anteriormente realizado. Para ello se tienen en cuenta los requisitos no funcionales del sistema ya que el principal propósito del modelado del diseño es crear un plano del modelo de implementación. También se define la arquitectura del sistema. Los casos de uso son realizados por las clases del diseño y sus objetos, a partir de los cuales se forma el diagrama de clases del diseño.

Para la elaboración del modelo de diseño, se define una estructura de paquetes que permite dividir el sistema en fragmentos manejables para su futura implementación. Se emplea el criterio de empaquetamiento por proceso, siguiendo la estructura de procesos definidos en el sistema.

3.2.1 Diagrama de paquetes.

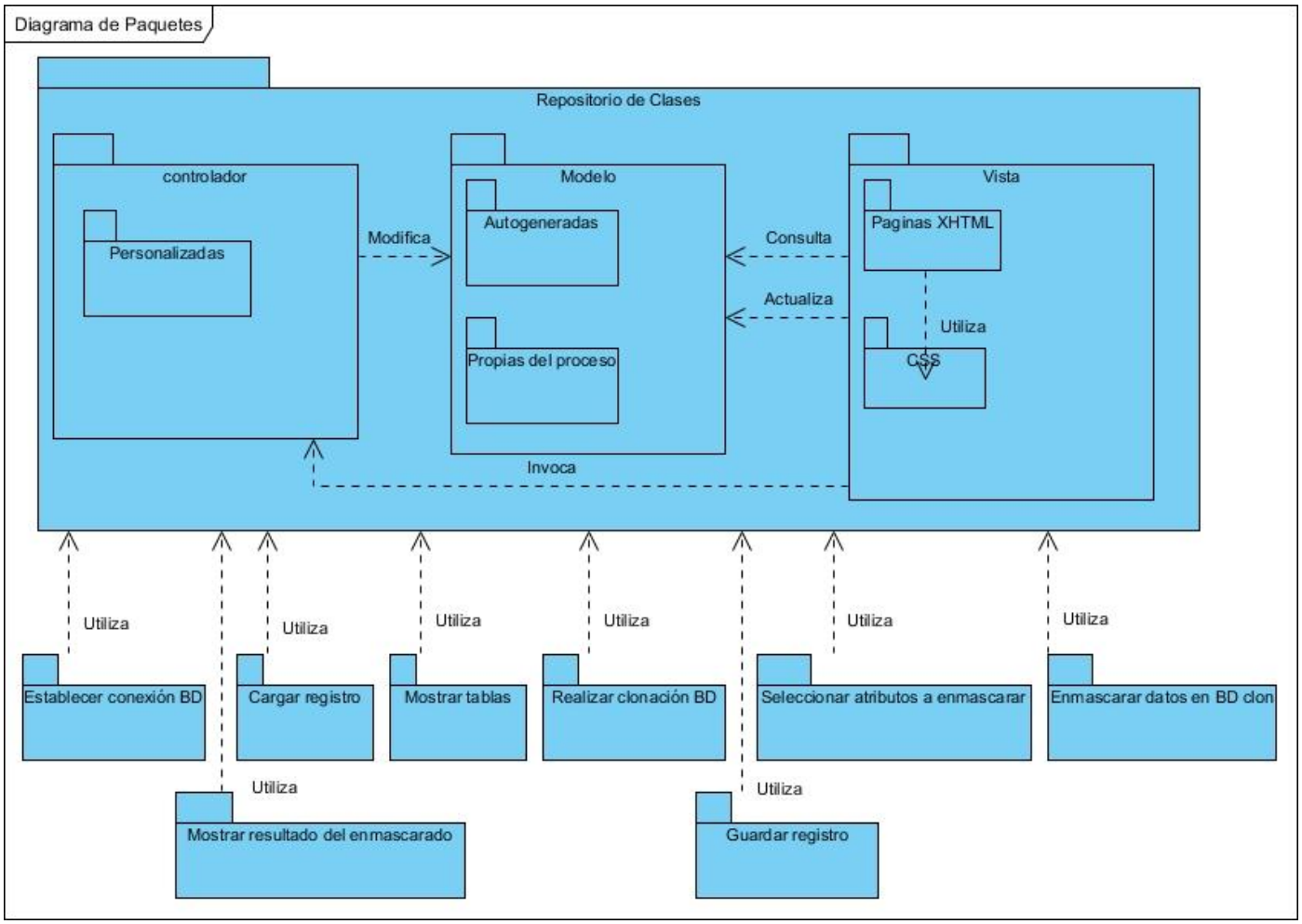


Figura 3.2 Diagrama de paquetes.

Como se refleja en el diagrama de paquetes de la figura 3.2, en el sistema se implementa el proceso Establecer conexión BD. Para este proceso se modela un diagrama de clases del diseño como se muestra en el ejemplo de la figura 3.3, y por cada escenario un diagrama de interacción. Como diagrama de interacción fue seleccionado el de secuencia ejemplificado en la figura 3.4.

En el **Diagrama de clases del diseño** se exponen un conjunto de interfaces, colaboraciones y sus relaciones. Se utilizan para modelar la vista de diseño estática de un sistema. Permiten visualizar, especificar y documentar modelos estructurales.

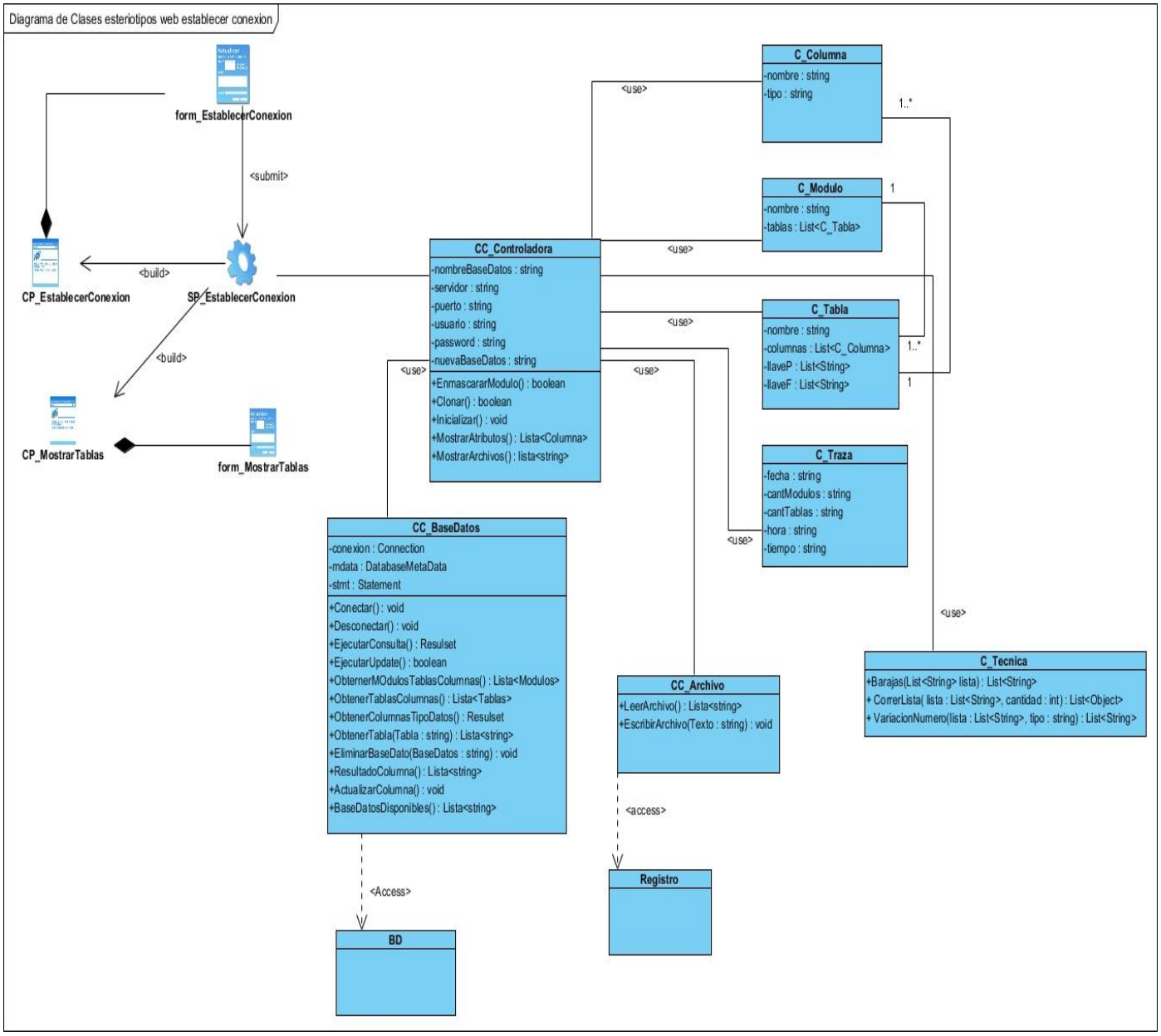


Figura 3.3 Diagrama de clases del diseño: Establecer conexión BD.

El **Diagrama de interacción:** muestra gráficamente cómo los objetos se comunican entre sí a fin de cumplir con los requerimientos. Este se emplea para modelar los aspectos dinámicos de un sistema. Dentro de los diagramas de interacción se definen los **Diagramas de secuencia** en el que se destaca la ordenación temporal de los mensajes como se muestra en la figura 3.4.

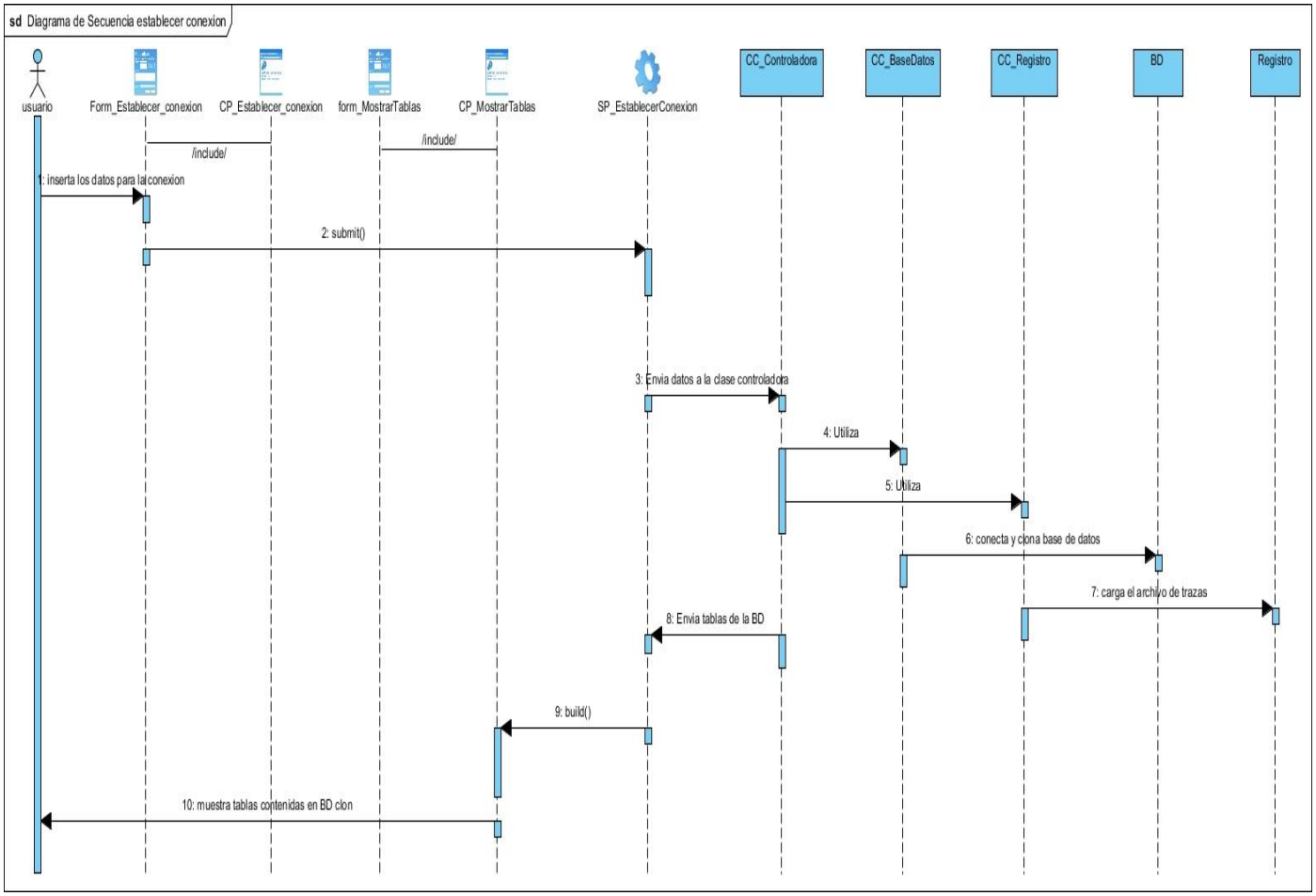


Figura 3.4 Diagrama de secuencia: Establecer conexión BD.

Las clases del diseño están agrupadas en:

Páginas Servidoras: Están compuestas por componentes Facelets, RichFaces, JSF, Seam UI, así como código HTML. Todo este código será ejecutado en el servidor web, generando páginas clientes que pueden ser representadas por los navegadores web.

Páginas Clientes: Están compuestas por código HTML, CSS, JavaScript. Son interpretadas por los navegadores web presentándole al usuario la interfaz con la que puede interactuar con el sistema.

Formularios: Un formulario HTML es una sección de un documento enmarcado entre tags <form> y que puede contener elementos especiales llamados controles, y rótulos en esos controles. Los usuarios

normalmente completan un formulario modificando sus controles, y lo envían al servidor donde estos son procesados. Es una manera de obtener en el servidor información entrada por el usuario en el cliente.

Controlador: La clase controladora implementa la lógica del negocio que se está informatizando, generalmente cada una de estas se encarga de la implementación de un caso de uso o un proceso en dependencia de la complejidad de los mismos.

Conclusiones Parciales.

Como resultado del estudio realizado en este capítulo se identificó el patrón de arquitectura a emplearse en la solución. También se elaboró el diagrama de clases del diseño para cada caso de uso del sistema, mediante los cuales se representaron las relaciones que se establecen entre las clases. Se identificaron las clases fundamentales que deben ser definidas para que el sistema funcione satisfactoriamente

Capítulo 4: Implementación y Pruebas.

La fase de implementación es la consecuencia de la fase de diseño. En este capítulo se describen las clases y subsistemas implementados en términos de componentes. Se muestra el Diagrama de Despliegue como parte del Modelo de Implementación, que indica la distribución física de la solución implementada. Además se proporciona una detallada explicación de dicha solución.

4.1 Modelo de implementación.

El modelo de implementación describe cómo los elementos del modelo de diseño se implementan en términos de componentes. Describe también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y en el lenguaje de programación utilizado, y cómo dependen los componentes unos de otros.

4.1.1 Diagrama de componentes.

Un Diagrama de Componentes es, como su nombre lo indica, un esquema que muestra las interacciones y relaciones de los componentes de un modelo; entendiéndose como componente a una clase de uso específico, que puede ser implementada desde un entorno de desarrollo, ya sea de código binario, fuente o ejecutable; dichos componentes poseen tipo, que indica si pueden ser útiles en tiempo de compilación, enlace y ejecución.

Este tipo de diagrama se representa mediante componentes unidos a través de relaciones de dependencia (generalmente de compilación)(37).

El diagrama de componentes de la herramienta está dividido en tres paquetes. En el paquete Vista se encuentran los componentes **enmascarar** y **enmascarar.page** para la interacción con los usuarios, dichos componentes hacen uso de **ajas4fsj.jar**, **richfaces.jar**, **jsf-faces.jar**, **seam-model.jar** y **jb-seam-UI.jar**, los cuales no son más que librerías que brindan facilidades para el diseño de las interfaces. En Controlador se encuentra el componente **jboss-seam.jar** el cual permite la integración entre la Vista y el Controlador, además se encuentran reflejadas las clases encargadas de la gestión y control de las funcionalidades. En Modelo se hace una representación de las entidades que sufrirán cambios luego de

realizado el proceso de enmascaramiento. En la figura 4.1 se muestra el diagrama de componentes correspondiente a la herramienta.

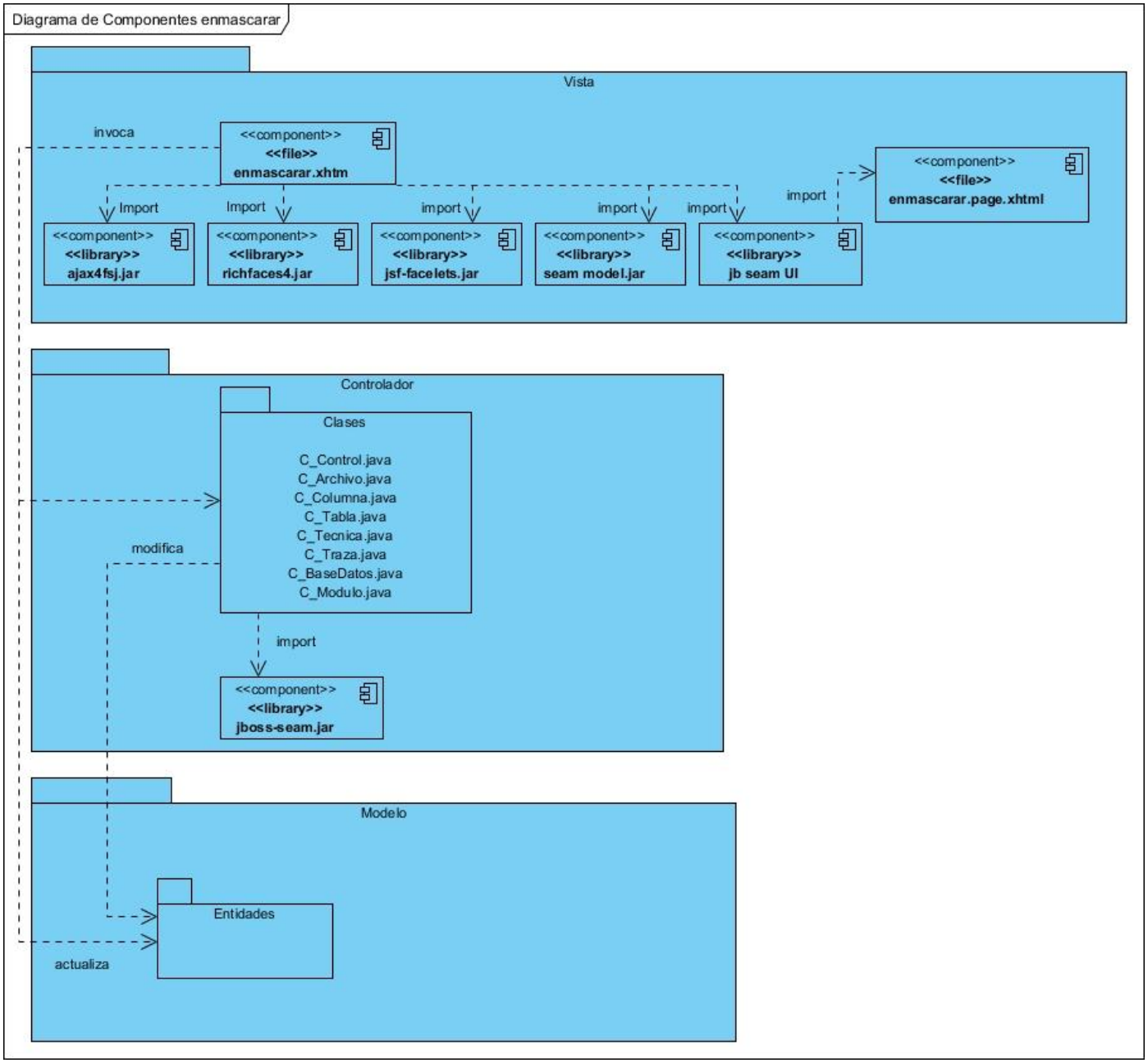


Figura 4.1 Diagrama de componentes.

4.1.2 Diagrama de despliegue.

Los Diagramas de Despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Un nodo es un recurso de ejecución tal como un computador, un dispositivo o memoria (38). Los estereotipos permiten precisar la naturaleza del equipo:

- ✓ Dispositivos
- ✓ Procesadores
- ✓ Memoria

La aplicación está distribuida en tres nodos como se muestra en la figura 4.2: uno representa las estaciones de trabajo (PC) clientes las cuales pueden tener instalado cualquier sistema operativo, el segundo se corresponde con el servidor de aplicaciones JBoss AS 2.0 que está conectado punto a punto con el tercer nodo que es el servidor de datos Postgresql-server 8.4.

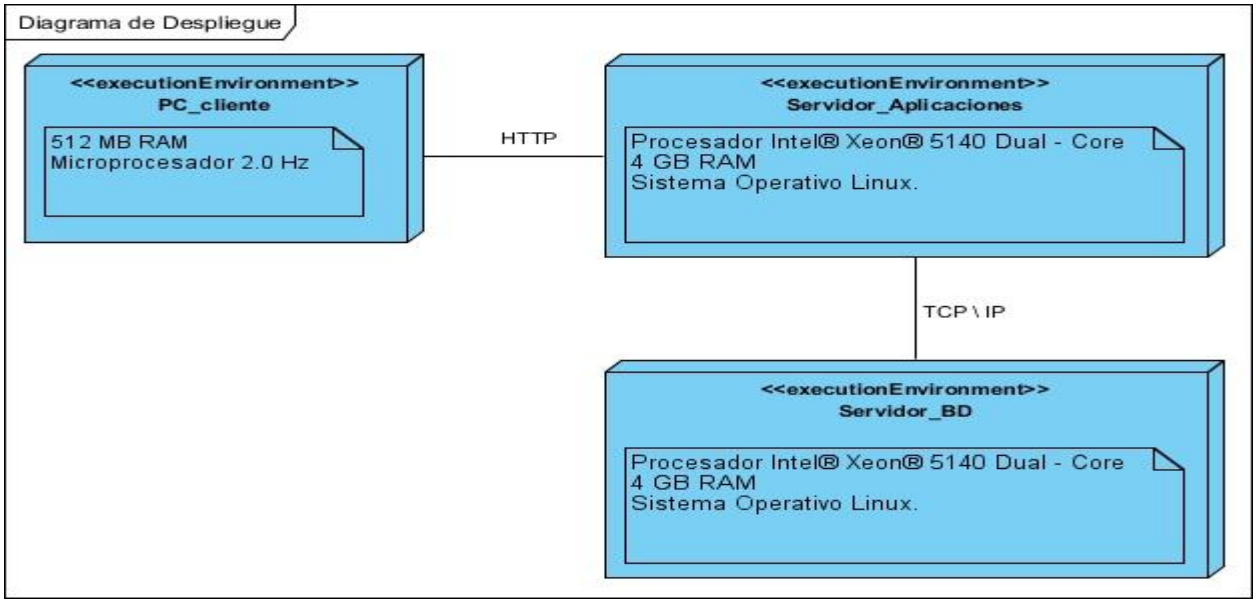


Figura 4.2 Diagrama de despliegue.

4.2 Tratamiento de errores.

Durante el tiempo de ejecución de un sistema pueden fracasar diferentes rutinas; es esto a lo que comúnmente se le llama excepción. El tratamiento de excepciones es un paso primordial para obtener un sistema óptimo, puesto que garantiza la integridad y confidencialidad de la información que se maneja en él.

En la herramienta propuesta, el control de las excepciones se lleva a cabo a toda porción de código, donde pueda surgir alguna situación inesperada, especialmente donde se ejecutan sentencias que manipulan los datos que viajan desde y hacia la base de datos como se muestra en la figura 4.3. También se controlan los errores que pueden surgir en la validación de datos provenientes de la interfaz de usuario, puesto que encierran una lógica compleja en cierta medida.

```
public boolean EjecutarUptade(String Consulta)
{
    try
    {
        stmt = conexion.createStatement();
        stmt.executeUpdate(Consulta);
    }
    catch (SQLException e)
    {
        System.out.println(e.getMessage());
    }
    return true;
}
```

Figura 4.3 Manejo de Excepciones.

4.3 Seguridad.

Para analizar la seguridad de un sistema se debe pensar en la forma en que el mismo pudiera sufrir determinada pérdida o daño, para lo cual es necesario identificar las debilidades del sistema. Las funcionalidades del sistema encargadas de la seguridad son: iniciar y cerrar sesión de trabajo.

Para iniciar sesión de trabajo el Administrador de Base de Datos (ABD) debe acceder al sistema e insertar servidor, nombre de base de datos, usuario y contraseña del gestor de base de datos, así como el

puerto si es necesario (estos datos solamente deben de ser conocidos por el ABD). El sistema verifica que los datos sean válidos y el usuario tendrá acceso a la aplicación.

4.4 Estrategias de codificación. Estándares y estilos de codificación.

Los estándares de codificación son reglas que se aplican para logar uniformidad en el código producido por un grupo de desarrollo de un sistema. Estos reducen perceptiblemente el riesgo de que los desarrolladores introduzcan errores. Los estándares de codificación no destapan problemas existentes, evitan más bien que los errores ocurran, lo que permite obtener un código de alta calidad.

Usar técnicas de codificación sólidas y realizar buenas prácticas de programación, con vistas a generar un código de alta calidad, es de gran importancia para la calidad del *software*. La aplicación de estándares de codificación además posibilita que el software que se obtiene sea fácil de comprender y de mantener en el tiempo(39).

Variables y constantes	
Aspectos generales	El nombre empleado, debe permitir que con solo leerlo se conozca el propósito de la misma.
Clases y Objeto	
Apariencia de clases y objetos	Primera letra en mayúscula. Los nombres de las clases deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCase. Ejemplo: MiClase ().
Apariencia de atributos	Primera letra en minúscula. El nombre que se le da a los atributos de las clases debe comenzar con la primera letra en minúscula, en caso de que sea un nombre compuesto se empleará notación camelCase.
Apariencia de los métodos	Apariencia de los métodos. Para nombrar los métodos se debe tratar de utilizar verbos que denoten la acción que hace el método. Se empleará notación PascalCase. Ejemplo: ObtenerTablas ().
Aspectos generales	El nombre empleado para las clases, objetos, atributos y

	métodos debe permitir que con solo leerlo se conozca el propósito de los mismos.

Tabla 4.1 Estándares y estilos de codificación.

4.5 Descripción de los métodos Barajas y Variación de números.

Barajas.

El método Barajas (List<String> lista): List<String> desplaza la lista mediante un indicador que es creado por un número aleatorio comprendido entre 0 y el tamaño de la lista, luego se crean dos variables enteras que representan posiciones en la lista (las variables van tomando valores aleatorios comprendidos entre 0 y el tamaño de la lista) se intercambian los datos de la lista en las posiciones especificadas por las variables, finalmente el método devuelve una nueva lista para actualizar la columna a la que pertenece.

Variación de números.

El método VariacionNumero (List<String> lista, String tipo): List<String> crea un número aleatorio comprendido entre 0 y el tamaño de la lista, luego se multiplica el valor de la lista por el número aleatorio creado y el resultado se divide entre la posición que ocupa en la lista, esto garantiza que el nuevo número nunca quede igual que el original.

4.6 Descripción del uso de la arquitectura de enmascaramiento.

La arquitectura ***In-Situ*** plantea que a partir de la BDO se crea una BDD, la cual será la encargada de contener los datos enmascarados, para esto se debe de clonar la BDO usando algún software que realice dicho proceso, por lo que en la herramienta dataHISMask se implementa una funcionalidad que realice la clonación de una base de datos especificada, conjuntamente los componentes de la BDO se obtienen en un objeto de tipo Módulo (el cual contiene una lista de tablas y un nombre, cada tabla tiene una lista de columnas (atributos de la tabla), una llave primaria y nombre, las columnas tienen un nombre y tipo de dato) brindando la posibilidad de seleccionar lo que se desea enmascarar y cuándo enmascarar sin afectar los datos de la BDO. AL realizar el proceso de enmascaramiento solo se enmascararán en la BDC ó BDD la información que contenga el objeto de tipo Módulo creado, que son los datos que el usuario ha considerado que son sensibles o confidenciales.

4.7 Funcionamiento del Sistema.

En la figura 4.4 se muestra una imagen de la interfaz mostrada al usuario para que inserte los campos necesarios para establecer conexión con una base de datos que desee enmascarar. Los campos a introducir son:

Servidor: El usuario escribirá los datos del servidor PostgreSQL al que quiere referirse.
Base de Datos: El usuario introduce el nombre de la base de datos que desea enmascarar y que se encuentra en el servidor especificado.
Nueva BD: En este campo el usuario introduce un nombre el cual adoptará la BDD la que se creará en el servidor especificado.
Puerto: Este campo es opcional, se inserta el puerto por donde se conectará la familia de protocolos TCP/IP, de no especificar ninguno tomará el puerto por defecto de PostgreSQL (5432).
Usuario: Se especifica el usuario del servidor de base de datos.
Contraseña: Se introduce la contraseña del servidor de base de datos, la cual solo puede ser conocida por el Administrador de base de datos.
Desconectar usuarios: Este campo es opcional, se selecciona si la base de datos está siendo utilizada por algún usuario en el momento que se quiere establecer la conexión que es cuando se realiza la clonación, de ser así se desconecta a todos los usuario conectados a la base de datos. Estos usuario podrán conectarse nuevamente luego de terminado el proceso de clonación.

Si se introduce algún campo incorrectamente el sistema muestra un mensaje de error, sugiriendo cual puede ser el error.

Figura 4.4: Establecer Conexión.

En la figura 4.5 se muestra la imagen de la interfaz de usuario mostrada luego de establecida la conexión, se refleja una lista de trazas de enmascaramiento realizadas anteriormente a la base de datos a la que está conectado.

Figura 4.5: Página principal.

Inicialmente todos los módulos del alas HIS están en la lista que será enmascarada, si el usuario quiere eliminar algún módulo de dicha lista solamente tiene que seleccionar en el combobox “Quitar módulo de la lista de enmascaramiento” el módulo que desee eliminar. Si lo que quiere es eliminar alguna tabla de un

módulo en específico desplegará de la lista de módulos el que desee y seleccionará la tabla que quiere eliminar, mostrando una interfaz similar a la de la figura 4.6, donde solo tendrá que dar clic en el botón para eliminar la tabla.

Si lo que quiere es eliminar un atributo en específico de alguna tabla, luego de seleccionada la tabla solo tiene que dar clic en el botón para eliminar el atributo.

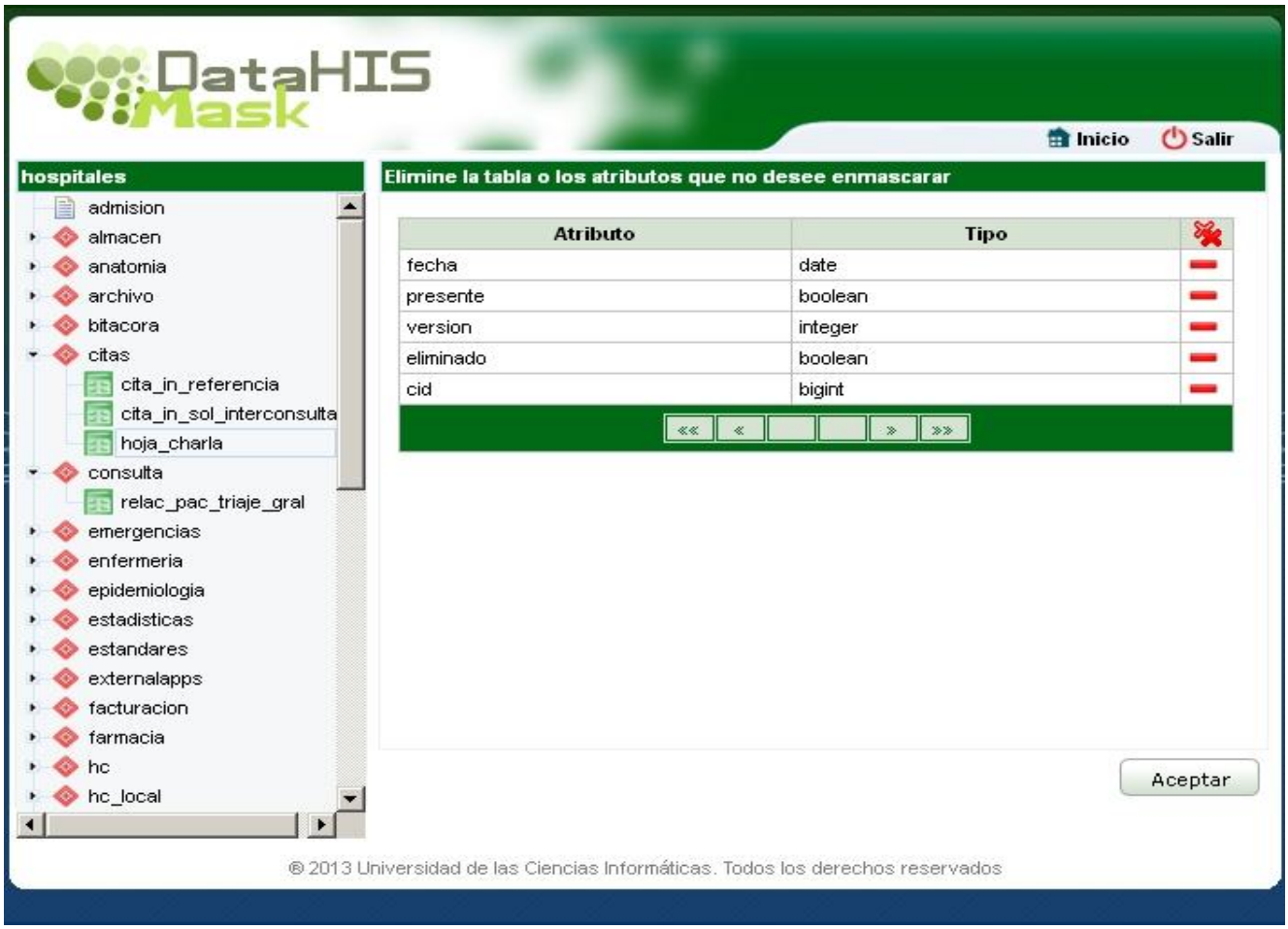


Figura 4.6: Seleccionar atributos a enmascarar.

Después de haber seleccionado los datos que quiere enmascarar solo le queda dar clic en el botón Aceptar, y luego de terminado el proceso de enmascaramiento se mostrará una interfaz que muestra el resultado de alguna tabla seleccionada aleatoriamente, donde se evidencian los cambios efectuados dependiendo de la técnica usada como se muestra en la figura 4.7.

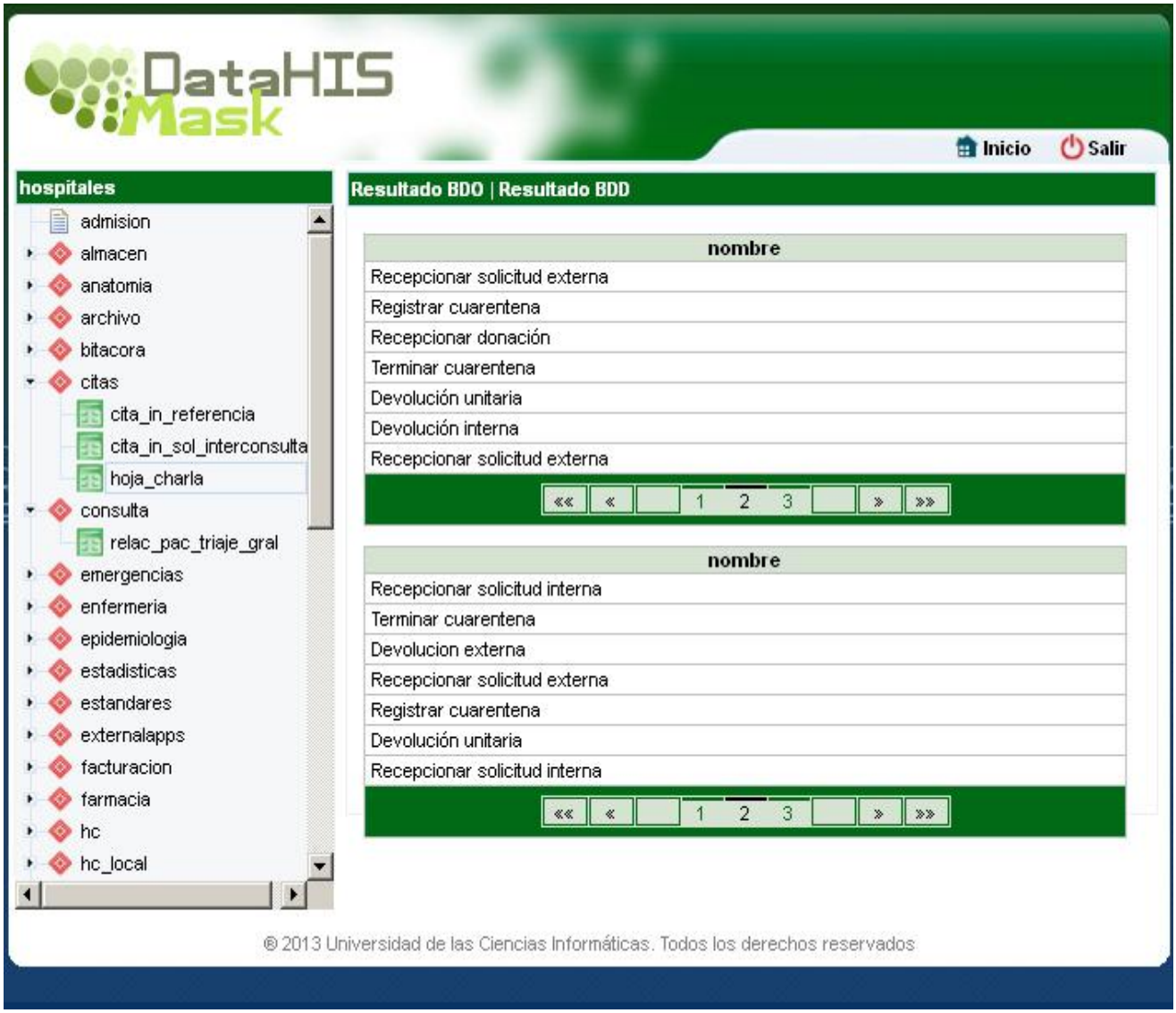


Figura 4.7: Resultados del enmascaramiento.

4.8 Pruebas.

Esta sección tiene como objetivo describir el proceso de las pruebas realizadas a la herramienta implementada, con el propósito de validar la solución propuesta y encontrar fallas no detectadas hasta entonces.

Para determinar si el resultado de un producto es el deseado, se hace necesario hacerle una serie de pruebas que determinarán la calidad con la que este saldrá. A continuación se muestran las técnicas a emplear de validación y pruebas al sistema, para garantizar que las tareas realizadas en el diseño y los requisitos cumplen con las normativas planteadas y se encarga de verificar que el producto funcione como se diseñó y que cumple con todas las necesidades expuestas por el proyecto alas HIS.

4.8.1 Estrategias de prueba.

Para el desarrollo de un software de calidad, la prueba es una de las tareas más importantes, las mismas evalúan el producto y permiten determinar si cumple con el objetivo previsto, por lo que es necesario diseñar un plan de pruebas que se adapte y sea coherente con la metodología de desarrollo, que proporcione un enfoque de fácil acceso a la estructura para verificar los requisitos y cuantificar su rendimiento, y que identifique las diferencias entre los resultados previstos y los reales. Es el proceso por el que se evalúa la correcta interpretación y aplicación de los requisitos especificados. Cualquier producto de ingeniería se puede probar de dos formas, estas son mediante pruebas de caja blanca y las pruebas de caja negra, explicadas a continuación(40).

Pruebas de caja negra.

También conocidas como Pruebas de Comportamiento, estas pruebas se basan en la especificación del programa o componente a ser probado para elaborar los casos de prueba. El componente se ve como una “Caja Negra” cuyo comportamiento solo puede ser determinado estudiando sus entradas y las salidas obtenidas a partir de ellas. No obstante, como el estudio de todas las posibles entradas y salidas de un programa sería impracticable, se selecciona un conjunto de ellas sobre las que se realizan las pruebas. Para seleccionar el conjunto de entradas y salidas sobre las que trabajar, hay que tener en cuenta que en todo programa existe un conjunto de entradas que causan un comportamiento erróneo en el sistema, y como consecuencia producen una serie de salidas que revelan la presencia de defectos. Entonces, dado que la prueba exhaustiva es imposible, el objetivo final es, encontrar una serie de datos de entrada cuya

probabilidad de pertenecer al conjunto de entradas que causan dicho comportamiento erróneo sea lo más alto posible. Para confeccionar los casos de prueba de Caja Negra existen distintos criterios.

Algunos de ellos son:

Particiones de Equivalencia: Es una técnica de prueba de caja negra que divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
Análisis de Valores Límite: Consiste en probar la habilidad del programa para manejar datos que se encuentran en los límites aceptables.

Análisis Causa-Efecto: Es una técnica que permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

Las pruebas de caja negra son las encargadas de comprobar que cada función del software es operativa. Se llevan a cabo sobre la interfaz del software y tratan de demostrar que las entradas se manejan de forma adecuada y que se produce el resultado esperado(40).

Se define como técnica a utilizar en la aplicación de las pruebas de caja negra, la **partición de equivalencia**, pues reduce el número total de casos de prueba a desarrollar, obteniéndose así en un conjunto manejable que evalúe el software. Es una de las más efectivas pues permite examinar los valores válidos e inválidos de las entradas existentes en el software y descubre de forma inmediata una clase de errores.

4.8.2 Casos de prueba.

Para desarrollar software de calidad, los casos de prueba son de vital importancia. Un buen caso de prueba, es aquel que tiene alta probabilidad de demostrar un error no descubierto hasta entonces. Un caso de prueba bien diseñado tiene gran posibilidad de llegar a resultados más fiables y eficientes (40).

Un caso de prueba es el conjunto de entradas de pruebas, condiciones de ejecución y resultados esperados desarrollados para cumplir un objetivo en particular. Se muestran a continuación los casos de prueba, en los que se describen las variables válidas e inválidas de entrada y salida en el caso de uso. En

la Tabla 4.5.2.1, se presentan las distintas variables. La Tabla 4.5.2.2, representa las distintas secciones en el caso de uso, además de los escenarios por sección y la descripción de la funcionalidad.

Caso de prueba: Establecer conexión.

Descripción de las variables

No.	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Servidor	Un cuadro de texto.	No	Admite letras y números, se puede escribir la dirección del servidor.
2	Base de datos	Un cuadro de texto.	No	Admite letras y números, se puede escribir el nombre de la BD.
3	Nueva BD	Un cuadro de texto.	No	Admite letras y números, se puede escribir el nombre de la BDC.
4	Puerto	Un cuadro de texto.	Sí	Admite solo números, se escribe el número del puerto.
5	Usuario	Un cuadro de texto.	No	Admite letras y números, se puede escribir el nombre de usuario.
6	Contraseña	Un cuadro de texto.	No	Admite letras y números, se puede escribir la contraseña.
7	Desconectar usuarios.	Checkbox	Sí	Desconecta a usuarios que estén conectados a la BDO.

Tabla 4.2 Descripción de las variables

Escenario	Descripción	S	BD	NBD	P	U	C	DU	Respuesta del sistema	Flujo central
EC 1.1 Establecer Conexión	Establece la conexión con la BD PostgreSQL.	V	V	V	V	V	V	V/F	Clona la BDO y establece la conexión mostrando la interfaz principal.	1-El usuario introduce los datos. 2-El usuario da clic en la opción

EC1.2 Existen datos incompletos	Existen campos incompletos.	I	I	V	V	V	V	V/F	Muestra un mensaje de error de conexión	aceptar. 3- El sistema clona la BD y muestra la interfaz principal con las tablas en la BDD.
EC 1.3 Existen datos incorrectos	Existen campos incorrectos	I	I	V	V	I	V	V/F	Muestra mensaje de error de conexión	

Tabla 4.3 Matriz de datos

Caso de prueba: Seleccionar atributos a enmascarar.

Descripción de las variables

No.	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Atributos que no desea enmascarar	Checkbox	si	Se seleccionan los atributos que no se desean enmascarar (Inicialmente todos están seleccionados).
2	Inicio	buttom	Si	Sale hacia la interfaz principal.
3	Salir	buttom	Si	Sale de la herramienta.

Tabla 4.4 Descripción de las variables.

Escenario	Descripción	ANE	I	S	Respuesta del sistema	Flujo central
EC 1.1 Seleccionar atributos que no desea enmascarar	Selecciona los tributos que no serán enmascarado	V	NA	NA	Elimina los atributos de la lista a enmascarar.	1-El usuario selecciona los atributos que no desea enmascarar.

EC 1.2Ir al inicio de la aplicación	Selecciona la opción ir a inicio.	NA	V	NA	Se muestra la interfaz de inicio de la herramienta	2-El sistema elimina los atributos automáticamente.
EC 1.3Salir	Selecciona la opción salir.	NA	NA	V	Sale de la herramienta.	

Tabla 4.5 Matriz de datos.

Además de las pruebas antes expuestas se realizaron otras pruebas para probar el rendimiento de la herramienta a la hora de realizar la clonación y de enmascarar los datos.

Proyecto	Cant. Módulos	Cant. Tablas	Tiempo de clonación(Min)	Tiempo de enmascarado(H)
alas HIS	37	2936	20 mn	02h:04mn:56s:645mls
Synta	5	49	00mn:35s	00h:00mn:03s:775mls

Tabla 4.6 Resultados de Prueba de Ejecución.

Estos tiempos fueron obtenidos realizando los procesos de clonación y enmascaramiento en una máquina de proyecto, con 1GB de RAM y un microprocesador Pentium(R) D CPU 3.00GHz, teniendo además la de BD montada en un servidor local en la misma PC de prueba.

4.8.3. Resultado de las pruebas.

La prueba de caja negra mediante la técnica **partición de equivalencia** fue realizada de manera estricta a partir de los casos de prueba elaborados. Realizar esta prueba permitió detectar, documentar y solucionar los errores existentes en el sistema implementado. Se validó que la herramienta desarrollada satisface los requisitos identificados para el que se desarrolló. El período de pruebas permitió obtener una aplicación que responde correctamente, cumpliendo en su totalidad con los objetivos planteados para la arquitectura del Sistema alas HIS y cumpliendo con todos los patrones definidos por el proyecto para el desarrollo de aplicaciones.

Conclusiones Parciales

Con el desarrollo del presente capítulo, se obtuvieron los siguientes artefactos: modelo de implementación, diagrama de despliegue y diagrama de paquetes. Basado en dichos artefactos se obtuvo un producto acabado y totalmente funcional. El mismo se encuentra correctamente documentado y cumple los requerimientos de seguridad necesarios para garantizar la confidencialidad y privacidad de la información que se procesa en todo el sistema. En el proceso de codificación se cumplió con los estándares definidos, lo que permitió obtener un sistema fácil de mantener en el transcurso del tiempo.

Conclusiones.

Con el desarrollo de la aplicación para el enmascarado de datos en el proyecto alas HIS, se concluye lo siguiente:

1. El proceso de desarrollo llevado a cabo permitió la correcta identificación de los requisitos de la aplicación y su cumplimiento durante el diseño e implementación. El modelado de dominio propició depurar las actividades manuales y definir las que fuesen funcionalidades del sistema.

2. Los sistemas informáticos relacionados con el enmascaramiento de datos analizados en la investigación, demostraron que no cumplen con las necesidades del DSGH.

3. Se generó la documentación y los artefactos definidos por la metodología de desarrollo RUP y se asimiló las tecnologías y arquitectura definidas por el DSGH, proponiendo una herramienta informática que logra la realización del proceso de enmascaramiento de datos.

4. Se desarrolló la herramienta informática dataHISMask para el enmascaramiento de datos en el proyecto alas HIS aplicando las pautas de diseño e implementación definidas por el DSGH.

Recomendaciones.

Con los conocimientos adquiridos a lo largo de la investigación y luego de realizada la herramienta informática para el enmascarado de datos para el proyecto alas HIS se recomiendan mejoras para potenciar aún más la herramienta desarrollada:

- ✓ Incorporar otras técnicas de enmascarado de datos.
- ✓ Extender la herramienta al gestor de base de datos MySQL.

Referencias Bibliográficas

1. NewNet S.A. *Enmascaramiento: una forma de reforzar la seguridad y privacidad de los datos.* [En línea] [Citado el: 02 de 12 de 2012.] http://www.newnetsa.com/2009/07/enmascaramiento-una-forma-de-reforzar-la-seguridad-y-privacidad-de-los-datos/.

2. Portal de la Universidad de las Ciencias Informáticas. *Entorno Productivo.* [En línea] [Citado el: 04 de 12 de 2012.] http://www.uci.cu/entorno-productivo.

3. **González, Rolando Alfredo Hernández León y Sayda Coello.** *EL PROCESO DE INVESTIGACIÓN CIENTÍFICA.* Cidad de La Habana : La Editorial Universitaria, 2011.

4. **deConceptos.com.** Concepto de confidencialidad. [En línea] [Citado el: 14 de junio de 2013.] http://deconceptos.com/ciencias-juridicas/confidencialidad.

5. **msdn-Microsoft.** Integridad de datos. [En línea] [Citado el: 14 de junio de 2013.] http://msdn.microsoft.com/es-es/library/aa291812(v=vs.71).aspx.

6. **Focus Business Solutions.** La Protección de los Datos en los entornos de Sistemas no Productivos. [En línea] [Citado el: 10 de enero de 2013.] http://cxo-community.com/articulos/blogs/blogs-management/2548-la-proteccion-de-los-datos-en-los-entornos-de-sistemas-no-productivos.

7. **Informatica Corporation.** Dynamic Data Masking - Informatica. *Dynamic Data Masking - Informatica.* [En línea] 2012. [Citado el: 09 de enero de 2013.] http://www.informatica.com/es/products/data-masking/dynamic-data-masking/.

8. **INFORMATICA The Data Integration Company.** Reducción del riesgo de incumplimiento de normativas y de inseguridad de los datos - PowerCenter. [En línea] [Citado el: 7 de febrero de 2013.] http://origin-wwwnew.informatica.com/es/products_services/powercenter/options/data_masking/Pages/data_mask_benefits.aspx.

9. **Dataguise.** Dataguise | DgMasker™. [En línea] [Citado el: 23 de enero de 2013.] http://www.dataguise.com/products/dgmasker.html.

10. **Carmen Rosa Bolaño Arias, Mikel Díaz Hernández.** *Herramienta de enmascarado de datos para bases de datos PostgreSQL.* La Habana : s.n., 2011.

11. Data Masking A Net 2000 Ltd. [En línea] [Citado el: 06 de enero de 2013.] http://www.datamasker.com/DataMasking_WhatYouNeedToKnow.pdf.

12. **Itera.** Rational Unified Process. [En línea] 2008. [Citado el: 17 de enero de 2013.] http://www.iteraprocess.com/index.php?option=com_content&task=view&id=18&Itemid=42.

13. **Mora, Francisco.** UML: Lenguaje Unificado de Modelado. [En línea] 2003. [Citado el: 17 de enero de 2013.] http://www.dccia.ua.es/dccia/inf/asignaturas/GPS/archivos/Uml.PDF.

14. **IES Miguel de Cervantes.** cervantes1bachdyg - Arquitectura cliente-servidor. [En línea] [Citado el: 17 de enero de 2013.] http://cervantes1bachdyg.wikispaces.com/Arquitectura+cliente-servidor.

15. **Monografías.** Definición arquitectura cliente servidor. [En línea] [Citado el: 17 de enero de 2013.] http://www.monografias.com/trabajos24/arquitectura-cliente-servidor/arquitectura-cliente-servidor.shtml.

16. MASTERMAGAZINE. *Definición de Cliente / Servidor .* [En línea] [Citado el: 17 de enero de 2013.] http://www.mastermagazine.info/termino/4294.php.

17. **desarrolloweb.com.** MVC (Modelo Vista Controlador). [En línea] [Citado el: 25 de enero de 2013.] http://www.desarrolloweb.com/wiki/mvc-modelo-vista-controlador.html.

18. **Pantoja, Ernesto Bascón.** El patrón de diseño Modelo-Vista-Controlador (MVC). [En línea] 2004. [Citado el: 23 de enero de 2013.] http://ucbconocimiento.ucbcba.edu.bo/index.php/ran/article/view/84.

19. **Maldonado, Daniel M.** El CoDiGo K. *Arquitectura de programación en 3 capas.* [En línea] [Citado el: 2013 de enero de 20.] http://www.elcodigok.com.ar/2007/09/arquitectura-de-programacion-en-3-capas/.

20. **Smart, Jhon Ferguson.** JSF Jumpstart. [En línea] 2007. [Citado el: 23 de enero de 2013.] http://www.wakaleo.com/public_resources/jsf-jumpstarter.pdf.

21. **Jboss.org.** Community Documentation. [En línea] 2008. [Citado el: 23 de enero de 2013.] http://www.jboss.org/file-access/default/members/jbossrichfaces/freezone/docs/devguide/en/html/Introduction.html.

22. **w3schools.** AJAX Tutorial. [En línea] [Citado el: 23 de enero de 2013.] http://www.w3schools.com/ajax/default.asp.

23. **JBoss Community.** JBoss Ajax4jsf. [En línea] 2007. [Citado el: 23 de enero de 2013.] http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html.

24. **Hookom, Jacob.** JSF Central tm. Inside Facelets . [En línea] 2005. [Citado el: 23 de enero de 2013.] http://www.jsfcentral.com/articles/facelets_1.html.

25. **Monografias.com.** Lenguaje de programación para paginas web. [En línea] 2010. [Citado el: 23 de enero de 2013.] http://www.monografias.com/trabajos7/html/html.shtml.

26. Consortium, Word Wide Web. W3C. [En línea] 2008. [Citado el: 23 de enero de 2013.] http://www.w3c.es/divulgacion/guiasbreves/hojasestilo.

27. **Web Application.** Plataforma J2EE. JBoss Seam Framework. [En línea] 2008. [Citado el: 23 de enero de 2013.] http://wilmanchamba.wordpress.com/2008/02/20/jboss-seam-framework/.

28. **Franky, María Consuelo.** Java EE 5. [En línea] [Citado el: 2013 de enero de 21.] http://www.acis.org.co/fileadmin/Conferencias/ConfConsueloFranky_Abr19.pdf.

29. **Jaramillo, Wilmer.** Software Libre de Venezuela 777, C.A. [En línea] [Citado el: 18 de enero de 2013.] http://wilmer.fedorapeople.org/files/presentations/JBoss.pdf.

30. **Lucifer, Prometeo.** Java Runtime Environment – JRE. [En línea] [Citado el: 2013 de enero de 23.] http://www.elleonplateadodeojosrojos.es/blog/java-runtime-environment-jre/.

31. **Ottinger, Joseph.** JBoss releases JBoss Tools, Eclipse Plugins including Exadel. [En línea] [Citado el: 23 de enero de 2013.] http://www.theserverside.com/news/thread.tss?thread_id=45933.

32. Java. [En línea] [Citado el: 24 de enero de 2013.] http://www.java.com/es/about/.

33. **Larraga, Eduardo Falces.** Ingenieria Informática. *J2EE Framework capa de presentación.* [En línea] [Citado el: 2013 de enero de 23.] http://openaccess.uoc.edu/webapps/o2/bitstream/10609/610/1/00752tfc.pdf.

34. **Free Download Manager.** Paradigma visual para UML (Plataforma Java). *Visual Paradigm for UML [Java Platform] 6.0.* [En línea] [Citado el: 23 de enero de 2013.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/.

35. **tldp.org.** Manuales de Ayuda.com. *Breve historia de PostgreSQL.* [En línea] 2006. [Citado el: 23 de enero de 2013.] http://www.manualesdeayuda.com/manuales/bases-de-datos/postgresql/breve-historia-de-postgresql-01831.html.

36. **Kabuntu-es.** Programas de desarrollo libres. [En línea] [Citado el: 23 de enero de 2013.] http://www.kubuntu-es.org/wiki/desarrollo-programacion/programas-desarrollo-libres.

37. **Díaz, Wilder.** Diagrama de Componentes: DEFINICION. [En línea] 18 de 05 de 2009. [Citado el: 23 de 04 de 2013.] http://diagramacomponente.blogspot.com/2009/05/definicion_18.html.

38. **Marca Huallpara Hugo Michael, Quisbert Limachi Nancy Susana.** Diagrama de Despliegue. [En línea] [Citado el: 23 de 04 de 2013.] http://www.google.com.cu/url?sa=t&rct=j&q=modelo+de+despliegue&source=web&cd=6&ved=0CEIQFjAF&url=htt

p%3A%2F%2Fvirtual.usalesiana.edu.bo%2Fweb%2Fpractica%2Farchiv%2Fdespliegue.doc&ei=7l93Ub6zE6TP0wGInYHACA&usg=AFQjCNGu7s__JDdYCEwEmbkYnxa2XI2Q2Q&bvm=bv.4558.

39. **Microsof.** MSDN. *Revisiones de código y estándares de codificación.* [En línea] [Citado el: 24 de 04 de 2013.] http://msdn.microsoft.com/es-es/library/aa291591(VS.71).aspx.

40. **Mora, Jhon Freddy Montes.** PRUEBAS DEL SOFTWARE "CAJA BLANCA Y CAJA NEGRA". [En línea] 2008. [Citado el: 2013 de mayo de 14.] http://angelmolsoftware.blogspot.com/2008/11/pruebas-del-software-caja-blanca-y-caja.html.

Bibliografía.

Acuña, Karenny Brito. RUP Diseño e implementación del sistema. [En línea] 2009. [Citado el: 23 de 04 de 2013.] http://www.eumed.net/libros-gratis/2009c/584/RUP%20Diseno%20e%20implementacion%20del%20sistema.htm.

Amaya Alvarez Lorenzo, Mirelio Mora Maure. *Desarrollo de funcionalidades para la especialidad de Psicología del módulo Consulta Externa del sistema alas HIS.* La Habana : s.n., 2012.

Análisis del Entorno Productivo. [En línea] 02 de 10 de 2012. [Citado el: 06 de 05 de 2013.] http://ept-ata.foroperu.org/t13-leccion-7-analisis-del-entorno-productivo.

Carmen Rosa Bolaño Arias, Mikel Díaz Hernández. *Herramienta de enmascarado de datos para bases de datos PostgreSQL.* La Habana : s.n., 2011.

CODECRITICON. *Manipulando Ficheros en Java.* [En línea] 02 de noviembre de 2011. [Citado el: 04 de mayo de 2013.]

Consortium, Word Wide Web. W3C. [En línea] 2008. [Citado el: 23 de enero de 2013.] http://www.w3c.es/divulgacion/guiasbreves/hojasestilo.

Data Masking A Net 2000 Ltd. [En línea] [Citado el: 06 de enero de 2013.] http://www.datamasker.com/DataMasking_WhatYouNeedToKnow.pdf.

Dataguise. Dataguise | DgMasker™. [En línea] [Citado el: 23 de enero de 2013.] http://www.dataguise.com/products/dgmasker.html.

deConceptos.com. Concepto de confidencialidad. [En línea] [Citado el: 14 de junio de 2013.] http://deconceptos.com/ciencias-juridicas/confidencialidad.

desarrolloweb.com. MVC (Modelo Vista Controlador). [En línea] [Citado el: 25 de enero de 2013.] http://www.desarrolloweb.com/wiki/mvc-modelo-vista-controlador.html.

Díaz, Wilder. Diagrama de Componentes: DEFINICION. [En línea] 18 de 05 de 2009. [Citado el: 23 de 04 de 2013.] http://diagramacomponente.blogspot.com/2009/05/definicion_18.html.

EcuRed. Arquitectura Cliente Servidor. [En línea] [Citado el: 17 de enero de 2013.] http://www.ecured.cu/index.php/Arquitectura_Cliente_Servidor.

EcuRed. Eclipse, entorno de desarrollo integrado. [En línea] [Citado el: 23 de enero de 2013.] http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado.

El Club del Programador. *Configuraciones básicas del Servidor JBOSS .* [En línea] 31 de enero de 2012. [Citado el: 04 de mayo de 2013.] http://www.elclubdelprogramador.com/2012/01/31/jboss-configuraciones-basicas-del-servidor-jboss/.

Focus Business Solutions. La Protección de los Datos en los entornos de Sistemas no Productivos. [En línea] [Citado el: 10 de enero de 2013.] http://cxo-community.com/articulos/blogs/blogs-management/2548-la-proteccion-de-los-datos-en-los-entornos-de-sistemas-no-productivos.

Franky, María Consuelo. Java EE 5. [En línea] [Citado el: 16 de enero de 2013.] http://www.acis.org.co/fileadmin/Conferencias/ConfConsueloFranky_Abr19.pdf.

Franky, María Consuelo. Java EE 5. [En línea] [Citado el: 2013 de enero de 21.] http://www.acis.org.co/fileadmin/Conferencias/ConfConsueloFranky_Abr19.pdf.

Free Download Manager. Paradigma visual para UML (Plataforma Java). *Visual Paradigm for UML [Java Platform] 6.0.* [En línea] [Citado el: 23 de enero de 2013.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/.

González, Rolando Alfredo Hernández León y Sayda Coello. *EL PROCESO DE INVESTIGACIÓN CIENTÍFICA.* Cidad de La Habana : La Editorial Universitaria, 2011.

Hennebrueder, Sebastian. Java tutorials and development. *First EJB 3 Tutorial showing a session and entity beans with annotations and JBoss.* [En línea] [Citado el: 16 de enero de 2013.] http://www.lalilуna.de/ejb-3-tutorial-jboss.html..

Hibernate.org. Hibernate Tools for Eclipse and Ant. [En línea] [Citado el: 23 de enero de 2013.] http://www.hibernate.org/255.html.

Hookom, Jacob. JSF Central tm. Inside Facelets . [En línea] 2005. [Citado el: 23 de enero de 2013.] http://www.jsfcentral.com/articles/facelets_1.html.

HTML.net. [En línea] 2010. [Citado el: 23 de enero de 2013.] http://es.html.net/tutorials/html/lesson2.php.

IES Miguel de Cervantes. cervantes1bachdyg - Arquitectura cliente-servidor. [En línea] [Citado el: 17 de enero de 2013.] http://cervantes1bachdyg.wikispaces.com/Arquitectura+cliente-servidor.

Informatica Corporation. Dynamic Data Masking - Informatica. *Dynamic Data Masking - Informatica.* [En línea] 2012. [Citado el: 09 de enero de 2013.] http://www.informatica.com/es/products/data-masking/dynamic-data-masking/.

INFORMATICA The Data Integration Company. Reducción del riesgo de incumplimiento de normativas y de inseguridad de los datos - PowerCenter. [En línea] [Citado el: 7 de febrero de 2013.] http://origin-wwwnew.informatica.com/es/products_services/powercenter/options/data_masking/Pages/data_mask_benefits.aspx.

Itera. Rational Unified Process. [En línea] 2008. [Citado el: 17 de enero de 2013.] http://www.iteraprocess.com/index.php?option=com_content&task=view&id=18&Itemid=42.

Jaramillo, Wilmer. Software Libre de Venezuela 777, C.A. [En línea] [Citado el: 18 de enero de 2013.] http://wilmer.fedorapeople.org/files/presentations/JBoss.pdf.

Java. [En línea] [Citado el: 24 de enero de 2013.] http://www.java.com/es/about/.

jboos.org. RichFaces Developer Guide. [En línea] 05 de mayo de 2010. [Citado el: 15 de marzo de 2013.] http://docs.jboss.org/richfaces/latest_3_3_X/en/devguide/html/.

JBoss Community. JBoss Ajax4jsf. [En línea] 2007. [Citado el: 23 de enero de 2013.] http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html.

Jboss.org. Community Documentation. [En línea] 2008. [Citado el: 23 de enero de 2013.] http://www.jboss.org/file-access/default/members/jbossrichfaces/freezone/docs/devguide/en/html/Introduction.html.

Kabuntu-es. Programas de desarrollo libres. [En línea] [Citado el: 23 de enero de 2013.] http://www.kubuntu-es.org/wiki/desarrollo-programacion/programas-desarrollo-libres.

La Protección de los Datos en los entornos de Sistemas no Productivos. [En línea] [Citado el: 10 de enero de 2013.] http://cxo-community.com/articulos/blogs/blogs-management/2548-la-proteccion-de-los-datos-en-los-entornos-de-sistemas-no-productivos.

Larraga, Eduardo Falces. Ingenieria Informática. *J2EE Framework capa de presentación.* [En línea] [Citado el: 2013 de enero de 23.] http://openaccess.uoc.edu/webapps/o2/bitstream/10609/610/1/00752tfc.pdf.

León, Anthony R. Sotolongo. *Compendio de consultas útiles al catálogo de postgreSQL.* La Habana : s.n., 2011.

Lockhart, Thomas. PostgreSQL. [En línea] [Citado el: 03 de mayo de 2013.] http://www.ibiblio.org/pub/linux/docs/LuCaS/Postgresql-es/web/navegable/todopostgresql/postgres.htm.

Lucifer, Prometeo. Java Runtime Environment – JRE. [En línea] [Citado el: 2013 de enero de 23.] http://www.elleonplateadodeojosrojos.es/blog/java-runtime-environment-jre/.

Lucifer, Prometeo. Java Runtime Environment – JRE. [En línea] [Citado el: 18 de enero de 2013.] http://www.elleonplateadodeojosrojos.es/blog/java-runtime-environment-jre/.

Maldonado, Daniel M. El CoDiGo K. *Arquitectura de programación en 3 capas.* [En línea] [Citado el: 2013 de enero de 20.] http://www.elcodigok.com.ar/2007/09/arquitectura-de-programacion-en-3-capas/.

Marca Huallpara Hugo Michael, Quisbert Limachi Nancy Susana. Diagrama de Despliegue. [En línea] [Citado el: 23 de 04 de 2013.] http://www.google.com.cu/url?sa=t&rct=j&q=modelo+de+despliegue&source=web&cd=6&ved=0CEIQFjAF&url=http%3A%2F%2Fvirtual.usalesiana.edu.bo%2Fweb%2Fpractica%2Farchiv%2Fdespliegue.doc&ei=7l93Ub6zE6TP0wGInYHACA&usg=AFQjCNGu7s__JDdYCEwEmbkYnxa2XI2Q2Q&bvm=bv.4558.

MASTERMAGAZINE. *Definición de Cliente / Servidor .* [En línea] [Citado el: 17 de enero de 2013.] http://www.mastermagazine.info/termino/4294.php.

Microsof. MSDN. *Revisiones de código y estándares de codificación.* [En línea] [Citado el: 24 de 04 de 2013.] http://msdn.microsoft.com/es-es/library/aa291591(VS.71).aspx.

Milena Sánchez Menéndez, Anniel Arencibia Morales. *Módulo Tarjeta Control para el producto Synta.* La Habana : s.n., 2012.

Monografías. Definición arquitectura cliente servidor. [En línea] [Citado el: 17 de enero de 2013.] http://www.monografias.com/trabajos24/arquitectura-cliente-servidor/arquitectura-cliente-servidor.shtml.

Monografias.com. Lenguaje de programación para paginas web. [En línea] 2010. [Citado el: 23 de enero de 2013.] http://www.monografias.com/trabajos7/html/html.shtml.

Mora, Francisco. UML: Lenguaje Unificado de Modelado. [En línea] 2003. [Citado el: 17 de enero de 2013.] http://www.dccia.ua.es/dccia/inf/asignaturas/GPS/archivos/Uml.PDF.

Mora, Jhon Freddy Montes. PRUEBAS DEL SOFTWARE "CAJA BLANCA Y CAJA NEGRA". [En línea] 2008. [Citado el: 2013 de mayo de 14.] http://angelmolsoftware.blogspot.com/2008/11/pruebas-del-software-caja-blanca-y-caja.html.

msdn-Microsoft. Integridad de datos. [En línea] [Citado el: 14 de junio de 2013.] http://msdn.microsoft.com/es-es/library/aa291812(v=vs.71).aspx.

NewNet S.A. *Enmascaramiento: una forma de reforzar la seguridad y privacidad de los datos.* [En línea] [Citado el: 02 de 12 de 2012.] http://www.newnetsa.com/2009/07/enmascaramiento-una-forma-de-reforzar-la-seguridad-y-privacidad-de-los-datos/.

Ort, R. B. Sun microsystems. *The Java Persistence API - A Simpler Programming Model for Entity Persistence.* [En línea] [Citado el: 18 de enero de 2013.] http://java.sun.com/developer/technicalArticles/J2EE/jpa/.

Ottinger, Joseph. JBoss releases JBoss Tools, Eclipse Plugins including Exadel. [En línea] [Citado el: 23 de enero de 2013.] http://www.theserverside.com/news/thread.tss?thread_id=45933.

Pantoja, Ernesto Bascón. El patrón de diseño Modelo-Vista-Controlador (MVC). [En línea] 2004. [Citado el: 23 de enero de 2013.] http://ucbconocimiento.ucbcba.edu.bo/index.php/ran/article/view/84.

Portal de la Universidad de las Ciencias Informáticas. *Entorno Productivo.* [En línea] [Citado el: 04 de 12 de 2012.] http://www.uci.cu/entorno-productivo.

SeamFramework.org. Chapter 8. *Conversations and workspace management.* [En línea] [Citado el: 08 de mayo de 2013.] http://docs.jboss.com/seam/latest/reference/en-US/html/conversations.html.

Smart, Jhon Ferguson. JSF Jumpstart. [En línea] 2007. [Citado el: 23 de enero de 2013.] http://www.wakaleo.com/public_resources/jsf-jumpstarter.pdf.

tldp.org. Manuales de Ayuda.com. *Breve historia de PostgreSQL.* [En línea] 2006. [Citado el: 23 de enero de 2013.] http://www.manualesdeayuda.com/manuales/bases-de-datos/postgresql/breve-historia-de-postgresql-01831.html.

Tutorial de Java . *Creación y Control de Hilos.* [En línea] [Citado el: 03 de mayo de 2013.] http://dhw.umh.es/alex-bia/teaching/PC/material/hilos_tutorial-java/cap10-2.htm.

w3schools. AJAX Tutorial. [En línea] [Citado el: 23 de enero de 2013.] http://www.w3schools.com/ajax/default.asp.

Web Application. Plataforma J2EE. JBoss Seam Framework. [En línea] 2008. [Citado el: 23 de enero de 2013.] http://wilmanchamba.wordpress.com/2008/02/20/jboss-seam-framework/.

Glosario de Términos.

Encriptación: Es el proceso mediante el cual cierta información es cifrada de forma tal que el resultado sea ilegible a menos que se conozcan los datos necesarios para su interpretación.

Des-Identificar o Enmascaramiento de datos: La transformación de los datos, convirtiéndolos en verídicos, pero no verdaderos, de manera que solamente la información necesaria quede expuesta al usuario final.

Modelo de Dominio: Es el punto de partida para el diseño del sistema. Permite mostrar de manera visual los principales conceptos u objetos del mundo real que se manejan, ayudando a los usuarios, desarrolladores e interesados a utilizar un vocabulario común.

BD PostgreSQL: Donde se almacena la información, bajo el uso del gestor de base de datos PostgreSQL.

Datos Sensibles: Información contenida en la BD PostgreSQL del proyecto, con un grado crítico para el funcionamiento del mismo, información sumamente importante.

Datos no Sensibles: Información contenida en la BD PostgreSQL del proyecto, son datos comunes, no tienen gran impacto en el funcionamiento del proyecto.

BDO: Base de Datos Original.

BDD: Base de Datos Destino.

Clon: Se refiere a la copia que se le realiza a la BDO.

BDC: Base de Datos Clonada.