Universidad de las Ciencias Informáticas

Facultad 3

Título: Desarrollo de un sistema para la gestión del proceso de transferencia de archivos del sistema SAREN.

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

Autor:

Marcel Curbelo Carmona

Tutor(s):

Yosvany Márquez Ruíz

Rudel Cárdenas Díaz

Ciudad de la Habana, Junio del 2009

"Todo hombre está obligado a honrar con su conducta privada, tanto como con la pública, a su patria".

José Martí.

DECLARACIÓN DE AUTORÍA

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

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

Marcel Curbelo Carmona

_____

Firma del Autor

Ing. Yosvany Márquez Ruiz

_____

Firma del Tutor

Ing. Rudel Cardenas Díaz

_____

Firma del Tutor

DATOS DE CONTACTO

Autor: Marcel Curbelo Carmona

Correo Electrónico: mcurbelo@estudiantes.uci.cu

Tutor: Yosvany Márquez Ruiz

Correo Electrónico: ymarquezr@uci.cu

Tutor: Rudel Cardenas Díaz

Correo Electrónico: rudel@uci.cu

AGRADECIMIENTOS

Agradezco a mis padres los cuales me han brindado todo su apoyo y amor, sin ellos no hubiera llegado a convertirme en la persona que soy hoy.

A mi Tri por estar siempre a mi lado y brindarme su amor y compresión, por ser mi novia y mi todo.

A mis tutores Yosvany y Rudel por su ayuda y preocupación.

A la Revolución Cubana por brindarme la posibilidad de estudiar en esta maravillosa escuela.

A todos mis compañeros y amigos con los cuales he pasado los momentos más maravillosos de mi vida.

Resumen

En los registros y notarías de la República Bolivariana de Venezuela se está llevando a cabo un proceso de modernización y digitalización de su sistema registral y notarial mediante la implantación del sistema SAREN el cual tiene como objetivo otorgar seguridad jurídica y garantizar los principios de libertad contractual y de legalidad de los derechos de las personas, actos, contratos y negocios jurídicos.

El presente trabajo muestra el desarrollo de un sistema para el proceso de transferencia de archivos del sistema SAREN con el cual se pretende mejorar y beneficiar a dicho proceso, el cual hasta el momento no cuenta con un sistema automatizado para realizar sus principales funciones.

Se expone la arquitectura del sistema, así como, las funcionalidades propuestas para brindar la solución informática que hoy necesita el proceso de transferencia de archivos del sistema SAREN. El mayor impacto que se alcanza con este sistema es que logra realizar las transferencias de la información que necesita el sistema SAREN para su correcto funcionamiento, brindándole seguridad, fiabilidad y protección a la información transferida.

Palabras claves

Sistema SAREN, sistema para el proceso de transferencia de archivos.

Abstract

In the registries and notaries of the Bolivarian Republic of Venezuela is undergoing a process of modernization and digitalization of the notarial and registry system through the introduction of SAREN system which aims to provide certainty and ensure the principles of contractual freedom and legality the rights of persons, acts, contracts and legal transactions.

This work shows the development of a system for the file transfer process of SAREN system that seeks to improve and benefit the process, which so far does not have an automated system to perform its functions.

It is presented the architecture that was developed with this system, as well as, of the functionalities proposed to give informatics solution currently needed the file transfer process SAREN system. The greatest impact is achieved with this system is able to make the transfers of information needed by the system SAREN for correct working by providing security, reliability and protection of information transferred.

Key Words

SAREN system, system for the file transfer process.

Índice

Introducción i

Capítulo 1: Fundamentación teórica 1

1.1. Introducción 1

1.2. Protocolos de transferencia de archivos 1

1.2.1. Xmodem transferencia de archivos entre ordenadores 1

1.2.2. HTTP (HyperText Transfer Protocol) Protocolo de transferencia de hipertexto 2

1.2.3. FTP (File Transfer Protocol) Protocolo de transferencia de archivos 2

1.2.4. FTP Extensiones de Seguridad 4

1.2.5. SSH (Secure SHell) 5

1.2.6. OpenSSH 6

1.2.7. SFTP, Secure File Transfer Protocol (Protocolo de transferencia de archivos seguro) 6

1.2.8. SSL/TLS Secure Sockets Layer/ Transport Layer Security 7

1.2.9. HTTPS (Hypertext Transfer Protocol Secure - Protocolo seguro de transferencia de hipertexto) 8

1.2.10. Protocolo FTPS (FTP/SSL) 8

1.2.11. AS2 (Applicability Statement 2) 11

1.3. Software para la transferencia de archivos utilizados a nivel mundial. 11

1.3.1. Tumbleweed SecureTransport 11

1.3.2. Sterling Managed File Transfer (MFT) 12

1.3.3. RepliWeb Managed File Transfer (RMFT) 12

1.3.4. CyberFusion Integration Suite (CFI) 13

1.3.5. Resumen general de características 14

1.4. Metodologías de desarrollo de software 14

1.4.1. Microsoft Solution Framework (MSF) 14

1.4.2. Extreme Programming (XP) 15

1.4.3. Rational Unified Process (RUP), Proceso Unificado de desarrollo de Software 16

1.4.4. Metodología para el desarrollo de la solución 20

1.5. Lenguaje de modelado para el desarrollo de la solución. 20

1.6. Herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) para el diseño de software. 21

1.6.1. Rational XDE (eXtended Development Environment) professional para Microsoft Visual Studio .Net 22

1.6.2. Enterprise Architect (EA) 22

1.6.3. Rational Rose Enterprise Edition 2003 23

1.6.4. Herramienta CASE utilizada para el modelado de la solución 23

1.7. Patrones de Diseño 23

1.7.1. Patrones GRASP (General Responsability Asignment Software Paterns) 24

1.7.2. Patrones GOF (Gang of Four) 25

1.8. Estilos y patrones de arquitectura 26

1.8.1. Arquitectura de la solución 28

1.9. Plataformas de desarrollo para la construcción de software. 29

1.9.1. Microsoft .NET 29

1.9.2. Plataforma de desarrollo MONO 31

1.9.3. Java 2 Enterprise Edition (Java 2EE) 31

1.9.4. Plataforma de desarrollo para la construcción de la solución 32

1.10. Paradigmas de programación 32

1.11. Seguridad e integridad de los archivos transferidos por la solución propuesta. 33

1.12. Descripción de las librerías utilizadas para el desarrollo del sistema. 34

1.12.1. Librería libcurl 34
1.12.2. Librería LibCurlNet 1.3 34
1.12.3. Librería DotNetZip 1.7 35
1.13. Conclusiones 35
Capítulo 2: Descripción de la solución propuesta 36
2.1. Introducción 36
2.2. Modelación del negocio 36
2.2.1. Procesos del negocio actual 36
2.2.2. Realización de los casos de uso del negocio 43
2.2.3. Proceso de negocio mejorado 46
2.3. Modelación del sistema 47
2.3.1. Requerimientos funcionales 47
2.3.2. Requerimientos no funcionales 47
2.3.3. Modelación de los casos de uso del sistema 48
2.4. Tratamiento de errores 58
2.5. Diseño del sistema 58
2.5.1. Descripción de las clases utilizadas 58
2.5.2. Diagrama de clases de la solución propuesta 70
2.5.3. Diagrama de clases persistentes 75
2.6. Modelo de implementación de la solución propuesta 77
2.6.1. Diagrama de componentes realizados 77
2.7. Modelo de despliegue de la solución propuesta. 79
2.8. Reglas de codificación 80
2.9. Conclusiones 82

Capítulo 3: Validación de la solución propuesta 83
3.1. Introducción 83
3.2. Pruebas de caja negra 83
3.2.1. Casos de prueba (CP) 84
3.3. Pruebas de caja blanca 91
3.3.1. Métrica de la complejidad ciclomática. 92
3.4. Conclusiones 101
Conclusiones generales 102
Recomendaciones 103
Referencias bibliográficas 104
Glosario de términos 106

Introducción

En el Ministerio del Poder Popular para Relaciones Interiores y Justicia de La República Bolivariana de Venezuela existe una dirección, Servicio Autónomo de Registros y Notarías (SAREN), la cual se encarga de gestionar, organizar y dirigir las notarías y registros de todo el país, en los cuales en la actualidad se lleva a cabo un proceso de modernización y digitalización de su sistema registral y notarial mediante la implantación del sistema SAREN el cual tiene como objetivo otorgar seguridad jurídica y garantizar los principios de libertad contractual y de legalidad de los derechos de las personas, actos, contratos y negocios jurídicos.

Un aspecto importante es la gestión de la transferencia de archivos que se lleva a cabo en el proceso registral o notarial, ya que dicho sistema se alimenta de diferentes entidades gubernamentales las cuales brindan a SAREN determinada información para su correcto funcionamiento, un ejemplo de esto es: La Oficina Nacional de Identificación y Extranjería (en lo adelante ONIDEX) que pone a disposición del sistema SAREN toda la información referente a la identificación de las personas para así evitar cualquier tipo de engaño o fraude en el proceso registral o notarial.

Actualmente este proceso de transferencia de archivos está lejos de efectuarse de un modo correcto, no existe un sistema de gestión centralizado para la transferencia de estos archivos que controle el proceso de transferencia y le brinde seguridad y fiabilidad a la información transferida, no está sujeto a ninguna forma de gestión y resulta vulnerable a problemas de comunicación aleatorios, la solución actual está basada principalmente en el protocolo FTP (File Transfer Protocol) un protocolo de red para la transferencia de archivos que ofrece poca seguridad en su modo nativo.

Los datos transferidos contienen información muy sensible para el correcto funcionamiento del sistema SAREN, por lo que se hace necesario que el proceso de transferencia cumpla determinadas normativas, se gestione de una forma centralizada por un sistema de gestión para la transferencia de archivos, el cual conste con un componente especializado para realizar sus operaciones y brindar seguridad, fiabilidad y confianza al proceso de transferencia de archivos.

A partir de la situación problemática antes expuesta el presente trabajo de diploma pretende darle respuesta mediante la solución del siguiente **Problema Científico**:

¿Cómo contribuir a la gestión del proceso de transferencia de archivos entre el sistema SAREN y las diferentes entidades gubernamentales?

Lo cual lleva al **objeto de estudio** que es el proceso de transferencia de archivos.

El **campo de acción** se centra en el proceso de transferencia de archivos del sistema SAREN.

El **objetivo general de la investigación** es:

- Diseñar e Implementar un sistema para la gestión del proceso de transferencia de archivos del sistema SAREN.

Hipótesis:

Si se realiza el diseño y la implementación de un sistema para la gestión del proceso de transferencia de archivos del sistema SAREN, entonces se contribuye al mejoramiento del proceso de transferencia de archivos del sistema SAREN.

Tareas de la investigación:

1- Realizar un estudio del proceso de transferencia de archivos que se llevan a cabo entre el sistema SAREN y la entidad gubernamental ONIDEX.

2- Realizar un estudio del estado del arte de las principales tendencias del proceso de gestión de transferencia de archivos de forma segura.

3- Realizar un estudio del estado del arte sobre las características de los principales software de gestión de transferencia de archivos utilizados por las empresas a nivel mundial.

4- Realizar un estudio del estado del arte de las principales tecnologías y metodologías de desarrollo de software haciendo énfasis en el diseño.

5- Diseñar el sistema para la gestión del proceso de transferencia de archivos del sistema SAREN, teniendo en cuenta la metodología de desarrollo de software seleccionada y la aplicación de patrones de diseño y de arquitectura.

6- Implementar, de acuerdo a la plataforma de desarrollo seleccionada, el sistema para la gestión del proceso de transferencia de archivos del sistema SAREN.

7- Realizar la validación y prueba del sistema mediante casos de prueba.

Métodos Científicos de Investigación:

Métodos Teóricos

Analítico - Sintético: Permitió la realización del análisis del proceso de transferencia de archivos llevado a cabo por el sistema SAREN para así poder entender mejor sus características y funciones. También se aplicó este método a través de la búsqueda de información bibliográfica en diferentes fuentes, documentos en Internet, etc. y arribar a las principales conclusiones de la investigación, así como para precisar las características del modelo del sistema propuesto.

Inductivo – deductivo: A partir del estudio de diferentes estilos de modelación de sistemas arribar al modelo del sistema en cuestión, se aplicó en el análisis de la información obtenida, a partir de la adaptación y generalización de los contenidos.

Modelación: Se realizaron varios modelos para poder caracterizar el sistema, procesos de la aplicación, modelos de clases y diagramas de componentes.

Métodos Empíricos

Observación: Se percibió de manera directa los problemas existentes a la hora de la transferencia de archivos entre el sistema SAREN y las diferentes entidades. La observación es abierta y no incluida porque el investigador no forma parte del grupo que se estudia y el objeto de la investigación conoce que está siendo observado.

Con el objetivo de contribuir al proceso de transferencia de archivos entre el sistema SAREN y las diferentes entidades de las cuales se alimenta se presentó esta propuesta de solución, la cual consiste en una aplicación de escritorio para la gestión del proceso de transferencia de archivos del sistema SAREN.

El presente trabajo de diploma está estructurado de la siguiente manera:

Capítulo 1: Fundamentación teórica, en el mismo se realiza un análisis sobre los protocolos de transferencias de archivos existentes, así como las características que presentan un conjunto de software utilizados a nivel mundial para la transferencia de información. Por otra parte se realiza un estudio sobre el estado del arte de las principales metodologías y tecnologías para el desarrollo de software.

Capítulo 2: Descripción de la solución propuesta, en este capítulo se comienza con el diseño e implementación de la solución propuesta, se aborda el tema referente a los procesos de negocio identificados, así como los requisitos funcionales y no funcionales que presenta el sistema.

Capítulo 3: Validación de la solución propuesta, en el mismo se realiza la prueba y validación del sistema, abordando temas como las diferentes pruebas de caja negra y caja blanca aplicadas a la solución, así como, el manejo de la herramienta Nunit para realizarle pruebas de unidad al código escrito.

Capítulo 1: Fundamentación teórica

1.1. Introducción

En el presente capítulo se realiza un estudio sobre el estado del arte de las principales tendencias del proceso de transferencia de archivos de forma segura basado en el estudio de las RFC (Request For Comments) referidas al tema, se describen las características de algunos de los principales software para la transferencia de archivos utilizados por las empresas a nivel mundial. Se analizan las principales metodologías para el desarrollo de software haciendo énfasis en el flujo de trabajo relacionado con el diseño teniendo en cuenta las tendencias actuales y las principales tecnologías utilizadas. Además se aborda brevemente el tema referente a las plataformas de desarrollo para la construcción de software.

1.2. Protocolos de transferencia de archivos

1.2.1. Xmodem transferencia de archivos entre ordenadores

"Xmodem es el protocolo de transferencia de archivos más antiguo, desarrollado en 1977 por Ward Christensen, es básicamente un método para la transferencia de archivos entre PC (Personal Computer)" [1].

La transferencia de archivos utilizando Xmodem se rige por el conocimiento de ambos ordenadores (transmisor y receptor) de donde encontrar los datos a transmitir y donde almacenar los datos recibidos.

Xmodem utiliza caracteres ASCII (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange - Código Estadounidense Estándar para el Intercambio de Información) especiales para su funcionamiento a continuación se listan algunos ejemplos:

Nombre	**Decimal**	**Hexadecimal**	**Descripción**
SOH	01	H01	Comienzo de encabezado.
ACK	06	H06	Confirmación positiva.
NAK	21	H15	Confirmación negativa.
CAN	24	H18	Cancelar.

La computadora receptora es la encargada de controlar el manejo del protocolo Xmodem ya que se encarga de enviar caracteres NAK hacia el transmisor que en este caso espera a la llegada de este tipo de carácter para comenzar con la transmisión.

1.2.2. HTTP (HyperText Transfer Protocol) Protocolo de transferencia de hipertexto

HTTP es un protocolo que ha sido usado en la Web (World-Wide Web) desde 1990, su primera versión se refiere como HTTP/0.9 un simple protocolo para transferir datos en bruto a través de Internet, seguido de HTTP/1.0 y HTTP/1.1.

HTTP sigue un esquema de petición-respuesta entre un cliente y un servidor, el cliente envía solicitudes al servidor y el servidor responde a dichas solicitudes. La mayoría de la comunicación HTTP es iniciada por un agente de usuario (el cliente que inicia la solicitud) y se compone de una petición que deberá aplicarse a un origen de recursos en algún servidor. Los recursos hacen referencia a la información que se transfiere y están identificados por una URL (Uniform Resource Locator, es decir, Localizador Uniforme de Recurso).

Mensajes HTTP

Un mensaje HTTP consiste fundamentalmente en las peticiones por parte del cliente al servidor y las respuestas del servidor al cliente. Está compuesto por un encabezado (message-header) lo que posibilita que se envíe información descriptiva en el mensaje, seguido de una línea en blanco indicando el fin del encabezado y por último está compuesto de algún dato (message-body).

1.2.3. FTP (File Transfer Protocol) Protocolo de transferencia de archivos

El protocolo FTP ha pasado por una larga evolución a través de los años, la primera propuesta de mecanismos para la transferencia de ficheros se presentó en 1971 por un grupo de investigadores del Instituto Tecnológico de Massachusetts (MIT) la cual quedo recogida en la RFC (Request For Comments) 114, la versión utilizada actualmente data de 1985 [RFC 959] con extensiones de seguridad [RFC 2228] de 1997.

“Los objetivos de FTP son: promocionar el uso compartido de ficheros (programas y/o datos), animar al uso indirecto o implícito (a través de programas) de servidores remotos, hacer

transparente al usuario las variaciones entre la forma de almacenar ficheros en diferentes ordenadores y transferir datos fiable y eficientemente" [RFC 959].

Los datos son transferidos desde un dispositivo de almacenamiento en el computador que envía los datos a otro dispositivo en el computador que los recibe. Existen diferentes tipos de datos entre ellos se encuentran:

- ASCII, el cual es el tipo por defecto y su propósito es transferir archivos de texto.
- EBCDIC (Extended Binary Coded Decimal Interchange Code) se utiliza para las transferencias de texto entre ordenadores que usan EBCDIC como su representación de carácter interna.
- Imagen se utiliza para las transferencias de datos binarios.

Es importante conocer el tipo de dato que se utiliza en la transferencia de los mismos, porque de no utilizarse las opciones correctas, la información contenida en los archivos se puede destruir a lo largo de la red.

El canal de comunicación se establece entre el usuario y el servidor donde el usuario es el encargado de enviar órdenes FTP e interpretar las respuestas recibidas, el servidor interpreta las órdenes FTP, envía respuestas y se encarga de controlar la conexión de datos así como la transferencia.

Entre las órdenes FTP se encuentran:

- Las órdenes de control de acceso.
- Las órdenes de parámetros de transferencias.
- Las órdenes de servicio FTP.

"Las respuestas a órdenes en FTP están pensadas para asegurar la sincronización entre peticiones y acciones en el proceso de transferencia de ficheros y para garantizar que el proceso de usuario siempre conoce el estado del servidor" [RFC 959].

Una respuesta FTP está compuesta por un número de tres cifras seguidas de un texto el cual se separa del código numérico por un espacio, dicho texto va dirigido a las personas, debido que resulta difícil leer el código numérico.

1.2.4. FTP Extensiones de Seguridad

En una comunicación FTP se crea un canal de control desde la máquina cliente al servidor, por dicho canal se realiza la transmisión de las órdenes y respuestas FTP. Los comandos que se utilizan para realizar el control de acceso (es decir para identificar al usuario) en el protocolo FTP definido en la RFC 959 envía los nombres de usuario y contraseñas en texto claro. Esto representa un riesgo para la seguridad ya que cualquiera puede robar dichos datos con solo monitorear la red. Debido a esta dificultad se le definieron un grupo de extensiones al protocolo FTP (especificado en la RFC 959) las cuales fueron recogidas en la RFC 2228.

Dichas extensiones proporcionan una sólida autenticación, integridad y confidencialidad en el canal de control con la introducción de nuevos comandos opcionales los cuales se listan a continuación:

Comandos opcionales introducidos en esta especificación

AUTH (Authentication/Security Mechanism): Indica los mecanismos de autentificación soportados, para poder negociar en el contexto de seguridad. El argumento de campo es una cadena la cual es sensible a las mayúsculas y sus valores de dicha cadena que comienzan con "X" están reservados para uso local.

ADAT (Authentication/Security Data): Permite el envío de datos de seguridad dependientes del mecanismo negociado. Los datos se refieren específicamente al mecanismo de seguridad especificado por el anterior comando AUTH. El comando ADAT debe ir precedido por un comando AUTH con éxito.

PBSZ (Protection Buffer Size): Establece el tamaño máximo para los bloques que serán cifrados, no hay tamaño predeterminado y debe estar precedido por un comando ADAT con éxito.

PROT (Data Channel Protection Level): Indica el nivel de protección que se usará en la conexión de datos (C: Abierto, S: Seguro, C: Confidencial, P: Privado). El nivel por defecto en caso de no especificar otro nivel es el nivel abierto. El comando PROT debe ser precedido por un comando PBSZ con éxito.

- Abierto, el canal de datos transportará los datos en bruto, sin seguridad aplicada.
- Seguro, indica que será protegida la integridad de los datos que son transferidos.

- Confidencial, indica que será protegida la confidencialidad de los datos que son transferidos.
- Privado, indica que será protegida la integridad y la confidencialidad de los datos que son transferidos.

CCC (Clear Command Channel): Indica que los siguientes comandos no van cifrados, este comando no tiene argumentos. Es utilizado en entornos donde es deseable utilizar un mecanismo de seguridad para la autenticación del cliente y el servidor.

MIC (Integrity Protected Command): Indica que se reestablece un canal de seguridad para comandos con verificación de la integridad.

CONF (Confidentiality Protected Command): Indica que se reestablece un canal de seguridad para comandos con verificación de la confidencialidad.

ENC (Privacy Protected Command): Indica que se reestablece un canal de seguridad para comandos con verificación de la privacidad.

1.2.5. SSH (Secure SHell)

SSH es un protocolo que se utiliza para acceder a máquinas remotas a través de una red y permite copiar datos de forma segura ya que la información es transferida por un canal seguro tunelizado mediante SSH.

Para el tema de la seguridad, SSH utiliza técnicas de cifrado por lo que la información viaja por el medio de comunicación de manera no legible, protegiendo de esta manera el usuario y la contraseña de la conexión, así como, la información que se escribe durante toda la sesión. Con estas medidas se evita que terceras personas que puedan estar monitoreando la red tengan acceso a los datos transferidos y de esta manera puedan manipular la información entre los destinos.

SSH está compuesto por tres componentes principales:

- El protocolo de la capa de transporte (ssh-trans), proporciona la autenticación del servidor, además provee integridad, confidencialidad y compresión opcional.

- El protocolo de autenticación de usuario (ssh-userauth), se apoya en los servicios provistos por la capa de transporte, provee los mecanismos para la autenticación del usuario.
- El protocolo de conexión (ssh-connect), se encarga de crear el túnel seguro por donde viaja la información.

A principios de 1999 se empezó a escribir una versión que se convertiría en la implementación libre por excelencia llamada OpenSSH.

1.2.6. OpenSSH

"OpenSSH es una versión libre del paquete de herramientas para la comunicación segura del protocolo SSH, una solución de seguridad que está ganando la confianza de un número cada vez mayor de usuarios de Internet" [2]. En OpenSSH el cifrado comienza antes de la autenticación y ninguna contraseña se transmite sin cifrar, así como, ningún otro tipo de información.

Para la distribución de OpenSSH se tienen dos equipos de trabajo, un equipo que se encarga tan solo de producir un código limpio, simple y seguro, lo que facilita el control de la calidad y la revisión del código. El otro equipo toma este código puro y lo convierte en una versión portable para que pueda utilizarse en otros sistemas operativos.

1.2.7. SFTP, Secure File Transfer Protocol (Protocolo de transferencia de archivos seguro)

El SFTP es un protocolo de red que proporciona la transferencia de archivos, donde todos los datos que circulan por la red se encuentran cifrados. Es utilizado normalmente con SSH a fin de asegurar la información transferida, pero puede ser utilizado con otros protocolos de seguridad. Por tanto la seguridad en sí no la provee el protocolo por sí mismo, sino que es provista por SSH o el protocolo subyacente utilizado para este cometido. Los programas de SFTP ofrecen una interfaz interactiva similar a los tradicionales programas de FTP lo que beneficia a los clientes que usan este protocolo.

En general este protocolo sigue un simple modelo de petición-respuesta, donde cada petición-respuesta contiene un número de secuencia y donde múltiples peticiones se pondrán

pendientes simultáneamente. Existe una cantidad alta de mensajes de petición, pero un número pequeño de mensajes de respuesta. Cada petición tiene una o varias respuestas.

En algunas ocasiones cuando se menciona SFTP se piensa que es el resultado de ejecutar FTP con SSH, esto es un error, realmente se está en presencia de un nuevo protocolo desarrollado por el grupo de trabajo IETF (Internet Engineering Task Force).

1.2.8. SSL/TLS Secure Sockets Layer/ Transport Layer Security

El protocolo SSL fue desarrollado por Netscape Communications Corporation, proporciona sus servicios de seguridad cifrando los datos intercambiados entre el cliente y el servidor, SSL versión 3.0 se publicó en 1996, que más tarde sirvió como base para desarrollar TLS versión 1.0 definido por primera vez en la RFC 2246.

“SSL protege los datos transferidos mediante el cifrado, activado por un certificado SSL en un servidor. Los certificados SSL contienen una clave pública y otra privada. La clave pública se utiliza para cifrar la información y la privada para descifrarla” [3].

En el protocolo SSL el cliente y el servidor negocian el algoritmo de cifrado que se usará en la comunicación, además ocurre un intercambio de claves públicas para la autenticación basada en certificados digitales y se cifra el tráfico de información con un cifrado simétrico.

El Protocolo TLS versión 1.0

El objetivo principal del protocolo TLS es proporcionar privacidad e integridad a los datos que se transfieren en la comunicación entre dos aplicaciones. Está compuesto por dos capas: El protocolo TLS record y el protocolo TLS handshake. El protocolo TLS record provee una conexión segura que consta de dos propiedades básicas:

- Conexión privada: La criptografía simétrica es usada para el cifrado de los datos utilizando como algoritmos de cifrado el DES (Data Encryption Standard), RC4 (Rivest Cipher 4), entre otros. Las llaves de este cifrado simétrico se generan únicas para cada conexión y se basan en un negociado secreto por otro protocolo (como el Protocolo TLS handshake). El Protocolo TSL record también se puede utilizar sin ningún tipo de cifrado.

- Conexión fiable: El mensaje que se transporta incluye un control de integridad basado en una clave MAC (Message Authentication Code), las funciones de seguridad hash (por ejemplo, SHA (Secure Hash Algorithm), MD5 (Message Digest 5), etc.) son usadas para conformar el MAC. El TLS record puede operar sin MAC pero por lo general solo se utiliza en este modo cuando otro protocolo está usando al TLS record como transporte para las negociaciones de los parámetros de seguridad.

Una de las ventajas del protocolo TSL es que se trata de un protocolo de aplicación independiente, protocolos de niveles superiores pueden ser capas superiores trasparentes al protocolo TSL.

1.2.9. HTTPS (Hypertext Transfer Protocol Secure - Protocolo seguro de transferencia de hipertexto)

HTTPS es el protocolo de red destinado a la transferencia de hipertexto de forma segura, es decir es la versión segura de HTTP. Es el resultado de combinar HTTP y SSL para crear un canal cifrado para el tráfico de la información, de esta manera se consigue enviar los datos que contienen información sensible (referente a los usuarios y contraseñas) en un formato no legible para cualquiera tercera persona que logre interceptar la transferencia de datos de la conexión.

El puerto por defecto utilizado por este protocolo es el 443 y para conocer si una dirección web utiliza el protocolo HTTPS se debe observar que en la barra de direcciones de nuestro navegador aparezca como protocolo “https” en lugar de “http” al inicio de la dirección.

En HTTPS el cliente realiza la conexión con el servidor, negocia la seguridad SSL y a continuación realiza la transferencia con SSL por el canal de datos.

1.2.10. Protocolo FTPS (FTP/SSL)

El protocolo FTPS es un término usado para identificar un número de formas en las que el protocolo FTP puede realizar transferencias de archivos de un modo seguro, cada forma conlleva al uso de una capa SSL/TLS subyacente al protocolo FTP estándar la cual se utiliza para cifrar el canal de control y/o datos. Es un error confundirlo con el protocolo SFTP, el cual suele ser usado con el protocolo SSH.

La forma más común de usar FTP y SSL es mediante dos mecanismos: FTPS explícito o FTPS implícito

FTPS explícito: En este mecanismo el cliente se conecta al servidor FTP utilizando el puerto por defecto (21) y se comienza así una sesión FTP normal, es decir, sin seguridad. Con el fin de establecer la conexión SSL/TLS es necesario que el cliente emita al servidor un comando específico después de establecerse la conexión.

FTPS implícito: En este caso el cliente se conecta al servidor FTP por un puerto (ejemplo 990) distinto definido por el servidor para conexiones seguras y se realiza la negociación SSL/TLS antes de enviar cualquier comando FTP al servidor.

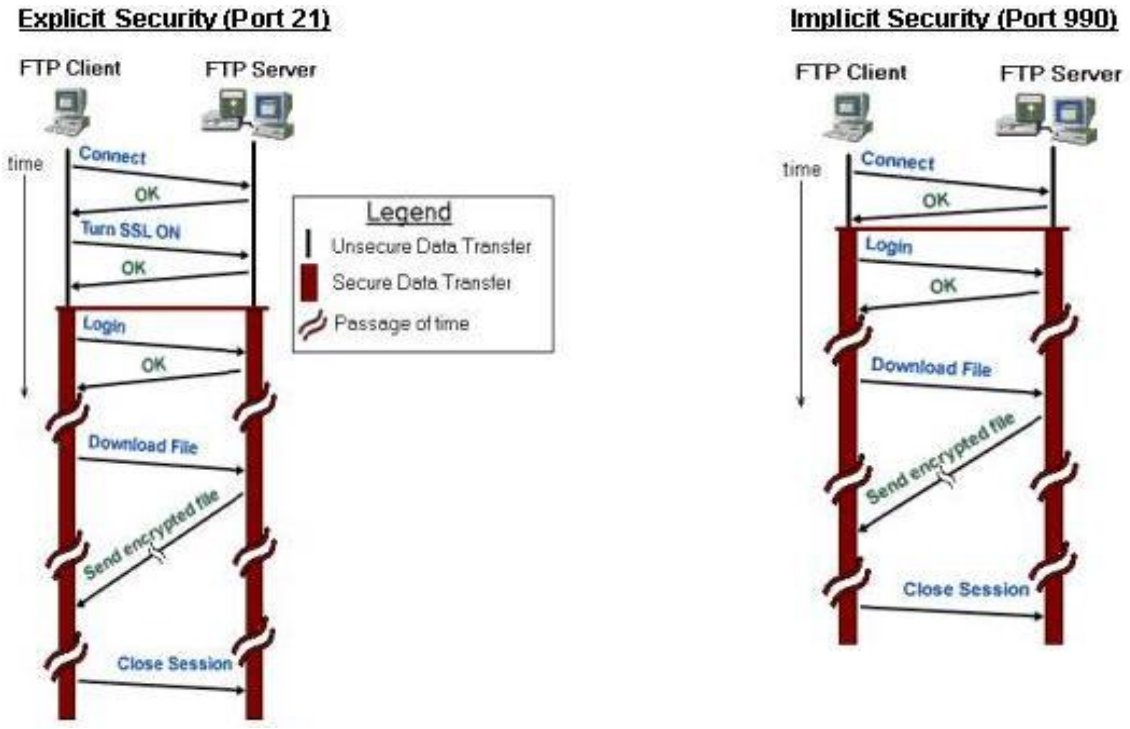

Figura 1: FTPS explícito e implícito.

Aseguramiento de FTP con TLS según RFC 4217

Aquí se combinan tres documentos para proporcionar un útil y seguro protocolo de transferencias de archivos. Estos documentos son los siguientes:

RFC 959 [RFC-959]

La descripción del protocolo de transferencia de archivos FTP.

RFC 2246 [RFC-2246]

La descripción del protocolo TLS desarrollada desde el protocolo SSL versión 3.0.

RFC 2228 [RFC-2228]

Extensiones de seguridad al protocolo FTP para permitir la autenticación, confidencialidad, e integridad de los mensajes.

Período de negociación sobre el control del puerto

El servidor escucha en el puerto de control FTP normal y la sesión de inicio no está asegurada en absoluto. Una vez que el cliente desea garantizar el período de sesiones, el comando AUTH es enviado al servidor y luego TLS permite que la negociación tenga lugar.

Mecanismos para el comando auth

El comando auth toma un único parámetro para definir la seguridad del mecanismo que se negoció. El nombre del mecanismo para la negociación de TLS es la cadena de caracteres determinados en tls-parm. Esto permite al cliente y al servidor negociar el control de la conexión TLS sin alterar la protección del canal de datos.

Seguridad de la conexión de datos

El nivel de seguridad está determinado por el comando **prot**, tal como se especifica en la RFC-2228, este comando permite que el cliente/servidor negocie el nivel de seguridad de la conexión de datos.

Para TLS, la conexión de datos puede tener uno de los dos niveles de seguridad.

1- Clear (abierto) especificado por el comando 'prot c'

2- Privado especificado por el comando 'prot p’

Con el nivel de protección “Clear” la conexión de datos es realizada sin TLS, es decir, la conexión realizada no posee autenticación, confidencialidad, ni integridad.

1.2.11. AS2 (Applicability Statement 2)

“Es una especificación técnica para transportar EDI (Electronic Data Interchange), XML (Extensible Markup Language) y otros documentos en Internet de una forma segura y confiable” [4]. AS2 permite que dos empresas puedan intercambiar cualquier tipo de documento por internet donde la única conectividad requerida es que ambas dispongan de sitios Web.

Para garantizar la seguridad AS2 ofrece distintas opciones como son: el envío de los datos por una conexión segura (HTTPS), encriptación completa del documento mediante certificados digitales, así como, el uso de la firma digital para garantizar de esta manera que el mensaje proviene de la fuente correcta y además por esta vía el receptor se asegura de la validez del mensaje.

El funcionamiento de AS2 se puede resumir de la siguiente manera: El servidor AS2 utiliza un certificado digital para encriptar un documento comercial firmado digitalmente, luego el servidor envía el documento por Internet al servidor AS2 de la empresa receptora, el servidor receptor envía un mensaje de acuse de recibo al servidor emisor y por último el servidor AS2 confirma la firma digital y desencripta los datos mediante el certificado digital.

1.3. Software para la transferencia de archivos utilizados a nivel mundial.

1.3.1. Tumbleweed SecureTransport

Es un paquete de software para gestionar la transferencia de archivos de forma segura que se lleva a cabo en las empresas, es utilizado actualmente en redes financieras para operaciones diarias y en la mayoría de los principales bancos de EEUU (United States of America) para atender decenas de miles de clientes corporativos.

SecureTransport permite a las organizaciones:

- Realizar transferencias de archivos de manera fiable a través de los siguientes protocolos: FTP, FTPS (SSL/TLS), SFTP (SSH), HTTP, HTTPs (SSL/TLS) y AS2.
- Elaborar informes y resúmenes una vez concluida la transferencia.

- Reducir los riesgos de seguridad en el intercambio de información.
- Monitorear y analizar la transferencia de archivos en tiempo real.

Las soluciones para gestionar la transferencia de archivos SecureTransport están compuestas de tres componentes principales: **SecureTransport Server**, **SecureTransport Client** y la pasarela optativa para la transferencia de archivos **SecureTransport Edge**.

1.3.2. Sterling Managed File Transfer (MFT)

Sterling Commerce Managed File Transfer ofrece soluciones de gestión de transferencia de archivos para empresas de todo el mundo.

Algunas de sus ventajas son:

- Elimina los riesgos asociados con la transferencia de archivos basada en FTP.
- Garantiza la seguridad y la protección de los datos, además permite monitorear lo que ocurre en tiempo real.

Las soluciones Managed File Transfer de Sterling son:

Connect Direct: se encarga de transferir archivos dentro y entre empresas.

Connect Enterprise: una solución para concentrar las transferencias de archivos de forma segura a través de la comunidad empresarial.

Sterling Secure Proxy: una aplicación proxy para ofrecer una seguridad más rigurosa en la transferencia de archivos.

Gentran Integration Suite for File Transfer: es una solución SOA (Service Oriented Architecture) donde se concentra la transferencia de archivos, permite la gestión tanto de archivos como de transacciones para la colaboración entre las empresas.

1.3.3. RepliWeb Managed File Transfer (RMFT)

RMFT proporciona una transferencia de archivos segura y fiable basada en diversos protocolos (FTP, FTPS, SFTP, HTTPS, SSH) de transferencia. Entre sus principales características se encuentran: una eficaz y automática recuperación ante fallas garantizando una entrega segura y una alta protección a la integridad de los datos. Para lograr que la información viaje segura

RMFT hace uso del protocolo SSL, además se utiliza la firma digital así como diversos métodos de codificación.

RMFT brinda opciones de procesamiento de datos antes y después de realizada la transferencia, lo que posibilita que se realicen inspecciones automáticas, barridos en busca de virus, conversión de archivos, firmas digitales, incorporación de marcas de agua, etc.

RMFT utiliza el protocolo LFA (Large Files Accelerator - acelerador de archivos de gran tamaño) por lo que se aumenta su eficacia en las transferencias de grandes volúmenes de información.

1.3.4. CyberFusion Integration Suite (CFI)

Es un paquete completo para la gestión de la transferencia de archivos que ayuda a garantizar la seguridad, el control, y la integración de la información transferida. Entre sus componentes se encuentran el CFI Platform Server, Internet Server y Command Center. El CFI Platform Server y el Internet Server pueden ser implementados independientes o juntos en dependencia de los requerimientos de la empresa y el módulo opcional Command Center proporciona un punto de control central.

Algunas características y beneficios:

- Supervisión (monitorear) y administración centralizada.
- Alertas (correo electrónico).
- Estado en tiempo real de la transferencia de archivos.
- Herramientas para solucionar problemas.
- Reportes de usuarios, grupos, etc.
- Cifrado utilizando algoritmos criptográficos.
- SSL, certificados digitales.
- Autenticación y autorización.

1.3.5. Resumen general de características

Las soluciones anteriormente mencionadas forman parte de la familia de software destinada a la gestión del proceso de transferencia de archivos de forma segura, donde, su principal objetivo es lograr la seguridad, fiabilidad e integridad de la información transferida. Además soportan la mayoría de los protocolos de transferencias de archivos existentes, ofrecen opciones de monitoreo en tiempo real y permiten la elaboración de informes acerca de la transferencia de archivos realizadas. Para lograr la seguridad de la información transferida hacen uso del empleo de diferentes técnicas entre las cuales se pueden citar: la utilización del protocolo SSL, certificados digitales, firma digital, así como, el empleo de algoritmos criptográficos para cifrar la información.

1.4. Metodologías de desarrollo de software

1.4.1. Microsoft Solution Framework (MSF)

Esta es una metodología que “proporciona un sistema de modelos, principios, y pautas para dar soluciones a empresas que diseñan y desarrollan de una manera que se asegure de que todos los elementos de un proyecto, tales como gente, procesos, y herramientas, puedan ser manejados con éxito” [5].

El modelo de proceso de MSF está compuesto de un conjunto de actividades que guían el desarrollo de software, esta caracterizado por su flexibilidad y se puede adaptar para el diseño y desarrollo de una amplia gama de proyectos de una empresa, además combina los mejores principios del modelo en cascada y del modelo en espiral.

Fases del modelo MSF

- Visión y Alcance: El resultado final de esta fase es la definición de la visión y alcance del proyecto, además se definen quienes estarán a cargo de la dirección del proyecto (líder y otros encargados), así como, las metas y objetivos del proyecto.

- Planificación: En esta fase se realiza el proceso de diseño del sistema incluyendo toda la recogida y análisis de los requerimientos, por lo que en esta fase se determina que desarrollar y se planea como crear la solución.

- Desarrollo: La meta de esta fase es la construcción de los elementos y entregables de la solución (código de los componentes y documentación).
- Estabilización: La meta de esta fase es determinar que la solución posee la calidad necesaria para pasar a la fase de despliegue, por lo que, el equipo de desarrollo se concentra en detectar y solucionar errores.
- Despliegue: Esta fase se obtiene la última aprobación del cliente, se despliega la tecnología de la solución y se traspasa la solución al personal de soporte y operaciones.

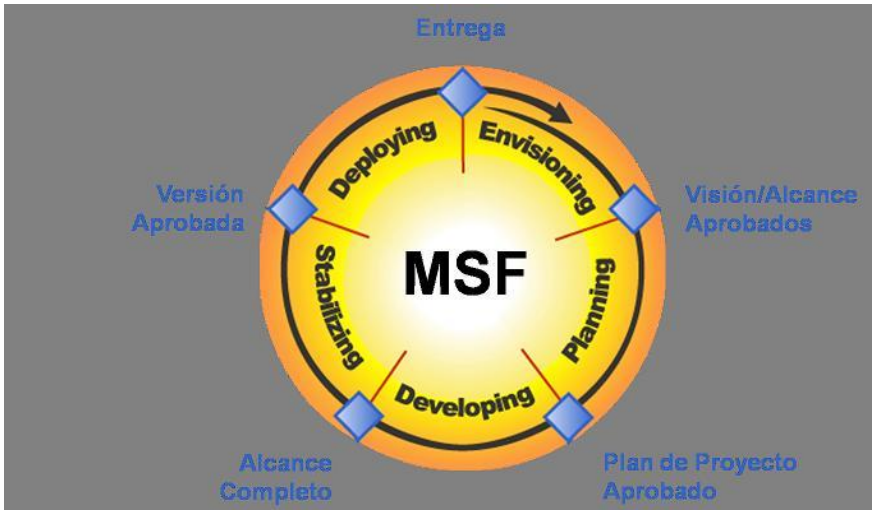

Figura 2: Fases del modelo de proceso de MSF

1.4.2. Extreme Programming (XP)

"XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo" [6].

Se recomienda que se utilice XP en un proyecto donde el cliente forma parte del equipo de trabajo y donde los requisitos son muy cambiantes e imprecisos.

Entre algunas de las características de XP tenemos:

- Desarrollo iterativo e incremental, se realizan pequeñas mejoras unas tras otras.

- Programación por parejas, se recomienda que el código sea escrito entre dos personas, se basa, en que la calidad ganada (de esta manera el código es revisado y discutido al mismo tiempo) es más importante que la posible pérdida de la productividad inmediata.
- Interacción del cliente, frecuente interacción entre el cliente y el equipo de desarrollo.
- Se corrigen todos los errores antes de añadir una nueva funcionalidad.
- Refactorización del código, se reescriben partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento, de esta forma se garantiza la simplicidad del código escrito.
- Código compartido, promueve que todo el equipo de desarrollo pueda corregir y extender cualquier parte del proyecto, en vez de dividir en grupos de trabajo la responsabilidad de desarrollar cada módulo.

"La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Mientras más simple es el sistema, menos se tendrá que comunicar sobre este" [7].

1.4.3. Rational Unified Process (RUP), Proceso Unificado de desarrollo de Software

Un proceso define "quién" está haciendo "qué", "cuándo" y "cómo" para alcanzar un determinado objetivo. Un proceso de desarrollo de software es la definición del conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, a modo de plantilla que explica los pasos necesarios para terminar el proyecto [8].

Como **RUP** es un proceso, en su modelación define como sus principales elementos:

- **Trabajadores** ("Quién"): Define el comportamiento y responsabilidades de un individuo, grupo de individuos, sistema automatizado o máquina, los cuales se encargan de realizar las actividades y es el responsable de una serie de artefactos.
- **Actividades** ("Cómo"): Es una tarea que tiene un propósito claro, se le asigna a un trabajador y manipula elementos.

- **Artefactos** (“Qué”): Productos tangibles del proyecto, son producidos, modificados y usados por las actividades.
- **Flujo de actividades** (“Cuándo”): Secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable.

RUP divide su ciclo de vida en cuatro fases dentro de las cuales se llevan a cabo un grupo de iteraciones.

Fases:

- **Conceptualización** (Concepción o Inicio): Se delimita el ámbito del proyecto haciendo énfasis en la descripción del negocio y el modelado de los requerimientos, además se identifican riesgos críticos.
- **Elaboración**: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen.
- **Construcción**: Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario.
- **Transición**: El release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores.

Flujos de Trabajos

- **Modelamiento del negocio**: Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
- **Requerimientos**: Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
- **Análisis y diseño**: Describe cómo el sistema será realizado a partir de los requerimientos previamente identificados, es un plano para la implementación del sistema.

- **Implementación**: Se define la organización del código y la implementación de las clases y objetos en términos de componentes. Además se prueban los componentes desarrollados y se realiza la integración de dichos componentes en un sistema ejecutable.
- **Prueba** (Testeo): Busca identificar los defectos y corregirlos antes de la instalación final del sistema, se verifica la integración apropiada de los componentes y que se satisfacen los requerimientos de los clientes.
- **Instalación** (Despliegue): Produce release del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.
- **Administración del proyecto**: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
- **Administración de configuración y cambios**: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto.
- **Ambiente**: Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto.

El ciclo de vida de RUP se caracteriza por ser: dirigido por casos de uso, centrado en la arquitectura y además por ser iterativo e incremental.

Los casos de uso reflejan lo que los usuarios futuros necesitan y desean lo cual se representa a través de los requerimientos, a partir de aquí los casos de uso guían el proceso de desarrollo, obteniéndose un producto final que cuenta con la calidad requerida por el cliente.

La arquitectura muestra una visión del sistema que describe los elementos del modelo que son importantes para la construcción del software, creándose de esta manera los cimientos del sistema que son necesarios como base para comprenderlo y desarrollarlo. RUP propone que se desarrolle el software mediante iteraciones comenzando por los casos de uso relevantes desde el punto de vista para la arquitectura.

RUP propone que cada fase se desarrolle en iteraciones, donde en cada iteración se involucran actividades de todos los flujos de trabajo, aunque se desarrollan fundamentalmente algunos flujos más que otros.

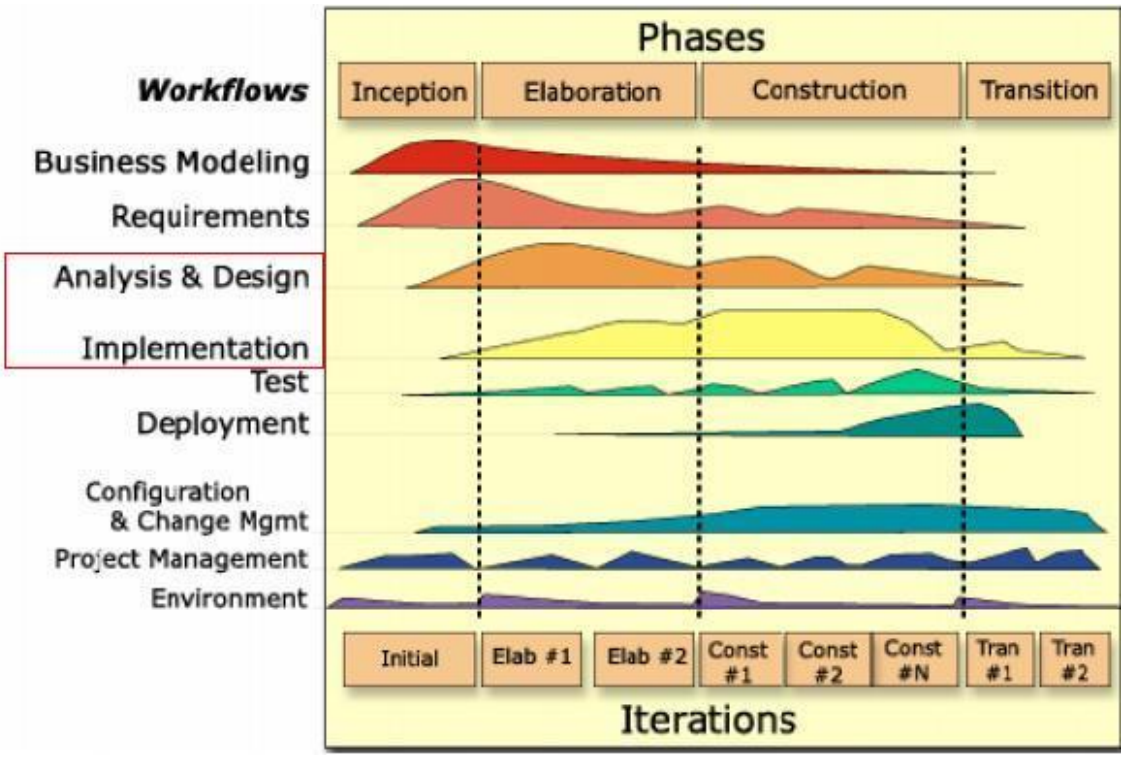

Figura 3: Vista de RUP en dos dimensiones

Diseño en RUP

El diseño se centra en las fases de elaboración y construcción. Esto contribuye a una arquitectura estable y sólida, además se crea un plano del modelo de implementación. Durante la fase de construcción, cuando la arquitectura es estable y los requisitos están bien entendidos, el centro de atención se desplaza a la implementación. El modelo de diseño está muy cercano al de implementación, lo que es natural para mantener dicho modelo a través del ciclo de vida completo del software.

Entre los artefactos generados en el diseño se encuentran: modelo de diseño, clases del diseño, realización de los casos de uso, subsistema de diseño, interfaz, descripción de la arquitectura y el modelo de despliegue.

1.4.4. Metodología para el desarrollo de la solución

Se seleccionó RUP como metodología a utilizar para el desarrollo de la solución propuesta teniendo en cuenta que:

Es un proceso formal que indica paso a paso como desarrollar un producto de alta calidad que satisfaga los requerimientos del usuario. Es flexible permitiendo ser configurado de acuerdo a las características y necesidades específicas de la organización y del proyecto. Se basa en la asignación de roles, por lo que, cada integrante del equipo de desarrollo se vincula directamente con la parte del proceso que le compete, filtrándose el resto. RUP contempla prácticamente en todo su ciclo de desarrollo las etapas de modelado del negocio, requerimientos y análisis, permitiendo que se identifiquen los cambios de forma temprana reduciendo su impacto en el desarrollo del software. Implementa un desarrollo iterativo del software que aborda las tareas rigurosas primero, logrando reducir los riesgos del proyecto y obtener un subsistema ejecutable tempranamente. RUP apoya el desarrollo de arquitecturas basadas en componentes, obteniendo una arquitectura flexible y fácil de modificar, además se promueve la reutilización de estos componentes.

1.5. Lenguaje de modelado para el desarrollo de la solución.

Se seleccionó UML (Unified Modeling Languaje) como lenguaje para realizar el modelado de la solución propuesta teniendo en cuenta que la metodología de desarrollo de software seleccionada propone y utiliza UML como lenguaje de modelado.

“UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software” [9].

“Un objetivo final de UML era ser tan simple como fuera posible pero manteniendo la capacidad de modelar toda la gama de sistemas que se necesita construir” [10].

UML posee una serie de elementos los cuales son combinados a través de reglas para conformar diagramas, está basado en las tecnologías orientadas a objetos, tales como, objetos, clases y componentes. Permite agrupar los modelos en paquetes, beneficiando la comprensión

y entendimiento de estos modelos, además de esta manera el equipo de trabajo puede dividir grandes sistemas en piezas de trabajo más manejables.

UML no es un proceso o método, es solamente un lenguaje que nos permite representar los modelos del sistema que se desarrolla, mas no ofrece ningún tipo de guía o criterios de cómo obtener dichos modelos.

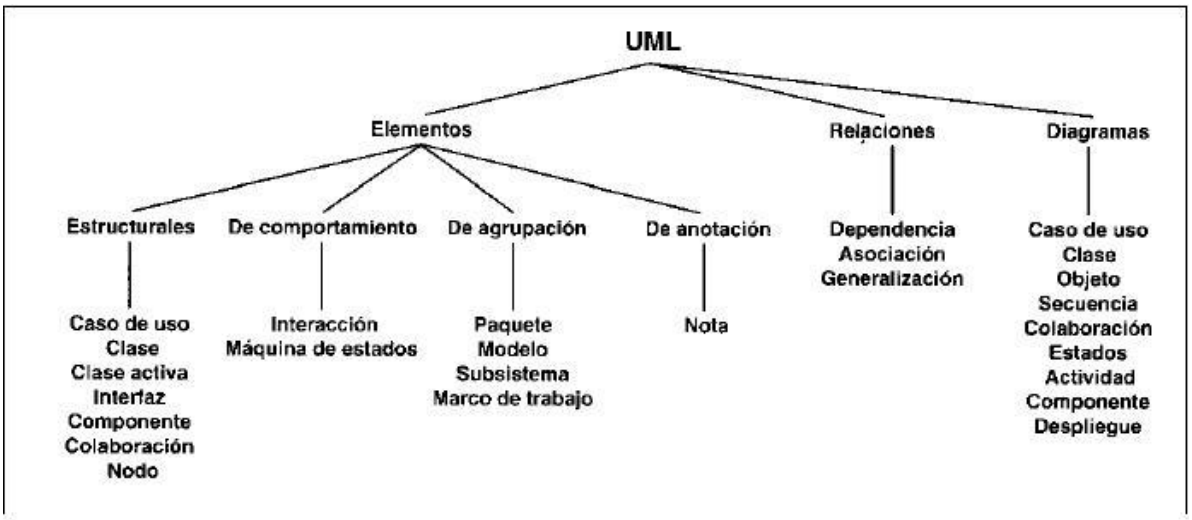

Figura 4: Vocabulario de UML

1.6. Herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) para el diseño de software.

Las herramientas CASE son aplicaciones informáticas, su principal objetivo es aumentar la productividad disminuyendo el tiempo y los costos en el desarrollo de software. Sus funcionalidades radican en ayudar al equipo de desarrollo en los distintos aspectos del ciclo de vida del proyecto, como son: el diseño, planificación, generación de código, la detección de posibles errores, entre otros.

1.6.1. Rational XDE (eXtended Development Environment) professional para Microsoft Visual Studio .Net

Rational XDE Professional para Visual Studio .Net acelera el proceso de desarrollo de software, debido a que permite al desarrollador programar y diseñar directamente desde su IDE (Integrated Development Environment) - Microsoft Visual Studio .NET sin tener que estar alternando entre dos herramientas diferentes.

Algunas de sus características son:

- Desarrollo basado en modelos con soporte para UML.
- Ingeniería bidireccional para lenguajes C++, Visual Studio y .NET.
- Sincronización de código-modelo automática o bajo petición.
- Patrones y plantillas de código definibles por el usuario.
- Modelado asistido con edición sensible al lenguaje.
- Diseño de bases de datos lógicas y físicas.
- Publicación web y generación de informes.

1.6.2. Enterprise Architect (EA)

Enterprise Architect proporciona un conjunto de funcionalidades para el diseño y construcción de sistemas de software. EA soporta UML como lenguaje para definir los distintos modelos de un proyecto. Cubre todos los aspectos del ciclo de desarrollo, proporcionando una trazabilidad completa desde la fase inicial del diseño a través del despliegue y mantenimiento. También provee soporte para pruebas, mantenimiento y control de cambio.

EA soporta la generación e ingeniería inversa de código fuente para muchos lenguajes, incluyendo C++, C#, Java, Delphi, VB.Net, Visual Basic, ActionScript y PHP. También se puede exportar rápidamente la documentación de los modelos creados en formato RTF (Rich Text Format) e importar a Word para su posterior personalización.

1.6.3. Rational Rose Enterprise Edition 2003

Rational Rose es la herramienta CASE desarrollada por los creadores de UML (Booch, Rumbaugh y Jacobson), cubre todo el ciclo de vida de un proyecto desde su inicio o conceptualización hasta la transición final con el cliente.

Rational Rose nos permite establecer una trazabilidad real entre los modelos construidos y el código ejecutable. Se encuentra entre las herramientas CASE más técnicas debido a que se encarga de llevar a cabo la realización de los distintos diagramas para la posterior generación de código, como para labores de ingeniería inversa(es decir, realización de los diagramas una vez conocido el código).

Es un entorno de modelado que permite generar código en los siguientes lenguajes C++, C++, CORBA, Java/J2EE, Visual C++ y Visual Basic. Al igual que todos los productos de Rational Rose, ofrece un lenguaje de modelado (UML) común que agiliza la creación del software.

1.6.4. Herramienta CASE utilizada para el modelado de la solución

Entre las herramientas CASE analizadas anteriormente se determinó utilizar Rational Rose Enterprise Edition 2003 teniendo en cuenta que:

Es el producto más completo de la familia Rational Rose desarrollado por los creados de UML (lenguaje de modelado escogido para modelar la solución), por lo que es una herramienta óptima para utilizar eficientemente dicho lenguaje y construir los distintos modelos de la solución, además cubre todo el ciclo de vida de RUP (metodología seleccionada para el desarrollo de la solución) desde el inicio del proyecto hasta la transición final con el usuario. Otro aspecto que se tuvo en cuenta es la familiaridad que existe con dicha herramienta, debido al poco tiempo que se contaba para realizar la solución propuesta.

1.7. Patrones de Diseño

Un patrón de diseño es una solución a un problema de diseño no trivial que es efectiva y reutilizable. Un patrón de diseño identifica: clases, instancias, roles, colaboraciones y la distribución de responsabilidades.

1.7.1. Patrones GRASP (General Responsability Asignment Software Paterns)

"Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones" [11].

Entre el conjunto de patrones GRASP tenemos:

Experto: Expresa simplemente la "intuición" de que los objetos hacen cosas relacionadas con la información que poseen.

Creador: El propósito fundamental de este patrón es definir quien es la clase encargada de crear las instancias de otra clase.

Bajo acoplamiento: Evita las dependencias entre las clases y ofrece una mayor reutilización, las responsabilidades se asignan de modo que se mantenga el bajo acoplamiento.

Alta cohesión: Establece que se asignen las responsabilidades de modo que se mantenga una alta cohesión, se logra que las responsabilidades colaboren para producir un comportamiento bien definido.

Controlador: Se encarga de asignar a una clase la responsabilidad de administrar un evento del sistema.

Polimorfismo: Asignar la responsabilidad del comportamiento a las clases en que varia dicho comportamiento, mediante operaciones polimórficas.

Fabricación pura: Busca brindar soporte a: una alta cohesión, un bajo acoplamiento y una gran reutilización, mediante la asignación de un conjunto de responsabilidades a una clase artificial que no representa nada en el dominio del problema.

Indirección: Se busca beneficiar el bajo acoplamiento, asignando a un objeto la responsabilidad para que medie entre otros componentes y servicios.

No hables con extraños: Obtener la visibilidad entre objetos directos (son del conocimiento del cliente) y objetos indirectos (no son del conocimiento del cliente), de modo que el cliente no necesite conocer el objeto indirecto.

Patrones GRASP utilizados en la solución

Para el desarrollo de la solución propuesta se tuvo en cuenta fundamentalmente la utilización de los siguientes patrones: experto, alta cohesión y bajo acoplamiento.

El patrón experto se aplicó en cada clase definida para la solución, las responsabilidades de las clases dependen de la información que poseen, permitiendo que cada clase sea experta en realizar las tareas para las cuales se definieron sus responsabilidades. La aplicación del patrón alta cohesión permitió que cada clase fuera responsable de lograr tareas bien definidas en aéreas funcionales específicas, las clases contienen el número de responsabilidades que necesitan ni más ni menos, de esta manera se logra un comportamiento muy bien definido para cada clase utilizada en el desarrollo de la solución. El patrón bajo acoplamiento disminuyó las dependencias, aumentado así la reutilización de las clases diseñadas (principalmente las clases diseñadas para el manejo del sitio de transferencia).

1.7.2. Patrones GOF (Gang of Four)

Según su propósito los patrones GOF se clasifican en: creacionales, estructurales y de comportamiento.

Patrones creacionales

Los patrones creacionales se encargan de la creación de los objetos de las clases, dentro de este grupo tenemos: abstract factory (fábrica abstracta), builder (constructor virtual), factory method (método de fabricación), prototype (prototipo) y singleton (instancia única).

Patrones estructurales

Se basan en las relaciones entre clases, las combinan y forman estructuras mayores. Tratan de conseguir que los cambios en los requisitos de la aplicación no ocasionen cambios en las relaciones entre los objetos. En este grupo se encuentran los siguientes patrones: adapter (adaptador), bridge (puente), composite (objeto compuesto), decorator (envoltorio), facade (fachada), flyweight (peso ligero) y el patrón proxy.

Patrones de comportamiento

Plantean la interacción y cooperación entre las clases y objetos, en este caso tenemos los siguientes patrones: chain of responsibility (cadena de responsabilidad), command (orden), interpreter (intérprete), iterator (iterador), mediator (mediador), memento (recuerdo), observer (observador), state (estado) y template method (método plantilla).

Patrones GOF utilizados en la solución

En la solución propuesta se utilizó en primer lugar el patrón de creación singleton, el cual está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. En la solución cada clase gestora para las operaciones tiene implementado este patrón, garantizando el acceso global a dichas clases y que solo se construya un único objeto para manejar las operaciones realizadas por la solución.

Además en la solución se utilizó el patrón estructural facade o fachada, el cual nos brinda una interfaz unificada que representa a todo un subsistema, en la solución cada subsistema o capa tiene su propia fachada, reduciéndose el número de objetos con los que trata el cliente y facilitándose de esta manera el uso de cada subsistema, además con el uso de este patrón se reducen las dependencias entre los subsistemas.

1.8. Estilos y patrones de arquitectura

En la actualidad existen cientos de definiciones de Arquitectura de Software pero desde el 2001 hay una definición que ha sido la más aceptada.

Arquitectura – IEEE 1471-2000. “La arquitectura de software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución”.

“Un estilo describe entonces una clase de arquitectura, o piezas identificables de las arquitecturas empíricamente dadas. Esas piezas se encuentran repetidamente en la práctica, trasuntando la existencia de decisiones estructurales coherentes” [12]. Algunos de los estilos arquitectónicos más difundidos son los que se muestran a continuación:

Estilos de Flujo de Datos

- Tubería y filtros

Estilos Centrados en Datos

- Arquitecturas de pizarra o repositorio

Estilos de llamada y retorno

- Model-View-Controller (MVC)

- Arquitectura en capas
- Arquitecturas orientadas a objetos
- Arquitecturas basadas en componentes

Estilos de código móvil

- Arquitectura de máquinas virtuales

Estilos heterogéneos

- Sistemas de control de procesos
- Arquitecturas basadas en atributos

Estilos Peer-to-Peer

- Arquitecturas basadas en eventos
- Arquitecturas orientadas a servicios
- Arquitecturas basadas en recursos

Frank Buschmann define los **patrones arquitectónicos** como sigue: “Un problema particular recurrente del diseño que surge en un contexto de diseño específico y presenta esquema genérico bien probado para la solución de dicho problema. Esta solución es especificada describiendo sus componentes constitutivos las responsabilidades de estos componentes, sus relaciones y la forman en que colaboran” [13].

A continuación se muestra un listado de algunos de los patrones arquitectónicos de acuerdo a Frank Buschmann:

Layers (Capas): permite estructurar aplicaciones, las cuales pueden estar descompuestas en grupos de subtareas donde cada grupo está en un nivel particular de abstracción.

Pipes and filtres (Tuberías y filtros): modelan una estructura para los sistemas que procesan una corriente de datos.

Blackboard (Pizarrón): se aplica para problemas cuya solución utiliza estrategias no determinísticas. Varios subsistemas ensamblan su conocimiento para construir una posible solución parcial ó aproximada.

Broker: es usado para estructurar sistemas de software, distribuidos con componentes desacoplados que interactúan por invocaciones a servicios remotos.

Model View Controller (Modelo-Vista-Controlador MVC): divide una aplicación interactiva en tres componentes. El modelo contiene los datos y funcionalidades de fondo. La vista exhibe información para el usuario. Los controladores manipulan las entradas del usuario. La vista y el controlador conjuntamente comprenden la interfaz de usuario.

1.8.1. Arquitectura de la solución

El desarrollo de la solución propuesta responde a un modelo basado en capas, donde cada una de las capas se alimenta de las funciones que le brinda la capa inmediatamente inferior. A continuación se muestra una imagen que responde a la interacción de las capas utilizadas en el sistema.

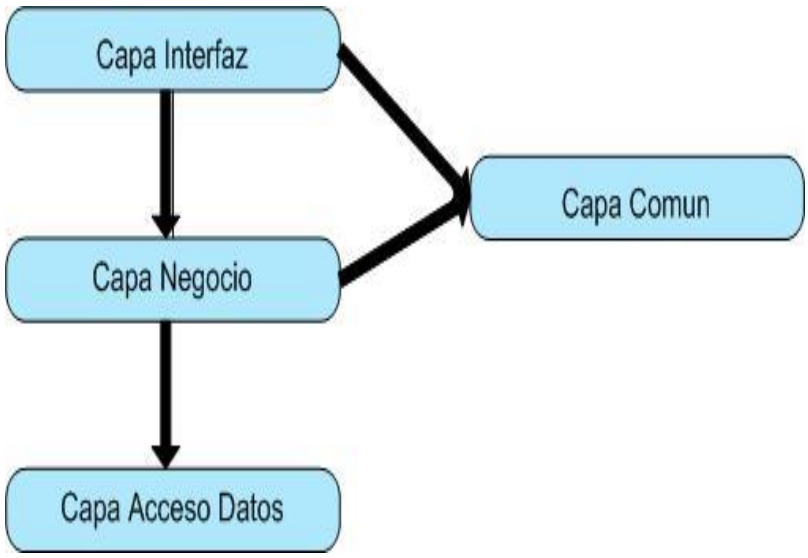

Figura 5: Modelo de capas

En la figura el sentido de las flechas indica la relación entre las capas del modelo, la capa que se encuentra en el origen de la flecha conoce la capa que está en el destino de la misma.

La capa interfaz está basada en los componentes de interfaz que brinda el framework de trabajo facilitando así el desarrollo del sistema. En el caso de la capa de Negocio, su diseño depende específicamente del negocio en cuestión a tratar. La capa común está compuesta por un conjunto de clases que sirven de ayuda o de soporte a la capa de interfaz y de negocio. Por último la capa de acceso a datos, para su creación se utilizó el conjunto de clases ADO.NET incluidas en el framework de trabajo. Esta capa está compuesta principalmente por un conjunto de clases que se encargan de todo el manejo de la información que se debe persistir así como la conexión hacia la base de datos en cuestión.

Para el manejo de la base de datos se acordó utilizar el sistema de gestión de bases de datos (SGBD) Microsoft SQL Server 2000, el cual está basado en el lenguaje Transact- SQL (Structured Query Language). Permite trabajar en modo cliente/servidor, donde la información es almacenada en el servidor y las PC clientes sólo acceden a ella.

La solución propuesta utilizará dicho SGBD para almacenar la información de forma ordenada y transparente a los usuarios. Para la selección del SGBD se tuvo en cuenta la facilidad de uso y la seguridad que brinda, además la solución no manejará volúmenes excesivos de información, así como, tampoco se necesitarán mayores exigencias.

1.9. Plataformas de desarrollo para la construcción de software.

En el presente epígrafe se hace un análisis de las tendencias y tecnologías más utilizadas a nivel mundial en el desarrollo de aplicaciones de escritorio.

1.9.1. Microsoft .NET

Microsoft .NET es un proyecto de Microsoft que ofrece un desarrollo rápido y económico de aplicaciones de software, permite desarrollar soluciones seguras, robustas y que se integren de manera ágil entre las empresas. Es considerado como una respuesta a la plataforma Java de Sun Microsystems.

Los principales componentes del marco de trabajo son:

- El conjunto de lenguajes de programación
- La Biblioteca de Clases Base o **BCL**
- El Entorno Común de Ejecución para Lenguajes o **CLR**

Algunos de los lenguajes desarrollados para el marco de trabajo .NET son: C#, Visual Basic, Delphi (Object Pascal), C++, J#, Perl, Python, Fortran, Cobol y PowerBuilder.

.NET Framework o marco de trabajo

"El Framework de .Net es una infraestructura sobre la que se reúne todo un conjunto de lenguajes y servicios que simplifican enormemente el desarrollo de aplicaciones" [14].

Actualmente, el framework de .Net no es incluido en los diferentes sistemas operativos distribuidos por Microsoft, por lo que es necesaria su instalación previa para poder ejecutar aplicaciones construidas con Microsoft .NET.

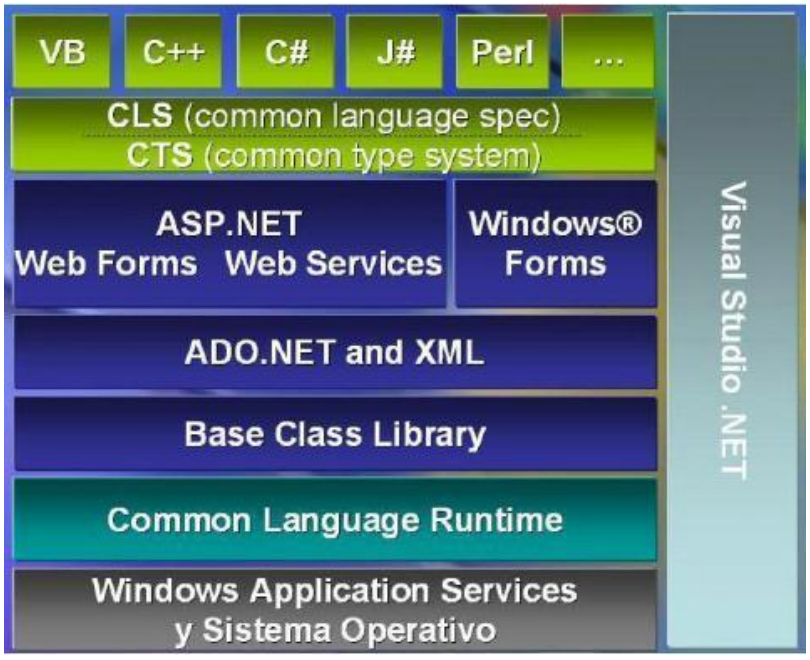

Figura 6: Diagrama detallado del Framework .NET

1.9.2. Plataforma de desarrollo MONO

En el 2001 nace el proyecto MONO dirigido por Miguel de Icaza como una implementación de código abierto de la plataforma .NET, su objetivo era simple, proporcionar una plataforma de software libre que facilitara el desarrollo de aplicaciones para Linux.

Mono implementa las siguientes partes de la tecnología .NET:

- Entorno común de ejecución para lenguajes.
- Compilador/Desensamblador lenguaje intermedio.
- Compilador C#.
- Compilador Visual Basic.NET.
- Librería de clases.
- Otras librerías de funcionalidades.

Con Mono se pueden escribir aplicaciones en múltiples lenguajes de programación, incluyendo entre ellos Python, Object Pascal, Nermele, y C#. Los programas construidos por la plataforma MONO se ejecutan sobre un entorno controlado de ejecución conocido como máquina virtual.

1.9.3. Java 2 Enterprise Edition (Java 2EE)

Java 2EE creado y distribuido por Sun Microsystems, comprende un conjunto de funcionalidades que permiten desarrollar y ejecutar aplicaciones escritas en el lenguaje de programación Java.

Arquitectura de J2EE

J2EE está basado en la arquitectura del lado del servidor la cual concentra la mayoría de los procesos de la aplicación en el servidor o en una parte de este. Este tipo de arquitectura requiere una separación entre la capa cliente y la capa servidor, permitiendo que una aplicación soporte a la vez distintos tipos de clientes o interfaces. Además las aplicaciones construidas pueden ser distribuidas en múltiples procesadores, aumentando de esta manera su escalabilidad, confiabilidad, disponibilidad y recuperabilidad.

La plataforma J2EE contiene un conjunto de librerías, interfaces y frameworks que brindan una infraestructura para la construcción de aplicaciones empresariales, entre las principales empresas que desarrollan para la plataforma se encuentran Sun Microsystems, IBM (International Business Machines), Oracle, Apache entre otras, las cuales aportan un gran número de componentes y herramientas que agilizan el trabajo de los desarrolladores en los proyectos.

1.9.4. Plataforma de desarrollo para la construcción de la solución

Para la construcción de la solución propuesta se seleccionó la plataforma Microsoft .NET especialmente Microsoft Visual Studio .NET 2005 y como lenguaje de programación Microsoft Visual C# .NET, teniendo en cuenta su familiaridad, facilidad de uso y su gran productividad, aspectos fundamentales debido al poco tiempo con el que se cuenta para la entrega de la aplicación.

1.10. Paradigmas de programación

Robert Floyd definió a los paradigmas de programación “como un proceso de diseño que va más allá de una gramática, reglas semánticas y algoritmos, sino que es un conjunto de métodos sistemáticos aplicables en todos los niveles del diseño de programas” [15].

Es una colección de modelos conceptuales que juntos modelan el proceso de diseño y determinan, al final, la estructura de un programa. Entre los tipos de paradigmas encontramos: programación descriptiva, programación imperativa, programación orientada a objetos, programación orientada a aspectos, programación orientada a agentes, programación dirigida por eventos, programación lógica y programación funcional. Todos tienen sus ventajas y sus desventajas, además existen situaciones donde un paradigma resulta más apropiado que otro.

Para el desarrollo de la solución propuesta se utilizó en algunas situaciones la programación dirigida por eventos, teniendo en cuenta que era necesario que el sistema respondiera a determinado eventos generados en su funcionamiento.

También se empleó la programación orientada a objetos, este paradigma permite el desarrollo modular de un software lo que posibilita la reutilización de estos módulos incluso en otros proyectos. Además brinda una mejor comprensión del código escrito, debido a que la información que componen a los objetos se agrupa en clases, las cuales se corresponden con

la información que trata el programa. Otro aspecto importante es que gran parte de los lenguajes más utilizados hoy en día están orientados a objetos, entre ellos se encuentran: Java, C++, PHP, Python, entre otros, incluyendo C# (lenguaje utilizado en el desarrollo del sistema).

1.11. Seguridad e integridad de los archivos transferidos por la solución propuesta.

La seguridad de los datos transferidos por la solución se garantiza mediante la utilización del protocolo FTP sobre SSL en su mecanismo explícito, es decir, utilizar el FTPS explícito como protocolo destinado para la transferencia de archivos, siendo este protocolo el responsable de brindar todos los mecanismos de seguridad requeridos para llevar a cabo un proceso de transferencia de archivos de forma confiable y segura.

Otro aspecto que fortalece la seguridad y confiabilidad del proceso de transferencia de archivos llevado a cabo por la solución propuesta, es la utilización de una VPN (Virtual Private Network-Red Privada Virtual) entre las entidades gubernamentales y el SAREN. Las VPN garantizan que todo el tráfico existente entre los puntos de comunicación sea privado evitando así que terceras personas tengan acceso a la información transferida. Además permiten que se lleve a cabo una transferencia transparente y segura como si los usuarios estuvieran conectados de forma local.

Para comprobar la integridad de los archivos transferidos por la solución se hará uso de un Hash, es decir, un código resumen que identifique de forma unívoca el archivo transferido. Dicho código se generará a partir de la función o algoritmo SHA1 (Secure Hash Algorithm), el cual es un algoritmo diseñado para producir un valor Hash de 160 bits. Todos los hash generados por un algoritmo hash tienen el mismo tamaño sea cual sea el mensaje utilizado como entrada.

Antes de subir el archivo al sitio de transferencia se calcula su hash mediante el algoritmo SHA1, cuando se concluye la transferencia se verifica la integridad del archivo aplicando el mismo algoritmo al archivo transferido. Si el archivo no ha sufrido modificaciones en el transcurso de la transferencia los códigos hash calculados deberán coincidir, en caso de no coincidir los códigos, se cancela el procesamiento del archivo transferido.

De esta manera la solución propuesta garantiza que el proceso de transferencia de archivos se realice de forma segura y que se compruebe la integridad de los archivos transferidos.

1.12. Descripción de las librerías utilizadas para el desarrollo del sistema.

En el desarrollo de la solución propuesta se utilizaron un conjunto de librerías, las cuales ayudaron resolver algunos temas como son: la conexión y la transferencia de archivos desde el servidor FTPS, así como para la manipulación del contenido de los archivos transferidos por la solución.

1.12.1. Librería libcurl

La librería LibcURL permite la programación de aplicaciones que necesitan el uso de protocolos de red para su funcionamiento. Es libre y soporta diversos protocolos de transferencia como: FTP, FTPS, HTTP, HTTPS, SFTP y otros más. Es altamente portable se construye y trabaja en diversas plataformas. Desarrollada por Haxx un grupo de 4 desarrolladores expertos que trabajan como consultores y ofrecen soluciones rápidas a difíciles problemas. Los integrantes de Haxx son: Daniel Stenberg, Linus Nielsen Feltzing, Kjell Ericson, Björn Stenberg.

La estructura de programación de libcurl se puede resumir en lo siguiente:

- Creación de un contexto: es necesario crear un contexto de libcurl que se utilizará para realizar las operaciones de red que sean necesarias.
- Configuración del contexto: mediante opciones, se debe configurar el contexto creado.
- Ejecución del contexto: ejecutar el contexto ya configurado.
- Tareas posteriores: tras la ejecución del contexto se pueden realizar otras tareas como resultados de la operación realizada, estadísticas de la red.

1.12.2. Librería LibCurlNet 1.3

Es una librería de código abierto construida para clientes de Microsoft .NET y C#, encapsula las funcionalidades de libcurl en su versión 7.13.0.0, por lo que soporta los diversos protocolos de transferencia de archivos fue desarrollada por Jeff Phillips. Es la librería principal que se utiliza en el desarrollo de la solución propuesta para resolver los aspectos relacionados con la transferencia de archivos desde el servidor FTPS.

1.12.3. Librería DotNetZip 1.7

DotNetZip es una librería de clases de código abierto pequeña pero de uso fácil para la manipulación de archivos con extensión ".zip", permite que se utilice en aplicaciones escritas en C# o en cualquier lenguaje .NET, además permite crear, leer y actualizar fácilmente archivos ".zip". Dicha librería se utilizó en la solución propuesta para resolver los temas relacionados con en el procesamiento de los archivos transferidos.

1.13. Conclusiones

En este capítulo se realizó un análisis sobre las tendencias actuales en la gestión del proceso de transferencia de archivos de forma segura, analizando los principales protocolos de transferencia de archivos y un conjunto de software utilizados a nivel mundial para llevar a cabo la transferencia de archivos. Además se analizaron algunas de las metodologías existentes para el desarrollo de software así como diferentes tecnologías para la construcción y modelado del software. A partir de este análisis se adoptaron las siguientes conclusiones:

- Utilizar la plataforma Microsoft .NET para el desarrollo de la solución propuesta, especialmente Microsoft Visual Studio .NET 2005 y como lenguaje de programación Microsoft Visual C# .NET.
- Usar RUP como metodología para el desarrollo de la solución propuesta.
- Emplear la herramienta CASE Rational Rose Enterprise Edition 2003 para crear los diferentes modelos del sistema, utilizando el lenguaje de modelado UML.
- Utilizar como paradigmas de programación la programación orientada a objetos y la programación dirigida por eventos.

Capítulo 2: Descripción de la solución propuesta

2.1. Introducción

En el presente capítulo se comienza con el diseño e implementación de la solución propuesta, se especifica cada una de las clases utilizadas, así como, el mecanismo empleado para el tratamiento de errores y excepciones, además se mencionan algunas de las reglas utilizadas en la codificación de la solución propuesta.

2.2. Modelación del negocio

A continuación se muestra una descripcion de los principales procesos del negocio así como el modelo de los casos de usos del negocio y el diagrama de actividades de dichos casos de uso.

2.2.1. Procesos del negocio actual

Proceso solicitar servicio de registro legal

El cliente se dirige al registro o notaría (donde es atendido por un funcionario del registro o notaría) para solicitar un trámite legal, presentando su documento de identidad y sus datos filiatorios.

El funcionario procede a verificar los datos filiatorios presentados por el cliente y los chequea contra su registro de clientes, si no existe diferencia procede a realizar la presentación y ahí termina este proceso, si existen diferencias el funcionario le informa al cliente los pasos a seguir para la actualización oficial de sus datos, los cuales son:

1. Dirigirse al organismo competente para que actualice sus datos personales, en este caso la ONIDEX.
2. Presentar constancia que le entrega dicho organismo ante el soporte SAREN para que actualice dicha información.

Proceso incluido (obligatorio) verificar los datos filiatorios

El funcionario realiza la búsqueda de los datos filiatorios del cliente en el registro de clientes, si encuentra dichos registro, verifica cada uno de los datos filiatorios del cliente, si existe algún

tipo de anomalía en los datos entonces crea y registra en un documento de errores filiatorios cada anomalía encontrada y declara este proceso insatisfactorio, si no encuentra anomalía alguna lo declara satisfactorio. Si en la búsqueda no encuentra el registro entonces crea y registra en un documento de errores filiatorios la anomalía encontrada y declara el proceso insatisfactorio.

Proceso actualizar datos filiatorios

El cliente se dirige al soporte SAREN y presenta el documento de constancia emitido por el organismo competente en este caso la ONIDEX en el cual vienen reflejados los cambios en los datos filiatorios del cliente, el soporte SAREN verifica el contenido y el pie de firmas del documento, garantizando así la seguridad y legalidad del mismo. Una vez verificado correctamente procede a actualizar el registro de clientes, notifica al cliente de su actualización y además le indica que puede dirigirse al registro o notaría nuevamente para realizar su solicitud de servicio de registro legal.

Si el documento al verificarse presenta problemas se le notifica al cliente de los errores y devuelve al cliente para que lo presente nuevamente a la ONIDEX para su corrección.

Actores del negocio

- Cliente

Trabajadores del Negocio

- Funcionario de SAREN
- Funcionario de registros y notarías

Entidades del Negocio

- Cédula de identidad
- Datos filiatorios
- Constancia
- Registro de clientes

- Documento de errores filiatorios

Casos de Uso (CU) del Negocio

- Solicitar servicio de registro legal
- Verificar los datos filiatorios
- Actualizar datos filiatorios

Caso de uso del negocio	Solicitar servicio de registro legal
Actores	Cliente
Resumen	El CU se inicia cuando el cliente se dirige al registro o notaría para solicitar el registro legal de algún documento.
Casos de uso asociados	Verificar los datos filiatorios
Flujo normal de eventos	
Acción del actor	**Respuesta del proceso de negocio**
1. El cliente solicita un servicio de registro legal.	
2. El cliente presenta su cédula de identidad y sus datos filiatorios.	3. El funcionario de registros y notarías verifica los datos filiatorios del cliente. **Ver caso de uso incluido "Verificar los datos filiatorios".** 4. El funcionario de registros y notarías realiza la presentación del trámite. Terminando así el caso de uso.
Flujos alternos	

Flujo alterno 1 “Datos filiatorios incorrectos”	
Acción del actor	**Respuesta del proceso de negocio**
	1. El funcionario de registros y notarías informa al cliente que debe actualizar sus datos filiatorios.
Mejoras propuestas	

Caso de uso del negocio	Verificar los datos filiatorios
Actores	
Resumen	El CU es iniciado cuando en el proceso de solicitar servicio de registro legal el funcionario de registros y notarías realiza la búsqueda de los datos filiatorios del cliente.
Casos de uso asociados	Solicitar servicio de registro legal
Flujo normal de eventos	
Acción del actor	**Respuesta del proceso de negocio**
	1. El funcionario de registros y notarías realiza la búsqueda de los datos filiatorios del cliente. 2. EL funcionario de registros y notarías verifica cada dato del cliente. 3. EL funcionario de registros y notarías declara el proceso satisfactorio. Terminando así el caso de uso

Flujos alternos	
Flujo alterno 1 "No existe el registro con los datos filiatorios"	
Acción del actor	**Respuesta del proceso de negocio**
	1. EL funcionario de registros y notarías crea y registra la anomalía encontrada en el documento de errores. 2. EL funcionario de registros y notarías declara el proceso insatisfactorio. Terminando así el caso de uso.
Flujo alterno 2 "Datos filiatorios incorrectos"	
Acción del actor	**Respuesta del proceso de negocio**
	1. EL funcionario de registros y notarías crea y registra la anomalía encontrada en el documento de errores. 2. EL funcionario de registros y notarías declara el proceso insatisfactorio. Terminando así el caso de uso.
Mejoras propuestas	

Caso de uso del negocio	Actualizar datos filiatorios
Actores	Cliente
Resumen	El CU es iniciado cuando el cliente se dirige al soporte SAREN con el objetivo de realizar la actualización de sus datos filiatorios.
Casos de uso asociados	

<table>
<tr><th colspan="2">Flujo normal de eventos</th></tr>
<tr><th>Acción del actor</th><th>Respuesta del proceso de negocio</th></tr>
<tr><td>1. El cliente se dirige al soporte SAREN.</td><td></td></tr>
<tr><td>2. El cliente presenta el documento constancia del cambio en sus datos filiatorios.</td><td>3. El funcionario de SAREN verifica el contenido del documento constancia presentado por el cliente.
4. El funcionario de SAREN actualiza el registro de clientes.
5. El funcionario de SAREN notifica al cliente de dicha actualización.
6. El funcionario de SAREN informa al cliente que puede dirigirse al registro o notaria nuevamente para continuar con su trámite. Terminando así el caso de uso.</td></tr>
<tr><th colspan="2">Flujos alternos</th></tr>
<tr><th colspan="2">Flujo alterno 1 “Documento con problemas”</th></tr>
<tr><th>Acción del actor</th><th>Respuesta del proceso de negocio</th></tr>
<tr><td></td><td>1. El funcionario de registros y notarias informa al cliente de los errores en el documento.
2. El funcionario de registros y notarias devuelve al cliente el documento. Terminando así el caso de uso.</td></tr>
<tr><td>Mejoras propuestas</td><td>Eliminar al cliente como intermediario, la notificación se realiza directamente de la ONIDEX a SAREN a través de</td></tr>
</table>

	una copia del documento de constancia con los cambios realizados.

Figura 7 Modelo de CU del negocio actual

2.2.2. Realización de los casos de uso del negocio

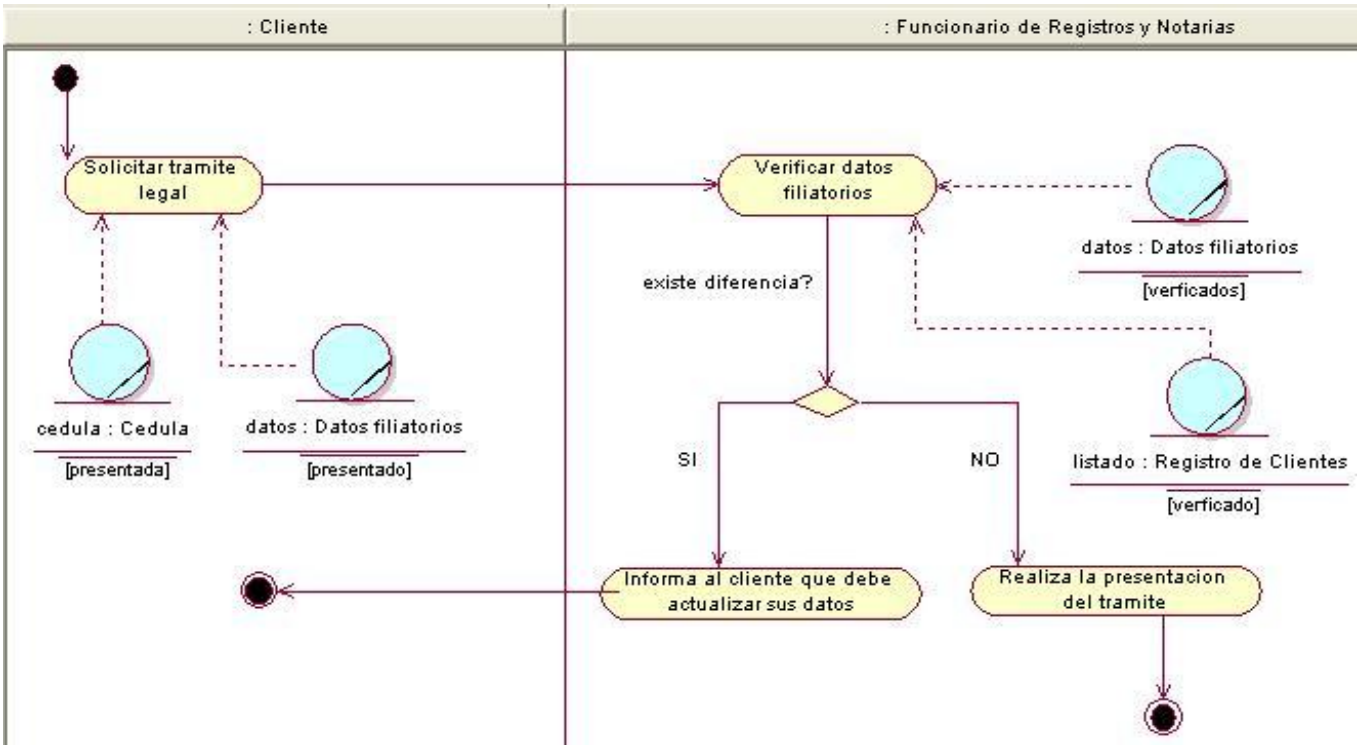

Figura 8 Diagrama de actividades: Solicitar servicio de registro legal

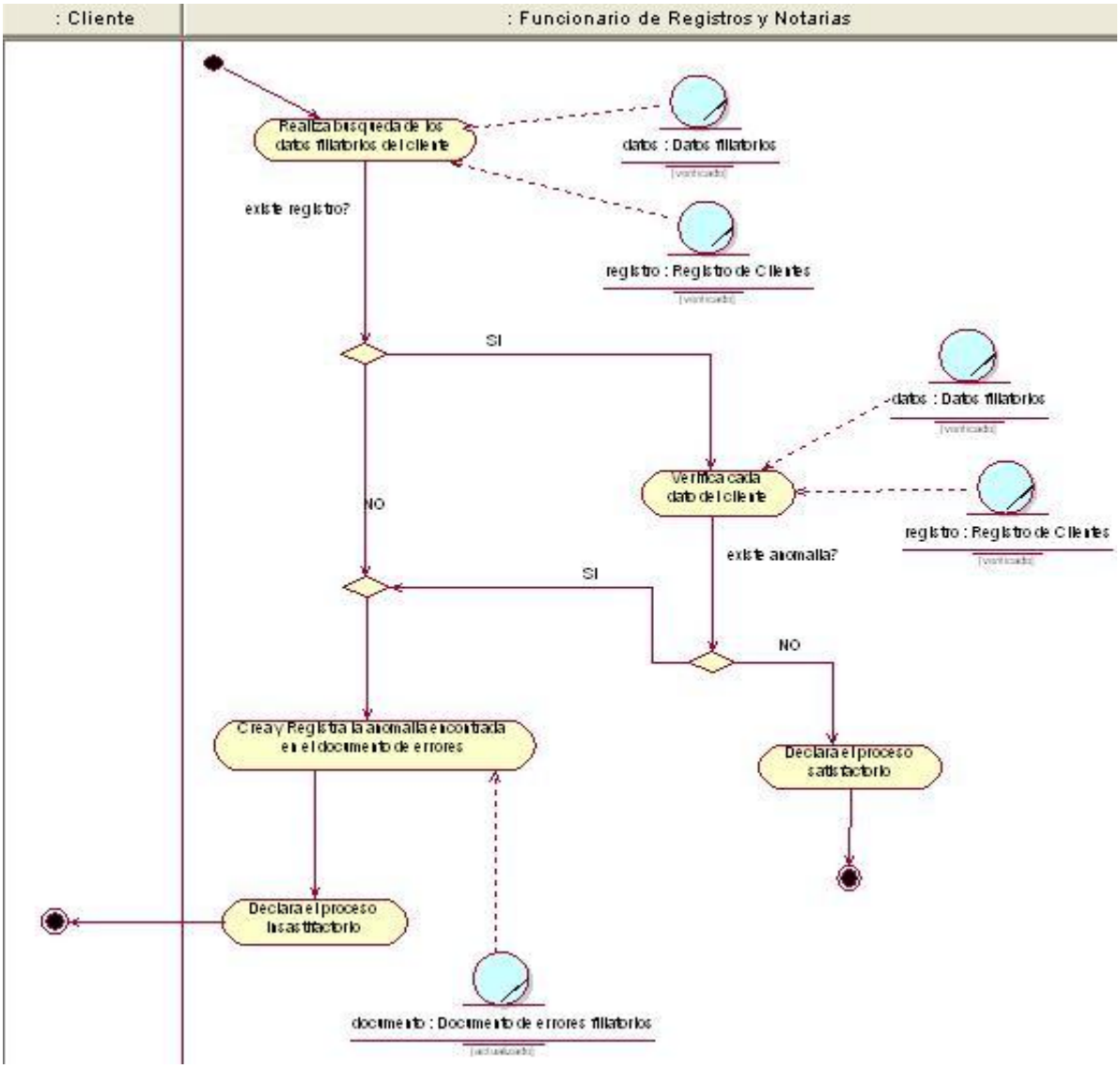

Figura 9 Diagrama de actividades: Verificar los datos filiatorios

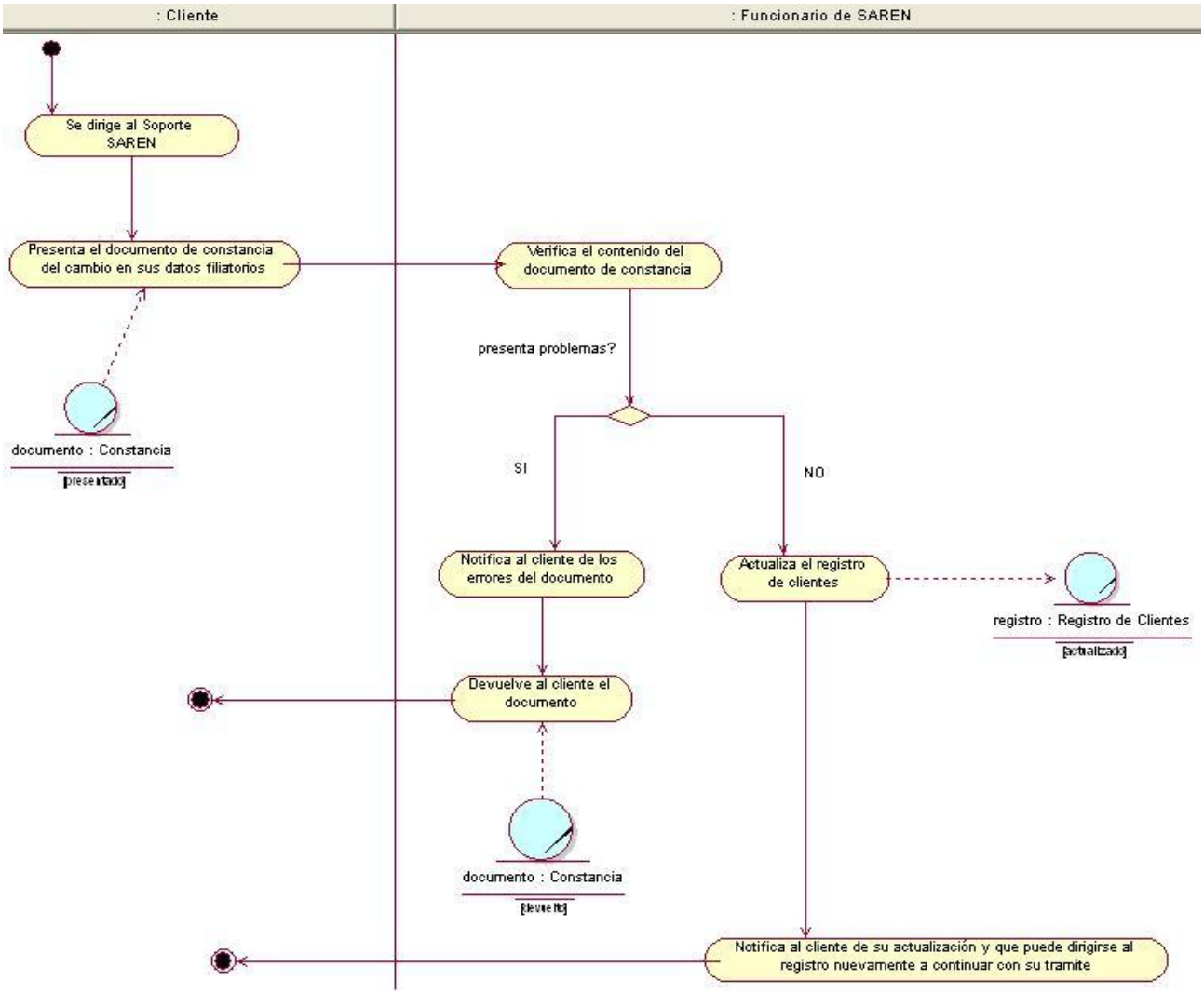

Figura 10 Diagrama de actividades: Actualizar datos filiatorios

2.2.3. Proceso de negocio mejorado

Las ventajas de este nuevo proceso mejorado consisten en agilizar el proceso de actualización de los datos filiatorios una vez declarados por la ONIDEX, dar mayor seguridad al proceso de actualización mediante la autenticidad de los datos y evitar cualquier tipo de molestias por parte del cliente pues se establece una comunicación directa entre ONIDEX y SAREN.

El proceso que sufre modificación es **actualizar datos filiatorios** donde se evita el intermedio del cliente y se notifica directamente de la ONIDEX a SAREN a través de una copia del documento de constancia con los cambios realizados.

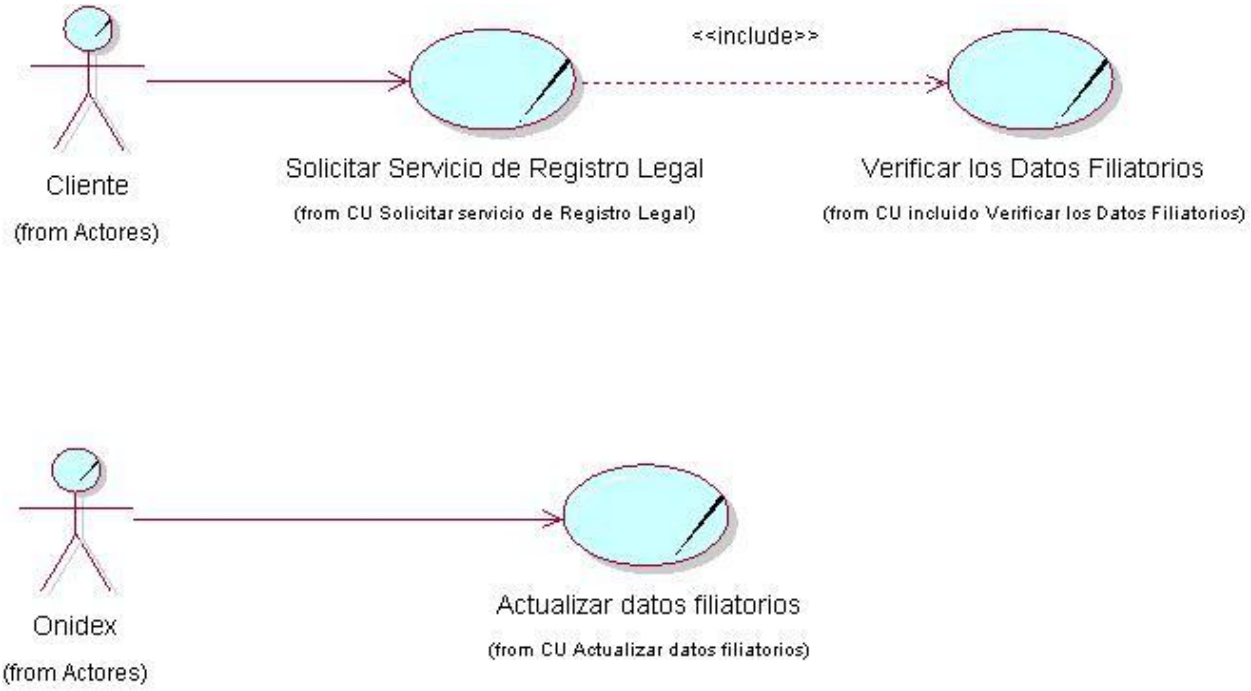

Figura 11 Modelo de CU del negocio mejorado

2.3. Modelación del sistema

En la construcción de la solución propuesta se tuvieron en cuenta los siguientes requerimientos:

2.3.1. Requerimientos funcionales

El sistema debe permitir:

1. Configurar nuevos sitios de transferencia.
2. Establecer la conexión con el sitio de transferencia.
3. Realizar la desconexión del sitio de transferencia.
4. Listar los directorios y recursos que se encuentran en el sitio de transferencia.
5. Descargar los recursos que se encuentran en el sitio de transferencia.
6. Subir un recurso al sitio de transferencia.
7. Eliminar un recurso del sitio de transferencia.
8. Procesar hacia la base datos un recurso descargado del sitio de transferencia

   8.1 Validar la consistencia del recurso a procesar.

   8.2 Configurar la conexión hacia la base datos.
9. Actualizar la información referente al sitio de transferencia.

2.3.2. Requerimientos no funcionales

Las propiedades o cualidades que el sistema debe tener son:

Requerimientos de software

- La PC cliente estará montada sobre el sistema operativo Windows XP.
- El servidor de base de datos será Microsoft SQL Server 2000.

Requerimientos de hardware

- El sistema estará montado sobre en una red WAN (Wide Area Network) corporativa y además se podrá utilizar una Red Privada Virtual (VPN).
- El sistema requiere como mínimo 128 Mb de RAM (Random Access Memory) para las estaciones de trabajo.

Requerimientos en el diseño e implementación

- Se utilizará Rational Rose Enterprise Edition 2003 como herramienta CASE para el modelado y obtención de los diferentes artefactos que requiere el software.
- El software se desarrollará sobre la plataforma Microsoft .NET y se utilizará C# como lenguaje de programación.

Requerimientos de apariencia o interfaz externa

- El sistema tendrá una interfaz gráfica uniforme incluyendo pantallas, menús y opciones.

Requerimientos de seguridad

- El software se basará en el protocolo SSL, como protocolo seguro para realizar las transferencias de archivos.
- El software hará uso de códigos Hash para comprobar la integridad de los archivos transferidos.

Requerimientos de usabilidad

- El sistema presentará sus mensajes y textos, en idioma español.
- El sistema facilitará la entrada de datos a los usuarios, presentando listas que permitan escoger valores descriptivos.

2.3.3. Modelación de los casos de uso del sistema

En la modelación de la solución propuesta se seleccionaron los siguientes actores y casos de uso:

Actor del sistema:

- Funcionario de SAREN

Casos de uso (CU) y su impacto en la arquitectura del sistema:

Caso de uso	Impacto
Conectar con el FTPS	Crítico
Listar Recursos	Crítico
Configurar Conexión con el FTPS	Crítico
Configurar Conexión con BD	Crítico
Configurar Conexión	Crítico
Desconectar con el FTPS	Crítico
Descargar un recurso	Crítico
Subir un recurso	Secundario
Borrar un recurso	Crítico
Actualizar el FTPS	Secundario
Procesar recurso	Crítico
Validar consistencia del recurso	Crítico

Descripción de los Casos de uso del sistema

Caso de Uso:	Conectar con el FTPS
Actores:	Funcionario de SAREN
Resumen:	El caso de uso inicia cuando el Funcionario de SAREN decide conectarse al servidor FTPS para realizar cualquier operación.
Referencia:	RF2, RF4.
CU asociados:	Listar Recursos y Configurar Conexión con el FTPS

Precondiciones:	El Funcionario de SAREN debe haber configurado la conexión con el FTPS.
Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Funcionario de SAREN se dirige al menú principal "Archivo" y selecciona el submenú Conectar al servidor FTP.	1.1- Ver sección Listar Recursos
	1.2- El sistema muestra un control de usuario con todos los directorios y recursos del servidor FTPS.
Sección "Listar Recursos"	
2- El Funcionario de SAREN selecciona la opción "Conectar al servidor FTP".	2.1- El sistema lista todos los directorios y recursos que contiene el servidor FTPS
Flujos Alternos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Funcionario de SAREN no configuró la conexión con el servidor FTPS	1.2- El sistema muestra un mensaje "Verifique que se haya configurado la conexión con el servidor FTP". Ir al paso **1**
Poscondiciones:	Queda conectado el sistema con el servidor FTPS.
Prioridad:	Crítico

Caso de Uso:	Desconectar con el FTPS
Actores:	Funcionario de SAREN
Resumen:	El caso de uso inicia cuando el Funcionario de SAREN decide desconectarse del sitio de transferencia.
Referencia:	RF3.
CU asociados:	
Precondiciones:	El Funcionario de SAREN debe estar conectado al FTPS.
Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Funcionario de SAREN se dirige al menú principal "Archivo" y selecciona el submenú Desconectar el servidor FTP.	1.1- El sistema elimina el control de usuario con todos los directorios y recursos del servidor FTPS.
Flujos Alternos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Funcionario de SAREN no está previamente conectado al servidor FTPS	1.2- El sistema muestra un mensaje "Debe estar conectado al servidor FTP para realizar

	esta operación". Ir al paso **1**
Poscondiciones:	Queda desconectado el sistema del servidor FTPS.
Prioridad:	Crítico

Caso de Uso:	Configurar Conexión
Actores:	Funcionario de SAREN
Resumen:	El caso de uso inicia cuando el Funcionario de SAREN decide configurar una conexión ya sea con el servidor FTPS o el servidor BD
Referencia:	RF1, RF8.2.
CU asociados:	Configurar conexión con el FTPS y Configurar Conexión con BD.
Precondiciones:	

Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Funcionario de SAREN se dirige al menú principal "Configuración".	
2- El Funcionario de SAREN selecciona uno de los submenús.	2.1- Si selecciona "Configurar conexión al servidor FTP" ver sección **Configurar conexión con el FTPS.** - Si selecciona "Configurar conexión al servidor de base de datos" ver sección **Configurar Conexión con BD.**
Sección "Configurar conexión con el FTPS"	
3- El Funcionario de SAREN selecciona la opción "Configurar conexión al servidor FTP".	3.1- El sistema muestra un formulario en el que pide llenar los datos de la conexión con el FTPS: Datos del FTP(direccionServidor, tipoFTP, puerto, usuario, contraseña), Datos del Proxy si lo hay(direccionServidor, tipo_Proxy, puerto, usuario, contraseña)
4- El Funcionario de SAREN inserta los datos a llenar.	4.1- Si todos los datos introducidos están correctos crea la conexión.
	4.2- El sistema notifica al Funcionario de SAREN de que se creó la conexión satisfactoriamente.
Sección "Configurar Conexión con BD"	
5- El Funcionario de SAREN selecciona la	5.1- El sistema muestra un formulario en el

opción "Configurar conexión al servidor de base de datos".	que pide llenar los datos de la conexión con el servidor de Base Datos(nombre Servidor, usuario, contraseña y BaseDatos)
6- El Funcionario de SAREN inserta los datos a llenar.	6.1- Si todos los datos introducidos están correctos crea la conexión.
	6.2- El sistema notifica al Funcionario de SAREN de que se creó la conexión satisfactoriamente.
Flujos Alternos	
Sección "Configurar conexión con el FTPS"	
Acción del Actor	**Respuesta del Sistema**
4- El Funcionario de SAREN deja en blanco datos de carácter obligatorio.	4.1- El sistema muestra un mensaje "Debe llenar todos los campos obligatorios". Ir al paso **3.1**
Sección "Configurar Conexión con BD"	
6- El Funcionario de SAREN deja en blanco datos de carácter obligatorio.	6.1- El sistema muestra un mensaje "Debe llenar todos los campos obligatorios". Ir al paso **5.1**
Poscondiciones:	Queda creada la configuración de la conexión
Prioridad:	Crítico

Caso de Uso:	Descargar un Recurso
Actores:	Funcionario de SAREN
Resumen:	El caso de uso inicia cuando el Funcionario de SAREN decide transferir un archivo desde el servidor FTPS hacia su ordenador.
Referencia:	RF5.
CU asociados:	
Precondiciones:	El sistema debe estar conectado al servidor FTPS y el usuario debe seleccionar previamente el recurso que desea descargar.
Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Funcionario de SAREN se dirige al menú principal "Operaciones" y selecciona el submenú "Descargar un recurso del servidor"	1.1- El sistema muestra una ventana de diálogo para que el Funcionario de SAREN seleccione la ubicación hacia donde se transferirá el recurso.
2- El Funcionario de SAREN selecciona la ubicación donde desea que se transfiera el	2.1- El sistema transfiere el recurso hacia la ubicación seleccionada.

recurso.	
	2.2- El sistema muestra un mensaje "La operación se ha realizado con éxito"
Flujos Alternos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1. El Funcionario de SAREN no está conectado al servidor FTPS	1.1- El sistema muestra un mensaje "Debe conectarse al servidor FTP para descargar". Ir al paso **1**
1- El Funcionario de SAREN no ha seleccionado el recurso a descargar.	1.1- El sistema muestra un mensaje "Debe seleccionar el recurso a descargar". Ir al paso **1**
Poscondiciones:	Se transfiere el recurso deseado a la PC cliente
Prioridad:	Crítico

Caso de Uso:	Subir un Recurso
Actores:	Funcionario de SAREN
Resumen:	El caso de uso inicia cuando el Funcionario de SAREN decide transferir un archivo desde su ordenador hacia el servidor FTPS.
Referencia:	RF6.
CU asociados:	
Precondiciones:	El sistema debe estar conectado al servidor FTPS.
Flujo Normal de Eventos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Funcionario de SAREN se dirige al menú principal "Operaciones" y selecciona el submenú "Subir un recurso al servidor"	1.1- El sistema muestra una ventana de diálogo para que el Funcionario de SAREN seleccione la ubicación del recurso a transferir.
2- El Funcionario de SAREN selecciona la ubicación del recurso.	2.1- El sistema transfiere el recurso hacia el servidor FTPS.
	2.2- El sistema muestra un mensaje "La operación se ha realizado con éxito"
Flujos Alternos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Funcionario de SAREN no está conectado al servidor FTPS	1.1- El sistema muestra un mensaje "Debe conectarse al servidor FTP para subir algún recurso al servidor FTP". Ir al paso **1**

Poscondiciones:	Se transfiere el recurso deseado al servidor FTPS
Prioridad:	Secundario

Caso de Uso:	Borrar un recurso	
Actores:	Funcionario de SAREN	
Resumen:	El caso de uso inicia cuando el Funcionario de SAREN decide eliminar un recurso del sitio de transferencia, con el objetivo de eliminar un recurso ya procesado o que debe ser descartado.	
Referencia:	RF7.	
CU asociados:		
Precondiciones:	El sistema debe estar conectado al servidor FTPS.	
Flujo Normal de Eventos		
Sección "General"		
Acción del Actor		**Respuesta del Sistema**
1- El Funcionario de SAREN se dirige al menú principal "Operaciones" y selecciona el submenú "Borrar un recurso del servidor"		1.1- El sistema elimina el recurso del sitio de transferencia.
		1.2- El sistema muestra un mensaje "La operación se ha realizado con éxito".
Flujos Alternos		
Sección "General"		
Acción del Actor		**Respuesta del Sistema**
1- El Funcionario de SAREN no está conectado al servidor FTPS		1.1- El sistema muestra un mensaje "Debe conectarse al servidor FTP para realizar esta operación". Ir al paso **1**
1- El Funcionario de SAREN no ha seleccionado el recurso a eliminar.		1.1- El sistema muestra un mensaje "Debe seleccionar el recurso a borrar". Ir al paso **1**
1- El Funcionario de SAREN selecciona un directorio a eliminar que contiene recursos, es decir, que no está vacío.		1.1- El sistema muestra un mensaje "El directorio no se puede eliminar porque no está vacío". Ir al paso **1**
Poscondiciones:	Se elimina el recurso seleccionado del sitio de transferencia	
Prioridad:	Crítico	

Caso de Uso:	Actualizar el FTPS	
Actores:	Funcionario de SAREN	
Resumen:	El caso de uso inicia cuando el Funcionario de SAREN decide actualizar el FTPS con el objetivo de comprobar alguna operación realizada en dicho FTPS.	
Referencia:	RF9.	
CU asociados:	Desconectar con el FTPS y Conectar con el FTPS.	
Precondiciones:	El sistema debe estar conectado al servidor FTPS.	
Flujo Normal de Eventos		
Sección "General"		
Acción del Actor		**Respuesta del Sistema**
2- El Funcionario de SAREN se dirige al menú principal "Operaciones" y selecciona el submenú "Actualizar el visor FTP"		1.1- El sistema actualiza el contenido mostrado proveniente del servidor FTPS.
		1.2- El sistema muestra un mensaje "La operación se ha realizado con éxito".
Flujos Alternos		
Sección "General"		
Acción del Actor		**Respuesta del Sistema**
3- El Funcionario de SAREN no está conectado al servidor FTPS		1.1- El sistema muestra un mensaje "Debe conectarse al servidor FTP para realizar esta operación". Ir al paso **1**
Poscondiciones:	Se actualiza el contenido mostrado proveniente del servidor FTPS	
Prioridad:	Secundario	

Caso de Uso:	Procesar recurso	
Actores:	Funcionario de SAREN	
Resumen:	El caso de uso inicia cuando el Funcionario de SAREN decide procesar un recurso descargado hacia la base de datos.	
Referencia:	RF8, RF8.1, RF8.2.	
CU asociados:	Validar consistencia del recurso y Configurar Conexión con BD	
Precondiciones:	El Funcionario de SAREN debe haber configurado la conexión con la Base de Datos.	
Flujo Normal de Eventos		
Sección "General"		
Acción del Actor		**Respuesta del Sistema**
1- El Funcionario de SAREN se dirige al menú principal "Operaciones" y selecciona el submenú "Procesar recurso descargado".		1.1- El sistema muestra una ventana de diálogo para que el Funcionario de SAREN seleccione el recurso a procesar.
2- El Funcionario de SAREN selecciona el		2.1- Ver sección Validar consistencia del

recurso a procesar	recurso
	2.2- El sistema muestra un mensaje "La operación se ha realizado con éxito".
Sección "Validar consistencia del recurso"	
2- El Funcionario de SAREN selecciona la opción "Procesar recurso descargado".	2.1- El sistema valida la consistencia del recurso seleccionado
Flujos Alternos	
Sección "General"	
Acción del Actor	**Respuesta del Sistema**
1- El Funcionario de SAREN no configuró la conexión con la Base de Datos	1.2- El sistema muestra un mensaje "Debe configurar la conexión con la Base de Datos". Ir al paso **1**
Sección "Validar consistencia del recurso"	
Acción del Actor	**Respuesta del Sistema**
2- El Funcionario de SAREN seleccionó un archivo que no contiene el formato de los recursos a procesar.	2.1**-** El sistema muestra un mensaje "El recurso no puede ser procesado porque se detecto alguna modificación no autorizada". Ir al paso **1**
Poscondiciones:	Queda procesado hacia la Base de Datos el recurso
Prioridad:	Crítico

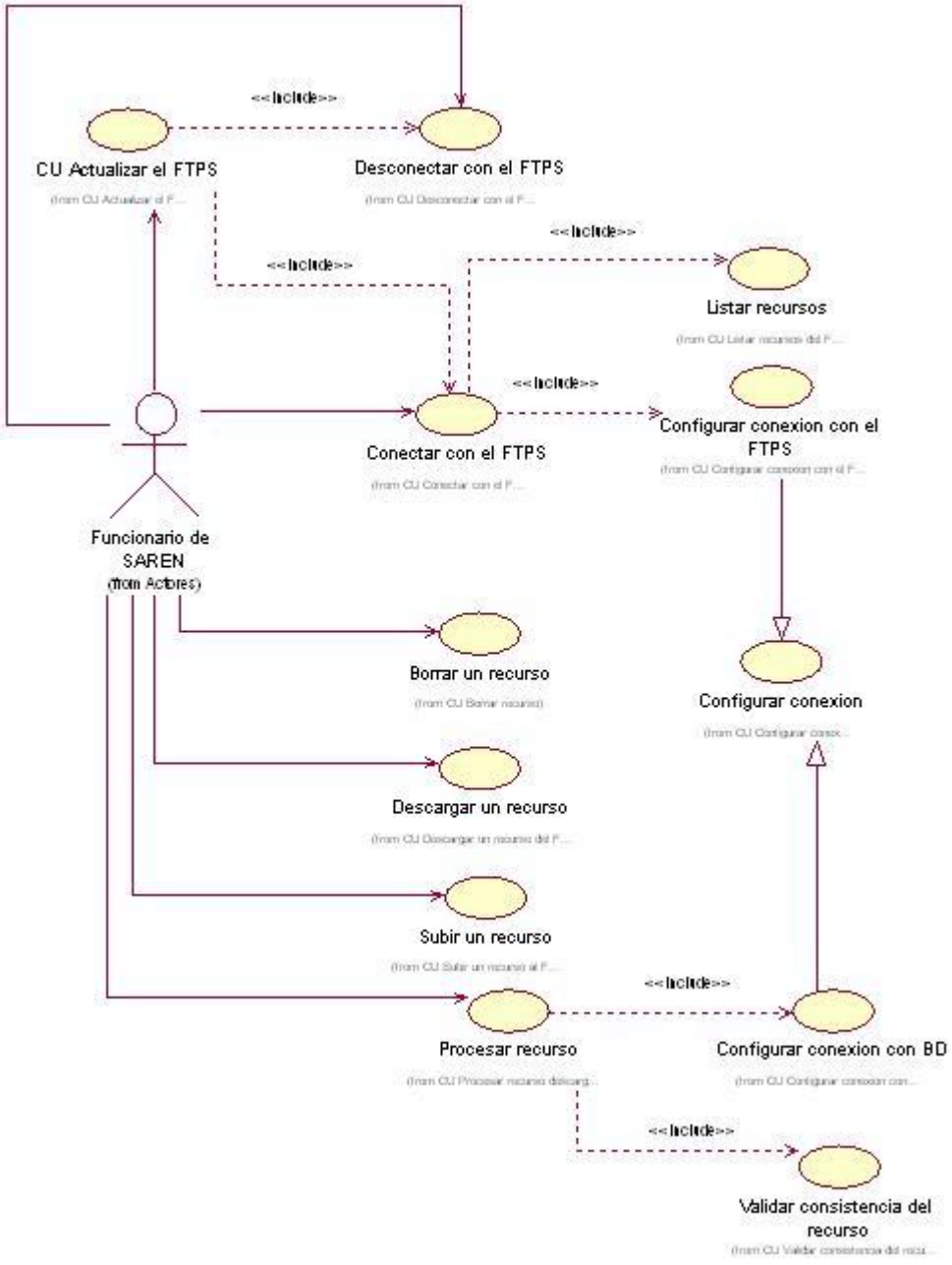

Figura 12: Modelo de CU del Sistema

2.4. Tratamiento de errores

En el sistema las excepciones se manejan de forma diferentes, en algunos casos se realizan las validaciones pertinentes y en los otros casos se capturan las excepciones lanzadas por las clases utilizadas, ya sean clases del propio framework o las clases construidas para el desarrollo de la solución propuesta. En todos los casos se lanzan mensajes haciendo uso de la clase (MessageError) definida en la solución para construir los mensajes y definir la apariencia de dichos mensajes que se le muestran al usuario en caso de algún error o excepción.

De esta manera el sistema se encontrará de forma fácil y segura controlando las excepciones así como el manejo de los mensajes. Se le mostraran al usuario los mensajes de una forma coherente y amigable que le ayudarán con el manejo de la solución.

2.5. Diseño del sistema

Antes de comenzar con el diseño, se debe aclarar que se valoró que para el desarrollo de la solución no se confeccionen los diagramas de interacción (secuencia y colaboración) que propone la metodología utilizada, teniendo en cuenta, que el sistema no es complejo y que la realización de dichos diagramas no aportarían nada significativo al desarrollo de la solución propuesta.

2.5.1. Descripción de las clases utilizadas

A continuación se detalla la descripción de cada una de las clases utilizadas en la solución propuesta:

Para el desarrollo de la solución propuesta se construyó la librería de clases **libconexion** que encapsula las funcionalidades de LibCurlNet 1.3, libconexion contiene las clases principales para la conexión y la transferencia de archivos desde el servidor FTPS. Además se construyó el **componente FTP**, el cual es un componente visual que tiene como funciones mostrar una vista del servidor FTPS para así facilitarle al cliente el uso de la aplicación y además se encarga de las distintas operaciones realizadas por la solución propuesta. El componente FTP y libconexion son los componentes principales de la capa interfaz.

Capa Interfaz

Descripción de las clases de libconexion

Nombre: FTP	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
direccionServidor	string
puerto	int
usuario	string
contraseña	string
tipo_FTP	ETipo_FTP
directorioInicial	string
Responsabilidad	
Descripción:	Clase que contiene las características de la conexión con el servidor FTP.

Nombre: Proxy	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
direccionServidor	string
puerto	int
usuario	string
contraseña	string
tipo_Proxy	CURLproxyType
Responsabilidad	
Descripción:	Clase que contiene las características de la conexión con el

	servidor Proxy.

Nombre: ETipo_FTP	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
FTP	enum
FTPS	enum
SFTP	enum
TFTP	enum
Responsabilidad	
Descripción:	Clase Enumerativa que contiene los tipos de servidor FTP con los que trabaja la solución propuesta.

Nombre: CURLproxyType	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
CURLPROXY_HTTP	enum
CURLPROXY_SOCKS4	enum
CURLPROXY_SOCKS5	enum
Responsabilidad	
Descripción:	Clase Enumerativa miembro de LibCurlNet v1.3 que contiene los tipos de servidor Proxy con los que trabaja la solución propuesta.
Nombre: Recurso	

Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
descripcion	String
Responsabilidad	
Descripción:	Clase base para los tipos de recursos del servidor FTP.

Nombre: Recurso_Hoja	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
tipo_recurso	ETipo_Recurso
Responsabilidad	
Descripción:	Clase para los recursos del servidor FTP.

Nombre: Recurso_Desplegable	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
tipo_recurso	ETipo_Recurso
Responsabilidad	
Descripción:	Clase para los directorios del servidor FTP.
Nombre: ETipo_Recurso	
Tipo de Clase: Entidad	

Atributo:	**Tipo de Atributo**
carpeta	enum
archivo	enum
Responsabilidad	
Descripción:	Clase enumerativa para los tipos de recursos del servidor FTP.

Nombre: ControlConexion	
Tipo de Clase: Gestor	
Atributo:	**Tipo de Atributo**
ftp	FTP
proxy	Proxy
existeProxy	bool
curlCode	CURLcode
Responsabilidad	
Descripción:	Clase manejadora de la conexión con el servidor FTP.

Nombre: ControlFTP	
Tipo de Clase: Gestor	
Atributo:	**Tipo de Atributo**
controlConexion	ControlConexion
listaRecursos	List<Recurso>
fs	FileStream
instance	ControlFTP
Responsabilidad	
Descripción:	Clase manejadora para las operaciones con el servidor FTP.

Descripción de las clases del componente FTP

Nombre: formPrincipal	
Tipo de Clase: Interfaz	
Atributo:	**Tipo de Atributo**
uc_VisorFTP	UC_VisorFTP
formConfigConexionFTP	formConfigurarConexionFTP
Responsabilidad	
Descripción:	Formulario principal del componente.

Nombre: UC_VisorFTP	
Tipo de Clase: Interfaz	
Atributo:	**Tipo de Atributo**
Responsabilidad	
Descripción:	Control de Usuario que muestra la vista del servidor FTP

Nombre: formConfigurarConexionFTP	
Tipo de Clase: Interfaz	
Atributo:	**Tipo de Atributo**
configuracionConexion	ConfiguracionConexion
Responsabilidad	
Descripción:	Formulario para la conexión con el servidor FTP

Nombre: FormConfiguraracionBD	
Tipo de Clase: Interfaz	
Atributo:	**Tipo de Atributo**
Responsabilidad	
Descripción:	Formulario para configurar las opciones de conexión con el servidor de Base de Datos

Nombre: ConfiguracionConexion	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
servidor_FTP	string
tipo_FTP	string
puerto_FTP	string
usuario_FTP	string
contraseña_FTP	string
existe_Proxy	string
servidor_Proxy	string
tipo_Proxy	string
puerto_Proxy	string
usuario_Proxy	string
contraseña_Proxy	string
Responsabilidad	
Descripción:	Clase serializable para crear el fichero de conexión con el servidor FTP.

Nombre: ControlOperaciones	
Tipo de Clase: Gestor	
Atributo:	**Tipo de Atributo**
configuracionConexion	ConfiguracionConexion
instance	ControlOperaciones
Responsabilidad	
Descripción:	Clase manejadora de las operaciones del componente FTP

Nombre: ConfiguracionConexionBD	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
nombreServidor	string
nombreBD	string
usuario	string
contraseña	string
Responsabilidad	
Descripción:	Clase serializable para crear el fichero de conexión con el servidor de Base de Datos

Capa negocio

Nombre: ArchivoZip	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
archivo_zip	ZipFile
directorio	string

Responsabilidad	
Descripción:	Clase para la creación del archivo .zip descargado del servidor FTP.

Nombre: ControlZip	
Tipo de Clase: Gestor	
Atributo:	**Tipo de Atributo**
archivo_zip	ArchivoZip
listaContenido	List<string>
Responsabilidad	
Descripción:	Clase manejadora para el procesamiento del archivo .zip descargado del servidor FTP.

Nombre: datos	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
letra_cedula	string
cedula_persona	string
primer_nombre	string
segundo_nombre	string
primer_apellido	string
segundo_apellido	string
fecha_nacimiento	string
fecha_cedulaOtorgada	string
codigo_objecion	string
codigo_oficina	string
codigo_estadoCivil	string
naturalizado	string

sexo	string
Responsabilidad	
Descripción:	Clase serializable utilizada para confeccionar el objeto de negocio a persistir a partir del fichero XML procesado.

Nombre: ControlDatosPersona	
Tipo de Clase: Gestor	
Atributo:	**Tipo de Atributo**
datosPersona instance	datos ControlDatosPersona
Responsabilidad	
Descripción:	Clase gestora que se encarga de leer el fichero XML procesado y de construir el objeto de la clase datos que se va a persistir.

Nombre: ControlOperacionesNegocio	
Tipo de Clase: Gestor	
Atributo:	**Tipo de Atributo**
instance	ControlDatosPersona
Responsabilidad	
Descripción:	Clase gestora que se encarga de realizar las operaciones de la capa de lógica del negocio.

Nombre: ValidarArchivo
Tipo de Clase: Gestor

Atributo:	**Tipo de Atributo**
datos	byte[]
codigoHASH	byte[]
fs	FileStream
nombreArchivo	string
instance	ValidarArchivo
Responsabilidad	
Descripción:	Clase gestora encargada de calcular el código HASH del recurso a procesar.

Capa Común

Nombre: FuncionesdeAyuda	
Tipo de Clase: Gestor	
Atributo:	**Tipo de Atributo**
instance	FuncionesdeAyuda
Responsabilidad	
Descripción:	Clase que provee funciones de ayuda para las clases que componen la capa de interfaz y de negocio.

Nombre: MessageError	
Tipo de Clase: Gestor	
Atributo:	**Tipo de Atributo**
instance	MessageError
mensaje	string
Responsabilidad	
Descripción:	Clase que define la apariencia de los mensajes que se le

	muestran al usuario en caso de algún error o excepción.

Capa de Acceso a Datos

Nombre: ConfiguracionBD	
Tipo de Clase: Entidad	
Atributo:	**Tipo de Atributo**
instance	MessageError
conexión	SqlConnectionStringBuilder
Responsabilidad	
Descripción:	Clase para configurar la cadena de conexión con la Base de Datos.

Nombre: GestorAccesoDatos	
Tipo de Clase: Gestor	
Atributo:	**Tipo de Atributo**
databaseCommand	SqlCommand
databaseConexion	SqlConnection
conexión	String
parametro	SqlParameter
instance	GestorAccesoDatos
Responsabilidad	
Descripción:	Clase gestora encargada de persistir los datos de las personas.

2.5.2. Diagrama de clases de la solución propuesta

Diagrama de clases de la capa interfaz

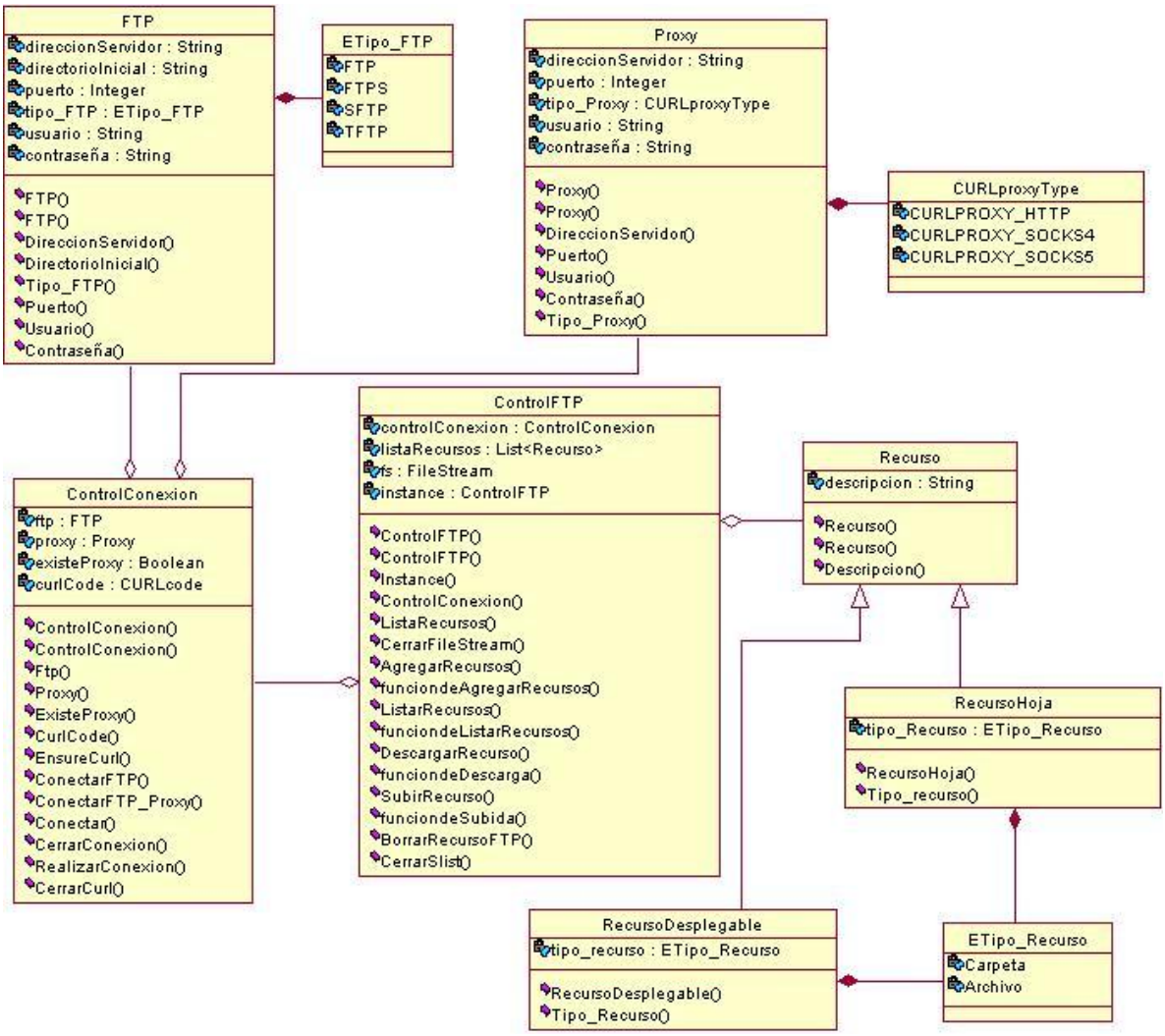

Figura 13: Paquete de clases de libconexion

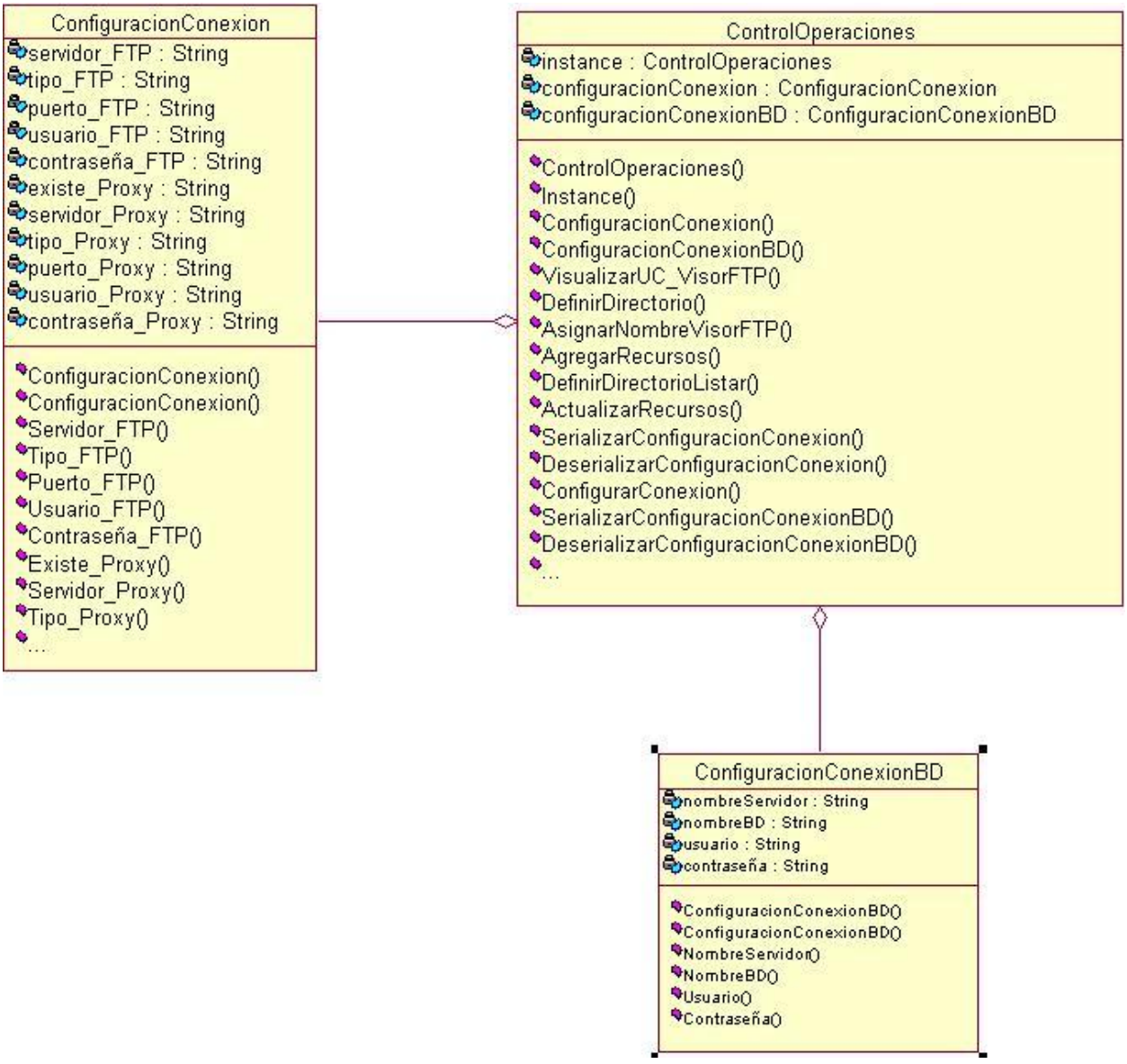

Figura 14: Paquete de clases del componente FTP

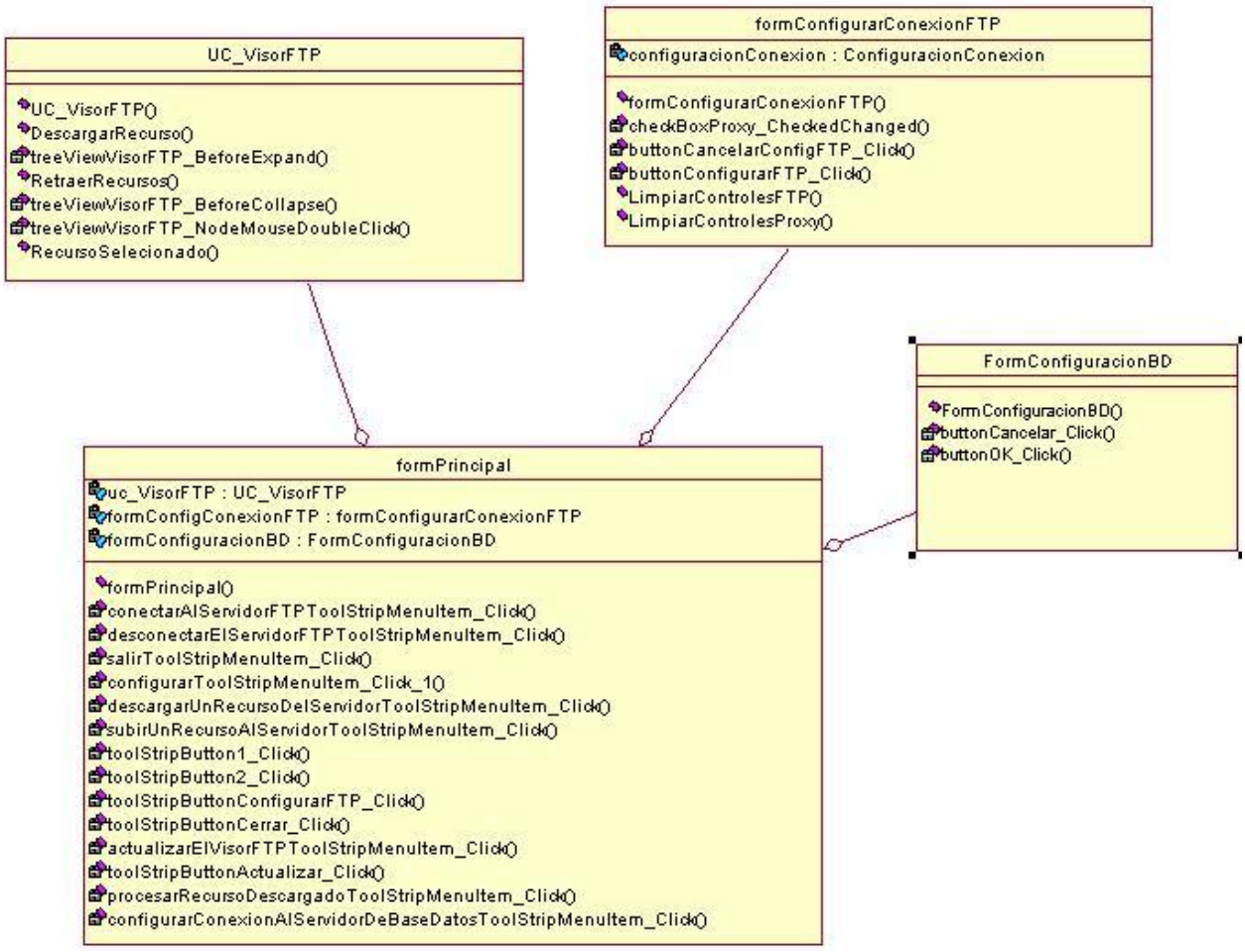

Figura 15: Vista del componente FTP

Figura 16: Paquetes del componente FTP

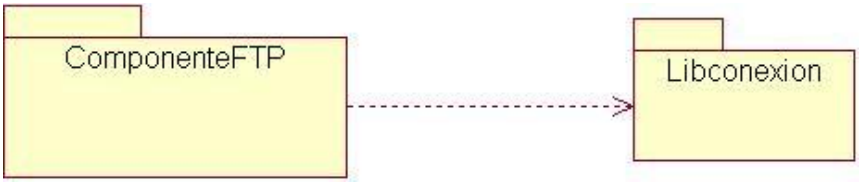

Figura 17: Paquetes de la capa de interfaz

Diagrama de clases de la capa negocio

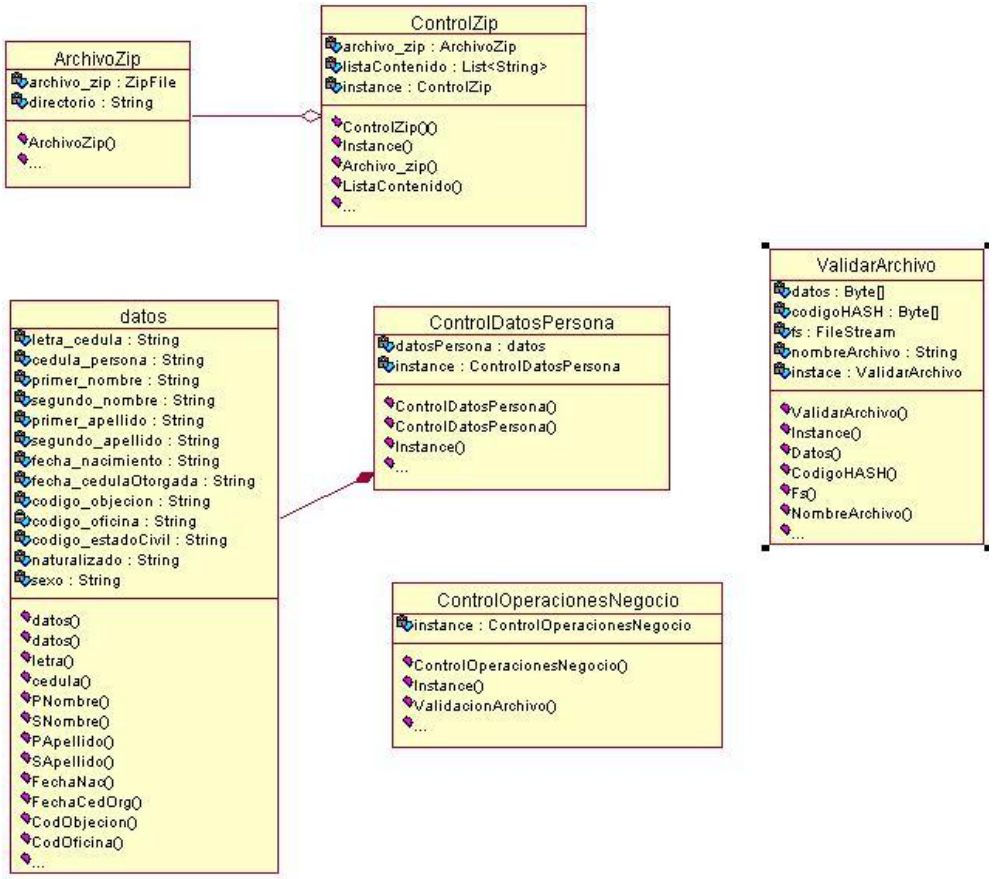

Figura 18: Capa de negocio

Diagrama de clases de la capa común

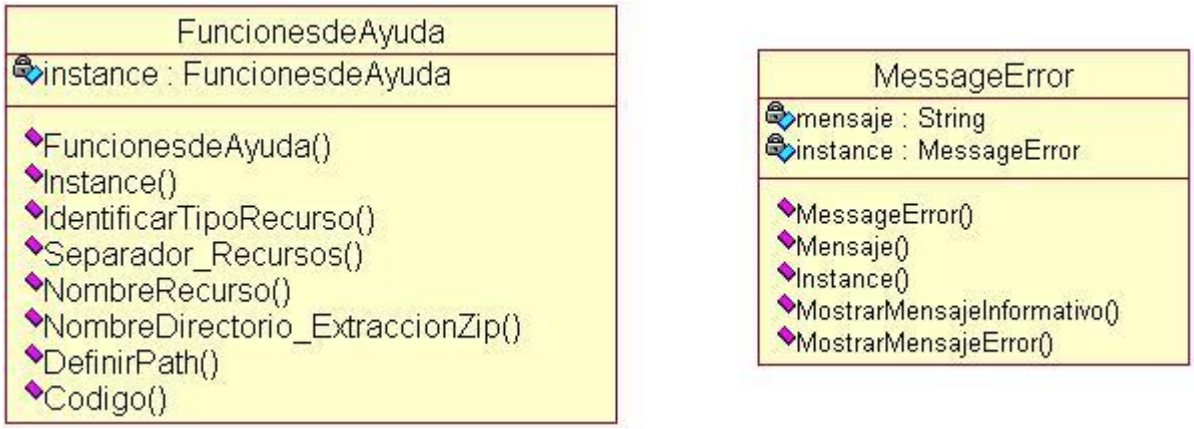

Figura 19: Capa común

Diagrama de clases de la capa acceso a datos

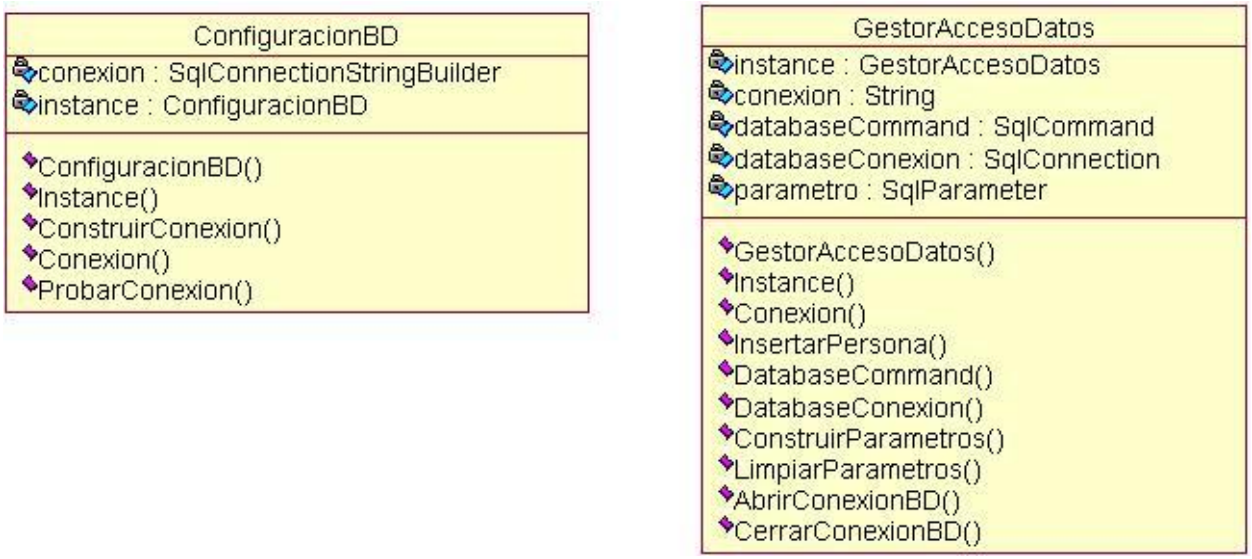

Figura 20: Capa de acceso a datos

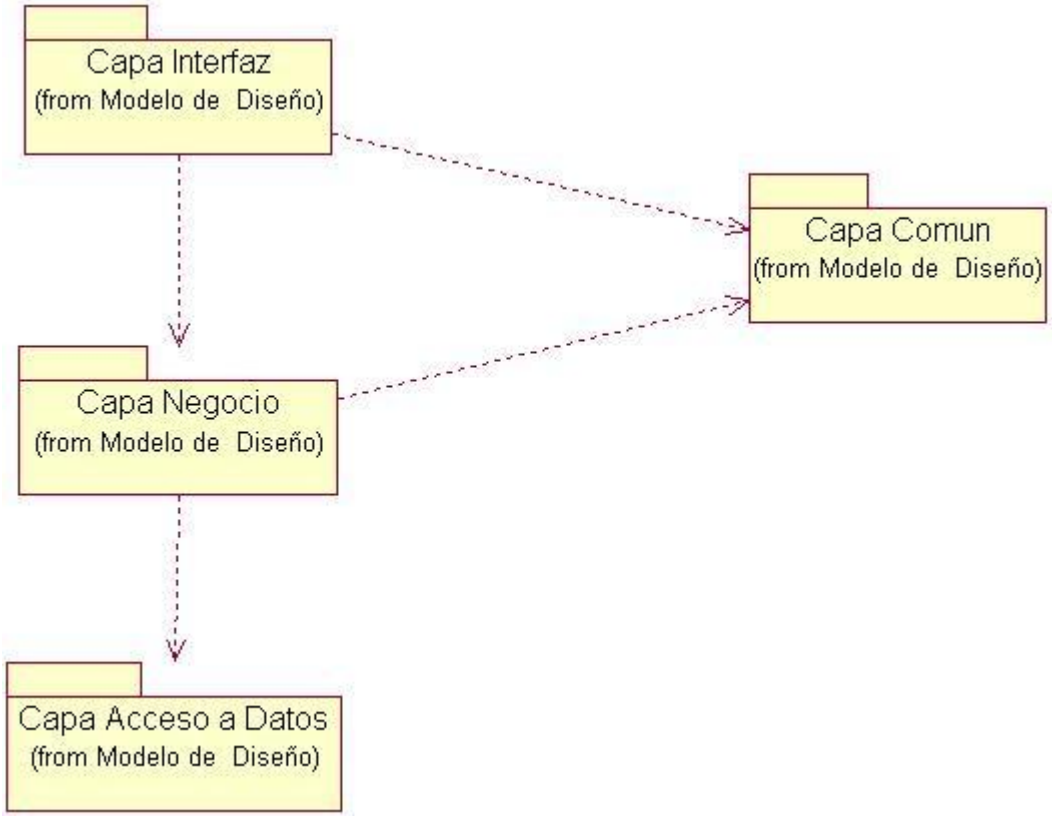

Figura 21: Diagrama de paquetes del diseño de la solución propuesta

2.5.3. Diagrama de clases persistentes

La solución propuesta se encarga solamente de persistir la información referente a los datos filiatorios de las personas por lo que el modelo de datos estará compuesto solamente de una clase, la cual representará los datos de las personas.

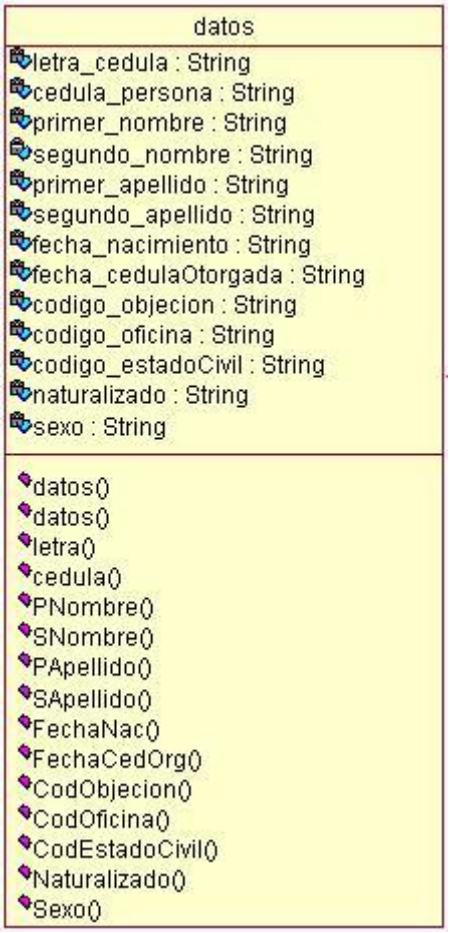

Figura 22: Clase persistente

2.6. Modelo de implementación de la solución propuesta

A continuación se muestran los diagramas de componentes pertenecientes a cada capa del sistema, así como el diagrama de componentes general de la solución propuesta.

2.6.1. Diagrama de componentes realizados

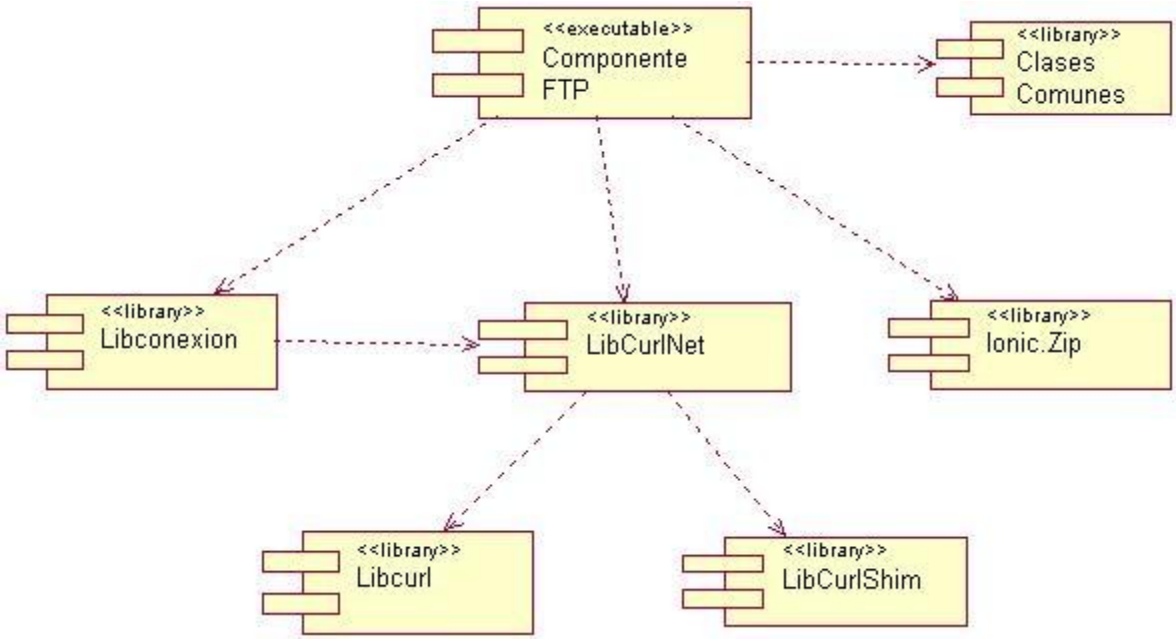

Figura 23: Capa interfaz

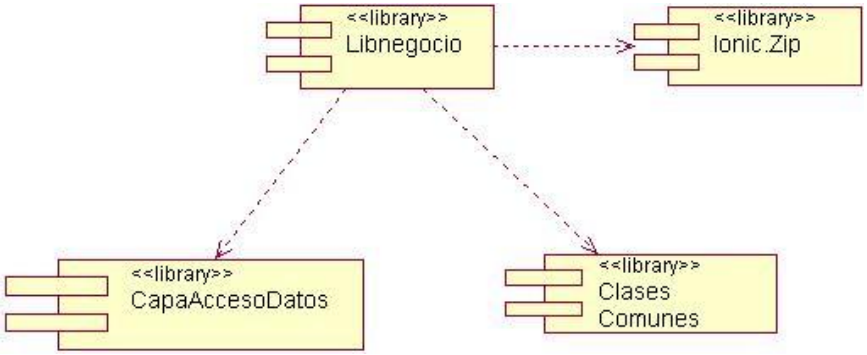

Figura 24: Capa negocio

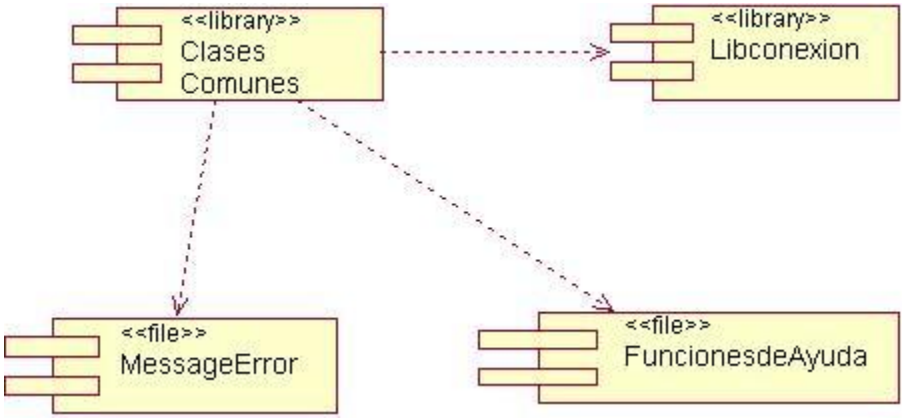

Figura 25: Capa común

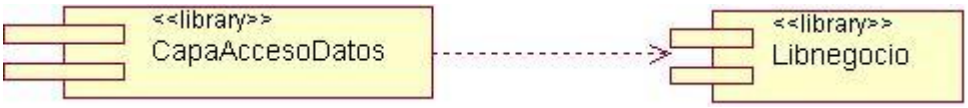

Figura 26: Capa acceso a datos

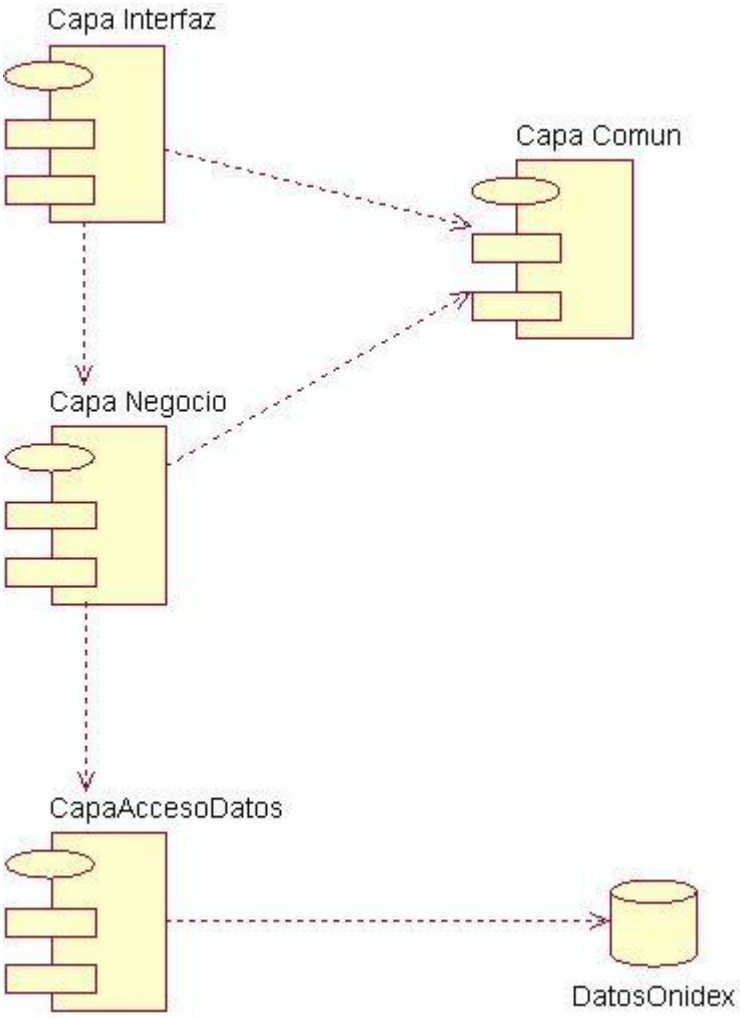

Figura 27: Diagrama de componentes de la solución propuesta

2.7. Modelo de despliegue de la solución propuesta.

Para el despliegue de la solución propuesta será necesario disponer de una PC cliente con una versión del sistema operativo Windows XP, en esta PC se va a instalar y a utilizar la solución propuesta. También será necesario instalar la base de datos ya sea en la misma PC cliente o en otro ordenador destinado para este uso. Además será necesario disponer del sitio de transferencia que utilizará la solución para transferir la información requerida. La transferencia de archivos se realizará mediante el uso del protocolo FTPS.

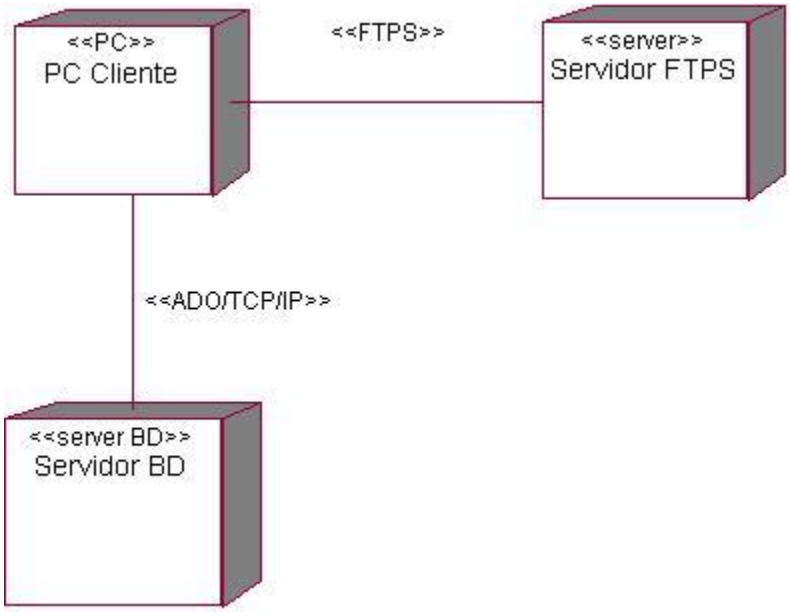

Figura 28: Diagrama de despliegue de la solución propuesta

2.8. Reglas de codificación

A continuación se definen las principales reglas que se tuvieron en cuenta en la codificación de la solución propuesta, lo que ayudó a tener un código más uniforme, claro y fácil de mantener.

Variables miembros de las clases

- **Regla:** Las variables miembros de clase se escriben con minúsculas, las variables con nombres compuestos, comienzan con la primer palabra enteramente en minúscula y el resto comenzando con mayúscula. Otra alternativa en nombres compuestos es la primera palabra minúscula seguida del prefijo '_' y el resto en minúscula.

- **Justificación:** Son las más usadas en todos los programas por lo que es importante que se puedan identificar fácil e intuitivamente y además esta es la forma más utilizada para nombrar este tipo de variables.

Clases

- **Regla:** Los nombres de clases comienzan con mayúsculas, con las palabras que la forman en minúsculas y separadas por mayúsculas y alternativamente puede estar separado por el prefijo '_' y seguidas por mayúsculas, de la misma forma que las variables miembro.
- **Justificación**: Los nombres de las clases abundan por todo el código del programa por lo que es necesario utilizar nombres intuitivos para ellas además este formato es compatible con estándares de notación para lenguajes como C#.

Métodos miembros de clases

- **Regla:** Los métodos miembros de clases siguen la notación idéntica a las clases, es decir comenzando con mayúscula y separadas y separadas por mayúsculas.
- **Justificación:** Estos métodos son muy utilizados por lo que es clave que su lectura sea fácil y rápida.

Parámetros (Argumentos) de métodos

- **Reglas:** Los parámetros siguen la notación idéntica de las variables.
- **Justificación:** De esta forma se identifican rápidamente ya que este formato es clásico para los parámetros.

Enumerados

- **Reglas:** Los nombres de los enumerados comienzan con la letra E mayúscula, seguido del nombre en un formato idéntico a las clases.
- **Justificación:** Con la letra E mayúscula al principio del nombre se identifica claramente el tipo de símbolo.

Comentarios

- **Regla:** Se usan los comentarios en línea para facilitar la comprensión del código.

Objeto	Prefijo	Ejemplo
Form	form	formPrincipal
TextBox	textbox	textboxNombre
Label	label	labelNombre
CheckBox	checkbox	checkboxExiste
Combobox	comobobox	comoboboxTipo
Button	button	buttonOK
GrupoBox	grupobox	grupoboxFTP
TreeView	treeview	treeviewArbolRecursos
TabControl	tabcontrol	tabcontrolConfiguracion

Tabla 1: Nombres de los objetos en los formularios

2.9. Conclusiones

Se crearon los diagramas de clases de diseño del sistema, además se hizo una descripción de cada una de las principales clases implementadas en la solución, especificándose su responsabilidad en la misma. Se realizó el modelo de implementación y despliegue del sistema. Además se definieron y pusieron en práctica estándares de codificación en la implementación de la solución propuesta, lo que ayudó a tener un código más uniforme, claro y fácil de mantener.

Capítulo 3: Validación de la solución propuesta

3.1. Introducción

En el presente capítulo se comienza con la validación de la solución propuesta a través de diferentes técnicas de pruebas. Dentro de dichas técnicas, se encuentran las pruebas de caja blanca y las pruebas de caja negra, además se crearon los casos de pruebas asociados a las mismas. El principal beneficio de realizar esta validación es que permite probar segmentos del programa para verificar que la solución propuesta funciona correctamente.

3.2. Pruebas de caja negra

Este tipo de prueba se aplicó con el principal objetivo detectar el incorrecto o incompleto funcionamiento del sistema, así como detectar posibles errores de interfaces. Estas pruebas se centraron fundamentalmente en los requisitos funcionales de software para verificar la calidad funcional de la solución propuesta.

Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejercitan completamente todos los requisitos funcionales del programa. Dentro de las técnicas de pruebas de caja negra, se utilizó la partición de equivalencia. La cual divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones de software.

La partición equivalente se dirige a la definición de casos de pruebas que descubran clases de errores. Para definir las clases de equivalencia se tuvieron en cuenta un conjunto de reglas:

1. Si una condición de entrada especifica un rango, entonces se confeccionan una clase de equivalencia válida y dos inválidas.
2. Si una condición de entrada especifica la cantidad de valores, se identifica una clase de equivalencia válida y dos inválidas.
3. Si una condición de entrada especifica un conjunto de valores de entrada y existen razones para creer que el programa trata en forma diferente a cada uno de ellos, se identifica una clase válida para cada uno de ellos y una clase inválida.
4. Si una condición de entrada especifica una situación de tipo “debe ser”, se identifica una clase válida y una inválida.

Posterior a la definición de las clases válidas e inválidas se procedió a diseñar los casos de pruebas para los casos de usos más significativos (configurar conexión con el FTPS, conectar con el FTPS, descargar un recurso, subir un recurso y procesar recurso).

3.2.1. Casos de prueba (CP)

CP Configurar conexión con el FTPS

Descripción de la funcionalidad

Se inicia por el responsable de la entidad SAREN, el mismo consiste en definir las opciones para la configuración de la conexión con el sitio de transferencia. Se identifican fundamentalmente dos escenarios, el primero de estos escenarios consiste en configurar las opciones de conexión sin la existencia de un servidor proxy, por último, el segundo de estos escenarios consiste en configurar las opciones de conexión excepto que en este caso si existe un servidor proxy.

Flujo central

1. El usuario selecciona la interfaz Configuración de la conexión FTP
2. El usuario selecciona la pestaña Opciones del FTP
3. El sistema muestra una interfaz con los datos a recoger para la configuración de la conexión con el servidor FTP.
4. El usuario entra al sistema todos los datos necesarios para la conexión con el servidor FTP.
5. El usuario ordena aceptar la operación.
6. El sistema acepta la operación y cierra la interfaz terminando así el caso de uso.

En caso de existir un servidor proxy se agrega al flujo central anterior el flujo que se muestra a continuación.

5. El usuario selecciona la pestaña Opciones del Proxy.

6. El sistema muestra una interfaz con los datos a recoger para la configuración de la conexión con el servidor FTP mediante un proxy.
7. El usuario entra al sistema todos los datos necesarios para la conexión con el servidor Proxy.
8. El usuario ordena aceptar la operación.
9. El sistema acepta la operación y cierra la interfaz terminando así el caso de uso.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
El usuario presiona el botón “OK”.		El sistema muestra un mensaje indicándole al usuario que la operación se realizado satisfactoriamente. Terminando así el caso de uso	Satisfactorio	
	El usuario presiona el botón “OK” con campos obligatorios vacios.	El sistema debe mostrar un mensaje informando al usuario que debe llenar todos los campos obligatorios.	Insatisfactorio	
El usuario ordena Cancelar la Operación.		Se cancela la operación. Terminando así el Caso de Uso.	Satisfactorio	

CP Conectar con el FTPS

Descripción de la funcionalidad

Se inicia por el responsable de la entidad SAREN, el mismo consiste en conectar el sistema al sitio de transferencia teniendo en cuenta las opciones de conexión definidas anteriormente.

Flujo Central

1. El usuario selecciona la opción Conectar al servidor FTP
2. El sistema muestra una interfaz con una vista de dicho servidor FTP terminando así el caso de uso.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
El usuario selecciona la opción "Conectar al servidor FTP".		El sistema muestra una interfaz con una vista del servidor FTP. Terminando así el caso de uso	Satisfactorio	
	El usuario selecciona la opción "Conectar al servidor FTP" sin antes haber definido las opciones de la conexión.	El sistema debe mostrar un mensaje informando al usuario que debe verificar que haya configurado la conexión con el servidor FTP.	Insatisfactorio	

CP Descargar un recurso

Descripción de la funcionalidad

Se inicia por el responsable de la entidad SAREN, el mismo consiste en descargar un recurso del sitio de transferencia.

Flujo Central

1. El usuario selecciona el recurso que desea descargar del servidor FTP.
2. El usuario selecciona la opción: Descargar un recurso del servidor.
3. El sistema muestra una ventana de diálogo para que el usuario seleccione la ubicación hacia donde desea almacenar el recurso.
4. El usuario selecciona la ubicación del recurso y selecciona guardar.
5. El sistema realiza la operación terminando así el caso de uso.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
El usuario selecciona la opción "Descargar un recurso del servidor".		El sistema realiza la operación. Terminando así el caso de uso	Satisfactorio	
	El usuario selecciona la opción "Descargar un recurso del servidor" sin antes	El sistema debe mostrar un mensaje informando al usuario que debe estar conectado al servidor FTP para realizar dicha	Insatisfactorio	

	conectarse al servidor.	operación.		
	El usuario selecciona la opción “Descargar un recurso del servidor” sin antes seleccionar el recurso que desea descargar	El sistema debe mostrar un mensaje informando al usuario que debe seleccionar un recurso válido para realizar dicha operación.	Insatisfactorio	

CP Subir un recurso

Descripción de la funcionalidad

Se inicia por el responsable de la entidad SAREN, el mismo consiste en subir o adicionar un recurso al sitio de transferencia.

Flujo Central

1. El usuario selecciona el directorio en el servidor FTP hacia donde desea subir el recurso.
2. El usuario selecciona la opción: Subir un recurso al servidor.
3. El sistema muestra una ventana de diálogo para que el usuario seleccione la ubicación del recurso que desea subir o adicionar al servidor.
4. El usuario selecciona la ubicación del recurso y selecciona abrir.
5. El sistema realiza la operación terminando así el caso de uso.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
El usuario selecciona la opción “Subir un recurso al servidor”.		El sistema realiza la operación. Terminando así el caso de uso	Satisfactorio	
	El usuario selecciona la opción “Subir un recurso del servidor” sin antes conectarse al servidor.	El sistema debe mostrar un mensaje informando al usuario que debe estar conectado al servidor FTP para realizar dicha operación.	Insatisfactorio	
	El usuario selecciona la opción “Subir un recurso al servidor” sin antes seleccionar una ubicación valida en el sitio de transferencia.	El sistema debe mostrar un mensaje informando al usuario que debe seleccionar una ubicación válida para realizar dicha operación.	Insatisfactorio	

CP Procesar un recurso

Descripción de la funcionalidad

Se inicia por el responsable de la entidad SAREN, el mismo consiste en procesar un recurso previamente descargado por él sistema.

Flujo Central

1. El usuario selecciona la opción: Procesar un recurso descargado.
2. El sistema muestra una ventana de diálogo para que el usuario seleccione la ubicación del recurso que desea procesar.
3. El usuario selecciona la ubicación del recurso y selecciona abrir.
4. El sistema realiza la operación terminando así el caso de uso.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Observaciones
El usuario selecciona la opción "Procesar un recurso descargado".		El sistema realiza la operación. Terminando así el caso de uso	Satisfactorio	
	El usuario selecciona la opción "Procesar un recurso descargado" sin antes configurar la conexión al	El sistema debe mostrar un mensaje informando al usuario que debe configurar la conexión al servidor de base de datos para realizar dicha	Insatisfactorio	

	servidor de Base de Datos.	operación.		
	El usuario selecciona la opción "Procesar un recurso descargado" y selecciona un archivo que no corresponde con el archivo transferido desde el sitio de transferencia.	El sistema debe mostrar un mensaje informando al usuario que no se puede realizar dicha operación.	Insatisfactorio	

3.3. Pruebas de caja blanca

Este tipo de prueba tiene como principal objetivo garantizar que se ejerciten por lo menos una vez todos los caminos independientes de cada módulo o método.

"La prueba de caja blanca es considerada como uno de los tipos de pruebas más importantes que se le aplican al software, logrando como resultado que disminuya en un gran por ciento el número de errores existentes en los sistemas y por ende una mayor calidad y confiabilidad" [16].

Para el desarrollo de estas pruebas unitarias se utilizó la herramienta Nunit, la cual ofrece un conjunto de funcionalidades que permiten implementar pruebas en un proyecto desarrollado en C#.

"Nunit es una herramienta que se encarga de analizar ensamblados generados por .NET, interpretar las pruebas inmersas en ellos y ejecutarlas. Utiliza atributos personalizados para interpretar las pruebas y provee además métodos para implementarlas. En general, Nunit compara valores esperados y valores generados, si estos son diferentes la prueba no pasa, en

caso contrario la prueba es exitosa" [17]. Además provee una interfaz gráfica para ejecutar y administrar dichas pruebas.

3.3.1. Métrica de la complejidad ciclomática.

Se le realizó el cálculo de la complejidad ciclomática de todos los bloques de código dentro del sistema, la cual ayudó a reflejar una medida de la complejidad lógica de la solución, obteniendo el número de caminos independientes dentro de cada fragmento de código analizado. Además permitió conocer la cuota máxima de pruebas a realizar para lograr que se ejecute cada sentencia del programa al menos una vez, la cual es exactamente el valor de la complejidad ciclomática en cada elemento o módulo.

A partir de esta medida, se diseñaron casos de prueba que forzaron el recorrido de cada sentencia del programa. A continuación se muestran algunos ejemplos de los principales segmentos de código analizados:

```
public void DescargarRecurso(TreeNode nodo)
{
    if (nodo.Tag is Recurso_Hoja)
    {
        string recurso = nodo.Text;
        string direccionDescarga = "";
        saveFileDialogDescargarRecurso.Title = "Descargar el recurso" + " " + recurso;
        saveFileDialogDescargarRecurso.FileName = recurso;
        if (saveFileDialogDescargarRecurso.ShowDialog() == DialogResult.OK)
        {
            direccionDescarga = saveFileDialogDescargarRecurso.FileName;
            ControlOperaciones.Instance.DefinirDirectorio(nodo);

            try
            {
                Easy easy = new Easy();
                ControlFTP.Instance.ControlConexion.Conectar(ControlFTP.Instance.ControlConexion.Ftp.Tipo_FTP,
                                                             easy);
                ControlFTP.Instance.DescargarRecurso(easy, direccionDescarga);

                ControlFTP.Instance.ControlConexion.RealizarConexion(easy);
                ControlFTP.Instance.ControlConexion.CerrarConexion(easy);
                ControlFTP.Instance.CerrarFileStream();
                MessageError.Instance.Mensaje = "La operación se ha realizado con exito";
                MessageError.Instance.MostrarMensajeInformativo();

            }
            catch (Exception exception)
            {
                MessageError.Instance.Mensaje = exception.Message;
                MessageError.Instance.MostrarMensajeError();
            }
        }

    }
    else
        if (nodo.Text == "No existen recursos disponibles")
        {
            MessageError.Instance.Mensaje = "NO existe el recurso";
            MessageError.Instance.MostrarMensajeError();
        }
}
```

Figura 29: Descargar un recurso del sitio de transferencia

Complejidad Ciclomática *V (G)*

V (G) = Nº de Aristas - Nº de Nodos + 2

V (G) =9-7+2

V (G) =4

Caminos Independientes:

• 1-2-3-4-7

• 1-2-3-7

• 1-5-6-7

• 1-5-7

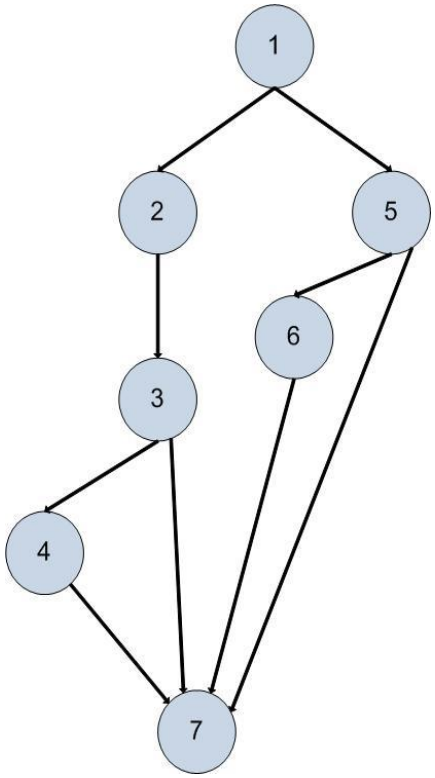

Figura 30 Grafo de flujo

```
public void SubirRecurso(TreeNode ubicacionSeleccionada, string nombreRecurso, string direccionSubida)
{
    if (ubicacionSeleccionada != null)
    {
        ControlOperaciones.Instance.DefinirDirectorio(ubicacionSeleccionada);
        ControlFTP.Instance.ControlConexion.Ftp.DirectorioInicial += "/" + nombreRecurso;
    }
    else
        ControlFTP.Instance.ControlConexion.Ftp.DirectorioInicial = nombreRecurso;

    try
    {
        Easy easy = new Easy();
        ControlFTP.Instance.ControlConexion.Conectar(ControlFTP.Instance.ControlConexion.Ftp.Tipo_FTP, easy);
        ControlFTP.Instance.SubirRecurso(easy, direccionSubida);


        ControlFTP.Instance.ControlConexion.RealizarConexion(easy);
        ControlFTP.Instance.ControlConexion.CerrarConexion(easy);
        ControlFTP.Instance.CerrarFileStream();
        MessageError.Instance.Mensaje = "La operación se ha realizado con exito";
        MessageError.Instance.MostrarMensajeInformativo();

    }
    catch (Exception exception)
    {
        MessageError.Instance.Mensaje = exception.Message;
        MessageError.Instance.MostrarMensajeError();
    }

}
```

Figura 31: Subir un recurso al sitio de transferencia

Complejidad Ciclomática V (G)

V (G) = Nº de Aristas - Nº de Nodos + 2

V (G) =5-5+2

V (G) =2

Caminos Independientes:

• 1-2-4-5

• 1-3-4-5

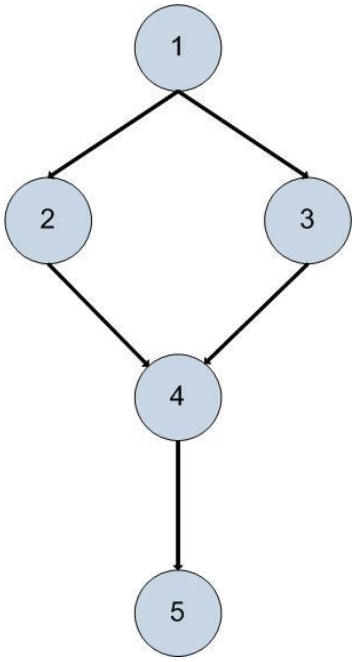

Figura 32 Grafo de flujo

```
public void ProcesarRecursoDescargado(string directorio_completo)
{

    string directorio_extraccion = "";
    string nombreArchivo = "";
    string codigoHash = "";
    List<datos> listaPersonas = new List<datos>();

    nombreArchivo = FuncionesdeAyuda.Instance.NombreRecurso(directorio_completo);
    ValidarArchivo.Instance.NombreArchivo = directorio_completo;
    codigoHash = FuncionesdeAyuda.Instance.Codigo(nombreArchivo);

    if (ControlOperacionesNegocio.Instance.ValidacionArchivo(codigoHash))
    {
        try
        {
            ArchivoZip archivoZip = new ArchivoZip(directorio_completo);
            ControlZip.Instance.Archivo_zip = archivoZip;
            ControlZip.Instance.ListarContenido_ArchivoZip();
            directorio_extraccion = FuncionesdeAyuda.Instance.NombreDirectorio_ExtraccionZip(directorio_completo);
            ControlZip.Instance.ExtraerContenido_ArchivoZip(directorio_extraccion);

            listaPersonas = ControlOperacionesNegocio.Instance.ListaDatosPersonaParaProcesar(directorio_extraccion);

            ControlOperacionesNegocio.Instance.SalvarDatosPersona(listaPersonas);


        }
        catch (ZipException)
        {
            MessageError.Instance.Mensaje = "El archivo seleccionado no tiene el formato correcto, verifique que el
            MessageError.Instance.MostrarMensajeError();
        }
    }
    else
    {
        MessageError.Instance.Mensaje = "El archivo no puede ser procesado, porque se ha detectado algun tipo de mo
        MessageError.Instance.MostrarMensajeError();
    }

}
```

Figura 33: Procesar un recurso descargado

Complejidad Ciclomática V (G)

V (G) = Nº de Aristas - Nº de Nodos + 2

V (G) =6-6+2

V (G) =2

Caminos Independientes:

• 1-2-3-4-6

• 1-2-3-5-6

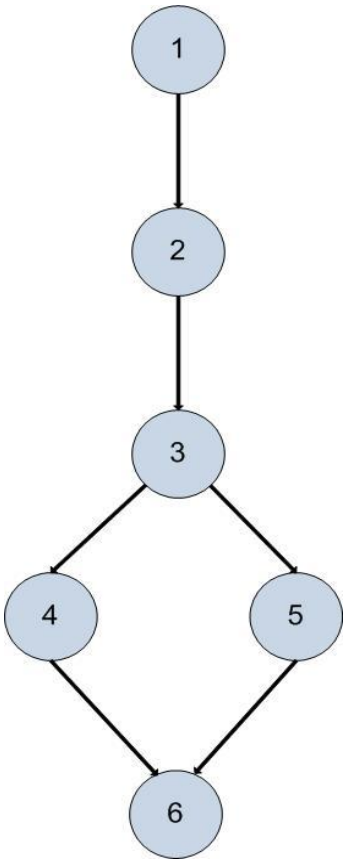

Figura 34 Grafo de flujo

```
public void ConfigurarConexion()
{

    configuracionConexion = DeserializarConfiguracionConexion();

    ControlFTP.Instance.ControlConexion.Ftp.DireccionServidor = configuracionConexion.Servidor_FTP;
    ControlFTP.Instance.ControlConexion.Ftp.Tipo_FTP = (ETipo_FTP)Enum.Parse(typeof(ETipo_FTP),
                                                       configuracionConexion.Tipo_FTP);

    ControlFTP.Instance.ControlConexion.Ftp.Puerto = int.Parse(configuracionConexion.Puerto_FTP);
    ControlFTP.Instance.ControlConexion.Ftp.Usuario = configuracionConexion.Usuario_FTP;
    ControlFTP.Instance.ControlConexion.Ftp.Contraseña = configuracionConexion.Contraseña_FTP;
    if (configuracionConexion.Existe_Proxy == "SI")
    {
        ControlFTP.Instance.ControlConexion.Proxy.DireccionServidor = configuracionConexion.Servidor_Proxy;
        ControlFTP.Instance.ControlConexion.Proxy.Tipo_proxy = (CURLproxyType)Enum.Parse(typeof(CURLproxyType)
                                                               configuracionConexion.Tipo_Proxy); ;
        ControlFTP.Instance.ControlConexion.Proxy.Puerto = int.Parse(configuracionConexion.Puerto_Proxy);
        ControlFTP.Instance.ControlConexion.Proxy.Usuario = configuracionConexion.Usuario_Proxy;
        ControlFTP.Instance.ControlConexion.Proxy.Contraseña = configuracionConexion.Contraseña_Proxy;
    }

}
```

Figura 35: Configurar conexión con el sitio de transferencia

Complejidad Ciclomática V (G)

V (G) = Nº de Aristas - Nº de Nodos + 2

V (G) =5-5+2

V (G) =2

Caminos Independientes:

• 1-2-3-4-5

• 1-2-3-5

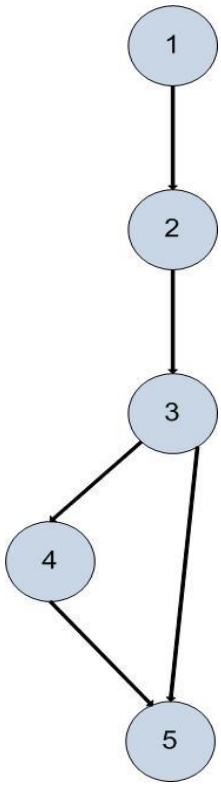

Figura 36 Grafo de flujo

Muchos actores han definido umbrales respecto al valor de la métrica de la complejidad ciclomática.

Valor de V(G)	**Evaluación**
1-10	Un programa simple sin mucho riesgo
11-20	Medianamente complejo
21-50	Un programa complejo
50+	Programa inestable

Tabla 2: Tabla de posibles valores

A partir del análisis del cálculo de la complejidad ciclomática en la solución propuesta, el cual se encuentra entre los valores 1-10, siendo este el rango de valores más alto de V (G) en la solución y del resultado de cada una de las pruebas. Se llegó a la conclusión de que de forma general los resultados obtenidos en las pruebas realizadas fueron positivos teniendo en cuenta que el sistema no es complejo y presenta un bajo riesgo de fallos.

3.4. Conclusiones

En este capítulo se realizó la prueba y validación de la solución propuesta. Se demostró que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta. La utilización del framework Nunit en las pruebas, permitió encontrar defectos y anomalías no esperadas, lo cual permitió crear un código sólido y sin errores, lo que resultó de suma utilidad teniendo en cuenta que el período de tiempo para desarrollar la solución fue relativamente corto y se necesitaba una aplicación que corriera con la mayor calidad y estabilidad posible.

Conclusiones generales

La realización de este trabajo de diploma ha aportado importantes conocimientos al autor en la creación de un sistema de transferencia de archivos. Para llevar a fin el objetivo principal se estudiaron las últimas tendencias y tecnologías actuales en cuanto se refiere al proceso de transferencia de archivos, así como, al diseño e implementación de software.

Se lograron identificar los elementos necesarios para elaborar el modelo de implementación que se refiere a la creación de un sistema para la transferencia de archivos del sistema SAREN basado en la situación problémica actual.

Llegando así a la propuesta de un sistema para el proceso de transferencia de archivos del sistema SAREN, la cual mejora y beneficia a dicho proceso.

Finalmente el presente trabajo de diploma ha servido para consolidar conocimientos en esta rama de la ingeniería informática.

Recomendaciones

Se recomienda migrar la solución propuesta hacia alguna plataforma de software libre, preferentemente el proyecto Mono lo cual permitirá la reutilización de todo el código ya desarrollado, de esta manera se está en consecuencia con el objetivo que persigue el polo de la facultad así como la dirección de la universidad, y además para que dicha solución pueda ser utilizado en proyectos semejantes.

Referencias bibliográficas

[RFC 114] A. Bhushan “A File Transfer Protocol”,16 de Abril de 1971.

[RFC 959] J. Postel y J. Reynolds, “File Transfer Protocol (FTP)”, Octubre 1985.

[RFC 2228] M. Horowitz, “FTP Security Extensions”, Octubre 1997.

[RFC 2246] T. Dierks, “The TSL Protocol Version 1.0”, Enero 1999.

[RFC 2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee ” Hypertext Transfer Protocol -- HTTP/1.1”, June 1999.

[RFC 4217] P. Ford-Hutchinson “Securing FTP with TLS”, Octubre 2005.

[RFC 4251] T. Ylonen, C. Lonvick “The Secure Shell (SSH) protocol architecture” Enero 2006.

[RFC 2818] E. Rescorla “HTTP over TLS” Mayo 2000.

[RFC 4130] D. Moberg, R. Drummond “MIME-Based Secure Peer-to-Peer Business Data Interchange Using HTTP, Applicability Statement 2 (AS2)” Julio 2005.

[1]. De: http://www.bolivar.udo.edu.ve/microinternet/articulos/Xmodem.pdf

[2]. De: http://www.openssh.com/es/index.html

[3]. De: http://www.verisign.es/ssl/ssl-information-center/ssl-basics/index.html

[4]. De: http://www.ekomercio.com/as2.htm

[5]. De: http://www.mentores.net/articulos/intro_microsoft_sol_frame.htm

[6]. De: http://www.willydev.net/descargas/prev/TodoAgil.pdf

[7]. De: http://adonisnet.files.wordpress.com/2008/06/articulo-metodologia-de-sw-formato.doc

[8] Jacobson, I.; Booch, G. y Rumbaugh, J.; “El Proceso Unificado de Desarrollo de software”. 2000.

[9] Jacobson, I., G. Booch, and J. Rumbaugh, El Lenguaje Unificado de Modelado. 1999.

[10] Jacobson, I., G. Booch, and J. Rumbaugh, El Lenguaje Unificado de Modelado. 1999.

[11] Larman, C. UML y PATRONES. Introducción al análisis y diseño orientado a objetos, 2004.

[12] De: http://www.willydev.net/InsiteCreation/v1.0/descargas/prev/estiloypatron.pdf

[13] Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M (1996). Pattern-Oriented Software Architecture: A System of Patterns. John Wiley,

[14] De: http://www.desarrolloweb.com/articulos/1328.php

[15] De: http://www.scribd.com/doc/9174723/Paradigmas-de-Programacion

[16] Pressman, R.S., Ingeniería de software. Un enfoque practico. Vol. Vol. 1. 1998.

[17]. De: http://www.elguille.info/colabora/puntoNET/giovannyf_NUnit.htm

Huerta, R.P.L.G.y.J.L.D.J., Herramientas Case.

Erich Gamma, R. H., Ralph Johnson, John Vlissides, ForeWord by Grady Booch. Desing Patterns, Elements of Reusable Object-Oriented Software, 1994.

Montecinos M.I. (2007) Análisis y Diseño para soluciones con Arquitectura Microsoft.NET.

AS2: entrega segura de documentos por la Web:
http://www.idg.es/comunicaciones/articulo.asp?id=145889

Sterling Managed File Transfer: http://www.sterlingcommerce.es/Products/MFT/

http://www.sterlingcommerce.com/apps/collaterallibrary/external/DownloadFile.asp?fil=101887

CyberFusion Integration Suite (CFI) ™ for Secure files Transfer:
http://www.proginetuk.co.uk/sections/products/secure-file-transfer/command-center/64/

Microsoft Solution Software MSF:
http://www.mentores.net/articulos/intro_microsoft_sol_frame.htm

Libcurl: http://curl.haxx.se/

Libcurl-net: http://sourceforge.net/projects/libcurl-net/

DotNetZip: http://www.codeplex.com/DotNetZip

SFTP: http://www.vandyke.com/technology/drafts.html

SSL: http://www.verisign.es/ssl/ssl-information-center/index.html

Glosario de términos

A

ASCII (American Standard Code for Information Interchange - Código Estadounidense Estándar para el Intercambio de Información): es un código de caracteres basado en el alfabeto latino tal como se usa en inglés moderno y en otras lenguas occidentales.

C

Cifrado simétrico es el método criptográfico que usa una misma clave para cifrar y descifrar mensajes. Las dos partes que se comunican han de ponerse de acuerdo de antemano sobre la clave a usar. Una vez que ambas tienen acceso a esta clave, el remitente cifra un mensaje usándola, lo envía al destinatario, y éste lo descifra con la misma clave.

E

EBCDIC (Extended Binary Coded Decimal Interchange Code): es un código estándar de 8 bits usado por computadoras mainframe IBM. Es un código binario que representa caracteres alfanuméricos, controles y signos de puntuación. Cada carácter está compuesto por 8 bits = 1 byte, por eso EBCDIC define un total de 256 caracteres.

H

Hipertexto: es el nombre que recibe el texto que en la pantalla de una computadora conduce a su usuario a otro texto relacionado. La forma más habitual de hipertexto en documentos es la de hipervínculos o referencias cruzadas automáticas que van a otros documentos.

I

IETF: El IETF (Internet Engineering Task Force, en castellano Grupo de Trabajo en Ingeniería de Internet) es una organización internacional abierta de normalización, que tiene como objetivos el contribuir a la ingeniería de Internet, actuando en diversas áreas, tales como transporte, encaminamiento, seguridad. Fue creada en EE. UU en 1986.

M

MAC: Message Authentication Code o MAC (del inglés, código de autenticación de mensajes) es un código que se genera a partir de un mensaje de longitud arbitraria y de una clave secreta compartida entre remitente y destinatario, y que sirve para autenticar el mensaje. También se le llama cryptographic checksum.

R

RFC (Request For Comments): Las Request For Comments —petición de comentarios— son una serie de notas sobre Internet que comenzaron a publicarse en 1969. Se abrevian como RFC. Cada una de ellas individualmente es un documento cuyo contenido es una propuesta oficial para un nuevo protocolo de la red Internet, donde se explica con todo detalle para que en caso de ser aceptado pueda ser implementado sin ambigüedades.

U

URL: URL significa Uniform Resource Locator, es decir, localizador uniforme de recurso. Es una secuencia de caracteres, de acuerdo a un formato estándar, que se usa para nombrar recursos, como documentos e imágenes en Internet, por su localización.

W

Web: World Wide Web (o la "Web") o Red Global Mundial es un sistema de documentos de hipertexto y/o hipermedios enlazados y accesibles a través de Internet. Con un navegador Web, un usuario visualiza páginas web que pueden contener texto, imágenes, vídeos u otros contenidos multimedia.