ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DISEÑO E IMPLEMENTACIÓN DEL PROTOTIPO DE UN SISTEMA COMPUTARIZADO DISTRIBUIDO ORIENTADO A LA UTILIZACIÓN DE LOS SERVICIOS EN UN CAMPUS UNIVERSITARIO A NIVEL LOCAL CON CAPACIDAD PARA 2000 A 5000 ESTUDIANTES

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y REDES DE INFORMACIÓN

DIANA ELIZABETH ALCOCER PULUPA
diany1186@hotmail.com

FABIÁN RODRIGO ORTIZ TAMAYO
vyper1985@hotmail.com

DIRECTOR: Ing. Mónica Vinueza
monica.vinueza@epn.edu.ec

Quito, Septiembre 2013

DECLARACIÓN

Nosotros, Diana Elizabeth Alcocer Pulupa y Fabián Rodrigo Ortiz Tamayo, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Diana Elizabeth Alcocer Pulupa

Fabián Rodrigo Ortiz Tamayo

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Diana Elizabeth Alcocer Pulupa y Fabián Rodrigo Ortiz Tamayo, bajo mi supervisión.

Ing. Mónica Vinueza

DIRECTOR DE PROYECTO

AGRADECIMIENTO

Primero agradezco a Dios por tantas bendiciones que me ha dado en la vida y me las sigue dando, agradezco a mis padres, hermanos y familia en general porque ellos han sido un pilar importante para que yo llegue a cumplir una más de mis metas y también agradezco a mi compañera por la paciencia, el amor y la dedicación para el término del presente trabajo.

Fabián

AGRADECIMIENTO

Agradezco a Dios por darme la fuerza y los medios para poder realizar este trabajo, a mis padres y mis hermanos por apoyarme incondicionalmente durante toda mi vida, y especialmente a mi esposo por ser la persona que me acompaña siempre.

Diana

DEDICATORIA

Dedicado a toda mi familia y en especial a mi Madre que siempre ha estado incondicional y a mi Padre por sus consejos de vida, también a mis amigos y profesores que son con los que he compartido esta vida estudiantil.

Fabián

DEDICATORIA

Dedico este trabajo a toda mi familia y amigos, especialmente a mis padres por todo el amor y apoyo que me brindaron, a mi esposo por su comprensión, y a mi hija querida por ser una alegría en mi vida.

Diana

CONTENIDO

ÍNDICE DE FIGURAS

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS

CAPÍTULO 2: DISEÑO DEL SOFTWARE

CAPÍTULO 3: IMPLEMENTACIÓN DEL SISTEMA

CAPÍTULO 4: PRUEBAS, RESULTADOS Y COSTOS DEL PROTOTIPO

ÍNDICE DE TABLAS

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS

CAPÍTULO 2: DISEÑO DEL SOFTWARE

CAPÍTULO 3: IMPLEMENTACIÓN DEL SISTEMA

CAPÍTULO 4: PRUEBAS, RESULTADOS Y COSTOS DEL PROTOTIPO

ÍNDICE DE ESPACIOS DE CÓDIGO

CAPÍTULO 3: IMPLEMENTACIÓN DEL SISTEMA

RESUMEN

En el presente proyecto se implementa un Prototipo de un Sistema Computarizado Orientado a la utilización de los servicios en un campus universitario a nivel local.

Este prototipo automatiza cuatro servicios que se brindan en un campus universitario, considerados más importantes para los estudiantes.

Multiservicios Estudiantiles es un Sistema compuesto de tres capas en tres niveles. La capa más externa del Sitio Web Multiservicios Estudiantiles, es una Interfaz amigable en donde los diferentes Usuarios podrán interactuar con el Sistema; la capa intermedia se compone de los diferentes servicios que brinda el Sistema, llamada también Servicios Web; esta capa y la anterior fueron desarrolladas en la plataforma *Microsoft Visual Studio 2010*; por último se tiene la capa de Datos que es en donde se almacenan todos los datos que los diferentes Usuarios necesitan para interactuar con el Sistema, fue desarrollada en el Motor de Bases de Datos de *Microsoft SQL Server 2008*. El sistema cuenta con reconocimiento de huella digital de los Usuarios por lo que se usó un lector de huellas comercial que se acople al Sistema (*SecuGen-Humster Plus*).

Además se consideró la seguridad en el Sistema, por lo cual se hizo uso de *Active Directory* e *IPSec*, para que la transaccionabilidad entre servidores del sistema sea más Segura, y https para la comunicación entre las terminales y el servidor web.

En el Capítulo I se exponen los fundamentos teóricos describiendo las principales características y sustentación teórica de las herramientas utilizadas para implementar la aplicación, entre estas: arquitectura en capas, herramientas de desarrollo de software, información general sobre biometría, introducción a los Servicios Web y criterios de diseño, así como los protocolos utilizados lograr una

comunicación segura entre servidores con *IPSec*, y con los clientes mediante el protocolo *HTTPS*.

En el Capítulo II se detalla el diseño del software, mediante el análisis de requisitos utilizando el estándar *IEEE 830-1998* con énfasis en los requisitos funcionales, el diseño de la base de datos así como su diagrama, y los diagramas de clases, de secuencia, de actividades y de despliegue, importantes en el diseño de un sistema de software.

En el Capítulo III se muestra la implementación de cada una de las capas, con lo que se tiene la descripción de las diferentes páginas web y código utilizado para las operaciones realizadas con el lector de huellas digitales; el detalle de los servicios web y ejemplos de código relevante en su implementación, y la descripción de los procesos almacenados de la base de datos. Además se explica cómo se realizó la interconexión entre las capas, y la configuración de los servidores.

En el Capítulo IV se exponen las pruebas y resultados realizados al sistema, así como el análisis de costo de implementación. También se detallan los beneficios tangibles e intangibles de la aplicación del sistema en un campus universitario.

Finalmente en el Capítulo V se presentan las conclusiones y recomendaciones obtenidas luego del desarrollo del proyecto.

PRESENTACIÓN

Dado el creciente avance tecnológico en el campo de las aplicaciones web y en el de reconocimiento seguro de un usuario se vuelve necesaria la creación de un Sistema como el que se diseña en el presente trabajo, ya que la combinación de estas ramas tecnológicas da como resultado un servicio remoto, más seguro y eficiente que brinda satisfacción, en este caso para el Estudiante.

Las Herramientas utilizadas en este proyecto son de fácil uso gracias a su compatibilidad con los Sistemas Operativos *Windows*, que son los de mayor difusión y tienen gran acogida por parte de los usuarios; además están provistas de características que nos brindan mayor rapidez, seguridad y escalabilidad en aplicaciones web que manejan transacciones monetarias como el presente proyecto. Por otro lado, ya que son licenciadas, constan de soporte para ayuda o mejora de las mismas.

Una de las herramientas utilizadas en este proyecto fueron los Servicios Web que brindaron una mayor facilidad al momento de interconectar la capa de datos con la interfaz de usuario, sin embargo en la actualidad ya están siendo reemplazados por los nuevos Servicios *WCF* que brindan seguridad adicional al encriptar y empaquetar la información.

La aplicación Web Multiservicios Estudiantiles es un prototipo de un Sistema Computarizado Orientado a satisfacer las necesidades de un Estudiante dentro de un campus Universitario; ya que este brinda los servicios de préstamo y devolución de libros en Bibliotecas, pago de artículos en copiadoras, reservación de citas médicas en las diferentes especialidades y pago de alimentos en el Bar tan solo con el uso de su Huella digital.

CAPÍTULO I

1. FUNDAMENTOS TEÓRICOS

1.1 ARQUITECTURA DE SOFTWARE EN CAPAS

1.1.1 CAPAS Y NIVELES [L1]

En la ingeniería de software es común hablar de arquitectura de capas (*Layers*), sin embargo es frecuente confundirlas con los niveles (*Tiers*). Aunque utilicen nombres similares se debe tener en cuenta que las capas se refieren a la división lógica de componentes por su funcionalidad sin importar su ubicación física, mientras que los niveles se refieren a la distribución física los componentes en servidores separados. El modelo más utilizado es el de 3 capas, aunque dependiendo de la complejidad del sistema se utilizará un número diferente de capas teniéndose una arquitectura "*N-Capas*", siendo "*n*" el número de capas.

Así mismo se denomina a los niveles de un sistema con el término "*N-Tier*", siendo "N" el número de niveles. En el Figura 1.1 se muestra un esquema *3-Tier* y un esquema *N-Layer* donde se pueden observar las diferencias comentadas.

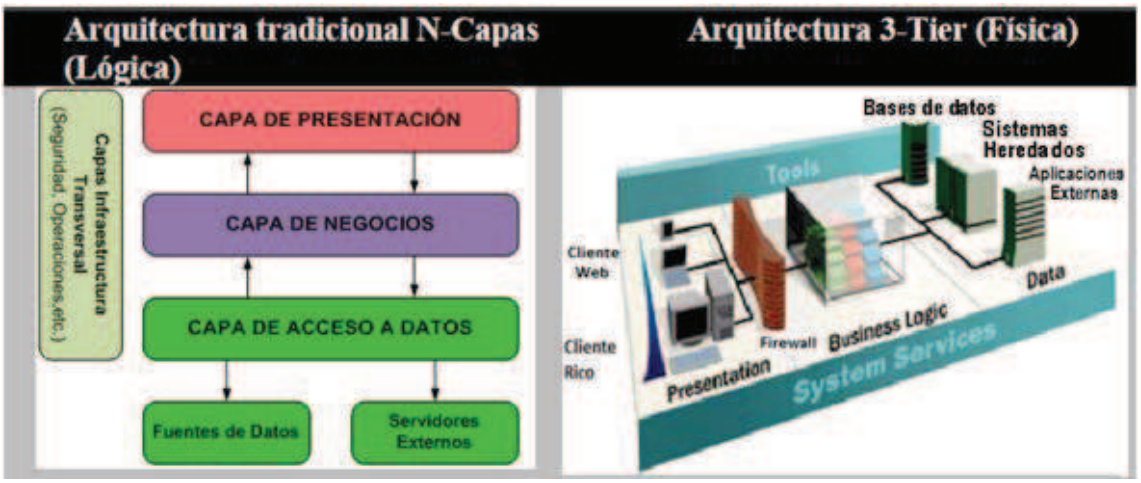

Figura 1.1 Arquitectura tradicional lógica vs. Arquitectura física [L1]

1.1.2 ARQUITECTURA EN CAPAS [L1, L2]

En este tipo de arquitectura se organiza el sistema en agrupaciones horizontales lógicas llamadas capas, las cuales proporcionan un conjunto servicios que ayudan a diferenciar entre los diferentes tipos de tareas a ser realizadas. De esta manera se maximiza la reutilización y sobretodo la escalabilidad. En resumen, se trata de aplicar el principio de "Separación de Responsabilidades" (*SoC - Separation of Concerns principle*) dentro de una Arquitectura.

Cada capa lógica de primer nivel puede tener un número concreto de componentes agrupados en sub-capas, las cuales realizan a su vez un tipo específico de tareas. Al identificar tipos genéricos de componentes que existen en la mayoría de las soluciones, se puede construir un patrón o mapa de una aplicación o servicio y usar dicho mapa como modelo del diseño. La estructura básica de una arquitectura en capas se representa en la Figura 1.2.

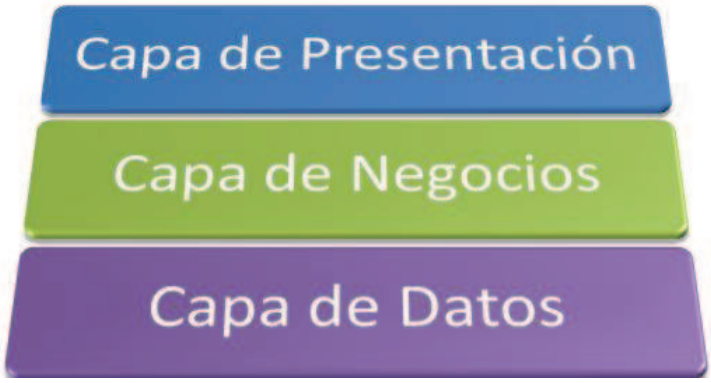

Figura 1.2 Arquitectura en capas [L2]

Entre las principales ventajas que presenta la arquitectura en capas se puede mencionar las siguientes:

- El mantenimiento de mejoras en una solución será mucho más fácil porque las funciones están localizadas, y el agregar o cambiar los métodos de una determinada capa no afecta el funcionamiento de las demás, lo que implica mayor escalabilidad.

- Se puede reutilizar las funciones implementadas en las diferentes capas por otros sistemas, es decir se las puede reutilizar y de esa manera se evita volver a escribir soluciones para problemas que ya fueron resueltos.

- Es más fácil aumentar los niveles de un sistema para hacerlo distribuido, si ha sido previamente dividido en capas lógicas.

- La distribución de capas (*layers*) en diferentes niveles físicos (*tiers*) puede, en algunos casos, mejorar la escalabilidad. Sin embargo este punto hay que evaluarlo con cuidado, pues puede impactar negativamente en el rendimiento, por ejemplo si el sistema es muy simple para necesitarlo.

Por otro lado la implementación de una arquitectura en capas puede conllevar las siguientes desventajas:

- La capa más externa del sistema no depende solamente de su predecesora inmediata, ya que las capas internas pueden proporcionar facilidades básicas que de ser requeridas por un usuario de un nivel superior tendría que “atravesar” las capas adyacentes para tener acceso a dichos servicios, lo cual dificultaría la estructuración del sistema.

- El rendimiento puede disminuir debido a que algunas veces se requieren múltiples niveles de interpretación de comandos, dicho de otra manera, si hay muchas capas, un servicio solicitado desde una capa superior puede tener que ser interpretado varias veces en varias capas antes de ser procesado.

Para evitar estos problemas, las aplicaciones tienen que comunicarse directamente con las capas interiores en lugar de usar los servicios proporcionados por las capas adyacentes.

Como aclaración final, no se debe confundir las capas de una arquitectura con las capas de un modelo, ya que en la arquitectura de software las capas se relacionan con las funciones que se cumplen dentro de un sistema, mientras que las capas de un modelo (como el modelo de referencia OSI) definen las diferentes fases por las deben pasar los datos para ir de un dispositivo a otro y sus protocolos.

1.1.3 ARQUITECTURAS CLIENTE-SERVIDOR DISTRIBUIDAS [L3]

En este tipo de arquitectura, una aplicación se modela como un conjunto de servicios proporcionados por servidores y un conjunto de clientes que usan dichos servicios y que necesitan conocer los servidores disponibles, aunque no necesariamente conozcan la existencia de otros clientes. Los modelos más populares son los de dos y tres capas.

En una arquitectura de dos capas de una aplicación cliente-servidor se presenta una capa que sintetiza la capa de presentación y de negocios, encargada de la interfaz con el usuario y del procesamiento de los datos producidos por el usuario y los devueltos por el servidor; y otra capa de bases de datos como se muestra en la Figura 1.3.

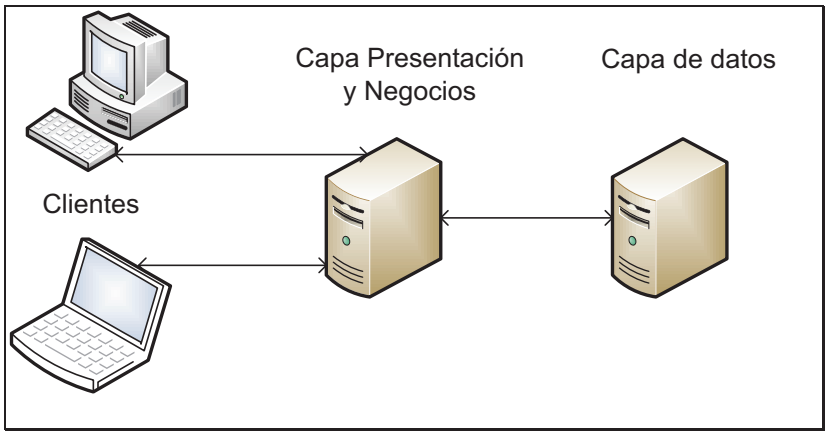

Figura 1.3 Arquitectura de dos capas [L3]

Este tipo de arquitecturas son útiles cuando no se requiere mucho procesamiento de datos. La arquitectura del servidor Web es un buen ejemplo de una arquitectura de dos capas, puesto que el navegador del cliente reside en la capa de lógica y presentación mientras que los datos del servidor Web (las páginas Web) residen en las capa de la base de datos. Otro ejemplo de aplicación en donde se emplearía normalmente una arquitectura de dos capas es una aplicación simple de entrada de datos, donde las funciones principales que ejercitan los usuarios es introducir los datos en una base de datos remota.

Por otro lado si la aplicación requiere un procesamiento considerable, se tienen cambios en la funcionalidad o está altamente orientada a sucesos específicos y

no a los datos subyacentes, es recomendable la utilización de una arquitectura de tres capas.

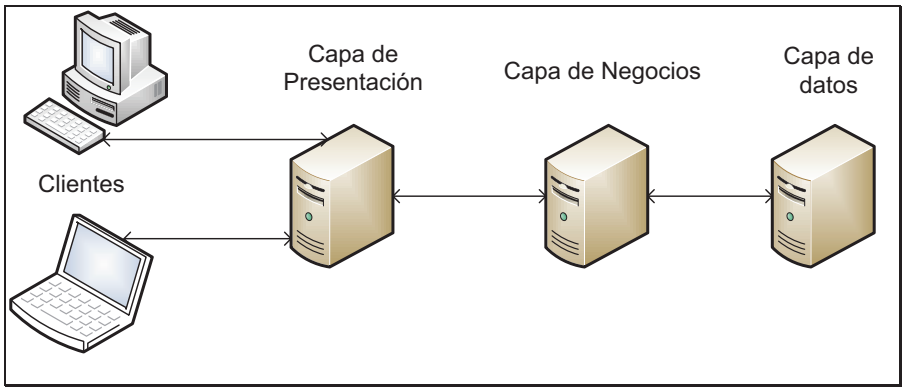

Figura 1.4 Arquitectura de tres capas [L3]

La Figura 1.4 muestra una arquitectura de tres capas. Se compone de una capa de presentación que se encarga de la interfaz con el usuario, una capa de negocios que se encarga de procesar los datos dados por el usuario y entregados por la base de datos, y una capa de base de datos que contiene los datos de la aplicación.

La capa de negocios es la que conlleva capacidad de mantenimiento y de reutilización debido a que contiene objetos definidos por clases reutilizables que se pueden utilizar una y otra vez en otras aplicaciones. Estos objetos se suelen llamar objetos de negocios y en general contienen constructores, métodos para realizar una gran gama de operaciones en comunicación con la capa de la base de datos. La capa de presentación envía mensajes a los objetos de esta capa de negocios, la cual dependiendo de la solicitud enviada responderá directamente o solicitará la información de la base de datos y la procesará antes de enviar una respuesta a la capa de presentación.

Las ventajas más notables al realizar una arquitectura de tres capas son las siguientes:

- Permite aislar a la tecnología que implementa la base de datos, de forma que sea fácil cambiar esta tecnología.

- La capa de negocios es la que realiza el procesamiento de datos manteniéndola alejada del cliente. Este código reside en un lugar (o en un número reducido de lugares si se utilizan servidores de copias de seguridad) y los cambios de mantenimiento ocurren de forma centralizada.

- La arquitectura en tres capas se acopla con las prácticas orientadas a objetos: todo el procesamiento tiene lugar por medio de los mensajes que se envían a los objetos y no mediante trozos de código asociados a cada objeto en la capa de presentación que se está ejecutando.

1.2 HERRAMIENTAS UTILIZADAS

1.2.1 MICROSOFT VISUAL STUDIO .NET [PW1, PW2]]

Microsoft Visual Studio .NET es un set completo de herramientas de desarrollo para construir aplicaciones Web ASP.NET, servicios Web XML(*eXtensible Markup Language*), aplicaciones de escritorio, y aplicaciones móviles en sistemas operativos *Windows*. Soporta los lenguajes de programación Visual Basic .NET, Visual C++ .NET, Visual C# .NET, y Visual J# .NET, los cuales utilizan el mismo entorno de desarrollo integrado IDE (*Integrated Development Environmen*), permitiendo crear soluciones de lenguaje combinado al compartir herramientas y facilidades. Además se cuenta con la funcionalidad de .NET Framework simplificando de esta manera el desarrollo de aplicaciones Web ASP y servicios Web XML. La versión más reciente es *Visual Studio 2013 Preview* acompañada *.NET Framework 4.5.1* apenas un año después del lanzamiento de la versión 2012, el cual provee compatibilidad con el desarrollo de aplicaciones para *Windows 8.1* y mejoras en la interfaz de usuarios.

Para la realización de este proyecto se utilizó la versión de *Visual Studio 2010 Professiona*l que al momento de inicio del proyecto era la versión más estable, el cual tiene entre otras las siguientes características.

- Compatibilidad con plataformas de desarrollo que incluyen Windows, Windows Server, Web, Cloud, Office y SharePoint, entre otras.

- Compatibilidad con varios monitores.
- Herramientas para pruebas dentro del entorno de desarrollo integrado (*IDE*).
- Admite el desarrollo aplicaciones para SharePoint y la Web
- Permite trabajar con múltiples versiones de .NET en un único entorno.

1.2.2 MICROSOFT .NET FRAMEWORK [L4, PW3]

La plataforma .NET es un ambiente multi-lenguaje para construir, desplegar y correr servicios Web *XML* y aplicaciones. El corazón del *.NET Framework* es el Entorno Común de Ejecución para Lenguajes (*Common Language Runtime CLR*), el cual administra el código en tiempo de ejecución y proporciona servicios centrales, como la administración de memoria, de subprocesos y la comunicación remota, además de realizar chequeos de seguridad.

Figura 1.5 El ambiente CLR [L4]

En la Figura 1.5 se muestra las dos porciones del entorno *.NET* donde se puede ver en la parte inferior el *CLR* y sobre ésta los ejecutables *CLR* o archivos Ejecutables Portátiles PE (*Portable Executable*), los cuales son unidades de despliegue que pueden ser archivos EXE o DLL (*Dynamic-Link Library*) los cuales constituyen la mayoría de la metadata y código.

La última versión estable de *.NET Framework* es la 4.5.1, sin embargo para este proyecto se utilizó la versión 4.0 incluida en *Visual Studio 2010*, la cual tiene las siguientes mejoras con respecto a las versiones anteriores:

CLR. Se han implementado mejoras en *Security, Parallel Computing*, rendimiento y diagnóstico, el nuevo DLR (*Dynamic Language Runtime*) y otros.

- *Security*: entre lo más llamativo está la simplificación y transparencia en las implementaciones.

- *Parallel Computing*: Este *Framework* incluye un nuevo modelo de programación paralela con lo cual se simplifica el desarrollo de aplicaciones multi-hilo permitiendo a los desarrolladores escribir éste código en lenguaje natural.

- *Dynamic Language Runtime*: Se encuentra incluido en el CRL para ejecución de código dinámico, el cual simplifica y facilita el desarrollo de código dinámico en *.NET*. y es el encargado de ejecutar código C# o VB en tiempo de ejecución.

Web. Las nuevas características del *framework* para aplicaciones web incluyen mejoras en *ASP.NET, Dynamic Data, Web Forms*, etc.

- *ASP.NET Web Forms*: Algunas de las mejoras que se incluyen son:
    - Posibilidad de establecer en *meta tags*.
    - Mejoras en el control del *View State*.
    - Mejoras en la generación de ID's de los controles *ASP.NET*.
    - Mejoras en el renderizado de los controles *FormView* y *ListView*.
    - *ASP.NET Chart Control*

Client. Se implementaron nuevas características en *WPF* (*Windows Presentation Foundation*) y una nueva librería llamada *MEF* (*Managed Extensibility Framework*), la cual permite construir aplicaciones extensibles utilizando metadatos sin la necesidad de cargar ensamblados para esas partes.

Data. Se tiene nuevas versiones de *Entity Framework* y *Data Services*.

- *Entity Framework*: Las nuevas características son:
    - Soporte para trabajar con clases de objetos propias (*Persistence-Ignorant Object*).

- Definición de *Foreign Keys* en el Modelo Conceptual.
- Nuevos métodos para el desarrollo de aplicaciones *N-Tier*, mejoras en la serialización a través de *WCF* y en el *Attach* y *Deatach* de los objetos.
- Soporte para desarrollar utilizando el método *Model-First*, lo cual permite desarrollar un modelo propio y obtener un script para generar nuestra base de datos.
- Nuevos tipos complejos.

1.2.3 ASP.NET FRAMEWORK [PW4]

ASP.NET es un *framework* para construir sitios web dinámicos, aplicaciones web y servicios web *XML* siendo la tecnología sucesora de la tecnología *Active Server Pages* (*ASP*); está construido sobre el *Common Language Runtime*, permitiendo a los programadores escribir código *ASP.NET* usando cualquier lenguaje admitido por el *.NET Framework*, lo cual representa una gran ayuda ya que la programación web básicamente es una mezcla de varios lenguajes de etiquetas, plataformas de servidor y un gran uso de lenguajes de script.

Las páginas de *ASP.NET*, conocidas como "*web forms*" (formularios web), son el principal medio de construcción para el desarrollo de aplicaciones web y están contenidos en archivos con una extensión *ASPX*; los cuales típicamente contienen etiquetas *HTML* (*HyperText Markup Language*) o *XHTML* (*HTML* extendido) estático, y también etiquetas definiendo Controles Web que se procesan del lado del servidor y Controles de Usuario donde se coloca todo el código estático y dinámico requerido por la página web.

Cabe mencionar que ASP.NET sólo funciona sobre el servidor de *Microsoft IIS*, lo que supone una desventaja respecto a otros lenguajes del lado de servidor, como *PHP*, *Perl* o *Python*, los cuales son ejecutables sobre otros servidores más populares como *Apache*.

1.2.4 ASP.NET AJAX [PW5]

Es un conjunto de extensiones para *ASP.NET* desarrollado por *Microsoft* para implementar la funcionalidad de *AJAX* (*Asynchronous JavaScript And XML*), las cuales permiten actualizar datos en una página web sin una recarga completa de la misma mediante componentes del lado del cliente y del servidor gracias al objeto *XMLHttpRequest*, junto con *Javascript* y *DHTML* (HTML dinámico).

El 11 de septiembre de 2006 fue lanzado como tres productos los cuales fueron llamados M*icrosoft AJAX Library*, que abarca las bibliotecas: *ASP.NET 2.0 AJAX Extensions* la cual contiene el código.*NET* del lado del servidor; *ASP.NET AJAX Control Toolkit*, que incluye controles de código compartido que pueden ser utilizados con *ASP.NET AJAX*, y *javascript.*

De estos productos, el utilizado en este proyecto fue *ASP.NET AJAX Control Toolkit*, el cual nació como un proyecto conjunto entre la comunidad de programadores y *Microsoft* y es de distribución gratuita. Contiene una serie de controles Web y extendedores con los que se puede utilizar las avanzadas características de *ASP.NET AJAX* con un simple arrastre de ratón.

Estos controles van desde un simple botón con una alerta asociada, hasta un complejo panel que se puede arrastrar por la pantalla; en ambos casos, mandando y recogiendo información entre el cliente y el servidor sin ningún tipo de recarga de página. *ASP.NET AJAX* fue liberado en enero de 2007 y finalmente fue incluido con la versión 3.5 del *.NET Framework*, incluida en *Visual Studio 2008* en noviembre de 2007.

1.2.5 CASCADING STYLE SHEETS CSS [PW6]

CSS (Hojas De Estilo En Cascada) es un lenguaje utilizado para organizar la presentación y el aspecto de una página web permitiendo elegir una multitud de opciones de presentación como colores, tipos y tamaños de letra, etc.

Su filosofía se basa en separar lo que es la estructura del documento *HTML* (el contenido) de su presentación haciendo que el contenido se vea de una forma u

otra. Usando esta filosofía, resulta muy fácil cambiarle el aspecto a una página web.

Una de sus opciones básicas es el poder cambiar el color de algunas típicas etiquetas *HTML*, pero también hay funciones más complejas, como introducir espaciado entre elementos o establecer imágenes de fondo.

Tras la aparición del lenguaje *CSS* surgió la necesidad de estandarizar su uso ya que existían muchas formas de escribir código *CSS*, además, los navegadores interpretaban algunas definiciones de estilo de distintas maneras y esto hacía más complicado el desarrollo de páginas web. El organismo encargado de la estandarización al respecto es el llamado *W3C* (*The World Wide Web Consortium*) que definió la primera versión *CSS1* en 1996, posteriormente se han desarrollado las revisiones 2, 2.1 y 3 que es la más actual.

1.3 SERVICIOS WEB [L10]

Un Servicio Web es un componente de software que se comunica con otras aplicaciones sin importar el lenguaje o la plataforma, codificando los mensajes en *XML* y enviando estos mensajes a través de protocolos estándares de Internet tales como el *Hypertext Transfer Protocol* (*HTTP*). Es decir, un Servicio Web es similar a un sitio web que en lugar de estar destinado a las personas está destinado a aplicaciones, obteniendo sus solicitudes a través de un mensaje en formato *XML*, realiza una tarea y devuelve un mensaje de respuesta también en *XML*, y al igual que una página web está definida por un *URL* (*Uniform Resource Locator*), un servicio web está definido por un *URI* (*Uniform Resource Identification*) además de su interfaz, a través del cual se puede acceder a él.

Esta tecnología se puede utilizar de muchas maneras, ya sea desde clientes de escritorio, o para la integración negocio a negocio B2B, Los servicios Web también pueden resolver el problema más amplio de la integración de aplicaciones empresariales la conexión de múltiples aplicaciones de una sola organización a múltiples aplicaciones tanto dentro como fuera del firewall. En todos estos casos, las tecnologías de servicios web proporcionan un estándar que

permite la conexión de diversas piezas de software tal como se muestra en la Figura 1.6.

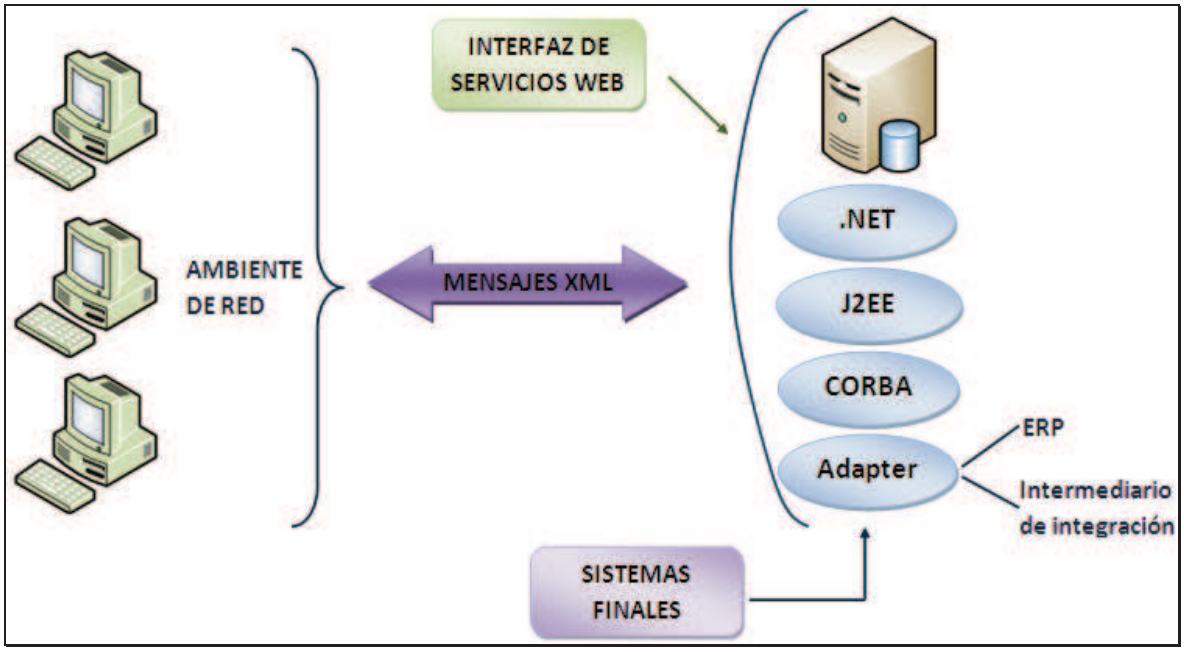

Figura 1.6 Interfaz de los Servicios Web con Sistemas Finales. [L10]

1.3.1 ARQUITECTURA DE MICROSOFT .NET WEB SERVICES FRAMEWORK [L4]

Los servicios web son componentes de software accesible a través de protocolos web estándar, los cuales permiten inter-operar con un amplio rango de clientes, los cuales simplemente necesitan comprender como analizar un flujo en formato *XML* transmitido a través de canales *HTTP*. La clave de la tecnología usada en los Servicios Web es *XML*, y es usada en las áreas de *Microsoft .NET Web Services Framework* descritas en los siguientes puntos.

1.3.1.1 Web Service wire formats

Es la tecnología que permite conocer cómo realizar el intercambio de datos entre el proveedor de servicio y el consumidor; es decir el formato de los datos para la solicitud y la respuesta.

Los Servicios Web de *Microsoft .NET* soportan tres protocolos: *HTTP GET, HTTP POST*, y *SOAP* (*Simple Object Access Protocol*), descritos a continuación. Estos

protocolos facilitan a las aplicaciones clientes el uso de los servicios, debido a que son estándares para la web.

- ***HTTP GET* y *HTTP POST*:** Estos dos métodos utilizan *HTTP* como su protocolo subyacente y codifican los parámetros de solicitud como pares nombre/valor en la solicitud *HTTP*. El método *GET* crea un flujo de solicitud y lo ajunta al *script* del *URL* del servidor, el cual se encarga de la respuesta. Para el método *POST*, los pares nombre/valor son pasados en el cuerpo del mensaje de respuesta *HTTP*.

- ***SOAP*:** Este método es similar a los explicados anteriormente, con la diferencia que utiliza formato *XML* para los mensajes de solicitud y respuesta, por lo que es extensible ya que está basado en *XML*, y posee una mejor estructura permitiéndole cubrir información más compleja que con los pares nombre/valor de los protocolos *HTTP GET/POST*. La especificación *SOAP* describe el formato *XML* de esas solicitudes y respuestas.

1.3.1.2 Descripción del Servicio Web en WSDL (Web Services Description Language).

Es un documento en esquema *XML* que describe las llamadas a los métodos y los parámetros de entrada y salida, o las interfaces que soporta el Servicio Web. El lenguaje describe como puede ser usado el servicio, con lo que cualquier cliente *SOAP* capaz de usar *WSDL* puede usar este archivo para comprender la interfaz e invocar los métodos del Servicio Web.

La parte fundamental de este documento es el elemento *definitions*, el cual provee una descripción abstracta y concreta del Servicio Web, y tiene los siguientes elementos:

Types: Es un contenedor para definiciones de tipo de datos.

Message: Es una definición resumida de los datos a ser intercambiados entre los clientes y los proveedores de Servicios Web. Cada método tiene dos mensajes:

Entrada que describe los parámetros para el método web, y Salida que describe los datos de retorno desde el método web.

- ***Port type:*** Es un conjunto resumido de operaciones soportadas por uno o más puntos finales.
- ***Operation:*** Es una descripción resumida de una acción soportada por un servicio. Cada operación especifica los mensajes de entrada y de salida.
- ***Binding:*** Es una especificación para un protocolo concreto y formato de dato para un tipo de puerto particular. Contiene operaciones, así como la entrada y la salida para cada operación, con la diferencia de que se habla del tipo de transporte concreto, y el formato de las entradas y salidas.
- ***Service:*** Es una colección de puntos finales-puertos de red. Cada método (HTTP GET/POST y SOAP) constituye un puerto de servicio.
- ***Port:*** Describe la especificación de protocolo y formato de datos a ser usada, así como la dirección de red donde los clientes de los Servicios Web pueden unirlos a los servicios.

Es posible leer y construir el *WSDL* manualmente, es posible autogenerarlo con otras herramientas como Microsoft Visual Studio.

1.3.1.3 Web Service Discovery

Es el proceso que permite encontrar un determinado servicio y verificar su funcionamiento. Hay dos formas de publicar el servicio: estática y dinámica, en las cuales XML conlleva la localización de los Servicios Web.

- ***Publicación Estática:*** Se crea un archivo *.disco* donde se especifica explícitamente la URL para todos los Servicios Web que se ofrece.
- ***Publicación Dinámica:*** Permite listar automáticamente todos los Servicios Web bajo una URL específica en el sitio web. Esto es útil para agrupar varios Servicios Web de acuerdo al tipo en diferentes directorios, y luego proveer un único archivo *dynamic discovery* en cada directorio.

1.3.2 EL NAMESPACE System.Web.Services

El *SDK* de *.NET* proporciona unas pocas clases en el *namespace Sustem.Web.Services*, de las cuales las de uso general más importantes son los siguientes:

- ***WebService.*** Es la clase base para todos los Servicios Web, la cual provee propiedades inherentes a la programación en ASP.
- ***WebServiceAttribute.*** Se usa para proveer más atributos sobre el Servicio Web en sí mismo, es decir se puede mostrar una descripción, así como el *namespace* al cual pertenece el Servicio Web.
- ***WebMethodAttribute.*** Permite aplicar atributos a cada método *public* del Servicio Web, asignando valores a algunos atributos.

1.3.3 CONSIDERACIONES DE DISEÑO DE SERVICIOS WEB [L11]

Los siguientes son unos cuantos pasos que se deben considerar al momento de escribir Servicios Web robustos y de alto rendimiento rápidamente.

- **Hacerlo conciso.** Un Servicio Web puede devolver solo un objeto como valor de retorno, sin embargo este puede ser cualquier objeto *.NET* que se consiga serializar en *XML*, es decir cualquier objeto que tenga un constructor por defecto (que no requiere parámetros), por lo que es posible regresar cualquier cantidad de datos como respuesta de una solicitud.

  También se debe tener en cuenta que al llamar a un Servicio Web la sobrecarga (*overhead*) de cada llamada es bastante alta, por consiguiente, se debe considerar qué cantidad de datos realmente se necesita regresar y cuantas llamadas se debe hacer para obtenerlos, sabiendo que por cada llamada se tendrá sobrecarga, y que el tiempo transcurrido al hacer una llamada es aproximadamente el mismo para unos pocos bytes que el transcurrido para unos cuantos miles de bytes.

- **Pensar cuidadosamente sobre su estado.** Los objetos en los servicios web por naturaleza no tienen estado, es decir se crea una instancia del

objeto para cada llamada y se la destruye al final de la llamada, esto es debido a que en la mayoría de casos cada llamada es autosuficiente y no hay razón de mantener nada de una a otra; sin embargo si se necesita mantener el estado llamadas se lo puede hacer con las colecciones de estado en ASP.NET *Session* y *Application* los cuales por defecto están en desactivadas, por lo que se debe activarlas explícitamente para cada método que requiera un estado de sesión poniendo un parámetro a la declaración del atributo *WebMethod.*

Se debe considerar que el uso excesivo de la colección *sesión* puede limitar la escalabilidad del sistema debido a que se necesita tiempo para procesar cada llamada que requiera un objeto *sesión* y a que se necesita espacio en disco para mantener la información entre llamada y llamada para cada cliente. Por otro lado se debe tener en cuenta el tiempo de vida del estado; para páginas que son visitadas casualmente por usuarios humanos es recomendable extender el tiempo de vida en el servidor, pero para funciones llamadas por otros programas es mejor mantener todos los estados en los clientes.

Por supuesto estos dos métodos tienen sus desventajas: en el primero se puede estar almacenando innecesariamente grandes cantidades de información para clientes que en realidad ya han terminado sus asuntos y se han ido, y en el segundo toda la información de estado debe viajar por la red, lo cual incrementa los requerimientos de ancho de banda. Cabe señalar que si se está comunicando entre sistemas Microsoft y se necesita mucho trabajo para mantener los estados, una mejor solución sería utilizar la técnica *.NET Remoting*.

- **Manejo de excepciones.** Para el manejo de excepciones.*NET* provee un manejo de excepciones estructurado, sin embargo esto solo funciona para clientes *.NET*, para otro tipo de clientes se utiliza un elemento *Fault* provisto por *SOAP*. Se debe tener en cuento que si el cliente no accede al servicio mediante éste método sino por *GET/POST* se debe usar valores de retorno especiales para enviar información útil sobre el error.

Cabe mencionar que en la actualidad se está utilizando *WCF* (*Windows Comunication Foundation*) para el desarrollo de aplicaciones distribuidas, el cual fue lanzado como parte del **.***NET Framework 3.0*.

1.4 PROTOCOLOS DE SEGURIDAD [PW7]

1.4.1 IPSEC

IPSec (*Internet Protocol Security*) es un conjunto de estándares del *IETF* (*Internet Engineering Task Force*) que incorpora servicios de seguridad en *IP* para satisfacer la necesidad de garantizar seguridad necesaria para las comunicaciones B2B y en el comercio electrónico.

1.4.1.1 Introducción

IPSec proporciona servicios de seguridad a la capa *IP* así como a todos los protocolos superiores basados en *IP* (*TCP* y *UDP*, entre otros) y aborda las carencias de seguridad del protocolo *IP*, las cuales son muy graves y afectan a la infraestructura misma de las redes *IP*.

Solucionó el problema de interoperabilidad que había con las soluciones anteriores ya éstas eran de tipo propietario por lo que los distintos entornos empresariales necesitaban contar de una misma plataforma. Al estar apoyado en estándares del *IETF* proporciona un nivel de seguridad común y homogéneo para todas las aplicaciones, así como independencia de la tecnología física empleada. *IPSec* se incluye por defecto en la versión actual de *IP* (*IPv6*) pero se integró desde *IPv4*.

Para que las redes IP se puedan desarrollar es necesario cubrir el requisito indispensable de la seguridad, por lo que casi todos los equipos de comunicaciones ya incorporan este estándar, así como las últimas versiones de los sistemas operativos más comunes, dándole de esta manera un gran apoyo. Al mismo tiempo se ha demostrado la interoperabilidad entre fabricantes, lo cual constituye una garantía para los usuarios.

Otra característica sobresaliente es su carácter de estándar abierto que se acopla perfectamente con la tecnología *PKI* (*Public-Key Infrastructure*), al establecer ciertos algoritmos comunes, pero por razones de interoperabilidad permite integrar algoritmos criptográficos más robustos que puedan desarrollarse próximamente.

Entre los principales beneficios que aporta IPsec están:

- Posibilitar nuevas aplicaciones como el acceso seguro y transparente de un nodo *IP* remoto.
- Facilitar el comercio electrónico de negocio a negocio, ya que brinda una infraestructura segura sobre la que se pueda realizar transacciones usando cualquier aplicación, como por ejemplo las *extranets*.
- Permitir la construcción de una red corporativa segura sobre redes públicas, eliminando la gestión y el coste de líneas dedicadas.
- Ofrecer a usuarios remotos el mismo nivel de confidencialidad que con una red local, eliminando la limitación de acceso a la información sensible por problemas confidencialidad.

Cabe recalcar que la palabra "seguro" no se refiere únicamente a la confidencialidad de la comunicación, sino también a la integridad de los datos, que dependiendo del entorno, puede ser un requisito mucho más crítico que la confidencialidad, la cual es proporcionada por *IPSec* como servicio añadido al cifrado de datos o como servicio independiente.

1.4.1.2 Descripción de IPSec

IPSec se encarga de brindar funciones de seguridad sobre *IP* como confidencialidad, integridad y autenticidad de datagramas *IP*, combinando tecnologías de clave pública (RSA), algoritmos de cifrado (DES, 3DES, IDEA, *Blowfish*), algoritmos de *hash* (MD5, SHA-1) y certificados digitales X509v3, tal como se puede ver en la Figura 1.7, donde se muestra la complementación de varias técnicas dentro de *IPSsec.*

En el protocolo *IPSec* se puede seleccionar un conjunto de algoritmos deseados sin afectar a otras partes de la implementación, aunque para asegurar la interoperabilidad dentro de Internet, se deben soportar algoritmos de referencia: *DES* y *3DES* para cifrado, y *MD5* y *SHA-1*, como funciones de *hash*; permitiendo además compatibilidad con otros algoritmos que para un entorno específico se consideren más seguros o adecuados.

Figura 1.7 Tecnologías utilizadas en IPSec [PW7]

En *IPSec* se distinguen los siguientes componentes:

- Mecanismos de seguridad para proteger tráfico *IP*: Protocolo *IP Authentication Header* (AH), y el Protocolo *IP Encapsulating Security Payload* (ESP).
- Un protocolo de gestión de claves *IKE* (*Internet Key Exchange*) que permite a dos nodos negociar las claves y todos los parámetros necesarios para establecer una conexión *AH* o *ESP*.

1.4.1.3 Protocolo AH

El *protocolo AH* proporciona al receptor procedimientos para garantizar la integridad de los datos verificando que no hayan sido alterados en el transcurso, y la autenticación del origen de los datagramas, mas no su confidencialidad, es decir la información puede ser vista por terceros.

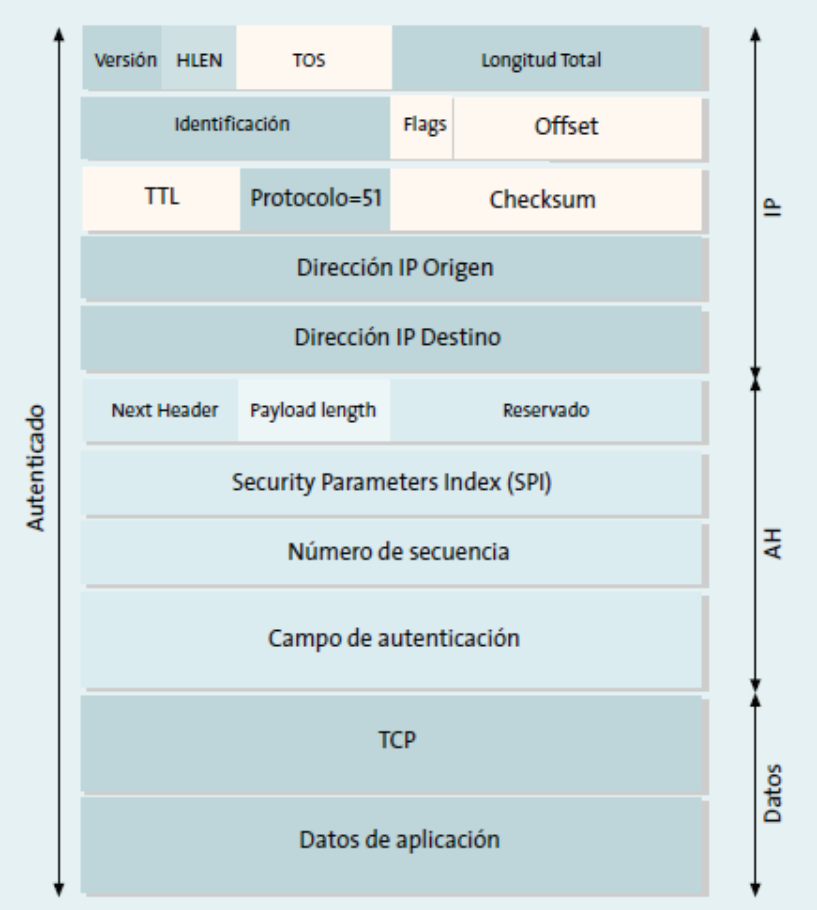

Figura 1.8 Estructura de un Datagrama AH [PW7]

El protocolo *AH* inserta una cabecera de autenticación entre la cabecera *IP* estándar de versión 4 o 6, y los datos transportados, pudiendo ser estos un mensaje *TCP, UDP o ICMP*, o incluso un datagrama *IP* completo tal como se muestra en la Figura 1.8. En la cabecera IP se coloca el valor 51 en el campo *Protocolo* en lugar de los valores 6 ó 17 asociados a *TCP* y *UDP* respectivamente, ya que éste es el valor asignado por el IANA (*Internet Assigned Numbers Authority*), mientras que la naturaleza de los datos de capa superior se indican en el campo *Next Header* de la cabecera AH. Cabe mencionar que la integridad y autenticidad de los datos transportados y de la cabecera IP, excepto los campos variables resaltados en la Figura 1.8.

Su funcionamiento se basa en un código de autenticación de mensajes mediante el algoritmo *HMAC* el cual consiste en aplicar una función *hash* a los datos de entrada y una clave, obteniéndose una cadena reducida de caracteres denominada *hash* o extracto, la cual es como la huella digital del mensaje ya que

es única y solo se puede generar con la clave que es de conocimiento exclusivo del emisor y del receptor. El procedimiento aplicado por este protocolo se muestra en la Figura 1.9 y es el siguiente: El emisor calcula un extracto del mensaje original, el cual se copia en el campo de autenticación de la cabecera *AH*. Este paquete se envía a través de la red, y en el extremo receptor se repite el cálculo del extracto comparándolo con el del paquete recibido. De ser iguales significa que el paquete no ha sido modificado en el viaje y que es de quien dice haberlo enviado; por lo tanto la seguridad del protocolo reside en el secreto de la clave.

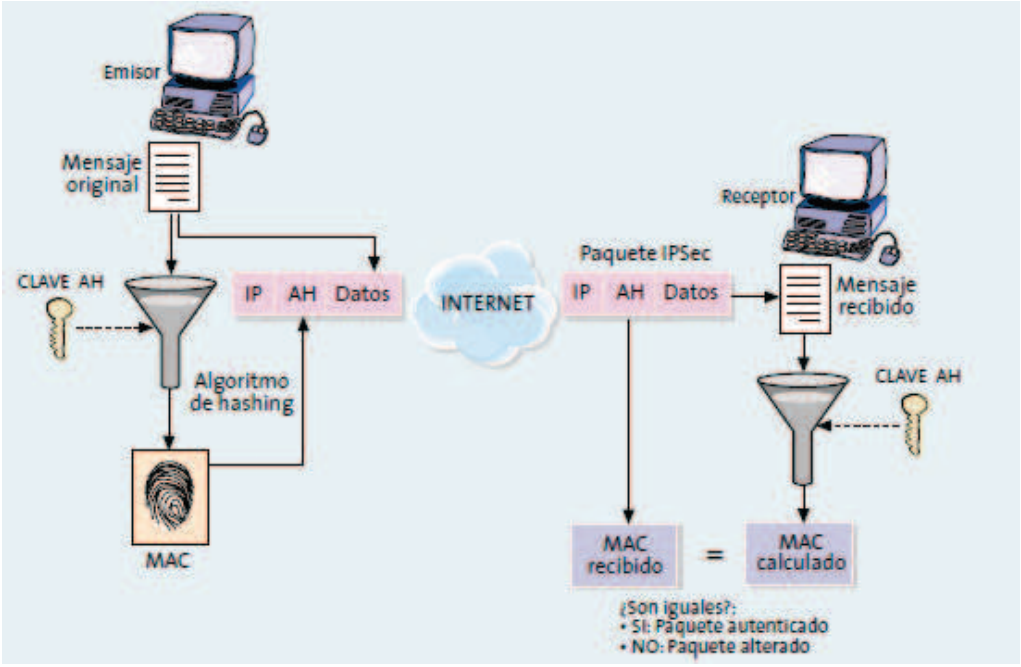

Figura 1.9 Funcionamiento del protocolo AH [PW7]

1.4.1.4 Protocolo ESP

Su principal objetivo es proporcionar confidencialidad lo cual logra cifrando los datos que se desean, aunque adicionalmente puede ofrecer integridad y autenticación del origen de los datos mediante un procedimiento parecido al del protocolo *AH*.

El formato de la cabecera *ESP* es más complejo que el de AH debido a que brinda más servicios; el cual tiene de una cabecera y una cola que encapsulan los datos transportado como se muestra en la Figura 1.10. Los datos encapsulados puede ser cualquier protocolo *IP*. El número asignado por el *IANA* es el 50 por lo

que éste es el número en el campo *Protocolo* de la cabecera *IP*, mientras que la naturaleza de los datos de capa superior se indican en el campo *Next Header* el cual al estar encapsulado hace imposible que un atacante conozca qué tipo de contenido se está transportando.

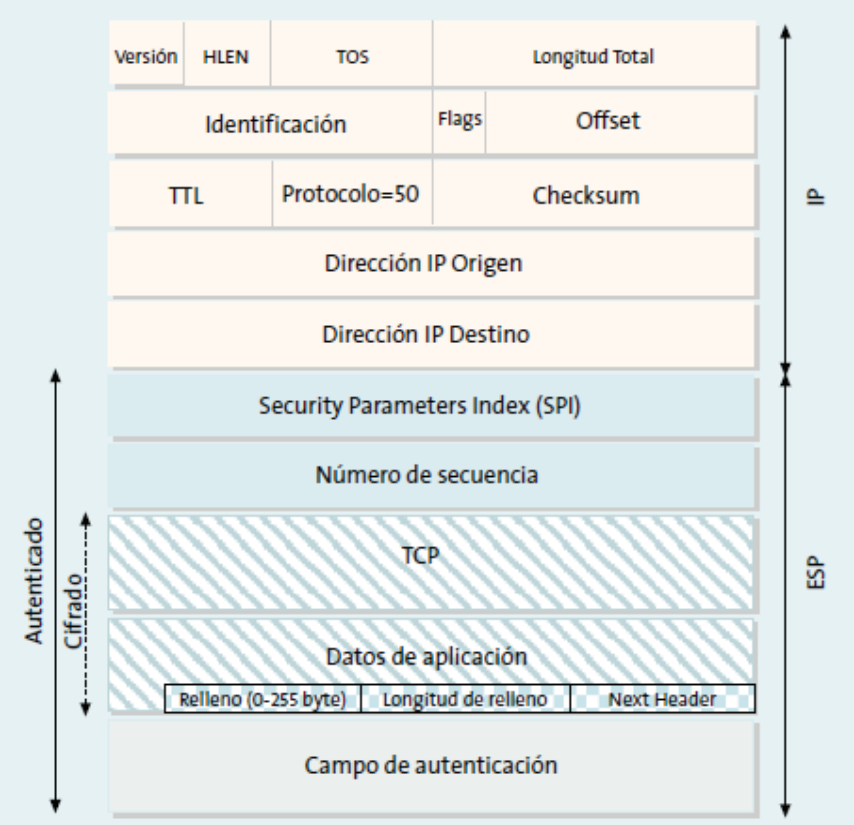

Figura 1.10 Estructura de un Datagrama ESP [PW7]

Este protocolo realiza el cifrado por un algoritmo de clave simétrica. Al utilizarse algoritmos de cifrado de bloque la longitud de los datos a cifrar debe ser un múltiplo del tamaño de bloque que en general es de 8 o 16 bytes, por lo que existe un campo de relleno, como se observa en la Figura 1.10, el cual también sirve para añadir caracteres de relleno al campo de datos, ocultando su longitud real y, por tanto, las características del tráfico evitando ataques con información que se pueda deducir como el retardo entre paquetes y su longitud.

El procedimiento de este protocolo es el siguiente: el emisor cifra el mensaje original con una clave determinada y lo coloca después de la cabecera *ESP*. Al estar encriptado, si un atacante intercepta un paquete solo obtendrá un mensaje que no podrá interpretar; mientras que el receptor aplicará nuevamente el

algoritmo de cifrado para recuperar descifrar la información. Este proceso se muestra en la Figura 1.11.

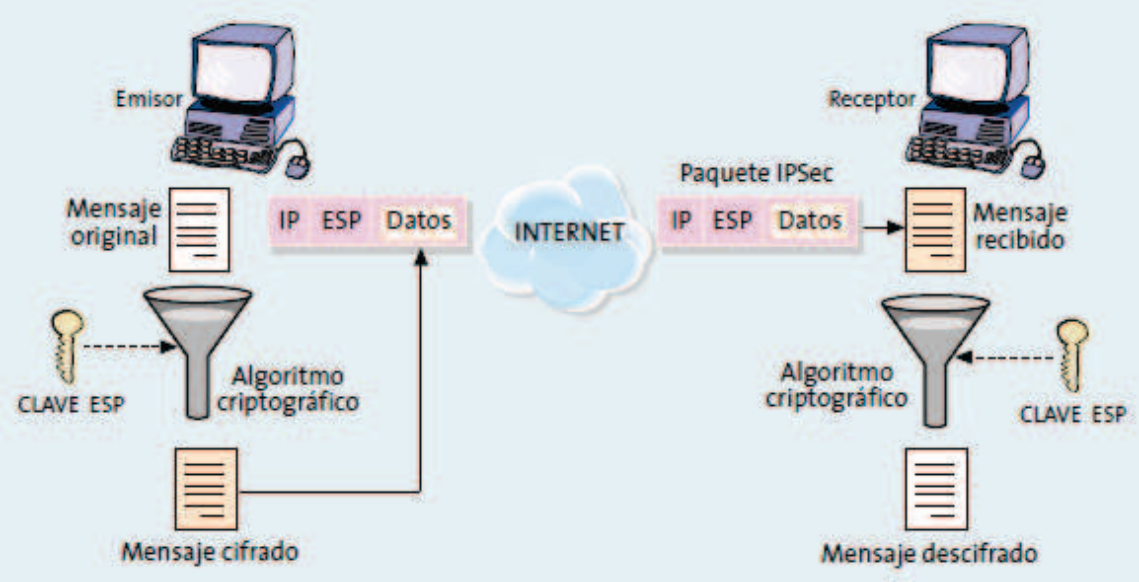

Figura 1.11 Funcionamiento del protocolo ESP [PW7]

Se puede observar que la seguridad de este protocolo, al igual que en *AH* reside en el secreto de la clave, por lo tanto es un requisito fundamental la distribución de claves de forma segura, además de estar de acuerdo en el algoritmo de cifrado y otros parámetros comunes.

1.4.1.5 Los modos transporte y túnel

Los dos modos de funcionamiento permitidos por *IPSec* tanto con *ESP* como *AH* se muestran en la Figura 1.12, y se explican a continuación:

- **El modo transporte.** Este modo asegura la comunicación de extremo a extremo siendo necesario que ambos extremos entiendan el protocolo *IPSec*. En este modo se inserta la cabecera *IPSec* entre la cabecera IP y los datos de las capas superiores a ser protegidas. El contenido es transportado dentro del datagrama *AH* o *ESP* y son datos de la capa de transporte como por ejemplo datos *TCP* o *UDP*.

- **El modo túnel.** Este método se usa normalmente cuando el destino final de los datos no es el que realiza las funciones *IPSec.* Aquí se toma un

datagrama *IP* incluida su cabecera *IP* y se le añade una cabecera *ESP* o *AH* y luego se le añade una nueva cabecera *IP* para encaminar el paquete en la red.

Este modo es empleado principalmente para identificar la red a proteger bajo una misma dirección *IP* y centralizar el procesamiento del tráfico *IPSec* en un equipo, por ejemplo en los *gateways IPSec*. También es útil, para ocultar la identidad de los nodos que se están comunicando utilizándolo junto con el protocolo *ESP*. Además se lo puede utilizar para establecer Redes Privadas Virtuales (RPV) a través de redes públicas (Internet), con lo que se puede interconectar de forma segura redes de área local en ambientes inseguros.

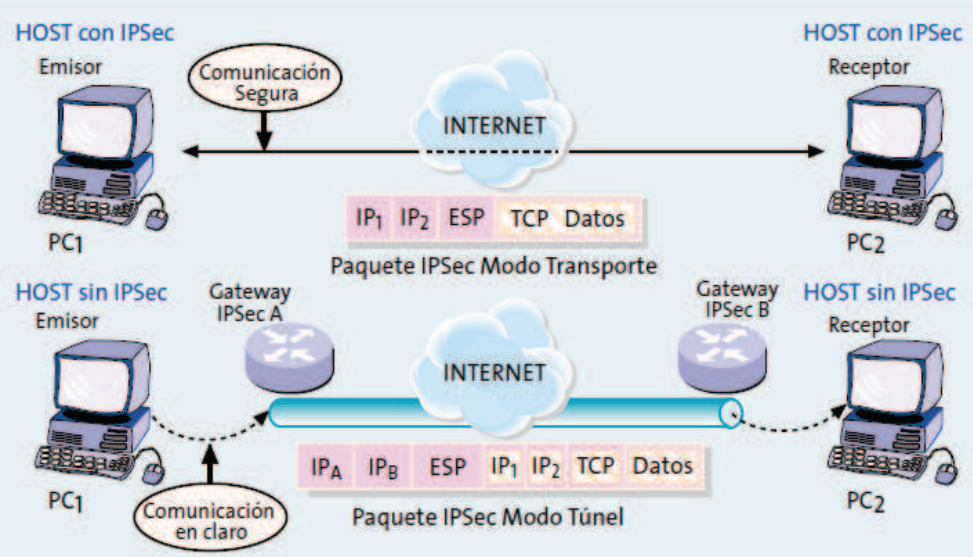

Figura 1.12 Los modos de funcionamiento: transporte y túnel de IPSec [PW7]

1.4.1.6 IKE: el protocolo de control

Una conexión *IPSec* se compone de dos asociaciones de seguridad (*SA*), la cual es un canal de comunicación unidireccional que conecta dos nodos, y en el que fluyen los datagramas protegidos mediante mecanismos criptográficos acordados previamente. Como se indicó anteriormente es necesario que ambos nodos estén de acuerdo tanto en los algoritmos criptográficos a emplear como en los parámetros de control, lo cual se puede lograr con una operación llamada negociación de SAs, el cual puede utilizar una configuración manual, o algún

protocolo de control que se encargue de la negociación automática de los parámetros necesarios

El *IETF* ha definido el protocolo *IKE* para realizar tanto la función de gestión automática de claves como el establecimiento de las *SAs* correspondientes; el cual es el resultado de la integración de dos protocolos complementarios: *ISAKMP* (*Internet Security Association and Key Management Protocol*) y *Oakley*: el primero define de forma genérica el protocolo de comunicación y la sintaxis de los mensajes que se utilizan en *IKE*, mientras que *Oakley* especifica la lógica de cómo se realiza de forma segura el intercambio de una clave entre dos partes que no se conocen previamente.

El objetivo principal de *IKE* consiste es ofrecer autenticación y confidencialidad a lo comunicación entre dos entidades, mediante una conexión a través de la cual se negocian los parámetros necesarios para establecer una asociación de seguridad *IPSec* y se la lleva a cabo en dos fases:

1. La fase común a cualquier aplicación, en la que ambos nodos establecen un canal seguro y autenticado mediante el uso de un algoritmo de cifrado simétrico y un algoritmo *HMAC,* mientras que las claves necesarias se obtienen a partir de una clave maestra que se obtiene mediante un algoritmo de intercambio de claves *Diffie-Hellman*, el cual al no garantizar la identidad de los nodos se necesita un paso de autenticación adicional, siendo los más comunes los siguientes:

   El primer método de autenticación se basa en un secreto conocido por cada par de extremos, el cual consiste en una cadena de caracteres que se utiliza en la función *hash* realizada en cada extremo para conseguir una autenticación mutua, ya que de esta forma se demuestra el conocimiento de dicho secreto sin necesidad de revelarlo. Debido a razones de seguridad es necesario que cada par de nodos tengan un secreto distinto, por lo que en entornos donde se necesite conectar muchos nodos *IPSec* es recomendable utilizar una autenticación basada en certificados digitales, caso contrario la gestión de claves se torna muy complicada. Para este

propósito se utiliza el segundo método en el cual se tiene certificados digitales X509v3 junto con una *PKI* (*Public-Key Infrastructure*), lo cual permite la distribución segura de la clave pública de cada nodo, permitiendo comprobar su identidad mediante la posesión de la clave privada y ciertas operaciones de criptografía pública.

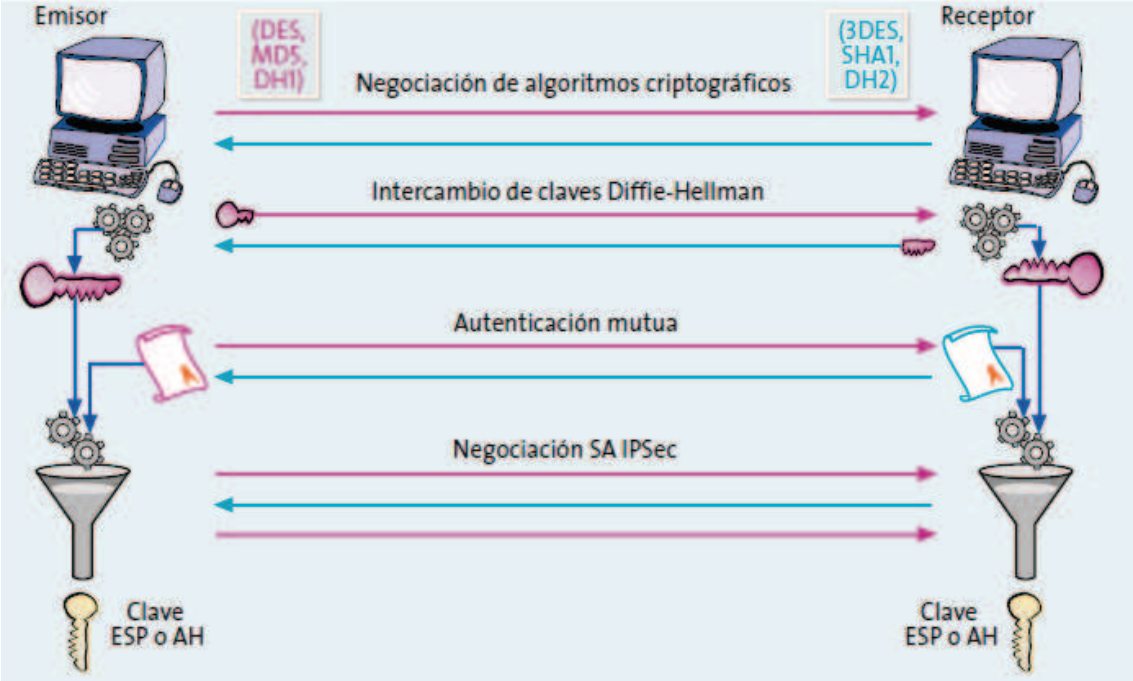

Figura 1.13 Funcionamiento del protocolo *IKE* [PW7]

2. En esta fase se usa el canal seguro *IKE* para negociar los parámetros de seguridad las características de la conexión *ESP* o *AH*, y todos los parámetros necesarios de la siguiente forma: El equipo que inicia la comunicación ofrece las opciones que tiene configuradas en su política de seguridad de acuerdo a su prioridad, luego el receptor acepta la primera opción que coincida con sus parámetros de seguridad; de igual forma se informa el tráfico a intercambiarse en la conexión entre los nodos.

En la Figura 1.13 se muestra de forma esquemática el funcionamiento del protocolo antes explicado así como la obtención de una clave de sesión utilizada para proteger las conexiones sean éstas *ESP* o *AH*.

1.4.1.7 Integración de IPSec con una PKI

La *PKI* aparece como respuesta a la necesidad de autenticar de modo fiable un conjunto numeroso de nodos mediante *IPSec* ya que centraliza el control de usuarios que se unen o se desligan del sistema, además de posibilitar la introducción de tarjetas inteligentes que soporten certificados digitales.

En *IPSec* la *PKI* engloba los nodos y los procedimientos utilizados para autenticar dichos nodos. Para esto se requiere que cada dispositivo disponga de un certificado digital que contiene su clave pública e información para identificar de forma particular al dispositivo por ejemplo su nombre *DNS*, toda esta información está avalada por la firma de la Autoridad Certificadora (*Certification Authority CA*) integrada en la *PKI* siendo ésta reconocida como válida por todos dispositivos los cuales dispondrán de una copia del certificado de la *CA*. Aunque no están especificados los protocolos para la comunicación entre los dispositivos y la *PKI* existiendo varias alternativas dependiendo del fabricante, se puede decir que todos utilizan el formato *X.509v3* para los certificados, y los estándares de la serie *PKCS* (*Public-Key Cryptography Standards*) para la solicitud y descarga de certificados.

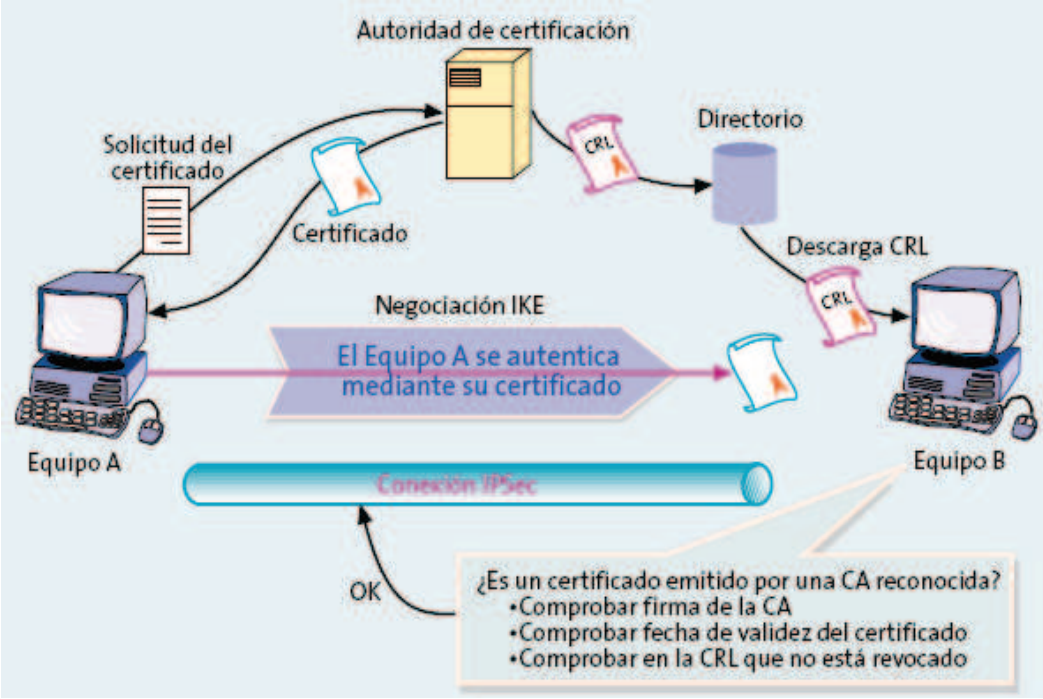

Figura 1.14 Integración de una PKI en IPSec [PW7]

Las operaciones que generalmente realizan los nodos *IPSec* con la *PKI* son: acceso al certificado de la *CA*, solicitud y descarga de un certificado, y comprobación de la validez del mismo. Para la solicitud y descarga del certificado se utiliza el estándar *SCEP* desarrollado originalmente por *Cisco* y *Verisign* el cual se basa en el intercambio de mensajes *PKCS* mediante el protocolo *HTTP*.

Por otro lado la validación de los certificados generalmente se los realiza mediante consultas de la Lista de Certificados Revocados (*CRL*) almacenada en el directorio de la *PKI*, de la cual tienen una copia todos los nodos y se actualiza periódicamente mediante una consulta *LDAP*.

Los flujos de comunicación que existen entre un nodo *IPSec* y la *PKI* se muestran en la Figura 1.14.

1.4.1.8 Servicios de Seguridad Ofrecidos por IPSec

Las características de los servicios de seguridad que ofrece *IPSec* son los descritos a continuación:

- **Integridad y autenticación del origen de los datos.** En el caso de que no requerir cifrado, el protocolo *AH* es el más adecuado ya que la protección que brinda incluye a la cabecera *IP,* a diferencia del protocolo ESP. Esta opción es de gran importancia para aquellas aplicaciones en las cuales es importante garantizar que el contenido de los paquetes *IP* no haya sido alterado en el transcurso.

- **Confidencialidad.** Este servicio se obtiene mediante la función de cifrado del protocolo *ESP*. Siempre es recomendable activar la opción de autenticación, ya que si bien un paquete interceptado no se pude interpretar por alguien que no conozca la clave, bien podría ser alterado y enviado al receptor.

- **Detección de repeticiones.** Este servicio soportado por *ESP* o *AH se* utiliza para detectar y rechazar paquetes repetidos mediante un número de secuencia incluido en sus cabeceras, el cual es incrementado por cada

datagrama enviado, el cual no puede ser modificado debido a que está protegido por la opción de integridad de los protocolos, generándose un error si fuera modificado.

- **Control de acceso: autenticación y autorización.** Las claves son distribuidas de modo seguro mediante una sesión *IKE*, garantizándose la correcta autenticación, sin embargo esto no conlleva acceso total a todos los recursos, puesto que *IPSec* proporciona funciones de autorización establecidas durante la negociación *IKE* en la que se especifica el flujo de tráfico *IP* que existirá en dicha conexión, considerándose el protocolo, las direcciones *IP* de los puertos origen y destino, el byte "*TOS*" y otros campos.

- **No repudio.** Este servicio se lo conseguiría al utilizar *IKE* ya que la autenticación se basa en la firma digital el cual gracias al vínculo entre la clave pública y la identidad que garantiza dicho certificado, es una prueba contundente de la conexión *IPSec* establecida es con los equipos correctos por lo que no se podrá negarla. Sin embargo cabe señalar que si bien este procedimiento es técnicamente posible, en la práctica se requeriría almacenar los mensajes de negociación IKE haciendo que sea muy compleja.

1.4.2 PROTOCOLO HTTPS [PW8]

HTTPS (*HTTP* sobre *SSL* o *Secure HTTP*) es un protocolo que fue desarrollado por *Netscape*, el cual *hace* uso de *Secure Socket Layer* (*SSL*) o *Transport Layer Security* (*TLS*) como una subcapa bajo la capa regular de Aplicación HTTP mediante el puerto 443, encriptando y desencriptando páginas solicitadas por el usuario, así como las respuestas del servidor Web; esto permite proteger del espionaje y de ataques de hombre en la mitad.

Al usar este protocolo, la información que se envía desde un formulario web, el navegador cifra la información, de igual manera la respuesta del servidor viajará encriptada, siendo el navegador el responsable de desencriptar la información.

Tanto *HTTPS* como *SSL* soportan el uso de certificados digitales X.509 desde el servidor, por lo que un usuario puede autenticar al emisor de ser necesario.

La eficacia de *HTTPS* puede estar limitada debido a la pobre implementación de software en el navegador o en el servidor, además del hecho que no soporta algunos algoritmos. *HTTPS* no se debe confundir con *S-HTTP*, el cual fue desarrollado y propuesto por el *EIT* como un estándar que brinda seguridad mejorada de *HTTP*.

1.4.2.1 Protocolo SSL [PW9]

La primera versión fue desarrollada por *Netscape* en 1994, aunque jamás fue implementada de forma pública, se publicó una nueva versión llamada *SSL 2.0* la cual a pesar de tener graves errores de diseño sí tuvo una implementación real. La versión 3.0 es el estándar que se utilizó por largo tiempo para crear un canal de comunicaciones seguro entre clientes y servidores en internet, desde su lanzamiento en noviembre de 1995; el cual es independiente del sistema operativo utilizado por ambos extremos, a la vez da la posibilidad de adaptarse a los nuevos adelantos de cifrado a medida que fueran saliendo, sin embargo cabe señalar que no fue creado para satisfacer las necesidades específicas del comercio electrónico, sino más bien como un protocolo seguro de propósito general.

Opera entre la capa de transporte y la de sesión del modelo *OSI* (o entre la capa de transporte y la de aplicación del modelo *TCP*) por debajo de protocolos de aplicación como *HTTP*, *IMAP*, *LDAP*, etc., pudiendo ser usado por todos ellos de forma transparente para el usuario. Está formado por dos capas y cuatro componentes bien diferenciados, los cuales se muestran en la Figura 1.15 y se explican a continuación:

- El protocolo de registro (*Record Protocol*) encargado de construir un canal de comunicaciones entre dos extremos, lo cual consigue encapsulando el trabajo de los elementos de la capa superior.

- El protocolo *Handshake* es el encargado de intercambiar la clave que se utilizará para crear un canal seguro mediante un algoritmo eficiente de

cifrado simétrico; además coordina los estados de ambos extremos de la transmisión. Es por estas razones que es la parte fundamental de SSL.

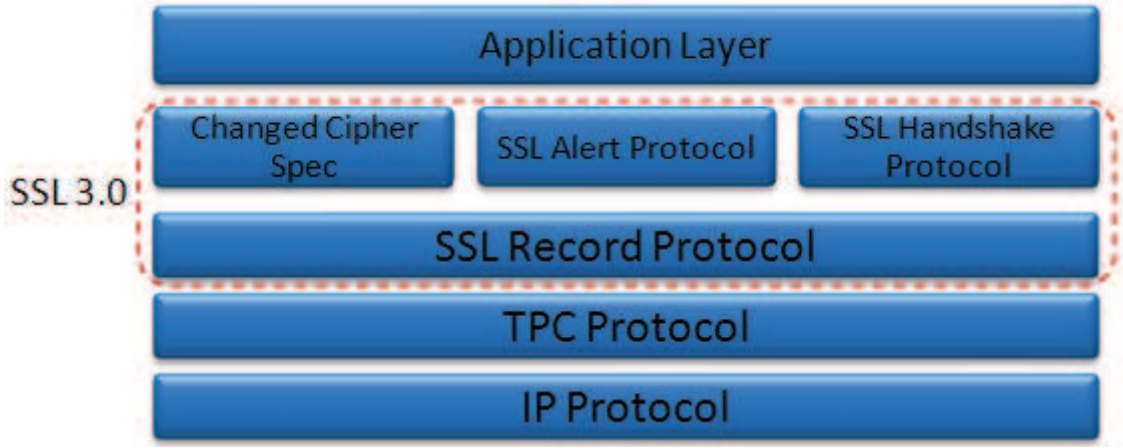

Figura 1.15 Componentes de SSL [PW9]

- El protocolo de Alerta es el encargado de señalizar problemas y errores concernientes a la sesión *SSL* establecida.
- El protocolo *Change Cipher Spec* se utiliza para notificar un cambio en la estrategia de cifrado y está formado por un único mensaje consistente en un solo byte de valor 1.

En general los pasos que se siguen para crear un canal seguro mediante SSL son los siguientes:

- Intercambiar una clave de longitud suficiente mediante un algoritmo de cifrado asimétrico.
- Establecer un canal seguro utilizando clave anterior mediante algoritmo simétrico previamente negociado.
- Fragmentar en bloques los mensajes a ser transmitidos, comprimirlos, aplicarles un algoritmo hash para obtener un resumen (*MAC*) y concatenarlo a cada uno de los bloques antes comprimidos para asegurar la integridad de los mismos.
- Realizar el cifrado y enviar los resultados.

Se tiene una máquina de control de estados para todas esas operaciones; cabe mencionar que una sesión *SSL* puede comprender múltiples conexiones, además de la capacidad de establecer múltiples sesiones *SSL* simultáneas.

Para negociar los atributos de sesión, el protocolo de *Handshake* sigue los pasos mostrados en la figura siguientes pasos.

- El cliente envía un mensaje *Client Hello* al servidor el cual responde con un mensaje *Server Hello* similar, en el cual dan a conocer algunas características como: la versión del protocolo usado, algoritmos de cifrado conocidos y preferidos, longitudes máximas de clave que admite para cada uno de ellos, funciones hash y métodos de compresión a utilizar. El servidor además asigna un identificador a la sesión en el que consta la fecha y hora de la misma, el cual es enviado al cliente en el mensaje de *Server Hello.*

  Si el servidor no responde con un mensaje de *Server Hello,* o no es válido o reconocible la sesión abortaría inmediatamente. Al ser el servidor el que generalmente recibe la solicitud, elige los algoritmos más fuertes de entre los soportados por el cliente, donde de no llegar a un acuerdo se envía un mensaje de error y se aborta la sesión.

- Luego del mensaje de *Server Hello*, el servidor puede enviar su Certificado con el objetivo de ser autenticado por el cliente, además de que éste reciba su clave pública. Como se indicó anteriormente dicho certificado suele ser un *X.509*. Otra forma de enviar su clave pública es mediante un mensaje de *Server Key Exchange* (o también si ha enviado su Certificado y este es únicamente para firma y autenticación), siendo claro que para establecer el canal seguro se necesita al menos uno de estos dos mensajes.

- Un último mensaje que puede enviar el servidor en esta fase de negociación es una solicitud de certificado al cliente, quien debe responder con él o en caso de no poseerlo con un mensaje de alerta. Seguidamente se envía un mensaje de *Client Key Exchange* en el cual el cliente envía al servidor la clave maestra la cual es un número aleatorio generado por él y

que servirá como clave del algoritmo simétrico acordado para el intercambio de datos; toda esta información va cifrada mediante la clave pública del cliente.

- Finalmente el servidor envía un mensaje de *Server Hello Done*.
- En el caso de que el cliente haya enviado un certificado con capacidades de firma, enviará adicionalmente un mensaje de *Certificate Verify* firmado digitalmente para que el servidor pueda verificar que la firma es válida.
- El cliente da por concluida la fase mediante un mensaje de *Change Cipher Spec* seguido, inmediatamente, de un mensaje de *Finished*, el cual va cifrado mediante los algoritmos y claves recién negociados.
- El servidor envía su propio mensaje de *Change Cipher Spec* como respuesta y, a continuación, su mensaje de *Finished* cifrado con los parámetros negociados.

En este momento finaliza la fase de *Handshake,* con lo que cliente y servidor pueden intercambiar datos libremente. En la Figura 1.16 se puede ver un esquema de este intercambio de mensajes.

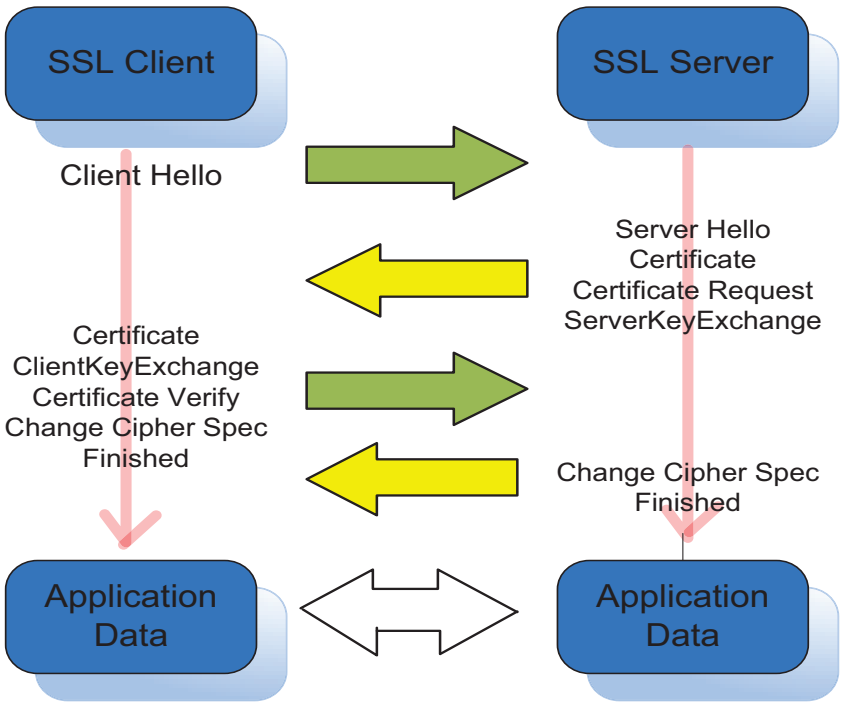

Figura 1.16 Proceso para establecer un canal seguro con equipos desconocidos [PW9]

Antes de su envío, el emisor fragmenta y comprime los mensajes por el protocolo de registro, y al otro extremo de la comunicación son descomprimidos y reconstruidos por el mismo protocolo. El algoritmo de compresión utilizado es negociado en la fase de *Handshake, y* es característico de cada sesión.

El intercambio de mensajes de la fase *Handshake* es mucho más reducido en el caso de que se establezca múltiples conexiones dentro de una misma sesión, o en el caso de reanudar una sesión previamente interrumpida. En la Figura 1.17 se muestran los pasos de esta negociación, los cuales se describen a continuación:

- El cliente usa el identificador de la sesión previamente negociada para enviar un mensaje de *Client Hello*.
- El servidor verifica la validez de dicho identificador es válido, en cuyo caso devuelve un mensaje de *Server Hello* usando el mismo identificador de sesión, luego de lo cual envía al cliente un mensaje de *Change Cipher Spec* y a continuación un mensaje de *Finished* cifrado ya con los parámetros de la sesión reanudada.
- Finalmente, el cliente responde con sus propios mensajes de *Change Cipher Spec* y *Finished* y seguidamente comienzan a intercambiar datos.

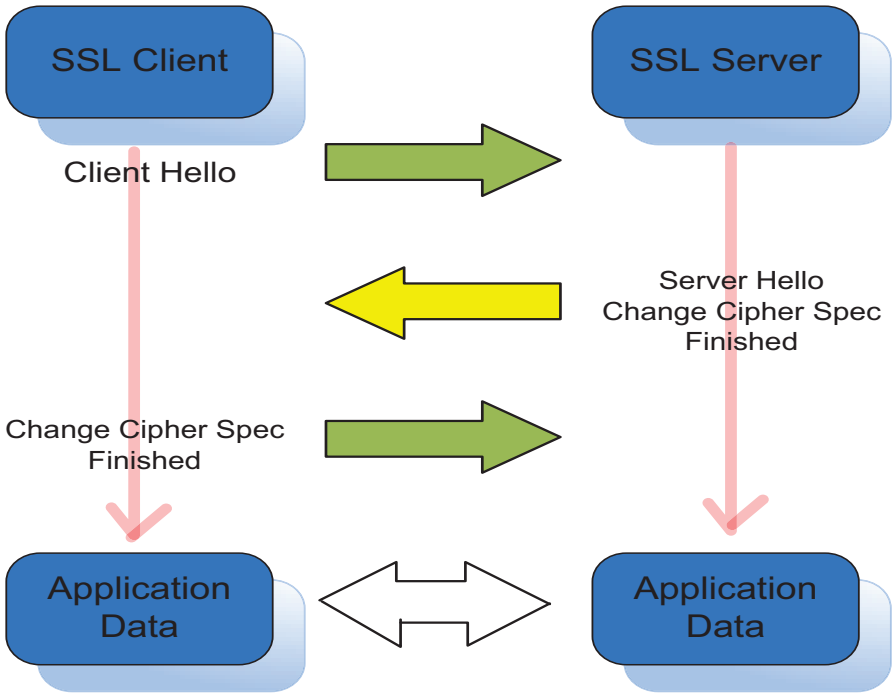

Figura 1.17 Proceso para establecer un canal seguro recuperando una sesión [PW9]

Como se explicó anteriormente, *SSL* es capaz de trabajar de forma transparente con todos los protocolos que trabajan sobre *TCP*., por tal motivo el *IANA* ha asignado un número de puerto por defecto a cada uno de ellos, los cuales se muestran en la Tabla 1.1.

Identificador	Puerto *TCP*	Descripción
https	443	*HTTP* sobre *SSL*
Smtps	465	*SMTP* sobre *SSL*
Nttps	563	*NTTP* sobre *SSL*
Ldaps	646	*LDAP* sobre *SSL*
Telnets	992	*TELNET* sobre *SSL*
Imaps	993	*IMAP* sobre *SSL*
Ircs	994	*IRC* sobre *SSL*
Pop3s	995	*POP3* sobre *SSL*
Ftps-data	989	*FTP*-Datos sobre *SSL*
Ftps-control	990	*FTP*-Control sobre *SSL*

Tabla 1.1 Puertos TCP sobre los que trabaja SSL y TLS [PW9]

1.4.2.2 Protocolo TLS [PW10]

El protocolo *TLS 1.0* (*Transport Layer Security*) se desarrolló en base a *SSL* 3.0, definiéndose en el *RFC 2246*. Posteriormente se lanzó la versión de *TLS 1.1* definida en el TLS 1.1 fue definido en el RFC 4346 en abril del 2006, siendo una actualización de su versión anterior. La última versión 1.2 lanzada fue *TLS 1.2*, y redefinida en el *RFC 6176* de marzo de 2011, aunque son muy pocos los exploradores que actualmente la soportan.

El objetivo de *TLS* es establecer una conexión segura entre un cliente y un servidor, garantizando autenticación y privacidad, el cual es utilizado de manera masiva para brindar seguridad a nivel de la capa de transporte, siendo aprobado su uso para garantizar seguridad en tendencias globales de *e-commerce* por *Visa*, *MasterCard*, *American Express* y muchas de las principales instituciones financieras.

1.4.2.2.1 Características de TLS

En orden de prioridad según el *RFC 4336*, las características de *TLS* son los siguientes:

a) Seguridad criptográfica**:** *TLS* puede ser usado para establecer conexiones seguras entre dos entidades.

b) Interoperabilidad: Brinda la posibilidad de desarrollar aplicaciones usando *TLS* permitiendo intercambiar parámetros criptográficos exitosamente, aun desconociendo el código de otro programador.

c) Extensibilidad: *TLS* permite añadir nuevos métodos de codificación cuando sea necesario, logrando de esta manera ser extensible.

d) Eficiencia relativa: *TLS* trata de reducir el número de conexiones que deben ser establecidas desde cero mediante la utilización de *session caching* ahorrando el uso del CPU. Además, se ha tratado de reducir la actividad de red que se genera debido a su uso.

1.4.2.2.2 Diferencias entre SSL y TLS

Las principales diferencias entre *SSL* 3.0 y *TLS* 1.1 son las siguientes:

- Durante el protocolo de *Handshake* en la versión 3.0 de SSL, si se solicita un certificado al cliente y éste no lo posee se envía un mensaje de alerta advirtiendo de que no lo tiene; mientras que en *TLS* 1.1 de no poseerlo, el cliente no responde al servidor a este requerimiento.
- Cálculo de las claves de sesión. El mecanismo utilizado para construir las claves de sesión es ligeramente diferente en *TLS* 1.1.
- *TLS* 1.1 no soporta el algoritmo de cifrado simétrico *FORTEZZA* debido a que de tipo propietario, mientras que si es soportado por *SSL* 3.0.
- *TLS* utiliza un mecanismo diferente y más seguro en el cálculo del MAC.
- *TLS* 1.1 introduce nuevos códigos de alerta no contemplados por *SSL* 3.0

- *TLS* 1.1 busca frustrar ataques basados en el análisis de la longitud de los mensajes, mediante un nuevo mecanismo en el relleno de los bloques.

Aunque todos los navegadores más importantes soportan TLS 1.0, son muy pocos los que soportan la versión 1.1 y la versión 1.2, siendo estos:

- Internet Explorer: A partir de la versión 8 para Windows 7 y 8 soporta todas las versiones de TLS.
- Google Chrome: A partir de la versión 22 soporta solo TLS 1.1.
- Opera: A partir de la versión 10 soporta todas las versiones de TLS.
- Mobile Safari/UIWebView: Soporta todas las versiones de TLS.

Cabe señalar que tanto Internet Explorer como Opera tienen deshabilitada por defecto la compatibilidad con las versiones 1.1 y 1.2 de TLS.

1.5 BIOMETRÍA [PW11]

La palabra biometría procede de los vocablos griegos *bios* que significa vida y *metron* que significa medida. Se encarga del estudio de métodos automáticos basados en uno o más rasgos conductuales o rasgos físicos intrínsecos para el reconocimiento único de humanos, lo cual es usado en las tecnologías de la información (TI) para la autenticación de personas. En la Figura 1.18 se muestran algunos de los rasgos usados para autenticación digital.

Figura 1.18 Rasgos Biométricos usados para autenticación digital (rasgos faciales, firma, ojo y huella dactilar e impresión vocal) [PW11]

Las características físicas más usadas en biometría son las huellas dactilares, la retina, el iris, los patrones faciales, de venas de la mano o la geometría de la palma de la mano, las cuales son estáticas, ya que son invariables en el tiempo, mientras que entre los ejemplos de características del comportamiento se incluye la firma, el paso y el tecleo las cuales son dinámicas. La voz por otro lado se considera como una mezcla de características físicas y del comportamiento.

1.5.1 HISTORIA

Los primeros conocimientos del uso de biometría datan del siglo XIV en China, donde los comerciantes usaban impresiones de las palmas de las manos y de los pies en papel con tinta, sin embargo en las culturas occidentales recién se empezó a hacer uso de técnicas biométricas a finales del siglo XIX pasando desde medidas exactas de la cabeza y el cuerpo, hasta llegar al reconocimiento por patrones del iris.

En estos últimos años el uso de la biométrica ha evolucionado notablemente empleándose muchos métodos que toman en cuenta varias medidas físicas y de comportamiento. De la misma manera ha habido un amento en aplicaciones de la biometría que ya no se restringen únicamente a identificación, sino que se han desarrollado sistemas de seguridad, y más. Como ya se sabe hasta hace poco las contraseñas y las tarjetas ID han sido utilizadas para guardar cosas personales o bien para controlar el acceso a ciertos lugares, pero estas no son casi nada confiables en comparación a la utilización de la biometría, ya que estos sistemas de seguridad son fácilmente violados con el hecho de divulgar la contraseña o al extraviar la tarjeta.

1.6 HUELLA DACTILAR [PW12]

Una huella dactilar es la impresión visible o moldeada que produce el contacto de las crestas papilares de un dedo de la mano sobre una superficie, y es utilizada como medio de identificación de las personas. En las Figuras 1.19 y 1.20 se muestran las crestas papilares y la impresión dactilar de un dedo.

Figura 1.19 Crestas Papilares [PW12]

Figura 1.20 Impresión Dactilar [PW12]

Juan Vucetich (nacido en Croacia, y nacionalizado argentino) fue el creador del sistema de identificación de las personas a través de las huellas, el cual se desarrolló y patentó en Argentina, donde también se usó por primera vez como un método para esclarecer un crimen. La disciplina científica que estudia las huellas dactilares se llama dactiloscopia.

1.6.1 DIBUJOS PAPILARES

Los dibujos papilares incluyen las crestas papilares las cuales son relieves epidérmicos situados en las palmas de las manos y en las plantas de los pies, y los surcos interpapilares los cuales están determinados por las depresiones que loa relieves o crestas.

1.6.1.1 Propiedades

Se ha demostrado científicamente que los dibujos papilares visibles en la epidermis son perennes, inmutables, diversiformes y originales por las siguientes razones:

Perennes: Desde su formación en el sexto mes de la vida intrauterina, permanecen indefectiblemente invariables en número, situación, forma y dirección hasta el momento que la putrefacción del cadáver destruye la piel.

Inmutables: Las crestas papilares no pueden modificarse fisiológicamente; es decir si hay un traumatismo poco profundo se regeneran, si es profundo la parte afectada resulta invadida por una cicatriz, mas no reaparecen con forma distinta a la que tenían.

Diversiformes: Las huellas de cada dedo son únicas en el mundo, incluso en el caso de gemelos.

Originales: Se puede establecer si fueron plasmadas de manera directa por la persona o si trata de un surco artificial, debido a que todo contacto directo de los surcos papilares naturales produce impresiones originales que tienen características microscópicas identificables del tejido epidérmico.

1.6.2 NORMAS TÉCNICAS

Existen normas técnicas relacionadas con la adquisición, la compresión, el intercambio y la representación de las huellas dactilares.

1.6.2.1 CJIS-RS-0010 Appendix F

Define las características técnicas que deben cumplir los escáneres de captura de huellas dactilares y las impresoras de huellas dactilares para asegurar que las imágenes obtenidas cumplan con criterios de calidad mínimos para ser usadas en procesos forenses manuales o automatizados de verificación o identificación dactilar. Esta norma fue creada en de los Estados Unidos por la FBI, actualmente, esta norma se encuentra en su versión 7, actualizada en 1999.

1.6.2.2 IAFIS-IC-0110

Este estándar también creado por la FBI y define el formato para la compresión de imágenes de huellas dactilares conocido como WSQ, el cual permite alcanzar niveles de compresión típicos de 15:1, manteniendo los detalles relevantes de la huella dactilar como las minucias y poros. Actualmente, esta norma se encuentra en la versión 3.1, actualizada en octubre de 2010.

1.7 LECTORES DE HUELLAS DIGITALES [PW13]

Los Lectores de Huellas Digitales son dispositivos electrónicos que pueden crear mediante procesos matemáticos una matriz con rasgos (crestas y valles) tomados de la yema de un dedo. Esta matriz se almacenará en una base de datos para posteriormente usarla en la verificación del Usuario para comprobar su identidad.

1.7.1 MÉTODOS DE IDENTIFICACIÓN DE HUELLAS DIGITALES

Debido a las características de las huellas digitales es posible la identificación de cada persona por medio de ellas. El uso de huellas digitales como método de identificación ha sido notable en los años, ya que posible denotar las cualidades de estas marcas biométricas con métodos relativamente sencillos. De este modo, y desde hace muchos años, se pudo clasificar a las huellas digitales en lazos, espirales, arcos y compuestas.

Existen varios métodos que se pueden utilizar para identificar los rasgos de las huellas digitales.

1.7.1.1 Sublimación con Yodo

En este proceso se unta con crema de manos en el dedo a tomar la huella, luego se lo presiona con firmeza en papel filtro el cual luego es depositado en un frasco de vidrio que contiene yodo cristalizado, el cual se sublima luego de pocos minutos de cerrar el frasco, con lo cual se obtiene las impresiones dactilares en el papel. En la Figura 1.21 se muestra parte de este proceso.

Figura 1.21 Toma de Huella con el método de sublimación con Yodo [PW13]

1.7.1.2 Carbono Activo

En este proceso se unta crema de manos en el dedo del que se desea obtener la huella, luego se lo presiona en una cartulina clara rociada con una cantidad de carbono activo, finalmente se retira el exceso con un pincel fino y se obtiene una impresión digital como la mostrada en la Figura 1.22.

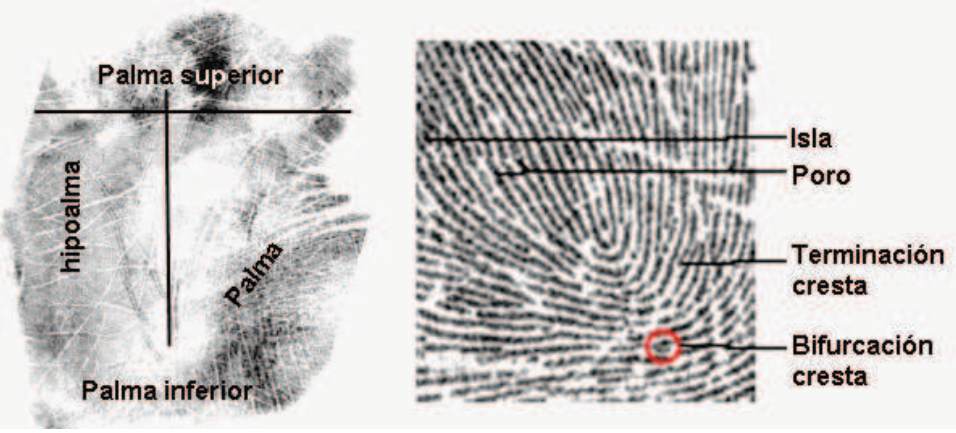

Figura 1.22 Resultado de la Toma de Huella con el método del Carbono Activo [PW13]

1.7.2 SENSORES DE HUELLAS DIGITALES [PW13]

En la actualidad existen varios tipos de sensores para el análisis de huellas digitales como: Ópticos, Capacitivos, Mecánicos, Térmicos, etc. En la Figura 1.23 se muestra un ejemplo de sensor de huellas digitales. En los siguientes puntos se mencionan los más conocidos.

Figura 1.23 Sensor de Huellas Digitales [PW13]

1.7.2.1 Sensores Ópticos

El objetivo de estos tipos de sensores es realizar una captura óptica de la huella digital. Éstos a su vez están clasificados en reflexivos y transmisivos.

- Reflexivos: Éstos funcionan al colocar el dodo sobre una base transparente, que es iluminada por un LED. La imagen de la huella se obtiene debido a que las crestas que están en contacto con el lector absorben la luz, mientras que los valles las reflejan, lo cual crea una

imagen con zonas claras y obscuras que conforman la huella digital. Este tipo de sensores dan problemas con huellas en mal estado, con la humedad de la piel, el polvo, y la suciedad de la platina, por lo que es recomendable la limpieza de la platina y del dedo a tomar la huella.

- Transmisivos: Este tipo de dispositivos emiten una luz que atraviesa completamente al dedo, mientras se toma una imagen detallada de la huella con una cámara, lo que permite obtener una imagen de la superficie y de la parte interna del dedo, es decir una imagen multiespectral; por lo tanto son prácticamente imposibles de engañar, además de no presentar los inconvenientes que se tiene con los lectores reflexivos.

1.7.2.2 Sensores Capacitivos

Este tipo de sensores utilizan dos electrodos, los cuales emiten una corriente al aplicar presión sobre el lector, esta corriente varía al entrar en contacto con las crestas y los valles, permitiéndole tomar imágenes de alta resolución. La respuesta de estos aparatos disminuye con el polvo, la humedad, la grasa, o p or huellas en mal estado, por lo que es recomendada la limpieza periódica del lector.

1.7.3 LECTORES ÓPTICOS DE HUELLAS DIGITALES [PW13]

Cada día se tienen más aplicaciones para lectores de huellas digitales, por ejemplo como control de entrada y salida de trabajadores en industrias, registro de clientes en instituciones bancarias, entre otros. En la Figura 1.24 puede observarse una de estas aplicaciones.

Figura 1.24 Lector Óptico de Huella Digital para control de acceso [PW13]

Este tipo de lectores utilizan sensores ópticos para obtener una imagen de las huellas dactilares, y antes de comparar la muestra obtenida con una almacenada previamente, el software comprueba que se haya obtenido una muestra válida, y hace los ajustes necesarios en caso de requerirse una nueva toma.

Para comparar una huella con otra se ubican los puntos de minucia, los cuales son las zonas donde terminan o se bifurcan las crestas, y se mide las posiciones que tienen, una forma de hacerlo es trazando líneas rectas sobre ellos, con lo cual se obtiene una figura que es única para cada dedo.

1.8 CORPORACIÓN SECUGEN [PW14]

Figura 1.25 Logo SecuGen [PW14]

Es el proveedor líder mundial de tecnología avanzada de reconocimiento de huellas digitales, productos, herramientas y plataformas, que incluyen sensores de huellas digitales basados en SEIR (*Surface Enhanced Irregular Reflection*) con resoluciones de 500 dpi y algoritmos propietarios de extracción y verificación. En la Figura 1.25 se muestra el logo de esta corporación.

La calidad de sus productos es reconocida por su durabilidad, extrema precisión y soporte para una amplia gama de plataformas; además de incluir certificados del FBI y componentes OEMI, kits de desarrollo, software biométrico, entre otros productos, por lo que los clientes de todo el mundo reconocen las ventajas de la implementación de aplicaciones con software y hardware de esta corporación.

1.8.1 ESTÁNDARES BIOMÉTRICOS DE SECUGEN [PW15]

SecuGen está plenamente comprometido con el cumplimiento de las normas de biometría de EE.UU. e internacionales y es compatible con estas especificaciones.

- **FIPS 201: Instituto Nacional de Estándares y Tecnología (NIST).** Verificación de Identidad del Personal (PIV) de los empleados federales y contratistas: Se utiliza para mejorar la identificación y autenticación de los empleados federales y contratistas, para acceder de forma segura a las instalaciones federales y otros donde existe la posibilidad de ataques terroristas. La documentación la puede encontrar en el siguiente enlace:

  http://csrc.nist.gov/publications/fips/fips201-1/FIPS-201-1-chng1.pdf

- **SP 800-76: Instituto Nacional de Estándares y Tecnología (NIST).** Especificaciones de biométricos para la Verificación de Identidad: Describe las especificaciones de técnicas de adquisición y formato para las credenciales biométricas del sistema PIV, cuyo principal objetivo es la interoperabilidad universal de gran rendimiento. La documentación se encuentra en el siguiente enlace:

  http://csrc.nist.gov/publications/nistpubs/800-76-1/SP800-76-1_012407.pdf

- **ANSI INCITS 378-2004: Instituto Nacional de Estándares Americano y el Comité Internacional de Estándares de Tecnologías de la Información.** Formato de las minucias de dedo para el intercambio de datos: Esta norma utiliza el concepto fundamental de minucias, especificando un formato genérico de concepto y datos para la representación de hullas dactilares, sin indicar requerimientos o características de aplicaciones específicas.

  El *NIST* estableció las Pruebas de Interoperabilidad para Intercambio de Minucias (*MINEX* Minutiae Interoperability Exchange), la cual es una evaluación continua de esta norma con el objetivo de efectuar mediciones de rendimiento e interoperabilidad, así como el cumplimiento de la misma. La documentación se la puede encontrar en el siguiente enlace:

  http://fingerprint.nist.gov/minex/index.html

- **ISO / IEC 19794-2:2005: La Organización Internacional de Normalización.** Formatos Biométricos de intercambio de datos - Parte 2:

Datos de las minucias de la Huella digital. Utiliza el concepto de las minucias para la representación de impresiones dactilares, la descripción de la determinación de las minucias y el formato de los datos. Por ser genérico puede ser utilizado en un amplio rango de aplicaciones.

- ***BioAPI* (*ANSI-INCITS 358-2002*): El Consorcio *BioAPI*, Instituto Nacional Americano de Estándares, y Comité Internacional para los estándares de las tecnologías de la Información.** La Implementación de este estándar permitirá el desarrollo rápido de aplicaciones biométricas, mayor aplicación de múltiples alternativas biométricas (huellas digitales, voz, cara, iris, etc), entre otras.

  Además proporcionará beneficios de negocios al proporcionar interfaces sencillas, un estándar de acceso modular para funciones, algoritmos, y dispositivos biométricos, manejo seguro y robusto de datos biométricos, y soporte para identificación biométrica en entornos informáticos distribuidos.

- **ISO / IEC 19794-4:2005: La Organización Internacional de Normalización.** Formatos biométricos de intercambio de datos - Parte 4: Datos de imagen Dactilar ISO / IEC 19794-4:2005. Especifica un formato de intercambio de registro de datos, para almacenamiento, registro, y transmisión de la información de uno o más dedos o áreas de la palma, con una estructura de datos ISO / IEC 19785-1 CBEFF. Esto puede ser utilizado para el intercambio y la comparación de datos. Esta información está destinada para el intercambio entre las organizaciones que se basan en dispositivos y sistemas automatizados, para la identificación o verificación basada en la información de las áreas de imagen dactilar. La información recopilada se puede grabar en medios legibles por máquina, o puede ser transmitida por las instalaciones de comunicación de datos.

- **CBEFF: Instituto Nacional de Estándares y Tecnología (NIST).** Este estándar facilita el intercambio de datos biométricos entre diferentes componentes de un sistema o entre sistemas, promoviendo la interoperabilidad, proporcionando compatibilidad con futuras mejoras tecnológicas.

1.8.2 SECUGEN HAMSTER PLUS [PW16]

Este lector de huellas digitales es la versión mejorada de la línea de productos de *SecuGen*, el cual cuenta con Auto-Encendido y Captura Rápida, con diseño cómodo y ergonómico. Cuenta además con el sensor óptico más avanzado y robusto de la industria que utiliza la tecnología biométrica patentada de huellas digitales *SEIR*.

Auto-Encendido es una tecnología de detección que comprueba la presencia de un dedo, que permite escanear el dedo en el momento que éste entra en contacto con el sensor.

La Captura Rápida garantiza la calidad de escaneo de huellas dactilares en condiciones difíciles, aún en dedos con piel seca, húmedo, con cicatrices, envejecida, o en condiciones ambientales brillantes, gracias al ajuste automático de brillo del sensor.

Figura 1.26 Lector de Huellas Hamster Plus [PW16]

Este dispositivo puede ser utilizado en funciones de autenticación, identificación y verificación de huellas digitales. La Figura 1.26 muestra uno de estos dispositivos.

1.8.2.1 Características

Las características que posee este dispositivo son las siguientes:

- Alto rendimiento, libre de mantenimiento del sensor óptico de huellas dactilares.

- Sensor resistente a los arañazos, golpes, vibraciones y choques electrostáticos.
- Auto-Encendido (Detección automática de colocación de los dedos).
- Captura Rápida (Ajuste automático del brillo de huellas dactilares).
- Conexión USB.
- Base extraíble.
- Compacto, ligero y portátil.
- Guía Dactilar Integrada.
- Fácil acceso para cualquier dedo.

Al utilizarse con el software *SecuGen* se suman las siguientes características:

- Verificación rápida y precisa.
- Imagen removible de impresión latente (sin copias temporales).
- Cifrado de plantillas de huellas dactilares (no se puede utilizar para reconstruir imágenes de huellas dactilares).
- Reconocimiento del Dispositivo de huellas dactilares (a través del número de serie programable).
- Conexión Multi-Dispositivo (mediante el número de serie programable).

1.8.2.2 Ventajas del uso de periféricos SecuGen

Los Periféricos *SecuGen* son fáciles de usar y se puede instalar rápidamente en cualquier PC con sistema operativo Windows, además de estar respaldado por la mejor garantía de la industria, lo que asegura un rendimiento constante en el desarrollo de aplicaciones de software para entornos de escritorio, de red, empresarial e Internet. Al ser utilizados para autenticación biométrica como parte de un programa general de seguridad pueden ayudar a reducir las molestias de la sobrecarga de contraseña, disminuyen los riesgos de violaciones a la seguridad,

mejoran la rendición de cuentas y añaden verificaciones no rechazadas, a la vez que son más cómodos e intuitivos para casi cualquier usuario.

Estas son algunas de las aplicaciones de seguridad en las que suelen utilizarse:

- Seguridad en Ordenador personal / estación de trabajo.
- Seguridad en Red / Empresa.
- La seguridad de contenidos en Internet.
- *E-commerce.*
- Transacciones B2B.
- Transacciones electrónicas.
- Banca y sistemas financieros.
- Sistemas de información médica.
- Cualquier aplicación basada en contraseñas.

1.8.2.3 Especificaciones

Nombre (Modelo)	Hamster Plus (HSDU03P)
Sensor Óptico de Huella Dactilar	SDU03P
Resolución de Imagen	500 DPI
Tamaño de Imagen	260 x 300 pixels
Tamaño Recuadro	16.1 mm x 18.2 mm
Área del Sensor	13.2 mm x 15.2 mm
Velocidad de captura de Imagen	0.2 a 0.5 sec con Captura Rápida
Temperatura de Operación	-20° a +65°C.
Humedad de Operación	90% RH o menor, no condensada.

Dimensiones y Peso	27 x 40 x 73 mm, 100 g (sin Soporte).
Voltaje Administrado / Corriente Max.	5 V DC / 140 mA.
Falsa Aceptación y falso rechazo	En el siguiente link:

(http://www.secugen.com/support/faqs.htm#10)

Interfaz	USB 1.1 Velocidad-Total, USB 2.0 Alta Velocidad.
Garantía	Un año.
Estándares Soportados	ISO 19794, INCITS 378, BioAPI.
Certificaciones	FCC, CE, RoHS.
Sistemas Operativos Soportados	Windows 7 / Vista / XP / 2000 / 9x
	Windows Server 2008 R2, 2003
	Windows CE, Java, Linux, SUN Solaris10
SDKs Soportados	FDx SDK Pro for Windows
	SecuBSP SDK Pro for Windows
	SecuGen BioAPI BSP Pro for Windows
	SecuSearch SDK Pro for Windows
	FDx SDK for Java
	FDx SDK for Windows CE
	SDU03 SDK for Linux / SPARC / X86 Solaris 10
	SecuGen ISO Image SDK – Windows

CAPÍTULO II

2. DISEÑO DEL SOFTWARE

2.1 ANÁLISIS DE REQUERIMIENTOS

El número de servicios que se ofrece en un campus universitario son muy variados, y al tener un número de usuarios elevado es primordial aumentar la eficiencia en la atención de éstos.

Tomando en cuenta como principales servicios ofrecidos, los problemas a solucionar serían los siguientes:

- Tardanza al momento de realizar cobro de consumos en copiadoras, bares, o multas de bibliotecas
- Búsqueda manual de la existencia de un libro requerido en cada biblioteca.
- Dificultad en la organización de los turnos para asistencia en cada consultorio médico.
- Falta de documentación digitalizada para cada paciente.
- Por otra parte el aumento en la delincuencia haría preciso crear medios por los cuales el estudiante pueda sustentar sus gastos dentro del campus sin la necesidad de contar con dinero en efectivo.

Tomando en cuenta lo anterior, se pensó en la creación de un sistema informático que pueda dar solución a los problemas señalados con ayuda de los nuevos adelantos tecnológicos. Se establecieron datos estimados de los requerimientos de usuarios, clientes web, etc., teniendo en cuenta las experiencias vividas como estudiantes de una institución universitaria.

2.2 ESPECIFICACIONES DE LOS REQUISITOS DEL SOFTWARE (ERS)

2.2.1 INTRODUCCIÓN

El ERS una herramienta que ayuda a describir las características que debe poseer una aplicación de software para alcanzar sus objetivos. El IEEE establece formatos que ayudan a este propósito siendo la versión más reciente el estándar ISO/IEC/IEEE 29148 publicado el 01/12/2011; sin embargo para el presente trabajo se seguirá el esquema dado por el estándar IEEE 830-1998 "*Recommended Practice for Software Requirements Specifications*" (Prácticas Recomendadas para Especificaciones de Requerimientos de Software), el cual era el formato vigente al momento de iniciación de este proyecto de titulación. Para mayor información sobre este estándar ver el Anexo A.

2.2.1.1 Propósito

El objetivo de esta especificación es definir de manera clara y precisa las funcionalidades y restricciones que tendrá el sistema a ser utilizado en un campus universitario para la optimización de los servicios que se brinden dentro del mismo. Está dirigida al equipo de desarrollo de software y a las personas que harán uso del sistema terminado.

2.2.1.2 Alcance

El sistema creado se llamará "*Multiservicios Estudiantiles*", el cual está diseñado como una aplicación web, el cual servirá para optimizar el tiempo de atención en los diferentes servicios dentro de un campus universitario.

El sistema se desarrollará como una aplicación web en una arquitectura de tres capas, donde cada una se encontrará albergada en un servidor diferente, por lo cual deberá cumplir con las siguientes características:

- Acceso a módulos mediante perfiles asociados al tipo de usuario.

- Gestionar la información de usuarios. Crear y modificar registros de usuarios.
- Autenticar a los estudiantes mediante sus huellas digitales al utilizar los diferentes servicios y pagar sus consumos en bares estudiantiles y copiadoras y el pago de multas en bibliotecas.
- Registrar el ingreso de saldo por parte de los estudiantes y retiro de dinero por parte de los administradores en sus respectivas cuentas.
- Gestionar transacciones. Registrar los movimientos en las cuentas de los usuarios y mostrar informes a cada usuario, así como a la entidad financiera.
- Gestionar el catálogo de libros. Permitir a los estudiantes consultar la existencia de un libro en las bibliotecas por título o autor; conceder préstamos de libros por parte de los funcionarios de las bibliotecas.
- Gestionar las citas médicas. Permitir a los estudiantes crear o eliminar citas médicas en los diferentes consultorios, y comprobar la lista de citas a los doctores en su respectiva especialidad; permitir a los funcionarios Doctor cancelar.
- Publicar notificaciones a los estudiantes.

Con la implementación del sistema los estudiantes se beneficiarán en los siguientes aspectos:

- No necesitarán contar con dinero en efectivo para realizar sus consumos o pago de multas dentro del campus universitario, por lo que ahorrarán tiempo al no necesitar contar el dinero ni esperar por el cambio y tendrán la confianza de que nadie más pueda utilizar su dinero gracias al uso del lector de huellas digitales.
- Tendrán una base de datos centralizada para la consulta del catálogo de libros web, por lo que ya no será necesario visitar cada biblioteca por separado.

- Podrán planificar las citas en los consultorios médicos vía web, por lo que tendrán la garantía de ser atendidos en el momento proyectado.

2.2.1.3 Definiciones, siglas y abreviaciones

- Cuenta: registro asignado a un estudiante identificado con un número único en el que se refleja las transacciones monetarias realizadas en el campus.
- Debitar: restar una cantidad de dinero del último saldo una cuenta.
- *DPI*: *Dots Per Inch* (Puntos por pulgada) [PW17].
- *Framework:* es una plataforma de software reusable la cual incluye herramientas de apoyo y motores de ejecución para facilitar el desarrollo de un proyecto o solución [PW18].
- *HTTPS*: *Hyper Text Transfer Protocol Secure* (Protocolo seguro de transferencia de hipertexto) [PW19].
- *INCITS: InterNational Committee for Information Technology Standards* (Comité internacional de estándares de tecnología de información) [PW20].
- *ISO: International Organization for Standardization* (Organización internacional de estandarización) [PW21].
- *IPsec: Internet Protocol security* [PW22].
- Lector de huellas*:* instrumento utilizado para el escaneo y registro de huellas digitales.
- Monedero electrónico: es una tarjeta con un chip electrónico que permite almacenar una cantidad variable de dinero, en general no muy grande para aquellas transacciones de bajo monto y alto volumen que requieren gran velocidad y seguridad, haciendo el proceso de pago rápido y sencillo [PW23].
- Pixel: es la menor unidad homogénea en color que forma parte de una imagen digital, ya sea esta una fotografía, un fotograma de vídeo o un gráfico [PW24].

- Platina: Superficie de vidrio del lector donde se ubican los dedos para el escaneo de huella [PW25].
- *SDK*: *Software Development Kit* (Kit de desarrollo de software); conjunto de herramientas que permite crear aplicaciones para un sistema concreto [PW26].
- Sistema de pago electrónico: es un sistema de pago que facilita la aceptación de pagos electrónicos, realizan la transferencia del dinero entre compradores y vendedores en una acción de compra-venta electrónica a través de una entidad financiera autorizada por ambos [PW27].
- SO: Sistema operativo.
- Transacción: movimiento de capital realizado por una persona de su cuenta, sea esta de depósito o débito.
- USB: *Universal Serial Bus* (Bus Universal en Serie) [PW28].

2.2.1.4 Apreciación global

Adicionalmente, el resto del ERS está organizado en dos partes siguiendo la siguiente estructura:

- La parte dos contiene una descripción global del sistema, es decir describe los factores y requisitos generales siendo estos la perspectiva, funciones, características, restricciones, atenciones y dependencias.
- La parte tres contiene los requisitos específicos del software organizada de acuerdo a la clase de usuario.

2.2.2 DESCRIPCIÓN GLOBAL

2.2.2.1 Perspectiva del producto

“Multiservicios Estudiantiles” ayuda a los funcionarios de los diferentes servicios de un campus a agilitar la atención, permitiendo que se identifique a los

estudiantes con su huella digital y hacer uso de su dinero mediante un sistema de pago automatizado similar a un monedero electrónico.

Este sistema es independiente de otros sistemas ya existentes en el campus, por lo cual no es necesario establecer interconexión con otros. Tendrá una interfaz web amigable con el usuario y de fácil acceso, lo que permitirá la agilización de la atención en los servicios del campus.

2.2.2.1.1 Interfaces del sistema

El sistema contará con tres tipos de interfaces correspondiente a las tres capas en las que se programará mostradas en la Figura 2.1.

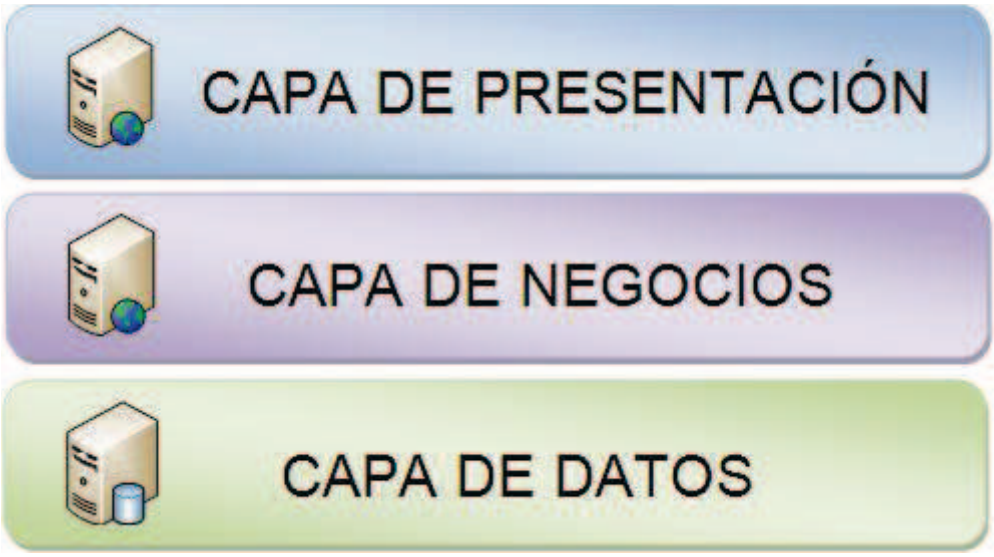

Figura 2.1 Esquema de capas del sistema

- Capa de Presentación: Interfaz web que permitirá la comunicación del usuario final con el sistema.
- Capa de Negocios: Un servicio web que implemente las reglas del negocio.
- Capa de Datos: Un servicio web para la comunicación entre la aplicación y la base de datos.

El sistema al ser independiente de otros no tiene interfaces para acoplarse a otros, sin embargo la base de datos será adaptable a otra que hubiere en la institución.

2.2.2.1.2 Interfaces con el usuario

El usuario ingresará a la aplicación mediante una página web de inicio de sesión, la que dará acceso a una página de bienvenida. Ésta y las siguientes páginas contendrán una barra de menú con las opciones de cada usuario dependiendo del perfil con el que haya accedido, y lo llevarán a las siguientes páginas que contienen la información seleccionada, brindando al usuario un ambiente intuitivo y amigable. La imagen 2.2 muestra un esquema general de las interfaces disponibles en el sistema.

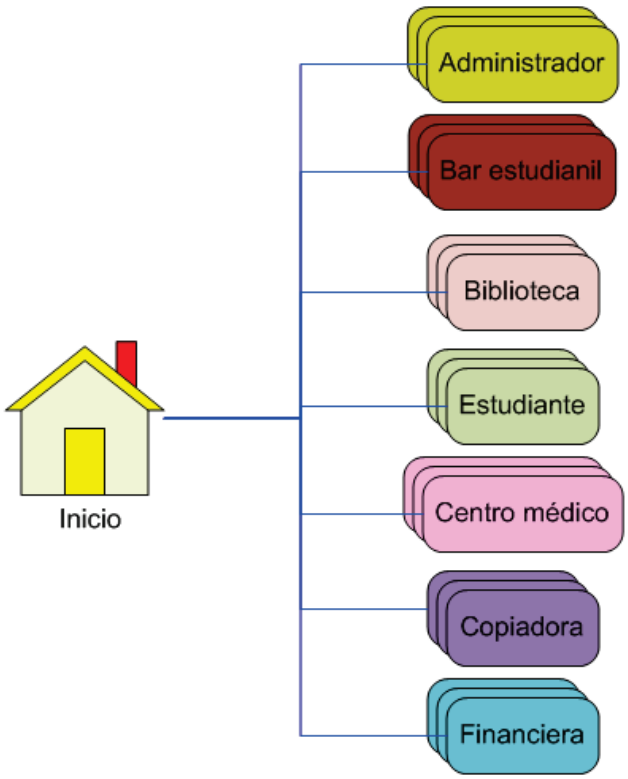

Figura 2.2 Esquema de interfaces con el usuario

2.2.2.1.3 Interfaces con el hardware

El sistema contará con las interfaces mostradas en la Figura 2.3 y listadas a continuación:

- Entrada: Ratón, teclado, lector de huellas digitales *"SecuGen Hamster Plus"*
- Salida: Monitor.

Figura 2.3 Interfaces con el hardware

El lector de huellas digitales será utilizado por todos los usuarios, excepto por los estudiantes y los funcionarios financieros.

2.2.2.1.4 Interfaces con el software

Para la realización del sistema será necesario lo siguiente:

- *Framework* de soporte para la aplicación y comunicación con la base de datos.
    - Nombre: *Microsoft .NET Framework*.
    - Versión: 4.0 *Service Pack 1.*
    - Fuente: *Microsoft Corporation*.
- *Software* para la comunicación entre el lector de huellas y el sistema.
    - Nombre: *SecuBSP SDK Pro*.
    - Versión: 1.4.
    - Fuente: *SecuGen Biometric Solutions*.

La Figura 2.4 muestra un esquema de la comunicación realizada con dichas interfaces.

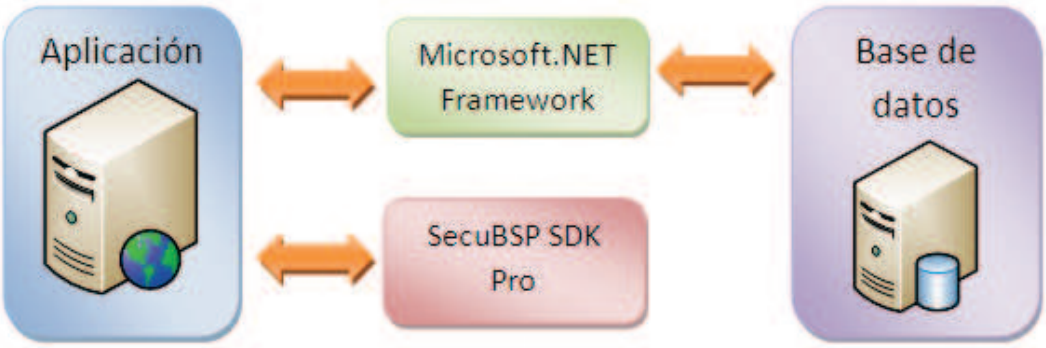

Figura 2.4 Interfaces con el software

2.2.2.1.5 Interfaces de comunicaciones

El esquema de la Figura 2.5 muestra los dos tipos de protocolos usados para la comunicación:

- Comunicación entre cliente y servidor: *HTTPS*.
- Comunicación entre servidores: *IPSec*.

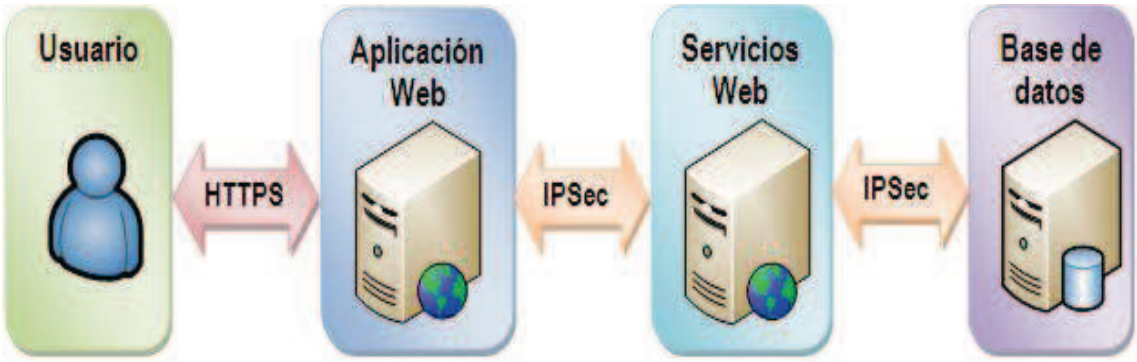

Figura 2.5 Interfaces de comunicaciones

2.2.2.1.6 Requisitos de adaptación del sitio

- Todos los equipos, tanto servidores como clientes deberán tener acceso a Internet, o en su defecto a la intranet del campus.
- Los equipos servidores necesitarán tener instalado *Microsoft Framework* 4.0.
- Será necesario que las máquinas cliente que utilicen lector de huellas instalen los controladores del mismo.

2.2.2.2 Funciones del producto

El sistema tendrán diferentes funciones dependiendo del tipo de usuario que lo use de esta manera se tendrán las mostradas en las Tablas 2.1 y 2.2:

Tipo de Usuario	Funciones
Administrador	• Autenticarse en el sistema. • Crear o modificar usuarios. • Habilitar las cuentas bloqueadas. • Acceder a sus datos personales. • Cambiar su contraseña.
Estudiante	• Autenticarse en el sistema. • Acceder a sus datos personales. • Cambiar su contraseña. • Acceder a información del catálogo de las bibliotecas. • Planificar citas médicas. • Acceder al historial de transacciones de su cuenta.
Biblioteca	• Autenticarse en el sistema. • Acceder a sus datos personales. • Cambiar su contraseña. • Gestionar el catálogo de libros de su biblioteca. • Gestionar los préstamos de su biblioteca. • Realizar cobro de multas pendientes. • Acceder al historial de transacciones de su cuenta. • Publicar notificaciones a estudiantes
Bar estudiantil	• Autenticarse en el sistema. • Acceder a sus datos personales. • Cambiar su contraseña. • Realizar el cobro de consumos a los estudiantes. • Acceder al historial de transacciones de su cuenta. • Publicar notificaciones a estudiantes.
Centro médico	• Autenticarse en el sistema. • Acceder a sus datos personales. • Cambiar su contraseña. • Verificar lista de pacientes. • Identificar a los pacientes. • Administrar las historias clínicas de los pacientes. • Publicar notificaciones a estudiantes.
Copiadora	• Autenticarse en el sistema. • Acceder a sus datos personales. • Cambiar su contraseña. • Realizar el cobro de consumos a los estudiantes. • Acceder al historial de transacciones de su cuenta. • Publicar notificaciones a estudiantes.

Tabla 2.1 Funciones de los usuarios. Parte I

Tipo de Usuario	Funciones
Financiera	• Autenticarse en el sistema. • Acceder a sus datos personales. • Cambiar su contraseña. • Verificar el historial de transacciones de las cuentas de los usuarios. • Administrar los depósitos y retiros de dinero de las cuentas de los usuarios. • Publicar notificaciones a estudiantes.

Tabla 2.2 Funciones de los usuarios. Parte II

2.2.2.3 Características del usuario

Los usuarios de "*Multiservicios Estudiantiles*", son en su mayoría estudiantes universitarios, los cuales tienen gran experiencia en navegación web, sin embargo los usuarios de tipo copiadora y bar estudiantil podrían tener un nivel de educación secundaria, con lo cual es necesario que tengan al menos un mes de experiencia en el uso de navegadores web.

Todos los usuarios deberán tener una capacitación para el correcto uso del lector de huellas. No es necesario el conocimiento de del uso de ningún tipo de lenguaje de programación o de bases de datos.

2.2.2.4 Restricciones

El sistema "*Multiservicios Estudiantiles*" tendrá las siguientes limitaciones:

- No contempla interfaces para comunicarse con otros sistemas ya existentes.
- La base de datos será compatible con otra preexistente si está desarrollada en el mismo motor de base de datos.
- La lectura de huellas digitales solamente funcionará con lectores de huellas "*SecuGen Hamster Plus*".
- Se apegará a las licencias del software correspondientes al *SDK* de los lectores de huellas digitales.
- La información enviada entre cliente y servidor será utilizado el protocolo *https*, mientras que para la información entre servidores se usará *IPsec*.

- Las interfaces web estarán disponibles únicamente en español.
- No creará un log para auditoría de software.
- El prototipo tendrá una cola máxima de 5000 peticiones.
- Los equipos clientes que necesiten de un lector de huellas deberán tener un puerto USB 1.0 de alta velocidad o superior para uso exclusivo del dispositivo.
- La aplicación estará optimizada para Internet Explorer 7.0 o superior; otros navegadores web tendrán funciones limitadas.

2.2.2.5 Atenciones y dependencias

Los factores que podrían afectar los requerimientos del sistema son los siguientes:

- Los servidores deberán tener un SO *Windows Server 2008* o superior.
- La velocidad de la conexión entre el cliente y el servidor limitará el tiempo de respuesta de la aplicación.
- Cambio en la marca o modelo de lector de huellas.

2.2.2.6 Mejoras al Proyecto

En futuras versiones, el sistema podría estar en capacidad de lo siguiente:

- Imprimir comprobantes tanto de depósito a los estudiantes y de retiro a los prestadores de servicios del campus por parte de la institución financiera del campus.
- Comunicarse directamente con las cuentas bancarias de la universidad y las de los prestadores de servicios, con lo que se habilitaría las transacciones interbancarias.
- Incrementar los parámetros de búsqueda de libros en el catálogo de biblioteca.

2.2.3 REQUISITOS ESPECÍFICOS

2.2.3.1 Requisitos de la interfaz externa

2.2.3.1.1 Interfaz del usuario

El usuario accederá al sistema mediante una pantalla de inicio, en la cual se tendrá en la parte superior el nombre del mismo y debajo de éste imágenes alusivas a los servicios brindados; en la parte inferior central estarán disponibles dos campos en los que el usuario pondrá su nombre de usuario y contraseña, los cuales al inicio serán por defecto su número de cédula.

Posteriormente se mostrará una pantalla de bienvenida en la que se verifica su rol y da acceso a las diferentes opciones dependiendo del tipo de usuario en una barra de menús desplegables que estará presente en todas las páginas del sistema y se encontrará situada bajo el logotipo de la aplicación. La Figura 2.6 muestra un esquema de las interfaces a la que se tendrá acceso de acuerdo al tipo de usuario. Todos los usuarios tendrán acceso a la opción datos personales, la cual mostrará su información personal y permitirá el cambio de su contraseña; y los funcionarios excepto Administrador, podrán publicar mensajes. A parte de esto, cada usuario tendrá funciones diferentes teniendo de esta manera:

Administrador:

- Ingreso usuario: permite ingresar los datos de un nuevo usuario, así como su rol, y la huella digital de los estudiantes.
- Modificar usuario: consulta un usuario existente en el sistema y modificar sus atributos, su rol y habilitar la cuenta en caso de estar bloqueada.

Biblioteca:

- Catálogo de libros: consulta la existencia de un libro por autor o título, y los préstamos existentes.
- Préstamo libros: inscribe el préstamo de un determinado libro a un estudiante identificado mediante su huella digital.
- Devolución libro: registra la devolución de un libro prestado a un estudiante.

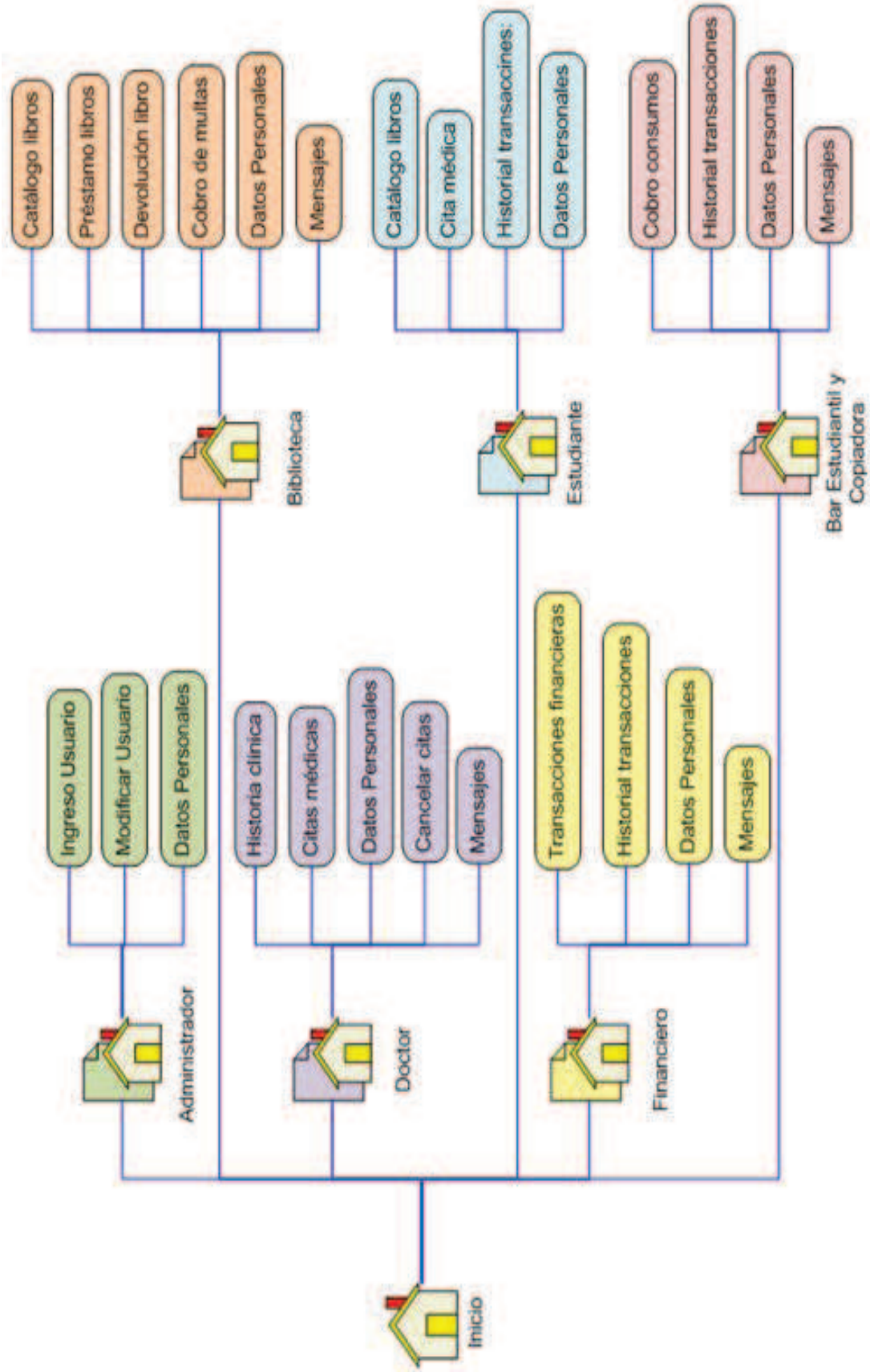

Figura 2.6 Esquema de interfaz de usuario.

- Cobro de multas: consiente el pago de multas en los préstamos atrasados de un estudiante mediante debito de su cuenta.

Bar estudiantil y copiadora:

- Cobro consumos: enlista los productos adquiridos por el estudiante, calcula el valor total, verifica el saldo y realiza el cobro mediante la huella digital.
- Historial transacciones: muestra las diez últimas transacciones realizadas, y el monto; además permite consultar las transacciones dentro de un rango de fechas.

Doctor:

- Historia clínica: permite registrar los datos médicos básicos de un estudiante como tipo de sangre, así como el diagnóstico y prescripción de cada cita.
- Citas médicas: despliega las citas planificadas por los estudiantes para la fecha actual.
- Cancelar citas: permite cancelar citas reservadas por un estudiante.

Financiero:

- Transacciones financieras: asienta el ingreso de dinero a la cuenta de un estudiante y el retiro de las cuentas de funcionarios.
- Historial transacciones: muestra las diez últimas transacciones realizadas por un usuario identificado por su número de cédula, y el monto; además permite consultar las transacciones dentro de un rango de fechas.

Estudiante:

- Catálogo libros: consulta la existencia de un libro por autor o título, y los préstamos existentes.
- Historial transacciones: muestra las diez últimas transacciones realizadas, y el monto; además permite consultar las transacciones dentro de un rango de fechas
- Cita médica: planifica citas en los diferentes consultorios médicos.

2.2.3.1.2 Interfaz con el hardware

El sistema obtendrá la mayoría de información de entrada por parte del usuario mediante el mouse y el teclado, pudiendo ser éstos de cualquier marca o tipo.

Las especificaciones generales de los lectores de huellas digitales se muestran en la Tabla 2.3.

Modelo	Hamster Plus (HSDU03P™)
Sensor óptico de huella digital	SDU03P™
Resolución de imagen	500 DPI
Tamaño de la imagen	260 x 300 pixeles
Tamaño de la platina	16.1 mm x 18.2 mm
Área de detección	13.2 mm x 15.2 mm
Velocidad de captura de imagen	0.2 a 0.5 s. con Smart Capture™
Interfaz	USB 1.1 Full-Speed, USB 2.0 Hi-Speed
Estándares soportados	ISO 19794, INCITS 378, BioAPI
SO soportados	Windows 7 / Vista / XP / 2000 / 9x Windows Server 2008 R2, 2003 Windows CE, Java, Linux, SUN Solaris10

Tabla 2.3 Especificaciones generales lector de huellas [PW16]

2.2.3.2 Requisitos funcionales [L5]

Debido a que el sistema tiene diferente comportamiento dependiendo del tipo de usuario con el que se ingrese, se ha visto conveniente el uso de diagramas de casos de uso para especificar las funciones.

Un caso de uso especifica una secuencia de acciones incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto; proporcionando un medio intuitivo y sistemático para capturar requisitos funcionales con un énfasis especial en el valor añadido para cada usuario individual o para cada sistema externo. La mayoría de los sistemas tienen muchos tipos de usuarios los cuales se representan mediante actores. Todos los actores y casos de uso del sistema forman un modelo de casos de uso.

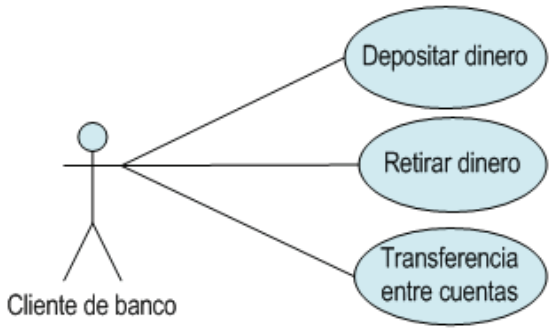

Figura 2.7 Ejemplo de diagrama de casos de uso

Un diagrama de casos de uso es un diagrama UML que muestra las asociaciones entre los actores y los casos de uso que interactúan, y su ventaja principal es la facilidad para interpretarlos. En la Figura 2.7 se muestra un ejemplo de estos diagramas.

2.2.3.2.1 Identificación de actores

Se tiene un gran número de tipo de usuarios que van a interactuar con el sistema y se los lista en la Tabla 2.4.

Actor	Descripción
Usuario	Persona que accede al sistema mediante un explorador de internet.
Administrador	Usuario autenticado como Administrador.
Biblioteca	Usuario autenticado como funcionario de una biblioteca.
Bar Estudiantil	Usuario autenticado como funcionario del Bar Estudiantil.
Copiadora	Usuario autenticado como funcionario de una copiadora.
Centro Médico	Usuario autenticado como funcionario de un centro médico.
Estudiante	Usuario autenticado como estudiante.
Financiera	Usuario autenticado como funcionario de la entidad financiera.
Base de Datos	Sistema gestor de base de datos.

Tabla 2.4 Actores de los casos de usos

2.2.3.2.2 Identificación de los casos de uso

En la Figura 2.8 se identifica en manera general los casos de uso del sistema y sus agrupaciones (paquetes). Su comportamiento se detallará en las Tablas 2.5 a 2.21, y se mostrará los diagramas de cada paquete desde la Figura 2.9 a la 2.18, indicándose su respectivo nombre.

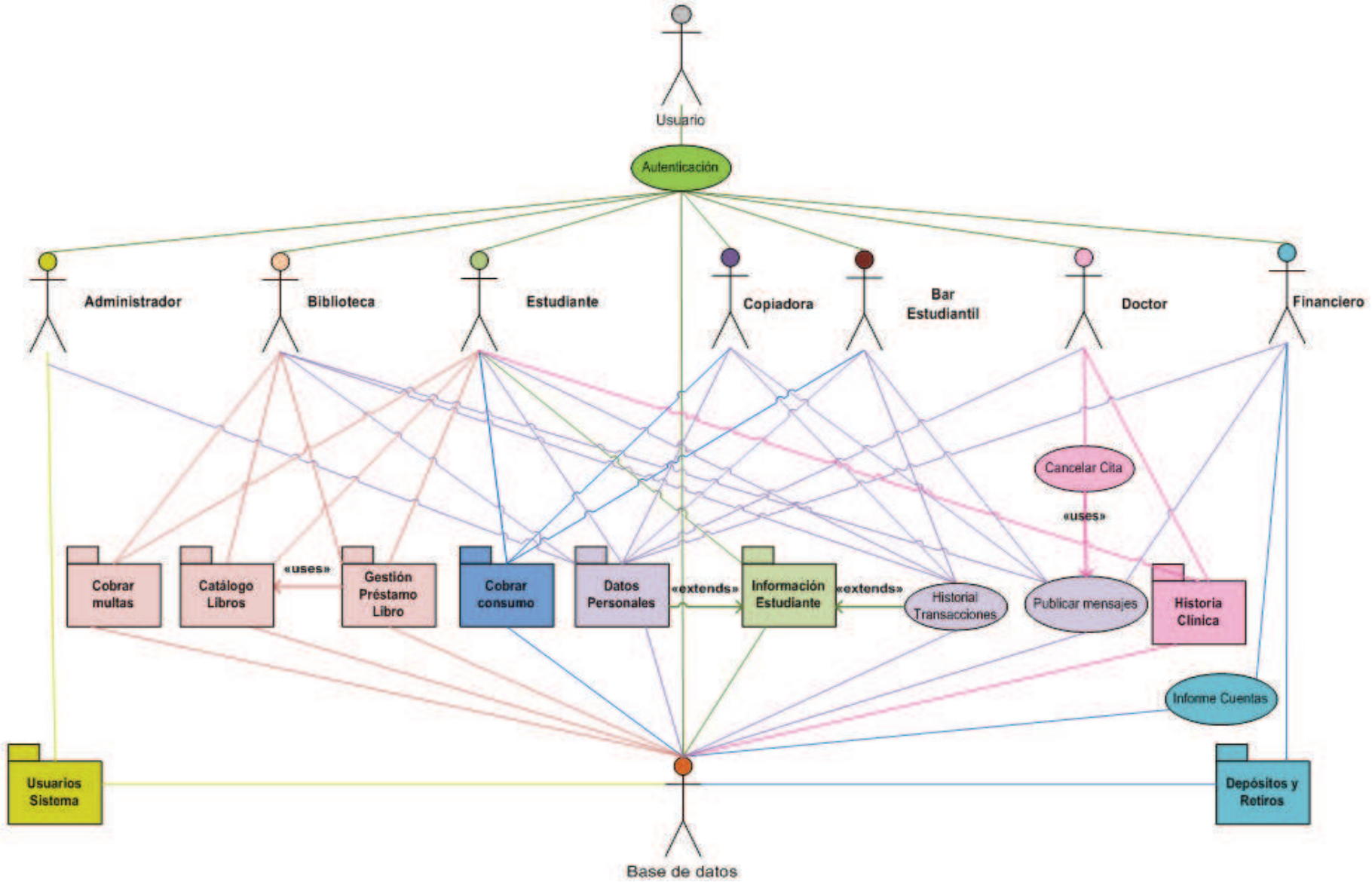

Figura 2.8 Diagrama de casos de uso global

Nombre	Autenticación
Descripción:	Permitir el ingreso al sistema a un usuario.
Actores:	Usuario, Base de datos.
Precondición:	Haber ingresado a la página web del sistema.
Flujo normal:	1. El usuario ingresa su nombre de usuario y su contraseña. 2. El sistema verifica los datos ingresados con los de la base de datos e identifica si la cuenta está habilitada y el tipo de usuario. 3. El sistema permite el ingreso al sistema y le muestra las opciones de acuerdo al tipo de usuario.
Flujo alternativo:	1. En 2 el nombre de usuario o la contraseña no son correctos. 2. Se ingresa 3 veces erradamente el nombre de usuario o la contraseña y se deshabilita la cuenta. 3. En 2 la cuenta está deshabilitada.
Pos condición:	El usuario es autenticado como estudiante, administrador, doctor, funcionario de bar estudiantil, biblioteca, financiero o copiadora.

Tabla 2.5 Descripción caso de uso Autenticación

Nombre	Historial Transacciones
Descripción:	Obtener la fecha y hora, descripción, monto y el saldo de las transacciones realizadas durante un intervalo de tiempo.
Actores:	Bar Estudiantil, Biblioteca, Copiadora, Estudiante.
Precondición:	Haber sido autenticado como funcionario de bar estudiantil, biblioteca, copiadora o estudiante.
Flujo normal:	1. El sistema muestra las últimas diez transacciones realizadas. 2. El usuario ingresa las fechas del intervalo de tiempo que desea revisar. 3. El sistema muestra todas las transacciones realizadas por ese usuario en el intervalo de tiempo ingresado.
Flujo alternativo:	1. En 2 el intervalo de tiempo no es válido.
Pos condición:	El sistema se encuentra listo para una nueva consulta.

Tabla 2.6 Descripción caso de uso Historial Transacciones

Nombre	Informe Cuentas
Descripción:	Obtener la fecha y hora, descripción, monto y el saldo de las transacciones realizadas durante un intervalo de tiempo por un determinado usuario.
Actores:	Financiero, Base de datos.
Precondición:	Haber sido autenticado como funcionario financiero.
Flujo normal:	1. El usuario ingresa la cédula de identidad del usuario del cual desea obtener la información. 2. Ingresa las fechas del intervalo de tiempo que desea revisar. 3. El sistema muestra todas las transacciones realizadas por ese usuario en el intervalo de tiempo ingresado.
Flujo alternativo:	1. En 1 no se encuentra el número de cédula. 2. En 2 el intervalo de tiempo no es válido.
Pos condición:	El sistema se encuentra listo para una nueva consulta.

Tabla 2.7 Descripción caso de uso Informe Cuentas

Nombre	Publicar mensajes
Descripción:	Publicar mensajes de interés general a los estudiantes.
Actores:	Bar Estudiantil, Biblioteca, Copiadora, Doctor, Financiero, Base de datos.
Precondición:	Haber sido autenticado como funcionario del bar estudiantil, biblioteca, copiadora, doctor o financiero.
Flujo normal:	1. El usuario ingresa el mensaje a publicar. 2. Ingresa la fecha de vigencia del mensaje. 3. El sistema guarda el mensaje en la base.
Flujo alternativo:	Ninguno
Pos condición:	Ninguno

Tabla 2.8 Descripción caso de uso Publicar mensajes

Nombre	Cancelar cita
Descripción:	Cancelar una cita médica reservada por un estudiante.
Actores:	Doctor, Base de datos.
Precondición:	Haber sido autenticado como funcionario Doctor
Flujo normal:	1. El usuario ingresa la fecha en la que quiere cancelar la cita. 2. El sistema devuelve una lista de citas para esa fecha. 3. El usuario selecciona la o las citas a cancelar. 4. Ingresa un mensaje y su fecha de vigencia. 5. El sistema guarda los cambios en la base.
Flujo alternativo:	1. En 2 no hay citas para ese día.
Pos condición:	Se han eliminado las citas seleccionadas de la base de datos.

Tabla 2.9 Descripción caso de uso Cancelar cita

- Sub-Casos de uso en el paquete Usuarios Sistema

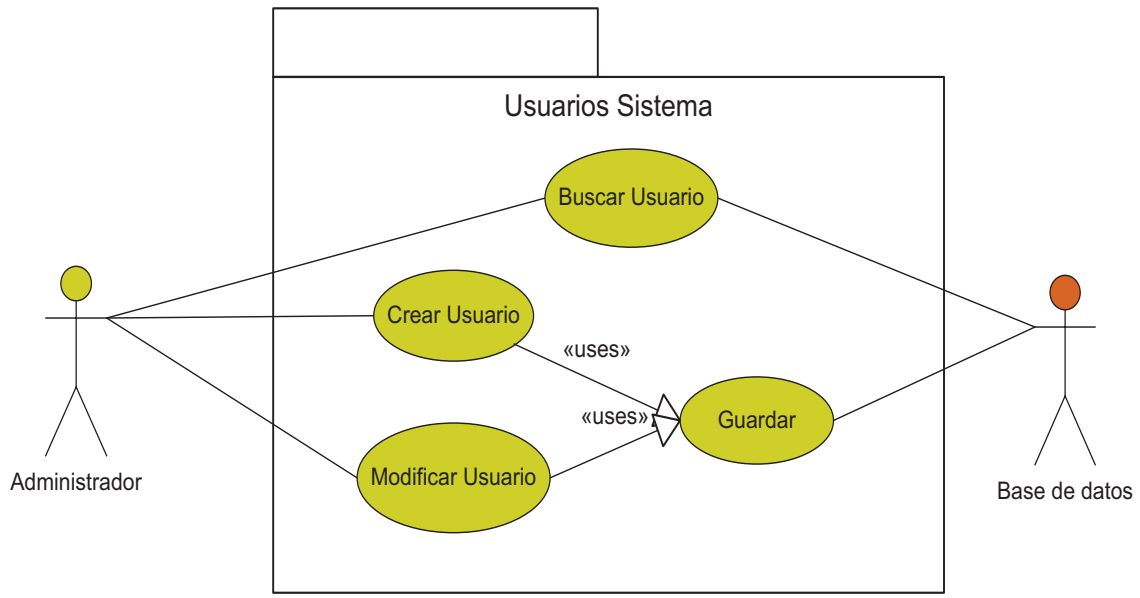

Figura 2.9 Sub-Casos de uso en el paquete Usuarios Sistema

Nombre	Crear Usuario
Descripción:	Permitir a un funcionario administrador crear nuevos usuarios para el sistema.
Actores:	Administrador, Base de datos.
Precondición:	Estar autenticado como funcionario Administrador.
Flujo normal:	1. El administrador selecciona la opción Crear Usuario. 2. Ingresa los datos del usuario. 3. Se toma la huella digital del nuevo usuario. 4. El sistema pide confirmación de la huella digital. 5. Se guarda la información en la base de datos.
Flujo alternativo:	1. En 4 no coinciden las huellas, por lo que se pide que se introduzca la huella nuevamente, luego de tres intentos fallidos, el proceso de toma de huella se reinicia. 2. En 5 la cédula ingresada ya contaba en el sistema y se notifica al administrador.

Tabla 2.10 Descripción Sub-Caso de uso Crear Usuario en el paquete Usuarios Sistema

Nombre	Modificar Usuario
Descripción:	Permitir a un funcionario administrador modificar los datos de un usuario del sistema.
Actores:	Administrador, Base de datos.
Precondición:	Estar autenticado como funcionario Administrador.
Flujo normal:	1. El administrador busca a un usuario determinado con el número de cédula. 2. El sistema muestra los datos y las opciones editar o eliminar. 3. El administrador selecciona editar, y modifica los datos deseados. 4. El sistema pide confirmación de la operación. 5. Se guarda la nueva información en la base de datos.
Flujo alternativo:	1. En 1 no existen coincidencias. 2. En 6 la cédula ingresada ya contaba en el sistema y se notifica al administrador.

Tabla 2.11 Descripción Sub-Caso de uso Modificar Usuario en el paquete Usuarios Sistema

- Sub-Casos de Uso en el paquete Datos Personales

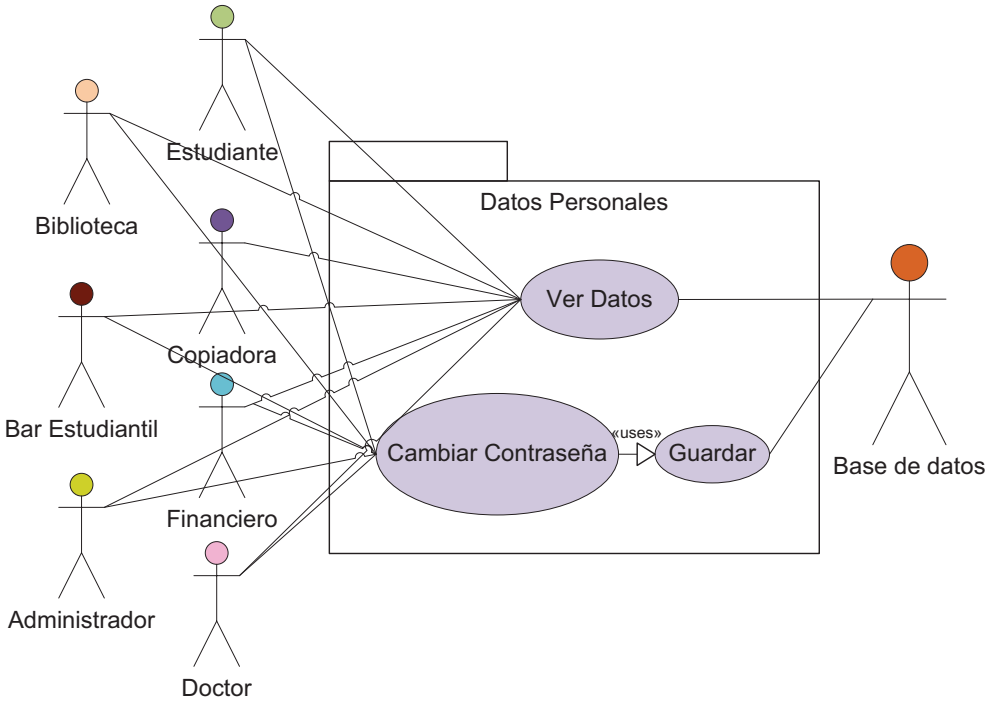

Figura 2.10 Sub-Casos de Uso en el paquete Datos Personales

Nombre	**Datos Personales**
Descripción:	Mostrar los datos personales y cambiar la contraseña.
Actores:	Administrador, Biblioteca, Bar Estudiantil, Copiadora, Doctor, Estudiante, Financiero, Base de datos.
Precondición:	Ser un usuario autenticado.
Flujo normal:	1. El usuario selecciona la opción Datos Personales, con lo cual el sistema despliega la información almacenada en la base de datos. 2. Selecciona la opción cambiar contraseña. 3. Ingresa la contraseña anterior, y la nueva contraseña. 4. El sistema verifica en la base de datos que la contraseña anterior sea la correcta. 5. Se almacena en la base de datos la nueva contraseña.
Flujo alternativo:	1. En 2 el usuario no desea cambiar la contraseña.
Pos condición:	Ninguna.

Tabla 2.12 Descripción Sub-Casos de uso en el paquete Datos Personales

- Sub-Casos de uso en el Paquete Información Estudiante

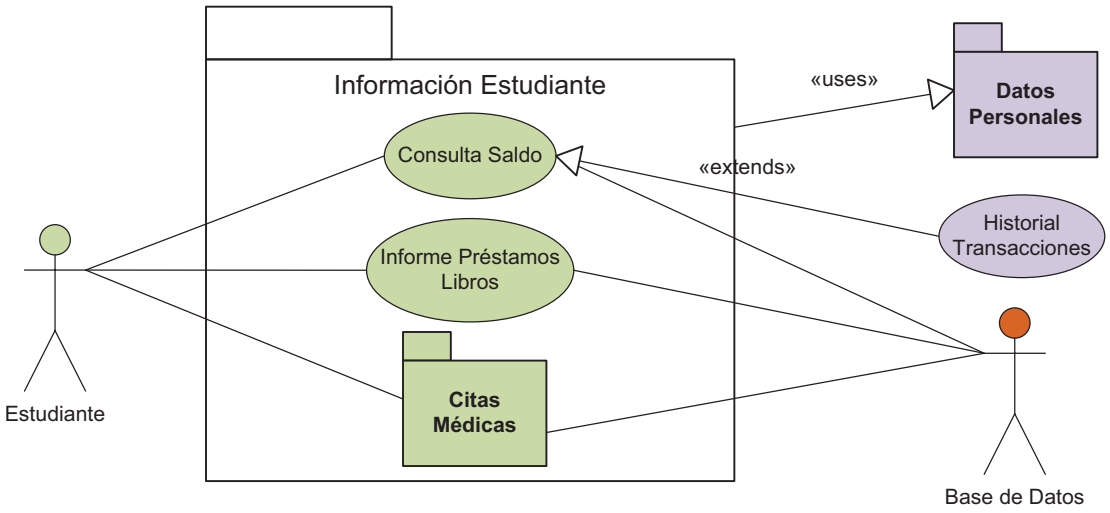

Figura 2.11 Sub-Casos de Uso en el Paquete Información Estudiante

Nombre	**Información Estudiante**
Descripción:	Mostrar los datos personales, historial de transacciones, los préstamos de libros de las diferentes bibliotecas, y planificar sus citas médicas.
Actores:	Estudiante, Base de datos.
Precondición:	Estar autenticado como estudiante.
Flujo normal:	1. El sistema muestra al estudiante todas las opciones disponibles para su rol. 2. El estudiante selecciona cualquiera de las opciones: Consulta saldo ver el saldo restante y si lo desea el historial de transacciones, Datos personales, Informe préstamos libros para ver los detalles de todos los préstamos que tenga activos en las diferentes bibliotecas, o Citas Médicas. 3. El sistema busca en la base de datos los datos solicitados y los muestra.
Flujo alternativo:	Ninguno.
Pos condición:	Ninguna.

Tabla 2.13 Descripción Sub-Casos de uso en el paquete Información Estudiante

- Sub-Casos de uso en el Paquete Citas Médicas

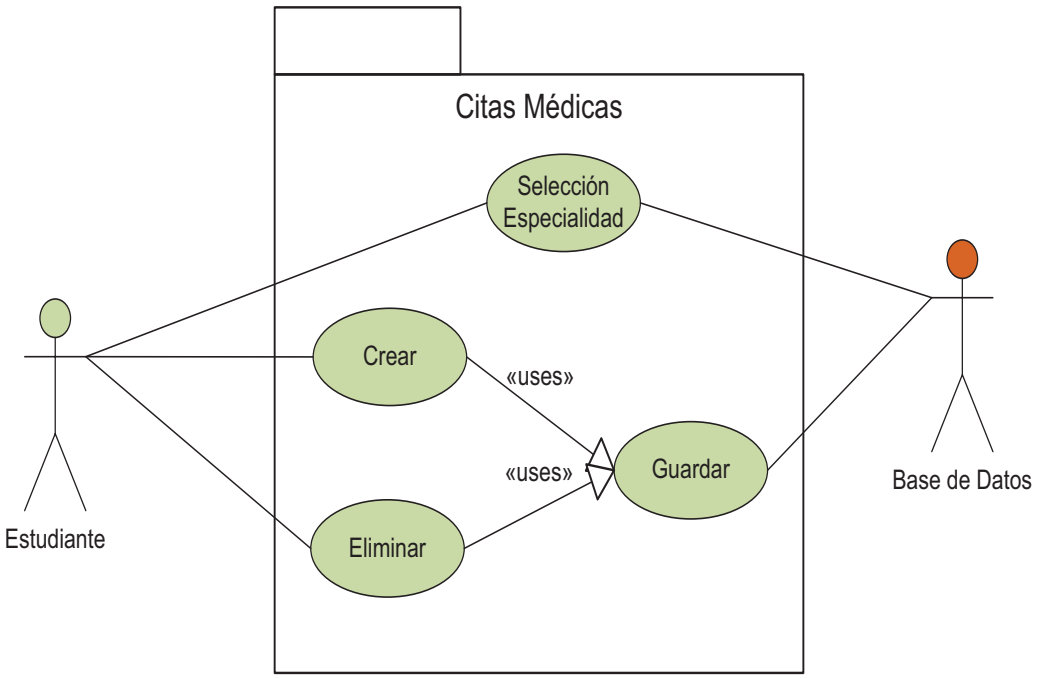

Figura 2.12 Sub-Casos de Uso en el Paquete Citas Médicas

Nombre	**Crear**
Descripción:	Crear citas con un doctor del servicio médico del campus.
Actores:	Estudiante, Base de datos.
Precondición:	Estar autenticado como estudiante.
Flujo normal:	1. El estudiante selecciona la especialidad de su interés. 2. Selecciona el nombre del doctor y la opción crear. 3. Elige una fecha y hora disponibles. 4. Guarda los cambios realizados en la base de datos.
Flujo alternativo:	Ninguno.
Pos condición:	La base de datos tiene registrada una nueva cita médica.

Tabla 2.14 Descripción Sub-Caso de uso Crear en el Paquete Citas Médicas

Nombre	Eliminar
Descripción:	Eliminar una cita en el servicio médico.
Actores:	Estudiante, Base de datos.
Precondición:	Estar autenticado como estudiante.
Flujo normal:	1. El estudiante selecciona la especialidad de su interés. 2. Selecciona el nombre del doctor y la opción eliminar. 3. Selecciona una cita ya creada anteriormente. 4. Guarda los cambios realizados.
Flujo alternativo:	1. En 3 no hay citas registradas.
Pos condición:	Se ha eliminado una cita ya creada.

Tabla 2.15 Descripción Sub-Caso de uso Eliminar en el Paquete Citas Médicas

- Sub-Casos de uso Paquete Cobrar Consumo

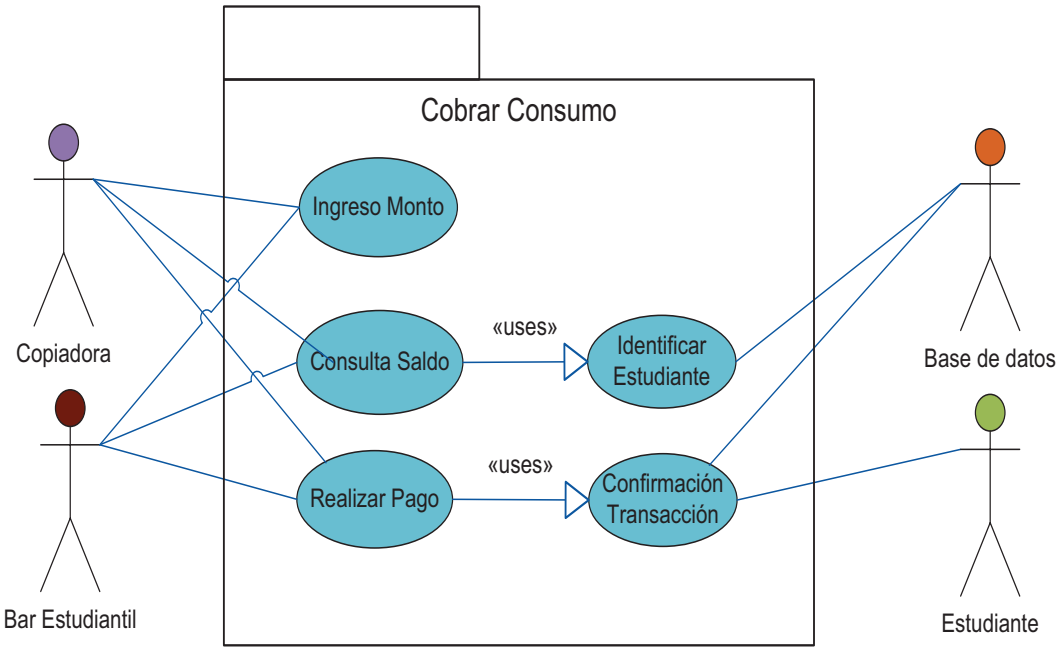

Figura 2.13 Sub-Casos de Uso en el Paquete Cobrar Consumo

Nombre	Cobrar Consumo
Descripción:	Permitir que un funcionario de Bar Estudiantil o Copiadora debite el valor de los consumos de la cuenta de un estudiante.
Actores:	Bar Estudiantil, Copiadora, Estudiante, Base de datos.
Precondición:	Estar autenticado como funcionario de Bar Estudiantil o Copiadora.
Flujo normal:	1. El funcionario ingresa la cantidad a ser debitada. 2. Consulta en la base de datos si el saldo del estudiante es suficiente para realizar la transacción ingresando el número de cédula del mismo. 3. Se muestra la cantidad a cancelar al estudiante, confirma la transacción mediante su huella digital y se la valida con la almacenada en la base de datos. 4. Se almacena en la base de datos la transacción debitando el monto de la cuenta del estudiante y sumándola a la cuenta del funcionario.
Flujo alternativo:	1. En 2 el número de cédula es incorrecto. 2. En 3 el saldo no es suficiente. 3. En 3 la huella digital no coincide con la almacenada en la base de datos.
Pos condición:	El sistema se encuentra listo para recibir un nuevo cobro.

Tabla 2.16 Descripción de Sub-Casos de Uso en el Paquete Cobrar Consumo

- Sub-Casos de uso Paquete Catálogo Libros

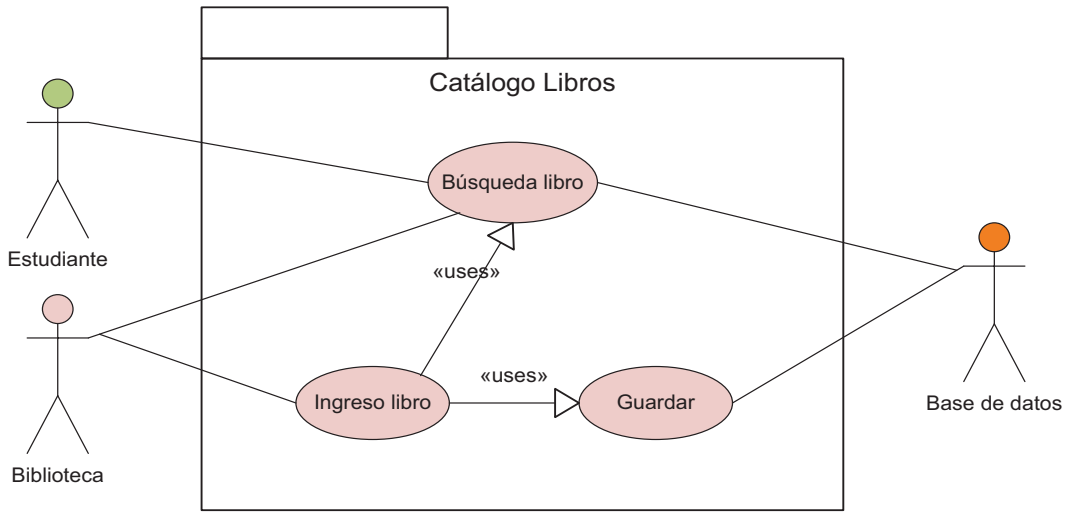

Figura 2.14 Sub-Casos de Uso en el Paquete Catálogo Libros

Nombre	Catálogo Libros
Descripción:	Conseguir la ubicación y existencias de un libro por título o autor, y verificar los préstamos actuales de dicho libro.
Actores:	Biblioteca, Estudiante, Base de datos.
Precondición:	Haber sido autenticado como funcionario de biblioteca o como estudiante.
Flujo normal:	1. El usuario escoge el modo de búsqueda del libro por título o autor, e ingresa los datos de búsqueda. 2. La base de datos verifica la información dada con la almacenada. 3. Si el usuario autenticado es funcionario de biblioteca, el sistema muestra la estantería en que se encuentra el libro, número de copias, y préstamos de haberlos en esa biblioteca, si es estudiante muestra la información anterior para cada biblioteca del campus. 4. Para el ingreso de libros, el funcionario busca en el sistema si ya existe el libro en cuestión. 5. La base consulta la existencia del libro y devuelve los resultados. 6. Si el libro existe, el bibliotecario simplemente ingresa el número de copias de ese libro a su biblioteca y la estantería, si el libro no existe ingresa el nombre, autor, año, estantería, y número de copias. 7. La base almacena la información dada.
Flujo alternativo:	1. En 3 no se encuentran coincidencias para la búsqueda.

Tabla 2.17 Sub-Casos de uso en el paquete Catálogo Libros

- Sub-Casos de uso en el Paquete Cobrar Multas

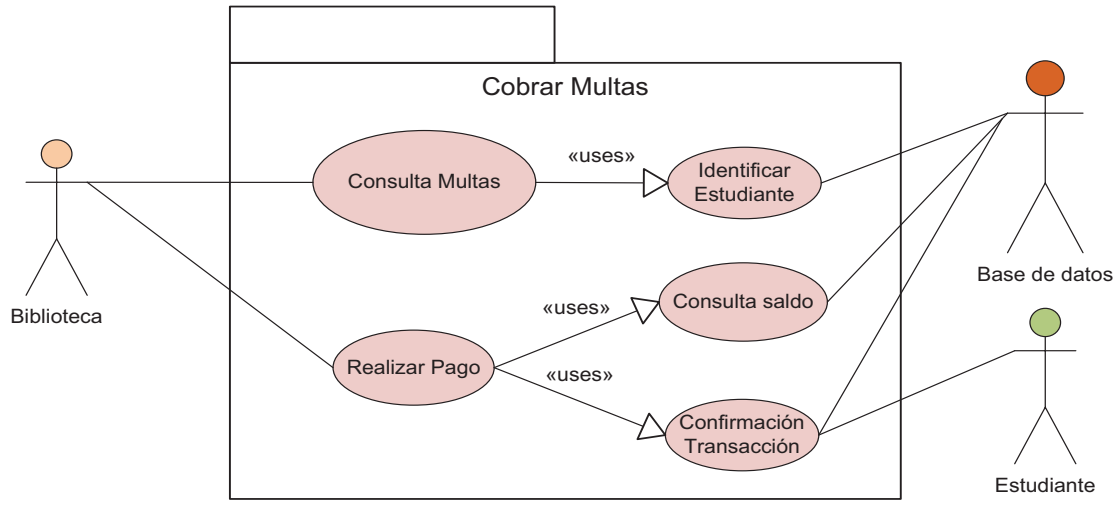

Figura 2.15 Sub-Casos de uso en el paquete Cobrar Multas

Nombre	Cobrar Multas
Descripción:	Permitir que un funcionario de Biblioteca realice el pago de multas de préstamos de las cuentas de los estudiantes.
Actores:	Biblioteca, Estudiante, Base de datos.
Precondición:	Estar autenticado como funcionario de Biblioteca.
Flujo normal:	1. El funcionario consulta las multas pendientes del estudiante mediante su número de cédula. 2. Se muestra la cantidad a cancelar al estudiante, el cual confirma la transacción mediante su huella digital, y se la valida en la base de datos del sistema. 3. Se almacena en la base de datos la transacción debitando el monto de la cuenta del estudiante y sumándola a la cuenta de la biblioteca.
Flujo alternativo:	1. En 1 no hay multas pendientes. 2. En 2 no hay saldo suficiente. 3. En 2 no coincide la huella.

Tabla 2.18 Descripción Sub-Casos de uso en el paquete Cobrar Multas

- Sub-Casos de uso de paquete Gestión Préstamo Libro

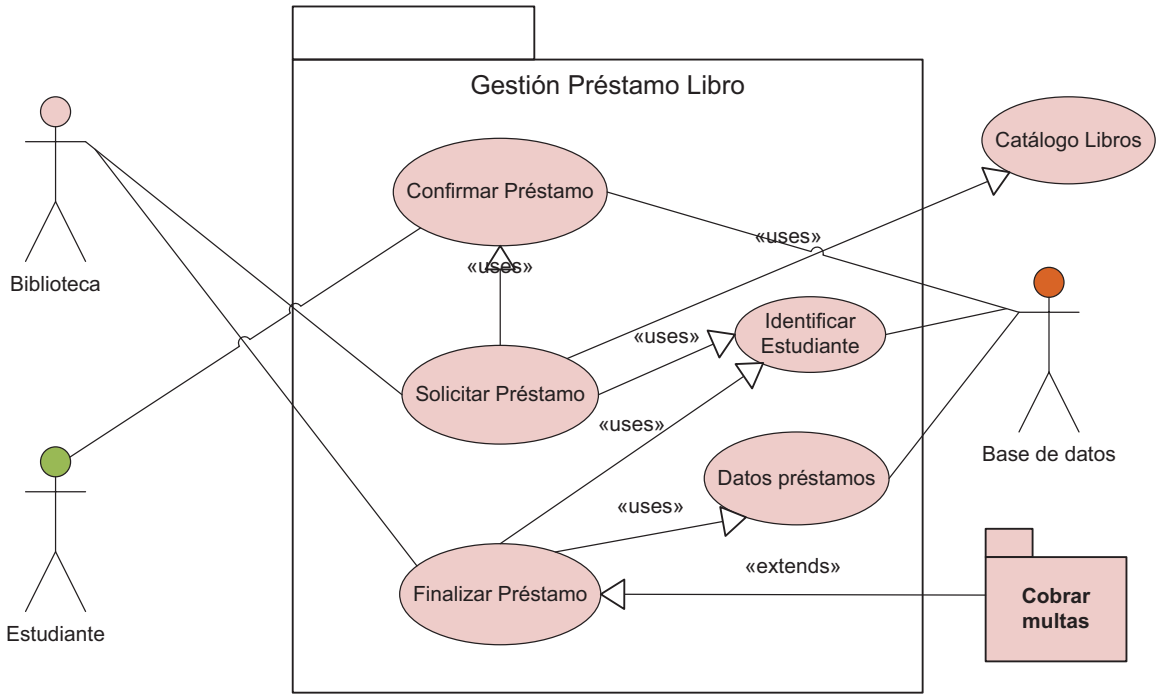

Figura 2.16 Sub-Casos de uso en el paquete Gestión Préstamo Libro.

Nombre	Gestión Préstamo Libro
Descripción:	Permitir que un funcionario de Biblioteca realice préstamos de libros a los estudiantes.
Actores:	Biblioteca, Estudiante, Base de datos.
Precondición:	Estar autenticado como funcionario de Biblioteca.
Flujo normal:	1. Para realizar un préstamo el funcionario verifica que haya al menos 2 ejemplares disponibles. 2. El funcionario verifica si el estudiante está registrado en la base de datos mediante su número de cédula. 3. Se ingresa la fecha de devolución del libro y se confirma el préstamo mediante la huella dactilar del estudiante. 4. Se almacena en la base de datos la información del préstamo. 5. Para realizar una devolución el funcionario identifica al estudiante mediante su número de cédula. 6. La base de datos devuelve los datos de los préstamos que tenga el estudiante y multas de haberlas. 7. El funcionario selecciona los libros a devolver y cobra las multas existentes. 8. La base de datos almacena los datos.
Flujo alternativo:	1. En 1 hay menos de 2 ejemplares disponibles. 2. En 2 y 5 el estudiante no está registrado. 3. En 3 la huella dactilar no coincide con la almacenada en la base de datos.

Tabla 2.19 Descripción Sub-Casos de uso en el Paquete Gestión Préstamo Libro

- Sub-Casos de Uso en el Paquete Depósitos y Retiros

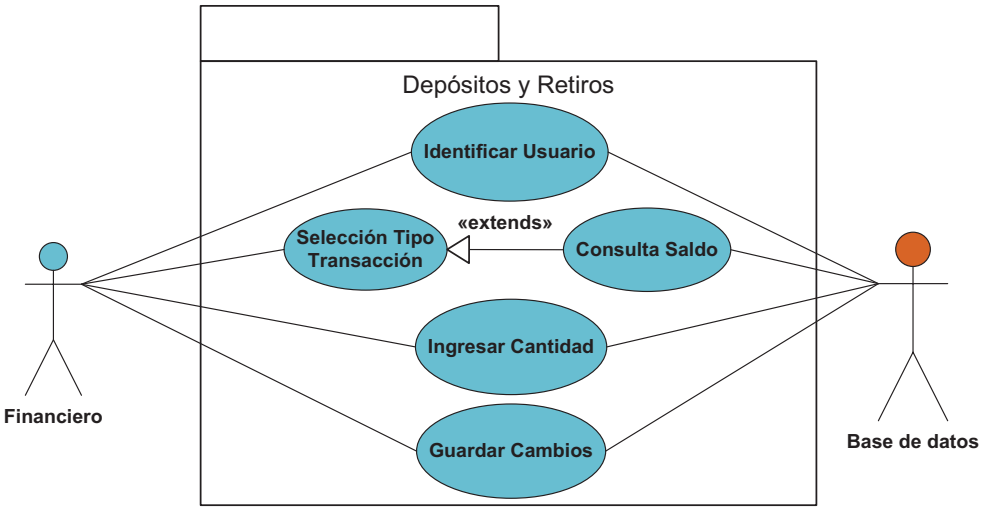

Figura 2.17 Sub-Casos de uso en el paquete Depósitos y Retiros

Nombre	Depósitos y Retiros
Descripción:	Permitir que un funcionario financiero realice depósitos en las cuentas de los estudiantes o retiros de las cuentas de los otros funcionarios.
Actores:	Financiero, Base de datos.
Precondición:	Estar autenticado como funcionario Financiero.
Flujo normal:	1. El funcionario identifica al usuario con su número de cédula. 2. Selecciona el tipo de transacción que realizará. 3. En caso de que se realice un retiro se consulta el saldo que tiene disponible en su cuenta. 4. Ingresa la cantidad. 5. Se guardan los cambios.
Flujo alternativo:	1. En 1 no existe el número de cédula.
Pos condición:	El sistema se encuentra listo para una nueva transacción.

Tabla 2.20 Descripción Sub-Casos de uso en el paquete Depósitos y Retiros

- Sub-Casos de uso en el paquete Historia Clínica

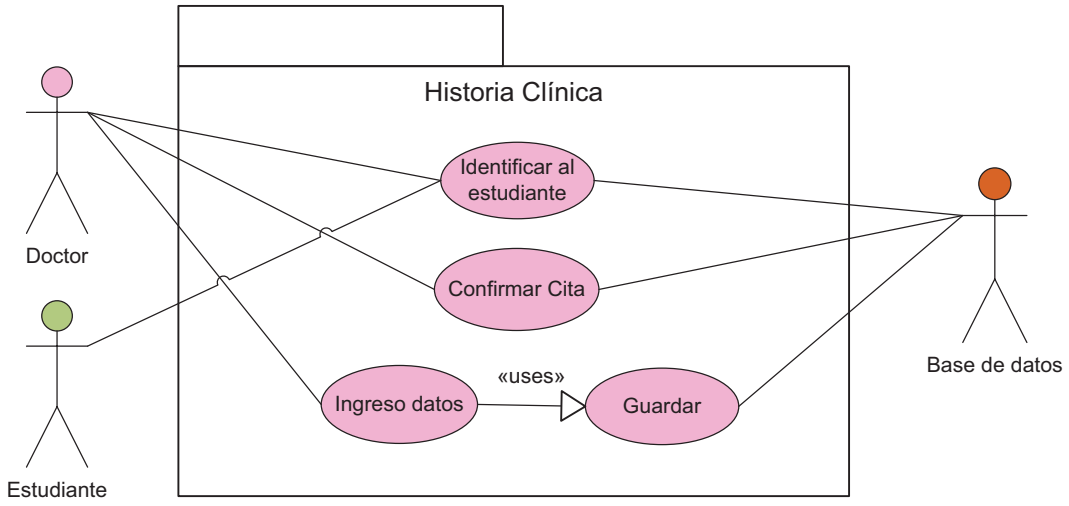

Figura 2.18 Sub-Casos de uso en el paquete Historia Clínica

Nombre	Historia Clínica
Descripción:	Permitir que un funcionario de centro médico ingrese datos en una historia clínica de un estudiante.
Actores:	Doctor, Estudiante, Base de datos.
Precondición:	Estar autenticado como funcionario Doctor.
Flujo normal:	1. El funcionario identifica al usuario con su número de cédula y su huella dactilar. 2. Confirma que tenga una cita programada para ese momento. 3. Ingresa los datos necesarios en los espacios dispuestos para ello y se guardan los datos.
Flujo alternativo:	2. En 1 no existe el número de cédula. 3. En 1 no coincide la huella.
Pos condición:	Ninguna.

Tabla 2.21 Descripción de Sub-Casos de uso en el paquete Historia Clínica

2.3 DISEÑO DE BASE DE DATOS

Durante las últimas décadas la utilización de bases de datos ha tenido gran impacto en la mayoría de áreas en las que se utilizan computadores debido a las enormes ventajas frente al sistema de archivos, entre las cuales se puede citar impedir la redundancia de datos, evitar la inconsistencia de datos, facilidad al acceso de la información, seguridad, entre otras; lo cual ha permitido que éstas tengan un mayor rendimiento y crecimiento tanto en el ámbito empresarial, comercio electrónico, líneas aéreas, ingeniería, etc.

Los datos son hechos conocidos que se pueden grabar y que tienen un significado implícito; el diseño de base de datos trata de definir la estructura de los mismos de un sistema de información determinado mediante un conjunto de programas.

Un sistema de administración de datos (DBMS, *database management system*) es una colección de programas que permite a los usuarios crear y mantener una

base de datos. Es un sistema de software de propósito general que facilita los procesos de definición, construcción, manipulación y compartición de bases de datos entre varios usuarios y aplicaciones. Definir una base de datos implica especificar los tipos de datos, estructuras y restricciones de los datos que se almacenarán en la base de datos. [L6]

2.3.1 ETAPAS DEL DISEÑO DE BASES DE DATOS [L7]

El diseño de una base de datos suele dificultarse debido a la complejidad de la información y la cantidad de requisitos de los sistemas de información, por lo que es conveniente descomponer el proceso en tres etapas las cuales dan un resultado intermedio que sirve de partida para la siguiente con lo que se simplifica el proceso.

- En la etapa del diseño conceptual se obtiene una estructura de la información de la base de datos independiente de la tecnología a utilizar, con lo cual permite concentrarse únicamente en la estructuración de la información.

- En la etapa de diseño lógico se transforma el resultado anterior para adaptarlo al DBMS con el que se desea implementar la base de datos.

- En la etapa del diseño físico se transforma la estructura obtenida en el diseño lógico con el objetivo de conseguir mayor eficiencia y se completa con aspectos de implementación física como la elección de estructuras físicas de implementación de las relaciones, selección del tamaño de las memorias intermedias, etc., las cuales dependerán del DBMS.

2.3.2 MODELOS DE LOS DATOS [L8]

El modelo de datos es una colección de herramientas conceptuales para describir los datos, relaciones entre datos, semántica y las restricciones de consistencia. Se dará una breve descripción de dos de los modelos más utilizados.

2.3.2.1 Modelo entidad-relación

El modelo de datos entidad-relación (E-R) está basado en una percepción del mundo real que consta de *entidades* los cuales son objetos en el mundo real y que son distinguibles de otros objetos, y de *relaciones* entre estos objetos. Las entidades se describen en una base de datos mediante un conjunto de atributos. Una relación es una asociación entre varias entidades. El conjunto de todas las entidades del mismo tipo se denomina conjunto de entidades, y el conjunto de todas las relaciones del mismo tipo se denomina conjunto de relaciones.

La estructura lógica general de una base de datos se puede expresar gráficamente mediante un *diagrama* ER, que consta de lo siguiente:

- *Rectángulos,* que representan conjuntos de entidades.
- *Elipses,* que representan atributos.
- *Rombos*, que representan relaciones entre conjuntos de entidades.
- *Líneas,* que unen los atributos con los conjuntos de entidades y los conjuntos de entidades con las relaciones.

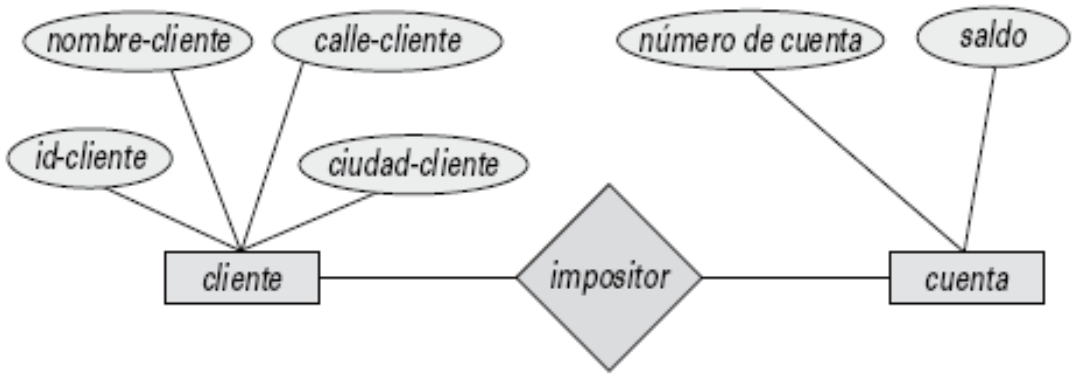

Figura 2.19 Ejemplo de diagrama E-R [L8]

El modelo E-R representa también ciertas restricciones que los contenidos de la base de datos deben cumplir, como la correspondencia de cardinalidades que expresa el número de entidades con la que otra entidad se puede asociar. En la

Figura 2.19 se muestra un ejemplo básico de este tipo de diagrama. El modelo E-R se utiliza habitualmente en la etapa del diseño conceptual.

2.3.2.2 Modelo relacional

El modelo relacional utiliza un grupo de tablas para representar los datos y las relaciones entre ellos. Cada tabla está compuesta por varias columnas, y cada columna tiene un nombre único. Cada tabla contiene registros de un tipo particular que define un número fijo de campos o atributos. Las columnas de la tabla corresponden a los atributos del tipo de registro.

El modelo de datos relacional es el modelo más ampliamente usado y se encuentra en un nivel de abstracción inferior al modelo E-R, por lo que a menudo se realiza el modelo E-R y luego se lo traduce al modelo relacional en la etapa de diseño lógico. La Figura 2.20 muestra un ejemplo básico de este diagrama.

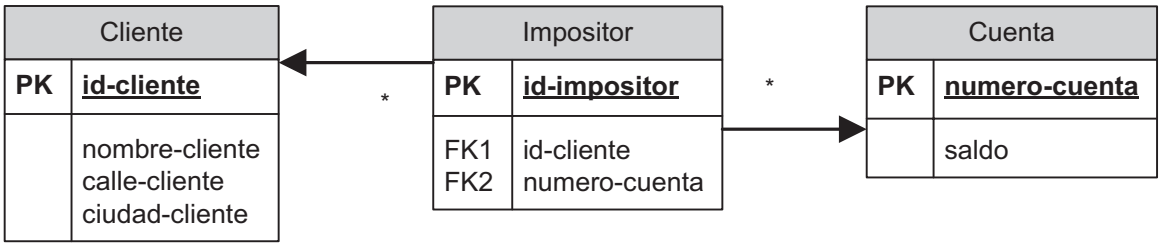

Figura 2.20 Ejemplo de diagrama relacional. [L8]

Los DBMS más populares que trabajan con este tipo de modelo están: *MySQL*, *PostgreSQL*, *Oracle*, *DB2*, *INFORMIX*, *Interbase*, *FireBird*, *Sybase* y *Microsoft SQL Server*.

2.3.3 DIAGRAMA DE LA BASE DE DATOS

Debido a que se eligió a Microsoft SQL Server 2008 como DBMS, se mostrará un diagrama del tipo relacional, mostrado en la Figura 2.21.

La definición de las tablas se encuentra en el Anexo C.

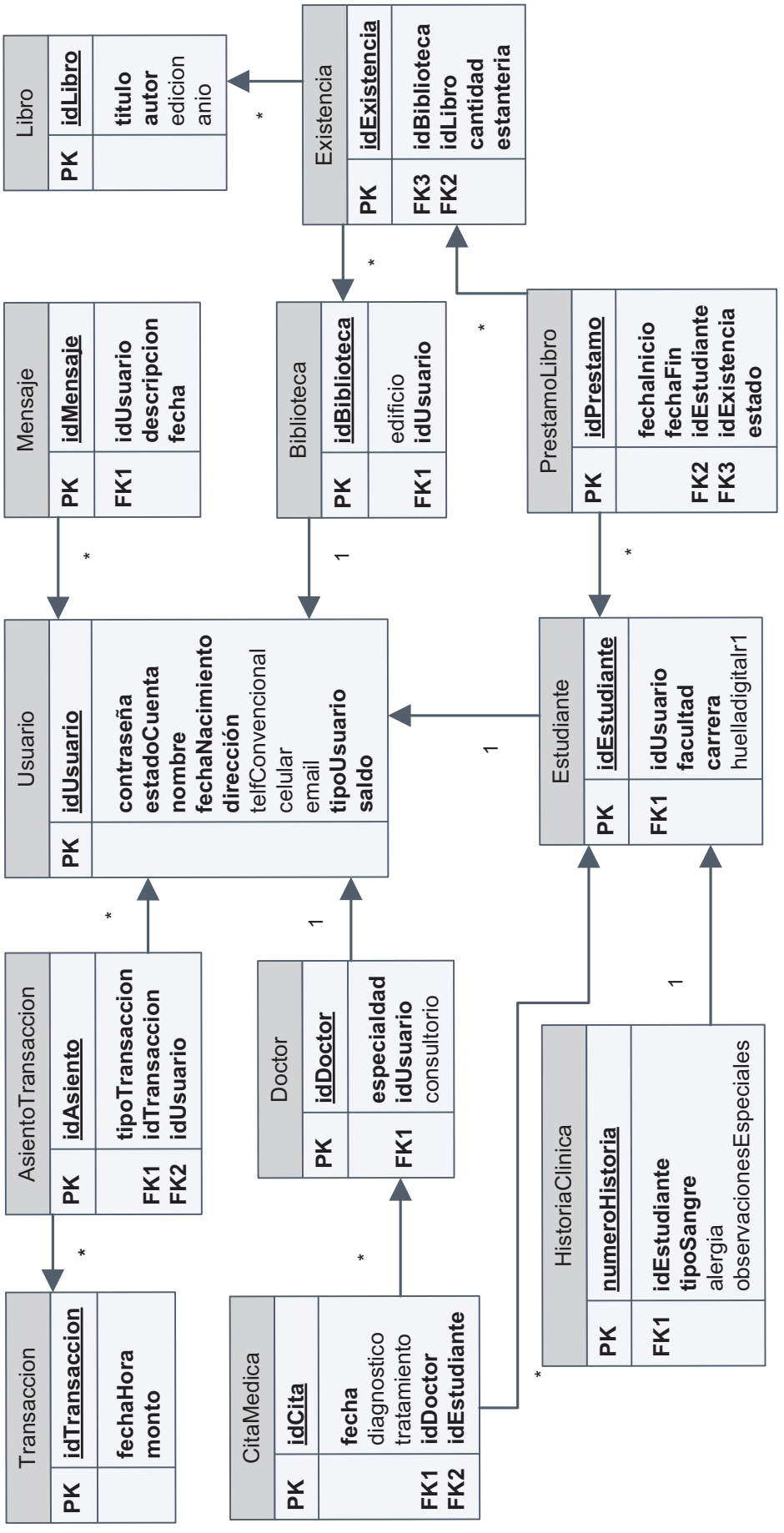

Figura2.21 Diagrama de la base de datos de Multiservicios Estudiantiles

2.4 DIAGRAMAS UML

2.4.1 DIAGRAMA DE CLASES [L9]

Los diagramas de clases muestran un conjunto de clases, interfaces, colaboraciones y sus relaciones. Estos diagramas son importantes ya que no sólo sirven para visualizar, especificar y documentar modelos estructurales, sino también para construir sistemas ejecutables aplicando ingeniería directa e inversa.

Los diagramas de clases se usan para modelar la vista de diseño estática de un sistema; esta vista soporta principalmente los requisitos funcionales de un sistema, los servicios que el sistema debe proporcionar a los usuarios finales. Principalmente, esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas.

Cuando se modela la vista de diseño estática de un sistema, normalmente se utilizarán los diagramas de clases de una de estas tres formas:

- Para modelar el vocabulario de un sistema, es decir se lo utiliza para especificar las abstracciones que son parte del sistema y sus responsabilidades.
- Para modelar colaboraciones simples, permitiendo visualizar y especificar un conjunto de clases, interfaces y otros elementos que colaboran entre sí.
- Para modelar el esquema lógico de una base de datos, puesto que mediante diagramas de clases se puede ayudar al diseño conceptual de una base de datos relacional u orientada a objetos, donde se almacenará la información persistente.

2.4.1.1 Diseño del Diagrama de Clases

El sistema estará dividido en tres capas, por lo que las clases que se encuentran en la capa de negocios son las predominantes, y se lo muestra en la Figura 2.22.

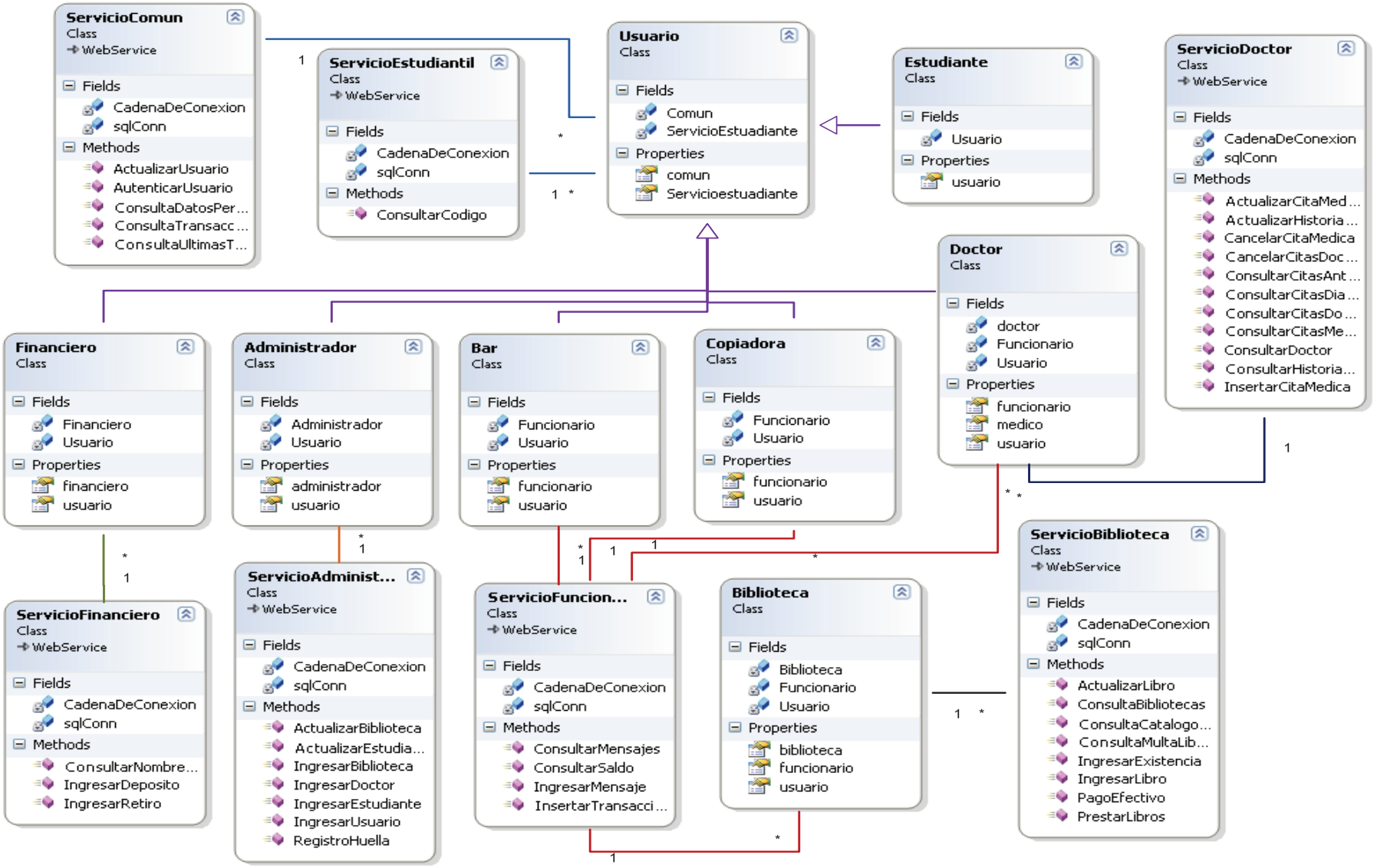

Figura 2.22 Diagrama de clases

2.4.2 DIAGRAMA DE SECUENCIA [PW36]

Un diagrama de secuencia indica forma en la que se comunican los módulos o clases que forman parte del sistema para realizar una tarea, mientras transcurre el tiempo de vida de los mismos, teniéndose detalles de la implementación de un escenario.

Debido a la gran extensión del proyecto, solo se mostrará un diagrama de secuencia correspondiente al cobro de consumos en copiadoras o bares en la Figura 2.23.

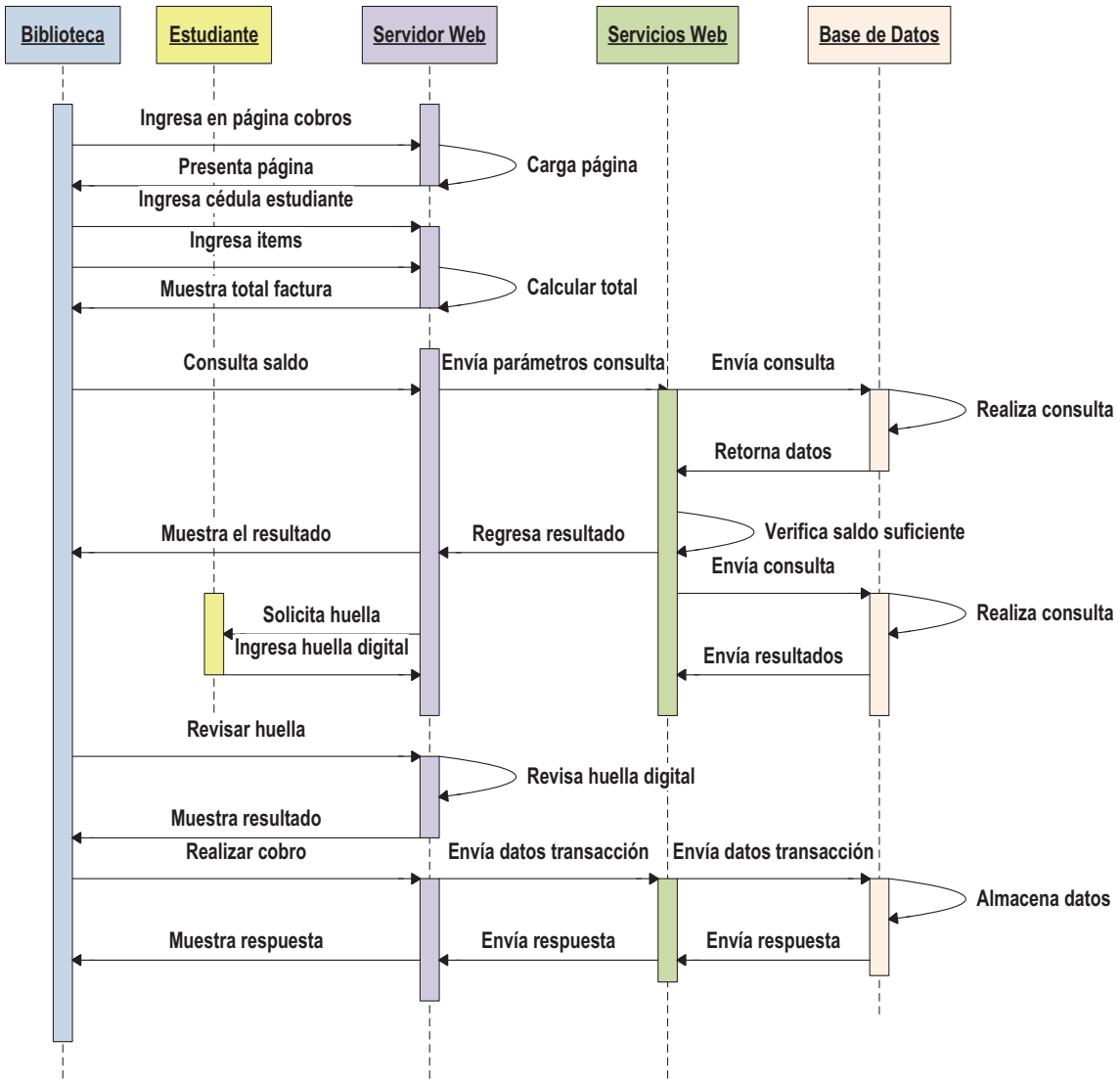

Figura 2.23 Diagrama de secuencia para Pago Consumo

2.4.3 DIAGRAMA DE ACTIVIDADES [PW37]

Estos diagramas se usan para mostrar la secuencia de actividades, mostrando el flujo de trabajo de inicio a fin, detallando rutas de decisiones existentes en el progreso de una actividad. También se usan para describir procesos en paralelo, siendo útiles para Modelado de negocios. Igual que en el caso anterior solo se mostrará un ejemplo. En la Figura 2.24 se muestra el diagrama de actividades para el préstamo de libros en una biblioteca.

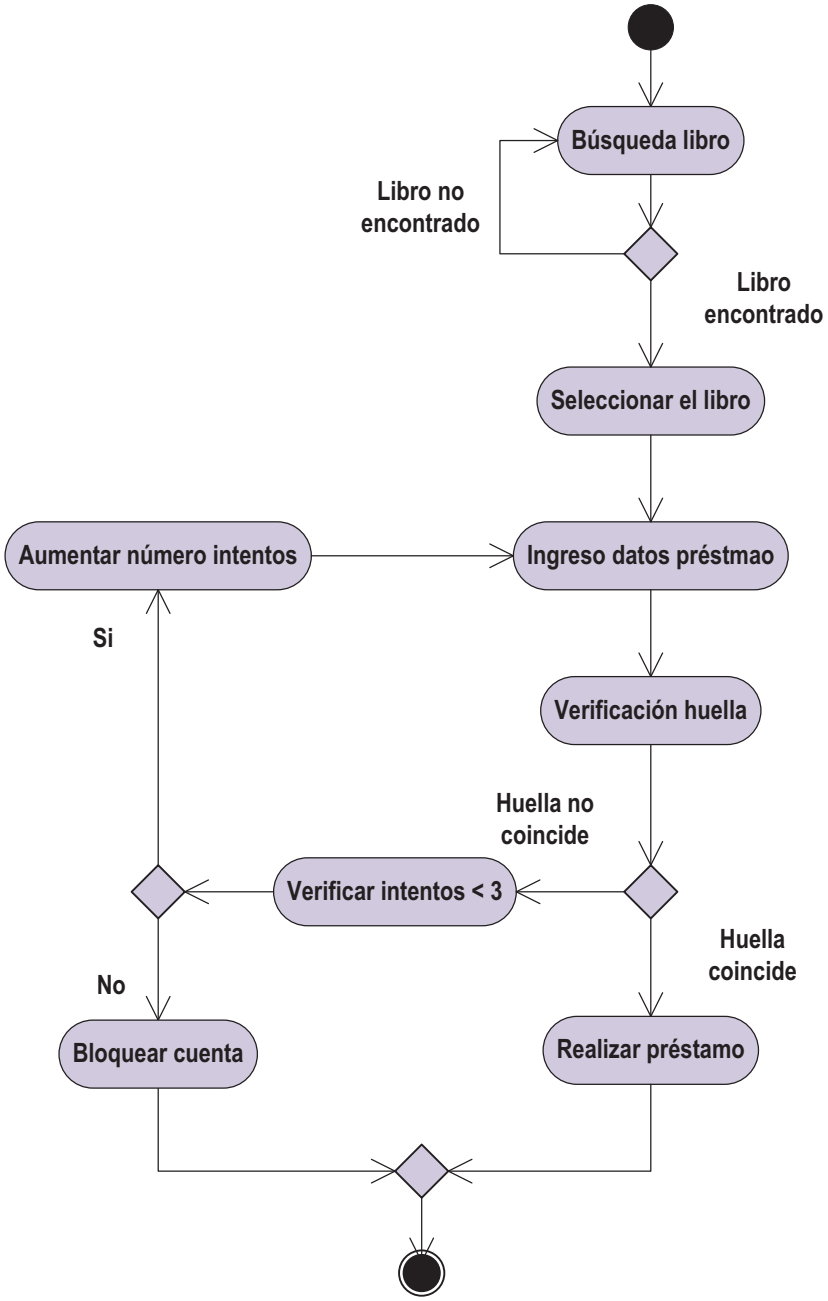

Figura 2.24 Diagrama de Actividades en Préstamo Libro

2.4.4 DIAGRAMA DE DESPLIEGUE [PW38]

Este tipo de diagrama es utilizado para modelar el Hardware de un sistema mediante las relaciones físicas entre los componentes de hardware y software, es decir describen su topología, además de la configuración en funcionamiento del sistema en términos de procesadores, dispositivos y componentes de software. En la Figura 2.25 se muestra el diagrama de despliegue del sistema.

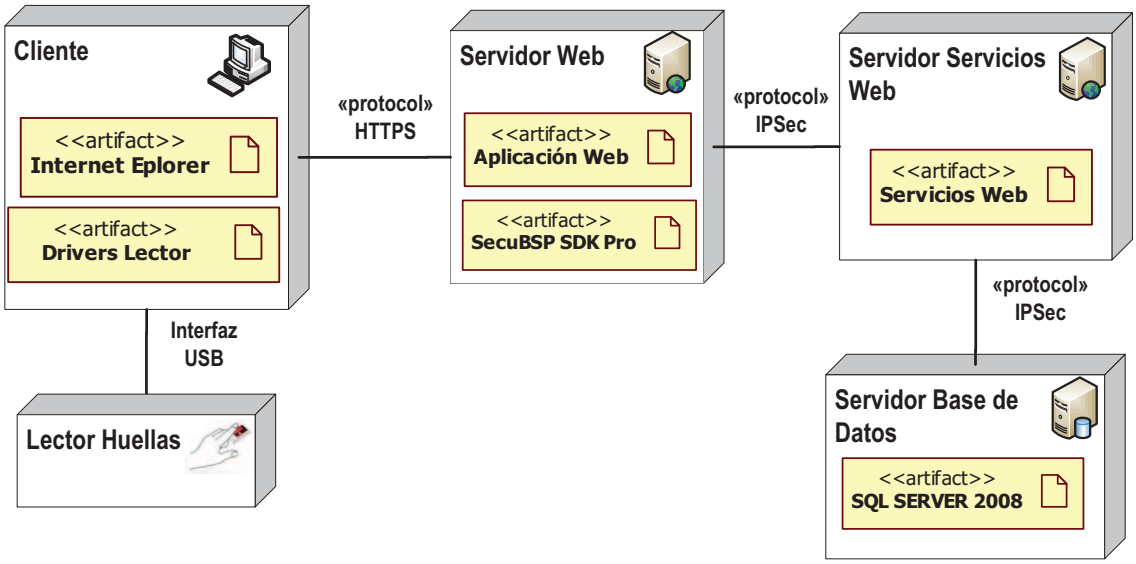

Figura 2.25 Diagrama de Despliegue

CAPÍTULO III

3. IMPLEMENTACIÓN DEL SISTEMA

El sistema se desarrolló para una topología física tipo estrella como se muestra en la Figura 3.1, utilizando la metodología de desarrollo RUP (*Rational Unified Process*), en una arquitectura de tres capas en tres niveles, por lo que el presente capítulo explicará, el desarrollo de cada una de ellas y la configuración realizada en los servidores para agregar las características y funciones requeridas, así como la implementación de los parámetros de seguridad utilizados.

3.1 CAPA DE PRESENTACIÓN

La capa de presentación se constituye de 14 páginas web para que los usuarios puedan interactuar con el sistema, las cuales se realizaron utilizando *Microsoft Visual Studio 2010* en lenguaje C#, además del manejo de *Java Script, Ajax Control Toolkit* y *CSS* los cuales mejoran la experiencia visual y logran un entorno amigable e intuitivo.

La página principal es llamada *Site.Master*, de la cual las demás páginas heredan la cabecera, el menú según el rol, y el botón de cerrar sesión; las páginas restantes se las puede clasificar en comunes para todos los usuarios y de acuerdo al rol de usuario.

Las páginas comunes son:

- *Default*: la cual sirve para el ingreso al sistema mediante autenticación.
- *About:* muestra información sobre la versión de la aplicación y sus desarrolladores.
- *DatosPersonales:* muestra la información personal de cada usuario y permite el cambio de contraseña.

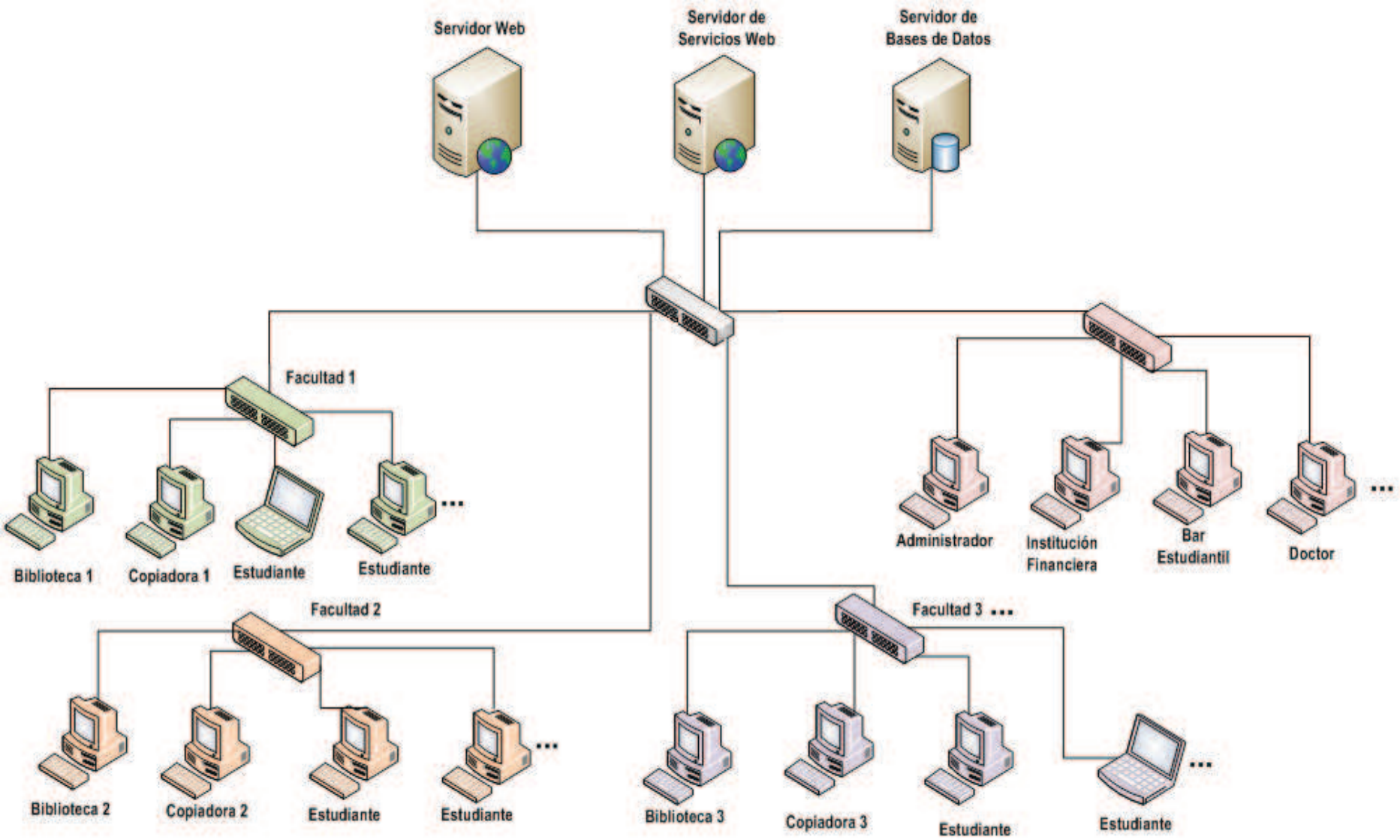

Figura 3.1 Diagrama general de red

- *Menú:* Pide confirmación sobre el rol con el que se ingresó y da paso al armado del menú del usuario según el rol.
- *ReporteTransacciones*: muestra las últimas diez transacciones, o las realizadas durante un período de tiempo. Ésta página no puede ser visualizada por el Administrador o Doctores, ya que ellos no realizan transacciones.

Las páginas de acuerdo al rol de usuario son:

- *Ingresos:* Página del Administrador y del Biblioteca, permite el ingreso de usuarios al administrador, y el ingreso de libros al bibliotecario.
- *Modificar:* Página del Administrador y de Biblioteca, permite la modificación de usuarios al administrador, y de libros al bibliotecario.
- *CatálogoLibros:* Página de Biblioteca y Estudiante, permite a los usuarios consultar el catálogo de libros de una biblioteca, y hacer el préstamo de un libro exclusivamente al bibliotecario.
- *CobroXItems:* Página de Biblioteca, Bar y Copiadora: Muestra a los usuarios un listado de los ítems a ser cancelados por el estudiante.
- *TransaccionesFinanciero:* Página de servicio Financiero, Permite realizar depósitos o retiros de un usuario.
- *HistoriaClinica:* Página de Doctor, muestra un listado desplegable de las citas para el día actual, además permite modificar la historia clínica de un estudiante, así como también el diagnóstico y tratamiento de la cita actual previo la verificación del estudiante por huella digital.
- *Mensajes:* Página de Copiadora, Bar, Biblioteca, Doctor y Financiero; permite a los usuarios ingresar un mensaje que será mostrado a todos los estudiantes, exclusivamente para el Doctor permite consultar las citas médicas de una fecha posterior a la actual y cancelarlas parcial o totalmente con justificativo.

- *CitasMedicas:* Página de Estudiante, permite escoger una especialidad médica y realizar una cita en un horario establecido.

3.1.1 DESCRIPCIÓN PÁGINAS WEB

3.1.1.1 Página Site.Master

Como se explicó anteriormente, esta página contiene algunos elementos que se mostrarán en las demás páginas; los cuales son: un logotipo del sistema, el menú que se presentará mediante previa confirmación del rol de usuario que ingresa, un panel que contendrá las demás páginas y un botón que permitirá al usuario cerrar su sesión. La Figura 3.2 muestra la página *Site.Master* conteniendo a la página menú.

Figura 3.2 Elementos Site.Master

3.1.1.2 Página Default

Esta página mostrada en la Figura 3.3 contiene una imagen referencial de los servicios prestados por el sistema, un panel que contiene tres secciones: la primera contiene cuatro imágenes con enlaces hacia las referenciadas; la segunda contiene los parámetros para la autenticación e ingreso del usuario al sistema; y la tercera contiene un acordeón explicativo de los servicios que ofrece el sistema. Este acordeón es un control dado por el *Ajax Control Toolkit*, que exclusivamente debe ir colocado dentro de un control panel de ASP.NET.

Figura 3.3 Elementos página Default

3.1.1.3 Página About

Esta página mostrada en la Figura 3.4, contiene información general sobre los creadores y la versión del sistema.

Figura 3.4 Página web About

3.1.1.4 Página DatosPersonales

Contiene una lista de los datos personales de un usuario (Figura 3.5), además un botón que habilitará el cambio de contraseña mediante una pequeña ventana para el ingreso con confirmación de la misma mostrada en la Figura 3.6.

Figura 3.5 Elementos página DatosPersonales

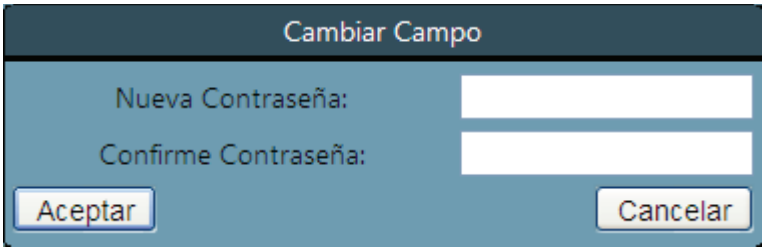

Figura 3.6 Ventana para cambio de contraseña

3.1.1.5 Página Menu

Al igual que la página *Default* contiene la imagen representativa de los servicios y a continuación un panel con tres secciones: la primera es idéntica a la de la página *Default*; la segunda contiene un mensaje de verificación del rol del usuario que al presionar el botón *Continuar* cambia por un mensaje de bienvenida; la tercera parte contiene una caja de texto donde aparecen mensajes publicados por los funcionarios de los servicios del campus, como se muestra en la Figura 3.7.

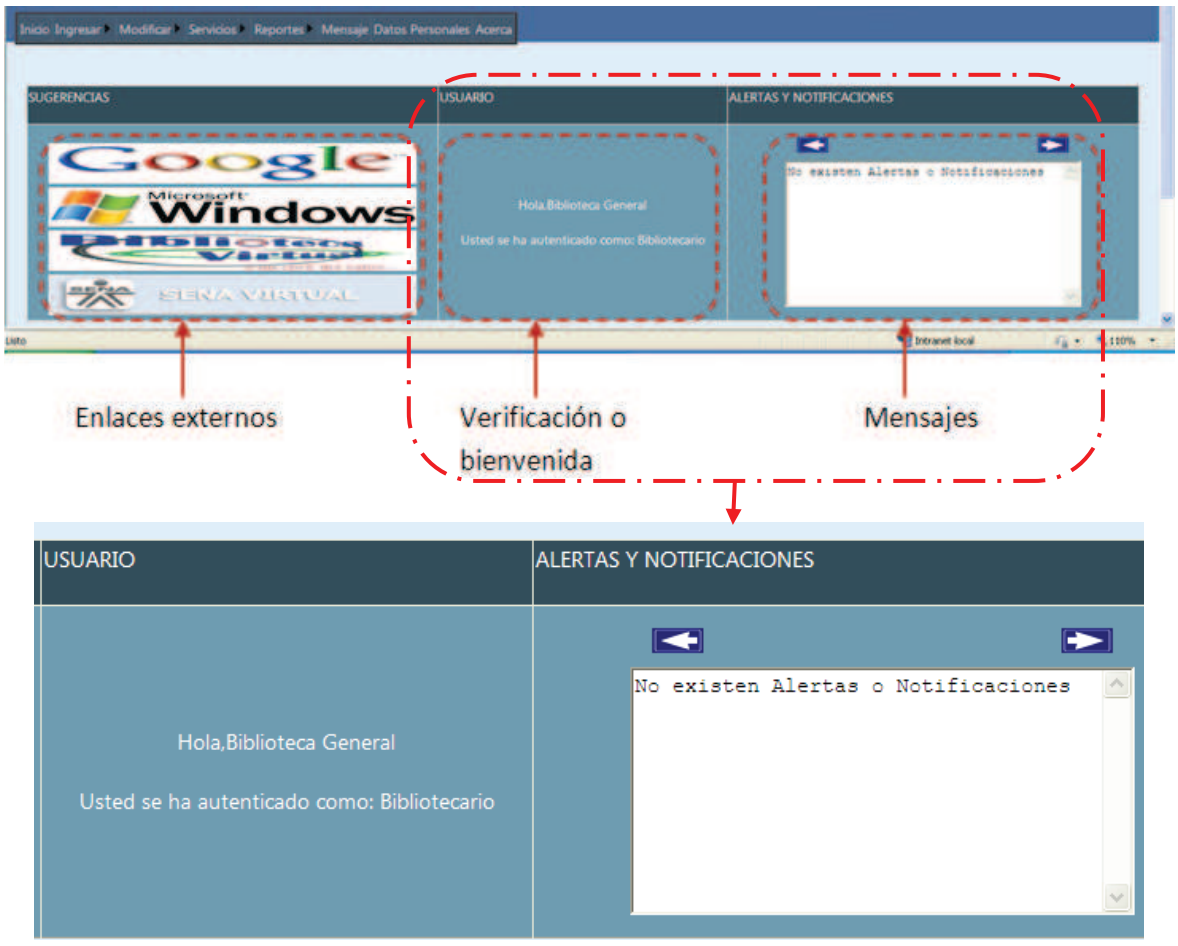

Figura 3.7 Elementos página Menu

Para la publicación de los mensajes se hizo uso de un control *timer*, el cual cambia el contenido y el título del mensaje según el servicio que lo publicó.

3.1.1.6 Página ReporteTransacciones

Presenta una tabla con un listado de las diez últimas transacciones realizadas, con su respectiva fecha, valor y descripción; bajo esta se encuentran dos cajas de texto en las cuales se podrá ingresar un rango de fechas en el cual se puede consultar las transacciones realizadas al presionar el botón *Consultar*. Para el caso de un funcionario Financiero se muestra una caja de texto donde se debe ingresar el código de un usuario del cual se desee conocer sus transacciones.

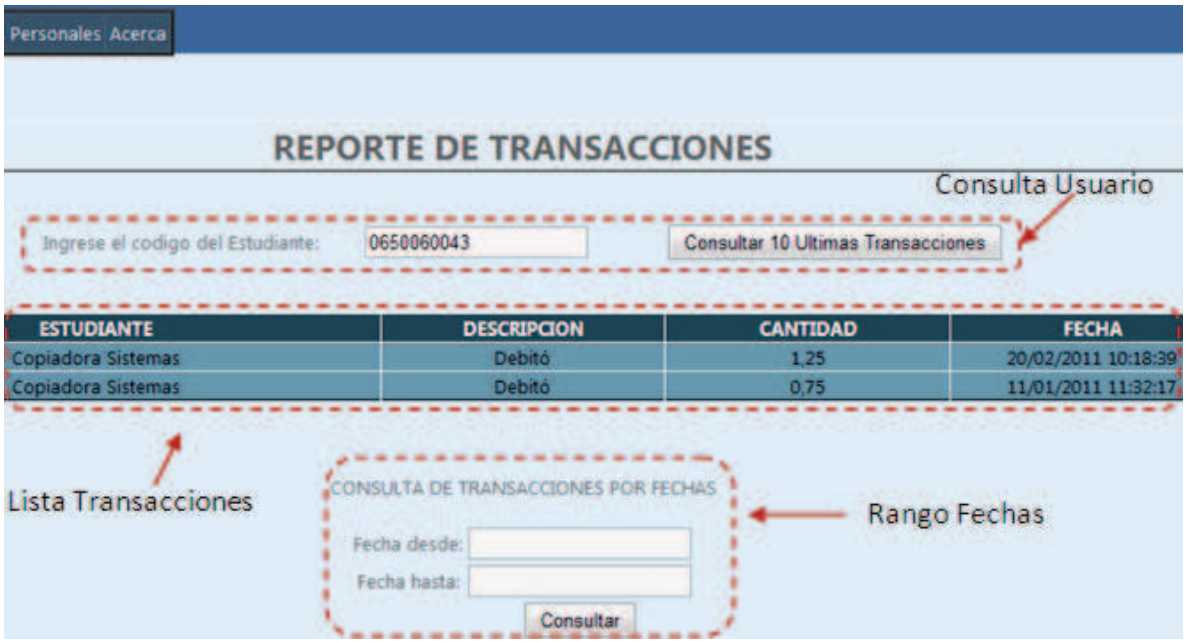

Figura 3.8 Elementos página ReporteTransacciones

3.1.1.7 Página Ingresos

Si se ingresa como Administrador se muestra un conjunto de cajas de texto donde se deben ingresar los datos de un nuevo usuario, y dependiendo del rol que se seleccione se habilitarán otros campos adicionales a ser llenados, siendo éstos:

- Estudiante: Facultad, Carrera y Huella digital.
- Doctor: Especialidad y Consultorio.
- Biblioteca: Edificio.

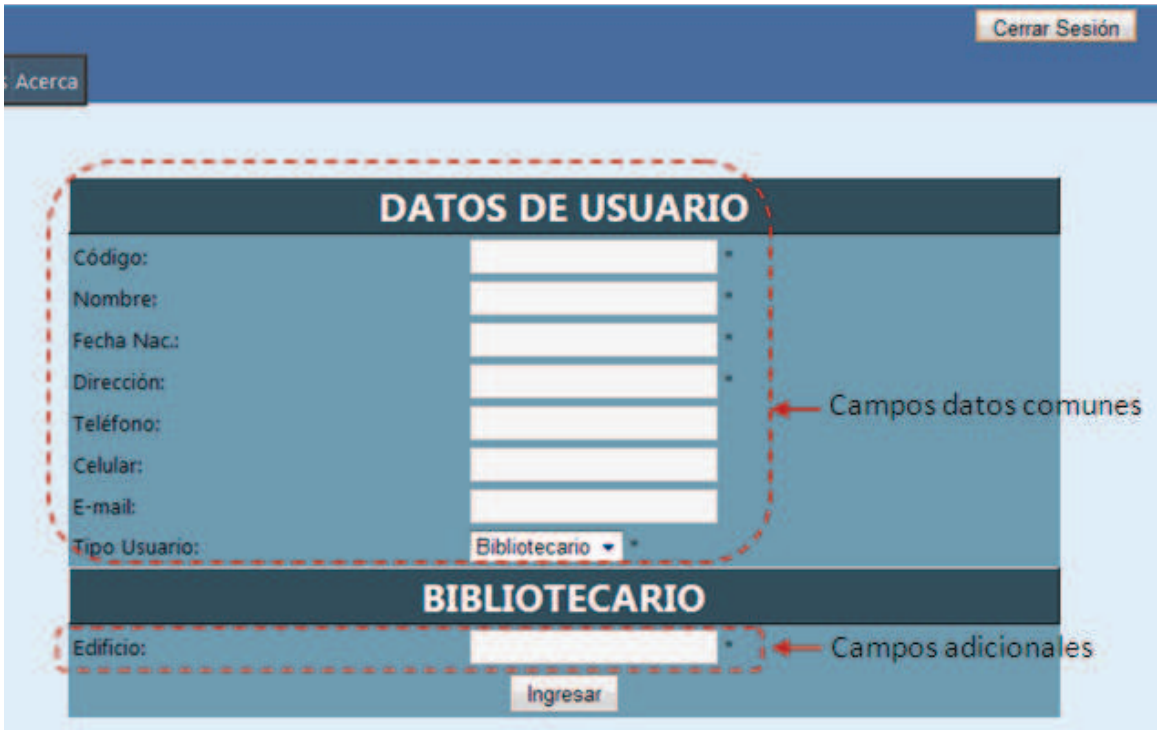

Figura 3.9 Elementos página Ingresos para Administrador

Para un usuario Biblioteca se muestra un conjunto de cajas de texto donde se deben ingresar los datos de un libro, así como también las existencias y la estantería donde se encuentra. La Figura 3.9 muestra la página para un usuario Administrador, y la Figura 3.10 la página para un usuario Biblioteca.

Figura 3.10 Elementos página Ingresos para Biblioteca

3.1.1.8 Página Modificar

Al ingresar como usuario Administrador se muestra primero una caja de texto donde se debe ingresar un código de usuario que devolverá los datos de un usuario existente en el sistema para su modificación. De igual manera para un usuario biblioteca se muestra una caja de texto donde se debe ingresar el título o el autor del libro existente, con lo cual se devolverá los datos del libro para su modificación. La Figura 3.11 muestra la interfaz para un usuario Administrador.

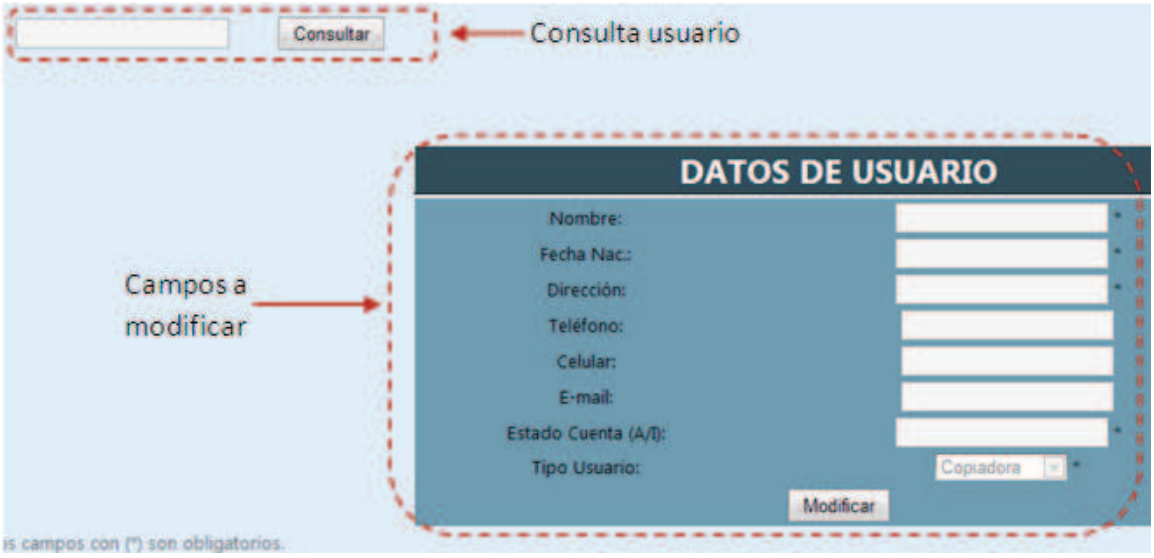

Figura 3.11 Elementos página Modificar

3.1.1.9 Página CatalogoLibros

Si se ingresa como Estudiante permite ingresar el título o el autor de un libro, escoger la biblioteca donde se quiere buscar, y al pulsar el botón *Consultar* se mostrará el resultado de la búsqueda en una tabla con una lista de libros con sus respectivos datos, como se muestra en la Figura 3.12.

Al ingresar como Biblioteca se escoge la búsqueda por título o autor y se ingresa el parámetro que al pulsar el botón *Consultar* retornará el resultado de la búsqueda en una tabla de la cual se puede seleccionar un libro para poder realizar un préstamo y un panel donde se verificará mediante el código de usuario y huella digital la identificación de un Estudiante, también se debe ingresar la fecha de devolución del libro. Figura 3.13

Figura 3.12 Elementos página Catalogo Libros para Estudiante

Figura 3.13 Página CatalogoLibros para Biblioteca

3.1.1.10 Página CobroXItems

En esta página, al ingresar como Bar o Copiadora se presenta una caja de texto para el ingreso del código de un estudiante, y luego un panel que contiene la cantidad, descripción y precio unitario de un ítem que al presionar el botón *Agregar Item* se añadirá a una lista de ítems; al pulsar el botón *Consultar Saldo* se consultará si el valor total de la lista de ítems es menor al saldo de la cuenta del estudiante, de serlo se verificará la huella del estudiante para realizar la transacción. El botón *Borrar* Factura permite eliminar la factura de no haber saldo suficiente. La Figura 3.14 muestra la página para un funcionario Bar o Copiadora.

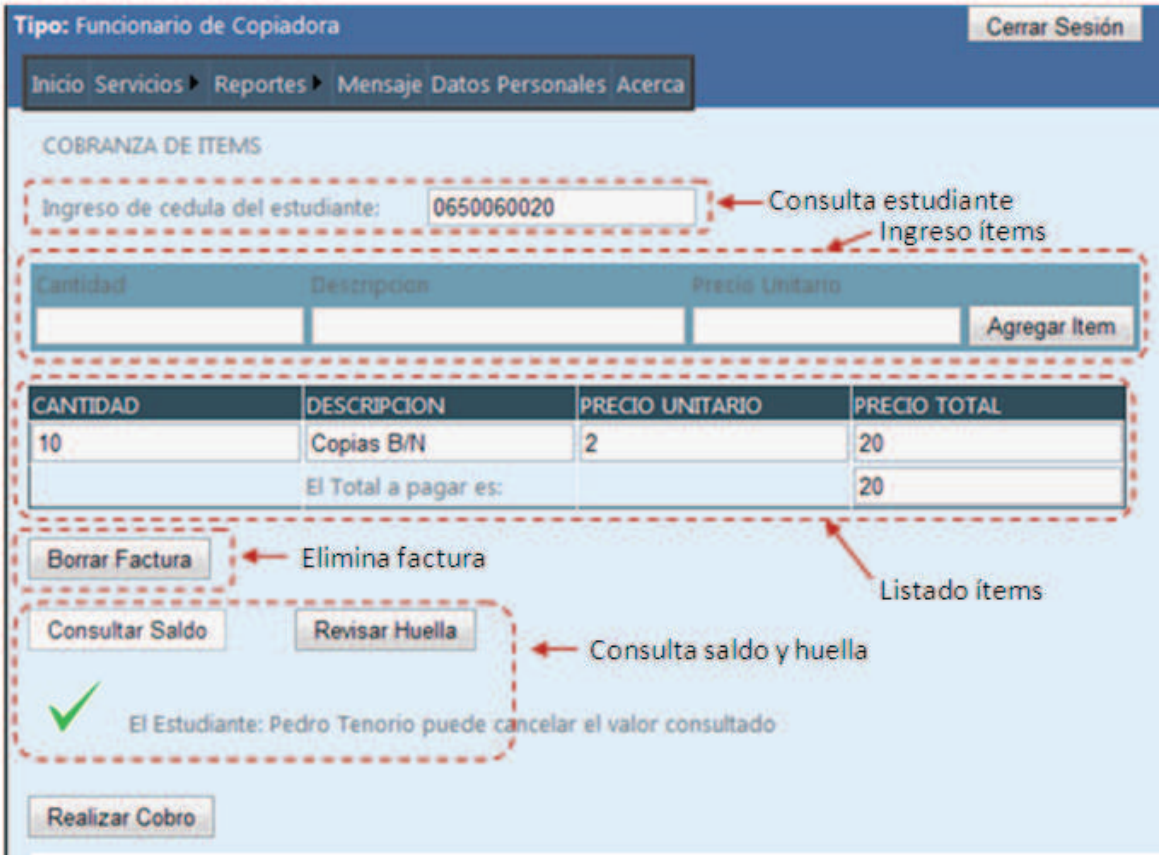

Figura 3.14 Elementos página CobroXItems para Bar o Copiadora

Al ingresar como bibliotecario se presenta una caja de texto para el ingreso del código de un estudiante, otra caja de texto para el ingreso de un valor de multa por día de retraso en la devolución del libro, que al pulsar el botón *Aceptar* se presentará una lista con los libros prestados al estudiante y un valor total de multas por retrasos; al pulsar el botón *Consultar Saldo* se seguirá el mismo procedimiento anteriormente descrito. Posee botones para cancelar la multa en

efectivo, adicionalmente botones al lado de cada multa para la cancelación individual para cada libro, o en la parte inferior para la cancelación total de las multas sea en efectivo o con débito de la cuenta del estudiante (Figura 3.15).

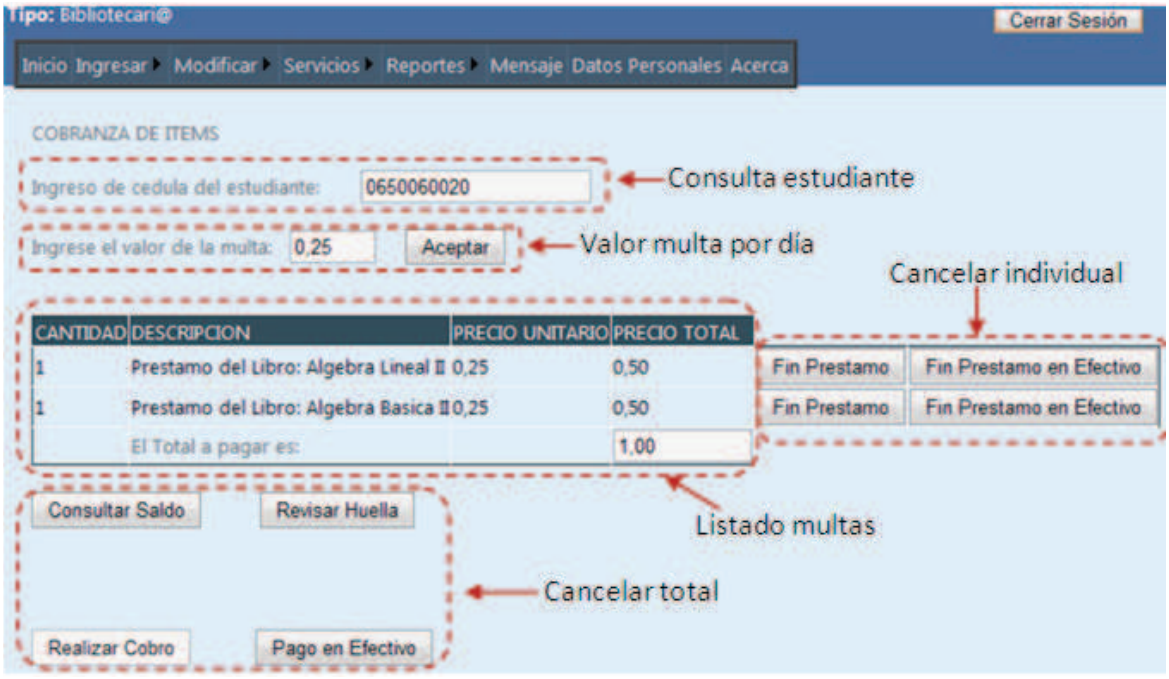

Figura 3.15 Elementos página CobroXItems para Biblioteca

3.1.1.11 Página TransaccionesFinanciero

Contiene una caja de texto donde se debe ingresar el código del estudiante, que al pulsar el botón *Consultar* nos muestra el nombre del usuario y su saldo; además bajo esto se muestra un acordeón de dos hojas, en las cuales se puede hacer depósitos y retiros; cada una de ellas contiene una caja de texto en donde se debe ingresar el valor de la transacción, y un botón que aceptará la misma. Ver Figura 3.16. Para la implementación del acordeón se utilizó *Ajax Control Toolkit*.

3.1.1.12 Página HistoriaClinica

En la parte superior se muestra un listado plegable de citas con el nombre y el horario de atención de un estudiante; luego se tiene una caja de texto para el ingreso del código del estudiante, al pulsar el botón consultar y autenticar al usuario se habilitará el botón *Revisar Huella* para la verificación del mismo. Ver Figura 3.17. El listado plegable se realizó con *Ajax Control Toolkit*.

Figura 3.16 Página TransaccionesFinanciero

Figura 3.17 Página HistoriaClinica

En la parte media se muestra un panel con los datos de la historia clínica del estudiante los cuales podrán ser modificados por el doctor.

En la parte inferior se muestra un listado de citas previas del estudiante y la cita actual para ser modificada por el doctor en los campos *Diagnóstico* y *Tratamiento*.

3.1.1.13 Página Mensajes

Si se ingresa como Doctor se tendrá una caja de texto donde se ingresa la fecha de consulta de citas, al presionar el botón *Consultar Citas* se mostrará un listado de las citas correspondientes a dicha fecha con botones para la cancelación parcial de las citas y un botón al final del listado para la cancelación total.

En la parte inferior se tiene una caja multilineal de texto, en la que se puede ingresar un mensaje de aviso a los estudiantes, una caja de texto donde se ingresa una fecha de vigencia del mensaje y un botón que guarda el mensaje.

Si se ingresa como otro tipo de funcionario solo se muestra la parte inferior de la página previamente explicada. La Figura 3.18 muestra la interfaz para Doctor.

Figura 3.18 Página Mensajes

3.1.1.14 Página CitasMedicas

Esta página mostrada en la Figura 3.19 contiene un listado de las especialidades médicas en las cuales el estudiante puede reservar una cita médica, una caja de texto donde se puede escoger la fecha para reservar una cita y una tabla con los posibles horarios, además una vez reservada la cita no se puede hacer otra cita en la misma especialidad hasta cumplida la fecha; también se la puede cancelar.

Figura 3.19 Página CitasMedicas

3.1.2 CAPTURA DE HUELLAS DIGITALES

El lector de huellas hizo uso del *SecuBSP SDK Pro for Windows* proporcionado por los fabricantes del dispositivo. Cabe mencionar que para la utilización del lector, todos los funcionarios que lo ocupen deberán tener instalado dicho SDK en sus terminales. El Espacio de código 3.1 muestra la función utilizada para la captura de huellas digitales.

```
//Funcion que captura la Huella del Estudiante
        function Capturar() {
            var err

            try // Exception handling
                  {
                // Abrir Dispositivo. [AUTO_DETECT]
                // Debes abrir el dispositivo antes de capturar la huella.
                DEVICE_FDP02 = 1;
                DEVICE_FDU02 = 2;
                DEVICE_FDU03 = 3;
                DEVICE_FDU04 = 4;

                DEVICE_AUTO_DETECT = 255;
                document.objSecuBSP.OpenDevice(DEVICE_AUTO_DETECT);
                err = document.objSecuBSP.ErrorCode; // Get error code

                if (err != 0)           // Fallo la inicializacion del dispositivo
                {
                    alert('No se pudo Inicializar el dispoditivo !');
                    return;
                }

                // Incripcion de la huella de usuario.
                document.objSecuBSP.Capture();
                err = document.objSecuBSP.ErrorCode; // obtiene el codigo de error

                if (err != 0)           // Incripcion Fallida
                {
                    alert('Fallo la Captura ! Error Numero : [' + err + ']');
                    return;
                }
                else// Captura exitosa
                {
                    // Obtiene texto codificado FIR data desde el modulo SecuBSP.
                    document.getElementById('<%=  hfiHuella.ClientID  %>').value  =
document.objSecuBSP.FIRTextData;
                    alert('Captura exitosa !');
                }

                // Cerrar dispositivo. [AUTO_DETECT]
                document.objSecuBSP.CloseDevice(DEVICE_AUTO_DETECT);

            }
            catch (e) {
                alert(e.message);
            }

            return;
        }
```

Espacio de Código 3.1 Función Java Script que captura la Huella Dactilar

Todas las operaciones realizadas con el lector de huellas digitales fueron desarrolladas en la capa de Presentación. Debido a que el código para la captura de huella necesita un objeto del SDK, no se pudo tener un archivo *.js* externo, por

lo cual, se tuvo que repetir el código de captura y verificación de huella en todas las páginas que lo requerían.

```
function fnVerificar(huella1, huella2, intentos) {

            var err
            var str1 = huella1;
            var str2 = huella2;

            try // Exception handling
                  {
                // Verificacion huella digital.
                document.objSecuBSP.VerifyMatch(str1, str2);
                err = document.objSecuBSP.ErrorCode;

                if (err != 0) {
                    alert('Error de Verificacion  ! Error Numero : [' + err + ']');
                }
                // Verificacion fallida.
                else {
                    if (document.objSecuBSP.IsMatched == 0) {
                        alert('Verificacion fallida !');
                        // Cuenta 3 intentos.
                        if (intentos == '') {
                            document.getElementById('<%= hfiIntentos.ClientID
%>').value = 'No valido1';
                        }
                        if (intentos == 'No valido1') {
                            document.getElementById('<%= hfiIntentos.ClientID
%>').value = 'No valido2';
                        }
                        if (intentos == 'No valido2') {
                            document.getElementById('<%= hfiIntentos.ClientID
%>').value = 'Bloqueado';
                        }
                    }
                    // Verificacion exitosa.
                    else {
                        alert('Verificacion exitosa !');
                        document.getElementById('<%= hfiIntentos.ClientID %>').value
= 'Huella Valida';
                    }
                }
            }
            catch (e) {
                alert(e.message);
                document.getElementById('<%= hfiIntentos.ClientID %>').value = 'No
valido';
            }

            return;
        }
```

Espacio de Código 3.2 Función Java Script de Verificación de Huella Dactilar y Bloqueo de Cuenta

Para la captura de la huella se utilizó una función en javaScript en donde primero se verifica y se inicializa al dispositivo viendo que no existan errores, luego se

procede a la captura de la huella y se la almacena en un control *HiddenField* de *ASP.Net* el cual oculta la matriz obtenida de la huella para ser enviada a la base de datos, luego de lo cual se llama a la función *CloseDevice* que expone el objeto SDK para cerrar el dispositivo.

3.1.3 VERIFICACIÓN DE HUELLAS

En la verificación de huellas, primero se debe realizar el proceso de captura de huella descrito anteriormente para tener almacenado la huella que se quiere verificar, además se debe consultar en la base de datos la huella que se almacenó previamente y depositarla en otro control *HiddenField*.

Por último se ejecuta la función de verificación que toma los contenidos de los dos controles *HiddenField* y se los comprueba mediante el método expuesto del objeto SDK *VerifyMatch,* el cual retorna un valor que permite saber si la huella fue válida, de lo contrario se tiene dos oportunidades más de comprobación, luego de lo cual si no se obtuvo un resultado positivo la cuenta quedará bloqueada. El Espacio de Código 3.2 muestra la función usada para la verificación de huellas digitales.

3.2 CAPA DE NEGOCIOS

Esta capa está conformada por los servicios web. Igual que en el caso anterior, se tiene un servicio común para todos los usuarios, y otros que son dependientes del tipo de usuario. Primero se darán ejemplos del código utilizado, y posteriormente se detallarán los métodos de cada servicio web.

3.2.1 CÓDIGO DE IMPLEMENTACIÓN

En esta sección se explicará los códigos que fueron relevantes a la hora de desarrollar la capa de negocios. En primer lugar tiene el código necesario para llamar un servicio Web desde el sitio Web, mostrado en el Espacio de Código 3.3.

```
ServicioWebDoctor.MultiServicioMedico servicioM = new
ServicioWebDoctor.MultiServicioMedico();
```

Espacio de Código 3.3 Llamado a un servicio desde sitio web

Esta línea de código se la puede utilizar solamente después de haber creado la referencia hacia el servicio solicitado, en este caso el ServicioWebDoctor.

En el Espacio de Código 3.4 se ve cómo se almacena en una variable de tipo string la Cadena de conexión que servirá para llamar a una base de datos. Esta Cadena de conexión está almacenada en un parámetro en el webConfig del Servicio Web.

```
string CadenaDeConexion =
ConfigurationManager.ConnectionStrings["DatosMultiservicios"].ConnectionString;
SqlConnection sqlConn;
```

Espacio de Código 3.4 Llamada a cadena de conexión

El siguiente es un ejemplo de cómo se llama a un proceso almacenado desde el servicio web para el ingreso de información en la base de datos. En este caso describe el procedimiento para almacenar un objeto biblioteca.

```
public void IngresarBiblioteca(string edificio, string idUsuario)
        {
            try
            {
                sqlConn = new SqlConnection();
                sqlConn.ConnectionString = CadenaDeConexion;
                SqlCommand cmd = new SqlCommand("spms_ingresar_biblioteca",
sqlConn);
                cmd.CommandType = CommandType.StoredProcedure;
                SqlParameter edif = cmd.Parameters.Add("@i_edificio",
SqlDbType.VarChar);
                edif.Direction = ParameterDirection.Input;
                edif.Value = edificio;
                SqlParameter numci = cmd.Parameters.Add("@i_idUsuario",
SqlDbType.Char);
                numci.Direction = ParameterDirection.Input;
                numci.Value = idUsuario;
                SqlDataReader reader;
                sqlConn.Open();
                reader = cmd.ExecuteReader();
                sqlConn.Close();
            }
            catch (Exception ex)
            {
            }
        }
```

Espacio de Código 3.5 Código para ingreso de objeto biblioteca

El método mostrado en el Espacio de Código 3.5 consta de dos parámetros de entrada de tipo string, en el primero se especifica el nombre del edificio en donde

se encuentra la biblioteca y el segundo parámetro solicita el Identificador del Usuario que va a dar atención en la biblioteca mencionada.

En primer lugar se crea la conexión a la base de datos, luego se especifica un comando a ejecutarse en la base de datos en el cual se ingresa como parámetro el nombre del proceso almacenado a invocar. Posteriormente se especifican los parámetros necesarios para la ejecución del comando. El objeto *reader* de tipo *SqlDataReader* va a realizar la ejecución del comando una vez abierta la conexión, y ya que esta función es de inserción no devolverá valor alguno. Por último se cierra la conexión.

```
[WebMethod]

public List<object> ConsultarSaldo(string idUsuario, decimal monto)
{
       List<object> lista = new List<object>();
       try
       {
              sqlConn = new SqlConnection();
              sqlConn.ConnectionString = CadenaDeConexion;
              SqlCommand cmd = new SqlCommand("spms_consultar_saldo", sqlConn);
              cmd.CommandType = CommandType.StoredProcedure;
              SqlParameter usuario = cmd.Parameters.Add("@i_idUsuario",
SqlDbType.Char);
              usuario.Direction = ParameterDirection.Input;
              usuario.Value = idUsuario;
              SqlDataReader reader;
              sqlConn.Open();
              reader = cmd.ExecuteReader();
              while (reader.Read())
              {
                     lista.Add(reader[0]);   //el registro de huella
                     if (monto <= Convert.ToDecimal(reader[1])) //saldo de la cuenta
                         lista.Add("verdadero"); //el saldo es suficiente
                     else
                         lista.Add("falso");  //el saldo no es suficiente
                     lista.Add(reader[2]);   //el nombre del estudiante
                     lista.Add(reader[3]);   //el estado de la cuenta
             }
             sqlConn.Close();
             return lista;
       }
       catch (Exception e)
       {
              return lista;
       }
}
```

Espacio de Código 3.6 Código para consulta de datos desde la base

En el Espacio de Código 3.6 se muestra un ejemplo de consulta a la base de datos, en este caso, la consulta de saldo de un estudiante. Este método recibe un parámetro de tipo string que contiene el identificador del usuario, y otro parámetro de tipo decimal que contiene el monto a debitar de la cuenta del usuario; y devuelve una variable genérica de tipo lista de objetos que contendrá la huella del estudiante, un indicador de saldo suficiente, el nombre del estudiante y el estado de cuenta. Cuando el monto a debitar es menor o igual al que contiene la cuenta, el indicador de saldo suficiente tendrá un valor verdadero, caso contrario tendrá un valor falso.

3.2.2 DETALLES DE ServicioComun

En este servicio se encuentran métodos de uso común para todos lo usuario, éstos son los mostrados en la Tabla 3.1.

Función	Variables de entrada	Tipo de Salida	Descripción
ConsultaDatosPersonales	Cedula	List	Consulta los datos personales de un usuario.
ConsultaUltimasTransacciones	idUsuario	List<List<object>>	Consulta las últimas 10 transacciones de un usuario.
ConsultaTransaccionesXFecha	idUsuario fechadesde fechahasta	List<List<object>>	Consulta las transacciones en un rango de fechas.
AutenticarUsuario	cedula contraseña	List	Autentica al usuario en el sistema y devuelve su nombre, tipo de usuario y estado de cuenta.
ActualizarUsuario	idUsuario contrasena estadoC nombre fechaN direccion telefono celular email tipoUsu	int	Actualiza los campos de un objeto usuario.

Tabla 3.1 Funciones de ServicioComun

3.2.3 DETALLES DE ServicioAdministrador

Contiene métodos que permiten gestionar la información de los usuarios mostrados en la Tabla 3.2.

Función	Variables de entrada	Tipo de Salida	Descripción
RegistroHuella	idUsuario regHuella	Int	Registra la huella dactilar de un Estudiante.
IngresarUsuario	idUsuario contrasena estadoC nombre fechaN direccion telefono celular email tipoUsu	void	Ingresa todos los datos de un Usuario Nuevo.
IngresarBiblioteca	edificio idUsuario	void	Ingresa los campos de un objeto Biblioteca.
ActualizarBiblioteca	edificio idUsuario	void	Actualiza los campos de un objeto Biblioteca.
IngresarDoctor	idUsuario especialidad consultorio	void	Ingresa los campos de un objeto Doctor.
IngresarEstudiante	idUsuario facultad carrera regHuella1	void	Ingresa los campos de un objeto Estudiante.
ActualizarEstudiante	idUsuario facultad carrera	void	Actualiza los campos de un objeto Estudiante.

Tabla 3.2 Funciones de ServicioAdministrador

3.2.4 DETALLES DE ServicioBiblioteca

Sus métodos permiten gestionar la información de libros, préstamos y pagos de multas en una biblioteca como se muestra en la Tabla 3.3.

3.2.5 DETALLES DE ServicioDoctor

Contiene métodos que permiten gestionar la información de citas e historia clínica. Ver Tablas 3.4.

3.2.6 DETALLE ServicioEstudiantil

Este método solamente contiene un método descrito en la Tabla 3.5 que devuelve la información de un estudiante.

Función	Variables de entrada	Tipo de Salida	Descripción
ConsultaCatalogoLibros	Cedula titulo autor	List<List<object>>	Consulta el catalogo de libros de una biblioteca.
PagoEfectivo	idUsuario idLibro	Bool	Realiza el regreso de un libro en efectivo.
PrestarLibros	cedula codigo fecha	int	Realiza el préstamo de un libro.
ConsultaMultaLibros	idUsuario valorMulta	List<string>	Consulta los libros prestados a un estudiante y el valor de las multas de los mismos.
ConsultaBibliotecas		List<List<object>>	Enlista las bibliotecas existentes.
IngresarLibro	titulo autor edicion anio idBiblioteca cantidad estantería	void	Ingresa los campos de un objeto Libro.
ActualizarLibro	idLibro titulo autor edicion anio idBiblioteca cantidad estanteria	void	Actualiza los campos de un objeto Libro.
IngresarExistencia	idBiblioteca idLibro cantidad estanteria	void	Ingresa los campos de un objeto Existencia.

Tabla 3.3 Funciones de ServicioBiblioteca.

Función	Variables de entrada	Tipo de Salida	Descripción
ConsultarDoctor		List<List<object>>	Enlista objs. Doctor.
ConsultarCitasDiarias	fecha fechaS doctor cedula	List<List<object>>	Consulta las citas de la fecha actual.
InsertarCitaMedica	fecha diagnostico tratamiento doctor estudiante	bool	Ingresa una cita médica.
CancelarCitaMedica	fechaA fechaS doctor cedula	bool	Cancela una cita hecha.
ConsultarHistoriaClinica	cedula	List<object>	Consulta los campos de un objeto Historia Clínica.
ConsultarCitasMedicas	cedula Doctor	List<List<object>>	Consulta las citas actuales de un doctor.
ActualizarHistoriaClinica	historia tipoSangre alergias observaciones	void	Actualiza los campos del objeto Historia Clínica.
ActualizarCitaMedica	historia fecha diagnostico tratamiento	void	Actualiza la cita médica (exclusivo de Doctor).
ConsultarCitasDoctor	fechaA fechaS doctor	List<List<object>>	Consulta las citas de Doctor en una fecha elegida.
ConsultarCitasAnteriores	fechaA usuario doctor	string	Consulta las citas históricas con un determinado doctor.
CancelarCitasDoctor	doctor codigo fecha fechas	bool	Cancela las citas de una fecha elegida, total o parcialmente (exclusivo de Doctor).

Tabla 3.4 Funciones de ServicioDoctor

Función	Variables de entrada	Tipo de Salida	Descripción
ConsultarCodigo	cedula	List<string>	Consulta el identificador, la facultad y la carrera de un Estudiante.

Tabla 3.5 Función de ServicioEstudiantil

3.2.7 DETALLE ServicioFinanciero

La Tabla 3.6 detalla los métodos que permiten gestionar los depósitos y retiros de una cuenta así como la consulta del nombre del titular y el saldo restante.

Función	Variables de entrada	Tipo de Salida	Descripción
ConsultarNombreySaldo	cedula	List<object>	Consulta el nombre y el saldo de un usuario.
IngresarDeposito	cedula monto	Void	Ingresa un depósito.
IngresarRetiro	cedula monto	Void	Ingresa un retiro.

Tabla 3.6 Funciones de ServicioFinanciero

3.2.8 DETALLES DE ServicioFuncionarios

Contiene métodos de uso general de los funcionarios de los servicios. Estos métodos están detallados en la Tabla 3.7.

Función	Variables de entrada	Tipo de Salida	Descripción
InsertarTransaccion	usuarioD usuarioA monto	bool	Inserta una transacción.
ConsultarSaldo	idUsuario monto	List<object>	Consulta el nombre, la huella y el saldo de un usuario.
Consul3.14tarMensajes		List<List<object>>	Consulta los mensajes que se publican por los funcionarios.
IngresarMensaje	idUsuario descripcion fecha	Bool	Ingresa un mensaje de notificación para los estudiantes.

Tabla 3.7 Funciones de ServicioFuncionarios

3.3 CAPA DE DATOS

Para realizar las todas las operaciones sobre la base de datos se utilizan una gran cantidad de procedimientos almacenados (sps), detallados en las Tablas 3.8 a 3.16.

3.3.1 PROCEDIMIENTOS COMUNES

Nombre sps	Variables de entrada	Tipo de Variable	Descripción
spms_autenticar_usuario	usuario contrasena	char(10) varchar(16)	Autentica a un usuario que ingresa al sistema.
spms_actualizar_usuario	idUsuario contrasena estado_cuenta nombre fechaNacimiento direccion telfConvencional celular email tipoUsuario	varchar(10) varchar(20) char(1) varchar(100) varchar(10) varchar(100) varchar(10) varchar(10) varchar(50) char(1)	Actualiza un usuario con datos ingresados
spms_consultar_datos_personales	usuario	char(10)	Consulta los datos personales de un usuario.
spms_consultar_ultimas_10_transacciones	idUsuario	char(10)	Consulta las 10 últimas transacciones realizadas por un usuario.
spms_consultar_transacciones_x_fecha	idUsuario fecha_desde fecha_hasta	char(10) varchar(10) varchar(10)	Consulta transacciones realizadas en un intervalo de tiempo.

Tabla 3.8 Descripción de procedimientos almacenados comunes

3.3.2 PROCEDIMIENTOS DE BIBLIOTECA

Nombre sps	Variables de entrada	Tipo de Variable	Descripción
spms_consultar_bibliotecas			Consulta las bibliotecas existentes.
spms_consultar_catalogo_libros	idUsuario Titulo Autor	char(10) varchar(50) varchar(50)	Consulta el catálogo de libros de una biblioteca por Título o Autor.
spms_consulta_multa_libros	idUsuario	char(10)	Consulta los datos personales de un usuario.
spms_pago_efectivo_multa	idUsuario idLibro	char(10) varchar(6)	Termina el proceso de préstamo del libro
spms_consultar_ultimas_10_transacciones	idUsuario	char(10)	Consulta las 10 últimas transacciones realizadas por un usuario.

Tabla 3.9 Detalle de procedimientos almacenados para Biblioteca. Parte I

Nombre sps	Variables de entrada	Tipo de Variable	Descripción
spms_consultar_transacciones_x_fecha	idUsuario fecha_desde fecha_hasta	char(10) varchar(10) varchar(10)	Consulta si el estudiante tiene multas de libros en la fecha indicada.
spms_prestar_libros	idUsuario codigo fecha_fin	char(10) char(6) date	Realiza el préstamo de un libro a un estudiante.
spms_ingresar_existencia	idBiblioteca idLibro cantidad estanteria	varchar(3) varchar(6) int varchar(3)	Ingresa una existencia nueva o la modifica.
spms_ingresar_libro	titulo autor edicion anio idBiblioteca cantidad estanteria	varchar(50) varchar(50) varchar(3) varchar(4) varchar(3) int varchar(3)	Ingresa un libro nuevo o lo modifica.
spms_actualizar_libro	idLibro titulo autor edicion anio idBiblioteca cantidad estanteria	varchar(6) varchar(50) varchar(50) varchar(3) varchar(4) varchar(3) int varchar(3)	Modifica un ítem libro existente.

Tabla 3.10 Detalle de procedimientos almacenados para Biblioteca. Parte II

3.3.3 PROCEDIMIENTOS DE FUNCIONARIOS

Nombre sps	Variables de entrada	Tipo de Variable	Descripción
spms_consultar_saldo	idUsuario	char(10)	Consulta el saldo de un estudiante.
spms_ingresar_transaccion	idUsuarioD idUsuarioA monto	char(10) char(10) decimal(7,2)	Ingresa una transacción entre dos usuarios.
spms_ingresar_mensaje	idUsuario descripcion fecha	varchar(10) varchar(300) varchar(8)	Ingresa un mensaje para los estudiantes.
spms_consultar_mensajes			Consulta los mensajes ingresados.

Tabla 3.11 Detalle de procedimientos almacenados para Funcionarios

3.3.4 PROCEDIMIENTOS DE SERVICIO MÉDICO

Nombre sps	Variables de entrada	Tipo de Variable	Descripción
spms_consultar_doctores			Consulta los doctores y sus especialidades.
spms_consultar_citas_diarias	fechaA fechaS idDoctor idUsuario	varchar(10) varchar(10) varchar(4) varchar(10)	Consulta las citas diarias de una especialidad.
spms_ingresar_cita_medica	fecha diagnostico tratamiento idDoctor idEstudiante	varchar(20) varchar(150) varchar(150) varchar(4) varchar(6)	Ingresa una cita médica.
spms_cancelar_cita	fechaA fechaS idDoctor idUsuario	varchar(8) varchar(8) varchar(4) varchar(10)	Cancela una cita médica previamente hecha.
spms_consultar_citas_medicas	idUsuario idDoctor	varchar(10) varchar(6)	Consulta las últimas citas hechas por un estudiante en una especialidad.
spms_consultar_historia_clinica	idUsuario	varchar(10)	Consulta los datos de historia clínica de un estudiante.
spms_actualizar_historia_clinica	historia tipo_Sangre alergias observaciones	varchar(6) varchar(2) varchar(50) varchar(100)	Actualiza los principales datos de la Historia Clínica.
spms_actualizar_cita_medica	historia fecha diagnostico tratamiento	varchar(6) varchar(25) varchar(150) varchar(150)	Actualiza los parámetros de la cita médica actual.
spms_consultar_citas_doctor	fechaA fechaS idDoctor	varchar(16) varchar(16) varchar(4)	Consulta los pacientes que va a atender en la fecha especificada.
spms_consultar_citas_anteriores	idUsuario idDoctor fecha	varchar(10) varchar(6) varchar(15)	Consulta citas anteriores del estudiante sin ser atendidas aun.
spms_cancelar_citas_doctor	idDoctor idEstudiante fecha fechaS	varchar(4) varchar(6) varchar(8) varchar(8)	Cancela parcial o totalmente las citas médicas.

Tabla 3.12 Detalle de procedimientos almacenados para Servicio Médico

3.3.5 PROCEDIMIENTOS DE LA INSTITUCIÓN FINANCIERA

Nombre sps	Variables de entrada	Tipo de Variable	Descripción
spms_consultar_nombre_saldo	idUsuario	char(10)	Consulta el nombre y el saldo de un usuario.
spms_ingresar_retiro	idUsuario retiro	varchar(10) decimal(7,2)	Ingresa una transacción de retiro desde la cuenta de un usuario.
spms_ingresar_deposito	idUsuario deposito	varchar(10) decimal(7,2)	Ingresa una transacción de depósito hacia la cuenta de un Usuario.

Tabla 3.13 Detalle de procedimientos almacenados para Institución Financiera

3.3.6 PROCEDIMIENTOS DEL ADMINISTRADOR

Nombre sps	Variables de entrada	Tipo de Variable	Descripción
spms_ingresar_usuario	idUsuario contraseña estado_cuenta nombre	varchar(10) varchar(20) char(1) varchar(100)	Ingresa un nuevo usuario con sus datos más relevantes.
spms_ingresar_usuario	fechaNacimiento direccion telfConvencional celular email tipoUsuario	varchar(10) varchar(100) varchar(10) varchar(10) varchar(50) char(1)	
spms_ingresar_biblioteca	edificio idUsuario	varchar(25) varchar(10)	Ingresa una biblioteca.
spms_ingresar_doctor	idUsuario especialidad consultorio	varchar(10) varchar(25) varchar(6)	Ingresa un nuevo Doctor.
spms_ingresar_historiaClinica	idEstudiante tipoSangre alergias observaciones	varchar(6) varchar(2) varchar(50) varchar(100)	Ingresa una nueva historia clínica.
spms_ingresar_estudiante	idUsuario facultad carrera huelladigitalr1	varchar(10) varchar(100) varchar(60) varchar(3000)	Ingresa un nuevo estudiante.
spms_actualizar_estudiante	idUsuario i_facultad carrera	varchar(10) varchar(100) varchar(60)	Modifica los datos de una estudiante existente.
spms_actualizar_biblioteca	edificio idUsuario	varchar(25) varchar(10)	Modifica parámetros de una biblioteca.
spms_actualizar_huella_estudiante	idUsuario huelladigitalr1	varchar(10) varchar(500)	Modifica la huella de un Estudiante.

Tabla 3.14 Detalle de procedimientos almacenados para Administrador. Parte I

Nombre sps	Variables de entrada	Tipo de Variable	Descripción
spms_actualizar_existencia	idBiblioteca idLibro cantidad estanteria	varchar(3) varchar(6) int varchar(3)	Modifica parámetros de un libro.

Tabla 3.15 Detalle de procedimientos almacenados para Administrador. Parte II

3.3.7 PROCEDIMIENTOS DEL ESTUDIANTE

Nombre sps	Variables de entrada	Tipo de Variable	Descripción
spms_consultar_codigo_estudiante	idUsuario	varchar(10)	Consulta el identificador, facultad y la carrera del estudiante.

Tabla 3.16 Detalle procedimiento almacenado para estudiante

3.4 CONEXIÓN ENTRE CAPAS

3.4.1 CONEXIÓN ENTRE LA CAPA DE DATOS Y DE NEGOCIOS

Para la interconexión entre las capas de datos y de negocios, se necesita crear un usuario con permisos para el uso de la base de datos con su respectivo login y contraseña siguiendo los pasos explicados más abajo en este capítulo, luego de lo cual y mediante el uso de Visual Studio se puede crear la llamada "cadena de conexión" siguiendo los pasos indicados en el *wizard* para la agregación de una conexión de base de datos en el *Server Explorer* (Explorador de servidores), que es la que se ingresa en el archivo web.config de la capa de negocios para su uso en la llamada de las distintas operaciones realizadas sobre la base de datos.

El Espacio de Código 3.7 muestra la cadena de conexión utilizada en este proyecto.

```
<connectionStrings>
      <add name="DatosMultiservicios"
        connectionString="Data Source=FAMOT\SQLEXPRESS;Initial
Catalog=MultiserviciosE;Persist Security Info=True;User ID=Vyper3000;Password=sofia"
           providerName="System.Data.SqlClient"/>
</connectionStrings>
```

Espacio de Código 3.7 Cadena de Conexión

El Espacio de Código 3.8 muestra la llamada de un proceso almacenado mediante el consumo de la cadena de conexión.

```
string CadenaDeConexion =
ConfigurationManager.ConnectionStrings["DatosMultiservicios"].ConnectionString;
        SqlConnection sqlConn = new SqlConnection();
                sqlConn.ConnectionString = CadenaDeConexion;
                SqlCommand cmd = new SqlCommand("spms_consultar_datos_personales",
sqlConn);
```

Espacio de Código 3.8 Llamada a un Proceso Almacenado usando la Cadena de Conexión

3.4.2 CONEXIÓN ENTRE LA CAPA DE PRESENTACIÓN Y DE NEGOCIOS

Para la interconexión entre la capa de presentación y la de negocios, es necesario publicar primero los servicios de la capa de negocios para obtener su dirección URL, la cual será utilizada para referenciar a cada uno de ellos en la capa de presentación; de la siguiente manera:

En el explorador de la solución se dio clic derecho en *References* y luego en *Add Web Reference.* En la ventana que aparece se escribe la dirección URL del servicio y se presionó la tecla *Enter,* con lo cual aparece los métodos expuestos por dicho servicio para su uso en la capa de presentación; en esta pantalla se tiene la opción de ingresar un nombre para el reconocimiento del servicio y un botón que añadirá dicha referencia. Ver Figura 3.20.

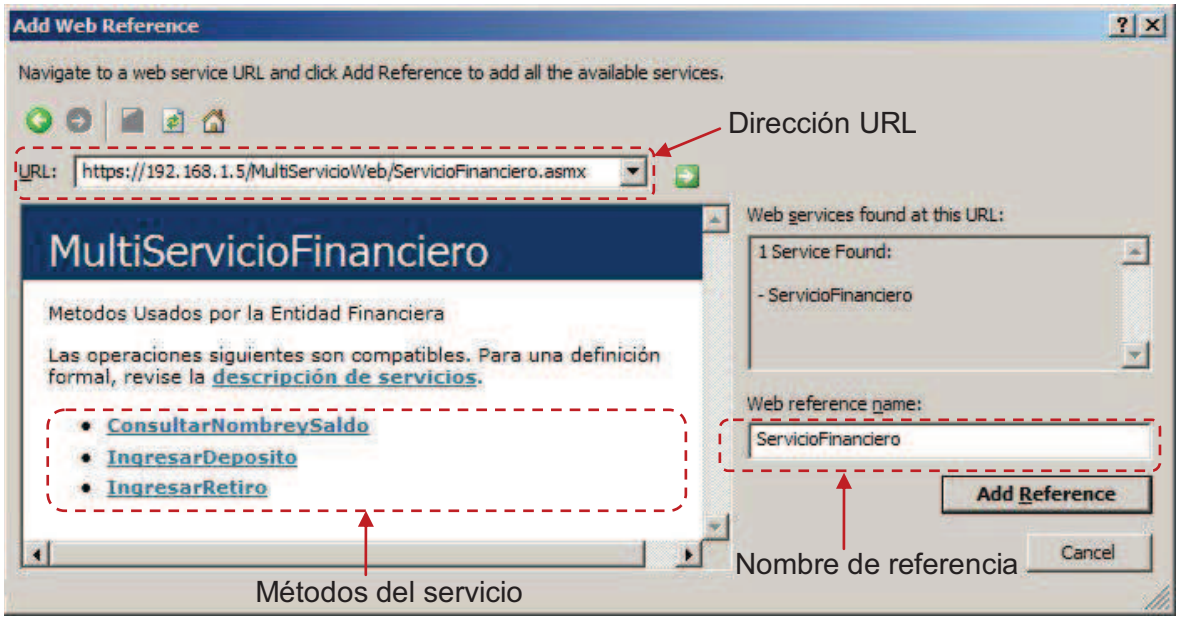

Figura 3.20 Pantalla para añadir referencia web

3.5 CONFIGURACIÓN DE SERVIDORES

3.5.1 SERVIDOR DE BASE DE DATOS

Como se explicó anteriormente, para el gestor de base de datos se utilizó Microsoft SQL Server 2008, el cual tiene la opción de admitir conexiones remotas bloqueadas, por lo que es necesario habilitarla con los siguientes pasos:

- Se abrió el *SQL Server Management Studio*, y se pulsó botón derecho sobre la instancia del Servidor, luego *Properties*, y en la pantalla que apareció se seleccionó *Connections*, donde se marcó el checkbox: "*Allow remote connections to this server"*, finalmente en *OK.* Ver Figura 3.21.

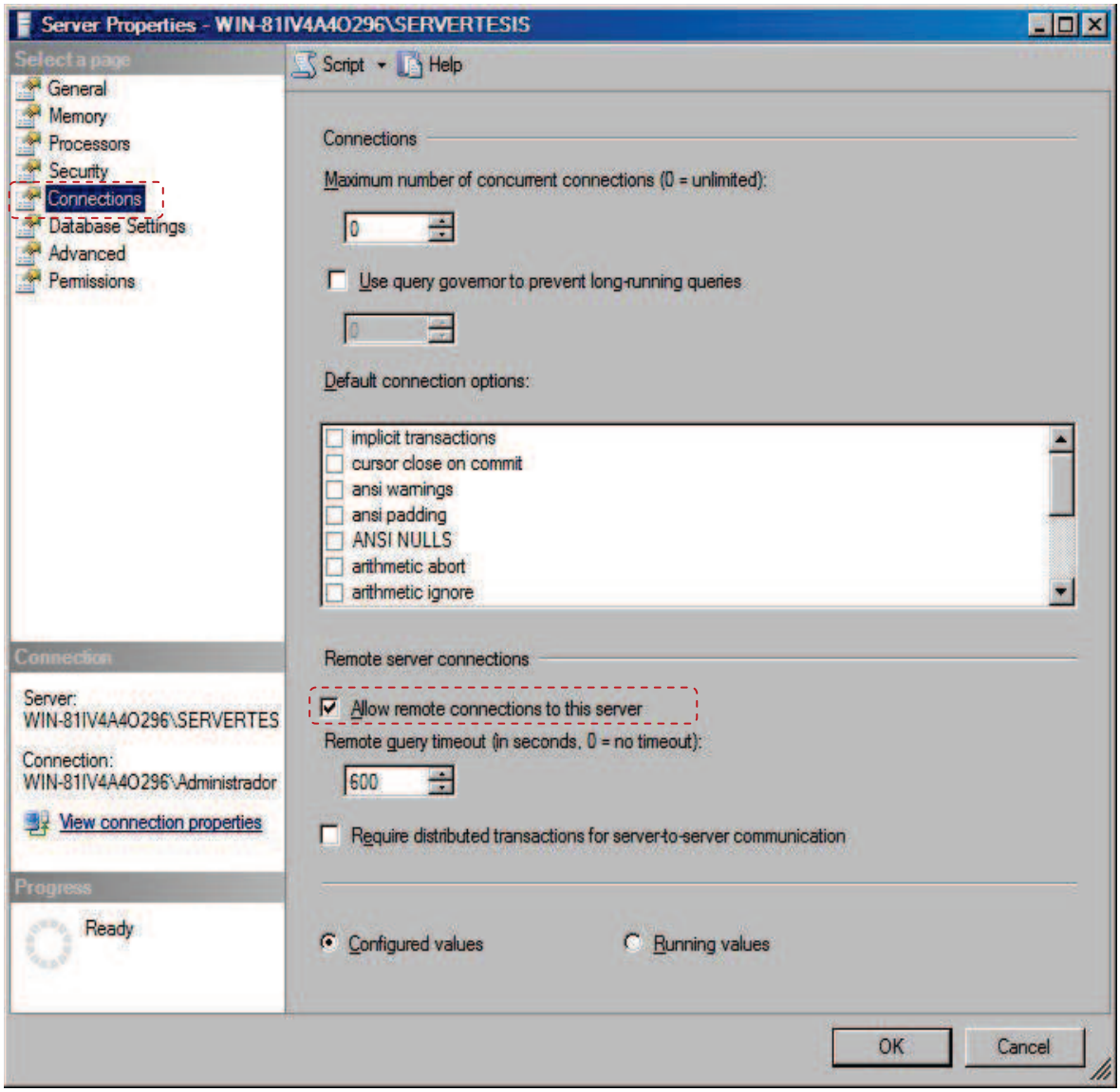

Figura 3.21 Configuración servidor de BDD paso 1

- Posteriormente desde el menú *Inicio* se dio clic en la opción *Programas*, luego se eligió *Microsoft SQL Server 2008*, a continuación *Configuration Tools*, y finalmente al presionar sobre *SQL Server Configuration Manager*, apareció la siguiente ventana de la Figura 3.22.

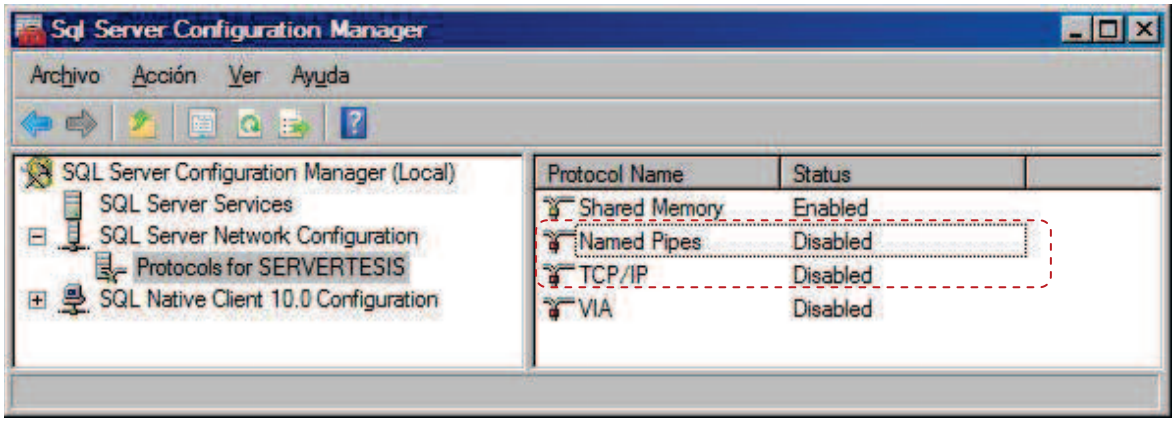

Figura 3.22 Configuración servidor de BDD paso 2

En el panel de navegación de la izquierda se expandió la opción *SQL Server Network Configuration*, y luego *Protocols for SERVERTESIS*.

Como se ve en la Figura anterior solo el protocolo de memoria compartida (*Shared Memory)* está habilitado que es el que se utiliza cuando se realiza una conexión a SQL desde el mismo servidor.

Para nuestros propósitos fue necesario habilitar los protocolos "Canalizaciones con nombre" (*Named Pipes*) y *TCP/IP*, lo cual se hizo con un clic derecho sobre el nombre de cada uno y luego en la opción *Enable*. Para que la configuración surta efecto fue necesario reiniciar el Servicio de SQL Server.

- Para reiniciar el servicio, en el Menú *Inicio*, en la opción *Ejecutar* se escribió *services.msc* y se presionó la tecla *Enter*, con lo que se abrió la Consola de Administración de Servicios. Figura 3.23.

Luego se desplazó hasta el servicio con nombre *"SQL Server (SERVERTESIS)"*, se situó sobre él y se pulsó el botón derecho del ratón luego de lo cual se seleccionó *Reiniciar*. Con esto se aplicaron los cambios efectuados en el paso anterior.

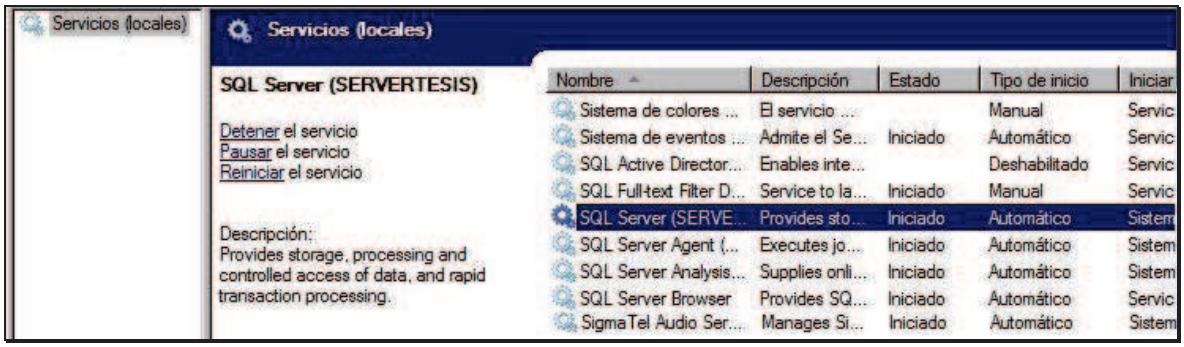

Figura 3.23 Configuración servidor de BDD paso 3

- Se habilitó el servicio *SQL Server Browser*, que se encuentra en la misma ventana de Servicios, situándonos encima del mismo y se dio doble clic. En el Tipo de Inicio, se seleccionó Automático y se pulsó Iniciar para que el Servicio arranque, luego en Aceptar para cerrar la pantalla. Figura 3.24.

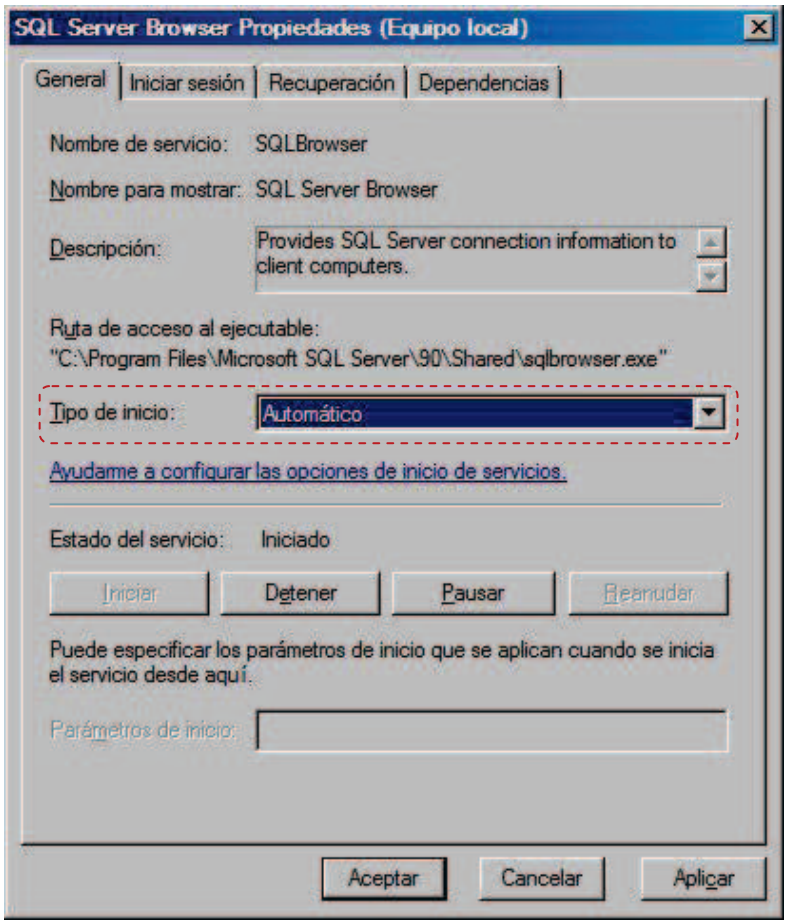

Figura 3.24 Configuración servidor de BDD paso 4

- Por último se configuró el *Firewall de Windows* para que los Servicios de SQL Server y SQL Browser puedan comunicarse con el exterior. Para esto

se fue al Menú de Inicio, se dio clic en Ejecutar, se escribió firewall.cpl y se pulsó Aceptar. Se abrió la ventana mostrada en la Figura 3.25.

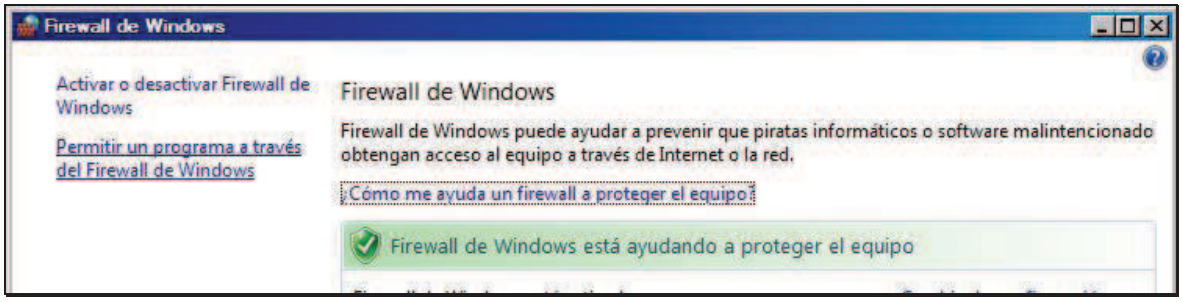

Figura 3.25 Firewall de Windows

Se pulsó la opción “Permitir un programa a través del Firewall de Windows” y se mostró la ventana de la Figura 3.26.

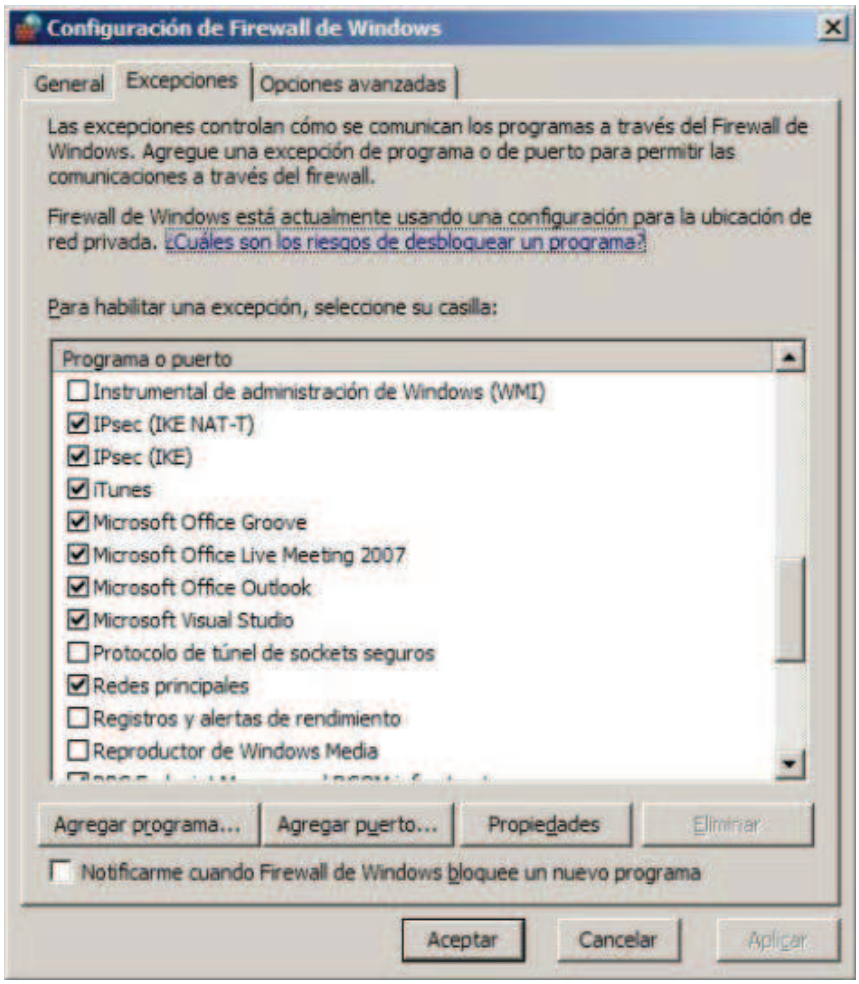

Figura 3.26 Configuración del firewall

Al pulsar agregar programa se mostró otra ventana en la que se dio clic en Examinar y se introdujo la dirección del Servicio de SQL Server: “C:\Program

Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\Binn" en la que se seleccionó el programa: "sqlservr.exe" y se pulsó Aceptar.

Se repitió la operación para añadir el *SQL Server Browser* que se encuentra en la carpeta: "C:\Program Files\Microsoft SQL Server\90\Shared". Se seleccionó el programa: "sqlbrowser.exe" y se pulsó el botón Aceptar. Después de eso ambas excepciones aparecieron en la pantalla de Configuración del Firewall. Figura 3.27.

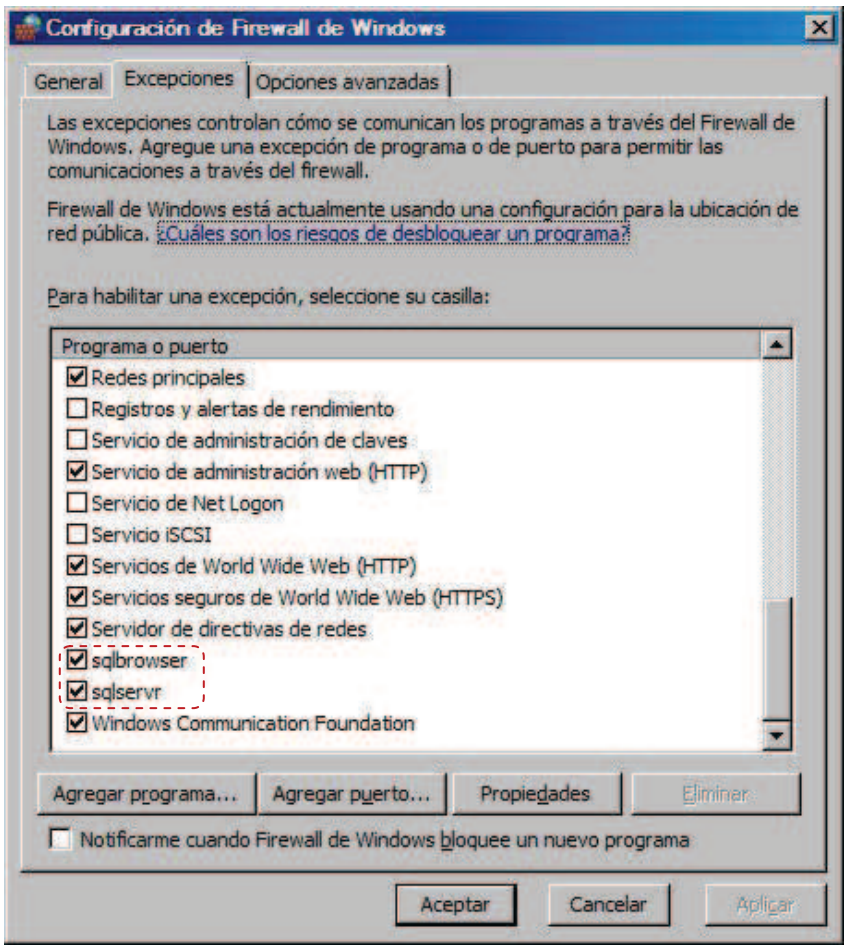

Figura 3.27 Excepciones en Firewall para SQL

Después de esto fue necesario crear un usuario para poder acceder a la base siguiendo los siguientes pasos. En el *SQL Server Management Studio* se pulsó botón derecho sobre la instancia del Servidor luego *Properties*. En la ventana que apareció se escogió la pestaña *Security* y se cambió el modo de autenticación de *Windows* a "*SQL Server and Windows Authentication mode"*, este cambio solo surtió efecto luego de reiniciar el servicio. Ver Figura 3.28.

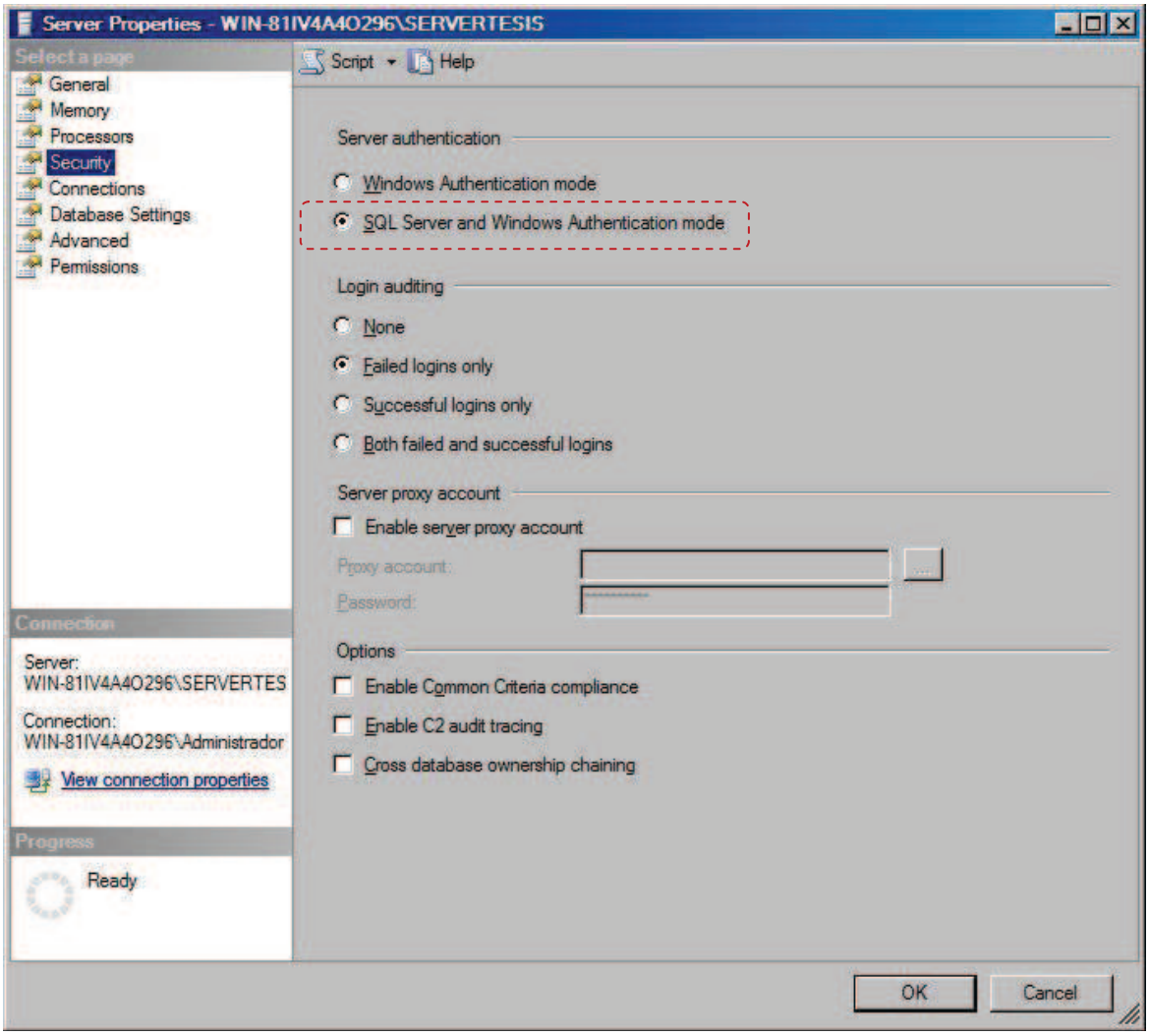

Figura 3.28 Creación de usuarios en MSSQL paso 1

Luego en el menú lateral se dio clic derecho sobre la pestaña *"Security"* luego *"New"* y a continuación *"Login"*. Figura 3.29.

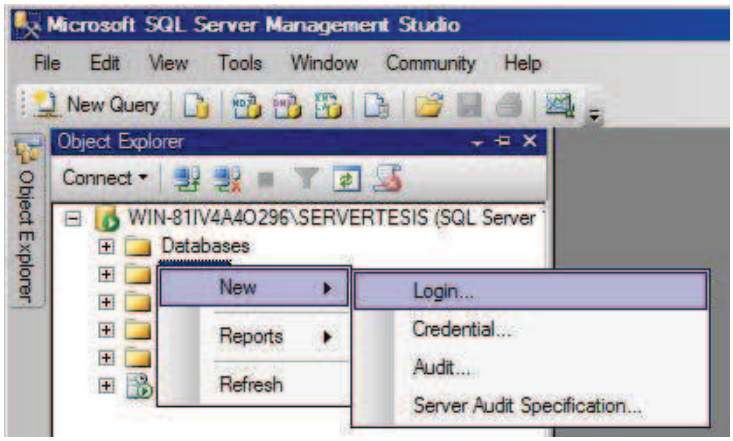

Figura 3.29 Creación de usuarios en MSSQL paso 2

En la ventana que se abrió (Figura 3.30) se dio clic en la pestaña *"General"*, se escogió *"SQL Server Autentication",* se dio un nombre de inicio de sesión que cumpla con las políticas de seguridad mínimas, (que tenga al menos 10 caracteres entre alfanuméricos y especiales), y se ingresó una contraseña segura, finalmente se marcó solo el checkbox "*Enforce password policy"*.

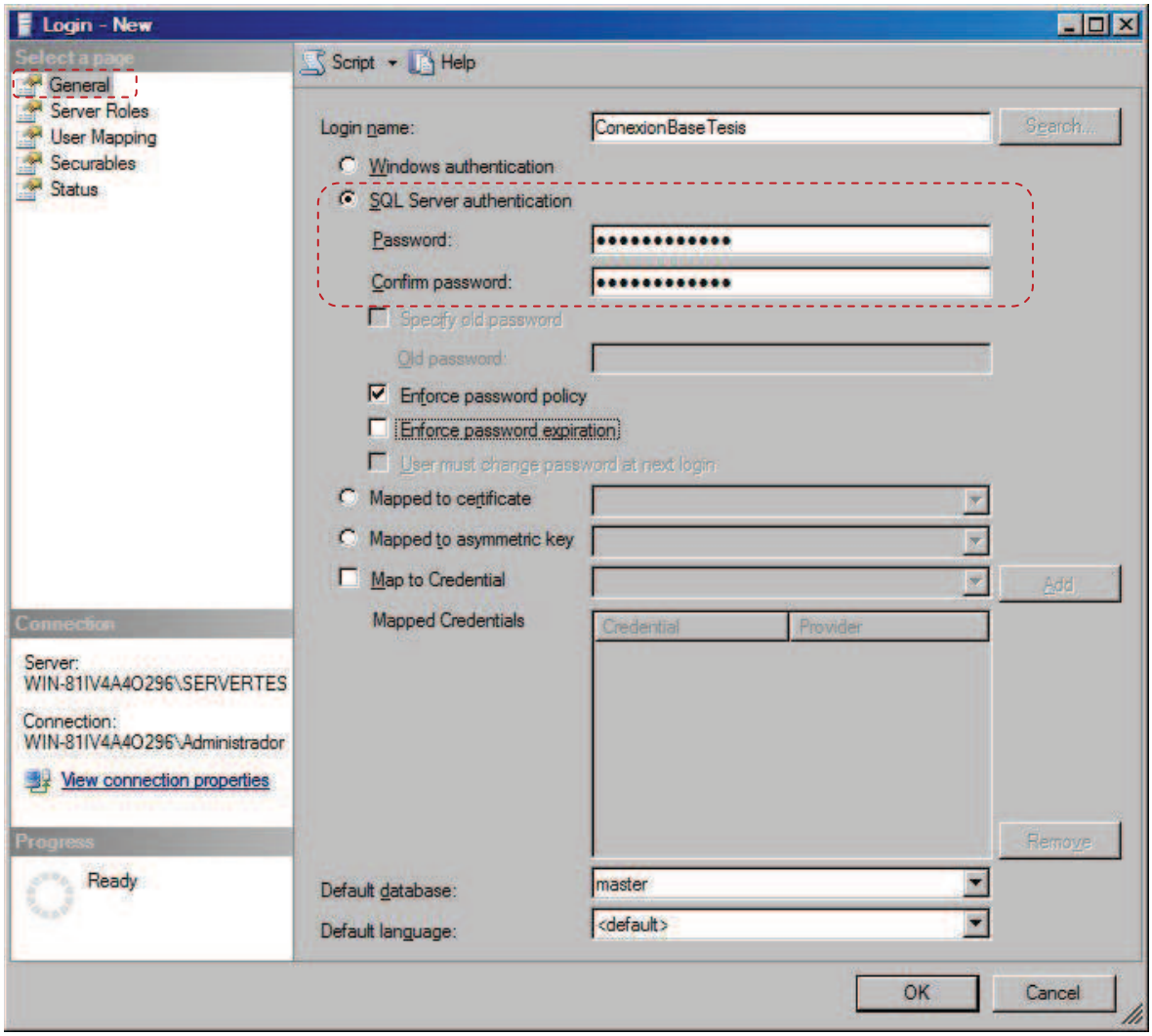

Figura 3.30 Creación de usuarios en MSSQL paso 3

Inmediatamente, en la misma ventana pero en la pestaña *"Server Roles"* se seleccionó la opción "*sysadmin*", con lo cual se otorgó los privilegios de administrador al usuario que se creó y luego clic en "OK" (Figura 3.31). Una vez terminado el proceso anterior se cerró el SQL Server 2008 y se reinició el servicio con el método explicado anteriormente.

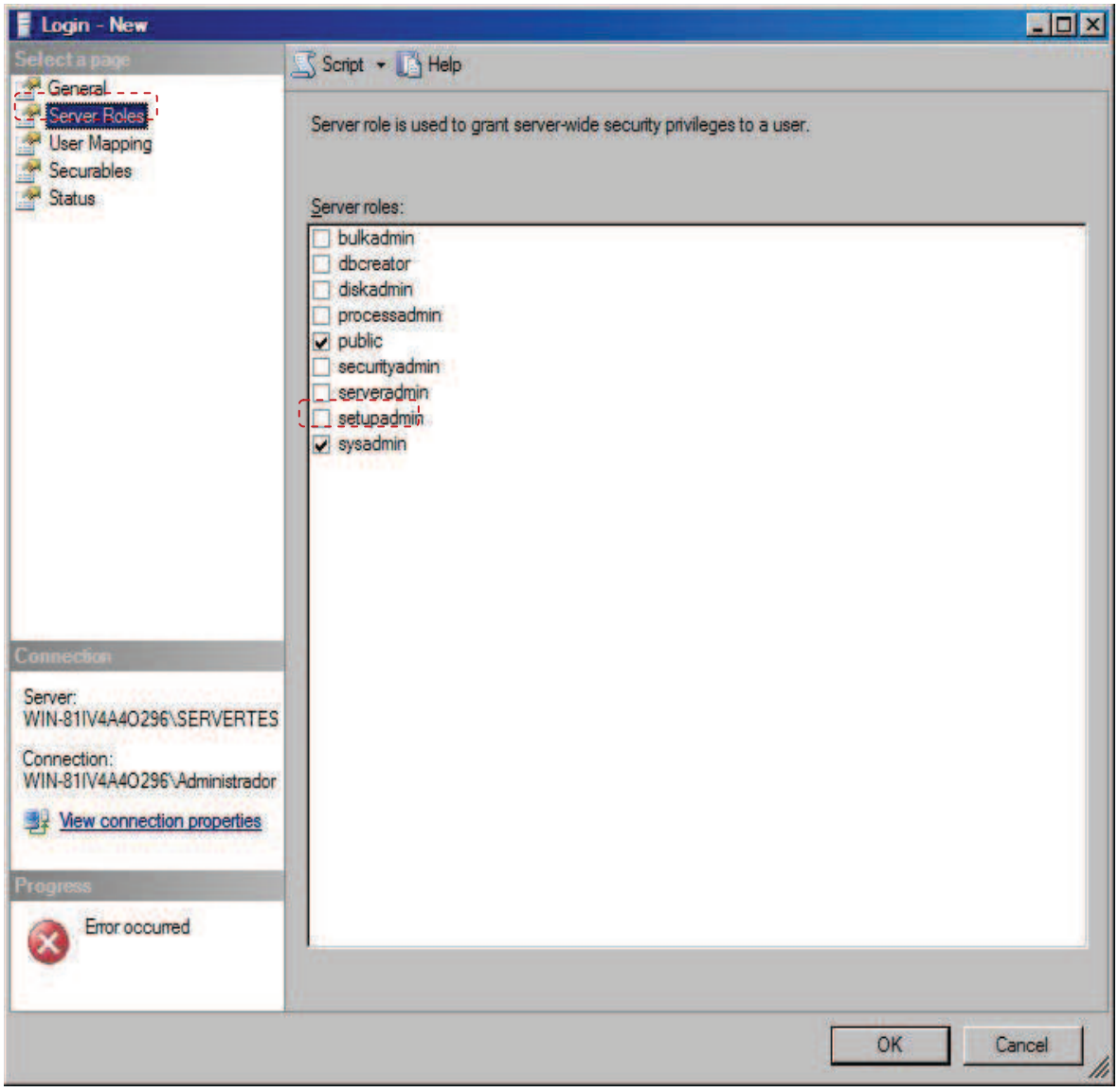

Figura 3.31 Creación de usuarios en MSSQL paso 4

3.5.2 SERVIDOR WEB

Para publicar la aplicación web, así como los servicios web, se instaló y configuró el *Internet Information Services (IIS) 7* que es el que viene por defecto en *Windows Server 2008*. Para poder brindar una mayor seguridad se realizó la comunicación entre clientes y servidores web mediante un canal seguro con el uso de un certificado *SSL*, el cual para efecto de pruebas fue creado en el propio servidor.

3.5.2.1 Creación de certificado SSL

Antes de enlazar reglas *SSL* al sitio, fue necesario importar e instalar un certificado de seguridad para usarlo en el enlace *SSL*.

Se ingresó al administrador del IIS haciendo clic en Inicio, Panel de Control, Herramientas Administrativas, *Administrador de Internet Information Services*. Una vez ahí, se hizo clic en el nodo raíz del árbol de la izquierda de la pantalla y se seleccionó el ícono Certificados de Servidor. Figura 3.32.

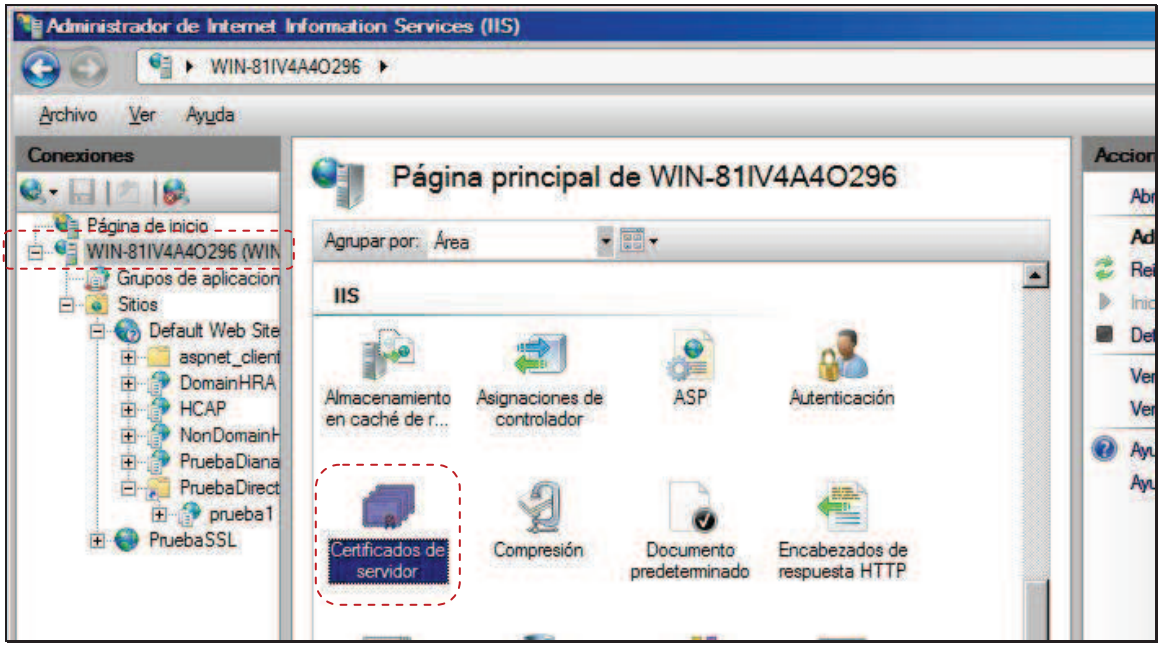

Figura 3.32 Creación de certificado *SSL* paso 1

Esto mostró una lista de todos los certificados registrados en la máquina (Figura 3.33), y permitirá importar y/o crear otros nuevos. Ya que solamente se utilizó para hacer pruebas, se creó un certificado propio haciendo clic en la opción *Crear Certificado Autofirmado.*

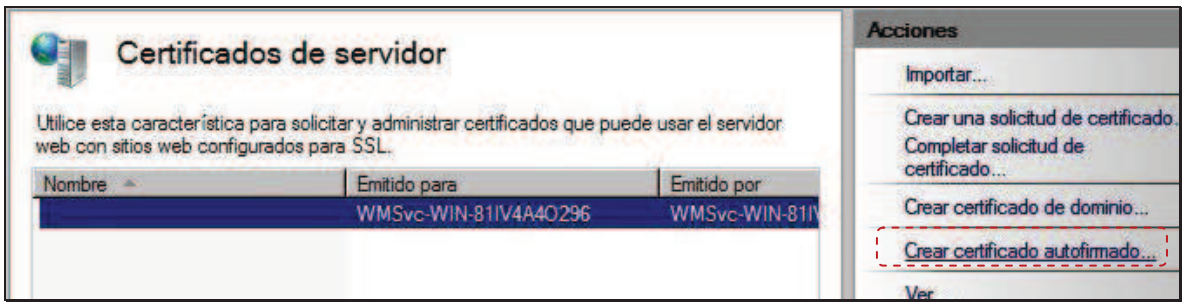

Figura 3.33 Creación de certificado *SSL* paso 2

Se ingresó el nombre para usar el certificado y se hizo clic en Aceptar (Figura 3.34). Con esto IIS7 creó automáticamente un nuevo certificado encriptado y lo registró en la máquina.

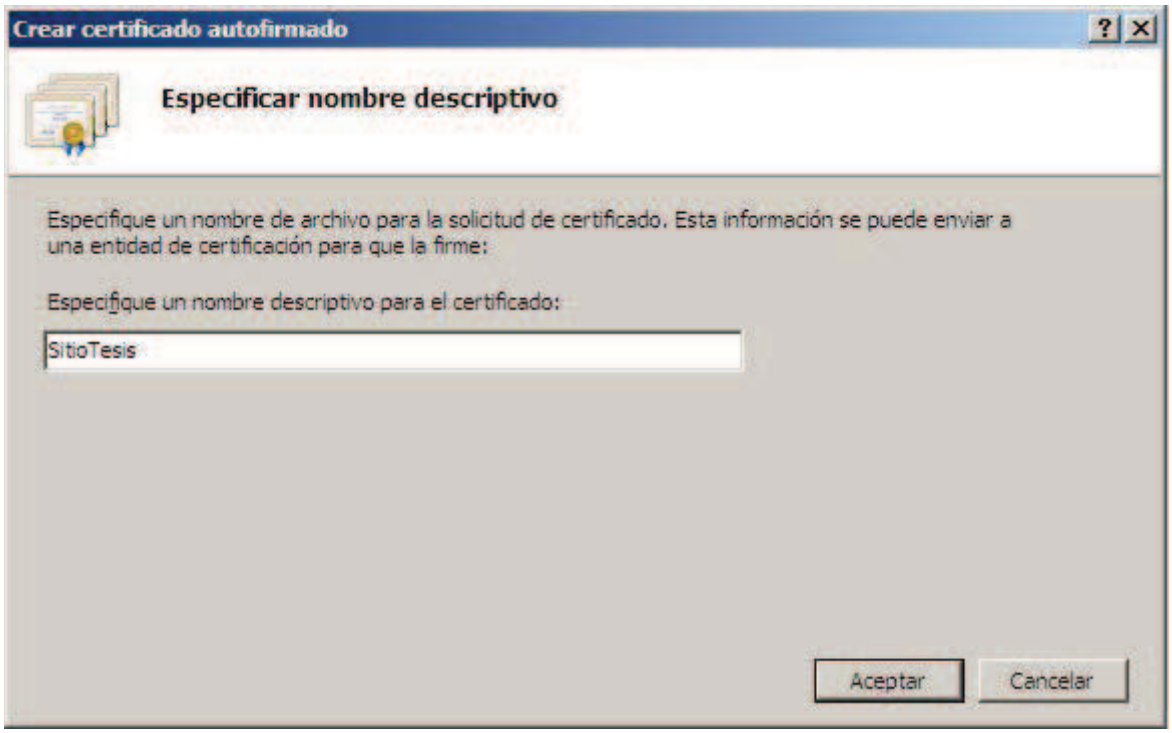

Figura 3.34 Creación de certificado *SSL* paso 3

3.5.2.2 Publicación de la aplicación web

Para publicar la aplicación web se hizo lo siguiente. En el menú *Inicio* se dio clic en *Herramientas administrativas*, y en *Administrador de Internet Information Services (IIS)*. Posteriormente en el árbol de la parte izquierda de la pantalla se detuvo el sitio web por defecto, y con clic derecho sobre *Sitios* se seleccionó *Agregar sitio web*. Ver Figura 3.35.

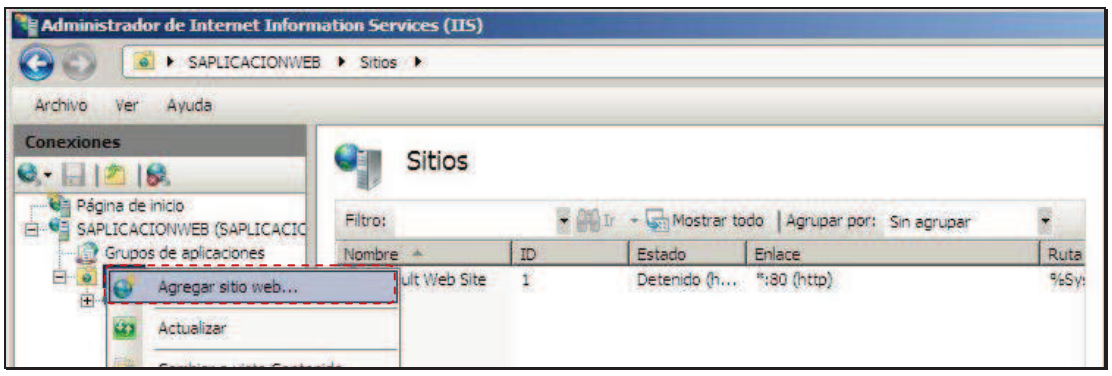

Figura 3.35 Publicación de la aplicación web paso 1

En la pantalla que aparece se escribió un nombre para el sitio web, en Grupo de aplicaciones se seleccionó *ASP.NET v4.0,* se seleccionó la dirección física y se dio clic en el botón *Conectar como*. Se mostró la pantalla donde se selecciona Usuario específico y se colocó el nombre de usuario y la contraseña. Figura 3.36.

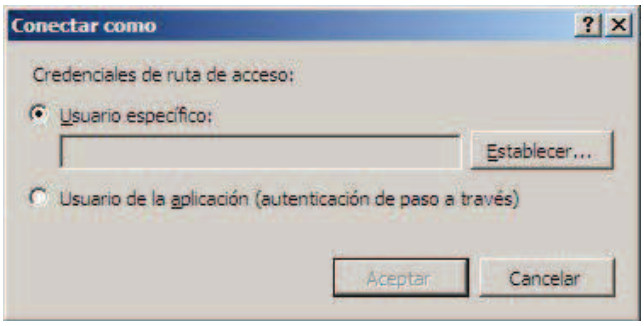

Figura 3.36 Publicación de la aplicación web paso 2

De vuelta en la pantalla anterior se dio clic sobre el botón *Probar configuración* con lo cual se comprobó que funcionaba bien (Figura 3.379. En la sección enlace se seleccionó tipo *https* con lo que se selecciona por defecto el puerto 443, y en *Certificado SSL* se seleccionó el creado anteriormente. Finalmente se dio clic en *Aceptar*.

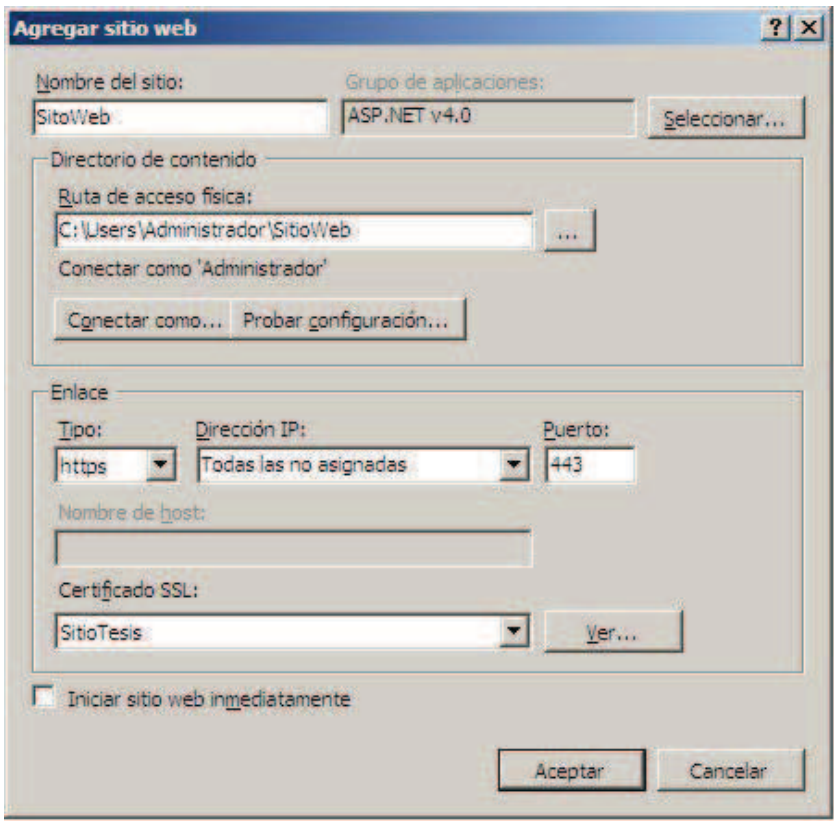

Figura 3.37 Publicación de la aplicación web paso 3

En el árbol de la parte izquierda de la pantalla se pudo observar el nuevo sitio web, al dar clic sobre él, se mostraron las carpetas que contiene la aplicación, se hizo clic derecho sobre ésta y se seleccionó Convertir en aplicación (Figura 3.38).

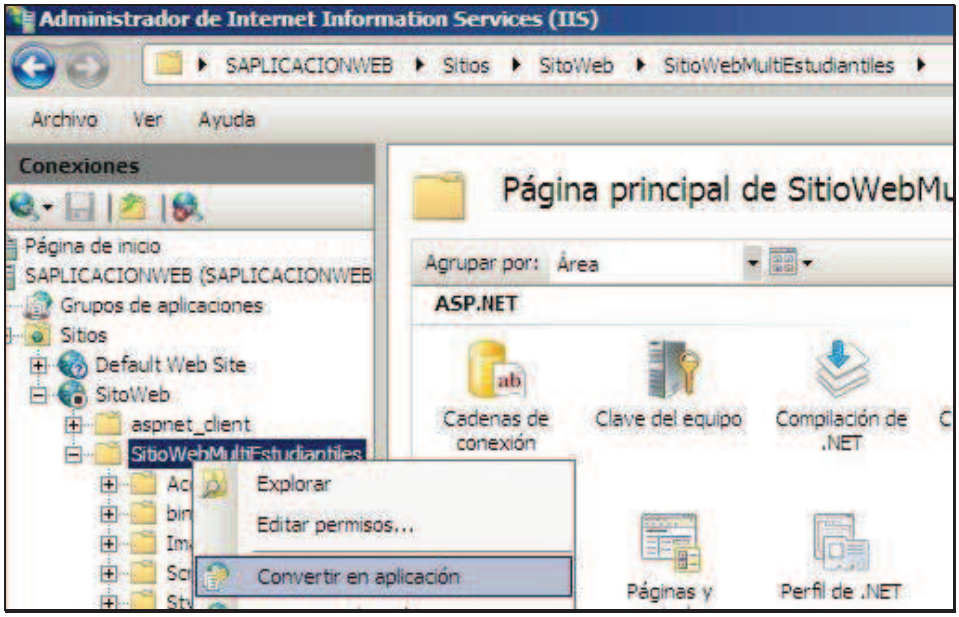

Figura 3.38 Publicación de la aplicación web paso 4

Se mostró la ventana mostrada en la Figura 3.39 donde se escribió un alias, en *Grupo de aplicaciones* se volvió a escoger *ASP.NET v4.0* y se confirmó la ruta física. Al presionar el botón *Conectar como,* se abrió una ventana donde se seleccionó *Usuario específico*, se colocó las credenciales y se dio clic en el botón *Aceptar.* Finalmente se dio clic en *Aceptar*.

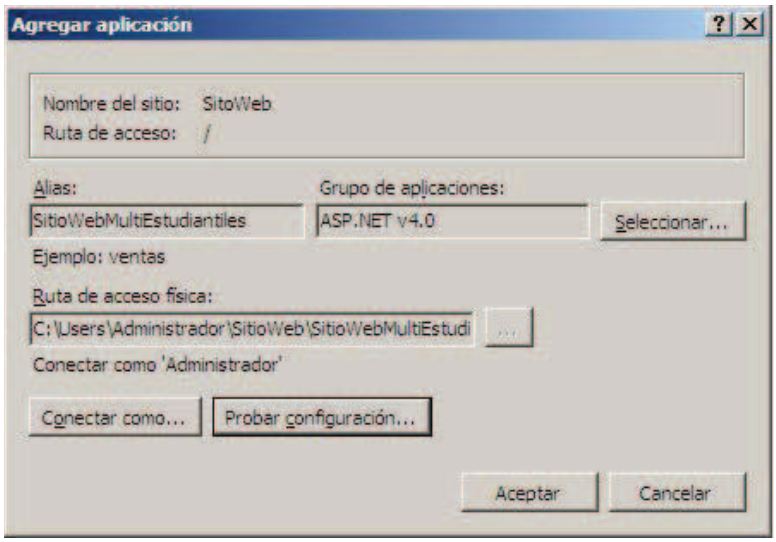

Figura 3.39 Publicación de la aplicación web paso 5

Para terminar en la parte derecha de la pantalla se dio clic en *Iniciar*, con lo cual el sitio entró en funcionamiento. Ver Figura 3.40.

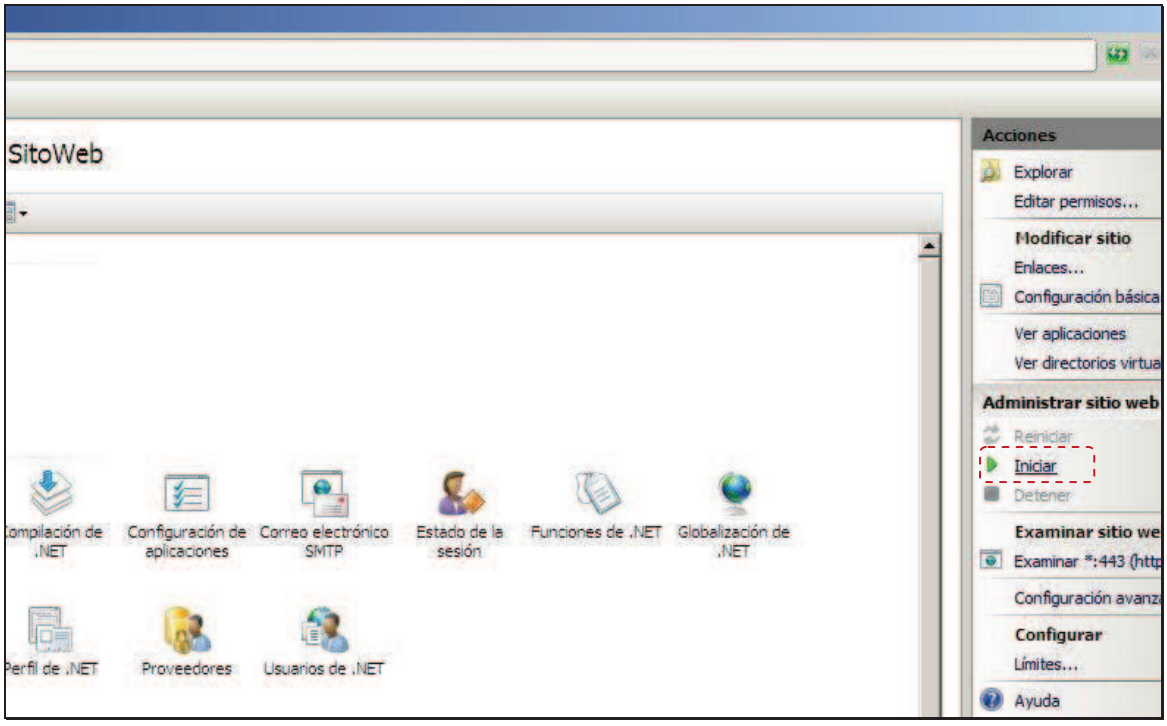

Figura 3.40 Publicación de la aplicación web paso 6

3.6 CONEXIÓN SEGURA ENTRE SERVIDORES

La implementación de IPsec en Microsoft Windows se basa en los estándares desarrollados por el Grupo de trabajo en ingeniería de Internet (IETF) de IPsec. La implementación de IPsec incluida en Windows Vista y en versiones posteriores de Windows se integra por completo en el nivel de red (nivel 3) del modelo de referencia de red de Interconexión de sistema abierto (OSI).

IPsec proporciona una variedad de servicios de seguridad de conexión al tráfico de la red. Puede configurar cada servicio para aplicarlo al tráfico de una red en particular creando una regla de seguridad de conexión en el Firewall de Windows con seguridad avanzada que identifique las características del tráfico de la red que debe proteger y la naturaleza de la protección que debe aplicar.

Aunque se puede configurar un servidor localmente usando el *Administrador de Directivas de Grupo* y otras herramientas directamente sobre el servidor, ese

método no es eficiente y no garantiza consistencia cuando se tiene que configurar muchas computadoras.

El objetivo de una configuración de *Firewall de Windows con Seguridad Avanzada* es el de brindar la seguridad de cada computadora bloqueando tráfico entrante no deseado en la red y protegiendo el tráfico deseado al atravesar la red. El tráfico de red que no cumple con las reglas configuradas en el *Firewall de Windows con Seguridad Avanzada* es excluido. La habilidad de manejar el *Firewall de Windows con Seguridad Avanzada* usando *Directivas de Grupo* permite al administrador aplicar reglas consistentes de una forma que no son fácilmente falsificadas por el usuario.

3.6.1 DIRECTIVAS DE GRUPO

Las Directivas de Grupo permiten realizar las tareas de administrador más eficientemente ya que habilita la administración centralizada de computadoras y usuarios, lo que reduce el costo total de propiedad de una infraestructura IT.

Las Directivas de Grupo son una tecnología disponible como parte de una implementación de los Servicios de Dominio de *Active Directory*. Cuando las computadoras miembros del dominio se conectan a su dominio de *Active Directory*, se recuperan y aplican objetos de Directivas de Grupo (GPOs) desde el controlador de dominio.

Un GPO es una colección de propiedades que pueden ser creadas por un administrador de dominio y luego aplicadas a grupos de computadoras o usuarios.

La configuración de propiedades y reglas que se quiera aplicar a las computadoras es almacenada en GPOs que son mantenidas en los controladores de dominio de un dominio de *Active Directory*. Los GPOs son automáticamente descargados a todas las computadoras asignadas cuando se conectan al dominio. Entonces ellas son unidas con los GPOs almacenados localmente en la computadora, y luego aplicados a la configuración activa de la computadora. Las directivas de grupo proveen fácil administración y control detallado de cuáles computadores reciben qué GPOs.

3.6.2 INSTALACIÓN DEL CONTROLADOR DE DOMINIO

Para instalar un controlador de dominio se ingresó al *Administrador del servidor* (Figura 3.41) y se hizo clic en *Agregar funciones.*

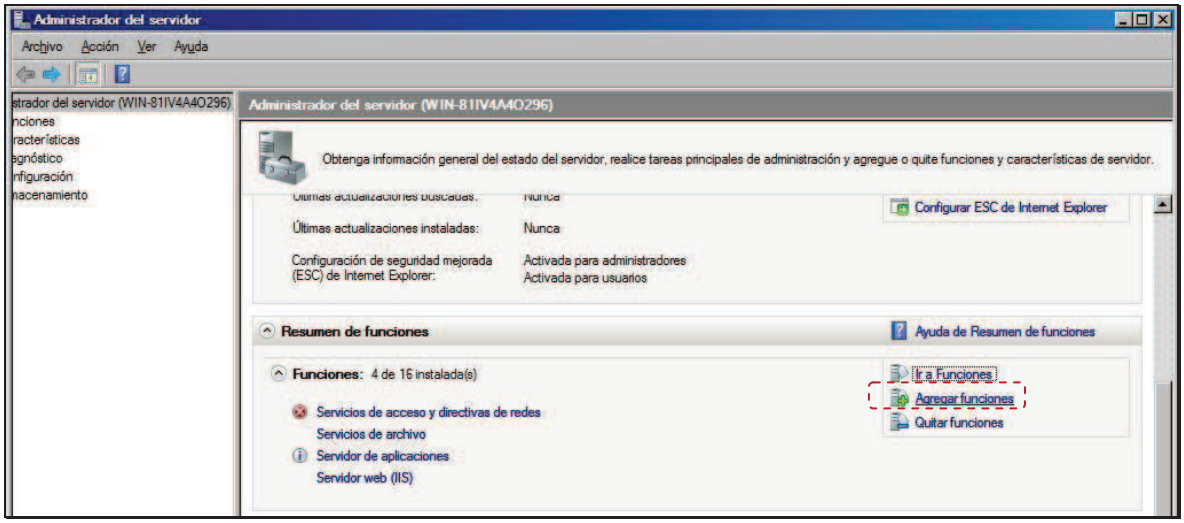

Figura 3.41 Instalación del controlador de dominio paso 1

En la pantalla mostrada en la Figura 3.42 se marcó la casilla *Servicios de dominio de Active Directory* y se dio clic en el botón *Siguiente.*

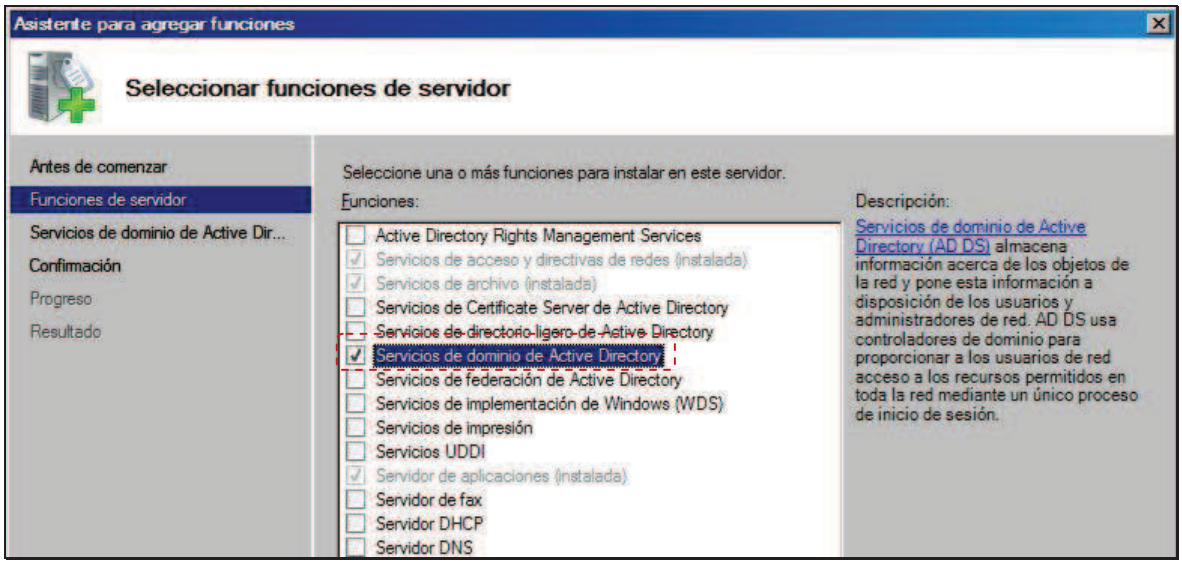

Figura 3.42 Instalación del controlador de dominio paso 2

Se observó la pantalla mostrada en Figura 3.43 con una breve introducción sobre la función que se seleccionó, en la cual se pulsó el botón *Siguiente*.

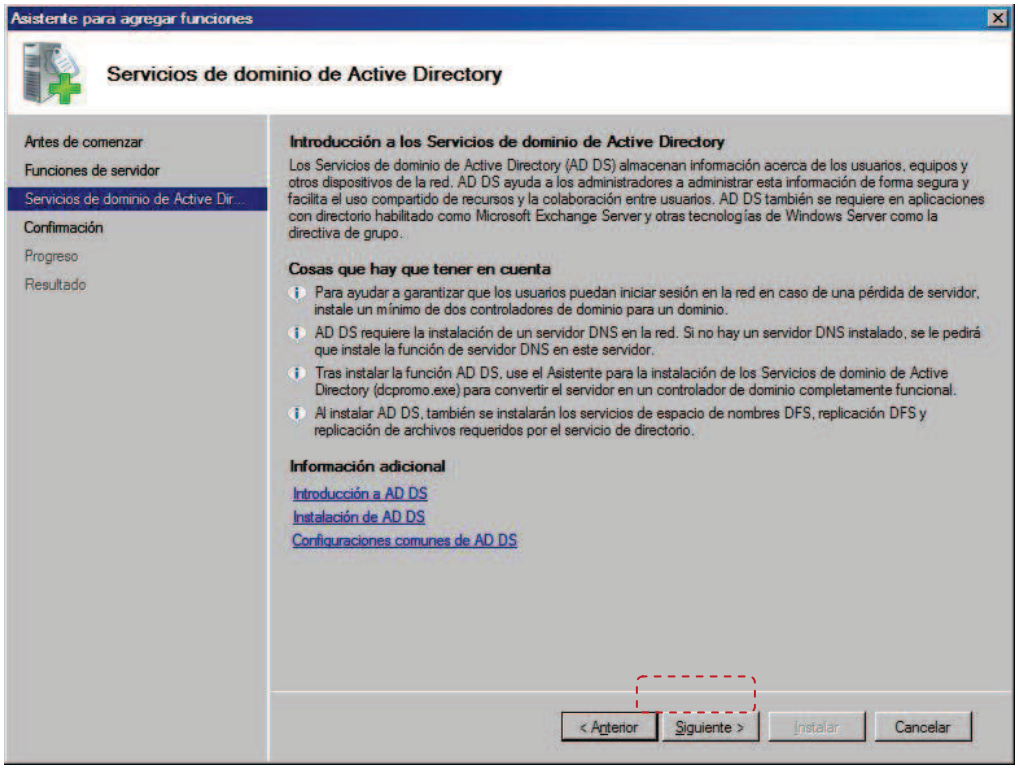

Figura 3.43 Instalación del controlador de dominio paso 3

Se observó la ventana para confirmar las selecciones de instalación (Figura 3.44), y tras comprobarlas se dio clic en *Instalar*, con lo cual inició la instalación.

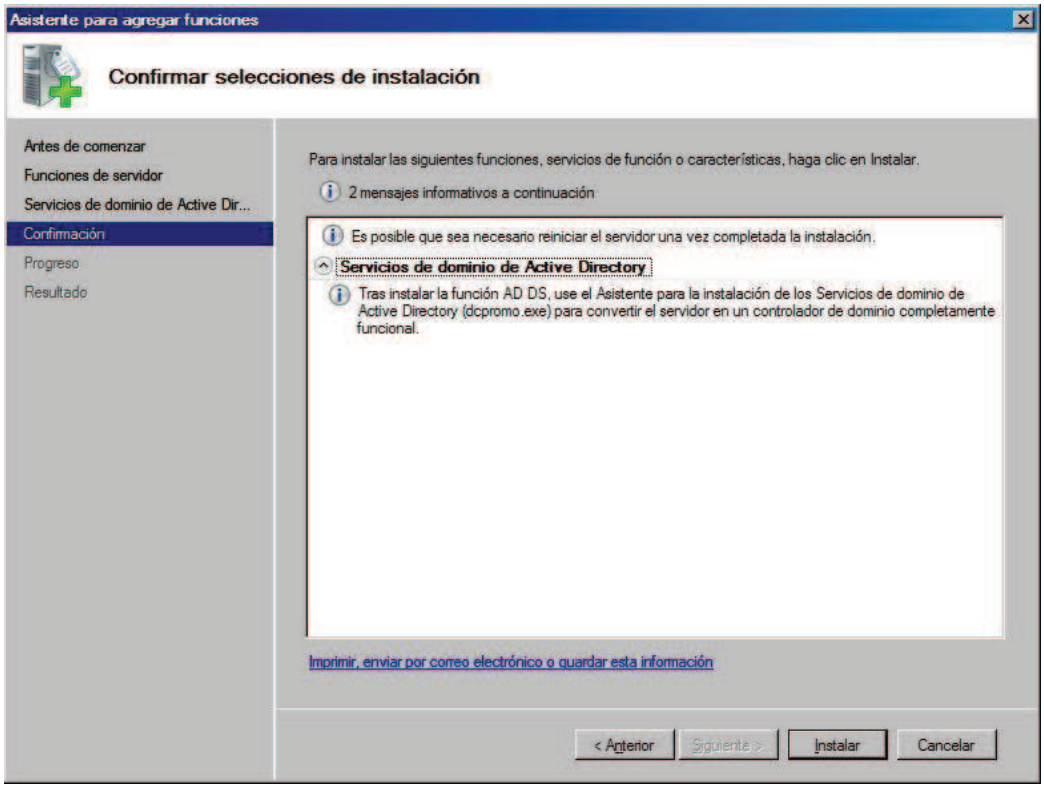

Figura 3.44 Instalación del controlador de dominio paso 4

Una vez terminada la instalación se cerró el asistente y en el menú Inicio, en ejecutar se escribió *dcpromo.exe* para iniciar el *Asistente para la instalación de los Servicios de dominio de Active Directory.* Al abrirse la ventana se dio clic en *Siguiente.*

Se observó una pantalla con información sobre compatibilidad donde se dio clic en el botón *Siguiente* y en las siguiente pantalla se seleccionó la opción *Crear un dominio nuevo en un bosque nuevo*, luego se dio clic en *Siguiente.*

En la siguiente pantalla mostrada en la Figura 3.45 se escribió un nombre para el dominio, que en este caso es *tesis.local* y se dio clic en *Siguiente.*

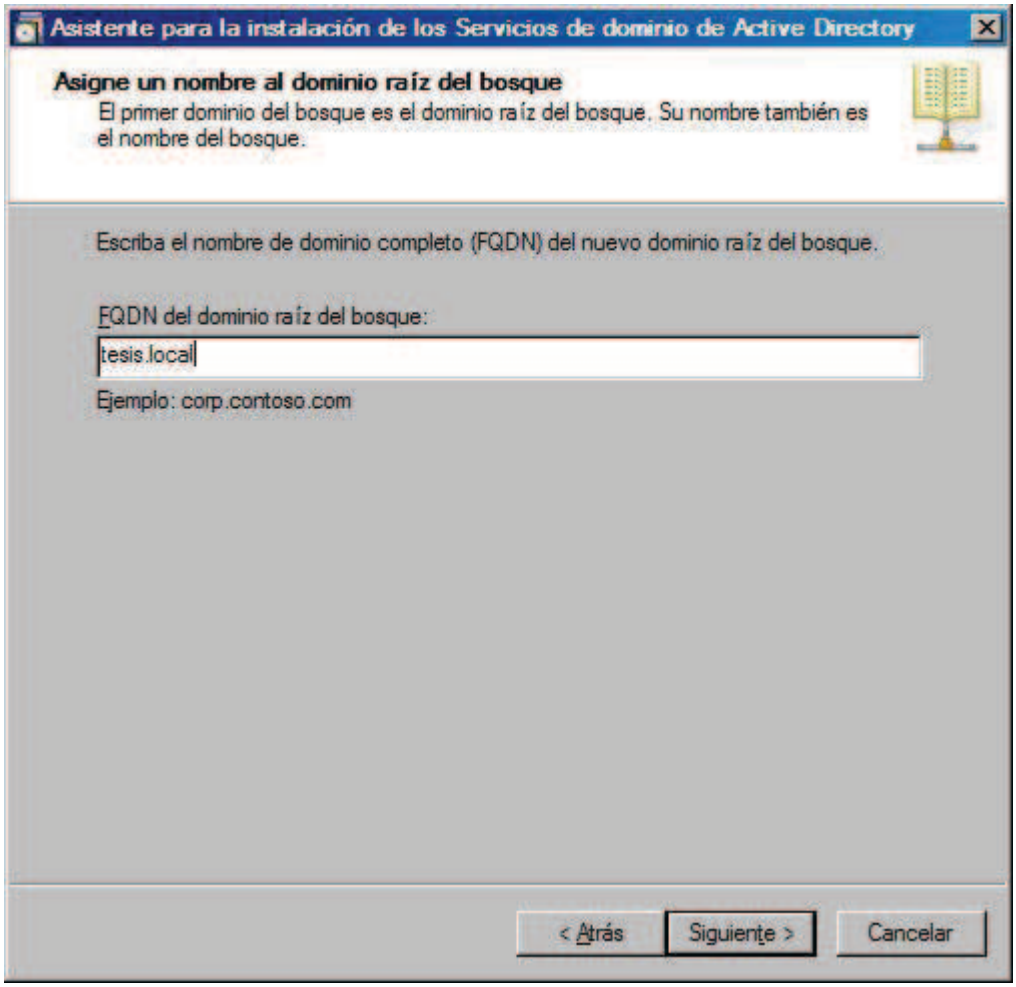

Figura 3.45 Instalación del controlador de dominio paso 5

En la siguiente pantalla (Figura 3.46) se seleccionó *Windows Server 2008* en el nivel funcional del bosque y luego en el botón *Siguiente.*

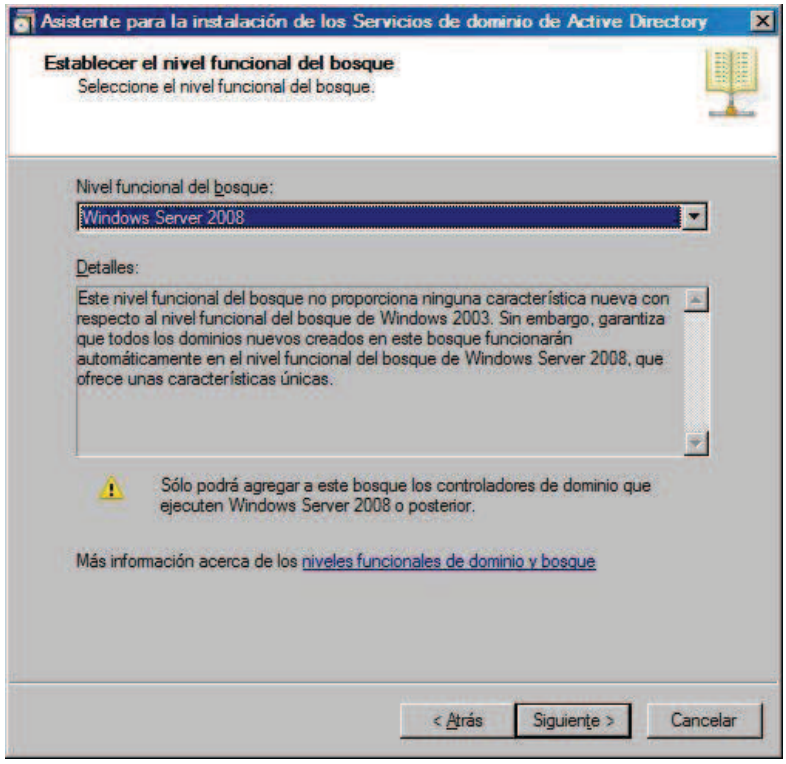

Figura 3.46 Instalación del controlador de dominio paso 6

En la siguiente pantalla (Figura 3.47) se muestran opciones adicionales para el controlador de dominio, donde se marcó el casillero *Servidor DNS,* luego se dio clic sobre *Siguiente.*

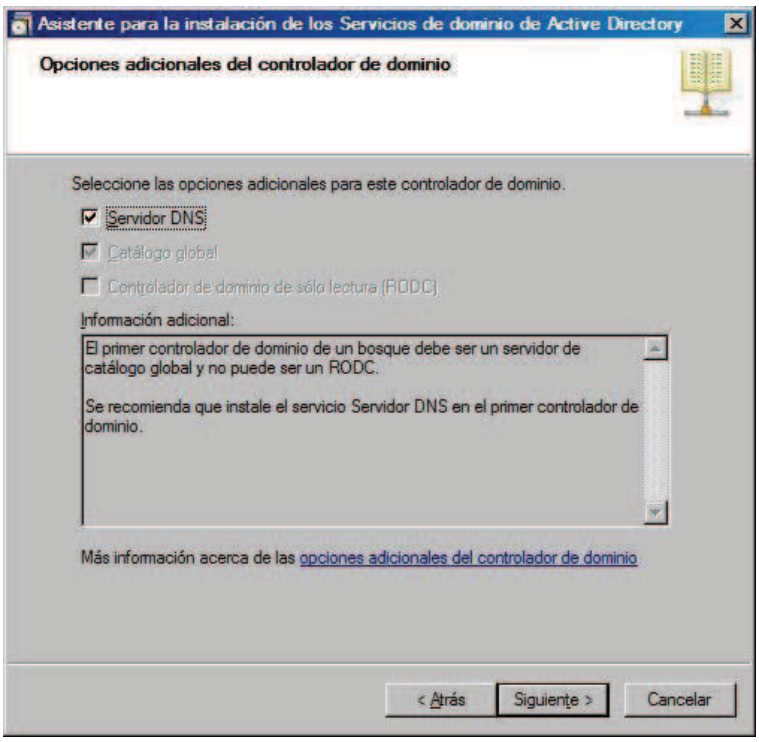

Figura 3.47 Instalación del controlador de dominio paso 7

Apareció la ventana de la Figura 3.48 que advierte de que la delegación no puede ser configurada para la zona principal, aquí se dio clic en el botón *Si*.

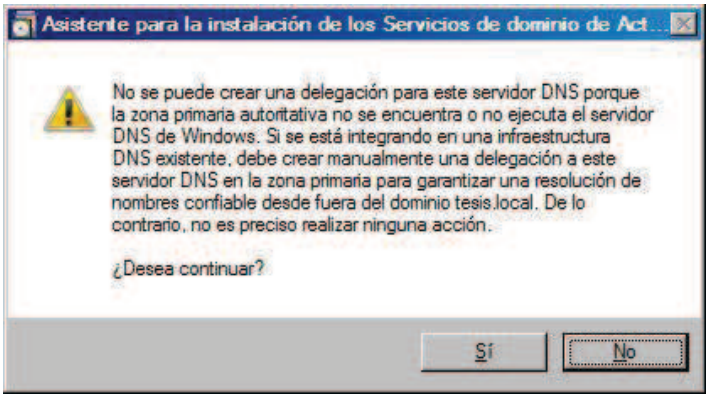

Figura 3.48 Instalación del controlador de dominio paso 8

Se mostró una ventana con la ubicación de la base de datos y los archivos de registro, donde se dio clic en el botón *Siguiente.* En la siguiente ventana (Figura 3.49) se escribió una contraseña segura, esta contraseña será necesaria en el caso que se necesite entrar en modo restauración de Active Directory. Después se hizo clic en *Siguiente.*

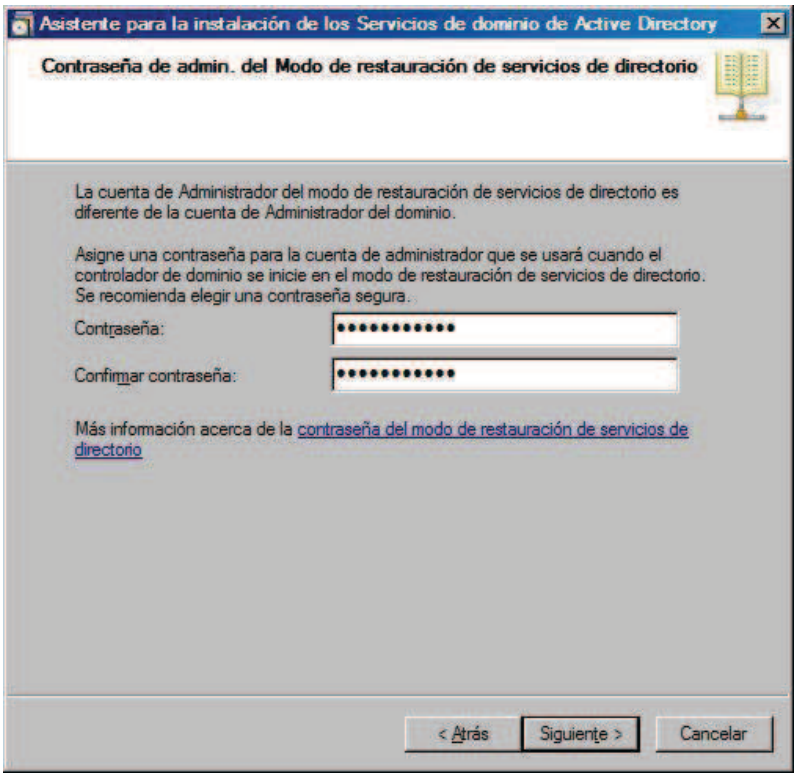

Figura 3.49 Instalación del controlador de dominio paso 9

En la siguiente pantalla (Figura 3.50) se comprobó que los datos son correctos y se pulsó sobre *Siguiente*.

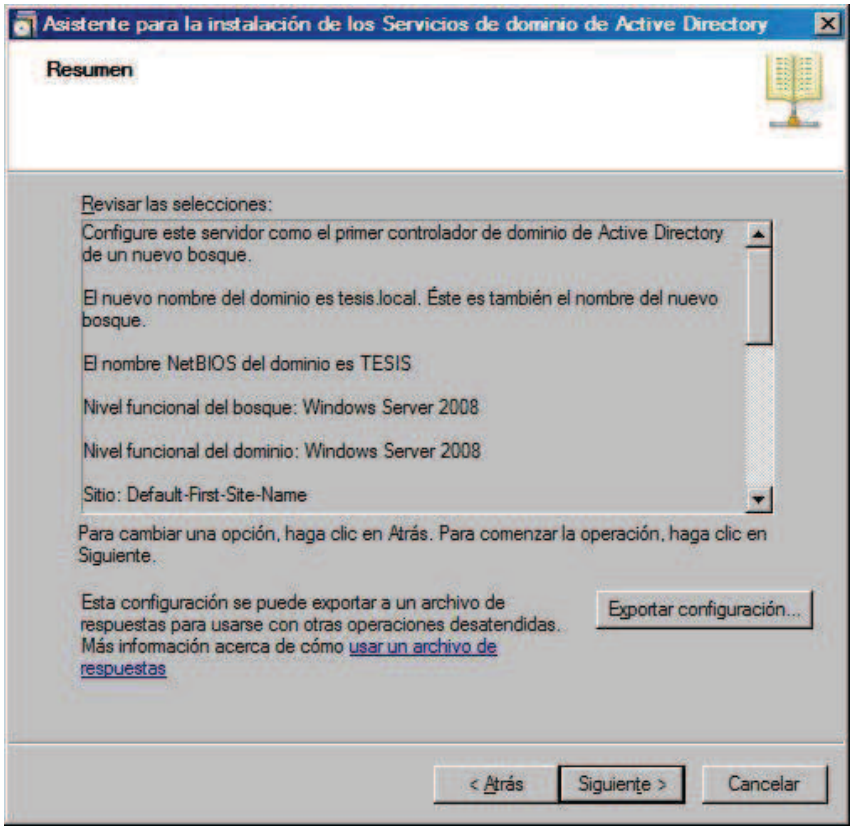

Figura 3.50 Instalación del controlador de dominio paso 10

En la siguiente ventana que se mostró (Figura 3.51), se seleccionó la casilla *Reiniciar al completar*.

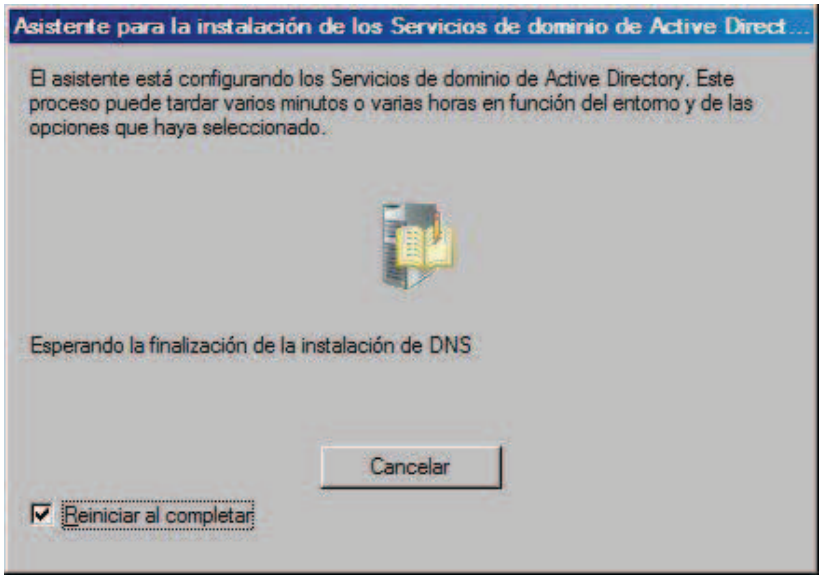

Figura 3.51 Instalación del controlador de dominio paso 11

En el *Administrador del servidor* en el panel de navegación se expandió *Funciones*, *Servicios de dominio de Active Directory, Usuarios y equipos de Active Directory*, *tesis.local*, y al hacer clic derecho sobre *Usuarios* se seleccionó *Nuevo*, *Usuario*, como se muestra en la Figura 3.52.

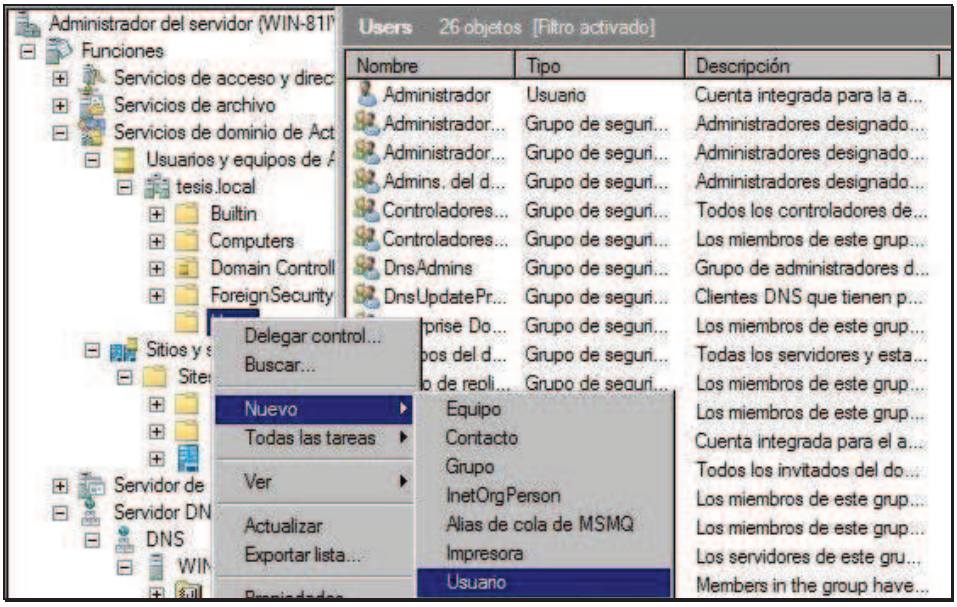

Figura 3.52 Creación de un usuario del dominio paso 1

Se abrió la pantalla mostrada en la Figura 3.53 y se colocaron los datos del nuevo usuario y se presionó el botón siguiente.

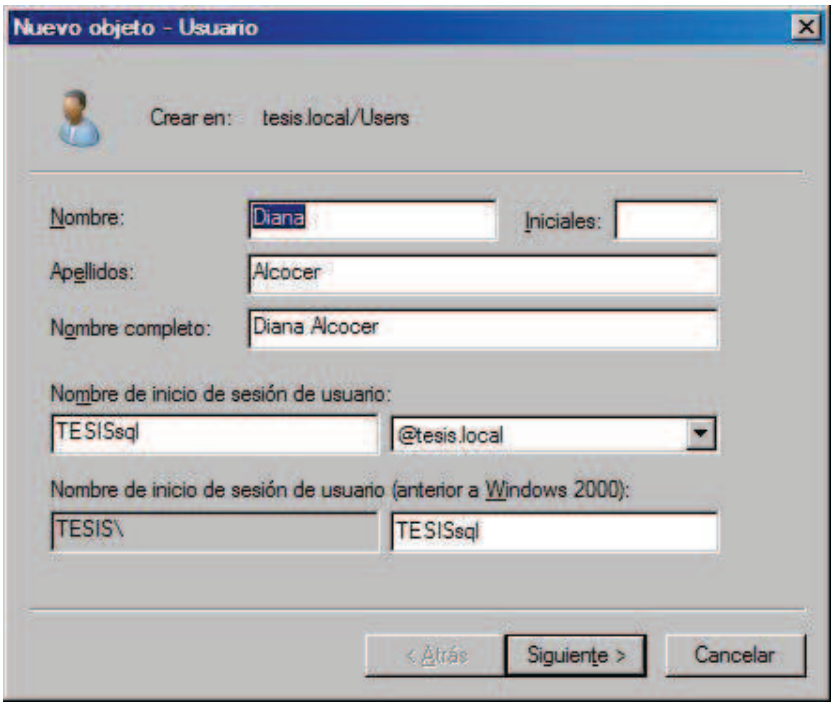

Figura 3.53 Creación de un usuario del dominio paso 2

Se escribió una contraseña segura para el usuario y se seleccionó la casilla *La contraseña nunca caduca*, luego se presionó el botón *Siguiente* (Figura 3.54). Se observó una ventana de resumen donde se dio clic en el botón *Finalizar*.

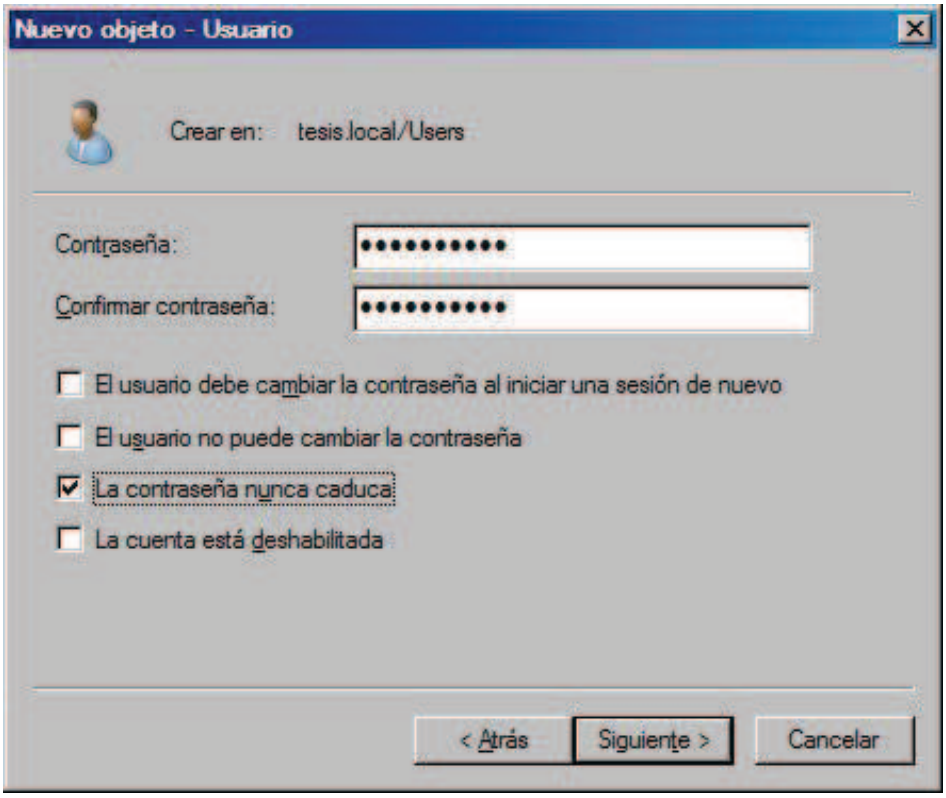

Figura 3.54 Creación de un usuario del dominio paso 3

Se unió el usuario creado al grupo de *Administradores del dominio* dando clic derecho sobre el nombre del usuario y seleccionando la opción Mover, luego se escogió la carpeta *Domain Controllers.*

3.6.3 CREACIÓN DE UNIDADES ORGANIZATIVAS

Para crear unidades organizativas y colocar las cuentas de usuario en ellas se siguieron los siguientes pasos:

En el servidor que tiene activado Active Directory, se hizo clic en el menú *Inicio*, luego en la opción *Herramientas administrativas*, y después en *Usuarios y equipos de Active Directory*. En el panel de navegación de la izquierda, se dio clic derecho en el árbol *tesis.local*, luego se dio clic en la opción *Nuevo*, y a continuación en *Unidad organizativa* como se muestra en la Figura 3.55.

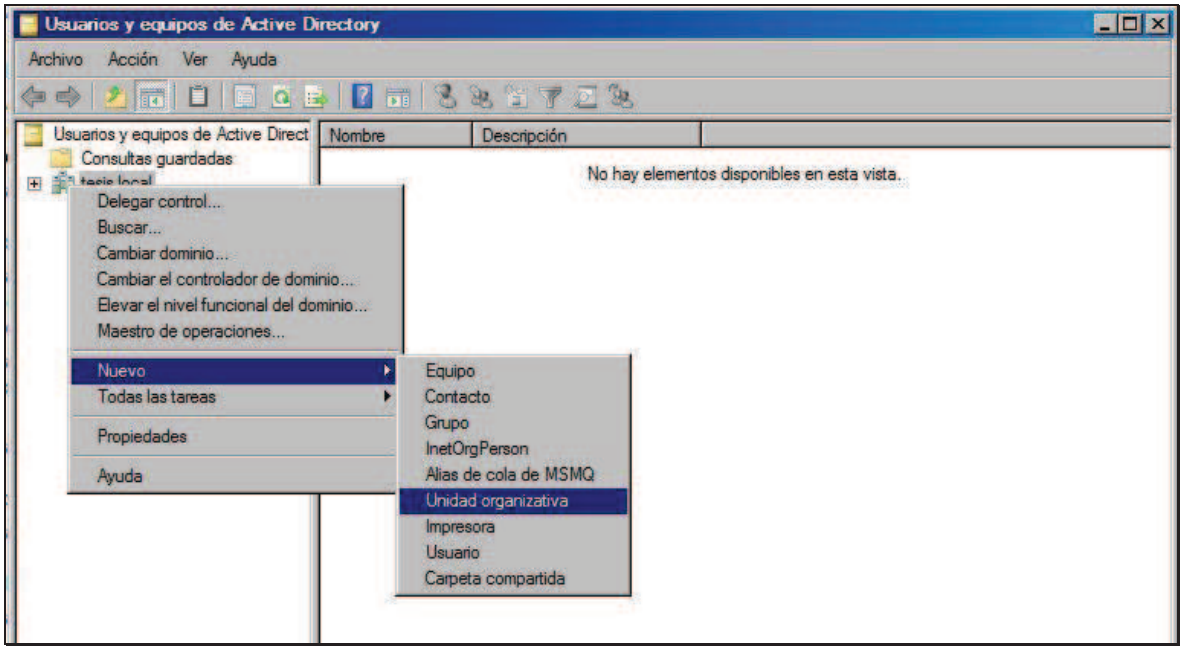

Figura 3.55 Creación de Unidades Organizativas paso 1

Apareció la ventana de la Figura 3.56 en la que se escribió el nombre *MisServidores*, y luego se dio clic en *Aceptar.*

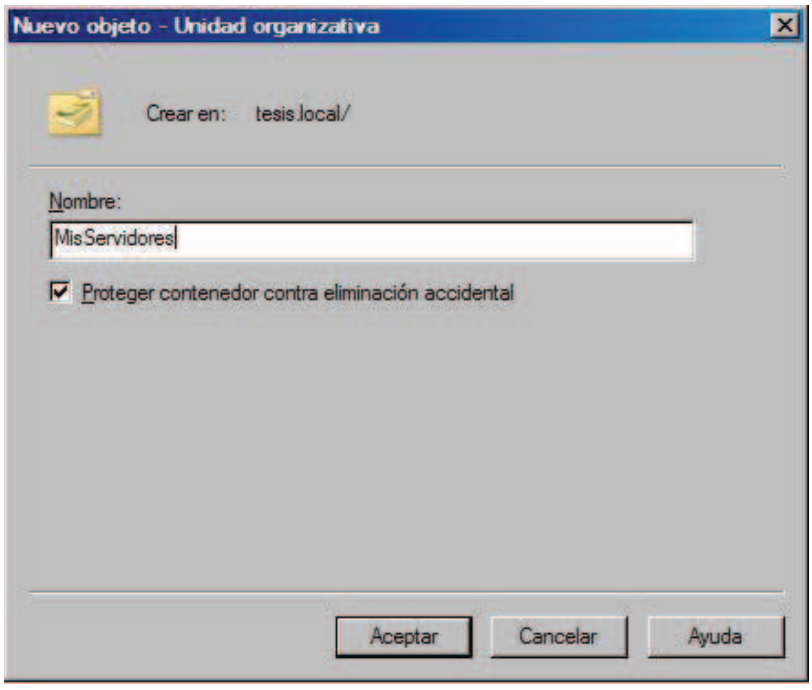

Figura 3.56 Creación de Unidades Organizativas paso 2

En el grupo creado irán los servidores de bases de datos y de servicios web. Para los clientes y para el servidor de la aplicación se creó otro grupo llamado MisClientes.

En el panel de navegación de la pantalla mostrada en la Figura 3.57 se dio clic en *Computers*, con lo cual se mostraron los equipos miembros del dominio. Se dio clic derecho en el nombre de cada equipo y se seleccionó la opción mover, con lo cual se escogió la unidad organizativa respectiva.

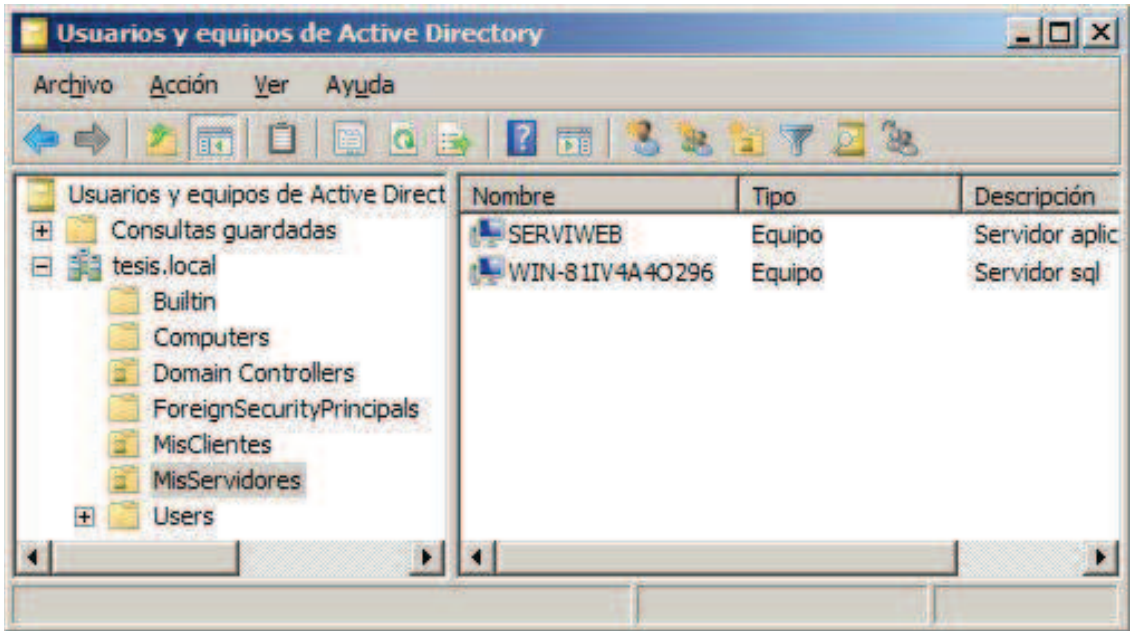

Figura 3.57 Creación de Unidades Organizativas paso 3

3.7 AISLAMIENTO DE SERVIDORES

El aislamiento de servidores restringe a los computadores miembros del dominio a requerir autenticación cuando se comunica con otros computadores miembros del dominio, y rechaza conexiones entrantes que no están autenticados, lo cual ayuda a mejorar la seguridad.

Se puede utilizar *IPsec* para proveer aislamiento de servidor. Al usar aislamiento de servidores, se puede restringir el acceso más ampliamente a datos sensibles, no solo al especificar a computadores y usuarios miembros del dominio, sino solo a esos usuarios y computadoras que tengan legítima necesidad. A menudo dichos datos deben ser encriptados durante la transmisión.

Al utilizar el *Firewall de Windows con Seguridad Avanzada*, se puede especificar qué conexiones de red específicas pueden ser accedidas solo por usuarios específicos, basados en su membrecía de grupo. Se puede también especificar

qué acceso es permitido solo por computadores específicos basados en una cuenta de membrecía en un grupo. Ambos tipos de restricción están basados en métodos de autenticación. Finalmente, también se puede especificar que esa conexión de red debe estar encriptada al usar uno o varios algoritmos de encripción.

3.7.1 CREACIÓN DE REGLAS DE SEGURIDAD PARA REFORZAR EL AISLAMIENTO DE SERVIDORES

Se creó reglas de seguridad de conexión para el dominio *tesis.local*, que causan que todos los computadores miembros del dominio requieran autenticación para tráfico de red entrante, y requieran autenticación para tráfico de salida. Como se dijo anteriormente se crearon dos grupos de dominio, uno para los servidores y otro para los clientes dentro del cual se encuentra el servidor de la aplicación web, esto con el objeto de crear una regla de excepción que solo permita comunicarse a dicho servidor con los del otro grupo y evitar que cualquier otro equipo que no pertenezca al grupo de servidores acceda a ellos.

Se creó un nuevo objeto de directiva de grupo (GPO) siguiendo los siguientes pasos. En el *Administrador de directivas de grupo*, se dio clic derecho sobre *Objetos de directivas de grupo*, y luego en *Nuevo*. Se abrió una ventana donde se escribió el nombre del objeto, y luego se dio clic en el botón *Aceptar*. (Fig. 3.58)

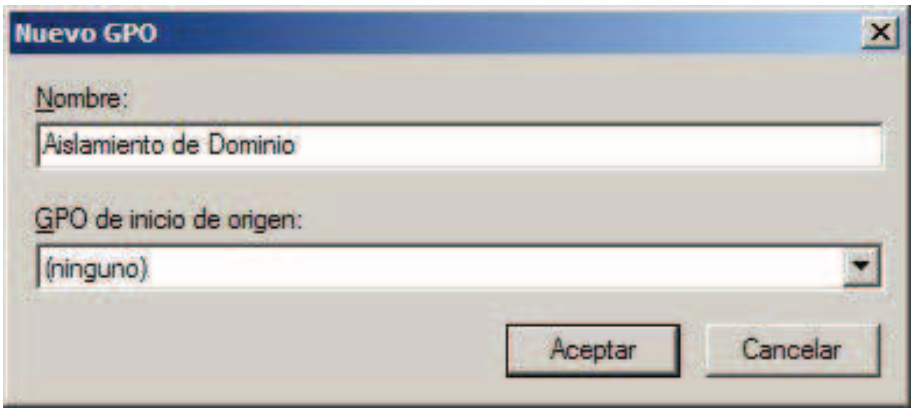

Figura 3.58 Aislamiento de servidores, paso 1

De vuelta en el *Administrador*, se dio clic derecho sobre el nuevo objeto creado, y se seleccionó *Editar*. En el panel de navegación del *Editor de administración de directivas de grupo*, se dio clic derecho sobre el nodo superior del GPO de

aislamiento de dominio, y luego se dio clic en *Propiedades*. Se seleccionó la opción *Deshabilitar los parámetros de configuración de Usuario* y luego se dio clic en *Si* en la pantalla de confirmación, y después en *Aceptar* (Figura 3.59).

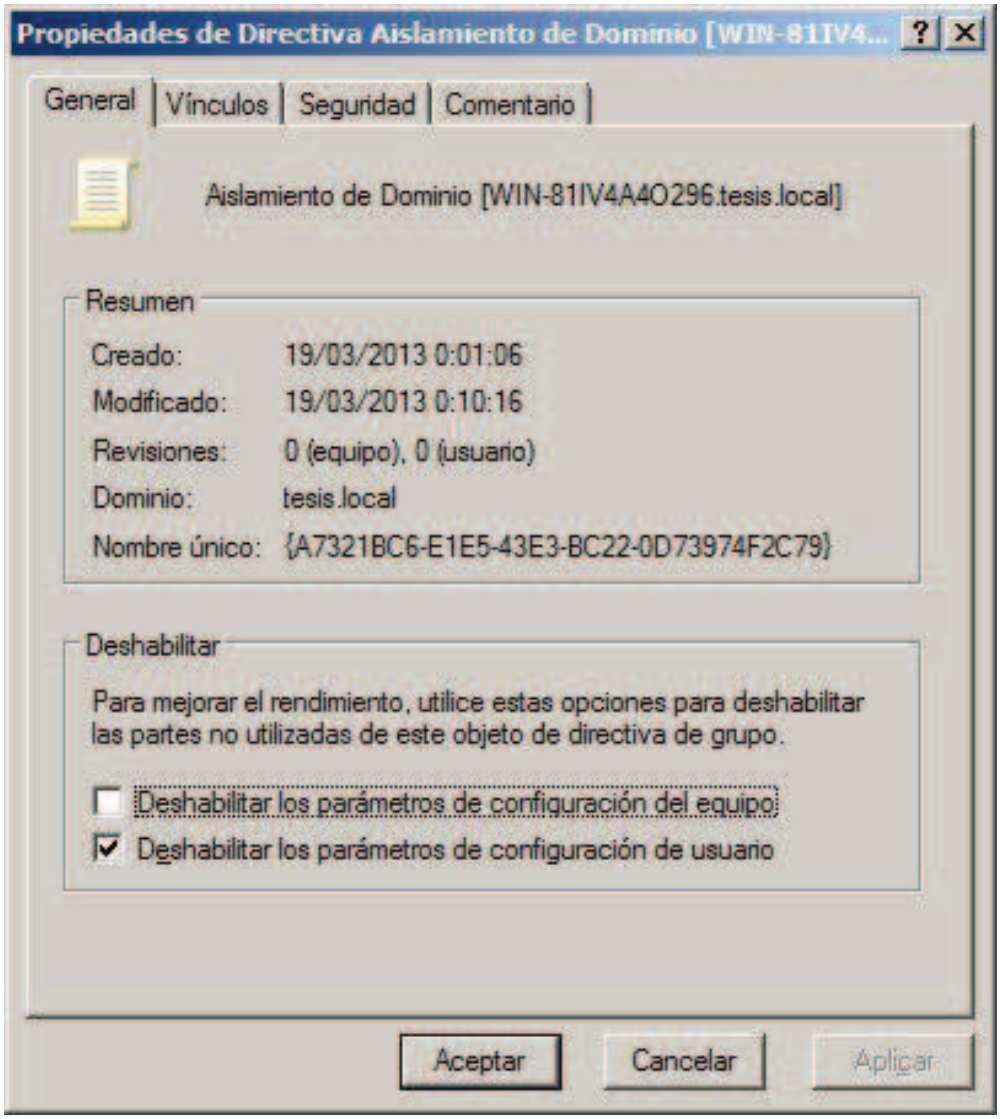

Figura 3.59 Aislamiento de servidores, paso 2

En el panel de navegación, se expandió *Configuración del equipo*, *Directivas*, *Configuración de Windows*, *Configuración de seguridad*, *Firewall de Windows con Seguridad Avanzada*, y finalmente *Firewall de Windows con seguridad avanzada-LDAP://cn={GUID},cn=policies,cn=system,DC=tesis,DC=local.* Ver Figura 3.60.

Figura 3.60 Aislamiento de servidores, paso 3

Se dio clic derecho en *Reglas de seguridad de conexión*, y luego en *Nueva regla*. En la página *Tipo de Regla* (Figura 3.61), se dio clic en *Aislamiento*, y luego clic en *Siguiente*

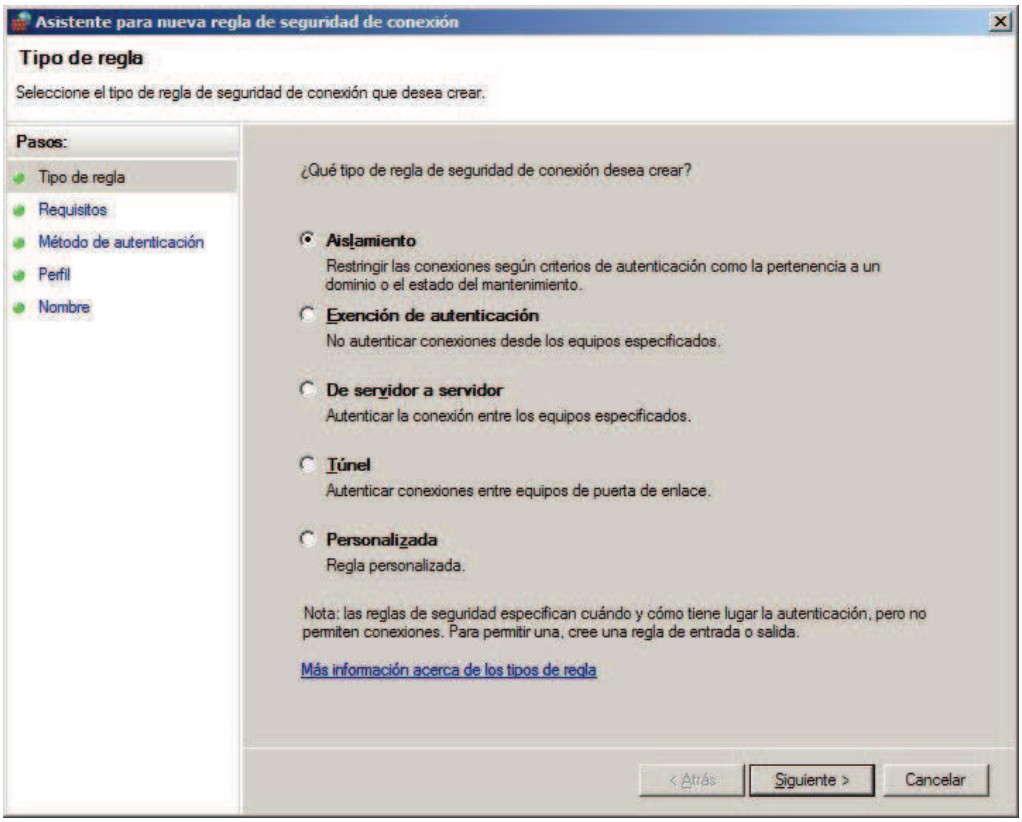

Figura 3.61 Aislamiento de servidores, paso 4

En la página *Requerimientos*, se confirmó que la casilla de *Solicitar autenticación para conexiones entrantes y salientes* estaba seleccionada y se dio clic en el botón *Siguiente*. En la página *Método de autenticación* se seleccionó la opción *Equipo y usuarios (Kerberos V5)* y luego se presionó el botón *Siguiente*. En la página *Perfil*, se seleccionó solo la casilla *Dominio* y se presionó el botón *Siguiente*. En la página *Nombre*, se escribió *Solicitud para Ingreso y Salida*, por último se dio clic en *Finalizar* (Figura 3.62).

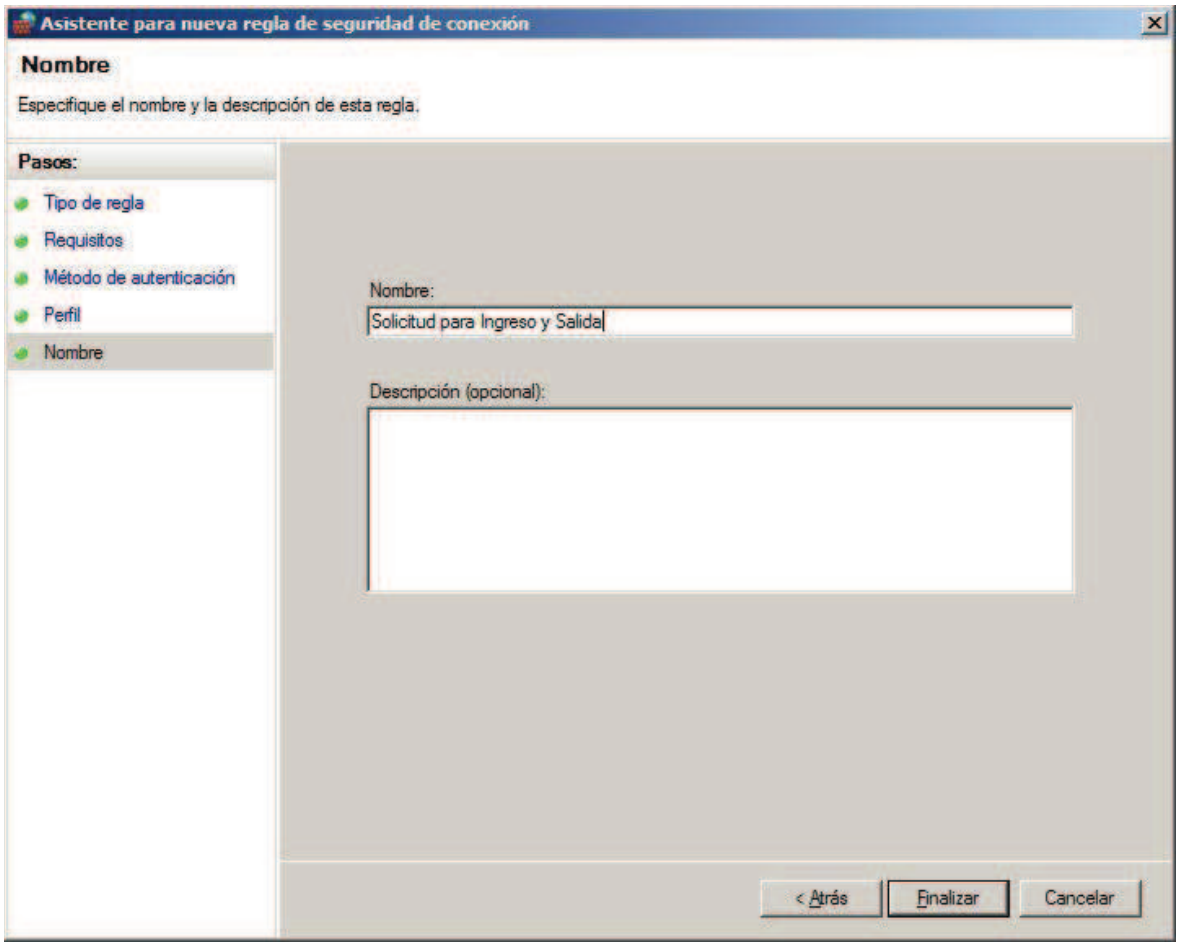

Figura 3.62 Aislamiento de servidores, paso 5

Para enlazar el GPO creado a una Unidad Organizativa (OU) apropiada, se abrió el *Administrador de directivas de grupo*, se dio clic derecho en *MisServidores* y luego se seleccionó *Vincular un GPO existente*. Se abrió la ventana mostrada en la Figura 3.63, donde se seleccionó la opción Aislamiento de dominio en *Objetos de directiva* de grupo y se dio clic en Aceptar.

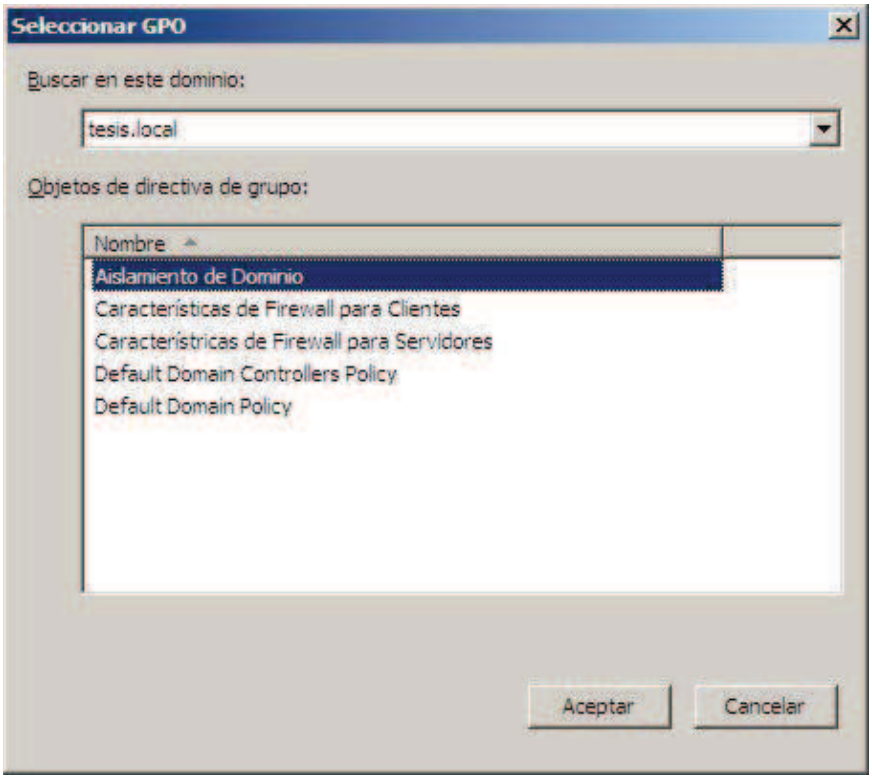

Figura 3.63 Aislamiento de servidores, paso 6

Para actualizar el GPO se escribió en la ventana de comandos *gpupdate/force*. Para comprobar la configuración se ingresó a la aplicación desde uno de los servidores, luego en el menú Inicio se dio clic en *Herramientas administrativas*, *Firewall de Windows con seguridad Avanzada*. Se expandió en el panel de navegación *Supervisión, Asociaciones de seguridad*, *Modo principal*. Aquí se despliega la información sobre las direcciones que intervienen, el método de primera y segunda autenticación, el cifrado, la integridad y el intercambio de claves, mostrada en la Figura 3.64.

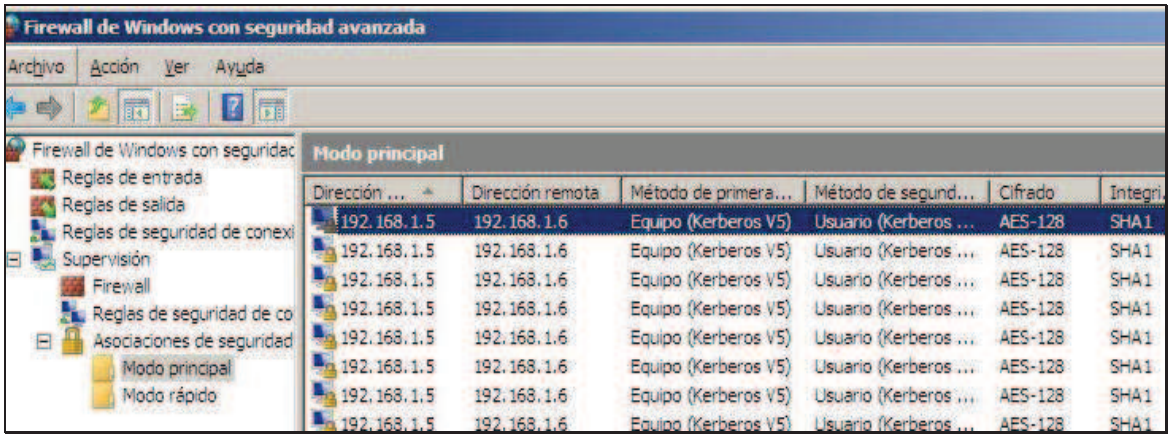

Figura 3.64 Aislamiento de servidores, paso 7

Se confirmó que la comunicación entre los servidores se realizó exitosamente, sin embargo para incrementar las medidas de seguridad se cambió la regla de aislamiento para que se requiera autenticación. Esto se logró al seguir los siguientes pasos: en el *Editor de administración de directivas de grupo*, se expandió el panel de navegación hasta encontrar la regla de seguridad de conexión que se creó de la forma explicada anteriormente; luego en el panel de resultados se dio clic derecho sobre la misma y se seleccionó *Propiedades.* En la ventana mostrada en la Figura 3.65 se dio clic sobre la pestaña *Autenticación* se cambió el modo de autenticación a *Requerir entrada y solicitar salida*, Finalmente se dio clic en el botón *Aceptar.*

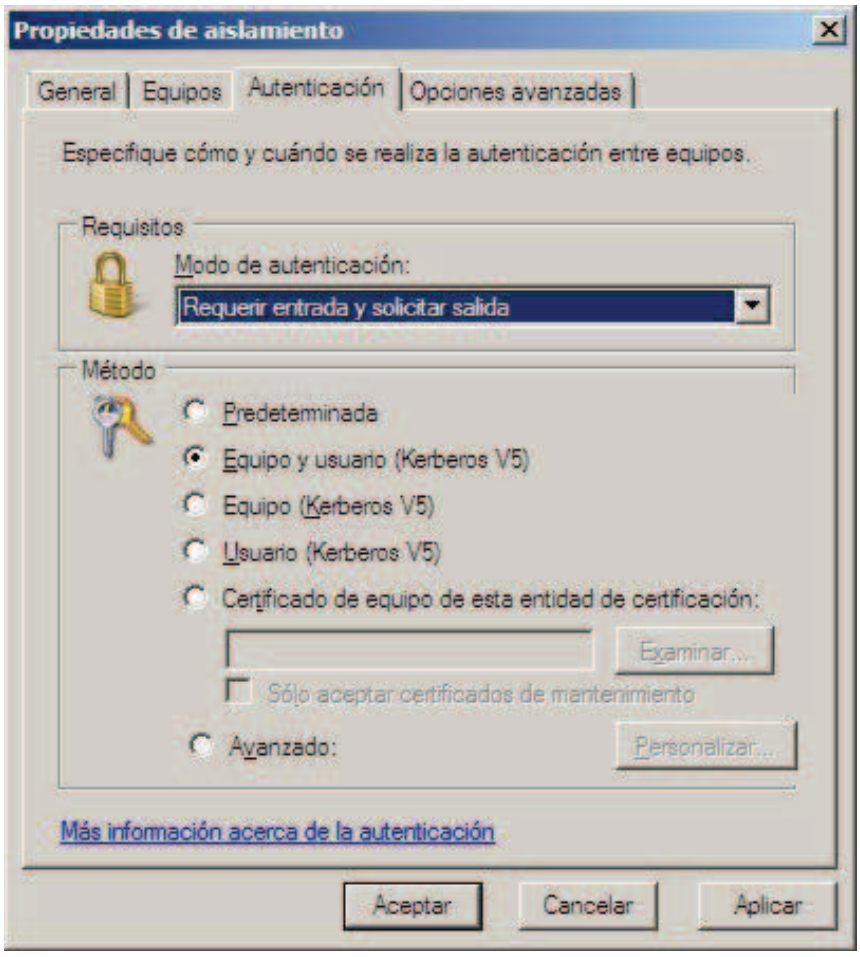

Figura 3.65 Aislamiento de servidores, paso 8

Se actualizó el GPO con el comando *gpupdate/force* y se comprobó que la comunicación entre los miembros del GPO era exitosa, mientras que desde un equipo de un GPO que no tiene esa regla no puede acceder a los servidores.

Con la finalidad de que solamente el servidor web tenga acceso a los otros servidores, se creó una regla de excepción. En el panel de navegación del *Editor de administración de directivas de grupo* se expandió el árbol hasta encontrar el *Firewall de Windows con seguridad avanzada*, se dio clic derecho en *Reglas de seguridad de conexión*, y se seleccionó la opción *Nueva regla*. En la ventana mostrada 3.66 se seleccionó la opción *Exención de autenticación* y se presionó el botón *Siguiente.*

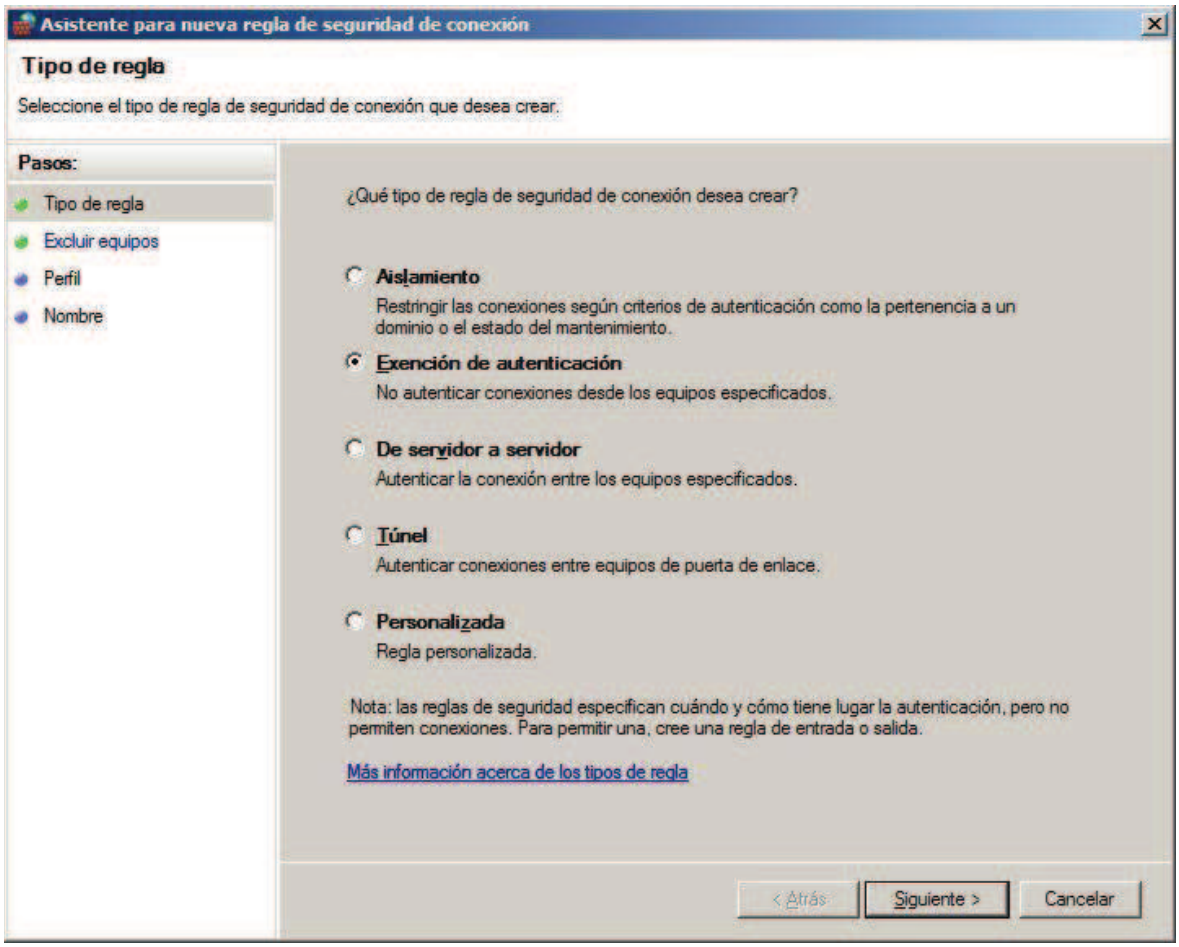

Figura 3.66 Aislamiento de servidores, paso 9

En la página *Excluir*, se escribió la dirección del servidor de la aplicación web y se presionó el botón *Siguiente.* En la página *Perfil* se seleccionó solo la opción *Dominio* y se presionó *Siguiente*. Por último se escribió un nombre para esta regla y se presionó el botón *Finalizar*.

Para reforzar el aislamiento de los servidores, se creó una regla de entrada de tráfico, para solicitar encripción. En el *Editor de administración de directivas*, se

expandió el panel de navegación hasta llegar a *Firewall de Windows con seguridad avanzada*, se dio clic derecho en *Reglas de entrada* y se seleccionó *Nueva Regla*.

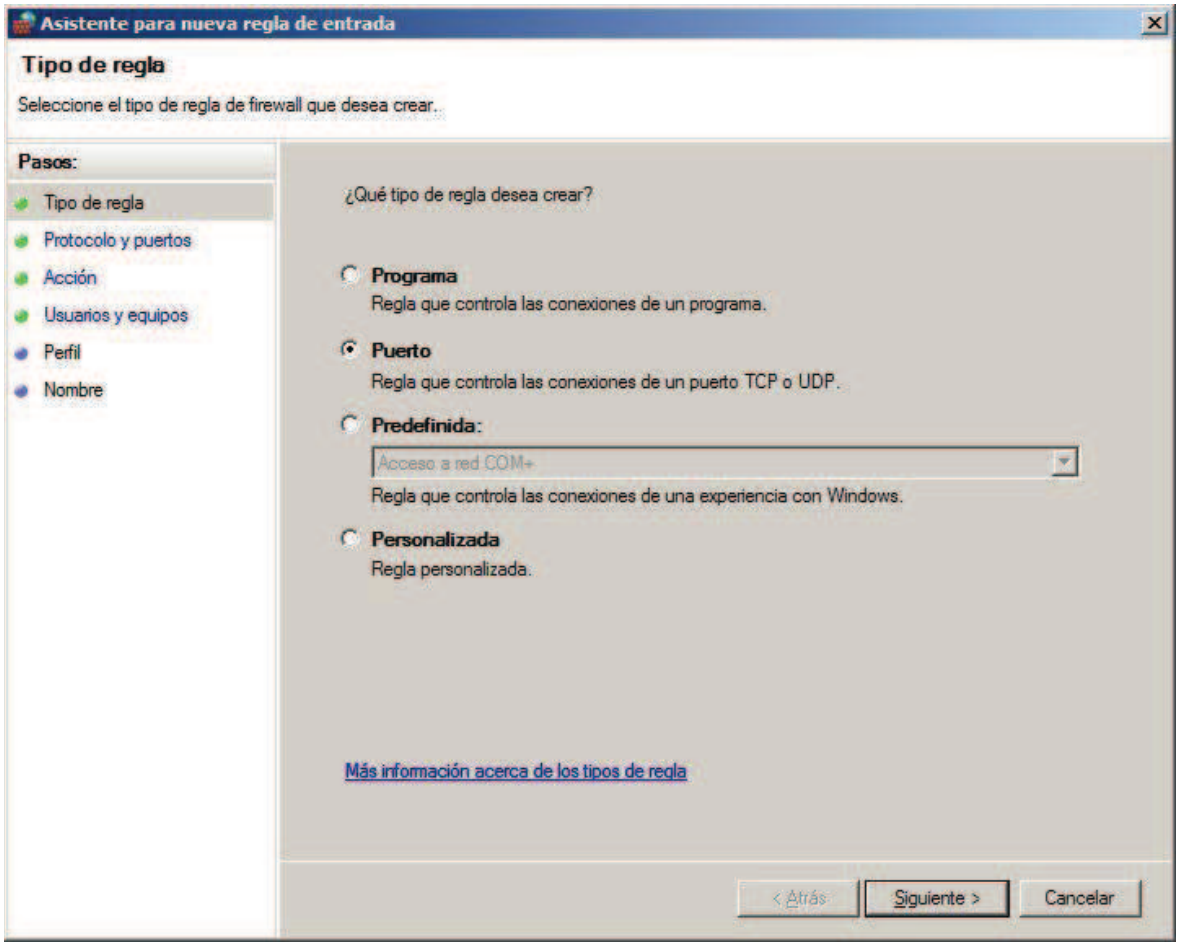

Figura 3.67 Aislamiento de servidores, paso 10

En la ventana mostrada en la Figura 3.67 se seleccionó *Puerto* y se presionó *Siguiente*. En la página *Protocolos y puertos* se seleccionaron las opciones *TCP*, y los puertos específicos *80, 443* y *1433*, que son los de los servicios http, https, y sql respectivamente. En la página *Acción* se seleccionó la opción *Permitir la conexión si es segura*, *Requerir cifrado de conexiones*, e *Invalidar reglas de bloqueo* (Figura 3.68).

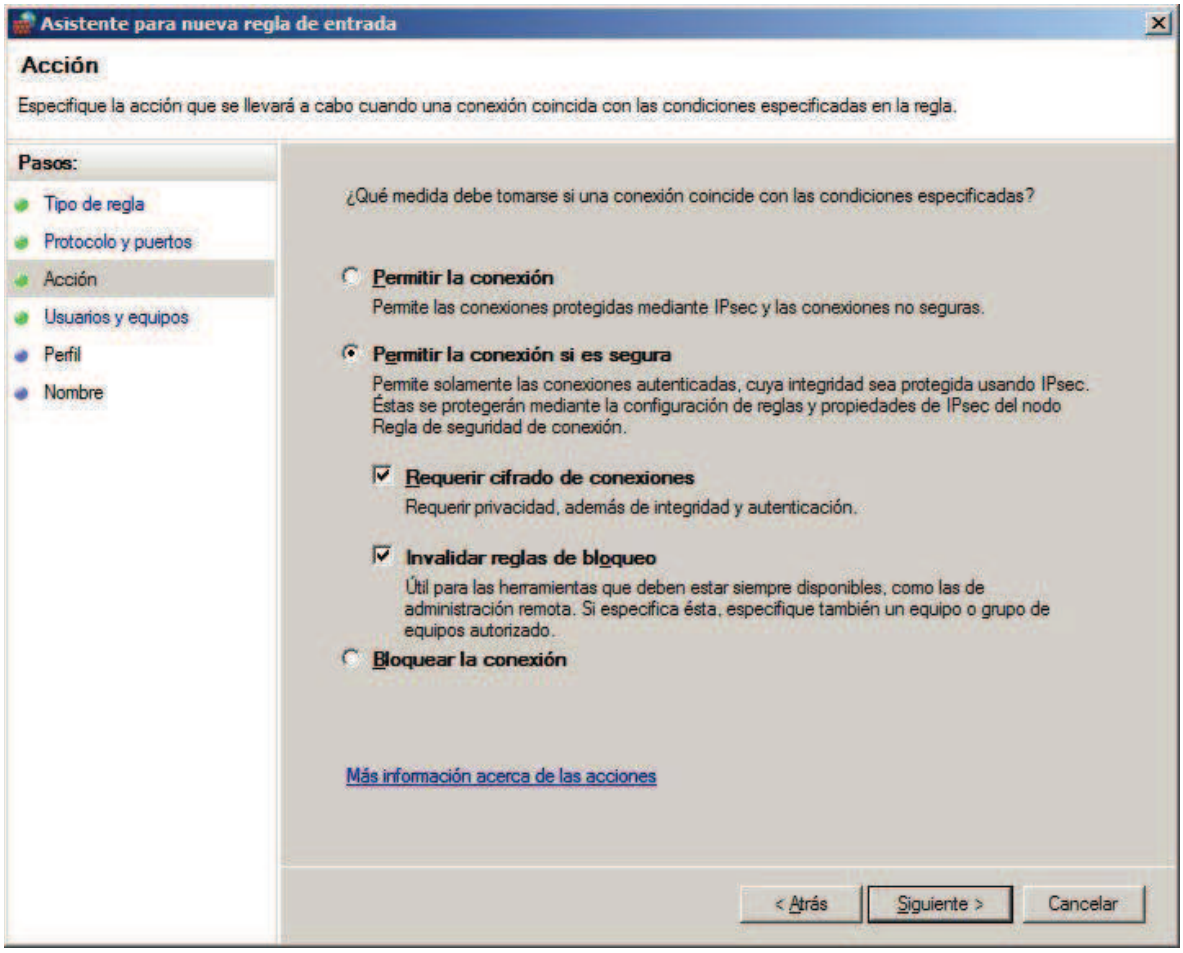

Figura 3.68 Aislamiento de servidores, paso 11

En la página *Usuarios y equipos*, se seleccionó la opción *Solo permitir conexiones de estos equipos,* se dio clic en el botón *Agregar*, se escribió el nombre del equipo del servidor de la aplicación web, y se presionó el botón *Siguiente*. En la ventana *Perfil* se seleccionó solo la opción *Dominio* y en la página *Nombre* se escribió un nombre para la regla. Por último se dio clic en el botón *Finalizar*.

CAPÍTULO IV

4. PRUEBAS, RESULTADOS Y COSTO DEL PROTOTIPO

4.1 PRUEBAS Y RESULTADOS

A continuación se exponen unas tablas resumen de las pruebas realizadas; las primeras pruebas se refieren a la comunicación entre las capas del Sistema, en donde se tuvieron que realizar algunas configuraciones y creación de usuarios administradores que tengan los suficientes permisos para poder lograr esta comunicación, que solo se permite si se autentica a los mismos; y las demás pruebas fueron hechas para comprobar el buen funcionamiento de cada uno de los procesos que facilita el aplicativo; además en las tablas se indica los problemas que se tuvo, en qué componente se tuvo el inconveniente y cómo se pudo solucionar el error para su posterior validación.

Los campos de la tabla de resumen son los siguientes:

- Detalle Prueba: indica el nombre de la prueba realizada
- Nº Prueba: aumenta en valor cuando se produjo un error, se lo corrige y se vuelve a realizar la misma prueba.
- Datos Prueba: indica los parámetros que se usaron para realizar las pruebas.
- Resultado Prueba: muestra “OK” si la prueba fue exitosa, caso contrario, muestra el error que sucedió y en la siguiente prueba la solución de este, si existe otro error también se lo muestra y se tendrá que hacer las pruebas necesarias hasta que el resultado de la prueba sea satisfactorio.

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Conexión entre capas	1	Config.	VS	SW, SrvW	Web.config	Config	Tramas	OK
Conexión entre capas	1	Config.	VS, SQL	SrvW, BD	Web.config, Archivo de Conf. de SQL.	Config	Tramas, Cadena de Conexión	Error: Permiso Denegado
Conexión entre capas	2	Config.	VS, SQL	SrvW, BD	Web.config, Archivo de Conf. de SQL.	Config	Tramas, Cadena de Conexión, credenciales de Usuario.	Creación de Usuario con Permisos, OK.
Autenticación de Usuario	1	C	VS	SW	Default.aspx.cs	PW	idUsuario	OK
Autenticación de Usuario	1	C	VS	SrvW	ServicioComun.asmx.cs	Srv	idUsuario	OK
Autenticación de Usuario	1	C	SQL	BD	Spms_autenticar_usuario.sql	SP	idUsuario	OK
Formación del Menú	1	C	VS	SW	Menu.aspx.cs, Site.Master.cs	PW	roles	OK
Creación de Usuario	1	I	VS	SW	Ingresos.aspx.cs	PW	idUsuario, Dirección Fecha Nac., Teléfono, Cel,E-mail	OK

Tabla 4.1 Pruebas y Resultados del Sistema. Parte I

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Creación de Usuario	1	I	VS	SrvW	ServicioAdministrador.asmx.cs	Srv	Datos de Usuario	OK
Creación de Usuario	1	I	SQL	BD	Spms_ingresar_usuario.sql	SP	Datos de Usuario	OK
Creación de Usuario con Rol Biblioteca	1	I	VS	SW	Ingresos.aspx.cs	PW	Edificio	OK
Creación de Usuario con Rol Biblioteca	1	I	SQL	BD	Spms_ingresar_biblioteca.sql	SP	IdUsuario, Edificio	OK
Creación de Usuario con Rol Doctor	1	I	VS	SW	Ingresos.aspx.cs	PW	Especialidad, Oficina	OK
Creación de Usuario con Rol Doctor	1	I	SQL	BD	Spms_ingresar_doctor.sql	SP	IdUsuario, Especialidad, Oficina	OK
Creación de Usuario con Rol Estudiante	1	I	VS	SW	Ingresos.aspx.cs	PW	Huella Dactilar, Carrera, Facultad	Error: No se reconoce el tipo de dato.

Tabla 4.2 Pruebas y Resultados del Sistema. Parte II

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Creación de Usuario con Rol Estudiante	2	I	VS	SW	Ingresos.aspx.cs	PW	Huella Dactilar, Carrera, Facultad	Conversión a tipo de dato string. Error: No se reconoce el lector.
Creación de Usuario con Rol Estudiante	3	I	VS	SW	Ingresos.aspx.cs	PW	Huella Dactilar, Carrera, Facultad	Reinstalación del driver. OK.
Creación de Usuario con Rol Estudiante	1	I	SQL	BD	Spms_ingresar_estudiante.sql	SP	IdUsuario, Huella Dactilar, Carrera, Facultad	Error: Tamaño del campo huella.
Creación de Usuario con Rol Estudiante	2	I	SQL	BD	Spms_ingresar_estudiante.sql	SP	IdUsuario, Huella, Carrera, Facultad	Varchar(MAX). OK.
Modificación de Usuario	1	M	VS	SW	Modificar.aspx.cs	PW	idUsuario, Dirección, Fecha Nac., Teléfono, Cel, E-mail.	Error: Fecha no es correcta.
Modificación de Usuario	2	M	VS	SW	Modificar.aspx.cs	PW	idUsuario, Dirección, Fecha Nac., Teléfono, Cel, E-mail.	Corregir Formato de Fecha en el RegEdit. OK.

Tabla 4.3 Pruebas y Resultados del Sistema. Parte III

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Modificación de Usuario	1	M	SQL	BD	Spms_actualizar_usuario.sql	SP	idUsuario, Dirección Fecha Nac., Teléfono, Cel,E-mail	OK.
Modificación de Usuario con rol Biblioteca	1	M	VS	SW	Modificar.aspx.cs	PW	Edificio	OK.
Modificación de Usuario con rol Biblioteca	1	M	SQL	BD	Spms_actualizar_biblioteca.sql	SP	idUsuario, Edificio	OK.
Modificación de Usuario con rol Doctor	1	M	VS	SW	Modificar.aspx.cs	PW	Especialidad, Oficina	OK.
Modificación de Usuario con rol Estudiante	1	M	VS	SW	Modificar.aspx.cs	PW	Huella Dactilar, Facultad, Carrera.	OK.
Modificación de Usuario con rol Doctor	1	M	SQL	BD	Spms_actualizar_doctor.sql	SP	idUsuario, Especialidad, Oficina	OK.
Modificación de Usuario con rol Estudiante	1	M	SQL	BD	Spms_actualizar_estudiante.sql	SP	idUsuario, Huella Dactilar, Facultad, Carrera.	OK.

Tabla 4.4 Pruebas y Resultados del Sistema. Parte IV

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Ingreso de Libro	1	I	VS	SW	Ingresos.aspx.cs	PW	Titulo, Autor, Edición, Año de Publicación	OK
Ingreso de Libro	1	I	SQL	BD	Spms_ingresar_libro.sql	SP	Titulo, Autor, Edición, Año de Publicación	OK
Ingreso de Existencia	1	I	VS	SW	Ingresos.aspx.cs	PW	Cantidad, Estantería	OK
Ingreso de Existencia	1	I	SQL	BD	Spms_ingresar_existencia.sql	SP	idLibro, Cantidad, Estantería	OK
Modificación de Libro	1	M	VS	SW	Modificar.aspx.cs	PW	Titulo, Autor, Edición, Año de Publicación	OK
Modificación de Libro	1	M	SQL	BD	Spms_actualizar_libro.sql	SP	Titulo, Autor, Edición, Año de Publicación	OK

Tabla 4.5 Pruebas y Resultados del Sistema. Parte V

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Modificación de Existencia	1	M	VS	SW	Modificar.aspx.cs	PW	Cantidad, Estantería	OK.
Modificación de Existencia	1	M	SQL	BD	Spms_actualizar_existencia.sql	SP	idLibro, Cantidad, Estantería	OK.
Consulta de Libros (Autor o Titulo)	1	C	VS	SW	CatalogoLibros.aspx.cs	PW	Titulo o Autor	Error: Numero de Parámetros
Consulta de Libros (Autor o Titulo)	2	C	VS	SW	CatalogoLibros.aspx.cs	PW	Titulo o Autor	Corrección del Número de Parámetros Error: Ítems repetidos.
Consulta de Libros (Autor o Titulo)	3	C	VS	SW	CatalogoLibros.aspx.cs	PW	Titulo o Autor, idBiblioteca	Aumento de Campo. OK.
Consulta de Libros (Autor o Titulo)	1	C	VS	SrvW	ServicioBiblioteca.asmx.cs	Srv	Titulo o Autor, idBiblioteca	OK.
Consulta de Libros (Autor o Titulo)	1	C	SQL	BD	Spms_consultar_catalogo_libros.sql	SP	Titulo o Autor, idBiblioteca	OK
Préstamo de un Libro	1	I	VS	SW	CatalogoLibros.aspx.cs	PW	idEstudiante, fechaActual, fechaFin, Huella Dactilar.	Error: Formato de fecha actual.

Tabla 4.6 Pruebas y Resultados del Sistema. Parte VI

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Préstamo de un Libro	2	I	VS	SW	CatalogoLibros.aspx.cs	PW	idEstudiante, fechaActual, fechaFin, Huella Dactilar.	Corrección de Formato de fecha actual y final. OK.
Préstamo de un Libro	1	I	VS	SrvW	ServicioBiblioteca.asmx.cs	Srv	idEstudiante, fechaActual, fechaFin, Huella Dactilar	Error: Proceso almacenado no encontrado.
Préstamo de un Libro	2	I	VS	SrvW	ServicioBiblioteca.asmx.cs	Srv	idEstudiante, fechaActual, fechaFin, Huella Dactilar.	Ejecución del Proceso almacenado OK.
Préstamo de un Libro	1	I	SQL	BD	Spms_ingresar_prestamo.sql	SP	idEstudiante, fechaActual, fechaFin.	OK.
Consulta de Saldo	1	C	VS	SW	CobroXItems.aspx.cs	PW	Valor Total, idEstudiante,	OK.
Consulta de Saldo	1	C	VS	SrvW	CobroXItems.aspx.cs	Srv	Valor Total, idEstudiante,	OK.
Consulta de Saldo	1	C	SQL	BD	Spms_consultar_saldo.sql	SP	Valor Total, idEstudiante,	OK.
Cobro de Multa(s)	1	I,M	VS	SW	CobroXItems.aspx.cs	PW	Valor Total, idEstudiante, Huella Dactilar	Error: No se puede transformar int a decimal.

Tabla 4.7 Pruebas y Resultados del Sistema. Parte VII

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Cobro de Multa(s)	2	I,M	VS	SW	CobroXItems.aspx.cs	PW	Valor Total, idEstudiante, Huella Dactilar	Arreglo del tipo Decimal. Inserción de Transacción Modificación del Préstamo. OK.
Cobro de Multa(s)	1	I,M	VS	SrvW	ServicioBiblioteca.asmx.cs, ServicioFuncionarios.asmx.cs	Srv	Valor Total, idEstudiante, Huella Dactilar	OK.
Cobro de Multa(s)	1	I,M	SQL	BD	Spms_ingresar_transaccion.sql, Spms_pago_efectivo_multa.sql	SP	Valor Total, idEstudiante, Huella Dactilar	OK.
Cobro en Efectivo de Multa(s)	1	M	VS	SW	CobroXItems.aspx.cs,	PW	Valor Total, idEstudiante, Huella Dactilar	OK.
Cobro en Efectivo de Multa(s)	1	M	VS	SrvW	ServicioBiblioteca.asmx.cs	Srv	Valor Total, idEstudiante, Huella Dactilar	OK.
Cobro en Efectivo de Multa(s)	1	M	SQL	BD	Spms_pago_efectivo_multa.sql	SP	Valor Total, idEstudiante, Huella Dactilar	OK.
Consulta Datos Personales	1	C	VS	SW	DatosPersonales.aspx.cs	PW	idUsuario	OK.
Consulta Datos Personales	1	C	VS	SrvW	ServicioComun.asmx.cs	Srv	idUsuario	OK.

Tabla 4.8 Pruebas y Resultados del Sistema. Parte VIII

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Consulta Datos Personales	1	C	SQL	BD	Spms_consultar_datos_personales.sql	SP	idUsuario	OK.
Consultar Ultimas 10 Transacciones	1	C	VS	SW	ReporteTransacciones.aspx.cs	PW	idUsuario	OK.
Consultar Ultimas 10 Transacciones	1	C	VS	SrvW	ServicioComun.asmx.cs	Srv	idUsuario	OK.
Consultar Ultimas 10 Transacciones	1	C	SQL	BD	spms_consultar_ultimas_10_transacciones.sql	SP	idUsuario	OK.
Consultar Transacciones de un rango de Fechas.	1	C	VS	SW	ReporteTransacciones.aspx.cs	PW	idUsuario, FechaInicial, FechaInicial,	Error: la fechafinal debe ser mayor a la fechaInicial
Consultar Transacciones de un rango de Fechas.	1	C	VS	SW	ReporteTransacciones.aspx.cs	PW	idUsuario, FechaInicial, FechaInicial,	Rango de fechas corregido. Error: las fechas deben ser anteriores a la actual.
Consultar Transacciones de un rango de Fechas.	1	C	VS	SW	ReporteTransacciones.aspx.cs	PW	idUsuario, FechaInicial, FechaInicial,	Corregido el rango de fechas 2do error. OK.
Consultar Transacciones de un rango de Fechas.	1	C	VS	SrvW	ServicioComun.asmx.cs	Srv	idUsuario	OK.

Tabla 4.9 Pruebas y Resultados del Sistema. Parte IX

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Consultar Transacciones de un rango de Fechas.	1	C	SQL	BD	spms_consultar_transacciones_x_fecha.sql	SP	idUsuario	OK.
Ingresar Depósito	1	I	VS	SW	TransaccionesFinanciero.aspx.cs	PW	idUsuario, idEstudiante, Monto, fecha.	OK.
Ingresar Depósito	1	I	VS	SrvW	ServicioFinanciero.asmx.cs	Srv	idUsuario, idEstudiante, Monto, fecha.	OK.
Ingresar Depósito	1	I	SQL	BD	spms_ingresar_deposito.sql	SP	idUsuario, idEstudiante, Monto, fecha.	OK.
Ingresar Retiro	1	I	VS	SW	TransaccionesFinanciero.aspx.cs	PW	idUsuario, idEstudiante, Monto, fecha.	OK.
Ingresar Retiro	1	I	VS	SrvW	ServicioFinanciero.asmx.cs	Srv	idUsuario, idEstudiante, Monto, fecha.	OK.
Ingresar Retiro	1	I	SQL	BD	spms_ingresar_retiro.sql	SP	idUsuario, idEstudiante, Monto, fecha.	OK.
Consultar Mensaje	1	C	VS	SW	Menu.aspx.cs	PW	idUsuario	OK.
Consultar Mensaje	1	C	VS	SrvW	ServicioFuncionarios.asmx.cs	Srv	idUsuario	OK.
Consultar Mensaje	1	C	SQL	BD	spms_consultar_mensajes.sql	SP	idUsuario	OK.

Tabla 4.10 Pruebas y Resultados del Sistema. Parte X

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Ingresar Mensaje	1	I	VS	SW	Todas las PW usadas por los funcionarios de cada servicio	PW	idUsuario, Mensaje, fechaVigencia	Error: El Mensaje no se pudo ingresar
Ingresar Mensaje	1	I	VS	SrvW	ServicioFuncionarios.asmx.cs	Srv	idUsuario, Mensaje, fechaVigencia	OK.
Ingresar Mensaje	1	I	SQL	BD	spms_ingresar_mensaje.sql	SP	idUsuario, Mensaje, fechaVigencia	OK.
Insertar Historia Clinica	1	I	VS	SW	Ingresos.aspx.cs	PW	idUsuario	OK.
Insertar Historia Clinica	1	I	VS	SrvW	ServicioAdministrador.asmx.cs	Srv	idUsuario	OK.
Insertar Historia Clinica	1	I	SQL	BD	Spms_insertar_historia_clinica.sql	SP	idUsuario	OK.
Actualizar Historia Clinica	1	M	VS	SW	HistoriaClinica.aspx.cs	PW	idUsuario, TipoSangre, Alergias, Observación	OK.
Actualizar Historia Clinica	1	M	VS	SrvW	ServicioDoctor.asmx.cs	Srv	idUsuario, TipoSangre, Alergias, Observación	OK.

Tabla 4.11 Pruebas y Resultados del Sistema. Parte XI

Detalle Prueba	N° Prueba	Acción	Software	Capa	Nombre de Archivo	Tipo Archivo	Datos Prueba	Resultado Prueba
Actualizar Historia Clinica	1	M	SQL	BD	Spms_actualizar_historia_clinica.sql	SP	idUsuario, TipoSangre, Alergias, Observación	OK.
Reservar Cita Medica	1	I	VS	SW	CitasMedicas.aspx.cs	PW	idUsuario, Especialidad, Fecha y Hora	Error: No se puede ingresar una fecha anterior o actual.
Reservar Cita Medica	2	I	VS	SW	CitasMedicas.aspx.cs	PW	idUsuario, Especialidad, Fecha y Hora	Corregido error fecha. OK.
Reservar Cita Medica	1	I	VS	SrvW	ServicioDoctor.asmx.cs	Srv	idUsuario, Especialidad, Fecha y Hora	OK.
Reservar Cita Medica	1	I	SQL	BD	Spms_ingresar_cita_medica.sql	SP	idUsuario, Especialidad, Fecha y Hora	OK.
Cancelar Cita Medica	1	M	VS	SW	CitasMedicas.aspx.cs	PW	idUsuario, Especialidad, Fecha y Hora	OK.
Cancelar Cita Medica	1	M	VS	SrvW	ServicioDoctor.asmx.cs	Srv	idUsuario, Especialidad, Fecha y Hora	OK.
Cancelar Cita Medica	1	M	SQL	BD	Spms_cancelar_cita.sql	SP	idUsuario, Especialidad, Fecha y Hora	OK.

Tabla 4.12 Pruebas y Resultados del Sistema. Parte XII

A continuación se describen las abreviaturas utilizadas en los campos de las tablas anteriores.

Acción:

Config. = Configuración

C = Consulta.

I = Ingreso.

M = Modificación.

Software:

VS = Visual Studio .Net 2010

SQL = SQL Server 2008 r2

Capa:

SW = Sitio Web.

SrvW = Servicio Web.

BD = Base de Datos.

Tipo Archivo:

Config = Configuración.

PW = Pagina Web.

Srv = Servicio.

SP = Store Procedure (Procedimiento Almacenado).

Con las pruebas realizadas se observó que el sistema cumple con los objetivos planteados en lo referente a mejoras de tiempo y seguridad en los servicios que este presta, pero ya que como es un prototipo y además es orientado al estudiante en algunos casos se debería realizar mejoras para que se cumpla totalmente también para los funcionarios porque hasta el momento se tiene una igualdad de tiempo para algunos servicios.

Aun así se tiene una ventaja al final del día porque se puede generar el reporte de todas las transacciones realizadas y por tanto saber los totales reales que se tiene en la cuenta del sistema, se vuelve además más cómodo el poder realizar una cita médica en el horario de conveniencia sin tener que esperar varios turnos y sin tener la total seguridad de que te van a atender, también se puede hacer la consulta de si existe un libro en alguna biblioteca sin tener que estar físicamente.

4.1.1 PRUEBAS DE SEGURIDAD

Para verificar el funcionamiento del protocolo de seguridad entre el cliente y el servidor web, se utilizó el programa Wireshark con el cual se pudieron realizar capturas de los paquetes enviados en el proceso de autenticación en la página de ingreso al sistema. Primero se capturaron paquetes enviados solamente con protocolo http.

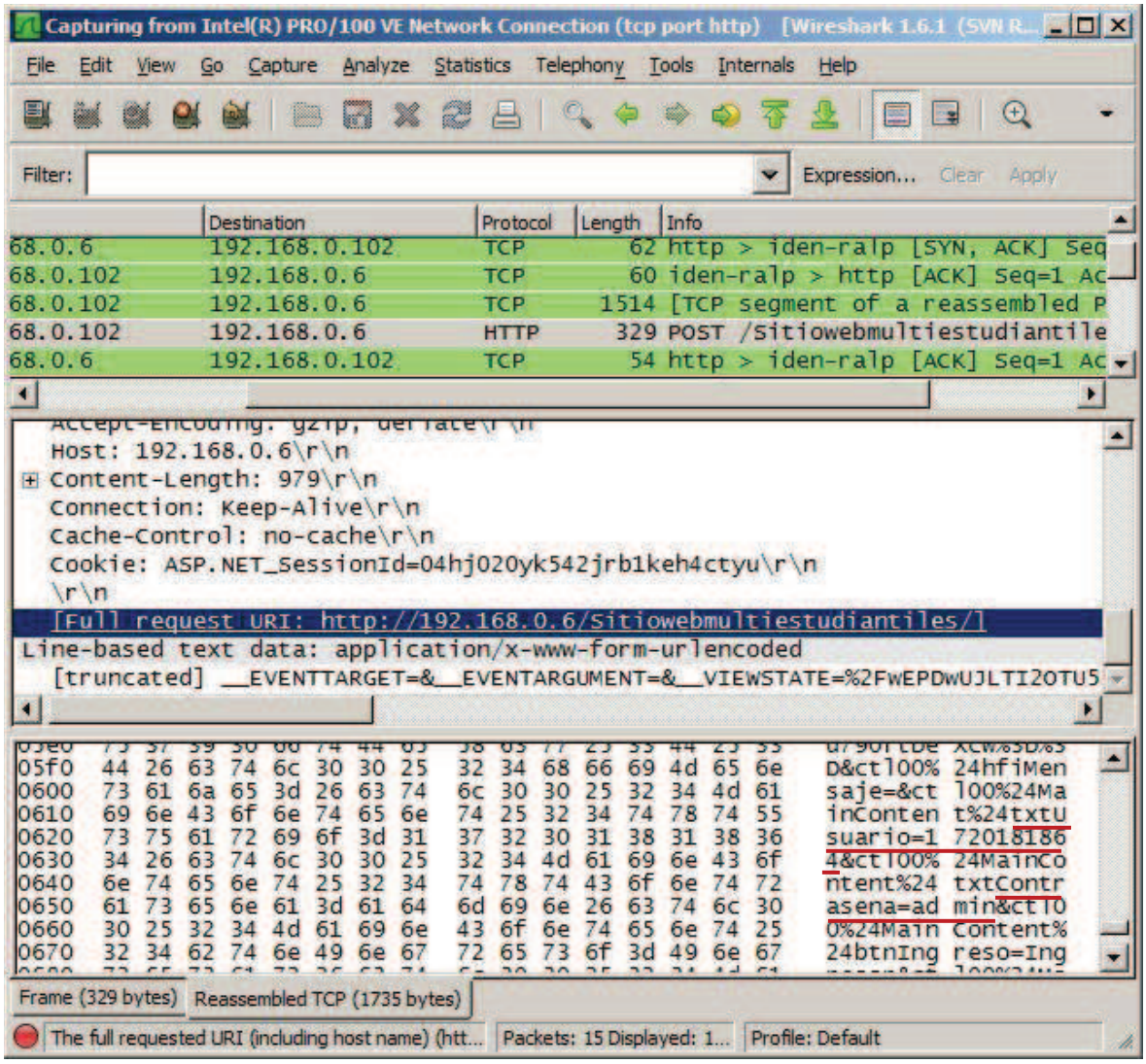

Figura 4.1 Captura de un paquete con protocolo HTTP

Como se puede observar en la Figura 4.1, toda la información viaja por la red en texto claro, por lo que cualquier persona con un programa simple de captura de paquetes puede tener acceso a información sensible como en este caso son el nombre de usuario y la contraseña.

Luego de eso se realizó la captura de los mensajes enviados desde de la misma página, pero esta vez se utilizó el protocolo HTTPS, obteniéndose los resultados mostrados en la Figura 4.2.

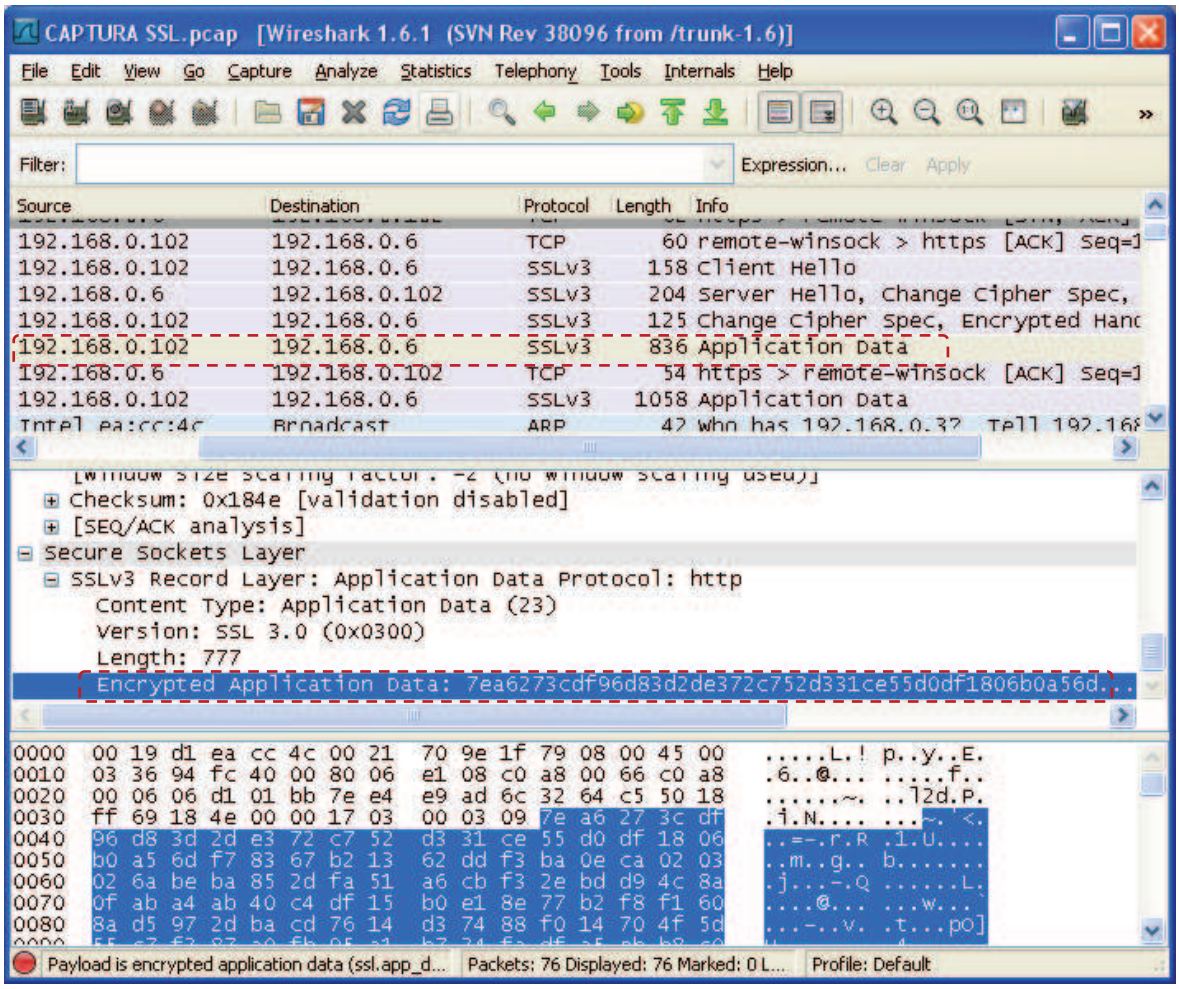

Figura 4.2 Captura realizada a paquetes con protocolo HTTPS

En esta captura se puede ver que los datos viajan con la información encriptada, con lo cual un programa de simple captura de paquetes solo puede dar información general de los mismos, como el tipo de protocolo utilizado, su versión, etc. Con esto se puedo comprobar la seguridad de la información intercambiada entre el cliente y el servidor web.

Por otra parte la seguridad en la comunicación entre los servidores se comprobó al momento de su configuración, como se muestra en la Figura 3.64 donde se ve la utilización del protocolo Kerberos, junto con los algoritmos AES 128 y SHA1.

4.2 COSTO DEL PROTOTIPO

La Tabla 4.13 muestra el número de bytes que ocupan las variables en *Microsoft SQLServer*: [PW29]

Tipo de Dato	Mínimo	Máximo	Almacena
Bigint	-2^63	2^63-1	8 bytes
Int	-2,147,483,648	2,147,483,647	4 bytes
Smallint	-32,768	32,767	2 bytes
Tynint	0	255	1 byte
Bit	0	1	1 byte
Decimal	-10^38+1	10^38+1	Precisión 1-9 = 5 bytes, Precisión 10-19 = 9 bytes, Precisión 20-28 = 13 bytes, Precisión 29-38 = 17 bytes.
Money	-2^63 / 10000	2^63-1 / 10000	8 bytes
Smallmoney	-214,748.3648	214,748.3647	4 bytes
Float	-1.79E+308	1.79E+308	Precisión < 25 = 4 bytes, precisión > 25 = 8 bytes.
Real	-3.40E+38	3.40E+38	4 bytes
Datetime	1753-01-01 00:00:00.000	9999-12-31 23:59:59.997	8 bytes
Datetime2	0001-01-01 00:00:00.0000000	9999-12-31 23:59:59.9999999	Precisión 1-2 = 6 bytes, precisión 3-4 =7 bytes, precisión 5-7 = 8 bytes.
Datetimeoffset	0001-01-01 00:00:00.0000000 – 14:00	9999-12-31 23:59:59.9999999 + 14:00	Precisión 1-2 = 8 bytes, precisión 3-4 =9 bytes, precisión 5-7 = 10 bytes.
Char	0 chars	8000 chars	# bytes = # caracteres previamente definidos.
Varchar	0 chars	8000 chars	# bytes = # caracteres.

Tabla 4.13 Características de Tipos de Datos usados en SQLServer 2008 [PW29]

Teniendo en cuenta los datos de la Tabla 4.13 y de los expuestos en el capítulo 2 acerca de la definición de tablas (Ver Anexo C), se ha llegado a obtener la Tabla 4.14 en donde se expone el crecimiento que va a tener la base de datos a través del tiempo, y por lo tanto del requisito de capacidad de Disco Duro necesitado en el servidor de Datos para que el Sistema pueda crecer sin problemas.

Los Resultados mostrados por la tabla dictan que se tendría un peso inicial total de la base de datos de 18,49 MB y con un crecimiento diario de 1,3 MB, lo que daría un crecimiento anual de 340,25 MB ya que solo se cuenta los días hábiles (261).

La Base de Datos necesita de 10 GB de disco duro por razones de crecimiento no contempladas y de estimación expuestos.

Además con esta estimación de la necesidad de capacidad de la base de datos también se puede evidenciar el número de llamadas que va a tener el sistema, el total de la columna “Nº de Peticiones Diarias” de la Tabla 4.14 muestra el número de llamadas que va a recibir el sistema (65380); dado que en la tabla no se expone el número de llamadas de consulta al sistema sino solamente las que terminan en asiento de registro, este valor se puede duplicar (130760).

Teniendo en cuenta también que existen temporadas donde habrá mayor demanda en los servicios prestados por el sistema se tendrá un alza de un 50% en las peticiones dándonos 196140, sin olvidarnos del 20% de factor de error que siempre se debe tener en cuenta tenemos un total de 235368 peticiones que deberá soportar el sistema en un día cuando lo exijan a su máxima capacidad.

Tomando en cuenta que las páginas pesan alrededor de 30 Kb y en el caso de una carga de 3000 estudiantes realizando peticiones al mismo tiempo se debería tener un ancho de banda aproximado de 12 Mbps, calculado con la fórmula “(((XX Kb *8) /60) * 3000) /1024”.

Para tener conocimiento de cuantos recursos se utilizan en hardware se hizo uso del programa “Everest”, el cual indica que en promedio cada usuario ocupa 1 MB de memoria RAM, por tanto si se tiene 3000 usuarios conectados simultáneamente se ocuparán 3 GB de Memoria por lo tanto se justifica los 4 GB

de memoria Mínima requerida, además se pudo constatar que cada petición es atendida en menos de medio segundo, por cuanto con 3000 peticiones al mismo tiempo se ocuparía 1GHz es decir la mitad del procesador estimado mínimo, pero esto solamente para peticiones estrictamente del sistema y no las demás que necesita el servidor para ponerse en funcionamiento.

Por todas las razones expuestas se llega a obtener las especificaciones que están indicadas en la Tabla 4.15 los requerimientos de software compatible con los requerimientos de hardware que se encuentran en la Tabla 4.16 y que ayudan a que el sistema pueda retornar una respuesta efectiva.

En la Figura 4.3 se muestra el servidor *HP ML150 G6 E5504 HP SAS/SATA US Svr 466132-001* que cumple con las características mínimas de inicio del Sistema y tiene un costo de $1349.95 + IVA; como los servidores que se necesita requieren de mejores características se cambia los componentes y aumentan los precios pero mejorando la relación calidad vs costo. [PW30]

Figura 4.3 Equipo Servidor [PW30]

Tabla	Peso x Registro (bytes)	N° Registros	Peso Inicial x Tabla (bytes)	Nº Peticiones Diarias	Peso Crecimiento Diario X Tabla (bytes)
Asiento - Transacción	19	10000	190000	40000	760000
Biblioteca	38	20	760	0	0
CitaMedica	322	0	0	100	32200
Doctor	45	10	450	0	0
Estudiante	3176	5000	15880000	0	0
Existencia	22	6000	132000	40	880
HistoriaClinica	164	5000	820000	0	0
Libro	113	6000	678000	40	4520
Mensaje	322	0	0	200	64400
PrestamoLibro	33	0	0	5000	165000
Transaccion	17	5000	85000	20000	340000
Usuario	315	5110	1609650	0	0
		Total:	19395860	65380	1367000
			Peso Crecimiento Anual X Base(bytes)		356787000

Tabla 4.14 Estimación del Tamaño de la Base de Datos

SOFTWARE					
Servidor	Sistema Operativo		Programa de Desarrollo		Costo (USD)
	Programa	Costo X Maquina(USD)	Programa	Costo X Maquina(USD)	
Sitio Web	Windows Server 2008	469	Visual Studio .NET 2010	713	1182
Servicios Web	Windows Server 2008	469	Visual Studio .NET 2010	713	1182
Base de Datos	Windows Server 2008	3000	SQL Server 2008 r2	885	3885
				Certificado Digital:	1100
				Total sin IVA:	7349
				IVA:	881.88
				Total:	8230.88

Tabla 4.15 Costo del Software del Sistema [PW31], [PW32], [PW33]

HARDWARE								
Servidor	Disco Duro (GB)		Memoria RAM (GB)		Procesador (GHz)		Costo (USD)	
	Mínimo	Recomendado	Mínimo	Recomendado	Mínimo	Recomendado	Mínimo	Recomendado
Sitio Web	40	60	4	8	2.0	2.6	1348.95	1551.29
Servicios Web	40	60	4	8	2.0	2.8	1348.95	1605.25
Base de Datos	60	80	4	8	2.4	3.0	1565.88	1957.35
						Total sin IVA:	4263.78	5113.89
						IVA:	511.65	613.67
						Total:	4775.43	5727.56

Tabla 4.16 Costo del Harware del Sistema [PW30]

Costo de Desarrollo, Implementación y Mantenimiento del Sistema					
	Salario Nominal (USD)	**IESS (USD)**	**Décimo Tercero (USD)**	**Décimo Cuarto (USD)**	**Salario Anual a Recibir (USD)**
Costo de Desarrollo e Implementación del Sistema					
Desarrollador 1	1000,00	93,50	906,50	318,00	12102,50
Desarrollador 2	1000,00	93,50	906,50	318,00	12102,50
				Total (USD):	24205,00
Costo de Mantenimiento					
Administrador	900	84,15	815,85	318,00	10924,05
				Total (USD):	10924,05

Tabla 4.17 Costo de Desarrollo y Mantenimiento del Sistema [PW34], [PW35]

En la Tabla 4.15 se exponen los costos que tendrá el sistema en temas de software, y que, anualmente se deben cancelar a causa de que el sistema se lo realizó con programas licenciados, asegurando con esto la asistencia por parte del proveedor (Microsoft).

Adicionalmente en la Tabla 4.16 se muestra los datos estimados de costo de Hardware, tomando en cuenta valores mínimos y recomendados; tomando en cuenta los últimos para el cálculo total de costos del sistema.

La Tabla 4.17 muestra los valores a cancelarse a causa de sueldos y demás beneficios de los desarrolladores que iniciaron la creación del sistema, el Administrador que será el que brinde el soporte inmediato al Sistema y de los desarrolladores que seguirán añadiéndole funcionalidades y mejoras al proyecto.

Por último la Tabla 4.18 totaliza los costos que tendrá el sistema en sus fases de desarrollo e implementación, así como también, en la fase de mantenimiento, tomando en cuenta como adicional los equipos lectores de huellas que se necesitan por cada funcionario del servicio (110) y su costo con IVA (USD 130).

Costos Totales del Sistema		
	Costo Inicial (USD)	**Costo Anual (USD)**
Hardware:	5727,56	0
Software:	8230,88	8230,88
Lectores de Huella (110X130 c/u):	14300,00	0
Desarrollo e Implementación:	24205,00	0
Mantenimiento:	0	10924,05
TOTAL:	52463,44	19154,93

Tabla 4.18 Resumen de Costos

4.3 BENEFICIOS TANGIBLES E INTANGIBLES

4.3.1 BENEFICIOS TANGIBLES

Son aquellos que percibirán la Institución Educativa Universitaria y los servicios de la misma al implementar el Sistema. A continuación se presentan los más relevantes para una Institución Educativa.

- Los servicios verán incrementadas sus ganancias (excepto en el caso del Doctor), ya que, gracias al manejo de dinero virtual no se necesita perder dinero cuando no se tiene la cantidad exacta del vuelto.
- Reducción de costos en equipos de red ya que los equipos servidores del presente sistema se podrán usar por más de 10 años.

4.3.2 BENEFICIOS INTANGIBLES

A estos beneficios no se los puede medir de una manera objetiva, pero representan gran importancia para una empresa. Los de mayor relevancia son:

- Ya que el sistema consta solo de los servicios más importantes se puede dar mayores servicios por lo que se dice que tiene una gran escalabilidad y además no significaría una inversión mayor.
- La información se la tiene al instante.
- El sistema da facilidades de uso del dinero al estudiante, por lo que los servicios con los que cuenta el sistema y los que se agreguen en el futuro se beneficiarán grandemente, ya que el estudiante no necesita de ninguna identificación, salvo el caso del Servicio Financiero, siendo suficiente su huella.
- Atención ordenada y de calidad es lo que obtendrá el estudiante al usar el sistema y esto a su vez dará como resultado mayor confianza en el uso de los servicios.

CAPÍTULO V

5. CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

- En este proyecto se hizo uso de la Biometría ya que brinda de seguridad al sistema en lo referente a autenticación de un usuario, debido a que los rasgos característicos de una persona son únicos en el mundo y casi no cambian con el pasar del tiempo, por ejemplo los surcos de los dedos (crestas y valles papilares) son únicos y se regeneran si se ha sufrido una herida leve. Además de este campo existen otros en los que se puede dar aplicabilidad de la biometría enlazada con la tecnología, por ello se puede deducir que todavía se tiene una amplia gama de desarrollo y creación de nuevas tecnologías.

- La creación de varias capas para el funcionamiento de este Sistema Informático ayuda en gran manera, ya que se tiene independencia de capas, lo que permite la distribución de las mismas en distintos equipos, es decir, se convierte en un Sistema Distribuido que puede estar en ambiente de escritorio tanto como en ambiente Web; además de que en cada capa se pueden hacer simples modificaciones en el código fuente para obtener grandes cambios que mejorarán el funcionamiento en el sistema general.

- Con la creación del presente Sistema Web se pretende conseguir dos objetivos principales: el primero es que un estudiante no tenga la necesidad de llevar mucho dinero en el bolsillo y lo tenga más seguro en una cuenta virtual de la que solo se podrá debitar mediante la lectura y verificación de su huella digital; el segundo, además de satisfacer las necesidades que se creyó más importantes en un campus universitario como son los servicios de Biblioteca, Citas Médicas, Copiadora y Bar que son los que mayormente hace uso un estudiante, es tener al estudiante y a la institución educativa en el actual conocimiento y manejo de Sistemas

Informáticos que contengan tecnologías actuales y novedosas, para no quedar rezagados tecnológicamente.

- Los dispositivos de lectura de Huella Digital no están tan difundidos en nuestro medio; existen mayormente aplicaciones de escritorio que los usan, pero el presente proyecto tuvo como complicación que por su naturaleza de aplicación web no había tanto material recursivo como para aplicaciones de escritorio, por lo que, se tuvo que hacer uso del lenguaje de programación *Java Script* para crear una interfaz de comunicación entre el cliente y el servidor de este dispositivo, además del *SDK* que viene con el dispositivo, el cuál expone los métodos con los que trabaja el lector de huellas *Hamster Plus*.

- Los Servicios Web son una parte importante del proyecto ya que son la Interfaz de comunicación entre la Capa de Datos y la Capa de Enlace con el Usuario, además estos servicios se los puede reutilizar en otros proyectos que cumplan con las firmas de los métodos que estos contienen, y si se desea modificar o agregar métodos se lo puede hacer más fácilmente que cuando se lo hace desde cero ya que están apropiadamente comentadas sus funciones, por lo que, no se presentan mayores inconvenientes.

- En el Servidor Web al configurar el protocolo *SSL* se requiere de un generador de números aleatorios seguros desde el punto de vista del cifrado, por lo que debe tenerse cuidado en el diseño de los mismos y de sus claves iniciales. Se consideran aceptables aquellos usados en operaciones seguras de *hashing*, como MD5 o SHA, pero por supuesto no pueden dar más seguridad que el tamaño del generador de estado de número aleatorio (por ejemplo, los generadores basados en MD5 usualmente proveen estados de 128-bits), además las implementaciones son responsables de verificar la integridad de los certificados y deberían generalmente soportar mensajes de revocación de certificados. Los certificados deben ser verificados siempre por una CA (*Certificate Authority*-Autoridad de Certificación) para asegurar que sean firmados

correctamente; también *SSL* soporta un cierto rango de tamaños de claves y niveles de seguridad, incluyendo algunos que son prácticamente inseguros. Es probable que una implementación realizada correctamente no acepte cierto tipo de *Cipher Suites*. Por ejemplo, un cifrado de 40 bits puede ser roto fácilmente, por lo que implementaciones que requieran de mucha seguridad no deben permitir claves de 40 bits. Deben evitarse comunicaciones anónimas, pues se pueden realizar ataques del tipo *man-in-the-middle* (un hombre en el medio) y deben imponer límites mínimos y máximos en el tamaño de las claves.

- En el proyecto el protocolo *IPSec* ayuda a aislar los servidores del resto de equipos de la red ya que es un estándar de seguridad extraordinariamente potente y flexible. Su importancia reside en que aborda una carencia tradicional en el protocolo *IP*: la seguridad. Gracias a esto es posible hacer transacciones. Se puede decir que es la solución ideal para aquellos escenarios en que se requiera seguridad, independientemente de la aplicación, de modo que es una pieza esencial en la seguridad de las redes *IP*. En este momento se puede considerar que es una tecnología suficientemente madura para ser implantada en todos aquellos escenarios en los que la seguridad es un requisito prioritario como en el presente caso.

5.2 RECOMENDACIONES

- Se recomienda la investigación y el uso de las nuevas tecnologías ya que brindan mucha ayuda en especial por las mejoras de seguridad en la red, que permiten hacer más confiable un Sistema con el accionar de varios factores como los son los protocolos de seguridad en red, la criptografía de la información y por supuesto el manejo de dispositivos periféricos que ayudan de interfaz para la conversión del mundo análogo en señales digitales.

- Para desarrollar programas siempre se debe tener en cuenta las múltiples herramientas que brinda el software utilizado, además de las herramientas que se pueden acoplar con dicho software que facilita la solución de algunos problemas de una mejor manera, porque a veces se esta tan

acostumbrado al uso de un control determinado que no satisface totalmente las necesidades que se presentan y que puede satisfacer otro control con características para solucionar un problema específico.

- Es recomendable desarrollar un Sistema Informático entendible, con estándares y bien documentado para que si en algún momento se necesita de su modificación o actualización se lo pueda lograr de una manera sencilla y práctica, además del uso de comentarios que complementen el entendimiento del mismo, se deben usar otras técnicas y material didáctico para la total comprensión del proyecto.

- Se recomienda que un sistema sea hecho de tal manera que se lo pueda utilizar por un largo tiempo, es por eso que se debe tener en cuenta muchos factores como: la escalabilidad es decir para que se puedan hacer mejoras al sistema, el crecimiento de almacenamiento de datos a causa del crecimiento de la población estudiantil y de las mejoras de los servicios en el sistema, además de compatibilidad con nuevas tecnologías.

- En el mercado existen muchos dispositivos de lectura de huellas digitales por esta razón es importante hacer un estudio previo de varios factores para realizar una compra a conciencia del mismo; se debe tener en cuenta las facilidades que brinda, el factor económico también es importante como también lo es el de multiplataforma es decir que se pueda usar en cualquier parte con cualquier Sistema Operativo; además tener el conocimiento de que la empresa creadora del dispositivo tenga garantía, experiencia y dominio en lo que hace y pueda dar un soporte técnico eficiente del producto gracias a internet en cualquier parte del mundo.

- Debido a que las capacidades de las reglas de firewall y la implementación de *IPSec* son significativamente acrecentadas en Windows Vista y versiones posteriores de Windows, es recomendable dejar las propiedades existentes de Objetos de Directivas de grupo (GPO) para versiones anteriores de Windows, y crear nuevas GPOs para computadoras en las que corren versiones de Windows con el *Firewall de Windows con Seguridad Avanzada.*

BIBLIOGRAFÍA

LIBROS [L]

[L1] C. De la Torre, U. Zorrilla, J. Calvarro y M. Ramos, *Guía de Arquitectura N-Capas Orientada al Dominio con .NET 4.0*, 1ra edición, Microsoft Ibérica S.R.L., España, 2010.

[L2] I. Sommerville, *Ingeniería del software*, 7ma edición, Pearson Educación, Madrid, 2005.

[L3] R. Pressman, *Ingeniería del software*. *Un enfoque práctico*, 5ta edición, McGraw-Hill Interamericana de España, Madrid, 2002.

[L4] T. Thai, H. Lam, *.NET Framework Essentials*, 2da edición, O'Reilly, Estados Unidos, 2002.

[L5] I. Jacobson, G. Booch, y J. Rumbaugh, *El proceso unificado de desarrollo de software* ,1ra edición, Pearson Educación, Madrid. 2000.

[L6] R. Elmasri y S. Navathe, *Fundamentos de Sistemas de Bases de Datos*. 5ta edición, Pearson Educación, Madrid, 2007.

[L7] D. Costal, *Introducción al diseño de bases de datos*, UOC.

[L8] A.Silberchatz, H. Korth, y S. Sudarshan, *Fundamentos de bases de datos*. 4ta edición, McGRAW-HILL Interamericana de España, Madrid, 2002.

[L9] R. Alarcón, *Diseño orientado a objetos con UML*, 1ra edición, Grupo EIDOS, Madrid, 2000.

[L10] E. Newcomer, U*nderstanding Web Services: XML, WQSDL, SOAP and UDDI*, 3ra edición, Pearson Education, Boston, 2004.

[L11] D. Platt, *Introducing Microsoft .NET*, 3ra edición, Microsoft Press, Washington, 2003.

PÁGINAS WEB [PW]

[PW1] *Introducing Visual Studio .NET* [En línea]
Disponible: http://msdn.microsoft.com/es-ec/library/fx6bk1f4%28v=vs.71%29.aspx

[PW2] *Microsoft Visual Studio* [En línea]
Disponible: http://es.wikipedia.org/wiki/Microsoft_Visual_Studio#Visual_Studio_2010

[PW3] Novedades *Framework 4.0* [En línea]
Disponible: http://geeks.ms/blogs/adiazmartin/archive/2009/12/28/novedades-framework-4-0.aspx

[PW4] *ASP.NET* [En línea]
Disponible: http://es.wikipedia.org/wiki/ASP.NET

[PW5] *ASP.NET AJAX* [En línea]
Disponible: http://es.wikipedia.org/wiki/ASP.NET_AJAX

[PW6] Qué es y para qué sirve el lenguaje CSS (*Cascading Style Sheets* - Hojas de Estilo). [En línea]
Disponible: http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=546:que-es-y-para-que-sirve-el-lenguaje-css-cascading-style-sheets-hojas-de-estilo&catid=46:lenguajes-y-entornos&Itemid=163

[PW7] Análisis del protocolo *IPSec*: el estándar de seguridad en *IP* [En línea]
Disponible: http://www.frlp.utn.edu.ar/materias/internetworking/apuntes/*IPSec*/ipsec.pdf

[PW8] *HTTPS (HTTP over SSL or HTTP Secure)* [En línea]
Disponible: http://searchsoftwarequality.techtarget.com/definition/*HTTPS*

[PW9] *SSL, Secure Sockets Layer* y Otros Protocolos Seguros para el Comercio Electrónico [En línea]

Disponible: http://pics.unlugarenelmundo.es/hechoencasa/ssl%20secure%20sockets%20layer%20y%20otros%20protocolos%20seguros%20para%20el%20comercio%20electronico.pdf

[PW10] *TLS Transport LayerSecurityProtocol* [En línea]

Disponible: http://www.cybsec.com/upload/espe_tls.pdf

[[PW11] Biometría [En línea]

Disponible: http://es.wikipedia.org/wiki/Biometr%C3%ADa

[PW12] Huella dactilar [En línea]

Disponible: http://es.wikipedia.org/wiki/Huella_dactilar

[PW13] Lector De Huella Digital [En línea]

Disponible: http://lectorhuelladigital.co/tag/lector-de-huella-digital/

[PW14] *About SecuGen* [En línea]

Disponible: http://www.secugen.com/company/index.htm

[PW15] *Biometric Standards* [En línea]

Disponible: http://www.secugen.com/company/standards.htm

[PW16] *SecuGen Hamster Plus* [En línea]

Disponible: http://www.secugen.com/products/php.htm

[PW17] *DPI and PPI Explained* [En línea]

Disponible: http://www.andrewdaceyphotography.com/articles/dpi/

[PW18] *Framework* [En línea]

Disponible: http://es.wikipedia.org/wiki/Framework

[PW19] *Hypertext Transfer Protocol Secure* [En línea]

Disponible: http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol_Secure

[PW20] *InterNational Committee for Information Technology Standards* [En línea]

Disponible: http://www.incits.org/

[PW21] *We're ISO, the International Organization for Standardization. We develop and publish International Standards* [En línea]
Disponible: http://www.iso.org/iso/home.html

[PW22] *An Illustrated Guide to IPsec* [En línea]
Disponible: http://www.unixwiz.net/techtips/iguide-ipsec.html

[PW23] Dinero-e [En línea]
Disponible: http://www.iec.csic.es/criptonomicon/comercio/tarjetas.html

[PW24] *Píxel* [En línea]
Disponible: http://es.wikipedia.org/wiki/P%C3%ADxel

[PW25] *Fingerprint recognition* [En línea]
Disponible: http://en.wikipedia.org/wiki/Fingerprint_recognition

[PW26] *Kit de desarrollo de software* [En línea]
Disponible: http://es.wikipedia.org/wiki/Kit_de_desarrollo_de_software

[PW27] Sistemas de Pago Electrónico [En línea]
Disponible: http://pyme.net.uy/documentos/sistemas_pago.htm

[PW28] *Universal Serial Bus* [En línea]
Disponible: http://es.wikipedia.org/wiki/Universal_Serial_Bus

[PW29] Requisitos de sistema de *Windows Server* 2008 [En línea]
Disponible: http://technet.microsoft.com/es-es/windowsserver/bb414778.aspx

[PW30] *HP ML150 G6 E5504 HP SAS/SATA US Svr 466132-001* [En línea]
Disponible: http://www.compuzone.com.ec/producto.php?prodcod=788

[PW31] *Windows Server 2008* [En línea]
Disponible: http://es.wikipedia.org/wiki/Windows_Server_2008

[PW32] ¿Cual es el precio de una licencia de windows 2008 server? [En línea]

Disponible:

http://espanol.answers.yahoo.com/question/index?qid=20110614041117AAlpd57

[PW33] *SQL Server 2008* – Esquemas de Licenciamiento [En línea]

Disponible: http://download.microsoft.com/download/A/A/C/AACDC5CD-7A28-4493-BB0D-79FE286DF471/SQLServer2008_pricing.pdf

[PW34] REMUNERACIONES Y BENEFICIOS ADICIONALES [En línea]

Disponible: http://es.scribd.com/doc/30896278/Remuneraciones-y-Beneficios-Adicionales

[PW35] Ecuador: aumento salarial permitirá cubrir el 103% de la canasta básica del 2013 [En línea]

Disponible:http://www.americaeconomia.com/economia-mercados/finanzas/ecuador-aumento-salarial-permitira-cubrir-el-103-de-la-canasta-basica-del

[PW36] DIAGRAMASDE SECUENCIA [En línea]

Disponible: http://www2.uah.es/jcaceres/capsulas/DiagramaSecuencia.pdf

[PW37] Diagrama de Actividades UML 2 [En línea]

Disponible:http://www.sparxsystems.com.ar/resources/tutorial/uml2_activitydiagram.html

[PW38] DIAGRAMA DE DESPLIEGUE [En línea]

Disponible: http://umldaniel.blogspot.com/2009/05/diagrama-de-despliegue.html