UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 6

Herramienta de enmascarado de datos para bases de datos PostgreSQL

Trabajo de Diploma para optar por el título de Ingeniero Informático

Autor(es): Carmen Rosa Bolaño Arias.
Mikel Díaz Hernández.

Tutor: Ing. Héctor Miguel Beltrán.

La Habana, Cuba
"Año 53 de la Revolución"
Junio, 2011

"El futuro de nuestra sociedad tiene que ser necesariamente un futuro de hombres de ciencia."

Fidel Castro Ruz

DECLARACIÓN DE AUTORÍA

Declaramos ser los únicos autores de este trabajo y autorizamos a la Facultad 6 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

_____ _____

Carmen Rosa Bolaño Arias. Mikel Díaz Hernández

_____

Ing. Héctor Miguel Beltrán.

DATOS DE CONTACTO

Tutor:

Ing. Héctor Miguel Beltrán

Universidad de las Ciencias Informáticas, La Habana, Cuba.

Email: hmbeltran@uci.cu

AGRADECIMIENTOS

A mi mamá preciosa por haberme dado la oportunidad de conocer este mundo maravilloso. Por estar siempre a mi lado, por todo el cariño y amor inmenso que me has transmitido durante el transcurso de todos estos años difíciles. Por darme siempre lo mejor de sí, por guiarme por el camino correcto y darme fuerzas en todo momento para seguir adelante las veces que tropecé y pensé que no podía continuar. Mami linda nunca podré pagarte todo lo que has hecho por mí. Quiero que sepas que te adoro y siempre serás mi vida.

A mi abuelita linda por ser más que una madre para mí, por darme su amor y apoyo incondicional, por estar siempre ahí cuando la necesité y confiar plenamente en mí. Por ser mí ejemplo a seguir en la vida. Un beso bien grande para ti. Te quiero mucho.

A mi papá por demostrarme que siempre que se quiere se pueden sacar las fuerzas para lograr las grandes cosas que nos proponemos en la vida.

A mis hermanitos por ser mi mayor inspiración, por darme fuerzas para salir adelante y hacer las cosas bien para darle el mejor ejemplo, y espero que así haya sido.

A mi nené lindo por llegar a mi vida justo en el momento que más lo necesité, por su amor y cariño inagotable. Te amo.

A mi mamita por darme la oportunidad de formar parte de su vida, por sus consejos, por confiar en mí y creer que siempre saldría adelante, por darme su cariño y apoyo incondicional.

A chago por siempre estar ahí y apoyarme tanto como si fuese su propia nieta.

A mi suegerita por hacerse cargo de esta niña malcriada como si fuese su hija.

A la Revolución cubana por darme la oportunidad de formar parte de esta universidad de excelencia.

A todas las personas lindas que han compartido conmigo durante estos 5 años de sacrificios, de tristezas y alegrías, a todos los que de una forma u otra han hecho posible que este sueño que parecía imposible se haga realidad.

A todos los profes del proyecto que nos ayudaron muchísimo con sus consejos y críticas para que este trabajo saliera adelante. Mil gracias por la paciencia y entrega.

A todas las muchachitas con las que he compartido apartamento durante estos 5 años, por soportar mis malacrianzas y mis malos momentos, por estar siempre ahí dándome su apoyo incondicional.

(A Alelí, Dayné que gracias a ella pude hacer esta tesis, lili, ceci, nary, ana, maryin, yani, vivi, yordy, any, milo, yahi)

A todos los profes que han contribuido con mi formación profesional y para la vida, a todos mis compañeros de aula, amigos y familia les agradezco infinitamente.

(A los chicos del 6304 y 6512, A Elennis que aunque no está aquí bastante que me malcrió, a Yanelis por ser como una madre para todos nosotros, a martica, Hayme, a Yanet por extenderme su mano cuando lo necesité, a Ileane por estar a mi lado en uno de los momentos más difíciles).

Agradezco infinitamente a mi dúo de tesis por ser tan sacrificado, paciente, entregado y querer que las cosas siempre sean un poquito mejores. Gracias por compartir juntos todos estos días de sacrificios y esfuerzos. Mil gracias a ti y siempre podrás contar conmigo.

Carmen Rosa

Gracias ante todo a esa mujer comprensiva y luchadora, mi guía, mi faro, mi vida, mi más grande tesoro; MI MADRE. Si me siento orgulloso de algo en esta vida MAMI, es de ser tu hijo.

Gracias a mi PADRE por darme su apoyo, comprensión y sobre todo la vida.

Gracias a ese SUPER HERMANO que tengo por apoyarme y ser mi inspiración.

Gracias a esa gigante que ha sabido soportarme y comprenderme todo este tiempo, MI CTA.

Gracias a toda mi FAMILIA por todo el apoyo incondicional que me dieron.

Agradecer a todas mis AMISTADES que me apoyaron en estos 5 años de estudio y dedicación.

Mikel

DEDICATORIA

A mis madres, mi abuelita, mis hermanos y a mi papá.

Carmen Rosa

Dedicado a toda mi familia especialmente a ti MAMÁ.

Mikel

RESUMEN

El presente trabajo surge de la necesidad de mantener la integridad y confidencialidad de los datos sensibles contenidos en las bases de datos (BD) PostgreSQL, para evitar la pérdida o sustracción inapropiada de los mismos, principalmente en los ambientes no productivos manteniendo las propiedades originales de la información almacenada en las BD. Por esta razón se decide desarrollar una aplicación que permita el enmascarado de datos en las BD PostgreSQL.

Para ello se realizó un estudio profundo del funcionamiento de las aplicaciones ya existentes que realizan el enmascarado de datos para los distintos gestores de base de datos, así como las arquitecturas y técnicas más utilizadas para este tipo de herramientas. La realización de la aplicación denominada Mask-PG constituirá una alternativa de gran ayuda en la utilización de las bases de datos con datos sensibles tanto en los ambientes productivos, como no productivos.

Palabras claves: base de datos, enmascarado de datos, PostgreSQL.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Desde que la informática ha asumido el protagonismo del tratamiento de información, casi cualquier dato, por no decir todos los que pasan por una organización o una institución determinada, se almacenan en algún soporte informático. Algo tan sencillo como registrar una solicitud, implica que los datos de contacto de los clientes (nombre, apellidos, número del documento de identidad, dirección particular, entre otros) se registren en las aplicaciones informáticas de los distintos organismos a los que ha accedido. Debido al crecimiento del volumen de la información, intensa recolección, uso y tratamiento de esta, las instituciones hacen uso de las bases de datos para el almacenamiento de dicha información, de ahí que estas, se conviertan en el principal blanco de ataque para el robo y sustracción de los datos.

Hoy día las empresas generalmente cumplen con las normativas de protección de datos en sus entornos productivos al contar con estrictos mecanismos de controles de acceso. Sin embargo olvidan que el mayor peligro puede encontrarse dentro de la misma organización al hacer uso de la información en los entornos no productivos (formación, desarrollo y pruebas).

Los ambientes no productivos utilizan con frecuencia para la realización exitosa de sus trabajos copias exactas de los datos sensibles contenidos en las bases de datos, ya que deben disponer de un gran volumen de información para testear el rendimiento o la respuesta del sistema. Cuando en entornos como éstos no se protegen los datos sensibles, se corre el riesgo de que estos sean utilizados con fines dañinos provocando una repercusión directa en la empresa, un efecto negativo en los medios, e incluso una pérdida de clientes.

Una de las técnicas utilizadas mundialmente para la protección de los datos es encriptar[1] los mismos, la encriptación es un escudo ante el robo directamente en las bases de datos, pero este método se torna débil y su trabajo se vuelve muy engorroso cuando se hace uso de él en entornos de desarrollo y pruebas. Cuando en entornos como estos se desarrolla un software o se ponen a prueba otros ya existentes que hacen uso de las bases de datos con los datos sensibles encriptados, los desarrolladores y probadores visualizarán muchos de estos en su estado auténtico cuando interactúen con las pantallas de la aplicación, ingresen datos y corran reportes. Si los datos son vistos en su estado original corren el riesgo

[1] Encriptar o cifrar: Proceso en el cual los datos a proteger son disfrazados convirtiéndolos en un formato ilegible.

de ser copiados, favoreciendo al robo o pérdida de la información, lo que constituye una violación a la confidencialidad de los datos almacenados.

Cuba no queda exenta de esta situación ya que muchas empresas, organismos e instituciones del país utilizan el gestor PostgreSQL para almacenar información que en su mayoría suele ser sensible. Un ejemplo de esto es la Universidad de las Ciencias Informáticas (UCI), institución que a pesar de ser muy joven, desempeña un rol importante en el desarrollo informático de la sociedad y juega un papel primordial respecto al uso del software libre debido a que un gran número de sus proyectos son realizados sobre herramientas libres como es el caso del Sistema Gestor de Bases de Datos (SGBD) PostgreSQL.

El Centro de Tecnologías de Gestión de Datos (DATEC), perteneciente a la UCI, tiene como objetivo principal contribuir a la soberanía tecnológica, potenciando tecnologías de bases de datos libres tomando como base el SGBD PostgreSQL. En este centro se realizan constantemente proyectos basados en dicho gestor, donde muchas veces las BD destinadas para la realización de los proyectos albergan datos sensibles. Dichos datos son visualizados en su formato original por los desarrolladores y probadores mientras realizan sus trabajos durante el ciclo de vida del proyecto a desarrollar. Lo que trae aparejado que los datos sensibles puedan ser sustraídos y utilizados con fines dañinos, violándose de esta manera la confidencialidad de la información almacenada.

Por lo anteriormente expuesto se ha identificado el **siguiente problema de la investigación:** ¿Cómo garantizar la privacidad y confidencialidad de los datos sensibles almacenados en bases de datos PostgreSQL?

Se define como **objeto de estudio de la investigación**: herramientas para el enmascarado de datos, enmarcado en el **campo de acción**: herramientas para el enmascarado de datos en el gestor PostgreSQL.

Para solucionar el problema antes mencionado se trazó como **objetivo general de la investigación**: Desarrollar una herramienta que permita el enmascarado de datos para garantizar la privacidad y confidencialidad de los datos sensibles contenidos en las bases de datos PostgreSQL.

A raíz del análisis del objetivo general se describen los siguientes **objetivos específicos**:

1. Identificar las funcionalidades de la herramienta para el enmascarado de datos.
2. Realizar el diseño de la herramienta para el enmascarado de datos.

3. Implementar la herramienta para el enmascarado de datos.
4. Realizar pruebas funcionales a la herramienta de enmascarado de datos.

Para dar cumplimiento al objetivo general se plantean las siguientes **tareas de la investigación**:

1. Investigación de las herramientas para el enmascarado de los datos.
2. Revisión bibliográfica de las tecnologías y herramientas para el desarrollo de la solución.
3. Investigación sobre las técnicas para el enmascarado de datos.
4. Obtención de los requisitos de la herramienta para el enmascarado de datos a partir de la definición de las historias de usuario.
5. Definición de la arquitectura de software para la herramienta de enmascarado de datos para PostgreSQL.
6. Diseño de la herramienta de enmascarado de datos para PostgreSQL.
7. Implementación de las historias de usuario definidas en la herramienta de enmascarado de datos para PostgreSQL.
8. Diseño de casos de pruebas a partir de las historias de usuario identificadas.

Con la realización exitosa de las tareas planteadas se espera como **posible resultado**: Herramienta para el enmascarado de datos sensibles contenidos en BD PostgreSQL.

El presente trabajo se ha estructurado de la siguiente manera:

Capítulo I: Fundamentos teóricos.
El capítulo comprende el estudio de las aplicaciones existentes de enmascarado de datos así como las principales empresas proveedoras de este tipo de aplicación. Se realiza además un análisis de las metodologías, herramientas y tecnologías a utilizar para el desarrollo de la aplicación de enmascarado de datos para el gestor PostgreSQL.

Capítulo II: Características y diseño del sistema.
En el capítulo se describe el flujo actual de procesos lo que permitirá un mejor entendimiento de las necesidades del sistema a desarrollar. Además se expondrán las principales características del sistema, su diseño, arquitectura, las historias de usuario identificadas y los requisitos no funcionales a tener en cuenta en su desarrollo.

Capítulo III: Implementación y prueba.
En el capítulo se describen los principales artefactos generados en las fases de implementación y prueba. Se realizan las tareas de ingeniería correspondientes a cada una de las historias de usuario para lograr implementar la herramienta de enmascarado de datos. Se expone el estándar de código utilizado y se describen las pruebas aplicadas a la herramienta con el objetivo de verificar que la misma cumple con todas las funcionalidades requeridas.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS

El presente capítulo comprende el estudio de las aplicaciones existentes de enmascarado de datos así como las principales empresas proveedoras de este tipo de aplicación. Se realiza además un análisis de las metodologías, herramientas y tecnologías a utilizar para el desarrollo de la aplicación de enmascarado de datos para el gestor PostgreSQL.

1.1 Sistemas Gestores de Bases de Datos

"*Un SGBD es un conjunto de programas que permiten crear y mantener una base de datos, asegurando su integridad, confidencialidad y seguridad*" (1).

Otro concepto de SGBD un poco más detallado es el de *"software o conjunto de programas que permiten crear y mantener una base de datos. El SGBD actúa como interfaz entre los programas de aplicación y el sistema operativo. El objetivo principal es proporcionar un entorno eficiente a la hora de almacenar y recuperar la información de las base de datos. Este software facilita el proceso de definir, construir y manipular bases de datos para diversas aplicaciones*" (2).

Los SGBD tienen como objetivo, proporcionar a los usuarios una visión abstracta de la información, es decir, el usuario no tiene necesidad de conocer los detalles de cómo se almacenan los datos, propiciando, la seguridad de la información mediante el control de acceso contando con un robusto subsistema de seguridad donde el administrador define restricciones para cada usuario, garantizando la integridad de los datos. Algunos ejemplos de SGBD son Oracle, DB2, PostgreSQL, MySQL, MS SQL Server entre otros.

Los SGBD entre otras tareas deben permitir:

- ✓ Definir una base de datos: especificar tipos, estructuras y restricciones de datos.
- ✓ Construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD.
- ✓ Manipular la base de datos: realizar consultas, actualizarla, generar informes.

Las principales características de los SGBD que los hacen ser de gran utilidad son:

- ✓ **Abstracción de la información**

  Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos por lo que son definidos varios niveles de abstracción.

- ✓ **Redundancia mínima**

  Un buen diseño de una base de datos logrará evitar la aparición de información repetida o redundante. De entrada, lo ideal es lograr una redundancia nula; no obstante, en algunos casos la complejidad de los cálculos hace necesaria la aparición de redundancias.

- ✓ **Flexibilidad de búsqueda**

  El empleo adecuado de índices en una relación acelera el acceso a la información, pero consume espacio considerable, es por esto que vale la pena hacer un análisis cuidadoso de cuáles atributos requieren ser indexados.

- ✓ **Independencia**

  La independencia de los datos consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.

- ✓ **Seguridad y confidencialidad integral**

  Normalmente, los SGBD disponen de un complejo sistema de permisos a usuarios y grupos de usuarios, que permiten otorgar diversas categorías de permisos.

- ✓ **Integridad**

  Un SGBD trata de proteger los datos ante fallos de hardware, datos introducidos por usuarios descuidados, o cualquier otra circunstancia capaz de corromper la información almacenada.

- ✓ **Respaldo y recuperación**

  Los SGBD deben proporcionar una forma eficiente de realizar copias de respaldo de la información almacenada en ellos, y de restaurar a partir de estas copias de los datos que se hayan podido perder.

1.1.1 Sistema Gestor de Bases de Datos PostgreSQL

PostgreSQL es un potente SGBD de código abierto del sistema de bases de datos objeto-relacional. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de fiabilidad e integridad de datos. Se ejecuta en los principales sistemas operativos, incluyendo Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows (3).

PostgreSQL es uno de los proyectos de código abierto más grandes y maduros que existe en la actualidad. La comunidad que le da soporte no sólo ha producido un SGBD, sino que también ha tenido en cuenta la confección de una buena documentación de código abierto disponible, cuenta además con diferentes empresas, contribuciones de proveedores comerciales y programadores de código abierto que apoyan y contribuyen de distintas maneras con el proyecto, basado en que el mismo está liberado bajo la licencia *Berkeley Software Distribution* (BSD) lo que permite su código fuente sea accesible para todos sin costo alguno, con la libertad de usar, modificar y distribuir PostgreSQL (3).

Debido a la gran variedad de ventajas que ofrece PostgreSQL como son: alto rendimiento, gran flexibilidad, respuesta rápida ante grandes volúmenes de datos y sobre todo que es un sistema estable, de código abierto y multiplataforma muchas empresas a nivel mundial hacen uso de este gestor para la realización exitosa de sus trabajos como es el caso de Affymetrix[2] (líder del mercado en la creación de herramientas para la industria de la investigación genética), utiliza PostgreSQL en el almacenamiento de las expresiones del Ácido Ribonucleico (ARN) a gran escala, otra empresa líder proveedora de productos empresariales y servicios basados en PostgreSQL es la EnterpriseDB[3] ofreciendo distribuciones de productos para este gestor.

1.2 Enmascarado de datos

La seguridad de los datos es tan fuerte como su eslabón más débil, los eslabones más débiles son a menudo los entornos de desarrollo y prueba. Aunque muchas empresas tomen todas las precauciones razonables para la protección de los datos sensibles en las bases de datos, no es suficiente para

[2] Más información en http://www.affymetrix.com/estore/index.jsp

[3] Más información en http://www.enterprisedb.com

garantizar la seguridad de los mismos en los entornos no productivos. La exposición de los datos sensibles es grave tanto para cuando el trabajo se realiza con personal propio de la empresa como para cuando es subcontratado. El desafío es producir aplicaciones que funcionen sin exponer los datos sensibles al mundo, regulando que ni los propietarios de dichas aplicaciones tengan acceso a esta información.

Des-Identificar o enmascarar es una vía para asegurar que el robo, exposición o pérdida de datos no pueda ser utilizada por nadie. Es una solución que transforma la información sensible en datos que no son verdaderos pero sí verídicos, es decir, que mantienen un aspecto similar al real, conservando las propiedades de los originales. Esta técnica permite alterar los datos sin perder sus atributos, de tal modo que garantice la confidencialidad de la información enmascarada (4).

1.2.1 Herramientas existentes que realizan el enmascarado de datos

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:

Informatica Data Masking[4] de la compañía de integración de datos informáticos: es un software completo, flexible y escalable que se emplea para gestionar el acceso a datos sensibles de aplicaciones (información de tarjetas de crédito, números de la seguridad social, nombres, direcciones, números telefónicos entre otros.). Este software evita la exposición involuntaria de información confidencial y disminuye el riesgo de filtración de datos.

Entre sus mayores beneficios encontramos:

- ✓ Reduce el riesgo de filtración de datos.
- ✓ Mejora la calidad de las actividades de desarrollo, pruebas y formación.

[4] Más información en http://www.informatica.com/es/products_services/data_masking/Pages/index.aspx

- ✓ Responde a la conformidad con normativas y disposiciones relacionadas con la privacidad de los datos.

DgMasker[5] 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.

Oracle Data Masking Pack[6]: es un paquete de Oracle que permite el cumplimiento normativo de los estándares de seguridad, basado en reglas, puede soportar gran variedad de formatos de máscaras y sólo necesita ser definido una vez. Esto ayuda a garantizar una aplicación coherente en las políticas de seguridad de la información y permite a las organizaciones compartir datos de forma rápida y escalable sin violar las regulaciones de privacidad.

El principal inconveniente de estas herramientas es que ninguna realiza el proceso de enmascarado de datos para el SGBD PostgreSQL, de esta premisa surge la necesidad de elaborar una herramienta que permita realizar esta importante tarea sobre dicho gestor.

1.2.2 Arquitecturas básicas para el enmascarado de datos

Fundamentalmente, existen dos tipos básicos de arquitecturas que se utilizan en el diseño de herramientas para el enmascarado de datos. Estas constan de dos bases de datos, una origen (contenedora de la información a enmascarar) y otra destino que será el resultado final que se obtendrá luego de realizar el proceso de enmascarado de los datos (4). A continuación se describen dichas arquitecturas.

[5] Más información en http://www.dataguise.com/products/dgmasker.html

[6] Más información en http://www.oracle.com/corporate/press/2007_nov/oem-datamasking-ow.html

- **Sobre la marcha, de servidor a servidor**

En esta arquitectura los datos no existen en la base de datos destino antes del enmascarado de los datos, se basa en mover la información que se va enmascarando de la base de datos origen a la destino.

Ventaja

- ✓ Los datos no están presentes en una forma original en la base de datos destino.

Desventaja

- ✓ Cualquier error en el proceso interrumpiría la transferencia de los datos.

- **En el lugar (In-situ)**

En esta arquitectura, se realiza un clon de la base de datos origen y luego el enmascarado de la información funciona sobre la base de datos clonada.

Ventajas

- ✓ Es posible aplicar otras operaciones de enmascarado en cualquier momento.
- ✓ Las operaciones de enmascarado son independientes del proceso de clon de la base de datos destino.

Desventaja

- ✓ Los datos se presentan en su estado auténtico en la base de datos destino por lo que será necesario durante ese tiempo mantener las medidas de seguridad.

Según el estudio realizado se decidió utilizar la arquitectura ***In-situ*** debido a que el uso de ella proveerá independencia entre los procesos de clonación de la base de datos y el enmascarado de la información favoreciendo el uso de otras técnicas de enmascarado en cualquier momento deseado.

1.2.3 Métodos o técnicas comunes del enmascarado de datos.

Existen diversas técnicas empleadas en el enmascarado de datos, entre las cuales se destacan: la sustitución, baraja, variación de números y fechas, cifrado, anulación de salida o eliminación,

enmascarado de datos de salida entre otros (4). A continuación se define el funcionamiento de algunas de ellas.

Sustitución

Esta técnica consiste en reemplazar al azar el contenido de una columna de datos con información que es similar pero que no tiene relación alguna con los datos reales.

La sustitución es muy eficaz 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.

Baraja

La técnica de baraja es similar a la sustitución, salvo que en este método 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 dependiendo de un porcentaje de variación de su valor real especificado por el cliente.

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 con un caracter de máscara (por ejemplo, una X). Este método oculta el contenido de los datos y conserva el mismo formato en las pantallas de interfaz e

informes. Por ejemplo, una columna de números de tarjetas de crédito podría ser: 4346 6454 0020 5379 quedando así después del enmascarado 4346 XXXX XXXX 5379.

Para el desarrollo de la herramienta de enmascarado de datos se implementarán los métodos de **sustitución**, **baraja** y **variación de números y fechas**, debido a que estas técnicas aseguran con su uso en el proceso de enmascarado de datos, que la información sensible sea sustituida con datos reales, manteniendo un aspecto similar al real y conservando las propiedades de los datos originales.

1.3 Metodología de desarrollo de software

En todo proceso de desarrollo de software se hace necesario el uso de una guía que posibilite el progreso eficiente en la construcción de aplicaciones. Como procedimientos y técnicas para toda la documentación y generación de artefactos del ciclo de vida del desarrollo del software, se requiere el uso de una metodología.

Una metodología para el desarrollo de un proceso de software es un conjunto de filosofías, fases, procedimientos, reglas, técnicas, herramientas, documentos y aspectos de formación para los desarrolladores de software.

Programación Extrema (Extreme Programming XP)

Es una metodología ligera de desarrollo de software que se basa en la simplicidad, la comunicación, la realimentación continua entre el cliente y el equipo de desarrollo así como la fácil reutilización del código desarrollado.

Posee como particularidades tener como parte del equipo, al usuario final y a expertos, pues son algunos de los requisitos para llegar al éxito del proyecto (5).

Entre las principales características de la metodología XP se encuentran:

- ✓ La comunicación, entre los usuarios y los desarrolladores.
- ✓ La simplicidad, al desarrollar y codificar los módulos del sistema.
- ✓ La retroalimentación concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales.

El uso de la metodología XP para el desarrollo de la herramienta de enmascarado de datos fue definido por el equipo de desarrollo del proyecto. Para más información ver el documento de arquitectura del proyecto PostgreSQL Empresarial.

1.4 Herramientas asociadas al desarrollo de la aplicación para el enmascarado de datos

Las herramientas utilizadas para el desarrollo de software son aplicaciones elaboradas con el fin de facilitar la realización de determinadas tareas. Para el desarrollo de la aplicación de enmascarado de datos se utilizarán las siguientes herramientas.

Gestión de Proyecto: Redmine GESPRO v1.0.

El Redmine es un completo gestor de proyectos web que ayuda a administrar y controlar las principales tareas de gestión que son necesarias para llevar a cabo los proyectos de trabajo. Liberado bajo los términos de la GNU, multiplataforma y de código libre. Cuenta con una interfaz sencilla de utilizar y está desarrollado con el moderno framework Ruby on Rais. El Redmine brinda características y funcionalidades que permiten al usuario disponer de información acerca de tareas de su proyecto y de esta manera contribuye a guiar de forma organizada el progreso del mismo (6).

Sistemas de Control de Versiones: Subversión v1.3.6

Es un controlador de versiones empleado para la administración de proyectos, permitiendo la realización de cambios y manteniendo un historial actualizado de los mismos, admitiendo además actualizar el proyecto con cualquier versión seleccionada, evitando así la pérdida de cualquier modificación realizada (7).

Herramienta de Modelado:

Una Herramienta CASE (del inglés ComputerAided Software Engineering) es una aplicación informática destinada a aumentar la productividad en el desarrollo de software, reduciendo el coste de las mismas en términos de tiempo y de dinero. Esta herramienta es de gran ayuda en todos los aspectos del ciclo de vida de desarrollo del software en tareas tales como, el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras (8).

Visual Paradigm for UML v6.1

Herramienta de UML para el desarrollo de software, soporta estándares de la industria clave, tales como Lenguaje de Modelado Unificado (UML), SysML entre otros. Además ofrece un completo conjunto de herramientas claves en los equipos de desarrollo de software necesarias para la captura de requisitos, la planificación de programas, la planificación de controles, la clase de modelado y modelado de datos (8).

Lenguaje de Programación:

En computación, un programa es una secuencia de instrucciones que permiten a un ordenador procesar una información conocida como datos de entrada (*input*) para producir una información de salida (*output*) o resultados. Esas instrucciones pertenecen a (o están escritas en) un lenguaje de programación determinado. Un lenguaje de programación es una construcción mental del ser humano para expresar programas. Está constituido por un grupo de reglas gramaticales, símbolos utilizables, términos con sentido único y una regla principal que resume las demás.

Existen diversos lenguajes de programación utilizados en la creación de aplicaciones de escritorios entre los que se encuentran: Java, C, C++, C#, entre otros. Estos son los más utilizados según la compañía TIOBE Software[7] la cual se dedica al estudio de estándares y tendencias en el mundo de la programación.

Java

Es un lenguaje de programación orientado a objetos, 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 y no hacer con los recursos críticos de una computadora (9).

Java también posee mecanismos para garantizar la seguridad durante la ejecución, comprobando antes de ejecutar el código, que este no viola ninguna restricción de seguridad del sistema donde se va a ejecutar. Cuenta con un cargador de clases, de modo que todas las clases cargadas a través de la red tienen su propio espacio de nombres para no interferir con las clases locales. Otra de sus características es que está preparado para la programación concurrente sin necesidad de utilizar ningún tipo de

[7] Más información en: http://www.tiobe.com

biblioteca. Finalmente, Java posee un gestor de seguridad con el que se puede restringir el acceso a los recursos del sistema.

Por las ventajas que proporciona y facilidades que brinda el trabajo con java como lenguaje de programación es seleccionado este para el desarrollo de la aplicación de enmascarado de datos.

Entorno de desarrollo integrado (IDE):

Un entorno de desarrollo integrado (*Integrated Development Environment* o IDE) es un programa compuesto por una serie de herramientas que utilizan los programadores para desarrollar códigos. Esta herramienta puede estar pensada para su utilización con un único lenguaje de programación o bien puede dar cabida a varios de estos. Dentro de los IDE utilizados para la programación en Java se encuentran el Eclipse y el NetBeans.

Eclipse

Eclipse es un IDE desarrollado por *International Business Machines* (IBM) y que se distribuye mediante una licencia de código abierto, la EPL (*Eclipse Public License).* En un primer momento era sólo un IDE para Java, pero mediante un sistema de plugins flexible se ha convertido en un IDE genérico soportado por la comunidad. Una de las desventajas de este IDE es lo complejo que se vuelve para desarrollar interfaces visuales así como al estar basado en plugins en ocasiones estos requieren versiones específicas para su correcto funcionamiento En la actualidad se puede utilizar para desarrollar aplicaciones en lenguajes de programación como: C++, Java, Python, Groovy, Perl y otros, soportando las plataformas de desarrollo *Java 2 Platform Standard Edition* (J2SE) y *Java 2 Platform Enterprise Edition* (J2EE).

NetBeans v6.9

Es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para otros lenguajes de programación. Dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones web, control de versiones, creación de aplicaciones compatibles con teléfonos móviles y resaltado de sintaxis Existe además un número importante de módulos para extender este IDE. Es un producto libre y gratuito sin restricciones de uso (10).

Características del NetBeans:

- ✓ Instalación y actualización simple.
- ✓ Diseñador visual de formularios para Swing GUI.
- ✓ Características visuales para el desarrollo web.
- ✓ Multi-plataforma.

Por lo anteriormente planteado se decide utilizar el IDE Netbeans en su versión 6.9 para la realización de la aplicación de enmascarado de datos.

PostgreSQL v9.0

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD (*Berkeley Software Distribution*) y con su código fuente disponible libremente. Las características de esta herramienta fueron mencionadas al inicio del presente capítulo.

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 que no existe ninguna herramienta que permita realizar el enmascarado de datos a las BD PostgreSQL. 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 **sustitución**, **baraja** y **variación de números y fechas** serán las implementadas en la herramienta para el enmascarado de datos, apoyados por la metodología de desarrollo XP, haciendo uso del lenguaje de programación Java con la ayuda del IDE de desarrollo NetBeans en su versión 6.9. Se utilizará además Redmine para la gestión de proyectos y Subversion para el control de versiones.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

En el presente capítulo se describe el flujo actual de procesos lo que permitirá un mejor entendimiento de las necesidades del sistema a desarrollar. Además se exponen las principales características del sistema, su diseño, su arquitectura, las historias de usuario identificadas y los requisitos no funcionales a tener en cuenta en su desarrollo.

2.1 Flujo actual de procesos

Los entornos no productivos son un agujero para la visualización y apropiación indebida de los datos sensibles. Muchas de las empresas inmersas en estos entornos exponen mediante el uso de BD modeladas en el gestor PostgreSQL, datos sensibles en su estado auténtico. Si algún desarrollador, probador o personal autorizado necesita hacer uso de las BD con datos sensibles para la realización de sus tareas, visualizará estos en su estado auténtico, pudiendo utilizar los mismos indebidamente. Actualmente muchos proyectos trabajan bajo estas condiciones, violándose factores primordiales en la seguridad informática, como son la confidencialidad e integridad de los datos.

2.2 Descripción del negocio

En el estudio realizado no fue posible definir claramente las necesidades reales del cliente, por lo que se decidió modelar el negocio teniendo en cuenta los objetos existentes que de una forma u otra están relacionados con el sistema a desarrollar, estableciendo las posibles relaciones que existen entre ellos. Además el modelo proporcionará un mejor entendimiento de los principales conceptos que se manejan en el negocio (Figura 1).

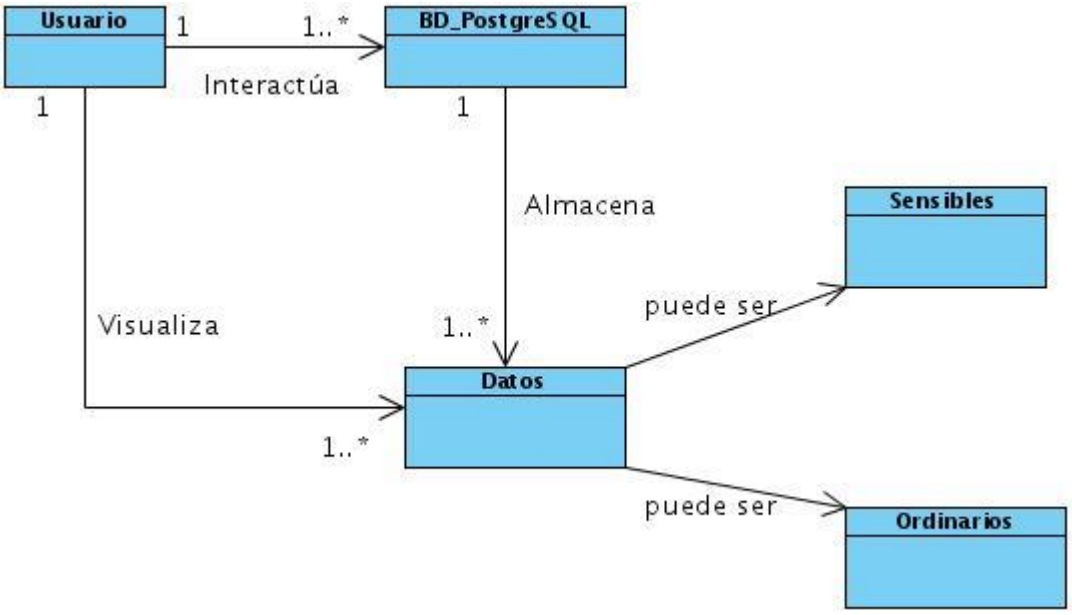


Figura 1: Diagrama del negocio

Un usuario es cualquier individuo con privilegios de acceso a una BD PostgreSQL, la cual almacena datos que pueden ser sensibles u ordinarios, donde el usuario podrá visualizar ambos tipos de información. Esta visualización indebida trae aparejado la apropiación de la información, pudiendo utilizarse la misma con fines dañinos, lo que constituye una violación a la integridad y confidencialidad de los datos almacenados en las BD PostgreSQL.

2.3 Propuesta del sistema

Para darle solución al problema planteado se decide desarrollar una herramienta que garantice la seguridad y confidencialidad de los datos sensibles almacenados en BD modeladas en el gestor PostgreSQL, mediante el uso del enmascarado de datos (Figura. 2). Permitirá que la pérdida o sustracción indebida de los datos sensibles no puedan ser utilizadas con fines dañinos, así como que los desarrolladores o probadores hagan uso de las bases de datos para la obtención de una mayor calidad en la realización de sus trabajos. La herramienta permitirá al usuario conectarse a la base de datos Origen (BDO), donde se podrá elegir por cada tabla contenida en la BDO los campos a enmascarar y la técnica a utilizar en cada caso. Luego se clonará la BDO para realizar el proceso de enmascarado sobre la BD clonada (BD Destino). Finalmente el usuario podrá visualizar y comparar los resultados.

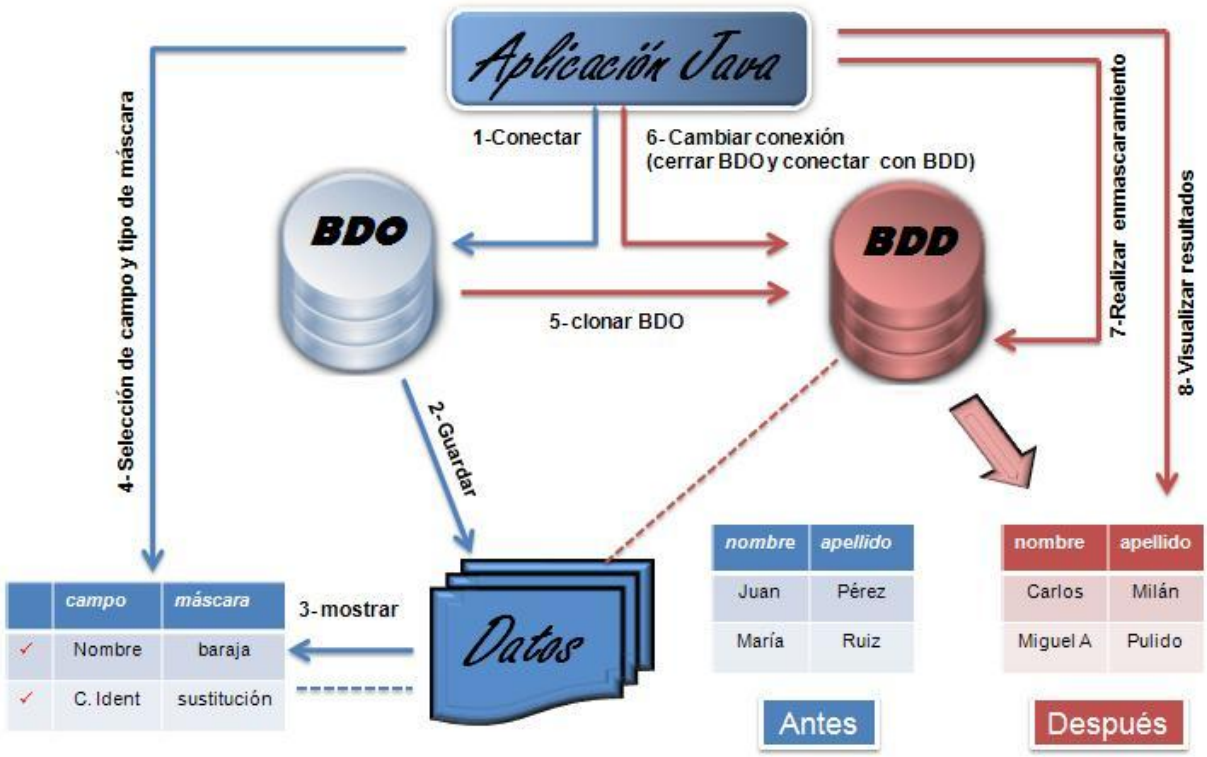


Figura 2: Flujo de datos a seguir para el enmascarado de datos

2.4 Definición de los requisitos

La definición de requisitos es una tarea de suma importancia a la hora de desarrollar un sistema. Estos permiten la obtención de un modelo bastante fiel del sistema a implementar y son de vital valor cuando de calidad se trata, ya que cuanto más alto sea el grado de aceptación por el cliente de los requisitos, más alta será la calidad del sistema desarrollado.

2.4.1 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. Son las propiedades que hacen al producto atractivo, usable, rápido y confiable.

Los requisitos no funcionales pueden ser clasificados en diversas categorías, entre las que se encuentran: Hardware, Software, Seguridad, Usabilidad, Soporte, Rendimiento, Fiabilidad, entre otros. A partir de estas clasificaciones se definieron los siguientes requisitos no funcionales:

- ✓ **Usabilidad**

La aplicación podrá ser utilizada por cualquier persona con conocimientos básicos del gestor PostgreSQL interesado en realizar el proceso de enmascarado a una BD que contenga datos sensibles.

- ✓ **Rendimiento**

El tiempo de respuesta dependerá de la cantidad de datos a enmascarar.

- ✓ **Portabilidad**

Será un sistema multiplataforma, lo que permitirá poder disponer del mismo en cualquier sistema operativo.

- ✓ **Software**

La computadora donde se vaya a utilizar la aplicación debe tener instalada la máquina virtual de java (JVM) en su versión 5 o superior.

- ✓ **Hardware**

La computadora debe tener como mínimo recomendado 128 mb de RAM así como un espacio libre en el disco duro de 15 mb.

- ✓ **Apariencia o interfaz externa**

La aplicación poseerá una interfaz amigable, solo tendrá la información necesaria para realizar el enmascarado, colores tenues, la apariencia estará en correspondencia con la de una aplicación de escritorio.

- ✓ **Requisitos de Seguridad**

Confidencialidad

Solo tendrá acceso a la visualización de los datos sensibles el usuario que posea los privilegios necesarios para esto.

Integridad

La aplicación no permitirá hacer cambios en los datos almacenados en la BD a enmascarar a menos que el usuario lo decida.

Disponibilidad

Se podrá hacer uso de la aplicación siempre que esté instalada y exista una BD para enmascarar.

- **Restricciones del diseño y la implementación**

Para el diseño e implementación de la aplicación se utiliza la metodología XP, haciendo uso del Visual Paradigm 6.1 para el modelado. También se utilizará como lenguaje de programación Java, como gestor de BD PostgreSQL 9.0 y como IDE de desarrollo el NetBeans en su versión 6.9.

2.4.2 Historias de usuario

Uno de los artefactos más importantes que genera la metodología XP son las historias de usuario (HU). Las mismas expresan su punto de vista en cuanto a las necesidades del sistema. Son descripciones cortas y escritas en el lenguaje del usuario sin terminología técnica. Otra de sus características es que solamente proporcionan los detalles sobre la estimación del riesgo y cuánto tiempo conllevará su implementación. Su nivel de detalle debe ser el mínimo posible, de manera que permita hacerse una ligera idea de cuánto costará implementar el sistema (11).

A continuación se muestran las HU definidas para el desarrollo de la herramienta para el enmascarado de datos sensible:

Tabla 1: HU Autenticar Usuario

Historia de Usuario	
Número: 1	**Nombre de la Historia de Usuario:** Autenticar Usuario.
Cantidad de modificaciones a la Historia de Usuario: 0	
Usuario: Mikel Díaz Hernández	**Iteración asignada:** 1

Prioridad en negocio: Muy Alta	**Puntos estimados:** 0.3 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 0.3 semanas
Descripción: Permitirá la autenticación de un usuario dado, el cual deberá poseer privilegios de súper-usuario. Además este debe proveer los datos: servidor, usuario, contraseña, base de datos a enmascarar y puerto.	
Observaciones: En caso de que los datos que se inserten sean incorrectos el sistema mostrará un mensaje de error.	
Prototipo de interfaces: Ver anexo 1	

Tabla 2: HU Mostrar campos de las tablas seleccionadas

Historia de Usuario	
Número: 2	**Nombre de la Historia de Usuario:** Mostrar campos de las tablas seleccionadas.
Cantidad de modificaciones a la Historia de Usuario: 0	
Usuario: Carmen Rosa Bolaño Arias	**Iteración asignada:** 2
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semanas
Descripción: Cuando el usuario selecciona la tabla que desea enmascarar se muestra un listado con todas las columnas que contiene dicha tabla a las cuales se les puede realizar el proceso de enmascarado de	

datos.
Observaciones: Si la tabla seleccionada no contiene columnas que puedan ser enmascaradas se mostrará un mensaje comunicándoselo al usuario.
Prototipo de interfaces: Ver anexo 2

Tabla 3: HU Realizar el enmascarado de datos

Historia de Usuario	
Número: 3	**Nombre de la Historia de Usuario:** Realizar el enmascarado de datos.
Cantidad de modificaciones a la Historia de Usuario: 1	
Usuario: Mikel Díaz Hernández	**Iteración asignada:** 2
Prioridad en negocio: Muy Alta	**Puntos estimados:** 3 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 3 semanas
Descripción: El usuario deberá seleccionar la columna que desea enmascarar y la técnica con que desea que se enmascare la misma. Este proceso se lleva a cabo aplicando las técnicas de enmascarado: **baraja**, **sustitución** y **variación de números y fechas**. Al realizar el proceso de enmascarado de datos los cambios se guardarán en la BD a la cual se está conectado o en una nueva BD especificada por el usuario (base de datos destino). Al concluir el proceso el sistema muestra un mensaje de éxito y brinda la posibilidad de visualizar el resultado obtenido.	

Observaciones: De fallar la conexión con la base de datos destino, la aplicación mostrará un mensaje de error. En caso de escoger la técnica de enmascarado **sustitución** el usuario debe seleccionar la base de conocimientos o agregar una nueva. Si se selecciona la técnica de variación de **números y fechas** el usuario deberá especificar el porcentaje de variación que desea utilizar.
Prototipo de interfaces: Ver anexo 3

Tabla 4: HU Realizar el clon de la base de datos origen

Historia de Usuario	
Número: 4	**Nombre de la Historia de Usuario:** Realizar el clon de la base de datos origen.
Cantidad de modificaciones a la Historia de Usuario: 0	
Usuario: Mikel Díaz Hernández	**Iteración asignada:** 2
Prioridad en negocio: Muy Alta	**Puntos estimados:** 3 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 3 semanas
Descripción: Permitirá realizar el clon de la BDO con el objetivo de aplicar las técnicas de enmascarado sobre dicha base de datos clonada o destino (BDD). Cuando se realice el clon no debe existir ninguna conexión sobre la BDO.	
Observaciones: De existir alguna conexión activa sobre la BDO, la aplicación mostrará un mensaje indicando la no	

realización de esta actividad.
Prototipo de interfaces: Ver anexo 4

Tabla 5: HU Visualizar resultados

Historia de Usuario	
Número: 5	**Nombre de la Historia de Usuario:** Visualizar resultados del enmascarado de datos.
Cantidad de modificaciones a la Historia de Usuario: 1	
Usuario: Carmen Rosa Bolaño Arias	**Iteración asignada:** 2
Prioridad en negocio: Alta	**Puntos estimados:** 1 semana
Riesgo en desarrollo: Alto	**Puntos reales:** 1 semanas
Descripción: Permitirá la visualización de los resultados obtenidos luego de haber realizado el proceso de enmascarado a los datos sensibles. Este resultado se mostrará en una vista previa que brindará el sistema.	
Prototipo de interfaces: Ver anexo 5	

Las HU describen las funcionalidades que debe realizar la herramienta de enmascarado de datos, además provee información al desarrollador que implementará dicha funcionalidad así como el tiempo estimado de duración para la implementación de la misma. También se especifica el grado de importancia (prioridad) que esta alcanza tanto en el negocio como en el desarrollo, para así valorar que HU debe ser

implementada primero sin poner en riesgo toda la realización del proyecto a desarrollar. Por último se describe la acción a realizar por cada HU, dejando plasmado en las observaciones los inconvenientes que se pueden presentar para la no realización exitosa de las mismas, además se especifica de manera visual como quedará conformada esta mediante el prototipo de interfaz.

2.5 Plan de iteraciones

La metodología XP contiene la filosofía de las metodologías ágiles, las cuales dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Al comienzo de cada ciclo se realiza una reunión de planificación de la iteración a realizar. Cada HU se traduce en tareas específicas de programación y para cada una de estas se realizan uno o más casos de pruebas. Dichas pruebas se realizan al final del ciclo en el que se desarrollan, pero también al final de cada uno de los ciclos siguientes, para verificar que subsiguientes iteraciones no han afectado a las anteriores. Después de ser descritas e identificadas las HU y estimado el esfuerzo propuesto para la realización de cada una de ellas, el siguiente paso es especificar cuáles de ellas serán implementadas para cada iteración del sistema (12).

Iteración 1: Tendrá como objetivo desarrollar las historias de usuario 1, 3 y 4 debido a que son las de mayor prioridad, estas forman la base del sistema a desarrollar siendo las encargadas de llevar a cabo el proceso de autenticación del usuario (conexión a la BD), clonación de la BDO y el enmascarado de los datos respectivamente.

Iteración 2: Una vez implementadas las HU de prioridad muy alta se desarrollarán las restante (2 y 5) las cuales poseen una prioridad alta para el desarrollo del enmascarado de datos.

2.6 Diseño del sistema

Para el diseño de aplicaciones informáticas la metodología XP no requiere la presentación del sistema mediante diagrama de clases utilizando notación UML, en su lugar se usan otras técnicas como las tarjetas CRC (Clase, Responsabilidad y Colaboración). El uso de estos diagramas puede aplicarse siempre y cuando intervengan en el mejoramiento de la comunicación, tratando que no sean un peso en su mantenimiento, no sean extensos y que enfoquen la información de mayor importancia (13).

2.6.1 Tarjetas CRC

Para poder diseñar el sistema como un equipo, se debe cumplir con tres principios: Cargo o Clase, Responsabilidad y Colaboración. Las tarjetas CRC permiten desprenderse del método de trabajo basado en procedimientos y trabajar con una metodología basada en objetos. Estas garantizan que el equipo completo contribuya en la tarea del diseño.

En sus responsabilidades describen las funciones que debe realizar una clase (cosas que conoce y las que realiza) mientras que en las colaboraciones se describen las demás clases con las que trabaja en conjunto para llevar a cabo sus responsabilidades.

A continuación se muestran las tarjetas CRC generadas para el diseño de la herramienta de enmascarado de datos.

Tabla 6: Tarjeta CRC Conexión

Tarjeta CRC	
Clase: Conexión	
Responsabilidades	**Colaboraciones**
➢ Gestionar conexión a la BD ➢ Ejecutar la consultas BD	

Tabla 7: Tarjeta CRC Manejo de datos

Tarjeta CRC	
Clase: Manejo de datos	
Responsabilidades	**Colaboraciones**
✓ Gestionar configuración de la aplicación ✓ Guardar datos	

Tabla 8: Tarjeta CRC Base de Datos

Tarjeta CRC	
Clase: Base de Datos	
Responsabilidades	**Colaboraciones**
✓ Realizar enmascarado ✓ Clonar BD ✓ Visualizar resultados ✓ Gestionar tablas	✓ Conexión ✓ Manejo de datos ✓ Tabla

Tabla 9: Tarjeta CRC Tabla

Tarjeta CRC	
Clase: Tabla	
Responsabilidades	**Colaboraciones**
✓ Gestionar las columnas de la tabla	✓ Columna

Tabla 10: Tarjeta CRC Columna

Tarjeta CRC	
Clase: Columna	
Responsabilidades	**Colaboraciones**
✓ Almacenar información	✓ ColumnaDate ✓ ColumnaInteger

Tabla 11: Tarjeta CRC ColumnaDate

Tarjeta CRC	
✓ **Clase:** ColumnaDate	
Responsabilidades	**Colaboraciones**
✓ Almacenar información de las columnas de tipo **Date**	

Tabla 12: Tarjeta CRC ColumnaInteger

Tarjeta CRC	
✓ **Clase:** ColumnaInteger	
Responsabilidades	**Colaboraciones**
✓ Almacenar información de las columnas numéricas	

2.7 Arquitectura del sistema

La arquitectura utilizada (**In-situ**) para el desarrollo de la herramienta de enmascarado de datos está compuesta por 3 componentes fundamentales (Figura. 3): una aplicación y dos bases de datos (origen y destino). La comunicación entre estos elementos será de la siguiente manera: se utilizará para la conexión de la aplicación con la base de datos el *driver* jdbc 9.0 (*Java Database Connectivity*). Este permite la ejecución de operaciones sobre la BD independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede. Para el manejo de los datos contenidos en las BD así como para clonar la BDO se utilizará el lenguaje SQL (*Structured Query Language*). Es un lenguaje declarativo de acceso a bases de datos relacionales el cual permite especificar diversos tipos de operaciones sobre dichas BD.

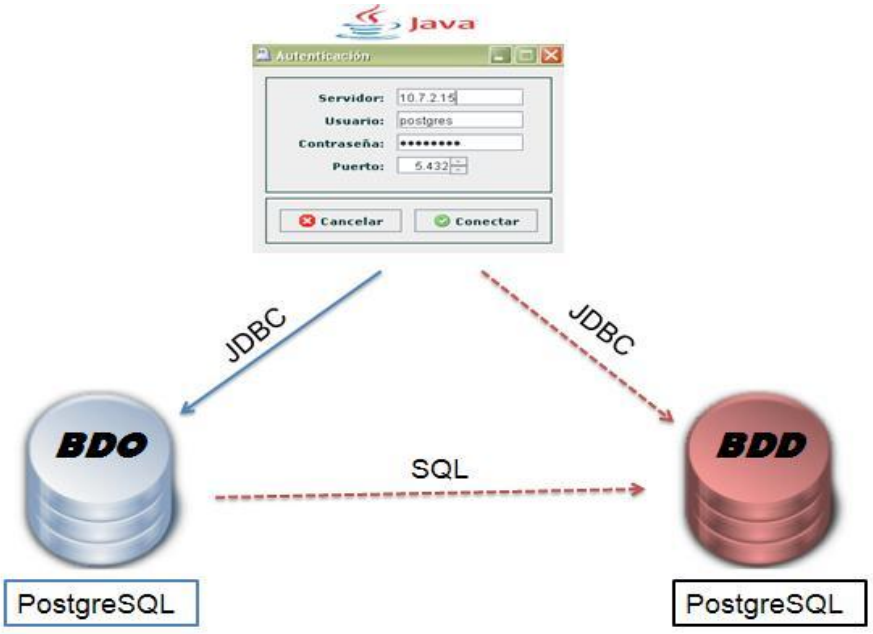


Figura 3: Arquitectura del sistema propuesto

2.7.1 Estilo arquitectónico utilizado

Un estilo es un concepto descriptivo que define una forma de articulación u organización arquitectónica. El conjunto de los estilos cataloga las formas básicas posibles de estructuras de software, mientras que las formas complejas se articulan mediante la composición de los estilos fundamentales (14).

Existen numerosos estilos arquitectónicos entre los que se encuentra el “Estilo de llamada y retorno” y dentro de él, el patrón de arquitectura Modelo-Vista-Controlador (MVC) (Figura. 4). Este patrón de arquitectura de software separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos.

Modelo: Contiene todo el código relacionado con el acceso a datos. Es importante que el código sea lo más genérico posible y se pueda reutilizar en otras situaciones y proyectos. Nunca se incluirá la lógica en el modelo, solamente consultas a la base de datos y validaciones de entrada de datos.

Vista: Contiene el código que representa la parte que será visualizada en pantalla por el usuario.

Controlador: Es el punto de entrada de la aplicación, se mantiene a la escucha de todas las peticiones, ejecuta la lógica de la aplicación, y muestra la vista apropiada para cada caso.

Esta separación permite construir y probar el modelo independientemente de la representación visual. Con el uso de este patrón se persigue mejorar la reusabilidad y que las modificaciones en las vistas impacten en menor medida en la lógica de negocio o de datos.

Ventajas:

- ✓ Soporte de vistas múltiples. Dado que la vista se halla separada del modelo y no hay dependencia directa del modelo con respecto a la vista, la interfaz de usuario puede mostrar múltiples vistas de los mismos datos simultáneamente. Por ejemplo, múltiples páginas de una aplicación Web pueden utilizar el mismo modelo de objetos, mostrado de maneras diferentes.
- ✓ Adaptación al cambio. Los requisitos de interfaz de usuario tienden a cambiar con mayor rapidez que las reglas de negocios. Los usuarios pueden preferir distintas opciones de representación, o requerir soporte para nuevos dispositivos como teléfonos celulares o PDAs. Dado que el modelo no depende de las vistas, agregar nuevas opciones de presentación generalmente no lo afecta.

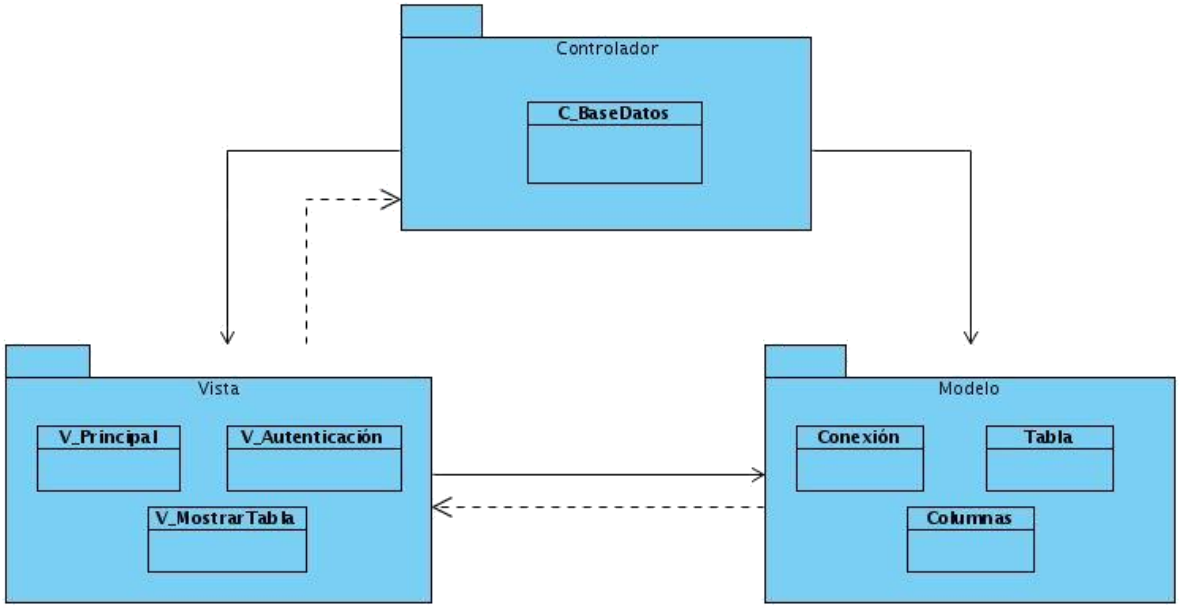


Figura 4: Patrón Modelo Vista Controlador

2.7.2 Patrones de Diseño

En la realización del diseño de aplicaciones en muchos casos se encuentran problemas que tienen presente un determinado patrón. Debido a esto se han estandarizado algunos patrones que son utilizados frecuentemente, estos son soluciones a problemas específicos y comunes del diseño orientado a objetos.

Los patrones GRASP (Patrones de Software para la asignación General de Responsabilidad) constituyen un apoyo para la enseñanza, que ayuda a entender el diseño de objetos. De los diferentes patrones que ofrece GRASP se emplearon para la modelación de la herramienta para el enmascarado de datos los siguientes:

Patrón Experto: El uso de este patrón garantiza que las clases contengan la información necesaria para cumplir con las responsabilidades por las que fueron creadas, sin depender de ninguna otra. Es decir, la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. El uso de este patrón se evidencia en la clase C_Conexión ya que la misma no necesita de la colaboración de ninguna otra clase para llevar a cabo sus responsabilidades (Figura 5). El uso de este patrón contribuye a un adecuado encapsulamiento, lo que favorece la robustez y fácil mantenimiento del sistema.

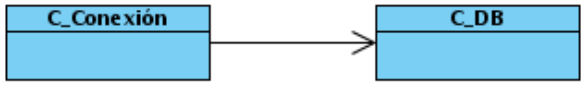


Figura 5: Ejemplo del uso del patrón experto

Patrón Creador: (Guía la asignación de responsabilidades relacionadas con la creación de objetos). Se asigna la responsabilidad a una clase de crear cuando, contiene, agrega, compone, almacena o usa otra clase, lo que brinda una alta posibilidad de reutilizar la clase creadora. El uso de este patrón se evidencia en la clase C_DB la cual crea objetos de las clases C_Tablas y C_Columnas para realizar sus responsabilidades (Figura 6).

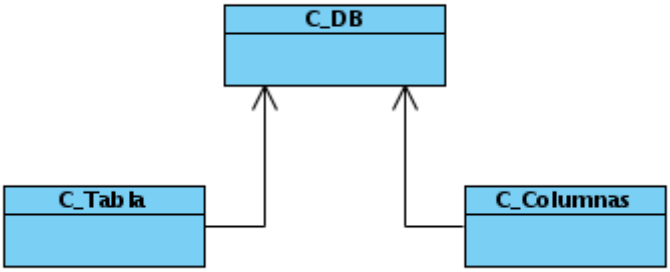


Figura 6: Ejemplo del uso del patrón creador

Patrón Bajo Acoplamiento: EL acoplamiento viene dado por la fuerza con que una clase está conectada a otras clases**.** Una clase con bajo acoplamiento es cuando dicha clase no depende de muchas otras. En el diseño desarrollado no existe una herencia profunda entre las clases, lo que significa que una clase no depende de muchas clases, evitando problemas como el difícil entendimiento de ellas y facilitando la reutilización de las mismas, al comunicarse entre ellas lo menos posible (Figura 6).

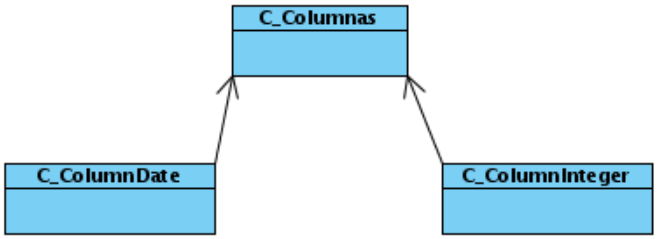


Figura 7: Ejemplo del uso del patrón bajo acoplamiento

Alta cohesión: El patrón consiste en asignar una responsabilidad de modo que la cohesión siga siendo alta. En el diseño orientado a objetos, la cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Las clases con baja cohesión a menudo representan un alto grado de abstracción o han asumido responsabilidades que deberían haber delegado a otros objetos. Este patrón se evidencia en el diseño del sistema ya que todas las clases contienen sólo

las funcionalidades que están relacionadas con ellas, garantizando que no realicen un trabajo excesivo y que puedan ser fácilmente reutilizables.

Conclusiones parciales

Se identificaron 5 HU las cuales responden a los requisitos funcionales de la aplicación así como se definió la iteración en la que será implementada cada HU. Se especificaron los requisitos no funcionales a tener en cuenta a la hora de utilizar la aplicación. Para el diseño de la aplicación se identificaron 7 tarjetas CRC, donde se utilizó para la modelación del sistema el estilo arquitectónico MVC y los patrones de diseño que ofrece GRASP, facilitando el trabajo del diseño de la aplicación para el enmascarado de datos.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA

En el presente capítulo se describen los principales artefactos generados en las fases de implementación y prueba. Se realizan las tareas de ingeniería correspondientes a cada una de las historias de usuario para lograr implementar la herramienta de enmascarado de datos. Se expone el estándar de código utilizado y se describen las pruebas aplicadas a la herramienta con el objetivo de verificar que la misma cumple con todas las funcionalidades requeridas.

3.1 Tareas de ingeniería

En el capítulo 2 se definieron las historias de usuario correspondientes a cada una de las iteraciones a desarrollar teniendo en cuenta el objetivo principal que es satisfacer las necesidades del cliente. Para lograr desarrollar estas HU con la calidad requerida, las mismas son descompuestas en tareas de programación (*task card*) o tareas de ingeniería y asignadas a los programadores para ser implementadas durante una iteración. Estas tareas no son para el conocimiento del cliente sino para el uso estricto de los programadores a la hora de implementar cada una de las historias de usuario identificadas.

A continuación se muestran cada una de las iteraciones y tareas de ingeniería vinculadas a cada una de las HU del sistema:

Iteración 1:

En esta iteración se le da cumplimiento a la implementación de las historias de usuario 1,3 y 4.

Tabla 13: Tarea de Ingeniería Conexión

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU 1
Nombre Tarea: Conexión	
Tipo de Tarea: Desarrollo	**Puntos Estimados:**1 semana
Programador Responsable: Mikel Díaz Hernández	

Descripción: Establecer la conexión con la base de datos a enmascarar.

Tabla 14: Tarea de Ingeniería Autenticar usuario

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** HU 1
Nombre Tarea: Autenticar usuario	
Tipo de Tarea: Desarrollo	**Puntos Estimados:**1 semana
Programador Responsable: Mikel Díaz Hernández	
Descripción: Realizar la autenticación del usuario el cual debe poseer permisos de súper-usuario.	

Tabla 15: Tarea de Ingeniería Enmascarado de datos

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU 3
Nombre Tarea: Enmascarado de datos.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 4 semanas
Programador Responsable: Mikel Díaz Hernández	
Descripción: Implementar las funcionalidades pertinentes para el enmascarado de datos.	

Tabla 16: Tarea de Ingeniería Clonar BD

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU 4
Nombre Tarea: Clonar BD.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2 semana
Programador Responsable: Mikel Díaz Hernández	
Descripción: Realizar clon a una BD especificada.	

Iteración 2:

En esta iteración se le da cumplimiento a la implementación de las historias de usuario 2 y 5.

Tabla 17: Tarea de Ingeniería Visualizar columnas

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU 2
Nombre Tarea: Visualizar columnas.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1 semana
Programador Responsable: Carmen Rosa Bolaño Arias	
Descripción: Mostrar las columnas de una tabla seleccionada por el usuario.	

Tabla 18: Tarea de Ingeniería Visualizar resultados

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU 5
Nombre Tarea: Visualizar resultados.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 4 semanas
Programador Responsable: Carmen Rosa Bolaño Arias	
Descripción: Visualizar los resultados del enmascarado de datos.	

3.2 Estándares de codificación

Para conseguir que el código se encuentre en buen estado y que cualquier persona del equipo pueda modificar cualquier parte del código es imprescindible que el estilo de codificación sea consistente. Esto es esencial para la programación en pares y para la propiedad colectiva del código. Un estándar de codificación es necesario para soportar otras prácticas de la metodología XP (15).

Los estándares de codificación son aquellos que permiten entender de manera rápida, fácil y sencilla el código empleado en el desarrollo de un software. Además, garantizan un mantenimiento óptimo de dicho código por parte del programador y mejoran la legibilidad permitiendo que otras personas puedan colaborar eficazmente ya que la redacción del código no les resulta incómoda.

Nomenclatura de nombres

- **Nombre de los paquetes:**

La primera letra en mayúscula.

Ejemplo: Clases, Visual.

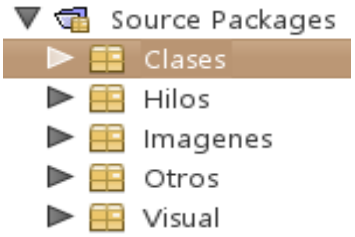


Figura 8: Ejemplo de nomenclatura de los paquetes

- **Nombre de las clases:**

Las clases se nombrarán en dependencia del tipo de clase, empezando con V_NombreClase (si la clase es visual) o C_NombreClase (si la clase es controladora) donde la primera letra de cada palabra debe ser mayúscula.

Ejemplo: V_CompararResultados, C_Conexion.

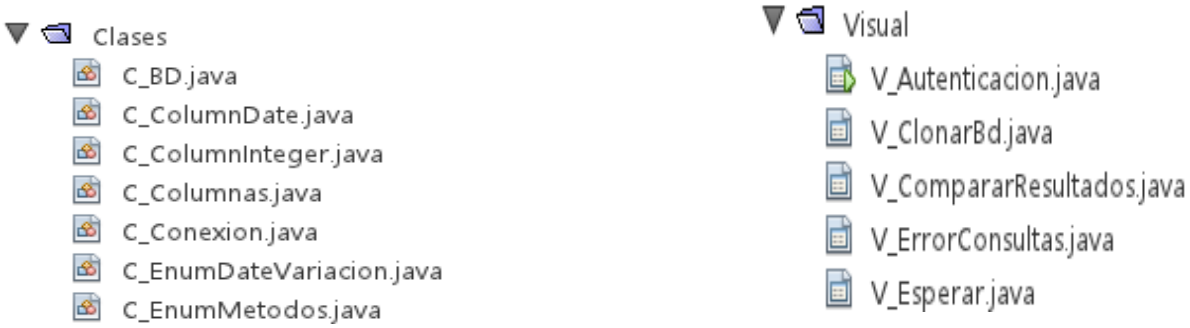


Figura 9: Ejemplo de nomenclatura de las clases

- **Nombre de métodos y variables:**

La primera letra de cada palabra en el nombre de los métodos será con mayúscula mientras que para las variables comenzará con minúscula y el resto de las primeras letras de las demás palabras será con mayúsculas.

Ejemplo de método: GetDatosColumnas(), BuscarTabla().

Ejemplo de variables: historialConsulta, contProgreso.

```
public Object[] GetDatosColumnas(C_Tabla tabla, int pos) {
    Object[] respuesta = new Object[tabla.cantrows];

    for (int i = 0; i < tabla.getCantrows(); i++) {
        respuesta[i] = tabla.getDatos()[i][pos];
    }

    return respuesta;
}

public int BuscarTabla(LinkedList<C_Tabla> listaTablas, C_Tabla tabla) {

    int pos = -1;
    C_Tabla c_Tabla = null;
    for (int i = 0; i < listaTablas.size(); i++) {
        c_Tabla = listaTablas.get(i);
        if (c_Tabla.getNombre().equals(tabla.getNombre())) {
            return i;
        }
    }
    return pos;
}
```

Figura 10: Ejemplo de nomenclatura de los métodos

```
LinkedList<String> historialConsulta;
C_ProgresoMask progreso;
int contProgreso = 0;
```

Figura 11: Ejemplo de nomenclatura de las variables

- **Declaraciones de variables:**

Se realizará una declaración por línea haciéndolas siempre al inicio de un bloque y utilizando nombres los más sugerentes posibles.

- **Sentencias:**

Siempre utilizar llaves ({}).

Las sentencias vacías se escriben en una sola línea.

```
while (tables.next()) {
    LNombTablas.add(tables.getString("TABLE_NAME"));
}
```

Figura 12: Ejemplo de sentencia

- **Líneas largas**

Cuando una línea no cabe en una única línea esta es fraccionada.

- **Nombres sugerentes**

Las clases, atributos y métodos contendrán nombres que permitan asociarlo con la actividad que se está realizando.

```
//MOSTRAR LAS TABLAS QUE CONTENGAN DATOS.
public LinkedList<C_Tabla> GetListTablasConDatos()
```

Figura 13: Ejemplo del uso de nombres sugerentes

- **Número de declaraciones por línea**

Se declara cada variable en una línea distinta, permitiendo que cada una de estas pueda ser comentada por separado.

3.3 Pruebas

El proceso de prueba es uno de los pilares fundamentales de la metodología XP, este le proporciona la posibilidad al cliente de verificar y concretar las funcionalidades de las HU, por lo que favorece la comunicación entre el cliente y el equipo de desarrollo. Esta filosofía ayuda a identificar y corregir fallos u omisiones cometidas en las mismas, por lo que se reduce el número de errores no detectados así como el tiempo entre la introducción de éste en el sistema y su detección. Permite identificar HU adicionales que no fueron obvias para el cliente o en las que este no hubiese pensado de no enfrentarse a dicha situación. Todo esto contribuye a elevar la calidad de los productos desarrollados y a la seguridad de los programadores a la hora de introducir cambios o modificaciones (16).

3.3.1 Pruebas de aceptación

Las pruebas de aceptación son pruebas de caja negra (Figura 14) definidas por el cliente para cada historia de usuario, y tienen como objetivo asegurar que las funcionalidades del sistema cumplen con lo que se espera de ellas. En efecto, las pruebas de aceptación corresponden a una especie de documento de requisitos en XP, ya que marcan el camino a seguir en cada iteración, indicándole al equipo de

desarrollo hacia donde tiene que ir y en qué puntos o funcionalidades debe poner el mayor esfuerzo y atención. Las pruebas de aceptación permiten al cliente saber cuando el sistema funciona, y que los programadores conozcan que es lo que resta por hacer.

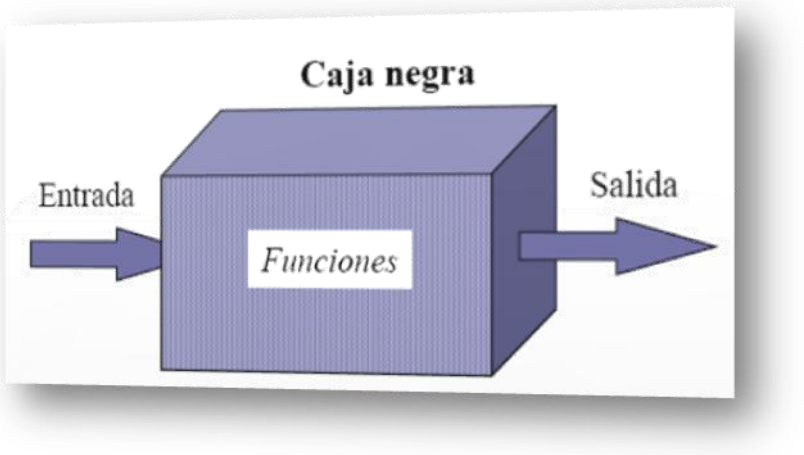


Figura 14: Prueba de Caja negra

A continuación se describen los casos de pruebas realizados para cada una de las HU del sistema:

Tabla 19: Diseño de caso de Prueba 1: HU 1: Autenticar Usuario

Escenario	Descripción	Variable 1	Variable 2	Variable 3	Variable 4	Variable 5	Respuesta del sistema	Flujo central
EC 1.1 Autenticar usuario correctamente.	Permitirá la autenticación de un usuario dado, el cual deberá poseer privilegios de súper-usuario.	V	V	V	V	V	El sistema muestra la interfaz principal.	1-El usuario introduce el ip del servidor. 2-El usuario introduce el nombre de la base de datos.
EC 1.2	El sistema mostrará	I	V	V	V	V	El sistema	

Autenticar usuario con campos incorrectos.	un error de autenticación.	V	I	V	V	V	muestra un error indicando que la autenticación falló.	3-Se introduce el nombre de usuario con permisos de súper–usuario. 4-El usuario introduce la contraseña. 5- El usuario introduce el puerto para la conexión. 6- El usuario presiona el botón aceptar.
		V	V	I	V	V		
		V	V	V	I	V		
		V	V	V	V	I		
EC 1.3 Autenticar usuario con campos vacíos.	El sistema no permitirá autenticar el usuario.	I	I	I	I	I	El sistema muestra un error indicando que la autenticación falló.	

Tabla 20: Descripción de las variables. Diseño de caso de Prueba 1

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
Variable 1	Servidor	Campo de texto	no	Se especifica el nombre o ip de la máquina servidora
Variable 2	base de datos	Campo de texto	no	Se especifica el nombre de la base de datos a la que se desea conectar
Variable 3	Usuario	Campo de texto	no	Se introduce el usuario

Variable 4	Contraseña	Campo de texto	no	Se especifica la contraseña para el usuario introducido.
Variable 5	Puerto	Campo de texto	no	Se introduce el número del puerto por el cual está escuchando peticiones el servidor especificado.

Tabla 21: Diseño de caso de Prueba 2: HU 2*:* Mostrar datos de las tablas seleccionadas

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Mostrar datos de las tablas seleccionadas que contengan información.	Cuando el usuario seleccione la tabla que desea enmascarar se mostrará un listado con todas las columnas que contiene dicha tabla a las cuales se les puede realizar el proceso de enmascarado.	El sistema muestra una lista con las columnas de la tabla a las cuales se le va a realizar el enmascarado.	1-El usuario selecciona la tabla de la cual desea visualizar sus campos.
EC 1.2 Mostrar datos de las tablas seleccionadas que no contengan información.	La aplicación deberá mostrar un mensaje de información comunicándoselo al usuario.	El sistema muestra un mensaje comunicándole al usuario que la tabla no contiene información para ser cargada.	

Tabla 22: Diseño de caso de Prueba 3: HU 3: Realizar enmascarado de los datos

Escenario	Descripción	Variable 1	Variable 2	Variable 3	Respuesta del sistema	Flujo central
EC 1.1 Realizar el enmascarado de los datos correctamente.	El usuario deberá seleccionar la columna que desea enmascarar y la técnica con que desea que se enmascare dicha columna.	V	V	V	El sistema muestra un mensaje indicándole al usuario que el enmascarado de la información se ha realizado correctamente.	1-El usuario selecciona los campos de la tabla a enmascarar. 2-El usuario selecciona la técnica de enmascarado que desea emplear en dicha columna. 3- El usuario especifica la base de datos destino. 4- El usuario presiona el botón aceptar.
EC 1.2 Realizar el enmascarado de los datos incorrectamente.	En caso de que el usuario no seleccione algún campo el sistema deberá mostrar un mensaje de error.	I	V	V	El sistema muestra un mensaje indicándole al usuario que debe seleccionar los campos correspondientes.	
		V	I	V		
		V	V	I		

Tabla 23: Descripción de las variables. Diseño de caso de Prueba 3

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
Variable 1	Campo	Campo de selección(Checkbox)	no	Se selecciona el campo a enmascarar
Variable 2	Técnica	Campo de selección(Checkbox)	no	Se especifica la técnica de enmascarado para el campo seleccionado
Variable 3	base de datos destino	Campo de selección(Radio Buton)	no	Se especifica la base de datos destinos que almacenará todos los cambios ejecutados

Tabla 24: Diseño de caso de Prueba 4: HU 4: Realizar clon de la base de datos origen

Escenario	Descripción	Variable 1	Respuesta del sistema	Flujo central
EC 1.1 Realizar el clon de la base de datos origen correctamente.	Permitirá la realización del clon de la base de datos origen.	V	El sistema muestra un mensaje indicándole al usuario que el enmascarado de la información se ha realizado correctamente.	1-El usuario especifica el nombre del clon a realizar. 2-El usuario oprime la tecla aceptar.
EC 1.2 Realizar el clon de la base de datos origen incorrectamente	No permitirá la clonación de la base de datos.	I	El sistema muestra un mensaje indicándole al usuario que la base de datos ya existe.	

Tabla 25: Descripción de la variable. Diseño de caso de Prueba 4

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
Variable 1	Nueva base de datos	Campo de Texto	no	Se especifica el nombre que recibirá la base de datos clonada

Tabla 26: Diseño de caso de prueba 5: HU 5: Visualizar resultados del enmascarado de los datos

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Visualizar resultados del proceso de enmascarado de los datos correctamente.	Permitirá la visualización de los resultados obtenidos luego de haber realizado el enmascarado de los datos sensibles.	El sistema muestra las tablas con los datos enmascarados.	1-El usuario selecciona la opción visualizar resultados.

Los resultados arrojados por los casos de pruebas diseñados no fueron del todo satisfactorios ya que se detectaron 3 no conformidades.

En la tabla 27 que se muestra a continuación se presentan las no conformidades detectadas durante el proceso de las pruebas a la herramienta para el enmascarado de datos sensibles.

Tabla 27: No Conformidades

Elemento	No	No conformidad	Aspecto correspondiente	Etapa de detección	Clasificación	Estado No conformidad	Respuesta equipo de desarrollo

Clonar base de dato	1	Error gramatical en clonar **base de dato**, debe ser base de datos.	En herramientas, en Clonar base de dato.	Revisión de la aplicación.	Significativa	PD 9/04/2011 RA 10/04/2011	Se puso base de datos.
Enmascarar datos	2	Error gramatical en Enmascarar datos en Aplicar Cambios, Nueva bd, debería ser bd con mayúscula.	En Enmascarar datos en Aplicar cambios, en Nueva bd.	Revisión de la aplicación	Recomendación	PD 9/04/2011 RA 10/04/2011	Se cambió bd por BD.
Clonar base de datos	3	Error gramatical en Clonar base de datos, en especifique el nombre de la nueva BD, de ser especifique con mayúscula.	Clonar base de datos, en especifique el nombre de la nueva BD.	Revisión de la aplicación	No significativa	PD 9/04/2011 RA 10/04/2011	Se cambió especifique por Especifique.

Una vez realizadas las pruebas y corregidas las no conformidades detectadas por el equipo de desarrollo, la herramienta desarrollada fue sometida a un proceso de revisiones por el departamento de calidad perteneciente al centro DATEC. En este proceso se llevaron a cabo dos iteraciones que arrojaron como resultado la existencia de 0 no conformidades, lo cual se ve reflejado en el Acta de liberación de productos de software emitida por dicho departamento.

Conclusiones parciales

De las 5 historias de usuario identificadas en la fase de análisis se logró implementar el 100% de las mismas, haciendo uso del NetBeans 6.9 y Java como lenguaje de programación. Se realizaron las pruebas funcionales a la aplicación a través del diseño de 5 casos de prueba uno por cada historia de usuario, arrojaron como resultado 3 no conformidades, las mismas fueron resultas por el equipo de desarrollo en un corto intervalo de tiempo. Por lo que se puede afirmar que la herramienta para el enmascarado de datos sensibles cumple con las funcionalidades requeridas por el cliente.

CONCLUSIONES

Con la elaboración de la aplicación de enmascarado de datos para las BD PostgreSQL se concluyó una intensa jornada de estudio y dedicación, dándole cumplimiento a los objetivos trazados en la investigación, y garantizando que las BD con información sensible puedan ser utilizadas por cualquier persona sin miedo alguno a que los datos puedan ser utilizados con fines dañinos.

- El estudio de las aplicaciones existentes de enmascarado de datos así como las técnicas de enmascarado permitieron identificar 5 historias de usuario, las cuales fueron de gran importancia para sentar las bases del desarrollo de la aplicación.

- Se diseñó la aplicación de acuerdo a las necesidades planteadas, haciendo uso de la arquitectura **in-situ** complementada con la utilización del patrón arquitectónico MVC.

- Se implementó la primera versión de la herramienta de enmascarado de datos para las BD PostgreSQL la cual garantiza la confidencialidad de los datos sensibles contenidos en las BD.

- Se realizaron pruebas funcionales a la aplicación, mediante el diseño de 5 casos de pruebas uno por cada HU arrojando 3 no conformidades que fueron solucionadas. Demostrando que la aplicación de enmascarado de datos para las BD PostgreSQL cumple con los resultados esperados.

A partir de lo anteriormente expuesto se concluye que los objetivos de la investigación fueron cumplidos satisfactoriamente.

RECOMENDACIONES

Con los conocimientos adquiridos a lo largo de la investigación y luego de realizada la herramienta para el enmascarado de datos en el gestor PostgreSQL se seleccionaron un conjunto de mejoras para potenciar aún más la herramienta desarrollada, entre las que se encuentran:

- Incorporar otras técnicas de enmascarado de datos.
- Crear una base de conocimiento para mejorar el enmascarado de los datos mediante la técnica de sustitución.
- Reducir mediante métodos más eficientes el tiempo de respuesta al realizar el proceso de enmascarado de los datos.

REFERENCIAS BIBLIOGRÁFICAS

1. Universidad de Jaén. . [En línea] 11 de 11 de 2010. http://wwwdi.ujaen.es/~barranco/publico/ofimatica/tema7.pdf..

2. **Cobo Yera, Angel.** Diseño y programación de bases de datos. . [En línea] http://books.google.com.cu/books?id=anCDr9N-kGsC&pg=PA7&dq=definicion+de+sistemas+gestores+de+base+de+datos&hl=es&ei=VgDcTMHyGovOngesrJQX&sa=X&o.

3. Global Development Group. PostgreSQL. . [En línea] 22 de 11 de 2010 . http://www.postgresql.org/..

4. Net 2000 Ltd. Data Masker. . [En línea] 1998. http://www.datamasker.com/DataMasking_WhatYouNeedToKnow.pdf..

5. **Sanchez Mendoza, María A. .** Informatizate. [En línea] 4 de 12 de 2010 . http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html..

6. **Fernández, Arturo.** Universidad de la Rioja. . [En línea] 1 de 12 de 2010. http://dialnet.unirioja.es/servlet/articulo?codigo=3317093..

7. Apache Subversion. . [En línea] 28 de 11 de 2010 . http://subversion.apache.org/..

8. Visual Paradigm International. [En línea] 3 de 12 de 2010. http://www.visual-paradigm.com/..

9. Java. . [En línea] 28. de 11 de 2010 . http://www.java.com/es/about/..

10. Oracle Corporation and/or its affiliates.NetBeans. . [En línea] 3 de 12 de 2010 . http://www.netbeans.org..

11. **Letelier, Patricio y Penadés, Carmen.** *Metodologías ágiles para el desarrollo de software. .* 2004.

12. **Wells, Don.** Extreme Programming. . [En línea] 24 de 02 de 2010. http://www.extremeprogramming.org..

13. **Silva, Soria.** *Elaboración de una aplicación que diagnostique el grado de acercamiento de los proyectos productivos al programa de mejoras de la UCI. .* 2010.

14. **Reynoso Billy, Carlos.** [En línea] 14 de Enero de 2009. http://www.willydev.net/descargas/prev/IntroArq.pdf.

15. **Wesley, Addison.** *Extreme Programing Explained.* 2000.

16. **Joskowicz, José.** *Reglas y Prácticas.* 2011.

BIBLIOGRAFÍA

2010 . Global Development Group. PostgreSQL. . [En línea] 22 de 11 de 2010 . http://www.postgresql.org/..

2010 . Java. . [En línea] 28. de 11 de 2010 . http://www.java.com/es/about/..

2010. Universidad de Jaén. . [En línea] 11 de 11 de 2010. http://wwwdi.ujaen.es/~barranco/publico/ofimatica/tema7.pdf..

2010 . Apache Subversion. . [En línea] 28 de 11 de 2010 . http://subversion.apache.org/..

Calero Solís, Manuel. 2010. Una explicación de la programación extrema (XP). [En línea] 20 de 11 de 2010. http://www.willydev.net/descargas/prev/explicaxp.pdf..

Cobo Yera, Angel. Diseño y programación de bases de datos. . [En línea] http://books.google.com.cu/books?id=anCDr9N-kGsC&pg=PA7&dq=definicion+de+sistemas+gestores+de+base+de+datos&hl=es&ei=VgDcTMHyGovOngesrJQX&sa=X&o.

2010. Cryptoforge. . [En línea] 15 de 11 de 2010. http://www.cryptoforge.com.ar/seguridad.htm..

2011. Estándar para codificación en lenguaje C++. . [En línea] 05 de 03 de 2011. http://progra.iteso.mx/estandares/estandar%20codificacion%20c++/estandarcodificacion.pdf..

2010. Extreme Programming. . [En línea] 29 de 11 de 2010. http://oreilly.com/catalog/extprogpg/chapter/ch05.pdf..

Fernández, Arturo. 2010. Universidad de la Rioja. . [En línea] 1 de 12 de 2010. http://dialnet.unirioja.es/servlet/articulo?codigo=3317093..

Joskowicz, José. 2011. Reglas y Prácticas. 2011.

2010. Kioskea.Lenguajes de programación. [En línea] 2 de 12 de 2010. http://es.kioskea.net/contents/langages/langages.php3..

Letelier, Patricio y Penadés, Carmen. 2004. Metodologías ágiles para el desarrollo de software. . 2004.

Letelier, Patricio y Penadés, Mª Carmen. Métodologías ágiles para el desarrollo de software:eXtreme Programming. . Valencia : s.n. : s.n.

Martínez., Rafael. 2010. PostgreeSQL. . [En línea] 2 de 12 de 2010. http://www.postgresql-es.org/..

1998. Net 2000 Ltd. Data Masker. . [En línea] 1998. http://www.datamasker.com/DataMasking_WhatYouNeedToKnow.pdf..

2011. Netbeans. [En línea] 17 de 1 de 2011. http://www.netbeans.org.

2010 .. Oracle Corporation and/or its affiliates.NetBeans. . [En línea] 3 de 12 de 2010 . http://www.netbeans.org..

2011. Patrones Arquitectónicos. . [En línea] 01 de 02 de 2011. http://www.lsi.us.es/docencia/get.php?id=1130 .

Reynoso Billy, Carlos. 2009.. [En línea] 14 de Enero de 2009. http://www.willydev.net/descargas/prev/IntroArq.pdf.

Sanchez Mendoza, María A. . 2010 . Informatizate. [En línea] 4 de 12 de 2010 . http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html..

Silva, Soria. 2010. Elaboración de una aplicación que diagnostique el grado de acercamiento de los proyectos productivos al programa de mejoras de la UCI. . 2010.

2010. Visual Paradigm International. [En línea] 3 de 12 de 2010. http://www.visual-paradigm.com/..

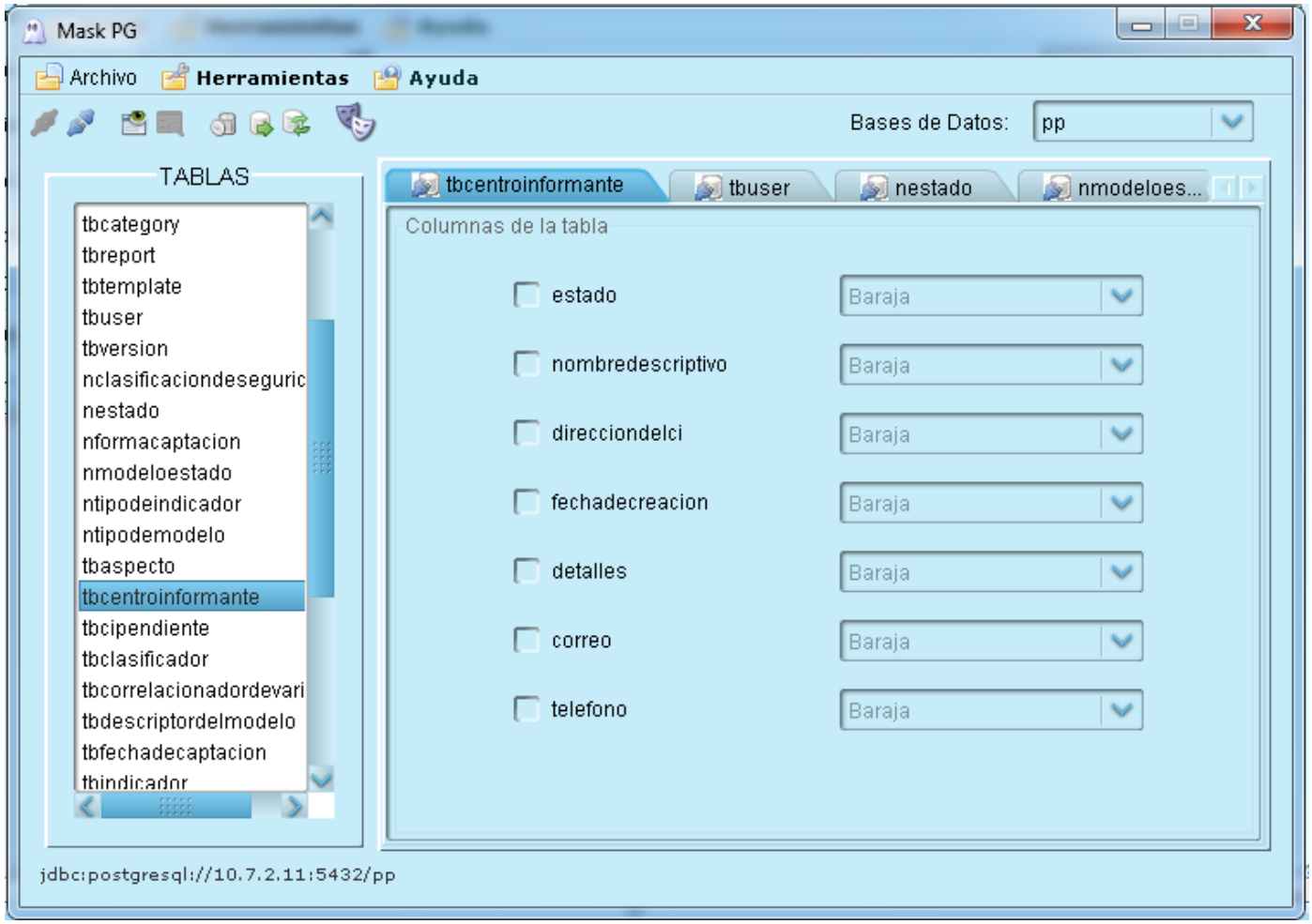
Wells, Don. 2010.. Extreme Programming. . [En línea] 24 de 02 de 2010. http://www.extremeprogramming.org..

Wesley, Addison. 2000. Extreme Programing Explained. 2000.

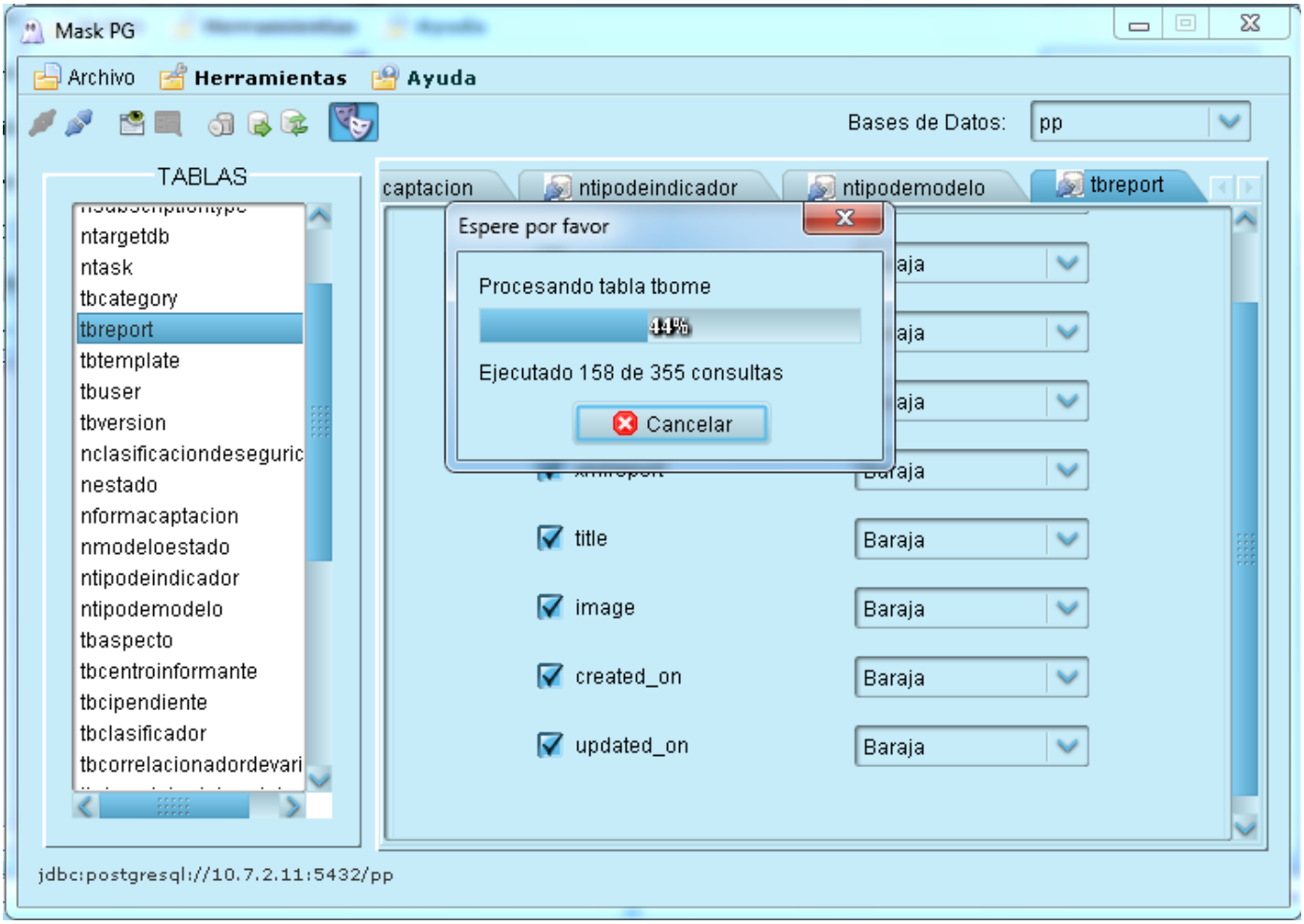
ANEXOS

Anexo 1: Interfaz de usuario. HU Autenticar usuario

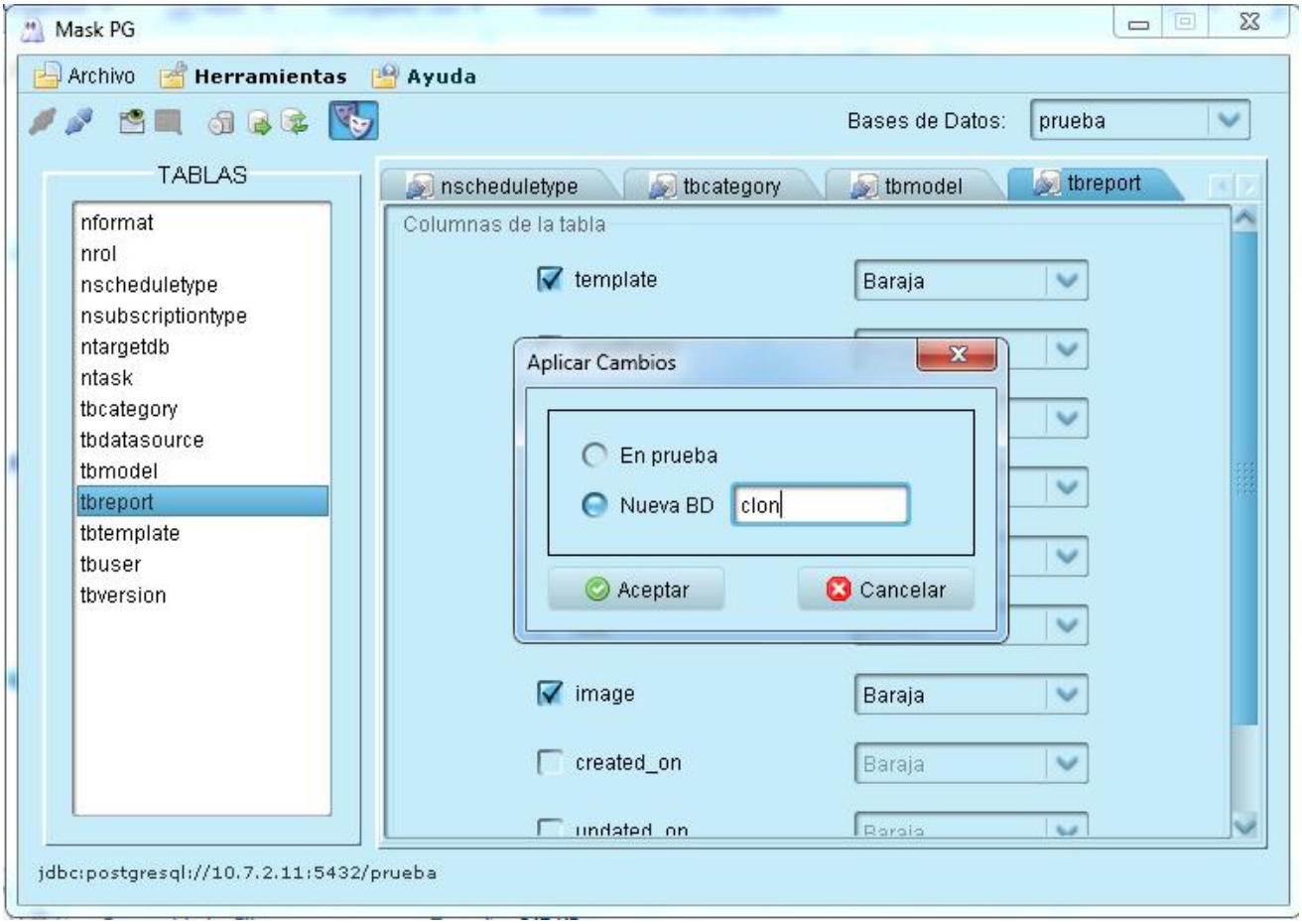
Anexo 2: Interfaz de usuario. HU Mostrar datos de las tablas seleccionadas



Anexo 3: Interfaz de usuario. HU Realizar el enmascarado de datos



Anexo 4: Interfaz de usuario. HU Realizar el clon de la base de datos origen



Anexo 5: Interfaz de usuario. HU Visualizar resultados del enmascarado