

Universidad de las Ciencias Informáticas
Facultad 2



Módulo de Preparación de Datos para la Herramienta informática
De Minería de Uso de la Web sobre los registros de navegación por
Internet



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

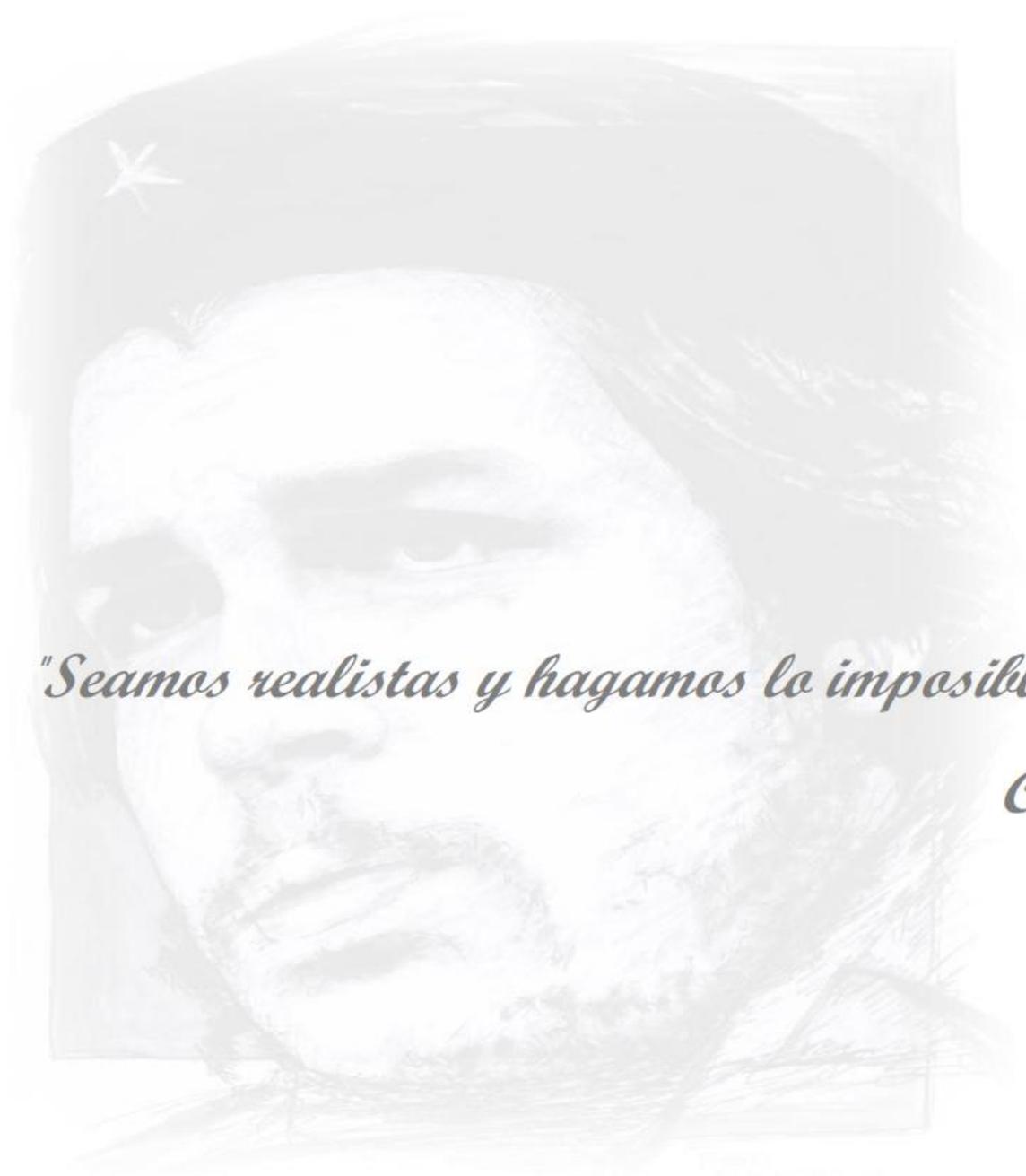
Autor(es): Yissel Pérez Rodríguez

Reynaldo del Toro Aguilera

Tutor: Ing. Yoanni Ordoñez Leyva

La Habana, Junio de 2012

“Año 54 del Triunfo de la Revolución”



"Seamos realistas y hagamos lo imposible."

Che.

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Facultad 2 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Firma de los autores:

Yissel Pérez Rodríguez

Reynaldo del Toro Aguilera

Yoanni Ordoñez Leyva

Firma del tutor

DATOS DE CONTACTO

Síntesis del Tutor:

Graduado en el año 2010 en Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas. Ha trabajado la Minería de Datos siendo autor de una herramienta para extraer patrones descriptivos sobre los registros de navegación por Internet llamada HERMINWEB. Ha sido programador y arquitecto de varios proyectos. Investiga la línea de Inteligencia Artificial relacionada con la Minería de Datos.

Correo: yordones@uci.cu

Provincia: Holguín.

AGRADECIMIENTOS

Agradezco en primer lugar a mis abuelos Lilia y José que aunque ya no están conmigo físicamente sé que donde se encuentren, siempre estarán orgullosos de la persona en la que me he convertido. A mis padres que me han dado su apoyo durante estos cinco años de carrera y me han formado como mujer y como persona. Ustedes representan el cariño más grande que alguien puede tener, siempre estuvieron ahí para empujarme hacia adelante aún cuando me faltaron las fuerzas. Por su amor, su entrega y su dedicación este título lleva sus nombres.

A mi esposo, que me esperó y me apoyó durante todo este año y cuidó a nuestra hija para que yo pudiera convertirme en una profesional. Gracias mi amor por estar siempre a mi lado. A mi bebé Yenifer, que es el regalo más lindo que la vida me ha dado. Todo lo que he hecho ha sido para asegurarle un futuro mejor. Gracias mi niña por darme las fuerzas para seguir adelante.

A mi compañero de tesis, por confiar en mí para realizar este trabajo, que más que una tesis ha sido un reto. Por asumir todas las responsabilidades de la tesis cuando yo no pude estar en la universidad y por ayudar en mi superación como ingeniera y como persona. A mis amigos Omevis, Yaniel, Yuniel y Jany por tantos momentos buenos y malos compartidos durante estos cinco años, la UCI no hubiera sido lo mismo sin ustedes. Ome gracias por tu amistad incondicional, más que mi amiga ya eres mi hermana. A Barbara Triana y a Daimara Martínez que más que mis profesoras en estos años han sido mis amigas, y estuvieron ahí siempre que las necesité, que me ayudaron a crecer como profesional y como persona. A mis amigos de grupo y de apartamento Yunislema, Yusdania, Surenis, Yanet, Erenio, Meralis, Frank, gracias por su amistad, su preocupación y su ayuda. Les deseo mucha suerte en la vida. A mi tía Elsa, mi tío Bernardo y mis primos Yasel y Wilde, porque fueron mi apoyo cuando estuve lejos de mi casa y porque son la familia más maravillosa que alguien puede tener.

A todos los profesores de esta universidad y de la facultad 2 que han contribuido en mi formación. A Yoanni Ordoñez nuestro tutor, por ser nuestro padre de tesis, ayudarnos, apoyarnos, obligarnos a estudiar, a escribir artículos científicos y por desinflarnos tantos globos sin fundamento cuando no sabíamos que responder. Gracias por confiar en nosotros, por creer que íbamos a sacar la tesis adelante, espero no te hayamos defraudado, porque sé que HERMINWEB es muy importante para ti. Gracias sobre todo por "fajarte" por estos discípulos tuyos que todavía no han aprendido a defenderse. Yissel.

Agradezco en primer lugar a mis padres, lo más grande que tengo en mi vida, por guiarme siempre por el buen camino, por darme la confianza que siempre necesité, por los consejos que me dieron y apoyarme en cada una de las decisiones que he tomado en mi vida. Gracias a ustedes soy hoy una mejor persona, por eso les dedico de todo corazón este sueño hecho realidad.

A mis abuelos, a mi abuelo Guelo que en estos momentos no puede acompañarme, pero sé que estaría orgulloso de mí, a mi abuelita Hilda, por su sabiduría, por los consejos con los que he crecido, privilegio que junto a sus mimos, me convierten en afortunado de tenerla cerca. También a mis abuelos Luis Mario y María Esther, que siempre estuvieron al tanto de mi vida, por su preocupación y su cariño, gracias.

A mi tío Jose, por su constante preocupación por mí y por brindarme su apoyo siempre que lo he necesitado, ya me hice gente. A mis primos: Darel, Luisito, Lilien y Yanela, por ser compañeros de juegos y travesuras, gracias a ustedes descargo toda la tensión acumulada en la UCI. A mis tíos Luis e Idania, por brindarme su sabiduría, que tanto me ha ayudado, por sus consejos y su ayuda que me permitieron convertirme en un profesional sin perderme lo maravilloso de la vida de la universidad. A mis tías Patricia, a Olguita, y a toda mi familia, por quererme y ayudar en mi formación como el hombre que hoy soy, gracias por ser una familia tan linda.

A mi compañera de tesis, sin ti a mi lado jamás hubiera terminado este trabajo, te convertiste en alguien imprescindible cuyos consejos y regaños lograron que todo saliera con la mejor calidad posible, gracias por despertarme, por empujarme a que diera más de mí, por la confianza en que podría hacerlo y no rendirte ni dejarme tirar la toalla.

A mi tutor, Yoanni, eres una persona que todo lo hace ver como si fuera lo más fácil del mundo, aunque parezca que todo está perdido, sabe encontrar una solución sencilla y elegante, sin tu guía no sería posible este sueño, gracias por el apoyo constante, por jamás decir que no aunque fueran las 5 de la mañana. Este también es tu trabajo.

A Darian Horacio, por ser el ángel guardián, por llegar en el momento adecuado y poner el punto final cada vez que las cosas no quedaban del todo claras. A Julio y Hussein, por responder mis preguntas en cualquier momento y lugar, por no decir jamás que no, por la tremendísima tesis que desarrollaron y que nos facilitó tanto el trabajo. A los iniciadores de la genial idea de Herminweb, también un agradecimiento

especial, aunque ya los haya mencionado era imposible dejar pasar el tremendo trabajo que hicieron sin la cual no hubiera sido posible nada más.

A Bárbara Triana, por sugerirme para realizar esta tesis, por confiar en mí cuando no tenía un tema de tesis, creo que no la defraudé. Por sus revisiones, por sus sugerencias y consejos que tan útiles nos fueron, brindándonos su experiencia en todo momento, gracias.

A todos mis amigos: Charlie, Reynol, Néstor, por estar siempre a pesar de mis días buenos y malos, por los momentos compartidos, por las bromas, que hicieron de mi paso por la universidad una etapa interesante y divertida.

A mis compañeros de cuarto, por soportar mi mal carácter de algunos días, por ser un grupo tan chévere, y hacer de la convivencia algo tan fácil: Reinier, Janier, Mario, y las dos Elizabeth, Jorge y Yarima, gracias por la compañía.

Al tribunal y a mi oponente, por lograr con sus sugerencias y señalamientos, que el trabajo saliera con la calidad deseada, y ayudar de esta forma en mi crecimiento como profesional.

A todos los que de una u otra forma me acompañaron y ayudaron a lo largo de estos 5 años, a todos los presentes hoy, gracias. Reynaldo

DEDICATORIA

A mi hija Yenifer, por ser lo más grande que tengo en el mundo.

A mi hermana Yessica, que me gustaría siguiera mis pasos algún día.

A mis padres, por hacer de mí la persona que soy.

A mi esposo, por su apoyo y su confianza.

Yissel Pérez Rodríguez.

DEDICATORIA

A mis padres, por darme la vida y dedicar la suya a que yo fuera una mejor persona.

A mis abuelos, por malcriarme hasta el cansancio.

A mi familia, por ser la mejor familia del mundo.

Reynaldo del Toro Aquilera

Resumen

Este trabajo presenta un Módulo de Preparación de Datos para lograr que la Herramienta de Minería de Uso de la Web aplicada a los registros del proxy (HERMINWEB) se pueda utilizar en otros entornos distintos a la UCI o a un diseño de información diferente. Mantiene el soporte a las mismas fuentes de datos: Base de Datos, Servicios Web y LDAP. Está desarrollado sobre una arquitectura basada en componentes, garantizando la reutilización y extensibilidad de la aplicación. Brinda un diseñador gráfico que permite al cliente configurar la preparación de datos en un flujo de pasos donde cada componente representa un paso dentro del proceso, unidos por conectores que convierten la salida de un componente en entrada del siguiente. Cada componente se configura para una tarea específica, posibilitando la obtención, transformación, integración o el almacenamiento de los datos. Permite definir uno a varios flujos en caso de tener los datos de los usuarios repartidos en varias fuentes que no se relacionen. La obtención, transformación y almacenamiento de los datos de los usuarios se realizan con la ETL Kettle de la suite de aplicaciones Pentaho. El procesamiento de los registros del proxy se realiza con un diseño distribuido, debido a que este proceso consume una elevada cantidad de tiempo y recursos. Permite mezclar la información de los usuarios con la información de la navegación o solo procesar los registros de navegación del proxy. El diseño de la base de datos está en correspondencia con los datos que se deseen analizar creándose dinámicamente.

Palabras clave: Arquitectura Basada en Componentes, PDI, Preparación de Datos.

TABLA DE CONTENIDOS

Resumen	I
Introducción	6
Capítulo 1: Fundamentación Teórica	14
1.1 Introducción	14
1.2 Proceso de extracción del conocimiento en Bases de Datos (KDD)	14
1.3 HERMINWEB	15
1.4 Arquitecturas Modulares	17
Software Basado en Componentes	18
Pyutilib Component Architecture	19
1.5 Pentaho Data Integration	19
Metodología, Lenguajes y Herramientas de Desarrollo	20
1.6 Sistema Gestor de Bases de Datos (SGBD).....	20
1.7 Lenguaje de Programación: Python.....	21
1.8 Interfaces Gráficas de Usuario: Qt.....	22
1.10 Modelado de los Procesos de Negocio con IDEF0	24
1.11 Protocolo de Comunicación entre Aplicaciones: ICE	25
Conclusiones del Capítulo	26
Capítulo 2: Características del Sistema	27
2.1 Introducción.....	27
2.2 Modelo de Negocio.....	27
2.2.1 Representación de los procesos de negocio	27
2.6 Especificación de los requisitos de software	29
2.6.1 Requisitos funcionales.....	29
2.6.2 Requisitos no funcionales	32
2.7 Modelo de Casos de Uso del Sistema	34
2.7.1 Actores del Sistema.....	34
2.7.2 Diagrama de Paquetes	34

2.7.3 Casos de Uso del Sistema	35
2.7.4 Diagrama de Casos de Uso del Sistema.	36
2.7.5 Descripción detallada de los Casos de Uso Críticos.....	36
Descripción detallada del Caso de Uso “Transformar Datos”	39
Descripción detallada del Caso de Uso “Procesar los registros (logs)”	39
Descripción detallada del caso de uso “Mezclar Datos”	40
Descripción detallada del caso de uso “Procesar fuentes de datos”	41
Descripción detallada del Caso de Uso “Procesar Blacklist”	42
Conclusiones del Capítulo	43
Capítulo 3: Diseño del Sistema	44
3.1 Introducción.....	44
3.2. Arquitectura	44
3.3 Patrones de Diseño	45
3.4 Seguridad en el Sistema.....	48
3.5 Mejoras en el proceso de preparación de los datos.....	50
3.6 Diagrama de paquetes del diseño.	53
3.7 Modelo Físico de Datos.	54
3.8 Diagramas de Clases del Diseño.....	54
Capítulo 4: Implementación y Pruebas.....	58
4.1 Introducción.....	58
4.3 Diagrama de Despliegue	58
4.2 Diagramas de componentes	59
4.4 Estrategias de prueba.	62
Conclusiones del capítulo.....	69
Conclusiones Generales.....	70
Recomendaciones	72
Referencias Bibliográficas.....	73
Glosario	75

Anexos.....¡ERROR! MARCADOR NO DEFINIDO.

Anexo 1: Descripción de los Casos de Uso del Sistema..... ¡Error! Marcador no definido.

Anexo 2: Configuración y Diseño de una Preparación de Datos.¡ERROR! MARCADOR NO DEFINIDO.

ÍNDICE DE FIGURAS

Figura 1:Etapas del proceso KDD [7]	15
Figura 2. Macro-Proceso de Negocio Preparar Datos.....	27
Figura 3 Proceso de Negocio Preparar Datos.....	28
Figura 4 Diagrama de paquetes de Caso de Uso del Sistema	35
Figura 5 Diagrama de Casos de Uso del Sistema agrupados por paquetes.....	36
Figura 6 Arquitectura del Sistema	45
Figura 7 Configuración de una preparación de datos.....	50
Figura 8 Interfaces de Configuración de Componentes.....	51
Figura 9 Configuración de los pasos para realizar el procesamiento de los registros.....	52
Figura 10 Diagrama de Paquetes del Diseño.....	54
Figura 11 Diagrama de Clases Diseño del Caso de Uso Procesar Fuentes de Datos.....	55
Figura 12 Diagrama de Clases del Diseño del Caso de Uso Procesar Blacklist.....	55
Figura 13 Diagrama de Clases del Diseño del Caso de Uso Procesar Registros del proxy.....	56
Figura 14 Diagrama de Clases del Diseño del Caso de Uso Transformar Datos.....	56
Figura 15 Diagrama de Clases del Diseño del Caso de Uso Mezclar Datos.	57
Figura 16 Diagrama de Despliegue.....	58
Figura 17 Diagrama de Componentes de los CU que intervienen en el Procesamiento de las fuentes de Información.....	60
Figura 18 Diagrama de Componentes de los CU que intervienen en la transformación y salva de los datos	61
Figura 19 Diseño de una preparación de datos (procesamiento de usuarios)	¡Error! Marcador no definido.
Figura 20 Configuración de la conexión a una base de datos	¡Error! Marcador no definido.
Figura 21 Configuración de una nueva conexión a base de datos	¡Error! Marcador no definido.
Figura 22 Explorador de la Base de Datos.....	¡Error! Marcador no definido.

Figura 23 Configuración de una transformación nominal	¡Error! Marcador no definido.
Figura 24 Configuración de acceso a una base de datos LDAP.....	¡Error! Marcador no definido.
Figura 25 Obtención de los campos de LDAP.....	¡Error! Marcador no definido.
Figura 26 Configuración de un componente integrador.....	¡Error! Marcador no definido.
Figura 27 Configuración del componente Salvar en Base de Datos.....	¡Error! Marcador no definido.
Figura 28 Configuración Componente Optimizar	¡Error! Marcador no definido.
Figura 29 Configuración de una transformación para los registros del proxy.¡Error! Marcador no definido.	
Figura 30 Configuración del componente Log.....	¡Error! Marcador no definido.
Figura 31 Configuración de los nodos de procesamiento.....	¡Error! Marcador no definido.
Figura 32 Selección de los campos de los logs para el análisis	¡Error! Marcador no definido.
Figura 33 Configuración de discretizaciones.....	¡Error! Marcador no definido.
Figura 34 Configuración de horario.....	¡Error! Marcador no definido.
Figura 35 Configuración del Componente Sesiones	¡Error! Marcador no definido.

ÍNDICE DE TABLAS

Tabla 1 Indicadores para medir Usabilidad	10
Tabla 2 Descripción de los actores del sistema	34
Tabla 3 Descripción de los nodos del diagrama de despliegue	59
Tabla 4 Caso de Prueba: Configurar Componente de Entrada Base de Datos para una base de datos PostgreSQL	65
Tabla 5 Descripción de las variables.....	65
Tabla 6 Características del Entorno de Pruebas	66
Tabla 7 Resultados de las pruebas.....	¡Error! Marcador no definido.
Tabla 8 Medición de la variable Dependencia.....	68
Tabla 11 Resultados de la medición de las variables.....	68

Introducción

El ser humano al reutilizar experiencias, ideas y artefactos no solo asegura no cometer los mismos errores del pasado, sino que logra construir cosas con bases firmes y buena calidad. Este concepto de reutilización se aplica en el desarrollo de software. Los sistemas informáticos de la actualidad son cada vez más complejos, deben ser construidos en poco tiempo y deben cumplir con los estándares más altos de calidad. Para hacer frente a esta necesidad se creó y perfeccionó lo que se conoce como Ingeniería de Software Basada en Componentes (ISBC) que se centra en el diseño y construcción de sistemas de software que utilizan componentes de software reutilizables. La complejidad de los sistemas computacionales actuales nos ha llevado a buscar la reutilización del software existente. El desarrollo de software basado en componentes posibilita reutilizar piezas de código ya elaboradas que permiten realizar diferentes tareas, conllevando a diversos beneficios como las mejoras a la calidad.

Se puede definir un componente de software como una pieza de código preelaborado que encapsula alguna funcionalidad expuesta a través de interfaces estándar. Los componentes son los ingredientes de las aplicaciones, que se juntan y combinan para llevar a cabo una tarea. El arte de ensamblar componentes y escribir código para que estos componentes funcionen se conoce como Desarrollo de Software Basado en Componentes. Estos componentes están diseñados para tener las mínimas dependencias de otros componentes y aplicaciones dentro y fuera del sistema [1].

Según la Real Academia de la Lengua Española una dependencia es la subordinación de una persona o cosa respecto de otra u otras por las que está regida o a las que está sometida. Necesidad física o psíquica que tiene un individuo de consumir algún producto, generalmente perjudicial para el organismo [2]. En el contexto de esta investigación una dependencia es la necesidad que tiene un programa o software de consumir determinados datos o servicios de otras aplicaciones para poder funcionar correctamente. A menudo las dependencias dificultan o hacen imposible la utilización de un software en ambientes determinados disminuyendo la usabilidad de la aplicación.

La ISO/IEC 9241 define usabilidad como la eficacia, eficiencia y satisfacción con la que un producto permite alcanzar objetivos específicos a usuarios específicos en un contexto de uso específico [3]. Durante el desarrollo de esta investigación se estará haciendo referencia al concepto de usabilidad desde

la perspectiva de la capacidad de una aplicación de ser usada en entornos con fuentes de información específicas y diseños de información diferentes.

Actualmente existen innumerables aplicaciones de software, dependientes de un contexto, con un diseño de información específico, que imposibilita su aplicación en otros entornos aunque cuente con las mismas fuentes de información. Un ejemplo de esto es la Herramienta de Minería de Uso de la Web aplicada a los Registros del Proxy (HERMINWEB) creada con el objetivo de ayudar a la Dirección de Redes y Seguridad Informática (DRSI) de la Universidad de las Ciencias Informáticas (UCI) en la toma de decisiones para la asignación de las cuotas de navegación por Internet a los usuarios.

La primera versión de esta herramienta se terminó en el 2010 y brinda la posibilidad de obtener patrones en términos de la tarea descriptiva Agrupamiento o Segmentación, la cual consiste en obtener grupos a partir de los datos, de forma tal que los objetos de un mismo grupo son muy similares entre sí y, al mismo tiempo, son muy diferentes a los objetos de otro grupo, con el propósito de analizar los registros del servidor proxy para extraer patrones que describan el comportamiento de los usuarios en el uso de sus cuotas de navegación por Internet [4].

En el período 2010-2011 se desarrolla una nueva versión de HERMINWEB, implementando entonces las tareas de Reglas de Asociación que tiene como objetivo identificar relaciones no explícitas entre atributos categóricos o nominales que se emplean frecuentemente para reconocer cómo la ocurrencia de un suceso o acción puede inducir o generar la aparición de otros. Es utilizada cuando el objetivo es realizar análisis exploratorios, buscando relaciones dentro del conjunto de datos que enriquecieron la descripción del comportamiento de los usuarios en el uso de las cuotas de navegación por Internet, apoyando de forma más precisa a la DRSI de la UCI en la toma de decisiones [4].

HERMINWEB es una herramienta que posee un módulo de preparación de datos diseñado para la información de los usuarios en la UCI. En la fase de Preparación de Datos HERMINWEB solo se apoya en la ETL¹ Kettle² de la suite de aplicaciones Pentaho³ para realizar las conexiones a la base de datos, y realizar las transformaciones a los datos. HERMINWEB define campos fijos tanto de la navegación como

¹ Herramienta de Extracción, Transformación y carga de datos (del inglés Extract, Transform and Load).

² Componente de la suite de aplicaciones Pentaho Business que permite la extracción, transformación y carga de datos.

³ Herramienta open source perteneciente a la familia de aplicaciones de Inteligencia de Negocio.

de los usuarios. En el caso de la navegación sólo usan de las peticiones el usuario, la dirección IP, la URL, la hora, el día y la cantidad de bytes consumidos. Los usuarios en la UCI se clasifican en tres tipos, los estudiantes, los trabajadores internos y los trabajadores externos a la universidad. Algunos de los datos recogidos y analizados de los estudiantes son el nombre y los apellidos, la facultad, la provincia de residencia, el número de carnet de identidad y el índice académico. En el caso de los trabajadores se recogen su nombre y apellidos, el número del carnet de identidad, el cargo y si es docente o no entre otros. Estos datos son fijos y no se les pueden agregar o quitar campos para ser analizados. Además la categorización de la cantidad de peticiones es fija así como la cantidad de peticiones de una sesión. Para realizar el análisis es necesario mezclar los datos de los usuarios con los datos de la navegación, pero no brinda la opción de analizar solamente los datos de la navegación. Esta herramienta define un concepto de sesiones⁴ de usuario que agrupa las peticiones que coinciden en los campos usuario, URL, subred, día y horario no permitiendo modificar esta definición, pues no se puede agregar campos o eliminar alguno de los definidos. La base de datos está diseñada para los campos que sus autores escogieron como importantes por lo que no soporta la extensión de nuevos campos o la eliminación de otros.

Por todas las limitaciones planteadas y la heterogeneidad en el diseño de la información con que cuentan las empresas e instituciones que poseen servicios de navegación por Internet se hace imposible aplicar HERMINWEB, en otro entorno distinto a la UCI, sin realizarle considerables modificaciones.

Por tanto el **problema de la investigación** es: ¿Cómo mejorar la usabilidad de HERMINWEB disminuyendo la dependencia del diseño de la información en la preparación de los datos?

El **objetivo general** es desarrollar un módulo que sea capaz de mejorar la usabilidad de la preparación de los datos de HERMINWEB disminuyendo la dependencia de los diseños de información, apoyados en un diseño en componentes, para que sea adaptado a otros diseños de información que utilice las mismas fuentes de datos

Para cumplir con el objetivo propuesto, se proponen los siguientes **objetivos específicos**:

- Seleccionar una herramienta para la selección, integración y transformación de datos.

⁴ Grupo de peticiones de un usuario que coinciden en los campos definidos.

- Desarrollar un editor gráfico de transformaciones que permita definir las fuentes de datos y las variables así como sus relaciones de integración y las transformaciones.
- Desarrollar un módulo que permita seleccionar, integrar y transformar los datos de las fuentes de datos más comunes como base de datos, servicios Web⁵, LDAP⁶, ficheros de trazas, que pueda ser aplicado en cualquier diseño de información que posea estos tipos de fuentes de datos.

Teniendo como **objeto de estudio** de esta investigación las aplicaciones modulares. El **campo de acción** queda enmarcado específicamente en los sistemas basados en componentes. La **hipótesis** de esta investigación plantea que con la implementación de un módulo que esté diseñado mediante componentes se logrará mejorar la usabilidad de la preparación de los datos de HERMINWEB así como la reducción de las dependencias de diseño de las fuentes de información.

La conceptualización de las variables define los rasgos esenciales de los fenómenos y sus diferencias respecto a otros según la posición teórica del investigador [5]. Las **variables** sobre las que está sustentada la hipótesis son:

- Usabilidad: Es la medida en que HERMINWEB puede ser usado en otros diseños de información, con las mismas fuentes de datos.
- Dependencia: Es la medida en que HERMINWEB depende de datos fijos.

La operacionalización de las variables se basa en sustituir una variable por otras más concretas describiendo cómo medirlas para convertirlas en indicadores observables y cuantificables [5]. Para medir las variables de la investigación se definieron los siguientes indicadores.

⁵ Pieza de software que utiliza protocolos y estándares para intercambiar datos entre aplicaciones.

⁶ del inglés *Lightweight Directory Access Protocol* (*Protocolo Ligero de Acceso a Directorios*)

Variable Conceptual	Dimensión	Indicadores
Usabilidad	HERMINWEB	Cantidad de fuentes aceptadas
		Cantidad de campos aceptados
		Cantidad de atributos aceptados en base de datos
		Cantidad de tipos de usuarios aceptados
	MPD ⁷	Cantidad de fuentes aceptadas
		Cantidad de campos aceptados
		Cantidad de atributos aceptados en base de datos
		Cantidad de tipos de usuarios aceptados

Tabla 1 Indicadores para medir Usabilidad

Variable Conceptual	Dimensión	Indicadores
DEPENDENCIA	HERMINWEB	Cantidad de componentes modificados ante una nueva fuente
		Cantidad de componentes modificados ante un nuevo campo
		Cantidad de componentes modificados para eliminar una fuente
		Cantidad de componentes modificados para eliminar un campo
		Cantidad de funciones en base de datos modificados ante una nueva fuente
		Cantidad de funciones en base de datos modificados al eliminar una fuente
		Cantidad de funciones en base de datos modificados ante un nuevo campo
		Cantidad de funciones en base de datos modificados al eliminar un

⁷ Módulo de Preparación de Datos para HERMINWEB

		campo
		Cantidad de tablas modificadas al eliminar un campo
		Cantidad de tablas modificadas al eliminar una fuente
		Cantidad de clases modificadas ante una nueva fuente
		Cantidad de clases modificadas ante un nuevo campo
	MPD	Cantidad de componentes modificados ante una nueva fuente
		Cantidad de componentes modificados ante un nuevo campo
		Cantidad de componentes modificados para eliminar una fuente
		Cantidad de componentes modificados para eliminar un campo
		Cantidad de funciones en base de datos modificados ante una nueva fuente
		Cantidad de funciones en base de datos modificados al eliminar una fuente
		Cantidad de funciones en base de datos modificados ante un nuevo campo
		Cantidad de funciones en base de datos modificados al eliminar un campo
		Cantidad de tablas modificadas al eliminar un campo
		Cantidad de tablas modificadas al eliminar una fuente
		Cantidad de clases modificadas ante una nueva fuente
		Cantidad de clases modificadas ante un nuevo campo

Tabla 2 Indicadores para medir Dependencia

En cumplimiento a los distintos objetivos antes mencionados, se pusieron en práctica los siguientes **métodos de investigación** [5]:

❖ Métodos teóricos:

- **Analítico – sintético:** Posibilitó procesar toda la información enfocada hacia la investigación de las arquitecturas modulares y basadas en componentes que permitan implementar el módulo que de solución a los problemas de usabilidad de HERMINWEB.
- **Histórico – lógico:** Para conocer los antecedentes y tendencias actuales en la arquitectura modular, los sistemas basados en componentes y las herramientas de extracción, transformación y carga de datos, las plataformas, lenguajes de programación y las metodologías que guiarán el proceso de desarrollo del software, para poder seleccionar aquellas que permitan un mejor desarrollo del sistema.

❖ Métodos empíricos:

- **Experimentación:** Permitió el desarrollo de pruebas para la verificación de las funcionalidades implementadas, y del código definido con el fin de detectar errores y comprobar el correcto funcionamiento del negocio.
- **Observación:** Para observar cómo funciona HERMINWEB, como integra los datos de las principales fuentes y poder readaptarlo a un nuevo ambiente y para planificar cómo desarrollar el nuevo sistema.

El tema a desarrollar en el presente trabajo está estructurado en cuatro capítulos que agrupan el contenido de la siguiente forma:

Capítulo 1 Fundamentación Teórica: Incluye un estudio del estado del arte del tema tratado. Se abordan elementos teóricos de la investigación como el concepto de módulo, las metodologías, herramientas, lenguaje de desarrollo y se realiza un análisis de los sistemas basados en componentes y las ventajas de su implementación.

Capítulo 2 Características del Sistema: Se realiza el modelado de los procesos de negocio. Se explican los requisitos funcionales y no funcionales de la aplicación que dan paso a los casos de uso del sistema de los que se brinda una descripción detallada.

Capítulo 3 Diseño del Sistema: El capítulo abordará lo referente al diseño de la solución, se explicarán conceptos relacionados a cuestiones propias del sistema que fueron adaptados a la arquitectura definida.

Capítulo 4 Implementación y Pruebas: Muestra cómo quedará contenido el software dentro del hardware mediante el diagrama de despliegue. Explica en los diagramas de componentes la relación entre las clases y subsistemas principales de la aplicación. Define la estrategia de pruebas para la validación de la solución.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

El desarrollo de un software requiere de un análisis profundo de su estado en el mundo, y las ventajas que brinda su implementación, así como un estudio de las tecnologías que serán utilizadas para su desarrollo. En el presente capítulo se abordan los principales conceptos, definiciones, herramientas y metodologías que serán utilizadas durante el desarrollo de la aplicación.

1.2 Proceso de extracción del conocimiento en Bases de Datos (KDD)

El proceso de extracción del conocimiento en Bases de Datos surge con la necesidad del hombre de procesar y analizar grandes volúmenes de información y se define como el proceso no trivial de identificar patrones válidos, novedosos y potencialmente útiles y en última instancia potencialmente comprensibles a partir de los datos contenidos en algún repositorio de información [6]. Este proceso consta de varios pasos (Figura 1), a través de los cuales se creará un modelo para el análisis de la base de datos.

Entre los pasos fundamentales dentro del proceso KDD se encuentran la recogida y preparación y de datos. En estas etapas se seleccionan las fuentes de datos más interesantes, y es necesario conocer qué información es importante, y cómo obtenerla e integrarla. También se aplican técnicas para la selección, limpieza y transformación de los datos. El resultado obtenido de estas fases es un conjunto de filas y columnas denominado vista minable. La vista minable integra datos de diferentes fuentes, se limpian, seleccionan, transforman y se tipan para prepararlos para la modelización.

Otro de los pasos dentro del proceso KDD es la etapa de minería, donde se aplican técnicas para extraer patrones de interés a partir de los datos contenidos en las vistas minables.

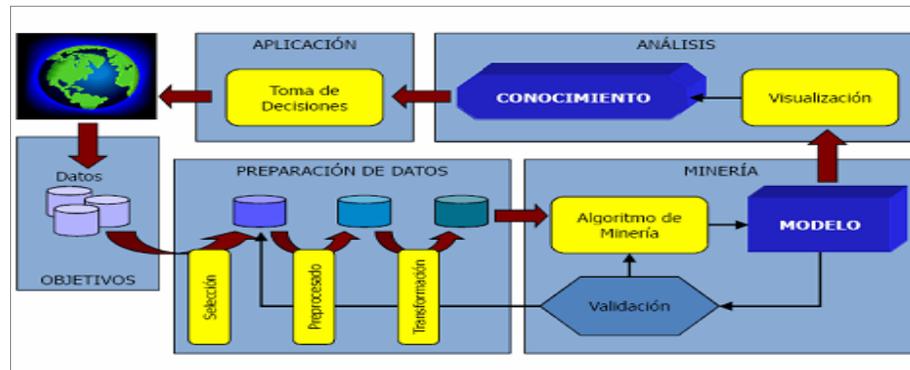


Figura 1: Etapas del proceso KDD [7]

Básicamente la minería de datos puede ser vista como la ciencia de explorar grandes conjuntos de datos para la extracción de información implícita, desconocida y potencialmente útil [8].

Es importante destacar que la minería de datos, en general, puede procesar distintos tipos de datos, de diferentes fuentes y dependiendo del tipo de dato a procesar es la clasificación de la tarea de minería. Uno de los tipos de minería que existe es la que se encarga de descubrir información o conocimiento potencialmente útil y desconocido a partir de la información de la Web; y dentro de ésta más específicamente se encuentra la *Minería de Uso Web* enfocada en el proceso de descubrimiento automático de patrones de acceso o uso de servicios de la Web, centrándose en el comportamiento de los usuarios cuando interactúan con la Web [9].

La Minería de Uso Web ayuda a descubrir tendencias y relaciones en el comportamiento de los usuarios cuando acceden a un sitio, partiendo de los datos que quedan registrados, como son los logs del servidor proxy. En la Universidad de las Ciencias Informáticas (UCI) se desarrolló una herramienta de minería de Uso de la web que permite extraer a partir del análisis de los registros del proxy, patrones de comportamiento de los usuarios, en el uso de las cuotas de navegación por Internet.

1.3 Herramienta de Minería de Uso de la Web aplicada a los Registros del Proxy (HERMINWEB)

HERMINWEB es una herramienta que automatiza un proceso KDD para extraer patrones de la navegación de los usuarios, mediante la aplicación de las tareas de minería Agrupamiento y Reglas de Asociación. Esta herramienta tiene un módulo de preparación de datos estático, específico para el diseño de la información de los usuarios de la UCI. La selección de los atributos a incluir en la vista minable está predefinida. Por cada fuente de información de las que se pueden seleccionar los datos se muestra un listado de atributos. De estos se pueden dejar atributos sin seleccionar, pero no se pueden agregar atributos nuevos. Por ejemplo HERMINWEB obtiene los datos de los trabajadores de los Servicios Web de Trabajadores Internos y Externos.

De los trabajadores externos se pueden obtener los siguientes datos: Área, Sexo, Edad, Cargo, Provincia y Categoría. De los trabajadores internos se obtiene: Área, Sexo, Edad, Cargo, Provincia, Militancia, Nivel Escolar y Tipo de Trabajo. Lo mismo sucede con los expedientes de usuario que se obtienen de LDAP y los datos de los estudiantes que se obtienen de la base de datos de Akademos. La conexión a la base de datos para la obtención de los datos necesarios se realizaba apoyada en la ETL Kettle de la suite de aplicaciones Pentaho, pero la información de LDAP y los Servicios Web se obtenía mediante código Python, por lo que los datos viajaban en flujos de información diferentes. HERMINWEB establece un concepto de sesión estático, definiéndolo como el grupo de peticiones de un usuario determinado, a un sitio web, desde un mismo edificio, en el mismo día y horario. Por ejemplo partiendo de las siguientes peticiones:

1- "Tarde" "Docente 1" 5942 "www.uci.cu" yordones

2 - "Tarde" "Docente 1" 15628 "www.cuba.cu" yordones

3 - "Noche" "Docente 1" 2639 "www.uci.cu" yordones

4- "Mañana" "Docente 5" 569832 "www.uci.cu" jfcruz

5- "Mañana" "Docente 5" 593258 "www.uci.cu" jfcruz

Se crean cuatro sesiones que serían: 1- (1), 2- (2), 3- (3), 4- (4,5). Esta definición no permite que sean agregados nuevos campos de los logs que resulten de interés para el análisis.

Como no se pueden agregar nuevos campos para el análisis, ya están predefinidos los campos a los que se le pueden realizar transformaciones, por ejemplo en los datos de los trabajadores internos solo se le pueden realizar transformaciones literales al tipo de trabajo, al nivel escolar y a la categoría. En el caso de los registros del proxy solo admite transformaciones literales la dirección IP. Para los datos que eran comunes entre todos los trabajadores también se realizan transformaciones a atributos como el área, la militancia, el sexo, el cargo y la provincia.

En la base de datos de la herramienta el diseño de la información también es estático, estando definidas previamente las tablas, con sus columnas y sus relaciones, por lo que también resulta imposible agregar nuevos atributos al análisis. Para la obtención de los patrones de la navegación es obligatorio analizar la información de los usuarios mezclada con la información de los logs, no se permite extraer patrones solamente a partir de los datos de la navegación.

Después de realizar un estudio detallado, se propone como solución al problema de la selección estática de la información durante la preparación de datos de HERMINWEB, el desarrollo de un sistema basado en componentes, que permita configurar de manera dinámica la obtención, transformación y almacenamiento de los datos, con el objetivo de poder incluir nuevas variables al análisis y adaptar la herramienta a otros entornos. El sistema se apoyará en la ETL Kettle de la suite de aplicaciones Pentaho para realizar el proceso de manera más eficiente.

1.4 Arquitecturas Modulares.

Un módulo es una porción de un programa de computadora que recibe como entrada la salida que haya proporcionado otro módulo, o los datos de entrada al sistema; produciendo una salida que puede ser utilizada por otro módulo o terminar la ejecución del sistema [10]. Se encarga de realizar una o varias de las tareas que debe ejecutar el software para cumplir con sus objetivos. Pueden estar organizados jerárquicamente en niveles, de manera que un módulo principal se encarga de hacer las llamadas a los módulos de los niveles inferiores.

Deben tener un tamaño relativamente pequeño, para aislar el impacto que puede tener un cambio en el programa o en uno de los algoritmos que lo componen. Además deben cumplir con la independencia modular, lo que implica que para desarrollar un módulo no es necesario conocer los detalles internos de

otros módulos, aumentando la flexibilidad y la reutilización del software. Cada módulo cuenta con una arquitectura N-Capas [11].

Una de las arquitecturas modulares más conocidas y utilizadas actualmente lo constituyen los sistemas basados en componentes.

Software Basado en Componentes.

Un componente de software se puede definir como una parte no trivial, casi independiente y reemplazable de un sistema que cumple una función dentro del contexto de una arquitectura bien definida. Un componente cumple con un conjunto de interfaces y provee la realización física de ellas. Una de las características más importantes de los componentes es que son reutilizables. Para ello los componentes deben satisfacer como mínimo el siguiente conjunto de características [1]:

Accesible solo a través de su interfaz: el componente debe exponer al público únicamente el conjunto de operaciones que lo caracteriza (interfaz) y ocultar sus detalles de implementación. Esta característica permite que un componente sea reemplazado por otro que implemente la misma interfaz.

Sus servicios son invariantes: las operaciones que ofrece un componente, a través de su interfaz, no deben variar. La implementación de estos servicios puede ser modificada, pero no deben afectar la interfaz. Una interfaz define el conjunto de operaciones que un componente puede realizar; estas operaciones son llamadas también servicios o responsabilidades. Las interfaces proveen un mecanismo para interconectar componentes y controlar las dependencias entre ellos.

Bajo el modelo de desarrollo de software basado en componentes, las nuevas aplicaciones se construyen mediante la integración o composición de componentes. La composición se puede concebir como una relación cliente-servidor entre dos componentes. El componente cliente solicita un servicio (operación) del componente servidor, el cual ejecuta la operación solicitada y devuelve los resultados al cliente. El servidor produce un resultado que es consumido por el cliente.

En esta investigación se propone la implementación de un software basado en componentes donde cada componente represente un paso dentro del proceso de preparación de los datos, capaz de interconectarse y definir las dependencias entre ellos, garantizando que en el momento necesario algún componente

pueda ser sustituido o agregado uno nuevo. Para garantizar la implementación de esta arquitectura se estará utilizando el framework Pyutilib Component Architecture [12].

Pyutilib Component Architecture

PCA⁸ está incluida dentro del paquete de software de PyUtilib. El corazón de PCA es provisto a través de un conjunto pequeño de clases dentro del paquete `pyutilib.component.core` de PyUtilib [12]. PyUtilib define una serie de conceptos para identificar las relaciones entre las clases del diseño:

Un plugin es una clase que implementa un conjunto de métodos relacionados en el contexto de una aplicación. Así, un plugin puede ser descrito como una definición de componente. Un service es una instancia de una clase plugin; ya sea de un plugin singleton o no singleton. Una clase interface describe las funcionalidades de los plugins. Una clase plugin incluye las declaraciones, que denotan, que implementa una o más interfaces. Una interfaz es precisada por métodos y datos que son utilizados. Un extension point es definido con respecto a una clase interfaz específica, proporcionando un mecanismo genérico a aplicaciones para utilizar la funcionalidad provista por otros servicios [12].

PCA incluye un registro de componentes global y un framework para automatizar la ejecución de servicios del plugin. Todos los plugins e interfaces automáticamente se registran en el registro, que luego actúa como un agente. Así, el desarrollador de la aplicación puede definir extension points sin saber cómo serán implementados, y los desarrolladores de plugin pueden registrar extensiones sin necesidad de conocer cómo o dónde son empleados [12].

Se decide utilizar PyUtilib porque resuelve el problema de la extensibilidad en cuanto a la adición de nuevos componentes al sistema. Ofrece la posibilidad de separar la implementación de la interfaz del componente, de su puesta en práctica, logrando una mayor flexibilidad en el diseño.

1.5 Pentaho Data Integration

⁸ del inglés PyUtilib Component Architecture

Pentaho es una alternativa de código abierto para la inteligencia de negocio. La ETL Kettle de la suite de aplicaciones Pentaho permite implementar los procesos de extracción, transformación y carga de datos. Con esta herramienta se construyen las transformaciones (mínimo nivel de diseño) utilizando los pasos (steps). PDI es un motor de transformación, donde los datos y sus transformaciones están separados. Las transformaciones y trabajos son almacenadas en formato XML, donde se especifican las acciones a realizar en los datos. Para construir las transformaciones, se utilizan los pasos o componentes, que se enlazan entre sí mediante saltos, que determinan el flujo de datos entre los diferentes componentes.

En la actual investigación se decide integrar el Módulo de Preparación de Datos para HERMINWEB con la ETL Kettle de la suite de Pentaho, para realizar las conexiones a las diferentes fuentes de datos, y transformar e integrar los datos seleccionados; por representar esta herramienta una solución completa para la integración de datos. Además brinda un amplio soporte multi plataforma, amplio soporte a fuentes de datos incluyendo aplicaciones empaquetadas, más de 30 bases de datos (open source y propietarias). Fácil integración de fuentes de datos, portales y aplicaciones [13]. Se utilizará la versión 4.0.1 que será ejecutada por líneas de comandos.

Metodología, Lenguajes y Herramientas de Desarrollo

El entorno de desarrollo, metodología, lenguaje de programación y modelado, así como herramienta CASE definidos para la construcción del software, fue producto de un estudio realizado para seleccionar aquellas que permitan la construcción de un sistema potente, eficiente y bien documentado. A continuación se brinda una breve descripción de cada herramienta a utilizar.

1.6 Sistema Gestor de Bases de Datos (SGBD)

PostgreSQL es un servidor avanzado de bases de datos, de código abierto. En las versiones anteriores de HERMINWEB se utiliza PostgreSQL como SGBD y en esta investigación se decide mantenerlo porque implementa un modelo cliente/servidor con funcionamiento multiprocesos en vez de multihilos, es decir que un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Es capaz de soportar los tipos de datos estándares de cualquier base de datos y brinda la posibilidad de definir nuevos

tipos. Su estrategia de almacenamiento en MVCC⁹, permite que mientras un proceso escribe en una tabla, otros procesos puedan acceder a ella sin necesidad de bloqueos, con lo que obtiene una mejor respuesta en ambientes de grandes volúmenes de datos. La posibilidad de definir herencia entre las tablas agiliza la obtención de datos. Cuando se trabaja con SGBD se necesita una herramienta para la gestión de las mismas, para administrar PostgreSQL se seleccionó PgAdmin III pues es una plataforma poderosa de administración y desarrollo para bases de datos PostgreSQL, gratis para cualquier uso. Los soportes gráficos de la interfaz lo hacen amigable y fácil de usar. Su utilización en esta investigación fue de gran importancia porque brinda la posibilidad de realizar consultas, administrar bases de datos, crear tablas, funciones, entre otros. Está disponible para una gran variedad de sistemas operativos.

1.7 Lenguaje de Programación: Python

Un lenguaje de programación es un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura, así como el significado de sus elementos y expresiones. Es utilizado para controlar el comportamiento físico y lógico de una máquina. Python es un lenguaje de programación con una sintaxis muy limpia y que favorece un código legible, interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos, con soporte para paquetes jerárquicos y error de excepción basada en la manipulación [14]. Facilita la realización de pruebas, detecta muchos de los errores de programación que escapan al control de los compiladores y proporciona información muy rica para detectarlos y corregirlos. Posee un rico juego de estructuras de datos que se pueden manipular de modo sencillo [15].

En la implementación anterior de HERMINWEB se utilizó Python como lenguaje de programación. En la versión actual se decide mantener la implementación en Python porque permite que el código de la aplicación sea más corto que su equivalente en otros lenguajes de programación y con el tipado dinámico se pudo manejar con mayor facilidad el cúmulo de información que se gestiona. Posee extensas bibliotecas estándar y módulos de terceros para prácticamente todas las tareas. Además la organización de los bloques de código hace más sencilla la lectura y comprensión de los mismos.

⁹ Del inglés Multiversion Concurrency Control.

Para la programación con Python es necesario un Entorno de Desarrollo Integrado (IDE) que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Cuando se habla de Python una de las opciones más atractivas para seleccionar un IDE es Eclipse.

Eclipse es un entorno de desarrollo integrado de código abierto, multiplataforma para desarrollar aplicaciones. La versión actual de Eclipse dispone de las siguientes características: editor de texto, resaltado de sintaxis, pruebas unitarias con JUnit y control de versiones con CVS¹⁰. Integración con Ant, herramienta para la realización de tareas mecánicas y repetitivas, durante las fases de compilación y construcción. Asistentes¹¹ : para creación de proyectos, clases, tests, entre otros., refactorización [16]. Para el desarrollo del módulo se decide utilizar Eclipse porque ofrece completamiento de código, resaltado de sintaxis, depuración de la ejecución y funcionamiento en varias plataformas.

1.8 Interfaces Gráficas de Usuario: Qt.

Los sistemas basados en componentes abstraen la implementación de la interfaz de su funcionamiento, siendo accesible los métodos solo a través de esta. Para la creación de las interfaces de usuario de la aplicación se seleccionó el framework Qt.

Qt es un framework para el desarrollo de aplicaciones multiplataforma, cuya función más conocida es la creación de interfaces de usuarios, aunque facilita otras tareas de programación como soporte para programación multihilos, conexión a bases de datos y manejo de cadenas. Utiliza C++ de manera nativa, pero ofrece soporte para otros lenguajes como Python mediante PyQt.

Para este framework existe un diseñador gráfico llamado Qt Designer. Esta herramienta está disponible para las mismas plataformas sobre las que está soportado el framework para el que construye las interfaces. Desarrolla interfaces multilenguaje debido a que genera un archivo XML cuyo contenido es el formato de dicha interfaz, pudiéndolo convertir con los programas pertinentes a cada lenguaje. Su utilización permitió la creación de las interfaces visuales de la aplicación sencilla y cómodamente, además de una fácil manipulación de las variables de configuración de cada uno de ellos [17].

¹⁰ del inglés Concurrent Version System.

¹¹ del inglés wizards

Qt ha sido extendido a otros lenguajes como Python. Existe una implementación para este framework llamada PyQt que establece una interfaz transparente para acceder desde Python a este marco de trabajo; así con la facilidad de Python sumada a la excelencia de Qt, se hace más agradable la programación visual. Cuenta con una amplia documentación proveniente de Qt y de Python a la vez. Con la utilización de PyQt durante el desarrollo de la aplicación se pudo crear una interfaz visual sencilla, pues este posee los componentes visuales necesarios para el desarrollo del software y una abundante documentación y ejemplos para la creación de las interfaces.

1.9 Metodología de Desarrollo del Software: RUP

El Proceso Unificado de Desarrollo (RUP) unido al lenguaje unificado de modelado constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. El ciclo de vida de esta metodología es soportado por la herramienta CASE Visual Paradigm, que permite representar todos los diagramas que se generan de cada una de las fases de RUP.

RUP¹² es un proceso que define como principales elementos: Quien (“Trabajadores”) hace que (“Artefactos”), como (“Actividades”) y cuando (“Flujo de Actividades”) lo hace. Está organizado en nueve grupos lógicos o flujos de trabajo, seis de ingeniería y tres de apoyo. Presenta cuatro fases conocidas como: Inicio, Elaboración, Construcción y Transición, por las que transcurren cada una de los nueve flujos de forma iterativa e incremental. [18]. El ciclo de vida de RUP se caracteriza por ser: *Dirigido por casos de uso, Iterativo e incremental y Centrado en la arquitectura.*

RUP fue seleccionada para dirigir la construcción del software porque permite avanzar de forma más efectiva de los requisitos a los demás flujos de trabajo agrupando las funcionalidades similares para que el proceso de software sea más comprensible. Brinda una arquitectura del ciclo de vida del software que es robusta y adaptable que permite sentar los cimientos de la arquitectura del sistema en los casos de uso arquitectónicamente significativos y minimiza el impacto de los riesgos. El proceso además está basado en el desarrollo en componentes, que se pone en práctica en la implementación de este sistema. RUP

¹² del inglés Rational Unified Process

utiliza el Lenguaje Unificado de Modelado (UML) para documentar todos los artefactos del sistema de software.

UML es un lenguaje para especificar, visualizar, construir y documentar los artefactos de los sistemas de software, así como también el modelado de negocios. Brinda la habilidad para analizar y diseñar un sistema desde la perspectiva de los objetos. RUP utiliza el Lenguaje Unificado de Modelado para preparar todos los esquemas de un sistema software. De hecho, UML es una parte esencial de RUP, sus desarrollos fueron paralelos[18] .

En esta investigación fue necesario utilizar UML para que el proceso de desarrollo quedara bien documentado en los manuales de usuario permitiendo una fácil comprensión por parte de otros equipos de desarrollo; además proporciona técnicas y especificaciones para la creación de diagramas que sirven de guías en el proceso de desarrollo del software.

Visual Paradigm para UML, es una herramienta de diseño que soporta todos los diagramas UML. Ofrece amplias características de modelado de casos de uso, incluyendo la función completa de Diagrama de Casos de Uso, editor de flujo de eventos, de casos de uso y la generación de diagramas de actividades. Los desarrolladores pueden diseñar la documentación del sistema con plantillas de diseño. [19].

Visual Paradigm fue escogido para esta investigación porque es una herramienta que utiliza UML como lenguaje de modelado y soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite realizar de forma rápida y organizada todos los diagramas propuestos por RUP. Se integra con Eclipse, Python y PostgreSQL y se puede utilizar en sistemas operativos como GNU/ Linux. Proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML.

1.10 Modelado de los Procesos de Negocio con IDEF0

IDEF¹³ consiste en una serie de normas que definen la metodología para la representación de funciones modeladas. Estos modelos consisten en una serie de diagramas jerárquicos junto con unos textos y

¹³ del inglés Integration Definition for Function Modeling

referencias cruzadas entre ambos que se representan mediante rectángulos o cajas y una serie de flechas. Uno de los aspectos de IDEF0 más importantes es que va introduciendo gradualmente más y más niveles de detalle a través de la estructura del modelo. Se utiliza como medio para comunicar las reglas y procesos de negocio y facilitar el análisis a la hora de identificar puntos de mejoras [20].

IDEF0 ha sido elegido para modelar esta investigación porque permite descomponer las funciones en diagramas más específicos hasta que se llegue al nivel necesario de descripción, con lo que facilita la detección de posibles áreas de mejoras. Es una técnica sencilla pero poderosa, capaz de explicar los procesos más complejos de una forma sencilla.

1.11 Protocolo de Comunicación entre Aplicaciones: ICE

Internet Communication Engine (ICE) es un middleware para el desarrollo de aplicaciones basadas en objetos distribuidos. Es ligero y abierto que pretende ser compatible con cualquier tipo de plataforma y con los principales lenguajes de programación [21]. Representa el modelo evolucionado del paradigma cliente/servidor que corresponde al concepto de objetos distribuidos. ICE define el protocolo de comunicación que soporta a través del sistema de comunicaciones, la invocación de métodos y el retorno de resultados. Define protocolos para TCP/IP, UDP y SSL. El protocolo ICE soporta mecanismos de compresión, que pueden ser configurados mediante propiedades ICE y sesiones de comunicación bidireccionales, las cuales son necesarias para atravesar los mecanismos de firewall [21].

Una aplicación ICE se puede usar en entornos heterogéneos: los clientes y los servidores pueden escribirse en diferentes lenguajes de programación, pueden ejecutarse en distintos sistemas operativos y en distintas arquitecturas, y pueden comunicarse empleando diferentes tecnologías de red. Además, el código fuente de estas aplicaciones puede portarse de manera independiente al entorno de desarrollo [22]. Se decide utilizar ICE para el procesamiento de los registros del proxy porque provee un conjunto de características que soportan el desarrollo de aplicaciones distribuidas en un amplio rango de dominio. Puede usarse en sistemas heterogéneos. Proporciona una implementación eficiente en ancho de banda, en uso de memoria, y en carga de CPU. Proporciona una implementación basada en la seguridad, lo que permite que sea usado en redes no seguras. Con su uso garantiza que se pueda manejar cuándo y de

qué forma distribuir el procesamiento de los registros en dependencia de la cantidad de nodos con los que se cuente.

Conclusiones del Capítulo

En este capítulo se ha realizado un resumen de los aspectos más importantes de la investigación para realizar la preparación del módulo de datos que permita a la herramienta HERMINWEB adaptarse a cualquier entorno en el que se desee aplicar y para ello se utilizarán las siguientes herramientas: Visual Paradigm como herramienta CASE, porque soporta el ciclo de vida completo del desarrollo de software, ayudando a una más rápida construcción de aplicaciones de calidad a un menor coste. Para modelar los procesos de negocio la metodología RUP capaz de adaptarse a las necesidades del cliente y entregar el producto en etapas iteradas, analizando en cada una de ellas la estabilidad y calidad del software, así como los riesgos involucrados; con la notación IDEF0 que mejora la toma de decisiones mediante gráficos ayudando en la identificación de las funciones que realiza el sistema y si lo hace bien o mal. En el IDE, Eclipse, para la programación con Python, una plataforma de código abierto que hace extensible su uso mediante módulos. Como lenguaje de programación Python, lenguaje de alto nivel, multiplataforma y con una sintaxis sencilla; y para el diseño e implementación de la interfaz de usuario, Qt, sistema integral para el desarrollo de aplicaciones multiplataforma. El protocolo ICE se utilizará para la comunicación entre aplicaciones porque proporciona servicios y herramientas para el soporte de las mismas. Para manejar los grandes volúmenes de información se seleccionó como SGBD PostgreSQL, potente plataforma de código abierto, que permite la gestión de grandes volúmenes de información, y bajos costos para su mantenimiento. Para la administración de este SGBD se utilizará PgAdmin III, aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, su interfaz gráfica soporta todas las características de PostgreSQL y facilita la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis y un editor de código para la parte del servidor. El software presentará una arquitectura en componentes garantizada por Pyutilib, capaz de abstraer la interfaz de su implementación, haciendo cada componente reutilizable en otras aplicaciones.

Capítulo 2: Características Del Sistema

2.1 Introducción

En el presente capítulo se abordan los procesos de negocio, diagramas y descripciones asociados a la preparación de los datos en términos de fuentes de información sobre los registros de navegación de los usuarios de un entorno distinto a la Universidad de las Ciencias Informáticas. Además, son relacionados los requisitos funcionales, no funcionales y el modelo de casos de uso del sistema del Módulo de Preparación de Datos para la herramienta HERMINWEB logrando que sea aplicable en un área fuera de la UCI.

2.2 Modelo de Negocio

Para la modelación de los procesos del negocio se utiliza la notación IDEF0, que garantiza la identificación de los mismos de una manera más simple. A continuación se muestran los procesos del negocio.

2.2.1 Representación de los procesos de negocio

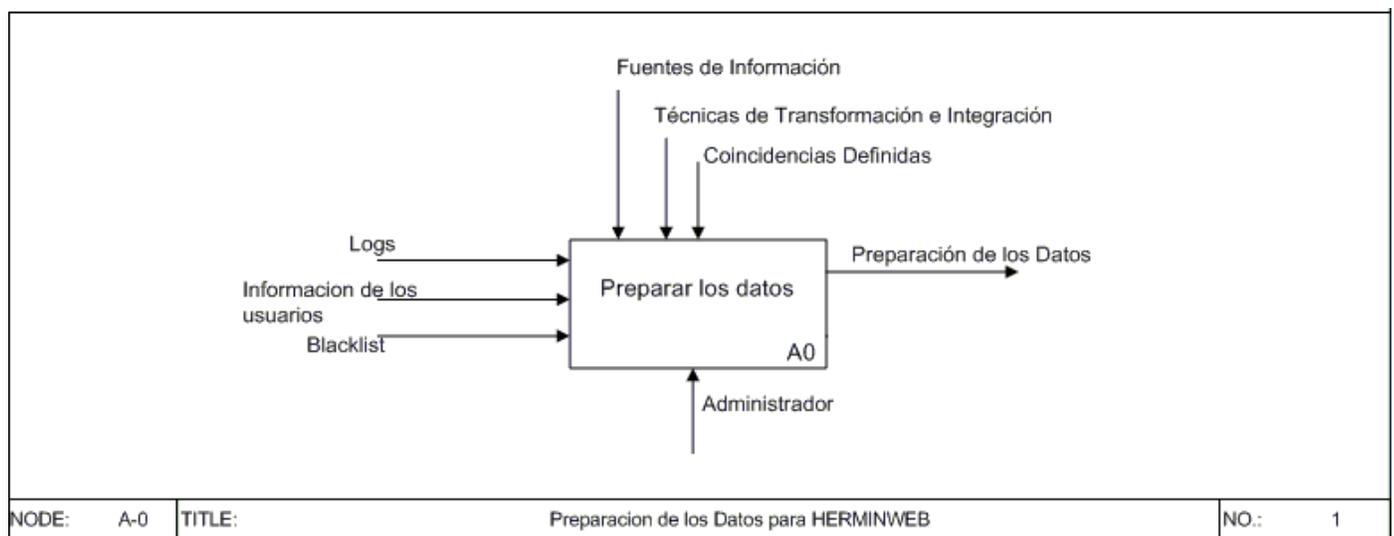


Figura 2. Macro-Proceso de Negocio Preparar Datos.

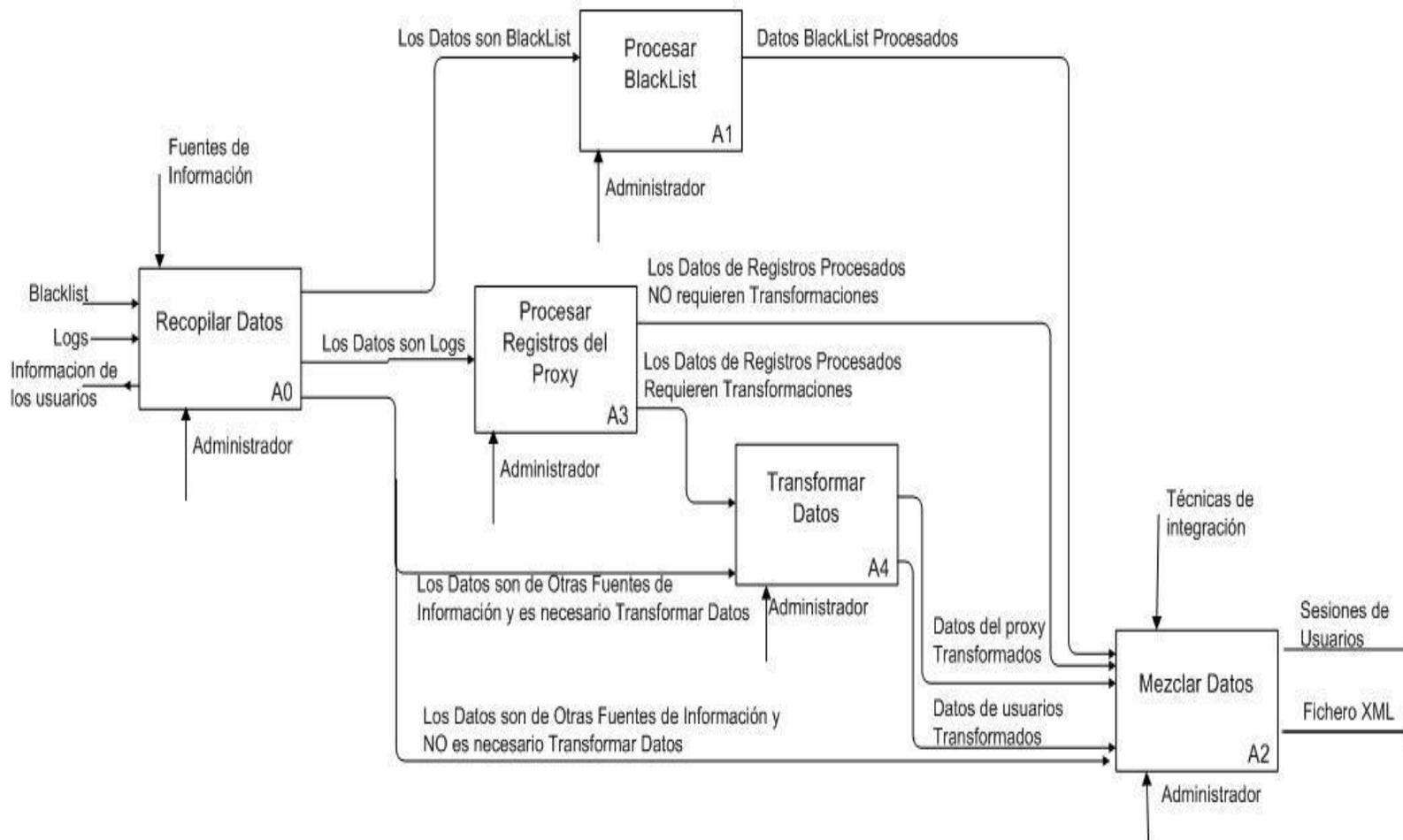


Figura 3 Proceso de Negocio Preparar Datos

Preparar Datos: Primero se recopilan los datos provenientes de las fuentes, teniendo presente los y atributos seleccionados para filtrar y crear las vistas minables, transformándolos de ser necesario; luego se procesan los registros del *proxy*. Finalmente se mezclan las clasificaciones de dominio con los datos de la navegación y los usuarios para obtener las sesiones de usuarios.

Recopilar Datos: Se recopilan todos los datos que se van a procesar (logs, blacklist¹⁴, otros datos almacenados en Base de Datos, servicios WEB o LDAP).

Procesar Blacklist: Se procesan todos los blacklist, obteniendo de ellos todas las clasificaciones de los dominios visitados por los usuarios.

Procesar Registros del Proxy: Se obtienen de los registros logs publicados en el servidor los datos seleccionados por el administrador para analizar el comportamiento de los usuarios en su navegación por Internet.

Transformar Datos: Se realizan transformaciones definidas por el Administrador, a los datos que así lo requieran.

Mezclar Datos: Se mezclan los datos obtenidos del procesamiento de los logs y las blacklist con los datos de los usuarios, obtenidos de las diferentes fuentes de datos, para crear las sesiones de usuario y las vistas minables.

2.6 Especificación de los requisitos de software

“El modelamiento del negocio brinda una visión general de los procesos que deben ser automatizados, permitiendo establecer una comunicación entre clientes y desarrolladores. La captura de los requisitos de software, es la clave del éxito en la producción de un software con calidad” [18].

2.6.1 Requisitos funcionales

¹⁴ *blacklist* es una lista donde se registran las direcciones *IPs* que generan spam de forma voluntaria o involuntaria.

“Los requisitos funcionales constituyen capacidades o condiciones que el sistema debe cumplir, los mismos se mantienen invariables sin importar con qué propiedad o cualidad se relacionen” [18].

A continuación se tratan los requisitos funcionales asociados al módulo:

R1: Procesar fuentes de datos: Obtiene los datos de los usuarios almacenados en las diferentes fuentes de datos.

R1.1: Procesar Bases de Datos: Establece la conexión a la base de datos y obtiene la información almacenada.

R1.2: Procesar Servicios Web: Establece la conexión a un Servicio Web y obtiene la información almacenada.

R1.3: Procesar LDAP: Establece la conexión a un LDAP y obtiene la información almacenada.

R2: Procesar Blacklist: Obtiene las clasificaciones de las páginas contenidas en los ficheros de blacklists y las almacena.

R3: Procesar los registros del proxy: Obtiene la información de la navegación de los usuarios almacenadas en los logs y la almacena.

R4: Transformar Datos: Transforma los datos, estableciendo los rangos y etiquetas definidos por el cliente.

R4.1: Transformar Nominal – Nominal. Transformación utilizada para convertir una cadena a otra.

R4.2: Transformar Numérico - Nominal (rangos). Transformación utilizada para convertir un número a una cadena.

R4.3: Transformar Edad a rangos: Transformación utilizada para etiquetar una edad determinada en un rango predeterminado.

R4.4: Transformar Horario a rangos: Transformación utilizada para etiquetar un horario determinado en un rango predeterminado.

R4.5: Clasificar Subred: Dado un Número IP clasificar la subred a la que pertenece.

R5: Preparar Datos: Se refiere a que una vez seleccionados y configurados los componentes, que forman parte del procesamiento de los datos, se ejecuta la preparación de los datos.

R6: Salvar Sesión: Es el componente que te da la posibilidad de agrupar todas las peticiones de un usuario y establecer nuevos atributos que serán utilizados en la extracción de patrones.

RF7: Configurar Componente: Se refiere a permitir configurar cualquier componente mediante una pantalla que solicita la información necesaria al administrador.

RF7.1: Configurar Componentes Entradas: Identifica las entradas de datos y permite configurar las conexiones de acuerdo a su tipo (Logs, Blacklist, Base de Datos, Servicio Web o LDAP)

RF7.2: Configurar Componentes Salidas: Permite configurar las opciones de conexión para la base de datos donde será almacenada la información después de realizada la preparación de los datos.

RF7.3: Configurar Componentes Transformación: Permite configurar las transformaciones según el tipo seleccionado.

RF7.4: Configurar Componentes Tablas: Configura las acciones a realizar sobre las tablas de la base de datos donde están almacenados los datos preparados.

RF8: Probar la conexión: Cuando se configura el acceso a una fuente de información permite comprobar si se puede establecer la conexión.

RF9: Seleccionar una muestra de datos: Permite seleccionar una muestra de datos de la vista minable para analizarla.

RF10: Ordenar datos: Permite ordenar los datos obtenidos de una fuente de información antes de integrarlos con los datos de otra fuente.

RF11: Integrar datos: Permite integrar los datos de los usuarios, con las clasificaciones de dominio y los registros de la navegación del proxy.

RF12: Salvar en la base de datos: Almacena la vista minable en la base de datos.

2.6.2 Requisitos no funcionales

Usabilidad

Se debe garantizar un acceso fácil y rápido a la aplicación para los usuarios autorizados de cada empresa. El sistema podrá ser usado solo por personas autorizadas en las empresas.

Soporte

Una vez desplegado el software se continuará asesorando a los clientes durante 2 semanas, además se realizará un mantenimiento total del producto cada semestre, adaptándolo a nuevas necesidades en caso solicitado.

Software

- En la estación de trabajo donde se utilice la herramienta deberá estar instalada la distribución de GNU/Linux: Debian, en sus versiones 6.04 o superior.
- Es necesaria la previa instalación del intérprete para el lenguaje Python, este lenguaje se utilizará en su versión 2.7.
- Es necesaria la previa instalación de la Máquina Virtual de Java (JDK).
- Se requiere un servidor de base de datos PostgreSQL instalado y configurado.
- Se requiere un servidor web Apache instalado y configurado en un ordenador que permita la realización de conexiones utilizando el protocolo SSH¹⁵, en el cual se compartirán los registros del proxy que serán procesados.

Han de estar instaladas las bibliotecas de Python:

¹⁵ del inglés Secure Shell

python-dev, pyRXP-1.13 , pyro 3.10, python-paramiko, python-pexpect, python-ldap, python-dnspython, python-pyparsing 1.5.2, python-impacket, python-reportlab, python-numpy, python-soappy, python-crypto 2.3, python-psycog2, python-sqlalchemy, python-qt4, pyqt4-dev-tools, python-qt4-dev, python-pymssql, python-zero-ice, python-xlwt, python-pycha, python-cairo, python-pymongo, libnspr4-dev, libnss3-dev.

Hardware

Los requisitos de hardware están estrechamente relacionados con la capacidad de almacenamiento y procesamiento, debido a la gran cantidad de información que se procesará y almacenará.

Como características mínimas que debe cumplir el hardware donde se instalará la herramienta se tienen:

- Procesador Dual Core a 2.1 GHz.
- 1 GB de RAM¹⁶.
- 300 MB de espacio libre en disco duro.

Es necesario aclarar que la cantidad de espacio en disco duro disponible debe tomarse en consideración con la cantidad de datos que serán procesados, la cual puede variar desde unos pocos Megabytes (MB) hasta crecer considerablemente. Se acepta la disponibilidad de varios nodos de procesamiento (otras computadoras) que servirán de apoyo en el procesamiento de los registros del proxy.

Seguridad

Disponibilidad: El sistema debe de estar disponible siempre que se necesite ejecutarlo. En caso de no encontrarse disponible alguna de las fuentes de datos el sistema debe informar al usuario esta situación.

Confidencialidad: La aplicación asegura que cada usuario solo pueda entrar al sistema autenticándose y con previa autorización. Las carpetas donde se reorganizan y comparten los registros de navegación de los usuarios por defecto deben estar protegidas con contraseña. Para reorganizar los registros se debe utilizar un protocolo seguro para la comunicación entre HERMINWEB y el servidor web.

¹⁶ del inglés Random Access Memory

Integridad: La información manejada por el sistema será objeto de cuidadosa protección.

Legales

Se prohíbe vender, reproducir o comercializar esta aplicación sin el debido permiso de los autores de la misma y los directivos de la Universidad de las Ciencias Informáticas.

2.7 Modelo de Casos de Uso del Sistema

2.7.1 Actores del Sistema

Actor	Descripción
Cliente	Persona que necesita conocer los patrones de navegación de los usuarios, buscando ayuda para la toma de decisiones.
Pentaho	Representa la ETL Kettle de la suite Pentaho donde se realizan las transformaciones de los datos.
Intermediario ICE	Aplicación utilizada en el procesamiento de los registros de navegación.

Tabla 2 Descripción de los actores del sistema

2.7.2 Diagrama de Paquetes

Los CU fueron agrupados en 3 paquetes principales en relación con los procesos de negocio descritos. En el paquete Fuentes de Información se encuentran los casos de uso relacionados con el procesamiento de las fuentes de información. El paquete Transformaciones agrupa los casos de uso que intervienen en el proceso de transformaciones y salva de datos y el paquete Configuración contiene los casos de uso que intervienen en la configuración de los componentes, necesario para el funcionamiento de los casos de uso agrupados en los demás paquetes.

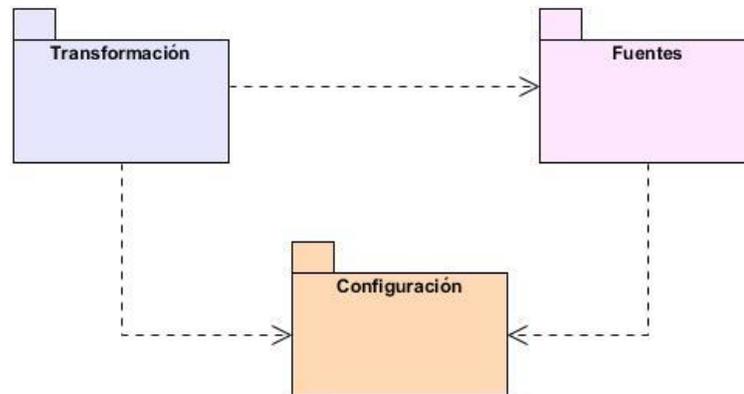


Figura 4 Diagrama de paquetes de Caso de Uso del Sistema

2.7.3 Casos de Uso del Sistema

“Los Casos de Uso (CU) son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Por tanto, establece un acuerdo entre clientes y el grupo de desarrollo sobre las condiciones y posibilidades (requisitos) que debe cumplir el sistema” [18].

Casos de Uso del Sistema:

CU1: Procesar Fuentes de Datos.

CU2: Procesar Blacklist.

CU3: Procesar los registros

CU4: Preparar Datos

CU5: Transformar Datos

CU6: Mezclar Datos.

CU7: Configurar Componentes Entrada

CU8: Configurar Componentes Salida

CU9: Configurar Componentes Tablas

CU10: Configurar Componentes Transformación

2.7.4 Diagrama de Casos de Uso del Sistema.

A continuación se presenta el Diagrama de casos de Uso del Sistema agrupados por paquetes. Cada color representa el paquete al que pertenece el Caso de Uso.

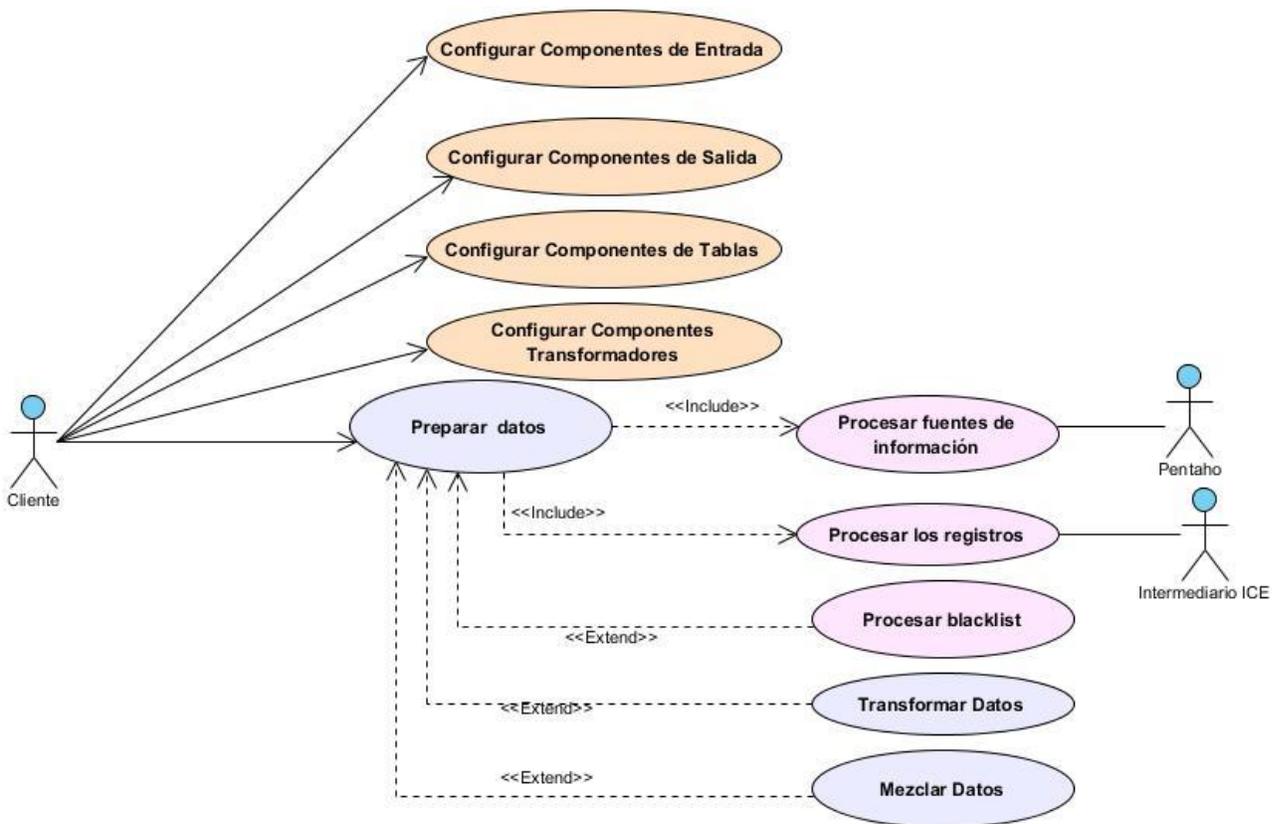


Figura 5 Diagrama de Casos de Uso del Sistema agrupados por paquetes

2.7.5 Descripción detallada de los Casos de Uso Críticos.

En esta investigación todos los CU son críticos para el funcionamiento del sistema pero solo serán descritos en esta sesión los CU que describen el proceso básico de la preparación de los datos, los restantes se pueden encontrar en los anexos [Anexo 1](#).

Descripción detallada del Caso de Uso “Preparar Datos”.

CU- 4	Preparar Datos	
Actor	Cliente	
Resumen	Integra la información obtenida de las diferentes fuentes con la obtenida del análisis de los ficheros logs y las blacklist por las coincidencias definidas por el cliente.	
Precondiciones	Configuración completada.	
Referencias	CU-1(inclusión), CU-3(inclusión), CU-5(extensión), CU-2(extensión), CU-6(inclusión),	
Flujo normal de eventos		
Sección "Ejecutar Transformación"		
Actor	Sistema	
1. El cliente selecciona la opción "Ejecutar Transformación".	2. Verifica que existan los componentes obligatorios: Nota: Verifica que se encuentre en el espacio de trabajo: <ul style="list-style-type: none"> • Uno o varios componentes de entrada. • Uno o varios componentes de salida. • Un solo componente Salvar Sesión. • Al menos un componente integrador si se tienen más de un componente de entrada. 	
	3. Verifica que todos los componentes estén configurados.	
	4. Verifica que el componente de entrada es Logs e Invoca el CU Procesar los Registros .	
	5. Verifica que existan componentes de Transformación e Invoca el CU Transformar Datos .	
	6. Invoca al CU Mezclar Datos .	
Flujo alternativo 2a: Faltan Componentes		
Actor	Sistema	

	2a.1 Verifica que no existan componentes de entrada y/o salida.
	2a.2 Muestra un mensaje especificando que faltan componentes necesarios para ejecutar la transformación.
Flujo alternativo 3a: Componentes NO Configurados	
Actor	Sistema
	3a.1 Verifica que exista al menos un componente sin configurar.
	3a.2 Muestra un mensaje indicando cuál(es) componente(s) está(n) sin configurar. Nota: se detiene la ejecución del CU hasta que se configuren los componentes y se seleccione la opción de Ejecutar Transformación nuevamente.
Flujo alternativo 4a: Componente de entrada es Blacklist	
Actor	Sistema
	4a.1 Verifica que el componente de entrada es Blacklist e Invoca el CU Procesar Blacklist . Ir a la Sección 6.
Flujo alternativo 4b: Componente de entrada es de otras Fuentes de Información	
Actor	Sistema
	4b.1 Verifica que el componente de entrada es otra fuente de información e Invoca el CU Procesar Fuentes de Información . Ir a la Sección 5.
Flujo alternativo 5a: Faltan Componentes de Transformación.	
Actor	Sistema
	5a.1 Verifica que no existan componentes de transformación.

Ir a sección 6.

Descripción detallada del Caso de Uso “Transformar Datos”.

CU- 5	Transformar Datos
Actor	Interesado
Resumen	Transforma los datos, estableciendo los rangos y etiquetas para la información.
Precondiciones	Configuración completada
Referencias	
Flujo normal de eventos	
Sección “Transformar Datos”	
Actor	Sistema
	1. Obtiene la configuración de los componentes transformadores.
	2. Realiza las transformaciones necesarias a los datos teniendo presente las configuraciones realizadas por el administrador.

Descripción detallada del Caso de Uso “Procesar los registros (logs)”.

CU- 3	Procesar los registros.
Actor	Intermediario ICE
Resumen	Obtiene la información de la navegación de los usuarios almacenadas en los logs y la almacena en la base de datos.
Precondiciones	Logs publicados en el Servidor Web.
Referencias	
Flujo normal de eventos	
Sección “Procesar los registros del proxy”	
Actor	Sistema

	1. Obtiene la configuración del componente Logs.
	2. Establece conexión con el directorio donde se encuentran almacenados los Logs.
	3. Con la configuración de los logs procesa de forma distribuida en los nodos de procesamiento para lo que se siguen los siguientes pasos: 3.1 Recibe la tarea. 3.2 Procesa los logs. 3.3 Obtiene el identificador de los usuarios. 3.4 Obtener la información de los usuarios.
	4. Une los flujos de información.
Flujos alternos 2a: Error en la conexión al servidor.	
Actor	Sistema
	2a.1 Error en la conexión al servidor: el sistema muestra un mensaje de error y muestra la interfaz para gestionar la configuración.

Descripción detallada del caso de uso “Mezclar Datos”.

CU- 6	Mezclar Datos.
Actor	Cliente
Resumen	Crea la tabla de sesiones en la Base de Datos y almacena en esta la información de los registros de navegación del proxy integrada con la información de los usuarios.
Precondiciones	Datos obtenidos y transformados. Componentes integradores configurados.
Referencias	
Flujo normal de eventos	
Sección “ Mezclar Datos”	

Actor	Sistema
	1. Obtiene de los componentes integradores, las coincidencias para integrar los datos.
	2. Mezcla los datos del procesamiento de los registros con las blacklist y la información de los usuarios obtenida de las diferentes fuentes de datos teniendo presente las coincidencias obtenidas del paso 1.
	3. Obtiene del componente Salvar en Base de Datos las configuraciones.
	4. Se conecta a la base de datos y almacena la información de los usuarios, unida con la información de la navegación para cada usuario.
Flujos alternos 4a: Error en la conexión al servidor.	
Actor	Sistema
	4a.1 Error en la conexión al servidor: el sistema muestra un mensaje de error y muestra la interfaz para gestionar la configuración.

Descripción detallada del caso de uso “Procesar fuentes de datos”.

CU- 1	Procesar fuentes de datos.
Actor	Cliente
Resumen	Configura la conexión a las fuentes de datos permitiendo extraer la información necesaria.
Precondiciones	Fuentes de datos disponibles.
Referencias	
Flujo normal de eventos	
Sección “Procesar Fuentes de Datos”	

Actor	Sistema
	1. Obtiene los datos de configuración de los componentes correspondiente a cada fuente de datos seleccionada.
	2. Establece la conexión al servidor, obtiene los datos de las tablas y los muestra.
3. Selecciona la información que sea de su interés para realizar la preparación de los datos.	4. El sistema almacena los datos temporalmente para poder realizar operaciones sobre ellos.
Flujo alternativo 2a: Verificar conexión al servidor.	
Actor	Sistema
	2a.1 Error en la conexión al servidor: el sistema muestra un mensaje de error y muestra la interfaz para gestionar la configuración.

Descripción detallada del Caso de Uso "Procesar Blacklist".

CU- 2	Procesar Blacklist.
Actor	Cliente
Resumen	Configura la conexión a la fuente donde se encuentran almacenados los Blacklist y los obtiene.
Precondiciones	Fuente de datos donde se almacenan los blacklist disponibles.
Referencias	
Flujo normal de eventos	
Sección "Procesar Blacklist"	
Actor	Sistema
	1. Obtiene del componente Blacklist las configuraciones de la conexión al directorio donde se encuentran las

	mismas almacenadas.
	2. Se conecta al directorio y obtiene el listado de Blacklist.
Flujo alternativo 2a: Verificar conexión al servidor.	
Actor	Sistema
	2a.1 Error en la conexión al servidor: el sistema muestra un mensaje de error y muestra la interfaz para gestionar la configuración.

Conclusiones del Capítulo

En el presente capítulo se realiza la propuesta del modelo de negocio, basado en los procesos que serán automatizados. Estos procesos fueron modelados con la notación IDEF0, que permite realizar diagramas con un nivel de detalle capaz de ofrecer una clara comprensión de los procesos y los mecanismos que los realizan. Se realizó la definición de los requisitos funcionales y no funcionales de la aplicación. De los requisitos funcionales se obtuvieron los Casos de Uso, prestando especial atención a los que representan una funcionalidad crítica para el sistema. Se ofrece una descripción detallada de estos casos de uso que serán una de las principales entradas del siguiente capítulo, siendo los mismos el soporte arquitectónico de la aplicación. Estos casos de uso fueron agrupados por paquetes en relación con los procesos de negocio definidos.

Capítulo 3: Diseño Del Sistema

3.1 Introducción

En el presente capítulo se expone una vista general de la arquitectura del sistema, así como los patrones de diseño utilizados. Se modelan las clases utilizadas en la implementación y sus relaciones, además de mostrar el modelo de datos. Describe de manera general como será realizado el sistema a partir de todas las funcionalidades previstas.

3.2. Arquitectura

HERMINWEB fue implementado sobre una arquitectura N- Capas, siendo la clase Principal quien comienza la ejecución del mismo. Para el desarrollo de este trabajo se mantendrá esta arquitectura como se muestra en la Figura 5. En la capa GUI se implementan interfaces gráficas de usuario que serán mostradas por el sistema. La capa Entornos de Ejecución representa la abstracción entre la capa GUI y la de Servicios; esta última contiene toda la lógica del negocio. El acceso y la modificación de los datos se realizan a través de la capa del mismo nombre. El sistema se apoya sobre el Framework (implementado por el equipo de desarrollo del proyecto Servicios Telemáticos) para la utilización de variadas funcionalidades como: conexión a bases de datos, control de procesos externos, ejecución de funciones en servidores remotos, entre otros. Todas las configuraciones de las interfaces e implementaciones del sistema son almacenadas en archivos XML, los cuales son utilizados por el Framework para el conocimiento de las clases empleadas, así como las interfaces que las mismas implementan [23]. Se ha incluido también el Framework Pyutilib, para garantizar una arquitectura basada en componentes, donde todos los componentes se relacionen pero con una mínima dependencia entre sí.

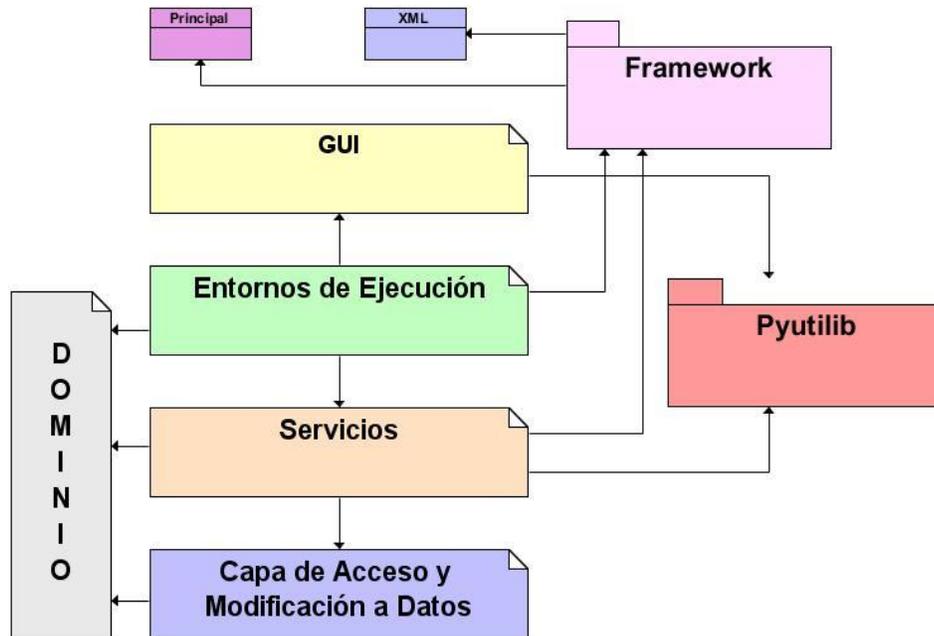


Figura 6 Arquitectura del Sistema

3.3 Patrones de Diseño

Un patrón de diseño es un par problema/solución con nombre que se puede aplicar en nuevos contextos, con consejos acerca de cómo aplicarlo en nuevas situaciones y discusiones sobre sus compromisos [24], que ayudan a que nuestro software alcance una mayor reusabilidad, flexibilidad y portabilidad. En el desarrollo de este trabajo se han mantenido algunos de los patrones implementados en HERMINWEB y se han agregado otros que se explican a continuación.

Proxy y Abstract Factory:

El patrón **Proxy** proporciona un representante o delegado que se encargue de controlar el acceso a un objeto, generalmente con motivos de eficiencia. Posee la ventaja de permitir el acceso a objetos que residen en espacios distintos de memoria, abstrayendo al programador de la ubicación del objeto solicitado [4].

El patrón **Abstract Factory**, brinda una interfaz para la creación de familias de objetos interdependientes o interrelacionados, sin la necesidad de especificar sus clases concretas. Potencia el encapsulamiento e incrementa la flexibilidad del diseño. Esto ayuda a modificar las clases que son encapsuladas sin la necesidad de realizar cambios que puedan representar trastornos en la implementación que las ha usado [4].

Estos patrones fueron implementados en la iteración anterior de HERMINWEB y fueron utilizados en la creación de los consultores para las conexiones a las distintas bases de datos. Se hizo necesario su uso pues el acceso a base de datos de la herramienta se ha de realizar a distintos gestores de bases de datos, los cuales no tienen las mismas interfaces de acceso [4].

Facade

Proporciona una interfaz de alto nivel unificada para facilitar el acceso a los métodos. Se utiliza para simplificar los métodos expuestos de una clase u objeto facilitando considerablemente su uso [4]. Se utiliza en la creación de una interfaz para cada una de las clases que administran el proceso de preparación de los datos. Con este patrón se simplifica el acceso a los diferentes plugin que implementan las funcionalidades que tiene la aplicación, y él se comunicaría con los subsistemas a través de la fachada, en este caso la interfaz IComponentXML. Dicha interfaz reenvía las peticiones a los objetos apropiados, logrando de esta forma que los que utilicen la fachada no necesiten acceder directamente a las subclases del módulo.

Observer y Chain of Responsibility

Observer define una dependencia uno a muchos entre objetos, de modo que cuando el estado de un objeto cambia, se notifica el cambio a todos los que dependen de él y se actualizan de forma automática [25].

El patrón **Chain of Responsibility** evita acoplar el emisor de una petición a su receptor, dando a más de un objeto la posibilidad de responder a la petición. Encadena los objetos receptores y pasa la petición a través de la cadena hasta que es procesada por algún objeto [25].

Fue necesario utilizar estos dos patrones en conjunto, pues en el sistema se ejecutarán en determinadas circunstancias procesos de manera concurrente, sobre los cuales en todo momento se necesitará conocer su estado, además de poder comunicarse entre sí.

Bridge

Desacopla una abstracción de su implementación, de modo que ambas puedan variar de forma independiente [25]. En este caso fue necesaria la utilización del patrón para poder luego desacoplar las implementaciones de las herramientas para la preparación de datos, las cuales pueden ser modificadas o sustituidas sin causar considerables modificaciones en la implementación de HERMINWEB.

Componentes.

Un componente de software es una unidad de composición con interfaces especificadas contractualmente y dependencias del contexto explícitas [26]. Fue utilizado en la implementación de componentes para cada una de las posibles transformaciones a los datos y las posibles entradas y salidas de tablas.

Decorator

Decorator brinda la posibilidad de añadir dinámicamente funcionalidades a un objeto. Esto ofrece un poco más de flexibilidad que la herencia a la hora de extender funcionalidades evitando que las clases más altas de la jerarquía se carguen de funcionalidades [4].

En determinados escenarios, se hace necesario brindarle a una función la posibilidad de ser ejecutada en varios hilos (mecanismo mediante el cual se puede dividir la aplicación en partes que pueden ejecutarse de manera concurrente), mediante este patrón se hace posible aplicarle esta funcionalidad en tiempo de ejecución sin la necesidad de modificar el código de la función o crear una clase con esta nueva funcionalidad multihilos [4].

Iterator

Este patrón define una interfaz que declara los métodos necesarios para poder acceder de forma secuencial a una colección de objetos. Las clases que utilizan esta interfaz para acceder a los objetos lo

hacen de forma independiente de la que implementa la interfaz. Permite que la estructura interna a la que se accede para iterar permanezca oculta [4].

Se aplica dicho patrón pues en numerosas ocasiones es necesario crear objetos que se comporten como secuencias, permitiendo además una personalización del acceso al contenedor interno sin necesidad de cargar todos los datos en memoria [4].

Alta Cohesión y Bajo Acoplamiento.

El patrón **Bajo Acoplamiento** es la medida de cuánto una clase está conectada (tiene conocimiento) de otras clases. Es un patrón evaluativo: un bajo acoplamiento permite que el diseño de clases sea más independiente. Reduce el impacto de los cambios y aumenta la reutilización [24].

Este patrón se utiliza para lograr la menor relación entre las clases, de tal forma que en caso de producirse una modificación en alguna de ellas, tenga la mínima repercusión posible en el resto de las clases. Potenciando así la reutilización y la mínima dependencia.

Alta Cohesión es una medida que indica cuán relacionadas están las responsabilidades de una clase. Es un patrón evaluativo: entre más alta cohesión más fácil de entender, de cambiar, de reutilizar [24].

Este patrón garantiza que las clases con responsabilidades estrechamente relacionadas no realicen un trabajo enorme, y que cada elemento del diseño realice una labor única dentro del sistema. La utilización de estos dos patrones en conjunto permitió relacionar todos los componentes del sistema, pero con una mínima dependencia entre ellos, dándole a cada uno la función de realizar una tarea específica dentro del proceso de preparación de datos.

3.4 Seguridad en el Sistema.

La seguridad es un aspecto importante a tener presente en el desarrollo de los sistemas informáticos. En HERMINWEB, una de las partes críticas en cuanto a seguridad es la entrada de los datos, la cual se puede realizar de dos formas: especificándolos visualmente o definiéndolos en un fichero de configuración. En ambos casos es necesario validar que posean el correcto formato y tipo para evitar que se introduzcan datos que comprometan el correcto funcionamiento de la herramienta [4].

Varias de las entradas de datos son credenciales; para la conexión a las base de datos, al servidor web, al servicio de directorio activo, así como la cuenta del remitente para el envío de las notificaciones por correo electrónico. Usualmente las aplicaciones de forma general definen que en sus ficheros de configuración las credenciales se pongan en texto claro, por varias razones; una es, hacer la especificación de los mismos de forma sencilla y no compleja al no tener que depender de otra herramienta para poder enmascarar dichas credenciales, además en el proceso de enmascarar se necesita una clave o llave, que es la que permitirá convertir esos datos a otro formato no comprensible y con la cual se hará el proceso inverso para obtener el texto claro que es el formato que se usa para las diferentes conexiones [23].

En la clave y en el método de enmascaramiento o cifrado (como más comúnmente se conoce) es donde radica la fortaleza del cifrado, por lo tanto la construcción de dicha clave no puede ser un proceso trivial, se recomienda que no se use una clave predeterminada, sino que se construya una diferente para cada vez que se vaya a utilizar en un ámbito distinto [4].

En el desarrollo de la primera iteración de HERMINWEB se realizó un estudio con el fin de definir la solución que se emplearía para dicho problema, detectando que era necesario utilizar una clave predeterminada para cifrar y descifrar los datos y que para la construcción de la misma se requiere una cuenta de administrador por las características del sistema operativo para el cual se desarrolló la herramienta. Esto crearía otro problema de seguridad, al tener que ejecutar la herramienta con una cuenta de administración, posibilitando a un atacante obtener todos los permisos para hacer lo que deseara con el equipo donde se estuviera ejecutando la aplicación, si lograra comprometerla. Al no poder construir la clave y como no es recomendable utilizar una clave predeterminada, durante el desarrollo de HERMINWEB, se llegó a la conclusión de que lo más factible era que las credenciales y todos los datos de configuración se almacenaran en el fichero de configuración en texto claro, centrando la seguridad de los datos en el administrador o usuario, el cual deberá proteger dicho fichero, dándole los permisos necesarios y suficientes para que solo puedan acceder a él los autorizados. Dicho esto, en la presente investigación se respetó este criterio y se mantuvieron en texto claro los datos en el archivo de configuración [23].

Para utilizar HERMINWEB es necesario autenticarse, lo que verifica que el usuario que intenta utilizar la aplicación realmente tiene la autorización para hacerlo. HERMINWEB permite acceder a un Servidor Web

que requiera autenticación teniendo presente que en él se comparten los registros del Proxy cuya información es de carácter sensible [4]. En la presente investigación se decide continuar utilizando las políticas de seguridad definidas para HERMINWEB en investigaciones anteriores y que fueron descritas.

3.5 Mejoras en el proceso de preparación de los datos.

Se diseñó un módulo (MPD¹⁷) que se encarga de realizar la preparación de los datos para HERMINWEB, capaz de realizar esta fase del proceso KDD en cualquier diseño de información. Es un sistema basado en componentes donde cada componente representa un paso dentro del proceso de preparación, que se configura de manera independiente y se relaciona con el componente adyacente a él mediante conectores que convierten la salida de un componente en entrada de otro (Figura 7).

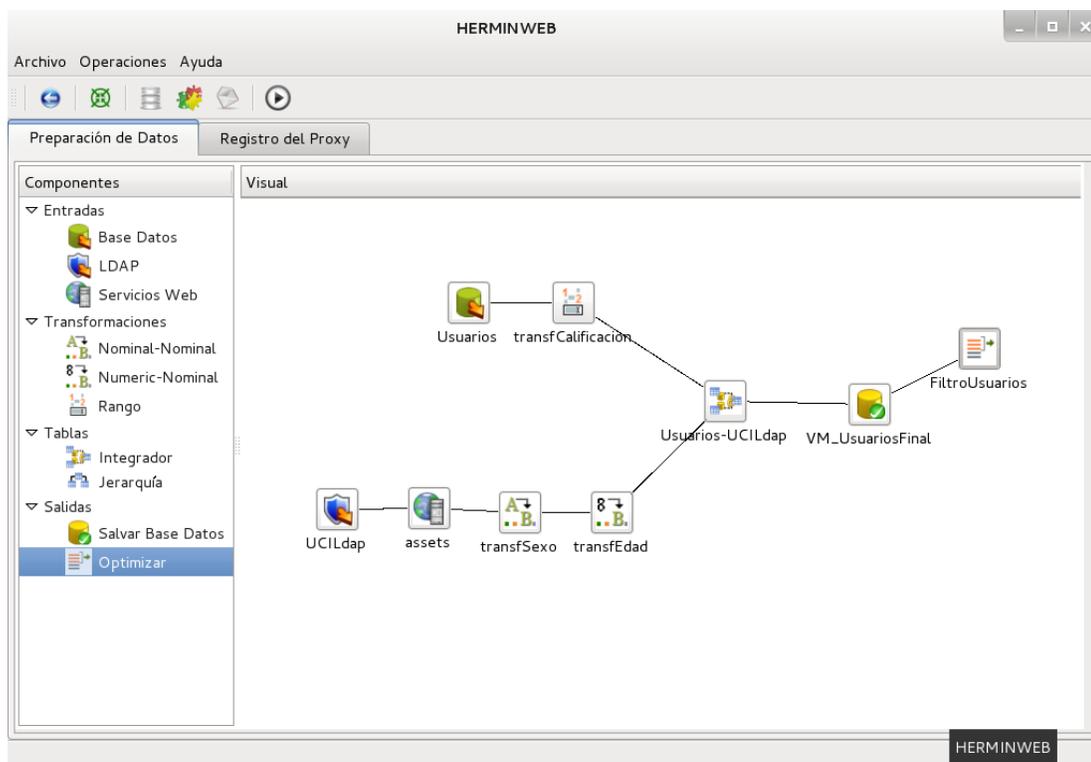


Figura 7 Configuración de una preparación de datos

¹⁷ Hace referencia al nombre del software Módulo de Preparación de Datos.

Cada uno de estos componentes muestra una interfaz que permite configurarlo como el ejemplo de la Figura 8.

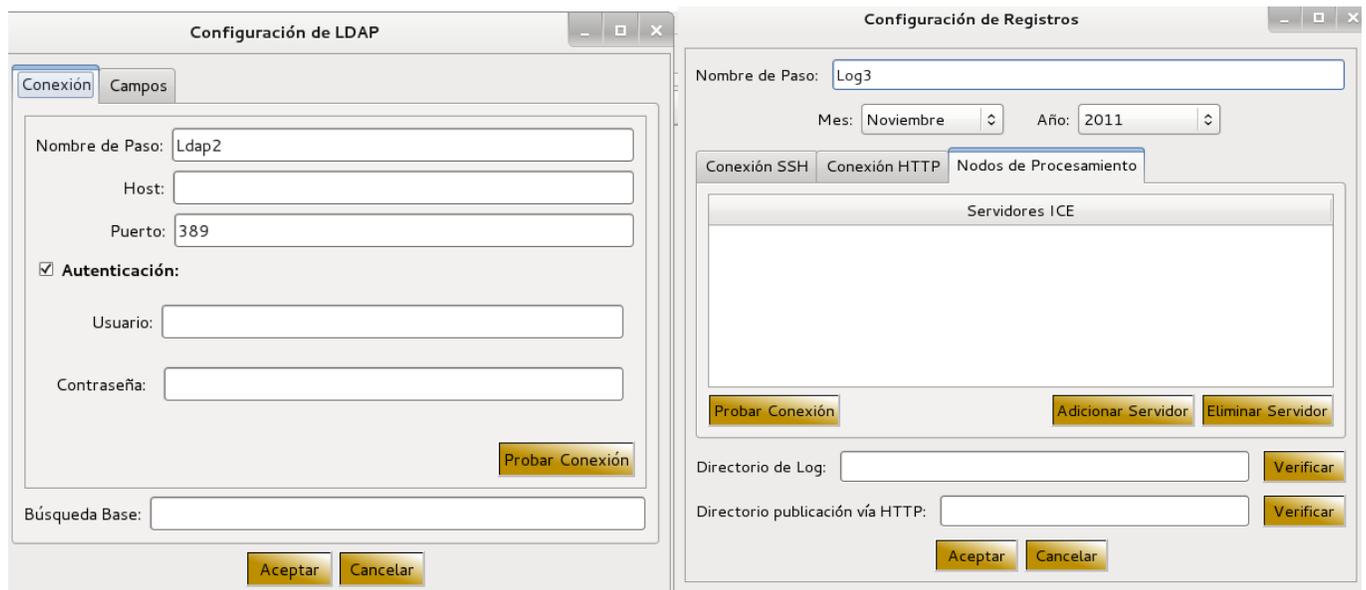


Figura 8 Interfaces de Configuración de Componentes

El sistema ofrece la posibilidad de consultar los datos de las fuentes de datos y seleccionar los que sean de interés para realizar la preparación, opción que no implementaba HERMINWEB. Para realizar las conexiones a las fuentes de información y transformar los datos, MPD genera un archivo XML que ejecuta Pentaho por líneas de comando reduciendo considerablemente el tiempo de ejecución de estas tareas. HERMINWEB solo se apoyaba en Pentaho para la conexión a las bases de datos. El administrador puede definir para las transformaciones los datos de entrada y el tipo de dato que tendrá como salida.

El procesamiento de los registros del proxy, también puede configurarse por pasos, pero no es ejecutado por Pentaho (Figura 9). El administrador selecciona los campos del log que sean de su interés para transformarlos. Además puede definir los campos por los que se agruparán las sesiones de usuario.

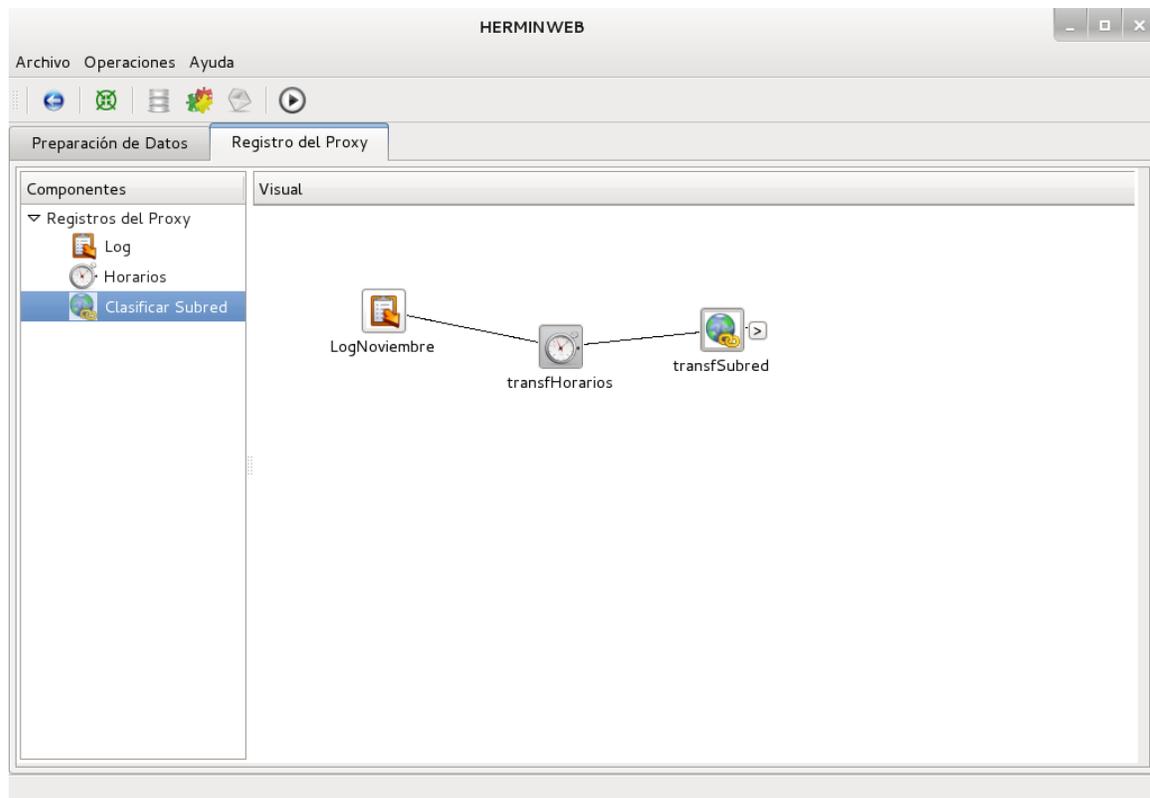


Figura 9 Configuración de los pasos para realizar el procesamiento de los registros

Debido a que este proceso consume una considerable cantidad de tiempo y recursos se creó un parser distribuido en varios clientes para realizar esta tarea en el menor tiempo posible y consumiendo la cantidad mínima de recursos. Se utiliza ICE para distribuir el procesamiento de los registros porque brinda una implementación eficiente en ancho de banda, uso de memoria y carga de CPU. Además puede ser utilizado en redes poco seguras.

Primeramente se definió la función que realiza el procesamiento de un registro y que debe ser publicada por cada nodo de procesamiento para que sea ejecutada desde el nodo central. Los módulos necesarios para esta tarea son: urllib2, cStringIO, re, time, psycpg2, os, base64, itertools. A continuación se almacenaron en una cola las tareas con los usuarios a procesar y se le enviaron a los nodos de procesamiento. Los nodos de procesamiento obtienen del servidor de registros los logs pertenecientes al usuario solicitado, y procesan las peticiones realizadas por el usuario, creando las sesiones, que son

formadas mediante el módulo itertools, agrupando las peticiones según los datos de la navegación que se hayan seleccionado para crear las vistas minables [4].

La aplicación realiza satisfactoriamente la integración de los datos de diferentes fuentes de datos, bases de datos (PostgreSQL, MySQL, Oracle), Servicios Web, LDAP y ficheros de traza. Las tablas y relaciones de la base de datos para el almacenamiento de las vistas minables se crean en tiempo de ejecución con los campos definidos por el administrador durante la preparación. Para la extracción de patrones en la fase de minería de datos no es necesario el análisis de toda la vista minable, se puede seleccionar una muestra y analizarla. No es necesario para la extracción de los patrones analizar los datos de los usuarios y los registros del proxy integrados, estos datos también se pueden analizar separados, crear la vista minable y aplicar la fase de minería. En los anexos se explica detalladamente cómo diseñar y configurar una transformación y cómo manejar el flujo de datos de un componente a otro ([Anexo 2](#)).

3.6 Diagrama de paquetes del diseño.

El diagrama de paquetes del diseño se estructuró en concordancia con las capas propuestas en la arquitectura del sistema, por lo que el color de cada paquete está relacionado con la capa en la que se encuentra. En el paquete **GUI** están contenidas todas las interfaces de usuario que serán mostradas por el sistema. En **Entornos** se almacenan las clases encargadas de realizar la abstracción entre las interfaces y la lógica del negocio. En el paquete **Pyutilib** están las clases que garantizan la arquitectura basada en componentes a utilizar, enfocándose principalmente en la descomposición del diseño en componentes funcionales que exponen interfaces de comunicación bien concretas. **Servicios** almacena las clases encargadas de realizar la lógica del sistema. El paquete **Framework** hace referencia al framework de la Plataforma de Servicios Telemáticos que brinda eventos que permiten ejecutar métodos almacenados en clases diferentes. En las **Aplicaciones** se hace referencia a las herramientas que apoyan la ejecución de las tareas del módulo. En el paquete **Dominio** están incluidas todas las entidades, que almacenan información de la configuración de cada componente y la vista minable. El **Acceso a Datos** hace referencia a las clases que se encargan de la modificación y el acceso a datos.

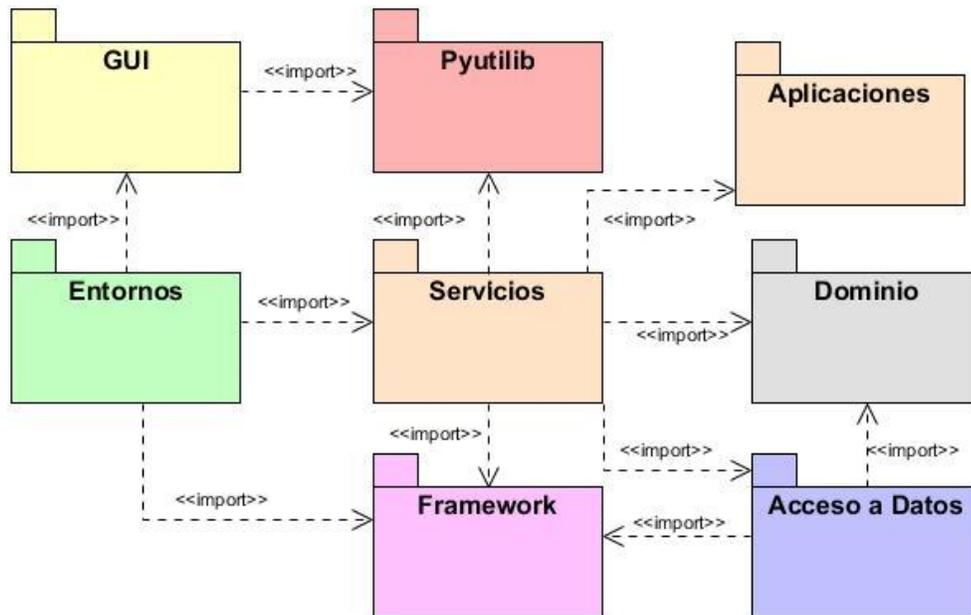


Figura 10 Diagrama de Paquetes del Diseño

3.7 Modelo Físico de Datos.

El módulo de preparación de datos para la herramienta HERMINWEB no cuenta con un modelo de datos definido, pero sí con una base de datos vacía, donde las tablas y sus relaciones serán creadas en tiempo de ejecución con aquellos atributos que el cliente defina como necesarios. En el componente Salvar Base de Datos se obtienen todos los campos que se manejan en el flujo de datos y se define el nombre para la tabla. Con esta información se ejecuta una consulta SQL, de creación de tablas, con los campos definidos, que crea la tabla en la base de datos. Por cada fuente de datos, que sea definida para obtener la información de los usuarios se crea una tabla en la base de datos, donde se van a integrar los datos de los usuarios con los datos de la navegación. Cada una de estas tablas de usuario va a heredar de una tabla principal donde se van a almacenar los datos de la navegación de los usuarios agrupados por sesiones.

3.8 Diagramas de Clases del Diseño

A continuación se muestran los diagramas de clases del diseño para los casos de uso arquitectónicamente significativos.

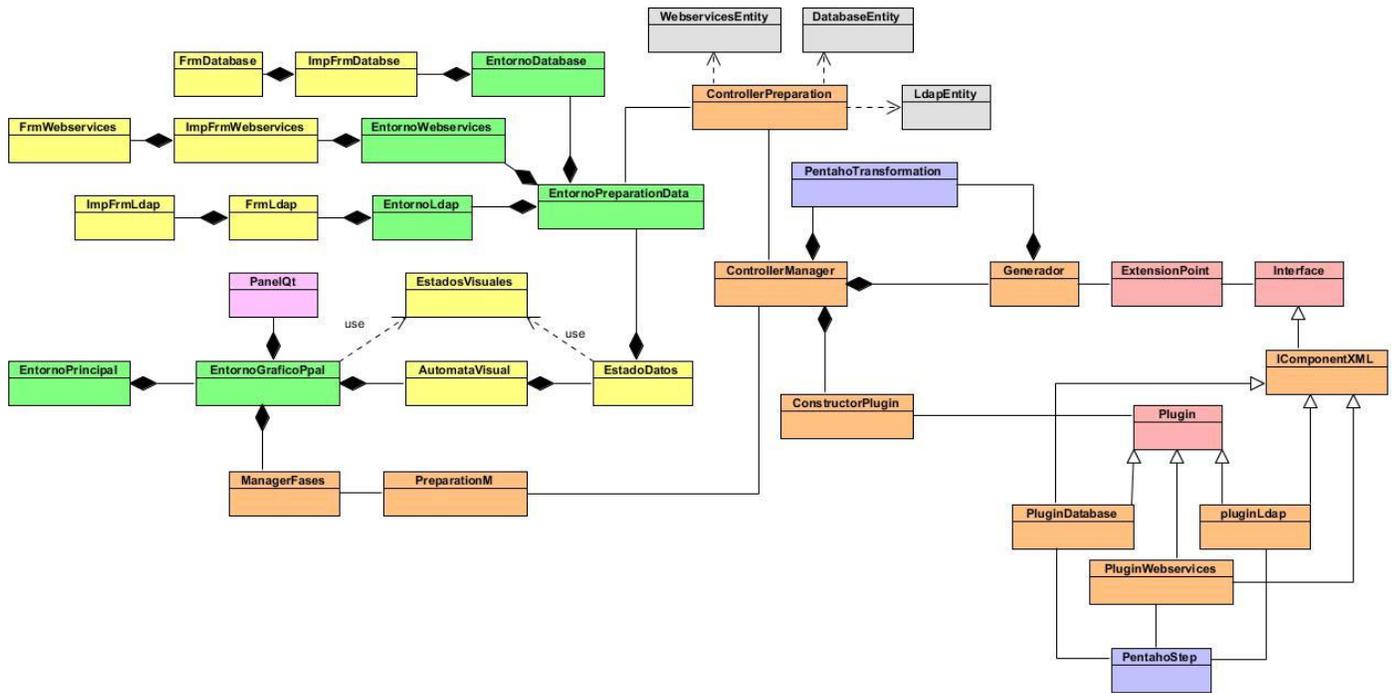


Figura 11 Diagrama de Clases Diseño del Caso de Uso Procesar Fuentes de Datos

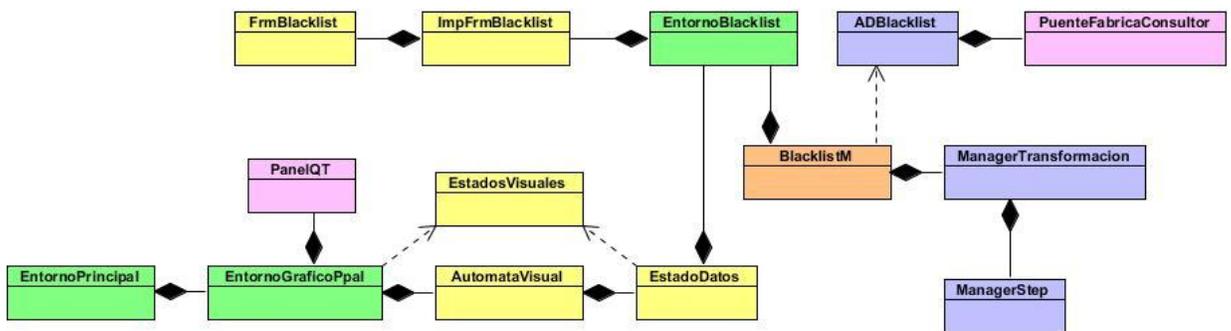


Figura 12 Diagrama de Clases del Diseño del Caso de Uso Procesar Blacklist

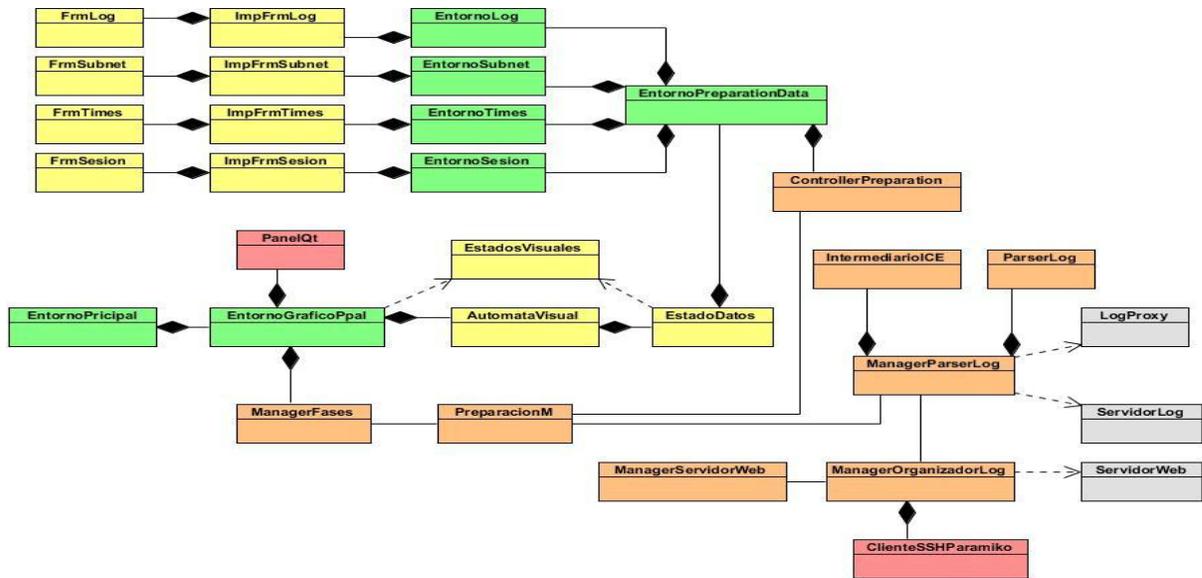


Figura 13 Diagrama de Clases del Diseño del Caso de Uso Procesar Registros del proxy

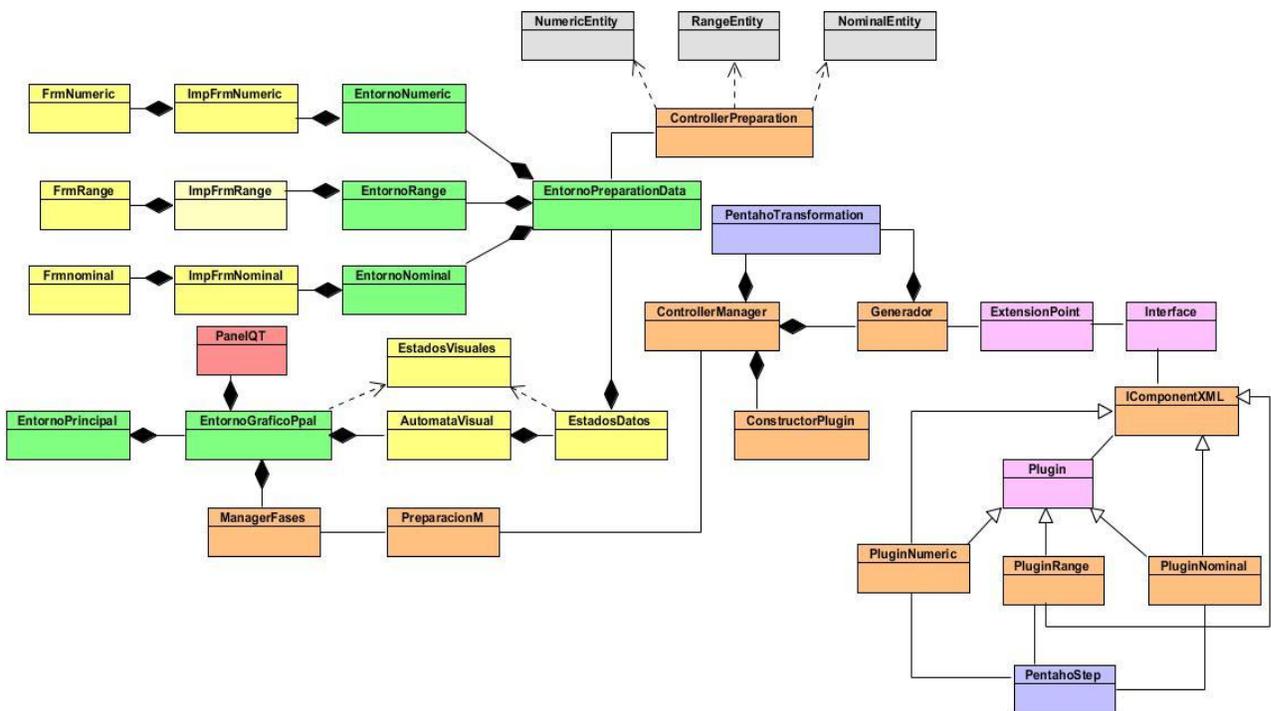


Figura 14 Diagrama de Clases del Diseño del Caso de Uso Transformar Datos

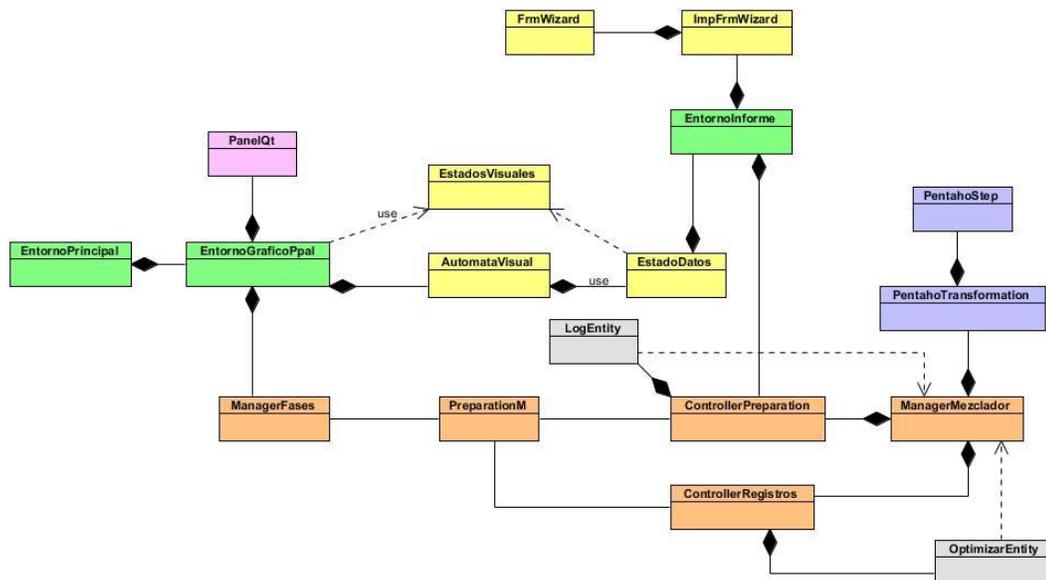


Figura 15 Diagrama de Clases del Diseño del Caso de Uso Mezclar Datos.

Conclusiones del capítulo.

En el presente capítulo se explicó de manera detallada como será implementado el sistema. Se realizó una valoración de los patrones de diseño que permitirán implementar con mayor robustez la arquitectura N-Capas definida, que apoyada en el Framework Pyutilib garantiza la reutilización, flexibilidad y extensibilidad del software. Fueron solucionadas las deficiencias encontradas en la iteración anterior de HERMINWEB, con lo que se obtuvo un software adaptable a cualquier diseño de información y con un procesamiento dinámico de los datos. También se realizó un estudio de la necesidad de gestionar la seguridad de los datos que maneja la aplicación y cómo aplicarla. Los diagramas de paquetes del diseño, y diagramas de clases del diseño modelados ofrecen una clara visión de la distribución y funcionalidad de las clases dentro de cada capa arquitectónica.

Capítulo 4: Implementación y Pruebas

4.1 Introducción

Es en la implementación del sistema donde se obtienen los componentes y subsistemas en un modelo de implementación, todo esto a partir del modelo de clases del diseño definido en el capítulo anterior. Describe los artefactos de la implementación además de explicar el modelo de despliegue del sistema que se aplicará una vez terminada en su totalidad la implementación del mismo [18].

4.3 Diagrama de Despliegue

En el diagrama de despliegue se muestra la situación física de los componentes lógicos desarrollados; en otras palabras se sitúa al software en el hardware que lo contiene [18].

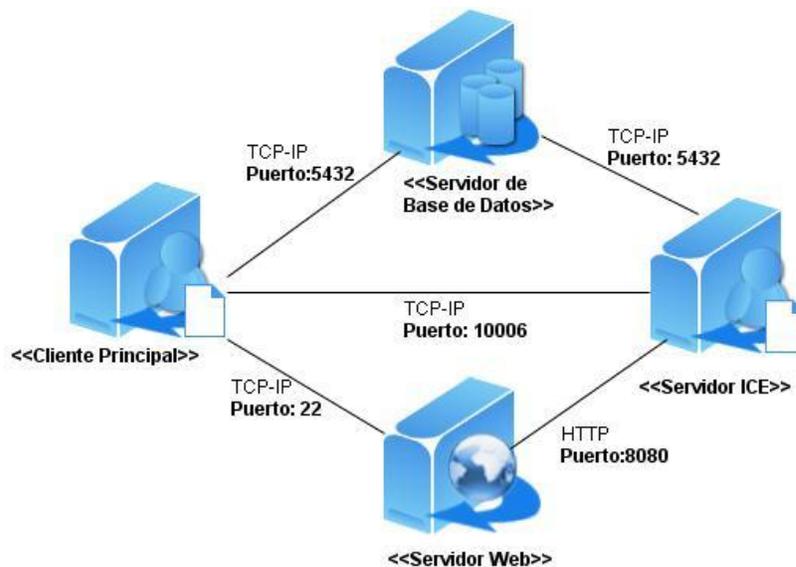


Figura 16 Diagrama de Despliegue

Nombre del nodo	Descripción
Cliente Principal	Computadora desde la que será ejecutada la herramienta.
Base de Datos	Servidor de Base de Datos donde se alojará toda la información recopilada y creada por la herramienta.
Servidor ICE	Nodo de procesamiento que participará en el <i>procesamiento</i> de los registros del <i>proxy</i> . Se utilizarán tantos nodos como se tengan disponibles.
Servidor Web	Servidor web donde serán compartidos los registros del <i>proxy</i> .

Tabla 3 Descripción de los nodos del diagrama de despliegue

4.2 Diagramas de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones, representan a su vez todos los tipos de elementos software que intervienen en la fabricación de aplicaciones informáticas, como ejecutables, librerías y dependencias lógicas que existen entre ellos. A continuación se realiza una breve descripción de los paquetes de componentes que integran el sistema:

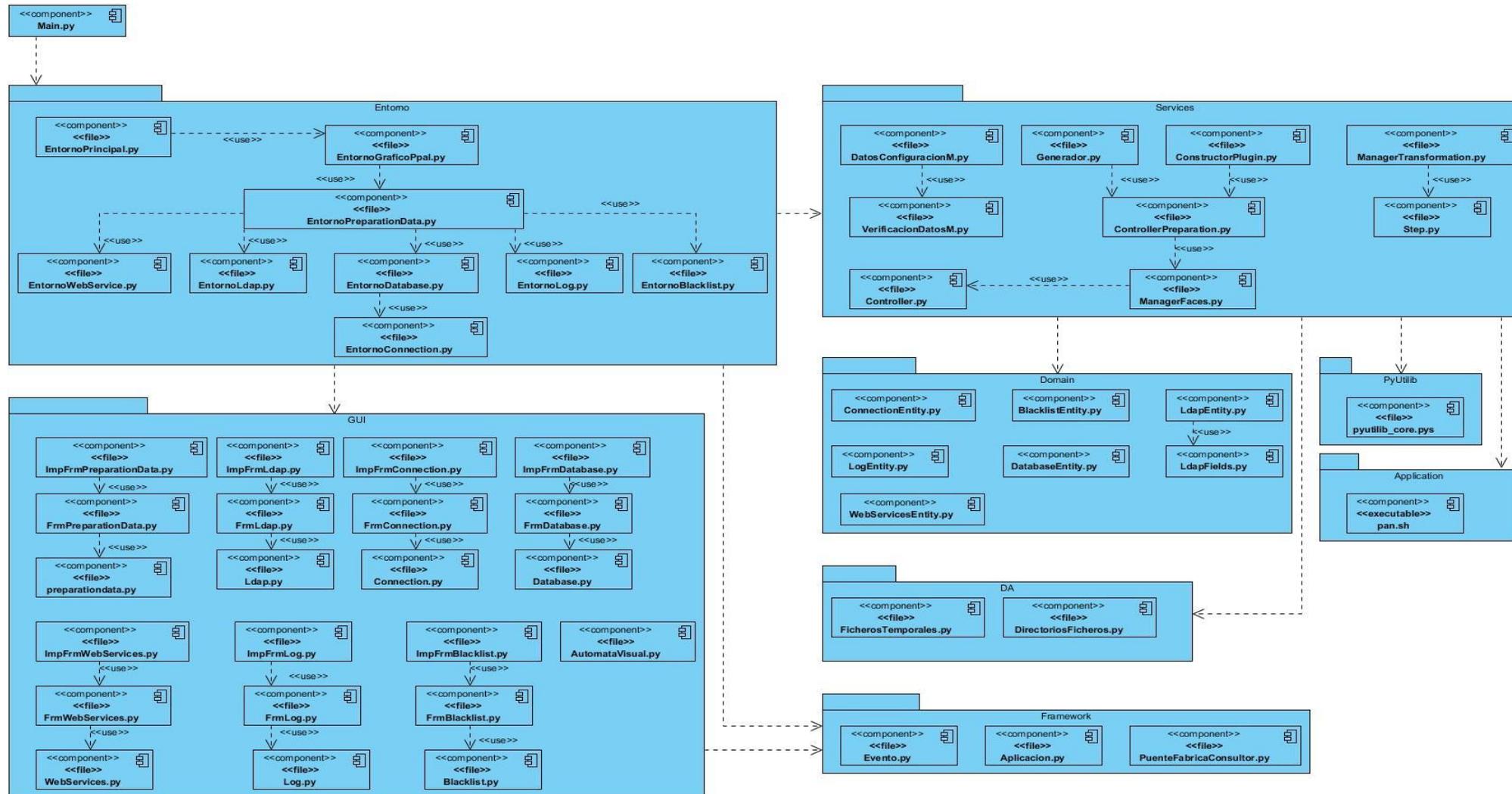


Figura 17 Diagrama de Componentes de los CU que intervienen en el Procesamiento de las fuentes de Información

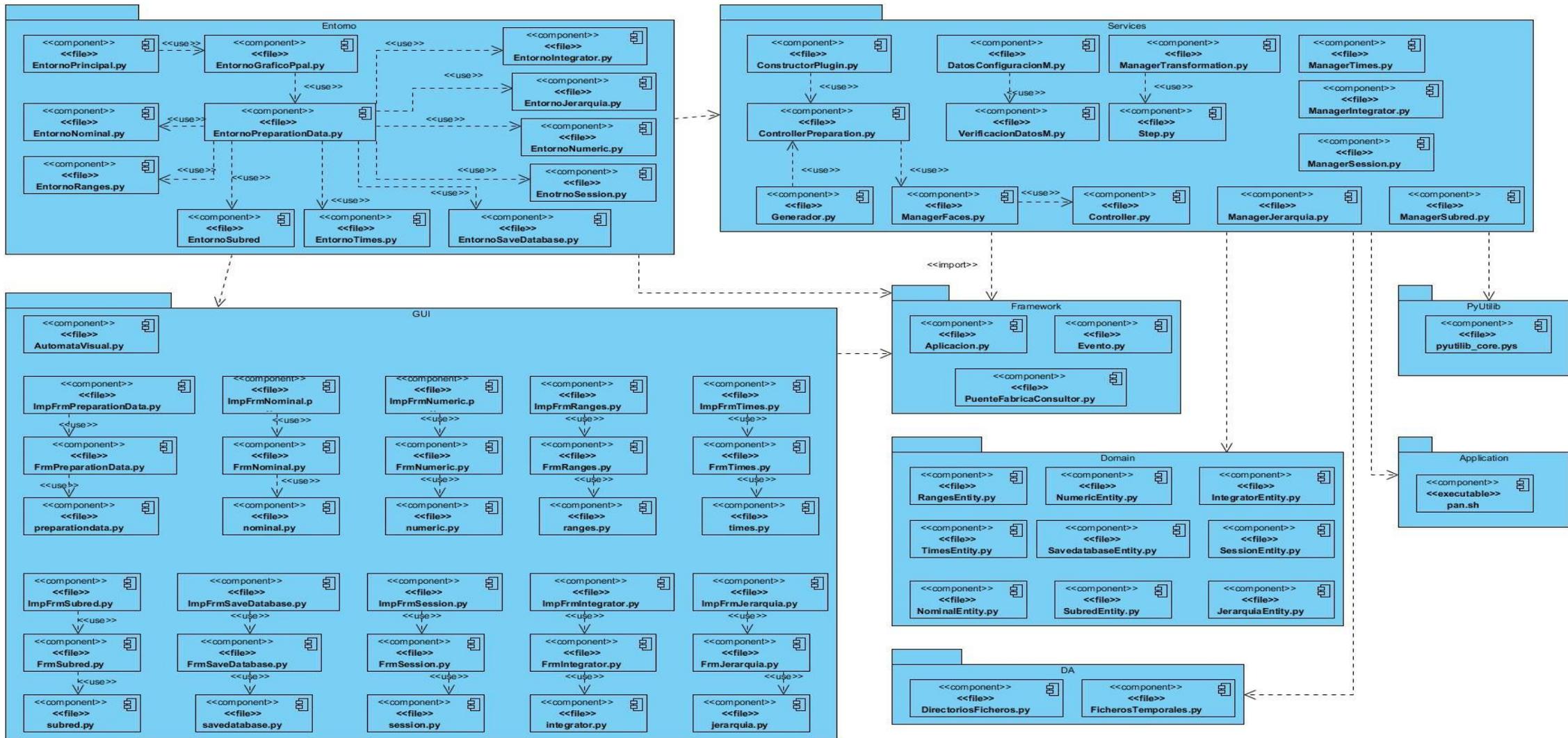


Figura 18 Diagrama de Componentes de los CU que intervienen en la transformación y salva de los datos

4.4 Estrategias de prueba.

Las pruebas de software son actividades en las cuales un sistema o uno de sus componentes se ejecutan en circunstancias especificadas. Los resultados se observan y registran y se realiza una evaluación de algún aspecto [18]. Con el creciente desarrollo del software y los costos asociados a un fallo del mismo han obligado a los desarrolladores a buscar estrategias de prueba cada vez mejor pensadas. Las estrategias proporcionan un mapa que integrando los métodos de diseño de casos de prueba desemboca en una eficaz construcción del software [27].

Se definió una estrategia en el nivel de pruebas Internas y dentro de ésta los niveles de Pruebas de Integración y Sistema. Las Pruebas de Integración son la fase de las pruebas de software en la cual módulos individuales de software son combinados y probados como un grupo. De este nivel de pruebas se realizó el tipo de prueba Funcionalidad Función centrando la atención en la validación de las funciones, métodos, servicios y CU [27].

Las Pruebas de Sistema permiten probar el sistema en su conjunto y con otros sistemas con los que se relaciona para verificar que las especificaciones funcionales y técnicas se cumplen. Los tipos de pruebas aplicados en este nivel se explican a continuación [27]:

- Funcionalidad

Seguridad: Realizadas para asegurar que los datos o el sistema solamente son accedidos por los actores deseados.

Volumen: Se realizaron con el objetivo de verificar las habilidades de los programas para manejar grandes cantidades de datos, tanto como entrada, salida o residente en la BD.

- Rendimiento

Carga: Se realizó generalmente para observar el comportamiento de la aplicación bajo una cantidad de peticiones esperada. Esta carga puede ser el número esperado de usuarios registrados en las fuentes de información o las peticiones de todos los archivos Logs publicados en el servidor. Esta prueba puede mostrar los tiempos de respuesta de todas las transacciones importantes de la

aplicación. Si la base de datos, el servidor de aplicaciones, y otros componentes también se monitorizan, entonces esta prueba puede mostrar cual es el cuello de botella en la aplicación.

El método utilizado en ambos niveles fue el de Caja Negra o Prueba de Comportamiento centradas en las funcionalidades del Software, permitiendo obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. Estas pruebas estuvieron encaminadas a encontrar errores de las siguientes categorías [27]:

- Funciones incorrectas o ausentes.
- Errores de Interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externas.
- Errores de rendimiento
- Errores de inicialización y rendimiento.

Para este método se utilizó la técnica de Partición de Equivalencia que se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así la cantidad de casos de prueba que hay que desarrollar. A continuación se muestra el diseño del caso de prueba para el CU Configurar Componentes de Entrada para el escenario “Configurar Componentes de Entrada Base de Datos”.

Caso de Prueba: Configurar Componente de Entrada Base de Datos para una base de datos PostgreSQL: Se muestran las interfaces para configurar los parámetros que establecen la conexión a la base de datos, establece la conexión y permite seleccionar los campos que serán utilizados en la preparación de los datos. El sistema almacena la configuración.

Escenario	Descripción	Nombre de Paso	Host	Puerto	Usuario	Contraseña	Base de Datos	Respuesta del sistema	Flujo central
EC 1.1: Configurar Componentes de Entrada Base de Datos para una base de datos PostgreSQL	Configura los componentes de entrada para la conexión y obtención de los datos de las fuentes de datos Base de Datos.	V conexion2	V localhost	V 8080	V yprdguez	V Yenifer	V Docente	Muestra una interfaz con los parámetros de configuración. Verifica que los datos introducidos sean correctos. Verifica la conexión a la base de datos. Permite seleccionar los campos a procesar.	Da doble clic encima del componente para configurarlo. Introduce los datos del formulario. Selecciona la opción "Probar Conexión". Selecciona los campos a obtener. Selecciona la opción "Aceptar".
EC 1.2: Verificar que los datos introducidos sean correctos	Muestra la interfaz con las opciones de configuración del componente y un mensaje de error.	I L@hdgf	V localhost	V 8080	V yprdguez	V Yenifer	V Docente	Verifica que los datos sean correctos. El sistema muestra un mensaje de error.	Introduce nuevamente los datos en el formulario.
		V Ldap2	I Localhost\$	V 8080	V yprdguez	V Yenifer	V Docente		
		V Ldap2	V localhost	I 80a8	V yprdguez	V Yenifer	V Docente		
		V conexion2	V localhost	V 8080	I ypr^guez	V Yenifer	V Docente	Muestra la interfaz con las opciones de configuración del componente.	
		V conexion2	V localhost	V 8080	V yprdguez	I aa	V Docente		
		V conexion2	V localhost	V 8080	V yprdguez	V Yenifer	I 125ADE		

EC 1.3: Error en la conexión al servidor.	Muestra la interfaz con las opciones para gestionar la configuración y un mensaje de error.							Muestra la interfaz con las opciones para gestionar la configuración.	Introduce nuevamente las variables de configuración.
---	---	--	--	--	--	--	--	---	--

Tabla 4 Caso de Prueba: Configurar Componente de Entrada Base de Datos para una base de datos PostgreSQL

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Nombre de Paso	Campo Texto	No	Cadena de caracteres
2	Host	Campo Texto ó Dirección de red	No	Cadena de caracteres
3	Puerto	Numérico	No	Número
4	Usuario	Campo Texto	No	Cadena de letras y números
5	Contraseña	Campo de Texto	No	Cadena de caracteres
6	Base de datos	Campo de Texto	No	Cadena de letras y números

Tabla 5 Descripción de las variables

Entorno de pruebas

Se definió, en correspondencia con los requisitos no funcionales un entorno de pruebas para validar el funcionamiento de la aplicación. Las características se exponen a continuación:

Hardware	Datos
Procesador	Dual Core CPU 2.1 GHz
Memoria	1GB
Tarjeta Madre	ASUS
Ancho de Banda	100 Mbs
Sistema Operativo	Debian 6.0
Cantidad de Servidores ICE	1

Tabla 6 Características del Entorno de Pruebas

Análisis de Resultados

Después de realizadas las pruebas se arrojaron los siguientes resultados. Fueron obtenidos un total de 12 no conformidades clasificadas de la siguiente forma:

Total	Alta	Media	Baja
12	3	4	5

Tabla 7 Resultados de las pruebas obtenidas

Entre las principales no conformidades encontradas se pueden mencionar:

- Tamaño de fuente muy pequeño en algunas interfaces.
- Faltas de ortografía (tildes) en las interfaces de configuración de los componentes.
- Problemas en el orden lógico de los pasos de la preparación.
- Mala validación de la configuración de los componentes.

Las no conformidades encontradas fueron resueltas satisfactoriamente.

4.5 Reporte de los resultados del proceso

A continuación se realizará la medición de las variables que sustentan la hipótesis (usabilidad, dependencia) de la investigación para validar el cumplimiento de los objetivos propuestos.

Variable: Dependencia

Dimensión	HERMINWEB	MPD
Indicadores		
Cantidad de componentes modificados ante una nueva fuente	5	0
Cantidad de componentes modificados ante un nuevo campo	6	0
Cantidad de componentes modificados para eliminar una fuente	5	0
Cantidad de componentes modificados para eliminar un campo	6	0
Cantidad de funciones en base de datos modificados ante una nueva fuente	2	0
Cantidad de funciones en base de datos modificados al eliminar una fuente	2	0
Cantidad de funciones en base de datos modificados ante un nuevo campo	0	0
Cantidad de funciones en base de datos modificados al eliminar un campo	0	0
Cantidad de tablas modificadas al eliminar un campo	3	0
Cantidad de tablas modificadas al eliminar una fuente	3	0
Cantidad de clases modificadas ante una nueva fuente	3	3

Cantidad de clases modificadas ante un nuevo campo	3	3
Total	38	6

Tabla 8 Medición de la variable Dependencia

Variable: Usabilidad

Dimensión	HERMINWEB	MPD
Indicadores		
Cantidad de fuentes aceptadas	3	3
Cantidad de campos aceptados	32	N campos
Cantidad de atributos aceptados en base de datos	45	N atributos
Cantidad de tipos de usuarios aceptados	3	N usuarios.
Total	83	3+N

Tabla 9 Medición de la variable Usabilidad

La hipótesis de esta investigación plantea que con la implementación del MPD, HERMINWEB podrá ser usado en otros diseños de información con la menor cantidad de dependencias, entonces para sustentar la hipótesis el MPD debe cumplir la proporción que a mayor usabilidad menor dependencia. Analizando los resultados de la tabla 10:

	HERMINWEB	MPD
Variables		
Usabilidad	83	3 + N
Dependencia	38	6

Tabla 10 Resultados de la medición de las variables

MPD puede ser usado en diferentes entornos, con las mismas fuentes de datos y soportando N campos, necesitando modificaciones solo cuando sea necesario agregar una fuente de información que no sea

soportada por la herramienta. De las fuentes de datos soportadas por la herramienta se pueden utilizar en la preparación de los datos tantas como sean necesarias y seleccionar de cada una los datos de interés sin que el software sufra modificaciones. Las funciones para el manejo de los datos de la base de datos al agregar o eliminar un nuevo campo o atributo no se modifican. El diseño de la base de datos es capaz de adaptarse a nuevos campos y relaciones entre tablas introducidos para el análisis. HERMINWEB solo puede ser usado en el entorno de información de la UCI y soportando las fuentes de información definidas para dicha institución. Los datos que utiliza para el análisis son específicos de cada fuente, lo que provoca que al agregarle una nueva fuente de información o nuevos campos sea necesario realizarle considerables modificaciones a las clases que intervienen en el proceso de preparación de los datos, así como a las funciones que administran los datos en la base de datos. Después de realizar este análisis queda demostrada la hipótesis y el cumplimiento de todos los objetivos de la investigación.

Conclusiones del capítulo

En este capítulo se brinda una explicación detallada de la implementación del sistema a través de los diagramas de componentes, que muestra las relaciones entre las clases del diseño y los paquetes que las contienen. El diagrama de despliegue se encarga de explicar cómo el software quedará contenido dentro del hardware, así como los protocolos y los puertos de comunicación entre los servidores. Se pudo definir una estrategia de pruebas en niveles, tipos de pruebas y técnicas que permitieron comprobar que las funcionalidades del software devolvieran los resultados adecuados. De estas pruebas se obtuvieron también las no conformidades, que radicarón mayormente en faltas de ortografía en las interfaces de configuración de los componentes, en la mala validación de los campos de configuración, y la pérdida del camino en la secuencia de pasos que definen la preparación. Estas no conformidades fueron resueltas satisfactoriamente. Se validó la hipótesis de la investigación resolviendo el problema de investigación y a su vez cumpliendo con los objetivos trazados.

CONCLUSIONES GENERALES

En este trabajo se realizó un estudio de las arquitecturas modulares, centrandó la atención en los sistemas basados en componentes con el objetivo de implementar un módulo de preparación de datos para que HERMINWEB pueda ser aplicado en entornos con diferentes diseños de información que utilice como fuentes de datos Bases de Datos, Servicio Web y LDAP.

Se realizó un análisis de las deficiencias encontradas en la preparación de los datos implementada en la versión anterior y que fueron solucionadas en la medida que avanzaba la investigación. En aras de adaptar HERMINWEB a cualquier entorno se implementó el módulo de preparación de datos, como una herramienta basada en componentes, donde cada componente implementa una funcionalidad sobre los datos permitiendo al interesado elegir sus fuentes de datos, seleccionar de ellas los datos que le resulten de interés y definir las transformaciones que permitirán hacer el análisis más sencillo. El cliente puede además definir el criterio por el que se van a integrar los datos de los usuarios con los registros de navegación del proxy y almacenarlos en una base de datos que creará sus tablas y relaciones en tiempo de ejecución con los datos que se definan. Para mejorar el tiempo de respuesta de la aplicación en cuanto a la obtención de los datos y las transformaciones realizadas a los mismos se integró el módulo de datos a la ETL Kettle de la Suite Pentaho BI.

La utilización de la programación distribuida demostró por los resultados obtenidos en las pruebas realizadas al módulo de procesamiento de los logs, que es una excelente opción cuando se tiene que procesar grandes volúmenes de información en computadoras con no muy altas prestaciones, pues redujo considerablemente el tiempo de procesamiento de los registros de navegación. La herramienta logró integrar satisfactoriamente varias fuentes de información con formatos diferentes como: ficheros de logs, base de datos relacionales (SQLServer, Oracle y PostgreSQL), LDAP y servicios web, siendo de gran importancia en este sentido la utilización de la ETL Spoon de la suite de aplicaciones Pentaho, que permitió acelerar el proceso de preparación de los datos y liberar de responsabilidades a la herramienta.

El diseño de la arquitectura N- capas que implementa la aplicación facilita la reutilización por el desacople de las funcionalidades que implica, pudiéndose utilizar en otras investigaciones y la utilización de PCA

garantiza, gracias a su arquitectura basada en componentes que el software pueda ser extendido en una manera dinámica.

HERMINWEB es una herramienta que automatiza un proceso KDD, aplicado a los registros de navegación por Internet almacenados por el servidor proxy, utilizando tecnologías libres, apoyado en la Minería de Uso de la Web, donde se obtienen modelos en términos de las tareas Agrupamiento y Reglas de Asociación y que puede ser instalado y configurado en cualquier ambiente de trabajo que posea su información almacenada en Bases de Datos, Servicios Web o LDAP.

RECOMENDACIONES

Diseñar e implementar un Datawarehouse para el almacenamiento de los datos durante la fase de preparación de datos, para lograr una mayor organización en los datos y agilizar de esta forma la creación de la vista minable.

Investigar técnicas de reducción de dimensiones que aplicadas sobre los datos haga más sencillo el análisis de la información.

Implementar un módulo para la gestión de las clasificaciones de dominio para mejorar este proceso.

Implementar un módulo dinámico para la fase de minería de datos que se pueda adaptar al MPD.

REFERENCIAS BIBLIOGRÁFICAS

1. Jonás A. Montilva C., N.A.y.J.A.C., *Desarrollo de Software Basado en Componentes*. 2003.
2. Española, R.A.d.I.L., *Gran Diccionario de la Lengua Española.*, ed. P.d.F. Rico. Larousse.
3. Ibañez, J.C.P. (2004) *Arquitectura de Información: Introducción al Proceso de Desarrollo en el Diseño de Interfaces de Usuario*.
4. Julio Antonio Hernández Pérez, H.D.R., *Herramienta Informática de Minería de Uso de la Web sobre los registros de navegación por Internet. Implementación del módulo para realizar las tarea descriptiva Reglas de Asociación*. 2011, Universidad de Ciencias Informáticas: La Habana.
5. Rolando Alfredo Hernández León, S.C.G., *El Proceso de Investigación Científica*. 2011, La Habana: Editorial Universitaria del Ministerio de Educación Superior.
6. Fayyad U., P.-S.G., Smyth P.y Uthurusamy P., *Advances in Knowledge discovery and data mining*. 1996: AAAI/MIT Press.
7. Aguilar Ruiz, J.S., *Curso preprocesado de datos y aprendizaje basado en reglas*. 2008, Santa Clara.
8. Gorunuesco, F., *Data Mining. Concepts, Model and Techniques*. 2006.
9. Jeria, V.H.E., *Minería Web de Uso y Perfiles de Usuario: Aplicaciones con Lógica Difusa*. . 2007, Granada: Granada.
10. *Microsoft Diccionario de Informática e Internet*.
11. Amy Brown, G.W., *The Architecture of Open Source Applications* 2011.
12. William E. Hart, J.S., *The PyUtilib Component Architecture*, S.N. Laboratories, Editor. 2006: Albuquerque.
13. Roland Bouman, J.v.D., *Pentaho Solutions: Business Intelligence and Data warehousing with Pentaho and MySQL*. 2009, Indianapolis.
14. Duque, R.G., *Python para todos*, España.
15. Andrés Marzal Varó, I.G., *Introducción a la Programación con Python*. Internet ed. 2003: Universitat Jaume I.
16. Garrido, J.M.M., *Plataforma Eclipse. Introducción Técnica*.
17. Rojas, A.D.O., *Qt 4 Manual Introductorio*. 2008.
18. Ivar Jacobson, G.B., James Rumbaugh *El proceso unificado de desarrollo del Software* ed. A.-. Wesley. 2000.
19. Paradigm, V., *Visual Paradigm Internacional*. 2010.
20. Eduardo Alvarez Romero, D.P., *Integration Definition For Funcion Modeling (IDEF0)* 2005: Sistemas Integrados de Fabricacion.
21. Drake, J.M., *Programación concurrente y distribuida: ICE Overview*. 2007.
22. Fernández, D.V., *Documentación de ZeroC ICE*, U.d.C.-L. Mancha, Editor. 2006: Castilla.
23. Yoanni Ordoñez Leyva, E.A.V., *Herramienta Informática de Minería de Uso de la Web sobre los Registros de Navegación por Internet*. 2010, Universidad de las Ciencias Informáticas: Ciudad de La Habana.
24. Larman, C., *UML y Patrones. Introducción al análisis y diseño orientado a objetos*. 1999.

25. Erich Gamma, R.H., Ralph Johnson, Jhon Vlissides, *Design Patterns: Elements of Reuseable Object Oriented Software*.
26. Carlos Reynoso, N.K., *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft*. 2004.
27. Pressman, R.S., *Ingeniería de Software. Un enfoque práctico*. Quinta Edición ed. 2000: Mc Graw Hill.

Glosario

APIs	Es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.
biblioteca	En clara referencia a las bibliotecas habituales, este término, cuando se utiliza en el mundo informático, se refiere a una colección o conjunto de programas desarrollados por un mismo fabricante que suelen ser compatibles e interoperables entre sí.
Blacklist	En Internet, una lista negra o <i>blacklist</i> es una lista donde se registran las direcciones <i>IPs</i> que generan spam de forma voluntaria o involuntaria.
C	Lenguaje orientado a la implementación de sistemas operativos.
C++	Extensión del lenguaje C que permite la manipulación de objetos.
ciclo de vida	Tiempo estimado que se espera esté un soporte, medio o aplicación en uso operativo. También se utiliza para designar las principales etapas que tienen lugar durante el desarrollo de un sistema.
CVS	Es una aplicación informática que implementa un sistema de control de versiones: mantiene el registro de todo el trabajo y los cambios en los ficheros (código fuente principalmente) que forman un proyecto (de programa).
Datawarehouse	Base de datos que almacena una gran cantidad de datos transaccionales integrados para ser usados en análisis gestionales por usuarios especializados.
Eclipse	Entorno de desarrollo integrado inicialmente con soporte para <i>Java</i> , luego extendido con soporte para otros lenguajes mediante el uso de <i>plugins</i> .
Framework	Estructura de artefactos o módulos concretos con base en la que otro proyecto de software puede ser desarrollado.
GNU/Linux	Sistema operativo creado con la combinación de las herramientas de sistema <i>GNU</i> y el <i>kernel</i> o núcleo similar a <i>Unix</i> , llamado <i>Linux</i>
html	Lenguaje de programación que se utiliza para el desarrollo de páginas de

	Internet. Se trata de la sigla de <i>Hyper Text Markup Language</i> , es decir, Lenguaje de Marcas de Hipertexto. Permite describir la estructura y el contenido en forma de texto, además de complementar el texto con objetos tales como imágenes. Este lenguaje se escribe mediante etiquetas, que aparecen especificadas por corchetes angulares (< ejemplo >).
Inteligencia de Negocio	Se denomina inteligencia empresarial, inteligencia de negocios o BI (del inglés Business Intelligence) al conjunto de estrategias y herramientas enfocadas a la administración y creación de conocimiento mediante el análisis de datos existentes en una organización o empresa.
IP	Es una etiqueta numérica que identifica, de manera lógica y jerárquica, a un interfaz (elemento de comunicación/conexión) de un dispositivo (habitualmente una computadora) dentro de una red que utilice el protocolo IP (<i>Internet Protocol</i>), que corresponde al nivel de red del protocolo TCP/IP.
Java	Un lenguaje de programación de alto nivel, orientado a objetos.
LDAP	Protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red. Es también considerado como una base de datos a la que se le pueden realizar consultas.
Lenguaje de alto nivel	Lenguaje donde es posible expresar los algoritmos mediante una sintaxis del modo más cercano a la capacidad cognitiva del hombre.
Lenguaje orientados a objetos	Lenguaje diseñado para cumplir con los conceptos de la programación orientada a objetos.
Logs	Un log es un registro oficial de eventos durante un rango de tiempo en particular. Usado para registrar datos o información sobre quién, qué, cuándo, dónde y por qué un evento ocurre para un dispositivo en particular o aplicación.
MB	El Megabyte es una unidad de medida de cantidad de datos informáticos. Es un múltiplo del byte u octeto, que equivale a 10^6 bytes.
MVCC	Sistema utilizado por PostgreSQL que permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.

Oracle	Sistema de gestión de base de datos relacional (o RDBMS por el acrónimo en inglés de <i>Relational Data Base Management System</i>), desarrollado por <i>Oracle Corporation</i> .
Plugins	Es un módulo de <i>hardware</i> o <i>software</i> que añade una característica o un servicio específico a un sistema más grande.
Proxy	Programa o dispositivo que realiza una acción en representación de otro. Su finalidad más habitual es la de servidor proxy, que consiste en interceptar las conexiones de red que un cliente hace a un servidor de destino, por varios motivos posibles como seguridad, rendimiento, anonimato, entre otros.
RAM	Es un tipo de memoria de ordenador a la que se puede acceder aleatoriamente; es decir, se puede acceder a cualquier byte de memoria sin acceder a los bytes precedentes. La memoria RAM es el tipo de memoria más común en ordenadores y otros dispositivos como impresoras.
Servicio Web	Una forma estándar de integrar aplicaciones basadas en Web utilizando los estándares abiertos: XML, SOAP, WSDL y UDDI. XML se utiliza para etiquetar los datos, SOAP para transferir los datos, WSDL para describir los servicios disponibles y UDDI para listar qué servicios están disponibles.
sesiones de usuario	Es un perfil que se crea con el objetivo de agrupar el comportamiento de los usuarios al navegar por Internet tomando factores como: el horario de acceso, la dirección <i>IP</i> , el sitio accedido, entre otros.
SSH	Es un protocolo que sirve para acceder a máquinas remotas de forma segura a través de una red. Permite manejar por completo la computadora mediante un intérprete de comandos.
TCP	Del inglés <i>Transmission Control Protocol</i> . Es un protocolo de comunicación del nivel de transporte, orientado a conexión.
URL	Es una secuencia de caracteres, de acuerdo a un formato modélico y estándar, que se usa para nombrar recursos en Internet para su localización o identificación, como por ejemplo documentos textuales, imágenes, vídeos, presentaciones digitales, entre otros.
XML	Sencillo formato de texto, diseñado especialmente para documentos web.

	Permite la creación de etiquetas propias.
--	---