Universidad de las Ciencias Informáticas

FACULTAD 6

Título: Componente de Pentaho Data Integration para procesar cadenas de caracteres homogéneas

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Yusliel Sánchez Enriquez

Tutor: Ing. Yosbel Rodríguez Rodríguez

Cotutor: Lic. Lisdan Rodríguez Pérez

Cotutor: Ing. Joanni Acanda Velázquez

La Habana, junio del 2012

"Año 54 de la Revolución"

Estoy convencido de que la mitad de lo que separa a los emprendedores exitosos de los no exitosos es pura perseverancia.

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y le reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Yusliel Sánchez Enriquez

_____

Firma del Autor

Ing. Yosbel Rodríguez Rodríguez

_____

Firma del Tutor

Ing. Joanni Acanda Velázquez

_____

Firma del Cotutor

Lic. Lisdan Rodríguez Pérez

_____

Firma del Cotutor

DATOS DE CONTACTOS

Autor:
Yusliel Sánchez Enriquez
Universidad de las Ciencias Informáticas,
Ciudad de la Habana, Cuba
E-mail: ysenrique@estudiantes.uci.cu.

Tutor:
Ing. Yosbel Rodríguez Rodríguez
Universidad de las Ciencias Informáticas,
Ciudad de la Habana, Cuba
E-mail: yrdguezro@uci.cu.

Cotutores:
Lic. Lisdan Rodríguez Pérez
Universidad de las Ciencias Informáticas
Ciudad de la Habana, Cuba
E-mail: lisdanrp@uci.cu.

Ing. Joanni Acanda Velázquez
Universidad de las Ciencias Informáticas
Ciudad de la Habana, Cuba
E-mail: jacanda@uci.cu.

AGRADECIMIENTOS

A mis padres y a mis hermanos los cuales me han apoyado siempre, y han hecho que hoy yo esté aquí.

Al resto de mi familia que siempre han estado pendientes de mí.

A Yuneidy por ser tan atenta, dedicada, comprensiva y por estar siempre a mi lado.

A mis compañeros de grupo 6507, que entre vientos y marea llegamos hasta el final y a los que vienen en camino que sé que también van a llegar.

A todos mis amigos y amigas de la UCI, los de otros cursos, los que ya no están físicamente pero estarán en mi mente por el resto de la vida

A mis tutores Lisdan, Joanni y porque no a Yosbel que aunque no esté presente fue el que planteó la idea de hacer el plugin, gracias a todos los cuales siempre tuvieron la confianza de que algún día se iba a utilizar el plugin.

A todos los compañeros del Proyecto principales probadores de la Aplicación.

Al tribunal que siempre estuvo al pendiente señalando lo mal hecho con el objetivo de mejorar el resultado final del trabajo.

A todos, muchas gracias. No hubiera podido hacer esta tesis sin ustedes.

DEDICATORIA

A mi mamá Maricela Enriquez Rodríguez y a mi papá Antonio Sánchez Simón los cuales siempre estuvieron a mi lado guiándome por el mejor camino, y sufriendo traspié a traspié que me fue poniendo la vida y nota a nota que obtuve en la carrera.

A mis hermanos Yusniel y Yusley que me han servido de ejemplo en los estudios y en la vida.

A mi medio hermano Alberto que también ha puesto su granito de arena para lograr ser quien soy hoy.

A Irania y Joanni mis más grandes tutores y amigos en esta universidad.

A toda mi familia en general que siempre me ha apoyado para cumplir esta larga meta que me he propuesto.

Este título es de ustedes que fueron los que confiaron en mí y los que supieron guiarme por los mejores caminos, los quiero mucho y siempre estarán en mi corazón.

RESUMEN

El desarrollo en las últimas décadas ha conducido a una situación en la que el volumen de datos en las organizaciones crece significativamente. El trabajo con información digital se ha visto dificultado debido a los diferentes formatos existentes.

En la investigación se abordan aspectos basados en el estudio de la herramienta Pentaho Data Integration y métodos de procesamientos de cadenas de caracteres que posibilitan llegar a soluciones factibles para una mejor organización de la información. Se desarrolló un componente sobre entornos libres que permite cargar un fichero, seleccionar una vía de solución enfocado en una línea del mismo y a partir de esto procesar toda la información.

El proceso completo permite documentar una serie de resultados que permitirán una futura profundización sobre el tema y comparación con soluciones, tecnologías y herramientas semejantes. Además se describen las características de la aplicación y las actividades relacionadas con la metodología XP adoptada para cubrir su ciclo de desarrollo.

PALABRAS CLAVES: Cadena de caracteres, componente, homogéneas.

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLA

ÍNDICE DE FIGURA

INTRODUCCIÓN

La ciencia y la innovación tecnológica en la actualidad son factores importantes en el desarrollo económico y social para elevar la calidad de vida de la población de cualquier país del mundo.

En Cuba, el impacto social de la ciencia y la tecnología constituye un tema de gran interés, el país está en vías de desarrollo, en un profundo y novedoso proceso de transformaciones educacionales y sociales como programas de la batalla de ideas. En estas circunstancias surge la Universidad de las Ciencias Informáticas (UCI), la misma cuenta con muchos proyectos, que en su mayoría aportan considerables ingresos a la economía del país, con el desarrollo de software y servicios informáticos para la sociedad cubana y para el mundo.

En la UCI se creó el Centro de Tecnologías de Gestión de Datos (DATEC), dicho centro está compuesto por cuatro departamentos de productos de software, PostgreSQL, Bioinformática, Integración de Soluciones y Almacenes de Datos. La línea de Almacenes de Datos se dedica al análisis de los datos y al desarrollo de aplicaciones que automatizan el control estadístico de cualquier institución, además cuenta con 4 grupos de trabajo, Análisis, Diseño, Inteligencia de Negocio y Extracción, Transformación y Carga (ETL, por sus siglas en inglés).

La información en muchas de las organizaciones se encuentra en diferentes formatos (Excel, Access, Bases de datos), dificultando la estandarización de la información. Pentaho Data Integration es una herramienta de ETL que se utiliza para extraer, limpiar e integrar esta valiosa información y la pone en manos del usuario.

Provee una consistencia y una sola versión de todos los recursos de información, siendo esto uno de los más grandes desafíos para las organizaciones de la informática y las telecomunicaciones hoy en día. El uso de la herramienta permite evitar grandes cargas de trabajo manual, frecuentemente difícil de mantener y de desplegar.

Esta investigación surge producto del inconveniente que posee la herramienta Pentaho Data Integration de no proveer un componente que permita el trabajo con cadenas de caracteres sin un criterio de separación. Como consecuencia se debe recurrir al uso del componente Java Script Modificado, desde donde implementa la funcionalidad requerida, utilizando para ello el lenguaje Java Script. Este proceder presenta múltiples desventajas:

- La funcionalidad al estar encerrada en un componente genérico, hace prácticamente imposible el análisis del flujo de datos sin revisar directamente el código fuente dentro de este.
- De la misma forma se incrementa la probabilidad de cometer errores de codificación, los cuales son difíciles de detectar y de corregir.

- Resulta engorroso el proceso de adaptar la codificación existente en caso de variaciones en el formato de la cadena de caracteres interpretada, dificultando en extremo darle mantenimiento y haciéndolo poco flexible ante los cambios.
- Resulta difícil reutilizar el código y en el peor de los casos se deben implementar desde cero las funcionalidades requeridas.

Por lo expuesto anteriormente el **problema de la investigación** queda definido en la siguiente interrogante ¿Cómo extraer información de cadenas de caracteres homogéneas con la herramienta Pentaho Data Integration?

El **objeto de estudio** de este trabajo son los diferentes métodos de procesamiento de cadenas de caracteres.

El **campo de acción** está enfocado en el desarrollo de un componente de entrada para la herramienta Pentaho Data Integration que procese cadenas de caracteres.

Se presenta como **idea a defender** el desarrollo de un componente para la herramienta Pentaho Data Integration, que facilite la correcta extracción de información de cadenas de caracteres sin un criterio de separación.

Se define como **objetivo general**: Desarrollar un componente para la herramienta Pentaho Data Integration, que permita extraer información de cadenas de caracteres homogéneas.

Desglosándose en los siguientes **objetivos específicos**:

- Realizar un estudio preliminar del negocio.
- Realizar el análisis y diseño del componente para la extracción de información de cadenas de caracteres homogéneos.
- Implementar el componente para la extracción de información de cadenas de caracteres homogéneos.
- Realizar pruebas para comprobar el correcto funcionamiento del componente.

Para dar cumplimiento a los objetivos planteados se trazaron las siguientes **tareas de la investigación**.

1. Estudio de la arquitectura y funcionamiento de la herramienta Pentaho Data Integration, necesarios para el desarrollo del componente.
2. Estudio de los diferente métodos de procesamiento de cadenas de caracteres.
3. Caracterización de las metodologías, herramientas y tecnologías a utilizar en el desarrollo del componente.

4. Modelado de los conceptos, entidades y eventos más importantes del dominio, lo cual ayuda a entender con mayor claridad el negocio existente.
5. Realizar el análisis del sistema para una mayor comprensión de la posible solución.
6. Implementación de las funcionalidades del componente.
7. Aplicación de pruebas unitarias y de aceptación para garantizar el correcto funcionamiento del sistema.

Estructuración del trabajo de diploma

El presente trabajo de diploma se ha estructurado en introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía y glosario, para una mejor comprensión se da una breve descripción de cada capítulo.

Capítulo 1: Fundamentos Teóricos. En este capítulo se realiza un profundo análisis del estado del arte de la herramienta Pentaho Data Integration, se seleccionan las metodologías, herramientas y tecnologías a utilizar en la confección del componente.

Capítulo 2: Exploración y Planificación. En este capítulo se hace alusión a los conceptos relacionados con el negocio, además se utiliza como apoyo auxiliar un modelo de dominio que los refleja sintetizadamente, se presentan las Historias de Usuario capturadas en la fase de Exploración y la Planificación realizada a partir de éstas como parte del trabajo llevado a cabo en la fase de igual nombre.

Capítulo 3: Diseño, Codificación y Pruebas. En este capítulo se describen los aspectos asociados al diseño, la implementación y las pruebas de la aplicación. Se presentan artefactos como las tarjetas CRC (Clase, Responsabilidad, Colaboración), el diagrama UML, las tareas de programación, con el fin de facilitar la implementación. También se detallan las pruebas funcionales realizadas para verificar la correspondencia según lo acordado con el usuario.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS

Introducción

En el presente capítulo se realizará un estudio del estado del arte de la herramienta en cuestión, se realiza un profundo análisis de las tecnologías, metodologías, lenguajes, herramientas y métodos de la investigación a utilizar en el desarrollo del componente.

1.1 Estado del arte

En el año 2001, el belga Matt Casters empezó el desarrollo de una herramienta para uso personal, consciente de las dificultades que había tenido durante su experiencia laboral como constructor de Almacén de Datos para la integración de sistemas. Durante los siguientes años, fue desarrollando la herramienta, utilizando Java y sus librerías gráficas. La herramienta fue creciendo en funcionalidades, acceso a bases de datos, tratamiento de ficheros y componentes hasta llegar al 2004 con la versión 1.2. El proyecto fue subido al portal de alojamiento de proyectos de código abierto Javaforge, donde los usuarios podían descargarlo y utilizarlo. En la versión 2.0 se incluyó un sistema de componentes para permitir el desarrollo de conectores de Pentaho Data Integration con otros sistemas como por ejemplo los Sistemas, Aplicaciones y Productos en Procesamiento de datos (Sap, por sus siglas en inglés), en 2005 fue liberado el código y puesto a disposición de todos en Javaforge. El proyecto creció con rapidez y la comunidad se involucró en su desarrollo con mucha actividad, hasta entrar dentro de la órbita de Pentaho al ser vendido por el autor, que lo introdujo como herramienta ETL en su gama de productos. Matt Casters ha estado desde entonces trabajando en Pentaho y desarrollando su arquitectura como parte del equipo de Pentaho, interviniendo en las diferentes versiones hasta llegar a la 3.2 y el desarrollo de la nueva versión 4.0, que salió en el 2010. (Espinosa, 2010)

La arquitectura de Pentaho Data Integration viene representada por el siguiente esquema.

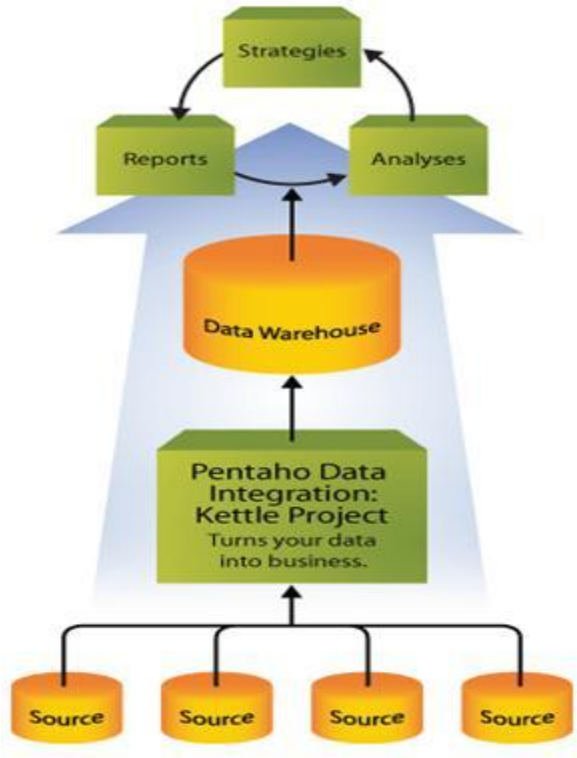

Figura 1: Arquitectura del Pentaho Data Integration

Propiedades

Además ser de código abierto y sin costos de licencia, las características básicas de esta herramienta son:

- Entorno gráfico de desarrollo
- Uso de tecnologías estándar: Java, XML, JavaScript
- Fácil de instalar y configurar
- Multiplataforma: Windows, Macintosh, Linux
- Basado en dos tipos de objetos: **Transformaciones** (colección de pasos en un proceso ETL) y **trabajos** (colección de transformaciones)

Pentaho Data Integration está formado por un conjunto de herramientas, cada una con un propósito específico.

Spoon: es la herramienta gráfica que nos permite el diseño de las transformaciones y trabajos. Incluye opciones para una previa visualización de los elementos desarrollados. Es la principal herramienta de trabajo de PDI con la cual se construirán y validarán los procesos ETL.

Pan: es la herramienta que permite la ejecución de las transformaciones diseñadas en Spoon (bien desde un fichero o desde el repositorio). También permite desde la línea de comandos preparar la ejecución mediante scripts.

Kitchen: similar a Pan, pero este se utiliza para ejecutar los jobs (trabajos).

Carte: es un pequeño servidor web que permite la ejecución remota de transformaciones y trabajos. (Díaz Ordaz, 2012)

Cuando se trabaja con Spoon, se tienen dos formas de guardar los elementos que fueron diseñados:

Repositorio: se dispone de una base de datos, con una estructura especial, donde son guardadas las transformaciones y trabajos construidos. Puede ser útil para el trabajo en equipo y para disponer de un lugar centralizado donde se va registrando todo lo realizado.

Ficheros: las transformaciones y trabajos son guardados a nivel del sistema de ficheros, en archivos XML (con extensión .ktr para las transformaciones y .kjb para los trabajos). Cada transformación y cada trabajo tienen un fichero asociado, que incluye en formato XML el metadato que define su comportamiento.

No se puede trabajar simultáneamente con los dos métodos, pero existe la opción de convertir de uno a otro modo utilizando componentes de PDI.

Ver ejemplo figura 2 Repositorio. (Espinosa, 2010)

Figura 2: Repositorio

Transformación

La transformación es el elemento básico del diseño de los procesos ETL en Pentaho Data Integration. Una transformación se compone de steps (pasos), que están enlazados entre sí a través de los hops (saltos). Los pasos son el elemento más pequeño dentro de las transformaciones. Los saltos constituyen el elemento a través del cual fluye la información entre los diferentes pasos (siempre es la salida de un paso y la entrada de otro).

Existe una amplia colección disponible de pasos que permite abordar casi cualquier necesidad en el diseño de los procesos de integración de datos. Los pasos están agrupados por categorías y cada uno

de ellos está diseñado para cumplir una función determinada. Cada paso tiene una ventana de configuración específica, donde se determinan los elementos a tratar y su forma de comportamiento.

Una transformación no es ningún programa ni un ejecutable, simplemente es un conjunto de metadatos en XML que le indican al motor de PDI las acciones a realizar. (Espinosa, 2010)

Trabajo

Al igual que las transformaciones, un trabajo no es ningún programa, es también un conjunto de metadatos en formato XML, que le describen al motor de PDI la forma de realizar las diferentes acciones. (Espinosa, 2010)

Un trabajo es similar al concepto de proceso. Un proceso es un conjunto sencillo o complejo de tareas con el objetivo de realizar una acción determinada. En los trabajos se pueden utilizar pasos específicos (que son diferentes a los disponibles en las transformaciones) como para recibir un fichero vía ftp, mandar un correo y ejecutar un comando. Además se puede ejecutar una o varias transformaciones de las que se hayan diseñado y organizar una secuencia de ejecución de ellas. Los trabajos estarían en un nivel superior a las transformaciones.

Interfaz de usuario

La interfaz de usuario es muy sencilla, disponiendo de dos perspectivas. Una vista donde se muestran los componentes que forman el trabajo o la transformación, y otra de diseño, donde se ven los pasos disponibles. Cuando se está trabajando con transformaciones o con trabajos, los pasos disponibles irán cambiando. En la figura 4 se muestra la perspectiva Diseño, a la izquierda se encuentran los diferentes pasos que se irán arrastrando a la sección de la derecha (red de diseño). Los pasos de transformaciones y de trabajos se irán enlazando con los correspondientes saltos.

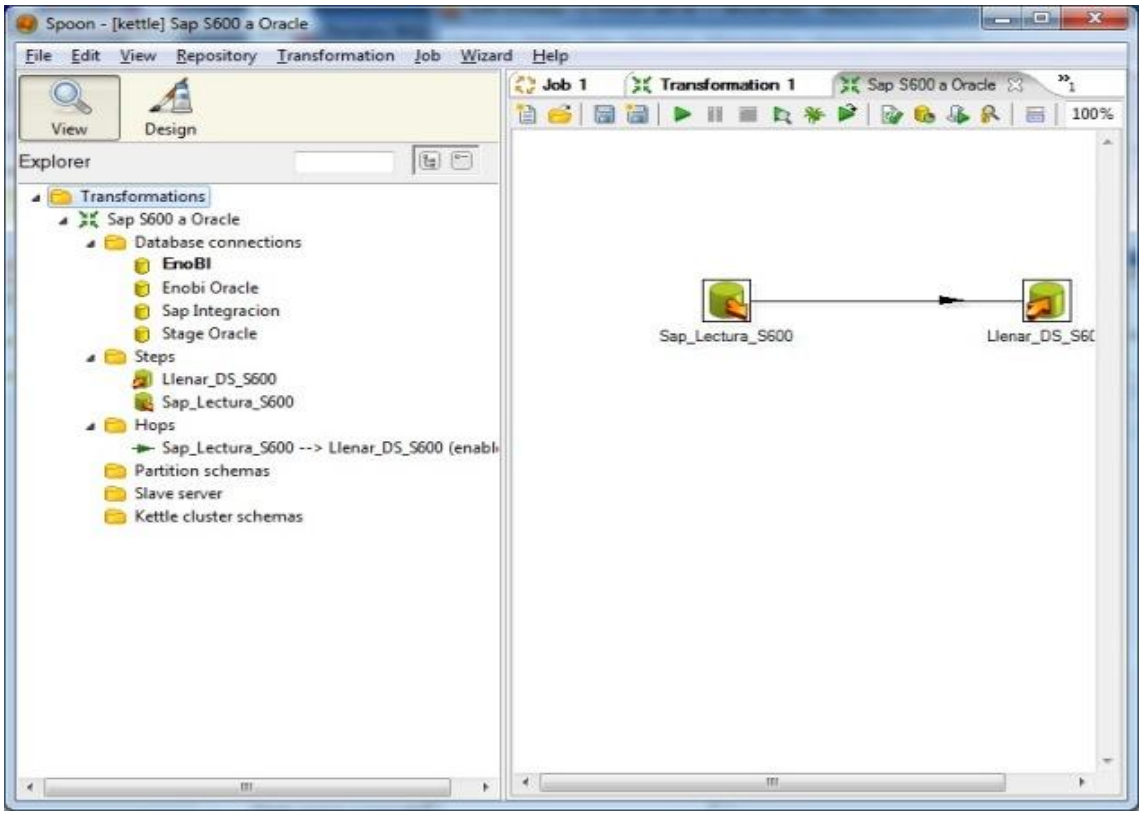

Figura 3: Perspectiva de Diseño

En la parte derecha de la pantalla (la cuadrícula), se puede acceder a las propiedades de cada elemento con doble clic sobre el componente o bien con el menú contextual del botón derecho del ratón. Todo de una forma muy sencilla e intuitiva. (Espinosa, 2010)

Cadenas de caracteres

Una cadena de caracteres es un conjunto de caracteres (incluido el espacio en blanco) que se almacenan en localidades contiguas de memoria. Se representa como un vector de caracteres donde cada elemento del vector representa un caracter de la cadena. (Walczuch, 2011)

Algunas de las aplicaciones del procesamiento de cadenas de caracteres son:

- Procesamiento del lenguaje natural
- Reconocimiento de patrones
- Visión artificial
- Comunicación entre agentes
- Teoría de juegos
- Lógica
- Bases de datos

- Aprendizaje en agentes
- Gramáticas y lenguajes

Las expresiones regulares son una manera potente y (mayormente) estandarizada de buscar, sustituir y analizar texto con patrones de caracteres complejos. Aunque la sintaxis de las expresiones regulares es compacta y diferente al código normal, el resultado puede acabar siendo más legible que una solución escrita a mano que use una larga serie de funciones de cadenas. (Pilgrim, 2009)

1.2 Metodologías de desarrollo

El desarrollo de todo software debe estar guiado por una metodología de desarrollo, de esta depende en gran medida que el software a desarrollar tenga la calidad requerida. La metodología que se escoja será la encargada de guiar en un orden lógico el desarrollo de un grupo de actividades. Para cumplirlas se utilizan herramientas, técnicas y modelos garantizando al concluir, un software con calidad. Existen dos grupos de metodologías en este tema, las Tradicionales y las Ágiles, no existe una metodología universal para cada tipo de proyecto, pues ésta generalmente se define según las características del equipo de desarrollo, el dominio de aplicación, el tipo de contrato, la complejidad y envergadura del proyecto, todos estos factores hacen necesario adaptar una metodología de desarrollo.

Metodologías tradicionales

Las metodologías tradicionales centran su atención en llevar una completa documentación, planificación y procesos de todo el proyecto.

La experiencia ha demostrado que, como consecuencia de las características del software, los resultados de los procesos no son siempre predecibles, también hay que tener en cuenta los altos costos al implementar un cambio y la falta de flexibilidad en proyectos donde el entorno es volátil.

Metodologías Ágiles

Las metodologías ágiles están especialmente orientadas para entornos variables, proyectos pequeños donde los individuos y las interacciones entre ellos, son más importantes que las herramientas y los procesos empleados y en donde se exige reducir drásticamente los tiempos de desarrollo manteniendo una alta calidad. La prioridad de este enfoque es satisfacer al cliente mediante tempranas y continuas entregas que le aporte un valor, por lo que es más importante crear un producto de software que funcione sin tener que escribir documentación exhaustiva, tributando con una elevada simplificación pero sin renunciar a las prácticas esenciales para asegurar la calidad del producto. (Canos, 2011)

Tabla 1: Comparación entre dos metodologías Agiles y dos Tradicionales

Metodología	Tamaño del proceso	Tamaño del equipo	Aprendizaje	Complejidad del problema	Tipo
RUP	Medio/Extenso	Medio/Extenso	Medio/Lento	Medio/Alto	T
MSF	Medio/Extenso	Pequeño /Extenso	Medio/Lento	Medio/Alto	T
XP	Pequeño/Medio	Pequeño	Rápido	Medio/Alto	A
SCRUM	Pequeño/Medio	Pequeño	Rápido	Medio/Alto	A

Tabla 2: Diferencias entre SCRUM y XP

SCRUM	XP
Las iteraciones de entregas son de 1 a 4 semanas.	Las iteraciones de entregas son de 1 a 3 semanas (algo más rápidas).
SCRUM es una metodología de desarrollo ágil basada en la administración del proyecto.	En cambio, XP se centra más en la propia programación o creación del producto.
Cada miembro del equipo trabaja de forma individual.	Los miembros del equipo trabajan en parejas.

Extreme Programming (XP)

Pertenece al grupo de metodologías ágiles, muy utilizada para producir software en breves espacios de tiempo y con un equipo reducido que se enfrenta a requisitos cambiantes. Hace énfasis en la adaptabilidad del proceso. En el desarrollo vincula estrechamente al usuario final con el equipo, propiciando un buen clima de trabajo como clave para lograr el éxito. Está basada en 5 valores básicos:

1. **Simplicidad:** enfocado más en un diseño sencillo del código generando sólo la documentación indispensable.
2. **Comunicación:** potenciada por el desarrollo en pares y la presencia del cliente, además de la simplicidad en cuanto al código.
3. **Retroalimentación:** propiciada por el protagonismo del cliente que participa activamente y por el trabajo en ciclos cortos.
4. **Coraje:** enfrentando decisiones en ocasiones complejas que pudieran afectar el tiempo de desarrollo y la calidad del producto.
5. **Respeto:** basado en estimar en toda su magnitud el trabajo de los demás. (Canos, 2011)

Sustentados en estos 5 valores los proyectos XP deben tener muy presente 24 prácticas valiosas, que se complementan unas con otras y que ofrecen una base sólida para un óptimo desempeño, alta productividad e inestimables beneficios. Entre ellas destacamos:

- Pruebas antes de programar.
- Diseño incremental.
- Ciclo semanal.
- Integración continua.
- Implicación real del cliente.

El ciclo de vida ideal consta de 6 fases

1. **Exploración:** en esta fase, los clientes plantean a grandes rasgos las Historias de Usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizan en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo.
2. **Planificación de Entregas:** en esta fase el cliente establece la prioridad de cada Historia de Usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente.
3. **Iteraciones:** esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de Entrega está compuesto por iteraciones de no más de tres semanas.
4. **Producción:** la fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente.
5. **Mantenimiento:** mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente.
6. **Muerte:** es cuando el cliente no tiene más Historias de Usuario para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. (Penadés, 2009)

Justificación de la metodología propuesta

Las metodologías ágiles son las más convenientes para este tipo de proyecto y en este caso XP, se ajusta a grupos de trabajos pequeños, simplificando el desarrollo de software y disminuyendo en tiempo y costo los esfuerzos invertidos en el proyecto. XP es mucho más fácil de implementar y de

aprender, por lo que los equipos jóvenes pueden incorporarla de manera más natural. La dimensión del proyecto es pequeña y está basado en el uso de nuevas tecnologías por lo que se suman los riesgos técnicos asociados a ellas, la planificación se realiza a corto plazo porque está sujeta a las necesidades de la producción donde se exigen entregas en breves intervalos, lo cual es perfectamente posible gracias a su naturaleza iterativa e incremental. Se harán pruebas, no sólo de cada nueva clase, sino que también los clientes comprobarán que el proyecto va satisfaciendo los requisitos.

1.3 Herramientas y Tecnologías utilizadas

Lenguaje de Programación

Un lenguaje de programación describe un conjunto de acciones que un equipo debe ejecutar. Está conformado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Al conjunto de instrucciones que se genera se conoce como código fuente de un programa. (Garcia, 2011)

Lenguaje de programación Java

Java es un lenguaje de plataforma independiente desarrollado por Sun Microsystems en los años 90. El lenguaje toma sintaxis de lenguajes como C y C++, pero a diferencia de este último, combina la sintaxis para programación genérica, estructurada y fue construido desde el principio para ser completamente orientado a objetos y todo reside en alguna clase. (Cursos, 2011)

Interpretado y compilado a la vez: Java es compilado, en la medida en que su código fuente se transforma en una especie de código máquina, los bytecodes, semejantes a las instrucciones de ensamblador. Por otra parte, es interpretado, ya que los bytecodes se pueden ejecutar directamente sobre cualquier máquina en la cual se encuentren el intérprete y el sistema de ejecución en tiempo real.

Indiferente a la arquitectura: Java está diseñado para soportar aplicaciones que serán ejecutadas en los más variados entornos de red, desde Unix a Windows NT pasando por Mac y estaciones de trabajo, sobre arquitecturas distintas y con sistemas operativos diversos. Para acomodar los requisitos de ejecución, el compilador de Java genera bytecodes: un formato intermedio indiferente a la arquitectura diseñada para transportar el código eficientemente a múltiples plataformas hardware y software. El resto de problemas los soluciona el intérprete de Java.

Portable: la indiferencia a la arquitectura representa sólo una parte de su portabilidad. Java especifica los tamaños de sus tipos de datos básicos y el comportamiento de sus operadores aritméticos, de manera que los programas son iguales en todas las plataformas. Estas dos últimas características se conocen como la Máquina Virtual Java (JVM). (Marañón, 1999)

Visual Paradigm 8.0

Visual Paradigm para UML es una herramienta profesional que soporta el 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 una rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Una de las características más importantes del Visual Paradigm es que es multiplataforma. (Larman, 2002)

Se integra con las siguientes herramientas: Eclipse/IBM WebSphere JBuilder NetBeans IDE Oracle JDeveloper BEA Weblogic, está disponible en varias ediciones, cada una destinada a unas necesidades: Enterprise, Professional, Community, Standard, Modeler y Personal. (Sierra, 2007)

IDE NetBeans 7.1

El IDE NetBeans 7.1 es un entorno integrado de desarrollo disponible para Windows, Mac, Linux y Solaris. NetBeans consiste en un IDE de código abierto y una plataforma de aplicaciones que permiten a los desarrolladores crear rápidamente aplicaciones web, escritorio y aplicaciones móviles utilizando la plataforma Java, así como JavaFX, PHP, Java Script y Ajax, Ruby y Ruby onRails, Groovy y Grails, y C/C++. NetBeans 7.1 introduce el Compositor de JavaFX, una herramienta de diseño visual para construir visualmente aplicaciones JavaFX interfaz gráfica de usuario, similar a la del constructor Swing GUI para aplicaciones Java SE con el compositor de JavaFX. Los desarrolladores en el Compositor de JavaFX pueden crear rápidamente, editar visualmente y depurar aplicaciones dinámicas de Internet y los componentes se unen a diversas fuentes de datos, incluidos los servicios Web. (Netbeans, 2012)

1.4 Métodos de investigación

La metodología de la presente investigación asume como principal criterio metodológico la concepción dialéctico materialista, garantizando así el desarrollo y evolución de este trabajo investigativo. Para el desarrollo de la investigación se utilizaron los métodos empíricos y teóricos.

Empíricos

Observación: la observación científica como método consiste en la percepción directa del objeto de investigación. La observación investigativa es el instrumento universal del científico. La observación permite conocer la realidad mediante la percepción directa de los objetos y fenómenos. La observación, como procedimiento, puede utilizarse en distintos momentos de una investigación más compleja: en su etapa inicial se usa en el diagnóstico del problema a investigar y es de gran utilidad en el diseño de la investigación.

Teóricos

Análisis y Síntesis: está integrado por el desarrollo del análisis y la síntesis , mediante el cual se descompone un objeto, fenómeno o proceso en los principales elementos que lo integran para analizar, valorar y conocer sus particularidades, y simultáneamente a través de la síntesis, se integran vistos en su interrelación como un todo.

El método comparativo: este permite establecer mediante la comparación las analogías y diferencias existentes entre los distintos objetos, fenómenos, procesos y sus propiedades.

Histórico – Lógico: a través de este método se establece la necesaria correspondencia entre los elementos de los métodos lógico e histórico, proyectando el análisis de la evolución histórica de los fenómenos, con la proyección lógica de su comportamiento futuro. (Sampieri, 1998)

1.5 Conclusiones

En este capítulo se realizó un análisis del tema a nivel nacional e internacional, se hizo una investigación de las principales metodologías de desarrollo de software, con el fin de justificar la empleada teniendo en cuenta las características del sistema. Por otra parte se realizó un estudio de las herramientas y tecnologías a utilizar para la realización de esta investigación, dentro de las cuales cabe destacar como IDE de desarrollo Netbeans 7.1, lenguaje de programación Java y para el modelado se usó Visual Paradigm en su versión 8.0.

CAPÍTULO 2: EXPLORACIÓN Y PLANIFICACIÓN

Introducción

En el presente capítulo se tiene como objetivo exponer el contexto en que se desarrolla el problema que da origen a las necesidades anteriormente planteadas. En términos de ingeniería se refiere al dominio alrededor del cual gira y facilita la comprensión de la situación en cuestión.

A partir de la metodología seleccionada se hace referencia a las cuestiones relacionadas con las fases de Exploración y Planificación, en la que se detallan las Historias de Usuario para establecer luego el orden en que serán implementadas atendiendo a su prioridad, para organizar el ciclo de vida del producto. Se hace alusión también a algunos artefactos generados que se requirieron complementariamente.

2.1 Descripción del Dominio

Un modelo del dominio es una representación de las clases conceptuales o entidades del mundo real. Esto ayuda a los usuarios, clientes y desarrolladores e interesados; a utilizar un lenguaje común para poder entender el contexto en que se desarrolla el sistema. Es decir, captura los tipos más importantes de objetos que existen o los eventos que suceden en el entorno donde estará el sistema y no incluyen las responsabilidades de las personas que ejecutan las actividades. La realización de un modelo de dominio surge por la necesidad de desarrollar el sistema, pues el negocio no está bien definido, no se puede describir el negocio con claridad, así como los casos de usos del negocio y sus trabajadores. (SI, 2011)

Para complementar el entendimiento de la dinámica asociada se hace uso de un modelo de dominio, aunque la metodología XP no propone artefactos en este sentido es flexible y puede contrastarse con algunos que simplifiquen y apoyen el proceso. Basado en esta definición, se muestra a continuación el modelo de dominio del componente para procesar cadenas de caracteres homogéneas para la herramienta Pentaho Data Integration.

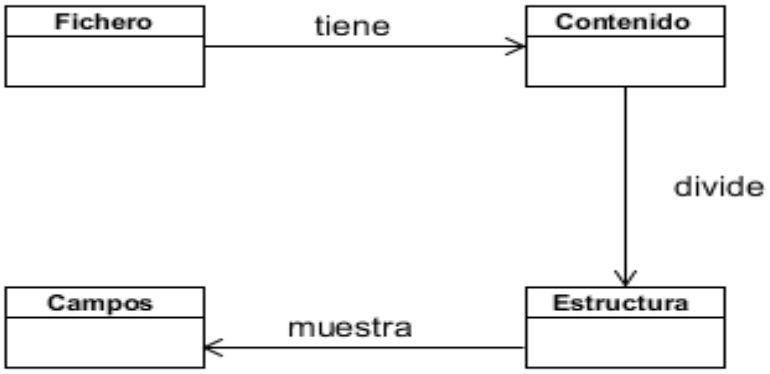

Figura 4: Modelo de Dominio

Propuesta de Solución

El presente trabajo pretende desarrollar un componente que permita analizar cadenas de caracteres las cuales actualmente no pueden ser procesadas por la herramienta.

El usuario realiza la entrada de los datos cargando un **Fichero** de texto. Una vez cargado el **Contenido**, escoge las líneas que desee procesar y después de esa selección pasa a completar las **Estructuras** para cada selección, posteriormente el sistema utiliza esta información y procesa automáticamente las demás líneas del fichero. Procesadas las estructuras se agrupa toda la información en **Campos** teniendo en cuenta los diferentes criterios seleccionados por el usuario del proyecto.

2.2 Exploración

El ciclo de vida ideal de un proyecto realizado con XP se inicia con la fase de Exploración, al final de la cual el equipo de desarrollo cuenta con suficiente material de trabajo traducido en Historias de Usuario, que se recopilan en esta etapa, como para producir una primera entrega. Además se produce el contacto necesario con las herramientas y tecnologías que se emplean para construir el sistema y algunas ideas experimentales. Se pueden explorar soluciones puntuales cuando no se tenga una concepción transparente de alguna funcionalidad. (SI, 2011)

Historias de Usuario

Una de las mejores prácticas adoptadas en el desarrollo de software es la administración de requerimientos. XP propone hacer uso de las Historias de Usuario (HU) como técnica para especificar las funcionalidades que brinda el sistema y constituye una manera dinámica de realizar esta actividad. Como su nombre lo indica son especificadas por los propios usuarios y por tanto redactadas en su

lenguaje; de manera sencilla y breve, evitando tecnicismos innecesarios que puedan crear confusión, aunque los programadores pueden contribuir en la tarea. Además son la base para realizar las pruebas de aceptación, así como la estimación y planificación del proyecto. Una vez identificadas las Historias de Usuario necesarias para liberar una primera versión operativa los programadores proceden a descomponer cada una en tareas específicas, las denominadas tareas de programación que están escritas técnicamente y que darán solución a su Historia de Usuario correspondiente. A continuación se muestra la plantilla utilizada para recoger la información relacionada con las Historias de Usuario determinadas.

Tabla 3: Plantilla de Historia de Usuario

<table>
<tr><th colspan="3">Historia de Usuario</th></tr>
<tr><td colspan="2">Nombre: Identifica la Historia de Usuario en cuestión.</td><td>Puntos (Ptos). Estimación: Permiten estimar duración de implementación.</td></tr>
<tr><td>No.: Números sucesivos a partir de 1.</td><td>Usuario: Quien ejecuta la Historia de Usuario.</td><td>Interacción: Precisa la iteración a la que pertenece la Historia de Usuario.</td></tr>
<tr><td colspan="3">Descripción: Explica en qué consiste la Historia de Usuario, teniendo en cuenta las acciones realizadas por el usuario y las respuestas brindadas por el sistema.</td></tr>
<tr><td colspan="2">Prioridad: Define la relevancia e impacto de la Historia de Usuario para el negocio de acuerdo a las necesidades del usuario.</td><td>Nivel de complejidad: Define la necesidad técnica que supone desarrollar la Historia de Usuario desde el punto de vista del programador.</td></tr>
<tr><td colspan="3">Información adicional: Información extra que se desee agregar para hacer más comprensible la Historia de Usuario. Por ejemplo: conceptos, pos condiciones.</td></tr>
</table>

Como resultado del trabajo realizado durante las fases de Exploración se identificaron las siguientes Historias de Usuario:

Tabla 4: HU Buscar Fichero

<table>
<tr><th colspan="3">Historia de Usuario</th></tr>
<tr><td colspan="2">Nombre: Buscar Fichero</td><td>Ptos. Estimación: 1</td></tr>
<tr><td>No.: 1</td><td>Usuario: Usuario del Proyecto</td><td>Interacción: 1</td></tr>
<tr><td colspan="3">Descripción: El usuario selecciona el archivo. El sistema debe mostrar la ubicación del archivo e informar en caso de error.</td></tr>
</table>

Prioridad: Alta	**Nivel de complejidad:** Baja
Información adicional:	

Tabla 5: HU Añadir Fichero

Historia de Usuario		
Nombre: Añadir Fichero		**Ptos. Estimación:** 1
No.: 2	**Usuario:** Usuario del Proyecto	**Interacción:** 1
Descripción: El usuario añade el fichero seleccionado. El sistema muestra la ubicación del archivo y la cantidad de líneas que contiene el fichero, también permite eliminar la selección.		
Prioridad: Alta	**Nivel de complejidad:** Media	
Información adicional:		

Tabla 6: HU Eliminar Fichero

Historia de Usuario		
Nombre: Eliminar Fichero		**Ptos. Estimación:** 1
No.: 3	**Usuario:** Usuario del Proyecto	**Interacción:** 1
Descripción: El usuario selecciona la opción eliminar fichero. En caso de no tener ningún fichero seleccionado a eliminar, el sistema muestra un mensaje de error.		
Prioridad: Baja	**Nivel de complejidad:** Media	
Información adicional:		

Tabla 7: HU Contenido Fichero

Historia de Usuario		
Nombre: Contenido Fichero		**Ptos. Estimación:** 1
No.: 4	**Usuario:** Usuario del Proyecto	**Interacción:** 1
Descripción: El usuario selecciona la opción Contenido Fichero. El sistema muestra en una forma visual el contenido del archivo seleccionado.		
Prioridad: Baja	**Nivel de complejidad:** Baja	
Información adicional:		

Tabla 8: HU Traer Líneas

Historia de Usuario		
Nombre: Traer Líneas		**Ptos. Estimación:** 1
No.: 5	**Usuario:** Usuario del Proyecto	**Interacción:** 2
Descripción: El usuario selecciona la opción Traer Líneas. El sistema muestra en una forma visual una lista con las líneas del fichero seleccionado y otra lista para que el usuario adicione las que desea procesar.		
Prioridad: Alta	**Nivel de complejidad:** Baja	
Información adicional: Tener en cuenta que cuando es un fichero procesado por Pentaho Data Integration, se haya escogido o no la primera línea del fichero esta se procesa para posteriormente asignar el nombre a sus respectivos campos.		

Tabla 9: HU Adicionar Líneas

Historia de Usuario		
Nombre: Adicionar Líneas		**Ptos. Estimación:** 1
No.: 6	**Usuario:** Usuario del Proyecto	**Interacción:** 2
Descripción: El usuario selecciona una o varias líneas y presiona la opción Adicionar. El sistema adiciona la o las líneas seleccionadas a procesar a una nueva lista.		
Prioridad: Alta	**Nivel de complejidad:** Media	
Información adicional: Tener en cuenta que cuando es un fichero procesado por Pentaho Data Integration, se haya escogido o no la primera línea del fichero, esta se procesa para posteriormente asignar el nombre a sus respectivos campos.		

Tabla 10: HU Adicionar Todas

Historia de Usuario		
Nombre: Adicionar Todas		**Ptos. Estimación:** 1
No.: 7	**Usuario:** Usuario del Proyecto	**Interacción:** 2
Descripción: El usuario selecciona la opción Adicionar Todas. El sistema adiciona todas las líneas a una nueva lista para ser procesadas.		

Prioridad: Alta	**Nivel de complejidad:** Media
Información adicional: Tener en cuenta que cuando es un fichero procesado por Pentaho Data Integration, se haya escogido o no la primera línea del fichero, esta se procesa para posteriormente asignar el nombre a sus respectivos campos. Si existiera una selección previa de algunas líneas se agregan todas las restantes a la nueva lista.	

Tabla 11: HU Eliminar Líneas

Historia de Usuario		
Nombre: Eliminar Líneas		**Ptos. Estimación:** 1
No.: 8	**Usuario:** Usuario del Proyecto	**Interacción:** 2
Descripción: El usuario selecciona una o varias líneas de la lista de selección y presiona la opción Eliminar. El sistema elimina las líneas de la selección y las adiciona a la lista original.		
Prioridad: Alta	**Nivel de complejidad:** Media	
Información adicional: Tener en cuenta que cuando es un fichero procesado por Pentaho Data Integration, se haya escogido o no la primera línea del fichero, esta se procesa para posteriormente asignar el nombre a sus respectivos campos.		

Tabla 12: HU Eliminar Todas

Historia de Usuario		
Nombre: Eliminar Todas		**Ptos. Estimación:** 1
No.: 9	**Usuario:** Usuario del Proyecto	**Interacción:** 2
Descripción: El usuario selecciona la opción Eliminar Todas. El sistema elimina todas las líneas de la selección y las adiciona a la lista original.		
Prioridad: Alta	**Nivel de complejidad:** Media	
Información adicional: Tener en cuenta que cuando es un fichero procesado por Pentaho Data Integration, se haya escogido o no la primera línea del fichero, esta se procesa para posteriormente asignar el nombre a sus respectivos campos. Si existiera una eliminación previa de algunas líneas o algunas líneas que no fueron seleccionadas, se agregan todas las restantes a la lista original.		

Tabla 13: HU Adicionar Selección

Historia de Usuario		
Nombre: Adicionar Selección		**Ptos. Estimación:** 1
No.: 10	**Usuario:** Usuario del Proyecto	**Interacción:** 2
Descripción: El usuario selecciona en la opción separador de campos, la opción (No), posteriormente selecciona manualmente la información de la primera línea del fichero y el sistema la adiciona a una lista que se muestra visualmente. En el caso de que el usuario presione en la opción separador de campos, la opción (Si) el sistema selecciona la información por un criterio entrado por el usuario, en el caso de que el usuario no entre el delimitador, el sistema muestra un mensaje de error.		
Prioridad: Alta	**Nivel de complejidad:** Alta	
Información adicional: Tener en cuenta que cuando es un fichero procesado por Pentaho Data Integration, se haya escogido o no la primera línea del fichero, esta se procesa para posteriormente asignar el nombre a sus respectivos campos.		

Tabla 14: HU Eliminar Selección

Historia de Usuario		
Nombre: Eliminar Selección		**Ptos. Estimación:** 1
No.: 11	**Usuario:** Usuario del Proyecto	**Interacción:** 2
Descripción: El usuario elimina de la lista la información que no desee procesar. En caso de no estar seleccionada ninguna información o estar vacía la lista, el sistema muestra un mensaje de error.		
Prioridad: Media	**Nivel de complejidad:** Baja	
Información adicional:		

Tabla 15: HU AdicionarER

Historia de Usuario		
Nombre: Adicionar Expresión Regular		**Ptos. Estimación:** 1
No.: 12	**Usuario:** Usuario del Proyecto	**Interacción:** 3
Descripción: El usuario adiciona o escribe la información en el campo y el sistema la adiciona		

a una lista que se muestra visualmente. En caso de no estar seleccionada ni escrita ninguna información en el campo el sistema muestra un mensaje de error.	
Prioridad: Alta	**Nivel de complejidad:** Baja
Información adicional: Esta información es muy delicada puesto que si no se entra la expresión regular que valide lo que en realidad el usuario quiere entonces el resultado no será el esperado.	

Tabla 16: EliminarSeleccionER

Historia de Usuario		
Nombre: Eliminar Selección Expresión Regular		**Ptos. Estimación:** 1
No.: 13	**Usuario:** Usuario del Proyecto	**Interacción:** 3
Descripción: El usuario elimina de la lista de expresiones regulares la que no desee procesar. En caso de no estar seleccionada ninguna información o estar vacía la lista, el sistema muestra un mensaje de error.		
Prioridad: Alta	**Nivel de complejidad:** Baja	
Información adicional:		

Tabla 17: SelecionarER

Historia de Usuario		
Nombre: Seleccionar Expresión Regular		**Ptos. Estimación:** 1
No.: 14	**Usuario:** Usuario del Proyecto	**Interacción:** 3
Descripción: El usuario selecciona información y el sistema la adiciona al campo donde Definir Expresión Regular. En caso de no estar seleccionada ninguna información el sistema muestra un mensaje de error.		
Prioridad: Alta	**Nivel de complejidad:** Baja	
Información adicional: Esta información es muy delicada puesto que si no se entra la expresión regular que valide lo que en realidad el usuario quiere entonces el resultado no será el esperado.		

Tabla 18: InsertarER

Historia de Usuario

Nombre: Insertar Expresión Regular		**Ptos. Estimación:** 1
No.: 15	**Usuario:** Usuario del Proyecto	**Interacción:** 3
Descripción: El usuario escribe una información y la inserta en el sistema para utilizarla en procesos futuros. En caso de estar el campo vacio el sistema muestra un mensaje de error.		
Prioridad: Alta	**Nivel de complejidad:** Media	
Información adicional: Esta información es muy delicada puesto que si no se entra la expresión regular que valide lo que en realidad el usuario quiere entonces el resultado no será el esperado.		

Tabla 19: EliminarER

Historia de Usuario		
Nombre: Eliminar Expresión Regular		**Ptos. Estimación:** 1
No.: 16	**Usuario:** Usuario del Proyecto	**Interacción:** 3
Descripción: El usuario selecciona una información y es eliminada del sistema. En caso de no estar seleccionada ninguna información el sistema muestra un mensaje de error.		
Prioridad: Alta	**Nivel de complejidad:** Media	
Información adicional: Esta información es muy delicada puesto que si no se entra la expresión regular que valide lo que en realidad el usuario quiere entonces el resultado no será el esperado.		

Tabla 20: HU Procesar Estructura

Historia de Usuario		
Nombre: Procesar Estructura		**Ptos. Estimación:** 2
No.: 17	**Usuario:** Usuario del Proyecto	**Interacción:** 3
Descripción: El usuario manualmente selecciona el tipo de dato correspondiente y llena los campos requeridos, presiona el botón procesar estructura y el mismo analiza la información, la procesa y la muestra en la ventana campos. En el caso de que el usuario presione el botón procesar información y exista algún campo vacio o no esté seleccionada ninguna información, el sistema muestra un mensaje de error.		
Prioridad: Alta	**Nivel de complejidad:** Alta	

Información adicional:

Tabla 21: Pre_visualizar Filas

Historia de Usuario		
Nombre: Pre_visualizar Filas		**Ptos. Estimación:** 1
No.: 18	**Usuario:** Usuario del Proyecto	**Interacción:** 3
Descripción: El usuario selecciona la opción Pre_visualizar Filas. El sistema muestra en una forma visual el contenido de las filas resultantes.		
Prioridad: Alta	**Nivel de complejidad:** Baja	
Información adicional:		

2.3 Prototipos de Interfaz de Usuarios

A continuación se muestran los prototipos de interfaz de usuarios correspondientes con cada Historia de Usuario.

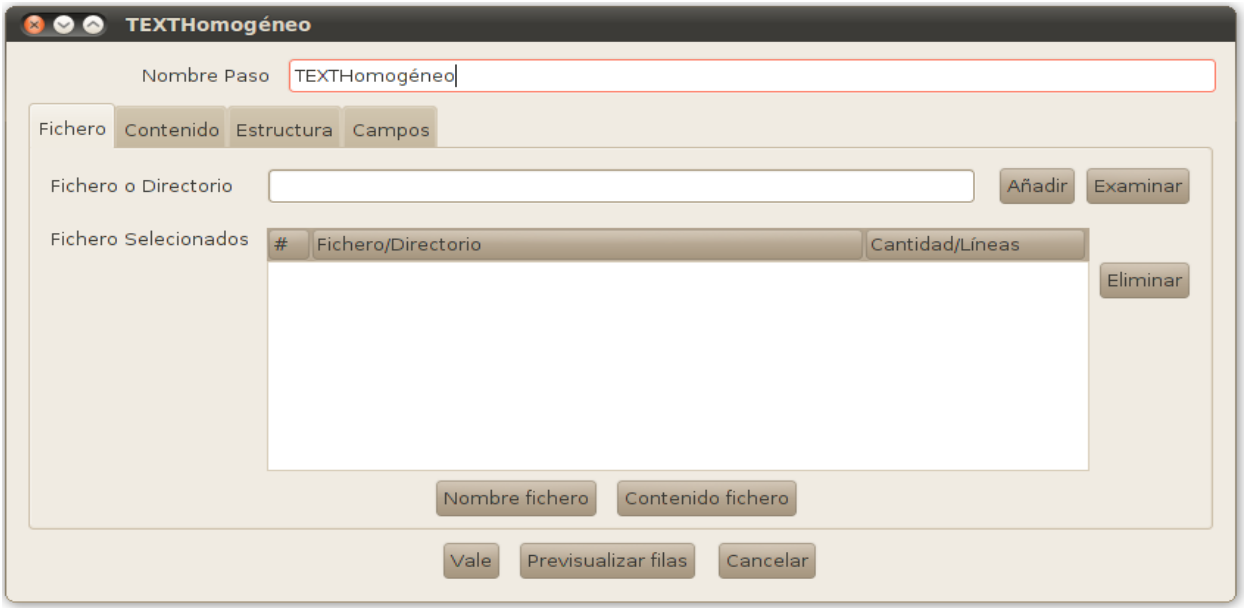

Figura 5: Historias de usuario 1, 2, 3, 4

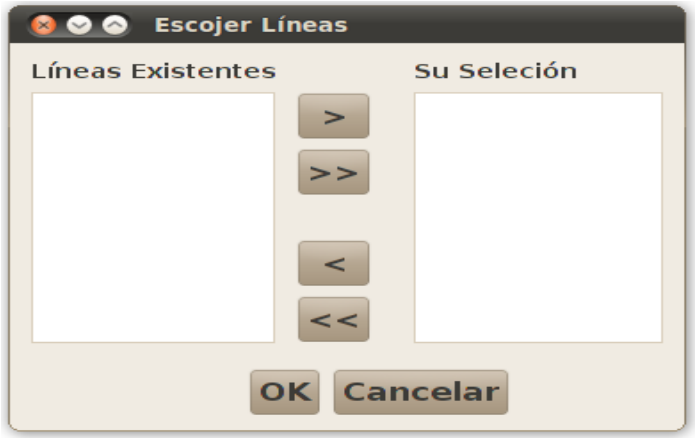

Figura 6: Historias de usuario 5, 6, 7, 8, 9

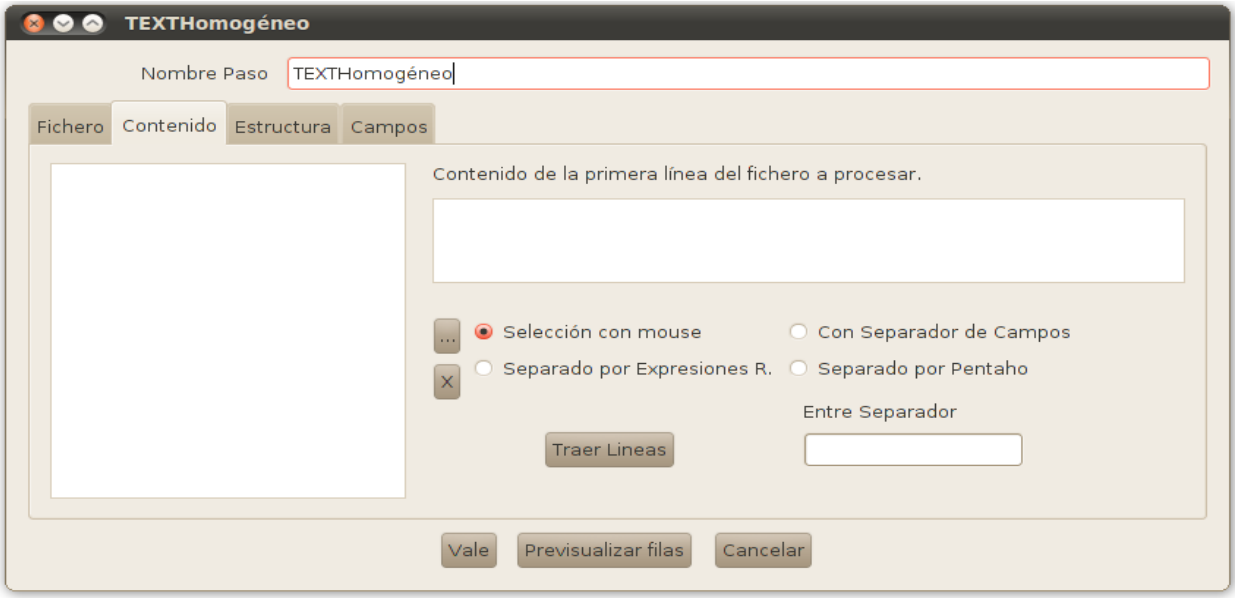

Figura 7: Historias de usuario 10, 11

Figura 8: Historias de usuario 12, 13, 14, 15, 16

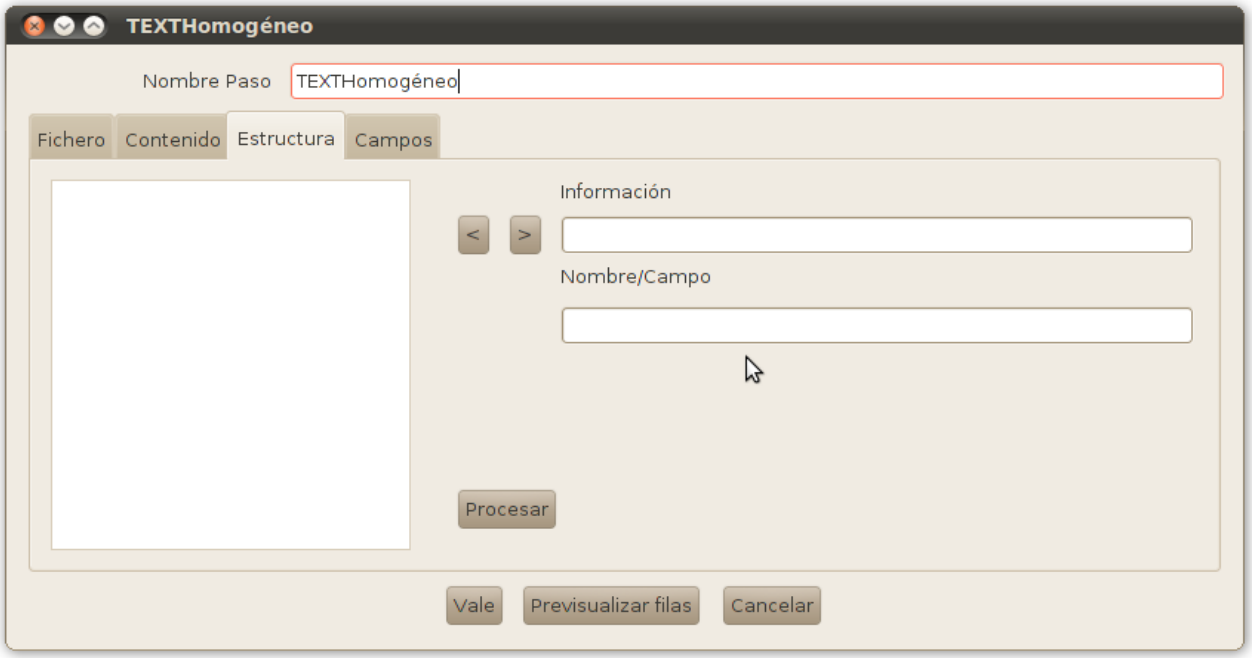

Figura 9: Historia de usuario 17

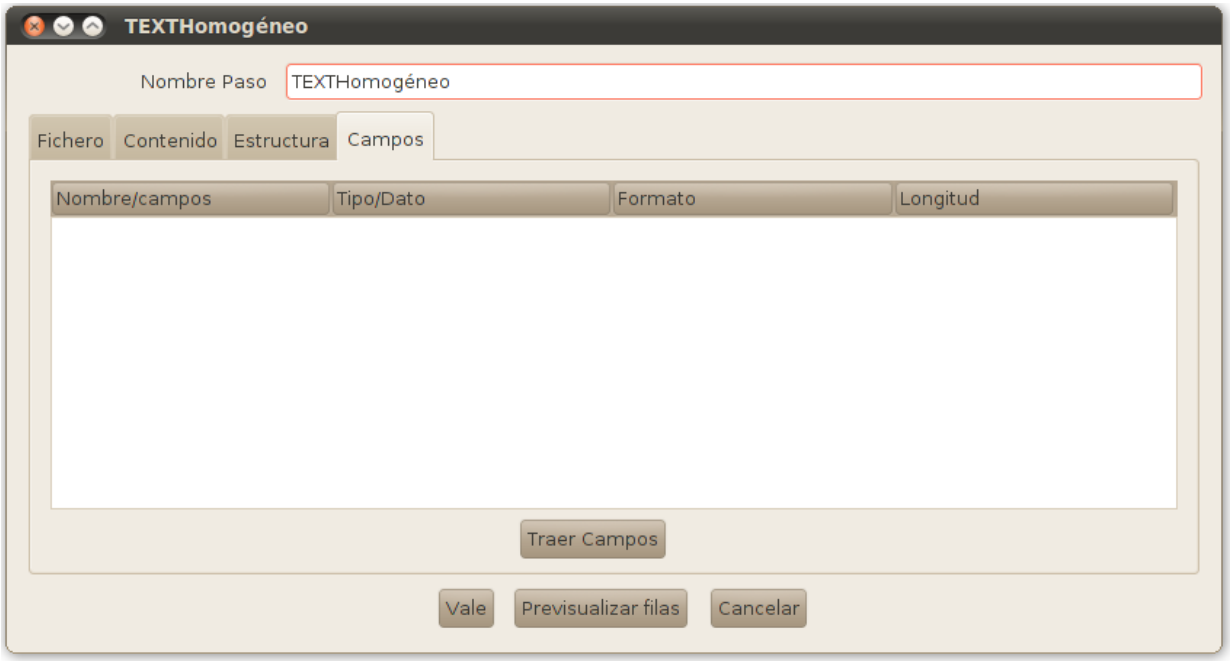

Figura 10: Historia de usuario 18

2.4 Requisitos No funcionales

Los requisitos no funcionales (RNF) son propiedades o cualidades que el producto debe tener. Estas propiedades o cualidades se refieren a las características que hacen al producto atractivo, usable, rápido o confiable. Por lo general los requisitos no funcionales son fundamentales en el éxito del producto; normalmente están vinculados a los requisitos funcionales, es decir, una vez que se conoce lo que el sistema debe hacer se puede determinar cómo ha de comportarse, qué cualidades o propiedades debe tener. (Requirements, 2011)

Los requisitos no funcionales: han de especificarse cuantitativamente, siempre que sea posible para que se pueda verificar su cumplimiento.

Requisitos de Software

- La herramienta Pentaho Data Integration.
- Máquina virtual de Java versión 1.5 o superior.

Requisitos de Hardware

- Procesador: Celeron, 2.0GHz
- Memoria RAM: 128Mb o superior.
- Espacio libre en disco duro: 200Mb

Restricciones del diseño y la implementación

- Se hace uso de la herramienta Pentaho Data Integration en su versión 4.2 y IDE NetBeans 7.1.

- El lenguaje de programación que será usado para la implementación es Java siguiendo el paradigma de la Programación Orientada a Objeto.

Requisitos de Usabilidad

1. Facilidad de uso por parte de los usuarios: La interfaz debe ser lo más descriptiva posible, permitiendo que las operaciones a realizar por los usuarios estén bien descritas, de manera que se puedan entender claramente.
2. La interfaz debe tener mensajes contextuales asociados a los objetos.

Requisitos de Portabilidad

- El plugin una vez integrado a la herramienta Pentaho Data Integration, se podrá utilizar en diferentes sistemas operativos por ser Pentaho Data Integration una herramienta multiplataforma.(forums.pentaho, 2010)

2.5 Planificación

A lo largo de esta fase los clientes establecen la prioridad de las Historias de Usuario de acuerdo a sus necesidades más inmediatas para luego asignarlas, por orden de relevancia, a las iteraciones planificadas. A partir de las Historias de Usuario se realiza la estimación del esfuerzo que se requiere por parte del equipo para su posterior implementación. El método escogido para realizar la estimación tiene como elemento el punto, el cual equivale a una semana perfecta de trabajo, esto se refiere a que solamente el equipo se dedica a labores relacionadas con la construcción del sistema sin la influencia o el retraso provocado por otros factores, lo que en la práctica es complejo lograr. (SI, 2011)

Estimación de esfuerzos por Historias de Usuario

Las estimaciones del esfuerzo para implementar las Historias de Usuario permiten tener una medida bastante real de la velocidad de progreso del proyecto y brindan una guía razonable a la cual ajustarse. Los resultados estimados se exhiben seguidamente.

Tabla 22: Puntos de Estimación por Historias de Usuario

No.	Historia de Usuario	Puntos de Estimación
1	Buscar Fichero	1
2	Añadir Fichero	1
3	Eliminar Fichero	1
4	Contenido Fichero	1
5	Traer Líneas	1
6	Adicionar Líneas	1

7	Adicionar Todas	1
8	Eliminar Líneas	1
9	Eliminar Todas	1
10	Adicionar Selección	1
11	Eliminar Selección	1
12	Adicionar Expresión Regular	1
13	Eliminar Selección Expresión Regular	1
14	Seleccionar Expresión Regular	1
15	Insertar Expresión regular	1
16	Eliminar Expresión Regular	1
17	Procesar Estructura	2
18	Pre_visualizar Filas	1

2.6 Plan de Liberaciones

El Plan de Liberaciones (entregas) tiene como objetivo definir el número de entregas que se realizarán en el transcurso del proyecto y las iteraciones que se requieren para desarrollar cada una. El cliente se encarga de decidir cuáles Historias de Usuario comprende la primera entrega según sus prioridades para darle valor a su negocio y que por tanto justifique su ejecución y así sucesivamente para las demás. Se decide utilizar un plan con fecha fija debido a las restricciones que impone el ambiente de producción. Para realizar el plan de entregas, el sistema propuesto se divide en módulos que contienen las Historias de Usuario relacionadas lógicamente de acuerdo a su propósito. Cada una de las entregas se corresponde con una iteración de desarrollo.

Tabla 23: Historias de Usuario por Módulos

Módulo	Historias de Usuario
Entrada fichero de texto	Buscar Fichero
	Añadir Fichero
	Eliminar Fichero
	Contenido Fichero
	Traer Líneas
	Adicionar Líneas
	Adicionar Todas
	Eliminar Líneas
	Eliminar Todas

	Adicionar Selección
	Eliminar Selección
	Adicionar Expresión Regular
	Eliminar Selección Expresión Regular
	Seleccionar Expresión Regular
	Insertar Expresión regular
	Eliminar Expresión Regular
	Procesar Estructura
	Pre_visualizar Filas

El plan de liberaciones previsto finalmente queda estructurado en 3 iteraciones, el mismo consta de 18 Historias de Usuario y como consecuencia de esa labor se obtienen los resultados en forma de versiones operacionales, expuestos en la siguiente tabla.

Tabla 24: Plan de Entregas

Módulo	**Iteración 1** (05/03/2012)- (23/03/2012)	**Iteración 2** (26/03/2012)- (13/04/2012)	**Iteración 3** (23/04/2012)- (11/05/2012)
Cadenas Homogéneas	**V0.1**	**V0.2**	**V1.0Final**

2.7 Plan de Iteraciones

En el plan de liberaciones se establece cuántas iteraciones serán necesarias realizar sobre el sistema para su éxito. Sin embargo, se precisa establecer el contenido de trabajo para todas y cada una de ellas y es aquí donde hace acto de presencia el plan de iteraciones, regulando la cantidad de Historias de Usuario a implementar dentro del rango establecido por la estimación efectuada. Tomando como referencia los aspectos antes tratados, el componente que se pretende construir se desarrollará en 3 iteraciones, explicadas más detalladamente a continuación:

Iteración 1

La iteración presente tiene como finalidad implementar las Historias de Usuario que se consideraran más necesarias atendiendo a su relevancia e impacto para el negocio y además establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Se da respuesta cabal a las funcionalidades de Buscar Fichero, Añadir Fichero, Eliminar Fichero y por último Contenido del Fichero.

Iteración 2

En esta iteración se permite adicionar y eliminar la información requerida después de seleccionar la o las líneas implicadas en el proceso. Se implementan las funcionalidades adicionar selección, eliminar selección, traer líneas, adicionar líneas, adicionar todas, eliminar líneas y eliminar todas pertenecientes a las Historia de Usuarios 5, 6, 7, 8, 9, 10 y 11. Finalizada esta iteración sale una versión con más de un 60% de la implementación del componente.

Iteración 3

Esta iteración se centra en dar solución a la Historia de Usuario Adicionar Expresión Regular, Eliminar Selección Expresión Regular, Seleccionar Expresión Regular, Insertar Expresión regular, Eliminar Expresión Regular, Procesar Estructura y Pre_visualizar Filas pertenecientes a las Historia de Usuarios 12, 13, 14, 15, 16, 17, 18, siendo la HU-18 Pre_visualizar Filas la de más alto nivel de prioridad en todo el proceso. Contemplan la realización de las acciones relacionadas con el procesamiento de la información, basado en el análisis de las estructuras y una posterior visualización de las filas resultantes. Al finalizar la iteración se obtiene un componente funcional listo para integrar a la herramienta.

Plan de duración de las iteraciones

XP es una de las metodologías que más enfatiza el uso de pruebas, proponiendo esta práctica como una de sus prioridades fundamentales para verificar la calidad y aplicándola desde el comienzo hasta la conclusión del proyecto. A modo general dentro de cada iteración se ejecutan tanto las pruebas unitarias como las de aceptación para tener la certeza de que se implementaron correctamente los requerimientos. Las funcionalidades que surgen con las ya elaboradas también se chequean mediante las pruebas de integración. Se debe integrar continuamente, y así se garantiza el seguimiento de cada una de las etapas lo cual tributa a las pruebas de regresión. A modo de resumen se presenta la siguiente tabla que muestra las 3 iteraciones analizadas previamente con las Historias de Usuario y su duración.

Tabla 25: Plan de duración de las iteraciones

Iteraciones	Historias de Usuario a implementar	Duración
Iteración 1	Buscar Fichero	3 semanas
	Añadir Fichero	
	Eliminar Fichero	
	Contenido Fichero	
Iteración 2	Traer Líneas	3 semanas

	Adicionar Líneas	
	Adicionar Todas	
	Eliminar Líneas	
	Eliminar Todas	
	Adicionar Selección	
	Eliminar Selección	
Iteración 3	Adicionar Expresión Regular	3 semanas
	Eliminar Selección Expresión Regular	
	Seleccionar Expresión Regular	
	Insertar Expresión regular	
	Eliminar Expresión Regular	
	Procesar Estructura	
	Pre_visualizar Filas	

2.8 Conclusiones

En este capítulo se efectuó un análisis exhaustivo de los conceptos relacionados con el negocio planteado empleándose para ello un modelo de dominio ilustrativo que sirvió de apoyo para una familiarización más adecuada. Se abordaron las peculiaridades de las fases de Exploración y Planificación y los artefactos que se generaron durante su desarrollo, entre ellos las Historias de Usuario y los planes de entregas e iteraciones. Ambas fases se repiten en cada iteración lo que posibilita realizar una estimación más exacta y real del esfuerzo necesario para cumplir con las Historias de Usuario negociadas y con las que el equipo de trabajo se ha comprometido.

CAPÍTULO 3: DISEÑO, CODIFICACIÓN Y PRUEBAS

Introducción

En el presente capítulo se hace referencia a los elementos que conforman el diseño del sistema a construir. Se describen las pruebas de aceptación que se le aplicaron al componente para verificar su correcto funcionamiento y que este responda a las necesidades del proyecto. Se hace alusión a algunos artefactos generados que se requirieron complementariamente.

3.1 Diseño

XP establece prácticas especializadas que inciden directamente en la realización del diseño para lograr un sistema robusto y reutilizable tratando de mantener su simplicidad, es decir, crear un diseño evolutivo que se va mejorando incrementalmente y que permite hacer entregas pequeñas y frecuentes de valor para el cliente. A la hora de darle cumplimiento a la actividad de diseñar, XP no especifica ninguna técnica de modelado, puede utilizarse indistintamente sencillos esquemas en una pizarra, diagramas de clases utilizando UML o tarjetas CRC (Clase, Responsabilidad y Colaboración) siempre que sean útiles, tributen a la comprensión y no requieran mucho tiempo en su creación.

Tarjetas CRC

Se usaron las tarjetas CRC como complemento para realizar el diseño del sistema a implementar. Esta técnica de modelado permite entender las características del sistema pensando en términos de objetos y clases. Se utilizó la estrategia tormenta de ideas que representa una parte importante en el uso de las tarjetas y constituye el paso inicial del proceso. La plantilla de tarjeta CRC utilizada y los resultados obtenidos se describen a continuación.

Tabla 26: Plantilla de Tarjeta CRC

Responsabilidades	Clases relacionadas
Es una descripción de alto nivel de la o las responsabilidades de una clase.	Indica con cuáles otras clases requiere relación para cumplir la responsabilidad.

Tabla 27: Tarjeta CRC Fichero

Responsabilidades	Clases relacionadas
Examinar Fichero	Principal
Añadir Fichero	Principal
Eliminar Fichero	Principal

Tabla 28: Tarjeta CRC Contenido

Responsabilidades	Clases relacionadas
Contenido Fichero	Principal, Ver contenido
Traer Líneas	Principal, Traer Líneas
Adicionar Líneas	Principal, Traer Líneas
Adicionar Todas	Principal, Traer Líneas
Eliminar Líneas	Principal, Traer Líneas
Eliminar Todas	Principal, Traer Líneas

Tabla 29: Tarjeta CRC Estructura

Responsabilidades	Clases relacionadas
Procesar Estructura	Principal
Pre_visualizar Filas	Principal, Pre visualizar Filas

Tabla 28: Tarjeta CRC Selección

Responsabilidades	Clases relacionadas
Adicionar Selección	Principal
Eliminar Selección	Principal

Estructura del componente

El proyecto está compuesto por varios paquetes. Su peculiar estructura se muestra a continuación:

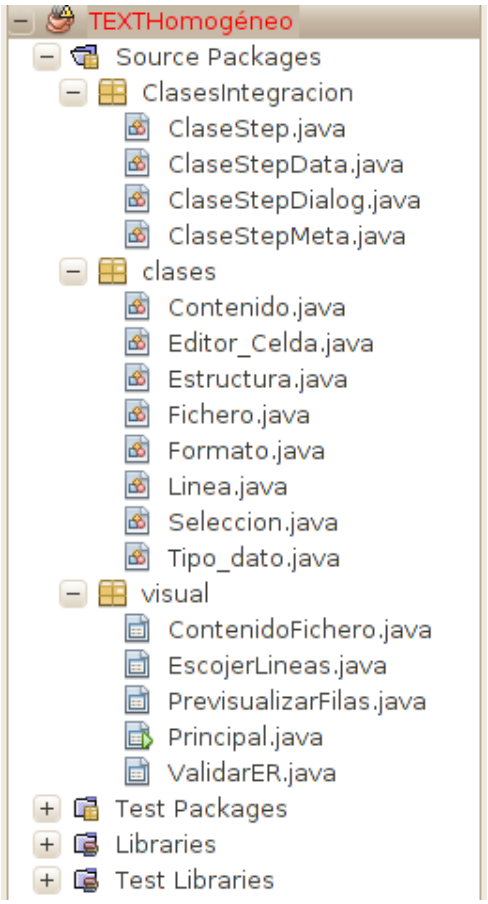

Figura 11: Estructura del componente

El paquete **ClasesIntegracion** es el encargado de la integración del flujo de información con la herramienta.

Arquitectura de plugin para Pentaho Data Integration

ClaseStepData: la clase de datos se utiliza para el almacenamiento de datos únicos a un hilo de ejecución, cuando el complemento se ejecuta. Aquí es donde las conexiones de base de datos almacenan en caché los identificadores de archivos, y otras cosas necesarias durante la ejecución.

ClaseStepMeta: la clase implementa la StepMetaInterface. Es responsable para sostener y fabricar en serie las escenas escogidas para un caso particular del paso. Sostiene el nombre del paso y el nombre del campo del rendimiento para nuestro paso plantilla.

ClaseStep: la clase paso implementa el StepInterface. Las instancias de esta clase hacen el procesamiento de filas real cuando se ejecuta la transformación. Cada hilo de ejecución está representado por una instancia de esta clase. Se administra las instancias de las clases de datos y meta, cuando se ejecuta.

ClaseStepDialog: la clase de diálogo implementa la interfaz de usuario del paso. Se muestra un cuadro de diálogo que permite al usuario configurar el comportamiento del paso de su agrado. Esta clase de diálogo está estrechamente relacionado con la clase meta, que realiza un seguimiento de los ajustes seleccionados. (Chodnicki, 2010)

Además del código, los plugin tienen otras características que permiten la unión del plugin con la herramienta, accesible a través de la interfaz del usuario. Existe un fichero (plugin.xml) que guarda toda la información externa necesaria para la visualización del plugin.

Escribiendo el archivo de plugin.xml

La configuración del archivo plugin.xml es bastante simple. Su función principal es decirle a la herramienta la clase que inicia el proceso (la clase meta), la descripción y localización del plugin.

```
plugin.xml
<?xml version="1.0" encoding="UTF-8"?>
<plugin

   id="TEXTHomogéneo"
   iconfile="TH_logo.jpg"
   description="TEXTHomogéneo"
   tooltip="Componente para extraer información de cadenas homogéneas"
   category="Entrada"
   classname="ClasesIntegracion.ClaseStepMeta">

               <libraries>
               <library name="TEXTHomog_neo.jar"/>
               </libraries>

</plugin>
```

Figura 12: Código XML para la incorporación del plugin

El identificador para el plugin debe ser globalmente único, y no debe cambiarse, como él se usa en la serialization. El archivo del icono es una imagen de extensión png. “La descripción” es el nombre del paso como aparece en el menú del árbol. “La categoría” es el nombre de la carpeta del árbol en la que aparece el plugin. (Chodnicki, 2010)

Patrones de diseño utilizados

En el diseño orientado a objetos, los Patrones Generales de Asignación de Responsabilidad de Software (GRASP) como su nombre lo indica son patrones generales de software para asignación de responsabilidades. Aunque se considera que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software.

Bajo Acoplamiento: debe haber pocas dependencias entre las clases. Si todas las clases dependen de todas. Uno de los principales síntomas de un mal diseño y alto acoplamiento es una herencia muy profunda. Siempre hay que considerar las ventajas de la delegación respecto de la herencia.

Experto: la responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada. Hay que tener en cuenta que esto es aplicable mientras estemos considerando los mismos aspectos del sistema.

Alta Cohesión: cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable.

Creador: se asigna la responsabilidad de que una clase B cree un Objeto de la clase A solamente cuando.

B contiene a A

B es una agregación (o composición) de A

B almacena a A

B tiene los datos de inicialización de A (datos que requiere su constructor)

B usa a A.

La creación de instancias es una de las actividades más comunes en un sistema orientado a objetos. En consecuencia es útil contar con un principio general para la asignación de las responsabilidades de creación. Si se asignan bien el diseño puede soportar un bajo acoplamiento, mayor claridad, encapsulación y reutilización.

Controlador: asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Esto facilita la centralización de actividades (validaciones, seguridad). El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión.

Se tuvo en cuenta al modelar el sistema los conceptos de bajo acoplamiento para evitar las dependencias excesivas, y la alta cohesión tratando de que cada clase realice labores únicas y bien relacionadas, siempre con la intención de lograr un punto de equilibrio entre ambos. Además, el patrón experto para la realización de las tareas, siendo responsabilidad de la clase que cuenta con los datos involucrados, y el controlador para manejar los eventos del sistema.

(Larman, 2002)

3.2 Codificación

En la fase inicial de Exploración, también conocida como Análisis en algunas bibliografías consultadas, se identifican y detallan las Historias de Usuario, posteriormente el cliente selecciona, en correspondencia con sus prioridades y restricciones de tiempo, define qué construir y el equipo de

desarrolladores entra en escena para darle cumplimiento a lo acordado. Las Historias de Usuario elegidas se descomponen en tareas de programación o ingeniería, escritas en lenguaje técnico, que a su vez son convertidas posteriormente a código. La programación en parejas, refactorización, la integración continua y la compilación de diez minutos son un subconjunto de las prácticas que guían durante esta crucial actividad.

Iteración 1

Se desarrolla la historia de usuario número uno, la cual permite seleccionar un Fichero.

- **HU1 Buscar Fichero**

Tabla 30: Tarea 1 "Buscar Fichero"

Tarea	
Número Tarea: 1	**Número Historia de Usuario:** 1
Nombre Tarea: Diseño de la interfaz para examinar un fichero.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1
Fecha Inicio: 05/03/2012	**Fecha Fin:** 23/03/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, el cual permite Buscar un fichero y visualizar su dirección.	

Se desarrolla la historia de usuario número dos, la cual permite después de seleccionado el fichero adicionarlo al sistema.

- **HU2 Añadir Fichero**

Tabla 31: Tarea 2 "Añadir Fichero"

Tarea	
Número Tarea: 2	**Número Historia de Usuario:** 2
Nombre Tarea: Diseño de la interfaz para añadir un fichero.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1
Fecha Inicio: 05/03/2012	**Fecha Fin:** 23/03/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, el cual permite añadir un fichero al	

sistema obteniendo de este su nombre, dirección, cantidad de líneas y contenido.

Se desarrolla la historia de usuario número tres, la cual permite eliminar un fichero.

- **HU3 Eliminar Fichero**

Tabla 32: Tarea 3 "Eliminar Fichero"

Tarea	
Número Tarea: 3	**Número Historia de Usuario:** 3
Nombre Tarea: Diseño de la interfaz para eliminar un fichero.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 05/03/2012	**Fecha Fin:** 23/03/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, el cual permite eliminar un fichero del sistema.	

Se desarrolla la historia de usuario número cuatro, la cual permite ver el contenido de un fichero.

- **HU4 Contenido Fichero**

Tabla 33: Tarea 4 "Contenido Fichero"

Tarea	
Número Tarea: 4	**Número Historia de Usuario:** 4
Nombre Tarea: Diseño de la interfaz para ver el contenido de un fichero.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 05/03/2012	**Fecha Fin:** 23/03/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, el cual permite visualizar el contenido de un fichero teniendo en cuenta cada línea y su respectivo número.	

Iteración 2

Se desarrolla la historia de usuario número cinco, la cual permite ver las líneas de un fichero.

- **HU5 Traer Líneas**

Tabla 34: Tarea 5 "Traer Líneas"

Tarea	
Número Tarea: 5	**Número Historia de Usuario:** 5
Nombre Tarea: Diseño de la interfaz para traer todas las líneas de un fichero.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 26/03/2012	**Fecha Fin:** 13/04/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, el cual permite visualizar las líneas de un fichero.	

Se desarrolla la historia de usuario número seis, la cual permite escoger las líneas a procesar.

- **HU6 Adicionar Líneas**

Tabla 35: Tarea 6 "Adicionar Líneas"

Tarea	
Número Tarea: 6	**Número Historia de Usuario:** 6
Nombre Tarea: Diseño de la interfaz para adicionar líneas.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 26/03/2012	**Fecha Fin:** 13/04/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, el cual permite seleccionar las líneas que se deseen procesar de un fichero.	

Se desarrolla la historia de usuario número siete, la cual permite adicionar todas las líneas de un fichero a procesar.

- **HU7 Adicionar Todas**

Tabla 36: Tarea 7 "Adicionar Todas"

Tarea

Número Tarea: 7	**Número Historia de Usuario:** 7
Nombre Tarea: Diseño de la interfaz para adicionar todas.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 26/03/2012	**Fecha Fin:** 13/04/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, el cual permite adicionar todas las líneas a una selección a procesar de un fichero.	

Se desarrolla la historia de usuario número ocho, la cual permite eliminar las líneas de la selección a procesar.

- **HU8 Eliminar Líneas**

Tabla 37: Tarea 8 "Eliminar Líneas"

Tarea	
Número Tarea: 8	**Número Historia de Usuario:** 8
Nombre Tarea: Diseño de la interfaz para eliminar líneas.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 26/03/2012	**Fecha Fin:** 13/04/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, el cual permite eliminar líneas de la selección a procesar de un fichero determinado.	

Se desarrolla la historia de usuario número nueve, la cual permite eliminar todas las líneas de la selección a procesar.

- **HU9 Eliminar Todas**

Tabla 38: Tarea 9 "Eliminar Todas"

Tarea	
Número Tarea: 9	**Número Historia de Usuario:** 9

Nombre Tarea: Diseño de la interfaz para eliminar todas las líneas.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 26/03/2012	**Fecha Fin:** 13/04/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, la cual permite eliminar todas las líneas de la selección a procesar.	

Se desarrolla la historia de usuario número diez, la cual permite Adicionar la Selección.

- **HU10 Adicionar Selección**

Tabla 39: Tarea 10 "Adicionar Selección"

Tarea	
Número Tarea: 10	**Número Historia de Usuario:** 10
Nombre Tarea: Diseño de la interfaz para adicionar selección sin delimitador	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 26/03/2012	**Fecha Fin:** 13/04/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa la funcionalidad para cuando el usuario seleccione manualmente la información de la primera línea de la selección a procesar el sistema la adiciona a una nueva lista a procesar.	

Tabla 40: Tarea 11 "Adicionar Selección"

Tarea	
Número Tarea: 11	**Número Historia de Usuario:** 10
Nombre Tarea: Diseño de la interfaz para adicionar selección con delimitador.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1

Fecha Inicio: 26/03/2012	**Fecha Fin:** 13/04/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa la funcionalidad para cuando el usuario entre manualmente el delimitador, el sistema separa la información de la primera línea de la selección a procesar por este criterio y adiciona las selecciones obtenidas automáticamente una nueva lista a procesar.	

Se desarrolla la historia de usuario número once, la cual permite Eliminar una Selección.

- **HU11 Eliminar Selección**

Tabla 41: Tarea 12 "Eliminar Selección"

Tarea	
Número Tarea: 12	**Número Historia de Usuario:** 11
Nombre Tarea: Diseño de la interfaz para eliminar una selección.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 26/03/2012	**Fecha Fin:** 13/04/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, la cual permite eliminar una selección de la nueva lista de selecciones.	

Iteración 3

Se desarrolla la historia de usuario número doce, la cual permite Adicionar una Expresión Regular.

- **HU12 Adicionar Expresión Regular**

Tabla 42: Tarea 13 "AdicionarER"

Tarea	
Número Tarea: 13	**Número Historia de Usuario:** 12
Nombre Tarea: Diseño de la interfaz para adicionar una expresión regular	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1

Fecha Inicio: 23/04/2012	**Fecha Fin:** 11/05/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, en la cual el usuario adiciona una expresión regular escrita o seleccionada por el a la lista de selección de expresiones regulares.	

Se desarrolla la historia de usuario número trece, la cual permite Eliminar Selección Expresión Regular.

- **HU13 Eliminar Selección Expresión Regular**

Tabla 43: Tarea 14 "EliminarSelecionER"

Tarea	
Número Tarea: 14	**Número Historia de Usuario:** 13
Nombre Tarea: Diseño de la interfaz para eliminar una expresión regular seleccionada.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 23/04/2012	**Fecha Fin:** 11/05/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, la cual permite que después de seleccionar alguna expresión regular esta pueda ser eliminada.	

Se desarrolla la historia de usuario número catorce, la cual permite Seleccionar Expresión Regular.

- **HU14 Seleccionar Expresión Regular**

Tabla 44: Tarea 15 "SelecionarER"

Tarea	
Número Tarea: 15	**Número Historia de Usuario:** 14
Nombre Tarea: Diseño de la interfaz para seleccionar una expresión regular.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 23/04/2012	**Fecha Fin:** 11/05/2012
Programador Responsable: Yusliel Sánchez Enriquez	

Descripción: Se implementa el diseño de la interfaz, la cual permite seleccionar expresiones regulares ya utilizadas anteriormente o predefinidas en el sistema.

Se desarrolla la historia de usuario número quince, la cual permite Insertar Expresión Regular.

- **HU15 Insertar Expresión Regular**

Tabla 45: Tarea 16 "InsertarER"

Tarea	
Número Tarea: 16	**Número Historia de Usuario:** 15
Nombre Tarea: Diseño de la interfaz para insertar una expresión regular.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 23/04/2012	**Fecha Fin:** 11/05/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, la cual permite Insertar expresiones regulares al sistema para una posterior reutilización de las mismas.	

Se desarrolla la historia de usuario número dieciséis, la cual permite Eliminar Expresión Regular.

- **HU16 Eliminar Expresión Regular**

Tabla 46: Tarea 17 "EliminarER"

Tarea	
Número Tarea: 17	**Número Historia de Usuario:** 16
Nombre Tarea: Diseño de la interfaz para eliminar una expresión regular.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 23/04/2012	**Fecha Fin:** 11/05/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, la cual permite eliminar expresiones regulares predefinidas en el sistema que no sean de interés para el usuario.	

Se desarrolla la historia de usuario número diecisiete, la cual permite Procesar una Estructura.

- **HU17 Procesar Estructura**

Tabla 47: Tarea 18 "Procesar Estructura"

Tarea	
Número Tarea: 18	**Número Historia de Usuario:** 17
Nombre Tarea: Diseño de la interfaz para procesar una estructura.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3
Fecha Inicio: 23/04/2012	**Fecha Fin:** 11/05/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, la cual después de escoger una información y llenar los campos requeridos permite procesar los datos.	

Se desarrolla la historia de usuario número dieciocho, la cual permite Pre_visualizar las Filas Resultantes.

- **HU18 Pre_visualizar Filas**

Tabla 48: Tarea 19 "Pre_visualizar Filas"

Tarea	
Número Tarea: 19	**Número Historia de Usuario:** 18
Nombre Tarea: Diseño de la interfaz para pre visualizar filas	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 23/04/2012	**Fecha Fin:** 11/05/2012
Programador Responsable: Yusliel Sánchez Enriquez	
Descripción: Se implementa el diseño de la interfaz, la cual después de procesar la información permite pre visualizar las filas resultantes en el proceso.	

Para un mayor entendimiento de las funcionalidades que procesan la información, se muestran en las siguientes figuras los métodos centrales implementados relacionados con el análisis de las cadenas de caracteres.

Se realiza una búsqueda de un patrón especificado por el usuario, en cada una de las líneas escogidas del fichero, ver figura 12.

```
public Vector SeparadoExprecionesR() {
    Vector valores = new Vector();
    boolean decidir = true;
    if (listaEscojida.size() != 0) {
        for (int h = 0; h < listaEscojida.size(); h++) {
            String cad = listaEscojida.get(h).getCadena();
            if (!cad.equals("")) {
                if (decidir == false) {
                    decidir = true;
                }
                int n = cad.length();
                for (int i = 0; i <= n; i++) {
                    for (int j = 0; j <= i; j++) {
                        String sub = cad.substring(j, i);

                        if (sub.matches(selec.getSubcadena()) && decidir == true) {
                            valores.add(sub);
                            decidir = false;
                        }
                    }
                }
                if (decidir == true && (h + 1) != listaEscojida.size()) {
                    valores.add("");
                }
            }
        }
    }
    return valores;
}
```

Figura 13: Método para separar la información con expresiones regulares

Se realiza una búsqueda por un separador especificado por el usuario, en cada una de las líneas escogidas del fichero, ver figura 13.

```
public Vector procesarSeparador(String sepa) {
    Vector valores = new Vector();
    for (int j = 0; j < listaEscojida.size(); j++) {
        for (int i = 0; i < listadeLineas.size(); i++) {
            if (listaEscojida.get(j).getNumeroLinea() == i) {
                if(!listadeLineas.get(j).getCadena().equals("")){
                valores.addElement(listadeLineas.get(i).getSeleccionde ListaSepa
                        (this.getSelec(), sepa).getSubcadena());
            }
        }
    }
    return valores;
}
```

Figura 14: Método para separar la información por un criterio dado

Se realiza una búsqueda por una selección especificada por el usuario, en cada una de las líneas escogidas del fichero, ver figura 14.

```
public Vector procesarSepararmouse() {
    Vector valores = new Vector();
    for (int j = 0; j < listaEscojida.size(); j++) {
        for (int i = 0; i < listadeLineas.size(); i++) {
            if (listaEscojida.get(j).getNumeroLinea() == i) {
                if (!listadeLineas.get(j).getCadena().equals("")) {
                        valores.addElement(listadeLineas.get(i).getSeleccionde Lista
                                (this.getSelec()).getSubcadena());
                }
            }
        }
    }
    return valores;
}
```

Figura 15: Método para separar la información por una selección dada

3.3 Pruebas

Pensar en utilizar la metodología XP sin tener en cuenta las pruebas no es un camino recomendable. Simplemente constituyen una parte fundamental de esta filosofía que no se debe obviar bajo ningún concepto y que debe aplicarse de manera frecuente y temprana. Varias de las principales prácticas que sugiere se apoyan en este enfoque como vía para obtener resultados satisfactorios. La actividad de probar el sistema y verificar que se encuentra libre de defectos, tiene muchos beneficios. Por ejemplo, la calidad es una variable muy importante para todo producto y uno de los caminos para garantizarla es siguiendo esta doctrina. La confianza en el equipo también se apoya sobre esta sólida base. Por último, proporciona una medida del progreso del trabajo que despliega el equipo.

XP se centra principalmente en los niveles de prueba de unidad, de aceptación, también denominadas funcionales, y de integración.

Pruebas funcionales

Las pruebas funcionales no sólo validan la transformación de una entrada en una salida, sino que validan una característica completa. De modo que las pruebas funcionales validan procesos y requieren de un escenario. Por consiguiente, se derivan directamente de las Historias de Usuario. Su número está limitado por la cantidad de escenarios que deban probarse para cada historia con que se cuenta hasta asegurar su funcionamiento adecuado.

Estas pruebas simulan la navegación del usuario, realizan peticiones y comprueban los elementos de la respuesta, tal y como lo haría manualmente un usuario para validar que una determinada acción hace lo que se supone que tiene que hacer.

Debido a la relevancia de las pruebas funcionales se determinó efectuar también casos de prueba. La plantilla utilizada y los resultados obtenidos se describen a continuación.

Tabla 49: Plantilla Casos de Prueba Funcional

Caso de Prueba Funcional	
Identificador: Código que identifica a la prueba.	**Historia de Usuario:** Número de la historia de usuario relacionada con la prueba que se realiza.
Nombre: Definición de la prueba que se realiza.	
Descripción: Breve descripción del objetivo con el que se realiza la prueba.	
Condiciones de Ejecución: Condiciones que deben satisfacerse para que pueda realizarse la prueba.	
Entrada/ Pasos de ejecución: Se describen los pasos de ejecución de la prueba en cuestión.	
Resultado Esperado: Proporciona las expectativas ideales para las cuales fue pensada la prueba.	
Evaluación de la Prueba: Calificación que recibe la prueba de acuerdo a los resultados obtenidos.	

Caso de prueba para la historia de usuario: Buscar fichero.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Buscar fichero. En esta historia de usuario se intenta buscar correctamente un fichero. Además se verifica que el fichero exista y no esté vacío.

Tabla 50: Caso de Prueba Buscar Fichero

Caso de Prueba Funcional	
Identificador: HU1-PF1	**Historia de Usuario:** 1
Nombre: Buscar fichero.	
Descripción: Prueba para la funcionalidad de buscar fichero.	
Condiciones de Ejecución: Se debe buscar un fichero de texto.	
Entrada/ Pasos de ejecución: El usuario selecciona el archivo y se muestra su ubicación.	
Resultado Esperado: Se muestra la dirección física del fichero de texto.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Añadir fichero.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Añadir fichero. En esta historia de usuario se intenta añadir un fichero comprobada su existencia y dada su ubicación. Además se verifica que el fichero exista y no esté vacío.

Tabla 51: Caso de Prueba Añadir Fichero

Caso de Prueba Funcional	
Identificador: HU2-PF1	**Historia de Usuario:** 2
Nombre: Añadir fichero.	
Descripción: Prueba para la funcionalidad de añadir fichero.	
Condiciones de Ejecución: Se debe seleccionar un fichero.	
Entrada/ Pasos de ejecución: Se añade el fichero seleccionado.	
Resultado Esperado: Se añade un nuevo fichero.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Eliminar fichero.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Eliminar fichero.

En esta historia de usuario se intenta eliminar un fichero.

Tabla 52: Caso de Prueba Eliminar Fichero

Caso de Prueba Funcional	
Identificador: HU3-PF1	**Historia de Usuario:** 3
Nombre: Eliminar Fichero.	
Descripción: Prueba para la funcionalidad de eliminar fichero.	
Condiciones de Ejecución: Debe existir un fichero.	
Entrada/ Pasos de ejecución: Se selecciona el fichero y se elimina.	
Resultado Esperado: Se elimina un fichero.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Contenido Fichero.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Contenido Fichero.

En esta historia de usuario se intenta visualizar el contenido de un fichero seleccionado por el usuario después de ser cargado y añadió al sistema.

Tabla 53: Caso de Prueba Contenido Fichero

Caso de Prueba Funcional	
Identificador: HU4-PF1	**Historia de Usuario:** 4
Nombre: Contenido Fichero.	

Descripción: Prueba para la funcionalidad de mostrar el contenido de un fichero.
Condiciones de Ejecución: Debe existir un fichero.
Entrada/ Pasos de ejecución: Se selecciona el fichero y se presiona el botón Contenido Fichero.
Resultado Esperado: Se muestra el contenido del fichero seleccionado.
Evaluación de la Prueba: Satisfactoria.

Caso de prueba para la historia de usuario: Traer Líneas

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Traer Líneas.

En esta historia de usuario se intenta visualizar las líneas del fichero seleccionado por el usuario después de ser cargado y añadió al sistema.

Tabla 54: Caso de Prueba Traer Líneas

Caso de Prueba Funcional	
Identificador: HU5-PF1	**Historia de Usuario:** 5
Nombre: Traer Líneas	
Descripción: Prueba para la funcionalidad de traer las líneas de un fichero.	
Condiciones de Ejecución: El fichero escogido por el usuario no debe estar vacio.	
Entrada/ Pasos de ejecución: Se selecciona el fichero y se presiona el botón Traer Líneas.	
Resultado Esperado: Se muestran las líneas del fichero seleccionado.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Adicionar Líneas.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Adicionar Líneas.

En esta historia de usuario se intenta adicionar una línea manualmente de una lista origen a una lista de selección.

Tabla 55: Caso de Prueba Adicionar Líneas

Caso de Prueba Funcional	
Identificador: HU6-PF1	**Historia de Usuario:** 6
Nombre: Adicionar Líneas.	
Descripción: Prueba para la funcionalidad de adicionar líneas.	
Condiciones de Ejecución: Debe existir al menos una línea.	
Entrada/ Pasos de ejecución: Se selecciona la línea y se presiona el botón Adicionar Líneas.	

Resultado Esperado: Se muestra la línea en la lista de selecciones y se elimina de la lista origen.
Evaluación de la Prueba: Satisfactoria.

Caso de prueba para la historia de usuario: Adicionar Todas

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Adicionar Todas

En esta historia de usuario se intenta adicionar todas las líneas manualmente de una lista origen a una lista de selección.

Tabla 56: Caso de Prueba Adicionar Todas

Caso de Prueba Funcional	
Identificador: HU7-PF1	**Historia de Usuario:** 7
Nombre: Adicionar Todas.	
Descripción: Prueba para la funcionalidad de adicionar todas.	
Condiciones de Ejecución: Debe existir al menos una línea.	
Entrada/ Pasos de ejecución: Se presiona el botón adicionar todas.	
Resultado Esperado: Se muestran todas las líneas en la lista de selecciones y se eliminan de la lista origen.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Eliminar Líneas.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Eliminar Líneas.

En esta historia de usuario se elimina una línea manualmente de la lista selección a la lista de origen.

Tabla 57: Caso de Prueba Eliminar Líneas

Caso de Prueba Funcional	
Identificador: HU8-PF1	**Historia de Usuario:** 8
Nombre: Eliminar Líneas.	
Descripción: Prueba para la funcionalidad de Eliminar Líneas.	
Condiciones de Ejecución: Debe existir al menos una línea en la lista de selecciones.	
Entrada/ Pasos de ejecución: Se selecciona la línea y se presiona el botón Eliminar Líneas.	
Resultado Esperado: Se elimina una línea de la lista de selecciones y se agrega a la lista de origen.	

Evaluación de la Prueba: Satisfactoria.

Caso de prueba para la historia de usuario: Eliminar Todas.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Eliminar Todas.

En esta historia de usuario se eliminan todas las líneas manualmente de la lista de selección a la lista de origen.

Tabla 58: Caso de Prueba Eliminar Todas

Caso de Prueba Funcional	
Identificador: HU9-PF1	**Historia de Usuario:** 9
Nombre: Eliminar Todas.	
Descripción: Prueba para la funcionalidad de eliminar todas las líneas de la lista de selecciones.	
Condiciones de Ejecución: Debe existir al menos una línea en la lista de selecciones.	
Entrada/ Pasos de ejecución: Se presiona el botón eliminar todas.	
Resultado Esperado: Se eliminan todas las líneas de la lista de selecciones y se agregan a la lista origen.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Adicionar Selección.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Adicionar Selección.

En esta historia de usuario se intenta adicionar una selección manualmente de la primera línea del fichero seleccionado.

Tabla 59: Caso de Prueba Adicionar Selección

Caso de Prueba Funcional	
Identificador: HU10-PF1	**Historia de Usuario:** 10
Nombre: Adicionar Selección.	
Descripción: Prueba para la funcionalidad de adicionar selección sin separador.	
Condiciones de Ejecución: Debe existir una selección.	
Entrada/ Pasos de ejecución: Se selecciona manualmente la información de la primera línea del fichero y el sistema la adiciona a una lista.	
Resultado Esperado: Se muestra la selección visualmente en una lista.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Adicionar Selección.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Adicionar Selección.

En esta historia de usuario se intenta adicionar una selección automáticamente después de entrar un criterio de separación para la información.

Tabla 60: Caso de Prueba Adicionar Selección II

Caso de Prueba Funcional	
Identificador: HU10-PF2	**Historia de Usuario:** 10
Nombre: Adicionar Selección.	
Descripción: Prueba para la funcionalidad de adicionar selección con separador.	
Condiciones de Ejecución: Debe existir un separador.	
Entrada/ Pasos de ejecución: El usuario debe entrar un separador.	
Resultado Esperado: Se muestra la selección visualmente en una lista.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Eliminar Selección.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Eliminar Selección.

En esta historia de usuario se intenta eliminar una selección manualmente. Además se verifica que como mínimo exista una selección.

Tabla 61: Caso de Prueba Eliminar Selección

Caso de Prueba Funcional	
Identificador: HU11-PF1	**Historia de Usuario:** 11
Nombre: Eliminar Selección.	
Descripción: Prueba para la funcionalidad de eliminar una selección.	
Condiciones de Ejecución: Debe existir una selección.	
Entrada/ Pasos de ejecución: Se selecciona la información y se elimina.	
Resultado Esperado: Se elimina una información.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Adicionar Expresión Regular.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Adicionar Expresión Regular.

En esta historia de usuario se adiciona una expresión regular manualmente. Además se verifica que exista la misma.

Tabla 62: Caso de Prueba AdicionarER

Caso de Prueba Funcional	
Identificador: HU12-PF1	**Historia de Usuario:** 12
Nombre: Eliminar Selección.	
Descripción: Prueba para la funcionalidad de Adicionar una expresión regular.	
Condiciones de Ejecución: Debe existir la expresión regular.	
Entrada/ Pasos de ejecución: El usuario confecciona la expresión regular o selecciona alguna de las que están en el sistema.	
Resultado Esperado: Se adiciona una expresión regular a la selección de expresiones regulares a utilizar.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Eliminar selección Expresión Regular.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Eliminar selección Expresión Regular.

En esta historia de usuario se elimina una expresión regular de la selección. Además se verifica que exista al menos una expresión en la selección.

Tabla 63: Caso de Prueba EliminarSeleccionER

Caso de Prueba Funcional	
Identificador: HU13-PF1	**Historia de Usuario:** 13
Nombre: Eliminar selección Expresión Regular.	
Descripción: Prueba para la funcionalidad de Eliminar de la selección una expresión regular.	
Condiciones de Ejecución: Debe existir al menos una expresión regular en la selección.	
Entrada/ Pasos de ejecución: El usuario selecciona una expresión de la lista de expresiones seleccionadas y la elimina.	
Resultado Esperado: Se elimina una expresión regular de la lista de selección de expresiones regulares a utilizar.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Insertar Expresión Regular.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Insertar Expresión Regular.

En esta historia de usuario se inserta una expresión regular en el sistema.

Tabla 64: Caso de Prueba InsertarER

Caso de Prueba Funcional	
Identificador: HU15-PF1	**Historia de Usuario:** 15
Nombre: Insertar Expresión Regular.	
Descripción: Prueba para la funcionalidad de insertar una expresión regular.	
Condiciones de Ejecución: Debe existir al menos una expresión regular a insertar.	
Entrada/ Pasos de ejecución: El usuario confecciona una expresión regular y la inserta.	
Resultado Esperado: Se inserta una nueva expresión regular en el sistema para facilitar una futura reutilización.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Eliminar Expresión Regular.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Eliminar Expresión Regular.

En esta historia de usuario se elimina una expresión regular guardada en el sistema.

Tabla 65; Caso de Prueba EliminarER

Caso de Prueba Funcional	
Identificador: HU16-PF1	**Historia de Usuario:** 16
Nombre: Eliminar Expresión Regular.	
Descripción: Prueba para la funcionalidad de eliminar una expresión regular.	
Condiciones de Ejecución: Debe existir al menos una expresión regular en el sistema.	
Entrada/ Pasos de ejecución: El usuario selecciona una expresión regular existente en el sistema con error o que no desea utilizar más y la elimina.	
Resultado Esperado: Se elimina una expresión regular existente en el sistema.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Procesar Estructura.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Procesar Estructura.

En esta historia de usuario se intenta procesar una estructura correctamente. Además se muestra visualmente el resultado del proceso a todo el fichero.

Tabla 66: Caso de Prueba Procesar Estructura

Caso de Prueba Funcional	
Identificador: HU17-PF1	**Historia de Usuario:** 17
Nombre: Procesar Estructura.	
Descripción: Prueba para la funcionalidad de procesar una estructura.	
Condiciones de Ejecución: Debe existir una Estructura.	
Entrada/ Pasos de ejecución: Se selecciona el tipo de dato correspondiente y se llenan los campos requeridos.	
Resultado Esperado: Se muestra en la ventana campos el nombre de todos los campos resultantes.	
Evaluación de la Prueba: Satisfactoria.	

Caso de prueba para la historia de usuario: Pre_visualizar Filas.

Esta sección cubre el conjunto de pruebas que se le realizaron a la historia de usuario: Pre_visualizar Filas.

En esta historia de usuario se intenta pre visualizar todas las filas del fichero después de procesada la información. Además se puede comprobar el nombre de los campos resultantes en la pestaña campos.

Tabla 67: Caso de Prueba Pre_visualizar Filas

Caso de Prueba Funcional	
Identificador: HU18-PF1	**Historia de Usuario:** 18
Nombre: Pre_visualizar Filas.	
Descripción: Prueba para la funcionalidad de pre visualizar filas.	
Condiciones de Ejecución: Debe haber procesada una estructura.	
Entrada/ Pasos de ejecución: Procesar una estructura.	
Resultado Esperado: Se muestra en la ventana campos el nombre de todos los campos resultantes.	
Evaluación de la Prueba: Satisfactoria.	

Cobertura de pruebas

Al sistema desarrollado se le realizó un conjunto de pruebas para obtener métricas necesarias para su posible utilización, como aconseja XP. Como se muestra en la tabla 67. Las métricas establecidas

fueron: cantidad de ficheros entrados y cantidad de ficheros no procesados. Las pruebas para determinar la fiabilidad del sistema se realizaron con 52 ficheros, procesados entre las 4 posibilidades de análisis, la fiabilidad obtenida fue de un 92.30% como se muestra en la tabla que aparece a continuación.

Tabla 68: Cobertura de Pruebas

Posibilidades de análisis	**Cantidad de ficheros entrados**	**Cantidad de ficheros no procesados**	**Porciento**
Separado con mouse	15	1	93.33%
Separado con expresiones regulares	16	0	100.00%
Con separador de campos	9	1	88.88%
Separado por pentaho	12	2	83.00%
Total	52	4	92.30%

3.2 Conclusiones

En el transcurso de este capítulo se expusieron los artefactos construidos como parte de las actividades de diseño y codificación, los cuales se repiten en cada iteración, tales como las tarjetas CRC. Las pruebas de aceptación o funcionales que se le realizaron al sistema a lo largo de todo el ciclo de vida, como dicta la metodología, también fueron descritas. Se logró diseñar pruebas las cuales garantizan la calidad del componente construido. La propuesta de solución de la aplicación a implementar se da por concluida simultáneamente con el fin del capítulo actual.

CONCLUSIONES

Como resultado del presente trabajo de diploma y para dar cumplimiento a los objetivos propuestos se concluye lo siguiente.

- Durante el desarrollo de este trabajo investigativo se determinó que la selección de la metodología XP fue la correcta, pues el ciclo de desarrollo se ajustó a sus fases y características.
- Se analizaron los aspectos fundamentales sobre el desarrollo de plugin para la herramienta Pentaho Data Integration facilitando la implementación del plugin.
- Se realizó la implementación del plugin en su versión 1.0 para procesar cadenas de caracteres, llevándolas a un formato estándar, facilitando el trabajo con este tipo de información.
- Se probó el correcto funcionamiento de la herramienta, a través de las pruebas funcionales o pruebas de aceptación las cuales se pusieron en práctica mediante los casos de prueba.

RECOMENDACIONES

Por los resultados obtenidos durante la investigación y con vista a enriquecer la solución, el autor sugiere.

- Aumentar la cantidad de pruebas con el objetivo de optimizar la solución.
- Seguir profundizando en el trabajo con cadenas de caracteres ya que la solución desarrollada es solamente para cadenas de caracteres homogéneas.
- Posteriormente adicionarle la funcionalidad de exportar a fichero de texto y así el plugin servirá a otras herramientas que procesan cadenas de caracteres.

REFERENCIAS BIBLIOGRÁFICAS

Canos, Letelier H. 2011. Metodologías Ágiles en el Desarrollo de Software. 2011.

Chodnicki, Slawomir. 2010. Adventures with Open Source BI. Adventures with Open Source BI. [En línea] Musings of a BI-Developer, 13 de Junio de 2010. [Citado el: 5 de Abril de 2012.] http://type-exit.org/adventures-with-open-source-bi/2010/06/developing-a-custom-kettle-plugin-a-simple-transformation-step/.

Cursos. 2011. Programación Orientada a Objetos. Programación Orientada a Objetos. [En línea] Cursos.aiu.edu., Noviembre de 2011. [Citado el: 5 de Febrero de 2012.] http://cursos.aiu.edu.

Díaz Ordaz. 2012. Gravitar. Gravitar. [En línea] 2012. [Citado el: 20 de Enero de 2012.] http://www.gravitar.biz/index.php/herramientas-bi/pentaho/caracteristicas-pentaho/.

Espinosa, Roberto. 2010. El rincón del BI. El rincón del BI. [En línea] 10 de Mayo de 2010. [Citado el: 10 de Noviembre de 2011.] http://churriwifi.wordpress.com/category/etl/page/2/.

Forums.pentaho2010 http://forums.pentaho.com

Garcia, Brandon. 2011. SlideSharede. SlideSharede. [En línea] SlideSharede, 18 de Noviembre de 2011. [Citado el: 4 de Diciembre de 2011.] http://www.slideshare.net/BraandonGaarciiaa/programa-informaticotecnicas..

Larman, Craig. 2002. UML y Patrones. 2002.

Marañón, Gonzalo Álvarez. 1999. Que es Java. Que es Java. [En línea] Instituto de Física Aplicada del CSIC, 1999. [Citado el: 18 de Noviembre de 2011.] http://www.iec.csic.es/criptonomicon/java/quesjava.html.

Netbeans. 2012. netbeans. netbeans. [En línea] 2012. [Citado el: 5 de marzo de 2012.] http://netbeans.org/index_es.html.

Penadés, Patricio. 2009. willydev.net. willydev.net. [En línea] 2009. [Citado el: 10 de Febrero de 2012.] http://www.willydev.net/descargas/masyxp.

Pilgrim, M. (2009). staff.not.iac.es. Recuperado el 2012, de staff.not.iac.es: http://staff.not.iac.es/~rcardenes/hg/diveintopython3-es/regular-expressions.html.

Requirements 2011. elvex.ugr.es Recuperado el 2012, de elvex.ugr.es http://elvex.ugr.es/idbis/db/docs/design/2-requirements.pdf. [Online]

Sampieri, Roberto Hernandez. 1998. Metodología de la Investigación. Mexico : McGRAW-HILL INTERAMERICANA EDITORES, S. A. de C. V., 1998.

SI. 2011. Ingeniería del Software. Ingeniería del Software. [En línea] 2011. [Citado el: 5 de marzo de 2012.] www.is.ls.fi.upm.es/docencia/is2/documentacion/ModeloDominio.

Sierra, Daniel. 2007. Visual Paradigm For Uml. 2007. IPN UPIICSA .

Walczuch, N. (2011). Cadenas de Caracteres. Los Andes: Escuela de Sistemas. Facultad de Ingeniería.

BIBLIOGRAFÍA

Almacenes de datos (datawarehouse), Recuperado el 20 de 10 de 2010, disponible en http://www.rhernando.net/modules/tutorials/doc/bd/dw.html.

Business Intelligence. Recuperado el 5 de 5 de 2011, disponible en http://www.productive.com.ar/productos_bi.php.

Business Intelligence fácil: DSS: Tipos de decisiones empresariales. Recuperado el 5 de 5 de 2011, disponible en http://www.businessintelligence.info/dss/toma-decisiones-business-intelligence.html.

Data Integration: Using ETL. EAI and EII Tools to Create an Integrate Enterprise. (s.f.). Recuperado el 20 de 11 de 2010, de http://www.bi-bestpractices.com/view-articles/4737.

Datawarehouse. (s.f.). Recuperado el 20 de 11 de 2010, disponible en http://www.sinnexus.com/business_intelligence/datawarehouse.aspx.

Hanik, F. (2007). INTRODUCTION TO APACHE TOMCAT 6.

http://www.sinnexus.com/business_intelligence/datamart.aspx © Copyright 2007 - 2011 -

Laura Haas. Beauty and the beast: The theory and practice of information integration. ICDT 2007: 28-43.

Maurizio Lenzerini," Data Integration: a Theoretical Perspective", Universita di Roma "La Sapienza". ACM PODS 2002.

Mondrian - El servidor OLAP Open Source. (s.f.). Recuperado el 27 de 10 de 2010, disponible en http://pentaho.almacen-datos.com/mondrian.html.

Pentaho Mondrian Documentación. Recuperado el 30 de 11 de 2010, disponible en http://mondrian.pentaho.com/documentation/index.php.

Pentaho.com Sitemap. (s.f.). Recuperado el 30 de 11 de 2010, disponible en http://www.pentaho.com/products/data_integration/r.

Ponniah, P. (2001). Data Warehousing Fundamentals (1a. ed.). New York: John Wiley.

Prof. Lauro Soto, Ensenada, BC, México http://www.mitecnologico.com/Main/DefinicionesConceptosMercadosDatos.

Siete Formas de integración de datos. (2008). Recuperado el 16 de 11 de 2010, disponible en http://www.muycomputerpro.com/centro-de-conocimientowhite-papers7-formas-de-integracion-de-atos_we9erk2xxdazdqyeebujg7or6shmcwyoivks1jrr48ecaalib5aehoygizgw2zmr/.

Sobre PostgresSQL. (s.f.). Recuperado el 10 de 11 de 2010, de http://www.postgresql-es.org/sobre_postgresql.

GLOSARIO

Linux: es un sistema operativo tipo Unix (también conocido como GNU/Linux) que se distribuye bajo la Licencia Pública General de GNU (GNU GPL), es decir que es software libre. Su nombre proviene del Núcleo de Linux, desarrollado en 1991 por Linus Torvalds. Las variantes de estos sistemas se denominan "distribuciones" y su objetivo es ofrecer una edición que cumpla con las necesidades de determinado grupo de usuarios. De esta forma existen distribuciones para hogares, empresas y servidores. Algunas son gratuitas y otras de pago, algunas insertan software no libre y otras contienen solo software libre.

Open Source: código abierto. En programación este término se utiliza para definir a aquel software o código al cual se puede acceder a sus fuentes, llámese fuente a los ficheros de código y todos los recursos que se usaron para la creación de dicho software.

Software Libre: se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades fundamentales de los usuarios del software planteadas por la Fundación de Software Libre (FSF), las cuales son: libertad de usar el programa para cualquier propósito, libertad de estudiar cómo funciona el programa y adaptarlo a sus necesidades, la libertad de distribuirlo, la libertad de mejorar el programa y hacer públicas las mejoras a los demás, logrando un beneficio a la comunidad.

Unix: sistema operativo atribuido a Ken Thompson y comercializado por la empresa ATT en la década de los 70s que alcanzó mucho éxito, sobre todo en las universidades y posteriormente en las empresas. Entre sus principales características tenemos que es: portable, robusto, y flexible actualmente goza de gran popularidad dentro de la tecnología de Internet.

Fichero: conjunto de información que se almacena para consultarse o utilizarse posteriormente.

UML: “Unified Modeling Language”. Lenguaje gráfico que brinda un vocabulario y reglas para especificar, construir, visualizar y documentar los artefactos de un sistema utilizando el enfoque orientado a objetos.

IDE: entorno de desarrollo Integrado.Herramientas que facilitan el trabajo.

Visual Paradigm: herramienta CASE (Computer-Aided Software Engineering) que permite realizar ingeniería tanto directa como inversa. Utiliza UML como lenguaje de modelado y RUP como metodología de desarrollo.

Java: es un lenguaje de programación orientado a objetos.

XML: es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C), que permite definir la gramática de lenguajes específicos.