INTERCONEXIÓN DE BASES DE DATOS SQL SERVER A PARTIR DE UNA BASE DE DATOS ORACLE

ADRIANA CRISTINA PEREZ URANGO
ALEJANDRO JOSE VILLAR TUÑON

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERIA DE SISTEMAS
MINOR EN DESARROLLO DE APLICAICONES .NET
CARTAGENA
2012

INTERCONEXIÓN DE BASES DE DATOS SQL SERVER A PARTIR DE UNA BASE DE DATOS ORACLE

Implementación de un *Web Services* para la actualización e interconexión de la base de datos SIRE en SQL Server con Banner UTB en Oracle

ADRIANA CRISTINA PEREZ URANGO
ALEJANDRO JOSE VILLAR TUÑON

Monografía como requisito final para optar por el título de Ingeniero de Sistemas.

OMER MANUEL SALCEDO GALVÁN
Ingeniero de Sistemas
Ms en Ingeniería
Director de Proyecto

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERIA DE SISTEMAS
MINOR EN DESARROLLO DE APLICAICONES .NET
CARTAGENA
2012

Nota de aceptación:

_____

_____

_____

_____

_____

_____

_____

Firma del Jurado

_____

Firma del Jurado

Adriana:

A Dios, creador de todas las cosas, quien me inspira a soñar y en su creación me enseña que todo es posible. A mi madre, por su amor, firmeza y devoción. Al premio a la Excelencia y Talento Caribe y sus benefactores, en especial a Argos S.A. – Planta Cartagena (Col Clinker), porque gracias a sus ideas e iniciativas hicieron algunos de mis sueños una realidad.

Alejandro:

Agradecer como primera instancia a Dios por permitir ser quien soy. A mis padres por su apoyo incondicional, amor y sobre todo confianza. Aquellas personas que me rodean cada día de mi vida para seguir luchando por mis metas y sueños trazados. Mi persona por haber alcanzado un logro más en mi proyecto de vida.

Expresamos nuestro reconocimiento a nuestros asesores técnicos en el Minor de Desarrollo .NET, a nuestros compañeros y colaboradores que contribuyeron al desarrollo del tema a lo largo de estos 4 meses de trabajo y los 5 años de estudio profesional: **Ms Ingeniería Omer Salcedo, Ms Ingeniería Edwin Puerta y Msc Moisés Quintana.**

Cartagena, 09 de Julio de 2012.

Señores
COMITÉ CURRICULAR
UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR
Ciudad

Respetados señores:

Con todo el interés me dirijo a ustedes para presentar a su consideración, estudio y aprobación la monografía titulada **INTERCONEXIÓN DE BASES DE DATOS SQL SERVER A PARTIR DE UNA BASE DE DATOS ORACLE- Implementación de un *Web Services* para la actualización e interconexión de la base de datos SIRE en SQL Server con Banner UTB en Oracle**, como requisito para obtener el título de Ingeniero de Sistemas.

Atentamente,

_____

ADRIANA CRISTINA PEREZ URANGO
C.C. 1.128.050.380 de Cartagena

Cartagena 09 de Julio de 2012.

Señores
COMITÉ CURRICULAR
UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR
Ciudad

Respetados señores:

Con todo el interés me dirijo a ustedes para presentar a su consideración, estudio y aprobación la monografía titulada **INTERCONEXIÓN DE BASES DE DATOS SQL SERVER A PARTIR DE UNA BASE DE DATOS ORACLE- Implementación de un *Web Services* para la actualización e interconexión de la base de datos SIRE en SQL Server con Banner UTB en Oracle**, como requisito para obtener el título de Ingeniero de Sistemas.

Atentamente,

_____

ALEJANDRO JOSE VILLAR TUÑON
C.C. 1.128.058.617 de Cartagena

AUTORIZACIÓN

Yo, ALEJANDRO JOSE VILLAR TUÑON, identificado con la cedula de ciudadanía número 1.128.058.617 de Cartagena, autorizo a la Universidad Tecnológica de Bolívar, para hacer uso de mi trabajo de monografía y publicarlo en el catalogo on-line de la biblioteca.

_____

ALEJANDRO JOSE VILLAR TUÑON

C.C. 1.128.058.617 de Cartagena

AUTORIZACIÓN

Yo, ADRIANA CRISTINA PEREZ URANGO, identificado con la cedula de ciudadanía número 1.128.050.380 de Cartagena, autorizo a la Universidad Tecnológica de Bolívar, para hacer uso de mi trabajo de monografía y publicarlo en el catalogo on-line de la biblioteca.

_____

ADRIANA CRISTINA PEREZ URANGO

C.C. 1.128.050.380 de Cartagena

TABLA DE CONTENIDO

Pág.

GLOSARIO

CONECTOR OLE DB: enlace e incrustación de objetos para bases de datos (Object Linking and Embedding for Databases), es una tecnología desarrollada por Microsoft usada para tener acceso a diferentes fuentes de información, o bases de datos, de manera uniforme.

ENTITY FRAMEWORK: permite a los desarrolladores crear aplicaciones de acceso a datos programando con un modelo de aplicaciones conceptuales en lugar de programar directamente con un esquema de almacenamiento relacional. El objetivo es reducir la cantidad de código y el mantenimiento necesarios para las aplicaciones orientadas a datos.

FRAMEWORK .NET: es un componente de software que puede ser o es incluido en los sistemas operativos Microsoft Windows. Provee soluciones pre-codificadas para requerimientos comunes de los programas y gestiona la ejecución de programas escritos específicamente para este framework.

JOB: un procedimiento(también llamado trabajo o Job) es un conjunto coherente de instrucciones para realizar un trabajo particular.

LLAVE FORANEA: identifica una columna o grupo de columnas en una tabla (tabla hija o referendo) que se refiere a una columna o grupo de columnas en otra tabla (tabla maestra o referenciada).

LLAVE PRIMARIA: es un conjunto de uno o más atributos de una tabla, que tomados colectivamente nos permiten identificar un registro como único.

ORACLE: es un sistema de gestión de base de datos – SGBD, objeto-relacional desarrollado por Oracle Corporation.

PROCEDIMIENTOS DE ALMACENADO: es un subprograma que ejecuta una acción específica y que no devuelve ningún valor. Un procedimiento tiene un nombre, un conjunto de parámetros (opcional) y un bloque de código.

SERVICIO WEB: los servicios Web (Web Services) son un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web y que intercambian datos entre sí con el objetivo de ofrecer servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web. Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario.

SIMPLE OBJECT ACCESS PROTOCOL (SOAP): protocolo Simple de Acceso a Objetos. Es un protocolo para intercambiar mensajes, basado en XML, y de extendido uso en Servicios Web, es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. Este protocolo deriva de un protocolo creado por David Winer en 1998, llamado XML-RPC (XML remote call procedure).

SISTEMA DE GESTIÓN DE BASE DE DATOS: son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

SQL SERVER: es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional. Sus lenguajes para consultas son T-SQL y ANSI SQL

URI: identificador de Recurso Uniforme (Uniform Resource Identifiers). Los URI son cadenas que funcionan como identificadores globales que hacen referencia a recursos en la Web tales como documentos, imágenes, archivos descargables, servicios, buzones de correo electrónico y otros.

WINDOWS COMMUNICATION FOUNDATION (WCF): es la plataforma de Microsoft para el desarrollo y ejecución de servicios Web.

WEB SERVICES DESCRIPTION LANGUAGE (WSDL): lenguaje de Descripción de Servicios Web (Web Services Description Language) permite definir lo que hace un Servicio Web según la funcionalidad que ofrece. Mediante este lenguaje se representa la interfaz de uso del servicio, lo que tendrán que tener en cuenta otros servicios a la hora de acceder a su funcionalidad.

WORLD WIDE WEB CONSORTIUM (W3C): comunidad internacional que desarrolla estándares que aseguran el crecimiento de la Web a largo plazo, sitio web en español http://www.w3c.es/

XML: lenguaje de etiquetado extensible (eXtensible Markup Language), es un lenguaje con una importante función en el proceso de intercambio, estructuración y envío de datos en la Web. Describe los datos de tal manera que es posible estructurarlos utilizando para ello etiquetas, como lo hace HTML, pero que no están predefinidas, delimitando de esta manera los datos, a la vez que favoreciendo la interoperabilidad de los mismos.

INTRODUCCIÓN

Con el impulso que proporciona el Internet, las nuevas tecnologíasy la escalabilidad que éstas incorporan a la administración de datos se requieren componentes de integración e interoperabilidad cuyo funcionamiento sea transparente al usuario final.

Es en esta interoperabilidad donde los servicios Web (WS) cobran importancia, ya que permiten la interoperación de sistemas distribuidos heterogéneos que sean independientes de las plataformas hardware y software usados. El *World Wide Web Consortium* (W3C), comunidad internacional que desarrolla estándares que aseguran el crecimiento de la Web a largo plazo, define un WS como una aplicación software identificada por un URI cuyas interfaces se pueden definir, describir y descubrir mediante documentos XML. Esto supone el uso no solo del concepto sino de frameworks robustos que soporten el trabajo de especificación de las mismas (1).

Windows CommunicationFoundation (WCF), la plataforma de Microsoft para el desarrollo y ejecución de servicios Web, integra un conjunto de tecnologías proporcionadas por .NET Framework, tales como ASP.NET Web Services, dotNETRemoting, entre otros, que implementan una serie de especificaciones WS-*, etc. WCF incluye un número significativo de tecnologías relacionadas con los WS entre las que destacan SOAP y WSDL (1) (2).

En el marco del Minor de desarrollo .NET, se consultaron soluciones y aplicaciones relacionadas con el manejo de servicios Web para la consulta segura de datos en una servidor SQL server a partir de un servidor en Oracle. Se exploraron ventajas y desventajas de dichos servicios en *Microsoft Developer Network* en su MSDN Library, además de descriptores del *Web Service Definition Language* (WSDL) del W3C.

En el presente documento se presentan los lineamientos bajo los cuales se desarrolló el trabajo de implementación del servicio web wcfSIRE. En primera instancia se dan a conocer los antecedentes y la situación actual del departamento de sistemas hacia el cual se enfoca el tema. Del mismo modo se expone la importancia que tiene la presente implementación y los objetivos que se pretenden alcanzar con ella. Seguidamente se dan a conocer las teorías bajo las cuales se soporta el estudio y el diseño metodológico a utilizar para la implementación y funcionamiento del WS. Finalmente, se hace una relación de los logros alcanzados, actividades a desarrollar y trabajos adicionales que éstas generan.

1. DEFINICIÓN DEL PROBLEMA

Ante las políticas de la Universidad Tecnológica de Bolívar – UTB, para el manejo y administración de su base de datos Banner, se requiere la implementación de una solución para acceder a algunas tablas donde desarrolladores de aplicaciones y algunos usuarios de las mismas se sirvan de información que les permita llevar el seguimiento de actividades y simular opciones de información. Una de las políticas de la UTB radica en la reposición de información en un único sistema de administración de información al cual se accede solo como administrador/empleado de la universidad.

Debido a esto, en el laboratorio de ingeniería de la UTB, se desarrolló una base de datos en SQL Server denominada Sistema de Información de Recursos Estudiantiles - SIRE, para consulta de Banner a través de vistas. Sin embargo el modelo al hacer diversas consultas en SQL arrojaba resultados con más de 10 segundos y excesivo tráfico en la red debido a que los filtros solo eran aplicados en SIRE a la tabla completa traída desde Banner.

De igual forma las vistas traídas desde Banner por su caracterización, tienen campos repetidos indistintamente en las tablas, de forma que la definición de relaciones a través de llaves primarias y foráneas no permite construir un modelo normalizado, se debe determinar cuáles tablas son fundamentales para acceder a Banner y permitir el acceso a las aplicaciones que, en el marco del Minor en desarrollo .NET, necesitan acceder a SIRE.

El acceso a SIRE debe estar garantizado a los usuarios de dichas aplicaciones en el rol correspondiente previa verificación de acceso concedido en Banner (login).

Finalmente, para el presente trabajo se parte del acceso limitado que se ha proporcionado desde banner, no se tienen en cuenta consideraciones de rendimiento externas al servidor donde se encuentra la base de datos SIRE, tampoco consideraciones de seguridad y otros directamente relacionados a Banner que no se ejecuten estrictamente a través del login. Se debe encontrar entonces una solución que permita interconectar Banner y SIRE cumpliendo las especificaciones antes mencionadas.

2. JUSTIFICACIÓN

En un primer enfoque se configuró el servidor de SQL Server a fin de que obtuviese el modelo de datos definido previamente en la base de datos de Oracle a través de *Integration Services*, haciendo un proceso de vinculación de Bases de datos. Se utilizó el driver OLE DB para esta configuración, sin embargo dicha vinculación generó algunos problemas especialmente para las vistas de gran volumen de datos. Esto supone que las aplicaciones que se conecten a SIRE pueden tener en un mediano plazo serias complicaciones de rendimiento.

Seguidamente como directiva de seguridad, no se debe almacenar información relacionada con el acceso vía web de los usuarios finales, ante esta situación el modelo de datos no posee dicha información.

Finalmente para suplir la información de acceso se deben consultar formas de autenticar a un usuario final en SIRE de manera que su información esté actualizada.

3. OBJETIVOS

3.1. General

Implementar un servicio web el cual permita a aplicaciones .NET conectarse a una DB en SQL SERVER a partir de una base de datos en ORACLE.

3.2. Específicos

- Establecer conectividad entre servidores SQLSERVER – ORACLE a través de un enlazador.

- Diseñar un modelo de datos SQL server hasta la forma normal necesaria para garantizar el correcto funcionamiento del sistema.

- Desarrollar un servicio de interconexión que permita conceder permisos de acceso a SIRE desde Banner.

- Implementar una técnica de encriptación para la comunicación Banner - SIRE.

4. MARCO REFERENCIAL

4.1. BASES DE DATOS

Las bases de datos permiten el almacenamiento estructurado de datos, desde grandes aplicaciones multiusuario hasta agendas electrónicas utilizan tecnología de bases de datos para garantizar la integridad de los datos y facilitar las tareas tanto de desarrolladores como de usuarios.

Existen programas denominados Sistemas Gestores de Bases de Datos, abreviado SGBD, que permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada.

4.1.1. Modelos de Datos

Los modelos de datos (2) son una descripción del contenedor de datos, son muy variados, e importantes para entender la lógica de funcionamiento de las bases de datos ya que conocer los conceptos y la tecnología asociados a estos temas permite tener éxito en cualquier proyecto que implique trabajar con bases de datos. Los modelos de datos pueden ser relacionales, orientados a objeto, transaccionales, entre otros.

4.1.2. Modelo de datos relacional

El modelo relacional para la gestión de una base de datos es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente.

Su idea fundamental es el uso de *relaciones*, pensando en cada relación como si fuese una tabla compuesta por registros o *tuplas* (llamadas filas de la tabla), y columnas (también llamadas campos)

4.1.3. Lenguaje SQL

Structured Query Language - SQL (3) es un lenguaje de programación de reconocimiento internacional para la definición y mantenimiento de bases de datos relacionales, inicialmente desarrollado por IBM a finales de 1970.

La popularidad de SQL aumentó aún más cuando el lenguaje se convirtió en una norma oficial en octubre de 1986, con el American National Standards Institute - ANSI. Este primer estándar SQL, SQL-86, se convirtió en internacionalmente aceptada en 1987, cuando la Organización Internacional de Normalización (ISO) - una federación mundial de organismos nacionales de normalización - aprobó el documento ANSI también.

4.1.4. Normalización

El proceso de normalización (3) de bases de datos consiste en aplicar una serie de reglas a las relaciones obtenidas tras el paso del modelo entidad-relación al modelo relacional. Las bases de datos relacionales se normalizan para:

- Evitar la redundancia de los datos.
- Evitar problemas de actualización de los datos en las tablas.
- Proteger la integridad de los datos.

En el modelo relacional es frecuente llamar tabla a una relación, aunque para que una tabla sea considerada como una relación tiene que cumplir con algunas restricciones:

- Cada tabla debe tener su nombre único.
- No puede haber dos filas iguales. No se permiten los duplicados.
- Todos los datos en una columna deben ser del mismo tipo.

Para normalizar una base de datos, se elaboran sencillos pasos llamados Formas Normales (3), por la sencillez en las relaciones y cantidad de tablas de SIRE consideraremos hasta la segunda forma normal:

- *Primera Forma Normal (1FN)*: Una tabla (relación) está en 1FN si
  - No existen registros duplicados en la tabla.
  - Cada celda es univalorada (es decir, no existen grupos repetitivos o matrices).
  - Las entradas en una columna (atributo, campo) son del mismo tipo.
- *Segunda Forma Normal (2FN)*: Una tabla está en 2FN si está en 1FN y si todos los atributos que no son una clave dependen sobre todo de la clave.

4.1.5. Sistemas de Gestión de Bases de Datos

Los sistemas de gestión de bases de datos – SGBD, son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

Con el crecimiento de las organizaciones, hay numerosas bases de datos – BD, y varios SGBD de diferentes tipos o proveedores. En la actualidad, gracias principalmente a la estandarización del lenguaje SQL, los SGBD de marcas

diferentes pueden darse servicio unos a otros y colaborar para dar servicio a un programa de aplicación.

4.1.6. SGBD SQL server

Microsoft SQL Server es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional. Sus lenguajes para consultas son T-SQL y ANSI SQL. Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle, PostgreSQL o MySQL.

4.1.7. SGBD Oracle 10g

Oracle es un sistema de gestión de base de datos objeto-relacional u ORDBMS por el acrónimo en inglés de *Object-Relational Data Base Management System*, desarrollado por *Oracle Corporation*.

En 2003 *Oracle Corporation* lanza Oracle 10g, donde la "g" viene de "Grid", que incorpora el manejo y administración de bases de datos en malla, el cual consiste en un conjunto de bases de datos cuya administración de espacio, recursos y servicios pueden administrarse como si fueran una sola.

4.1.8. Interconectividad entre SGBD

.NET Framework *Data Providerfor OLE DB (OleDbConnection)* OLE DB (4) es una metodología de acceso a datos estándar abierto que utiliza un conjunto de Modelo de objetos de componentes (COM) de interfaces para acceder y manipular

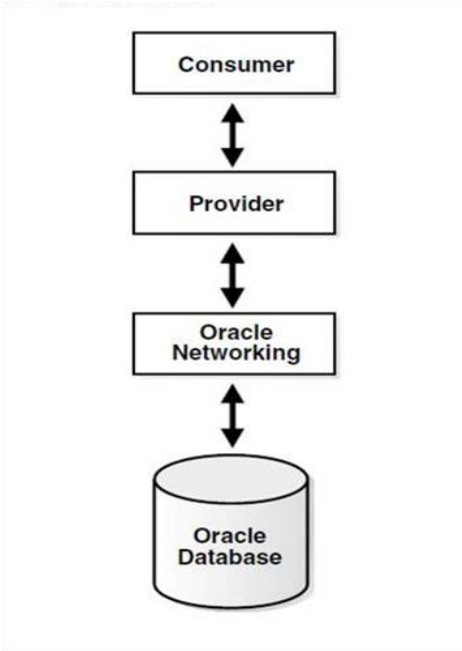
diferentes tipos de datos. Estas interfaces están disponibles en diferentes proveedores de bases de datos.

El Diseño de OLE DB se centra en el concepto de consumidor y proveedor. En la ilustración1 - "Flujo OLE DB", es un ejemplo del sistema de OLE DB. El consumidor representa el cliente tradicional. El proveedor coloca los datos en un formato tabular y devuelve al consumidor.

OLE DB DATA PROVIDER son un conjunto de componentes COM que transfieren datos desde una base de datos fuente a un consumidor.

OLE DB PROVIDER posiciona los datos en forma tabular en respuesta a las llamadas de un consumidor. Los proveedores pueden ser simples o complejos.
Un *PROVIDER* puede devolver una tabla, puede que el consumidor determine el formato de la tabla, y realice operaciones sobre los datos. Cada *PROVIDER* implementa un conjunto estándar de interfaces COM para tramitar las solicitudes de el consumidor. Un *PROVIDER* puede implementar interfaces opcionales COM para proporcionar funcionalidad adicional.

Ilustración 1 - "Flujo OLE DB"



Fuente:http://docs.oracle.com/cd/A91202_01/901_doc/win.901/a90171/intro.htm

Con las interfaces estándar, cualquier consumidor OLE DB puede tener acceso a datos desde cualquier *PROVIDER*. Debido a los componentes COM, los consumidores pueden acceder a ellos en cualquier lenguaje de programación compatible con COM, como C + +, Visual Basic, Visual C# y Java.

*El OLE DB DATA CONSUME*R es cualquier aplicación o herramienta que utiliza las interfaces de OLE DB PROVIDER para acceder a una amplia gama de datos.

Oracle Provider for OLE DB, es un OLE DB proveedor de datos que ofrece alto rendimiento y un acceso eficiente a los datos de Oracle por OLE DB DATA CONSUMER.

Con la llegada de .NET Framework, soporta y se ha proporcionado el uso del OLE DB.NET Data Provider con Oracle Provider for OLE DB. Con el atributo de

conexión adecuada OLE DB.NET Data Provider puede utilizar Oracle ProviderFor OLE DB para acceder a Oracle.

Para hacer Oracle Provider for OLE DB compatible con OLE DB. NET Data Provider, establezca el atributo de cadena de conexión OLEDB.NET en True.
Cuando se utiliza Oracle Provider for OLE DB con el OLE DB. NET Data Provider, el OLEDB.NET atributo de conexión se debe establecer en TRUE como se muestra en la Ilustración 2 – Ejemplo conexión en C#:

Ilustración 2 - Ejemplo conexión en C#:

```
OleDbConnection con = new OleDbConnection();
con.ConnectionString = "Provider=OraOLEDB.Oracle;User Id=scott;" +
"Password=tiger;Data Source=Oracle;OLEDB.NET=true;"
con.Open();
```

Fuente:http://docs.oracle.com/cd/B28359_01/win.111/b28431/using.htm#i1017293

4.2. SERVICIOS

Los Servicios Web (5) o Web Services (WS) surgieron ante una necesidad de estandarizar la comunicación entre distintas plataformas (PC, Mainframe, Mac, etc.) y lenguajes de programación (PHP, C#, Java, etc.).

Anteriormente se realizaron intentos de crear estándares que no tuvieron el suficiente éxito, algunos de ellos son DCOM y CORBA, por ser dependientes de la implementación del vendedor DCOM - Microsoft, y CORBA - ORB (a pesar que CORBA de múltiples vendedores pueden operar entre sí, hay ciertas limitaciones

para aplicaciones de niveles más altos en los cuales se necesite seguridad o administración de transacciones).
Otro gran problema es que se hacía uso de RPC (RemoteProcedureCall) para realizar la comunicación entre diferentes nodos. Esto, además de presentar ciertos problemas de seguridad, tiene la desventaja de que su implementación en un ambiente como es Internet, es casi imposible (muchos firewalls bloquean este tipo de mensajes, lo que hace prácticamente imposible a dos computadoras conectadas por Internet comunicarse).

Los Web Services surgieron para finalmente poder lograr la tan esperada comunicación entre diferentes plataformas. En la actualidad muchos sistemas legacy están pasando a ser web services. Es por esto que en 1999 se comenzó a plantear un nuevo estándar, el cual terminaría utilizando XML, SOAP, WSDL, y UDDI.

A pesar de mucho limitar el uso de los WS al protocolo HTTP, los Web services no fueron pensados para un protocolo en particular, es decir, nada impide utilizar SOAP sobre algún otro protocolo de Internet (SMTP, FTP, etc.).
Se utiliza principalmente HTTP por ser un protocolo ampliamente difundido y que se encuentra menos restringido por firewalls (generalmente se bloquean puertos como el FTP, pero el HTTP es muy probable que no esté bloqueado).

4.2.1. Funcionalidad

WS es un estándar de comunicación entre procesos y/o componentes. No importa en qué lenguaje esté programado, diseñado para ser multiplataforma y multilenguaje, estos son accesibles y utilizables por otras aplicaciones desarrolladas en otras plataformas o lenguajes de programación.

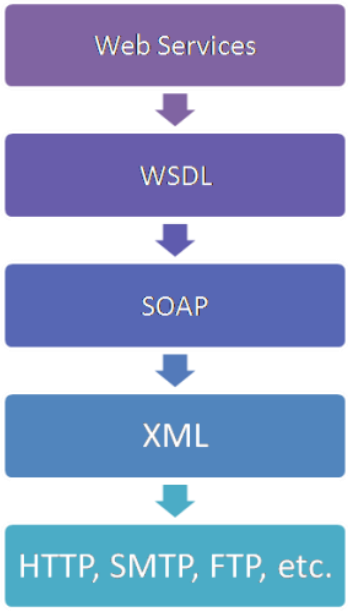
4.2.2. Protocolo HTTP

El HTTP que significa Hyper Text Transfer Protocol se compone de 2 mensajes, el primer mensaje es originado por el cliente y es el requerimiento inicial de la comunicación. Este requerimiento llamado Request está dividido en una cabecera y un mensaje de texto.

El protocolo HTTP no necesariamente tiene que usar el puerto 80, si bien es cierto que este es el más utilizado, podremos configurar el Servidor Web para que utilice este protocolo en otro puerto y desde un navegador poder acceder a él de la siguiente manera: http://www.misitio.com:XXXX/ donde XXXX sería el numero del puerto.

La ilustración 3 - Tecnología y protocolos en WS, muestra el orden jerárquico de las tecnologías y protocolos utilizados en los Web Services

Ilustración 3 - Tecnología y protocolos en WS



Fuente:http://img.redusers.com/imagenes/libros/lpcu104/capitulogratis.pdf

Antes de entrar en detalle con respecto a la utilidad o para qué sirven los Web Services, se dice que el desarrollo y la programación de sistemas orientado a objetos o componentes ha llevado a lo largo del tiempo a tener la necesidad de reutilizarlos en diferentes proyectos. Los Web Services facilitan la reutilización de funciones de una aplicación en distintas plataformas o lenguajes ya sea para un uso personal en distintos proyectos, para comercializarlos o adquirir prestaciones de terceros.

4.2.3. Extensible MarkupLanguage – XML

Es un lenguaje utilizado para definir formatos de documentos o mensajes. Estos se componen por Tags (palabra entre caracteres <y>). Estos tipos de documentos han sido aceptados y adoptados por la mayoría de los fabricantes de software para proveer extensibilidad de los datos debido a que no está atado a ninguna plataforma o lenguaje de programación.

Aparte de los Tags hay otras restricciones y validaciones que los documentos XML deben cumplir, eso depende de un archivo de definiciones llamado DTD (Document Type Definitions) o de un archivo de esquemas de XML, los cuales pueden estar asociados con el documento XML. En estos archivos DTD's podremos definir si un elemento (llamado Nodo) deberá al menos aparecer una vez, una o más veces, una a ninguna vez. Cada elemento podrá tener uno o más atributos, eso depende también del archivo de definiciones o esquema. Estos documentos XML son la base de los documentos WSDL y SOAP, base de los Web Services.

4.2.4. Web Services Description Language – WSDL

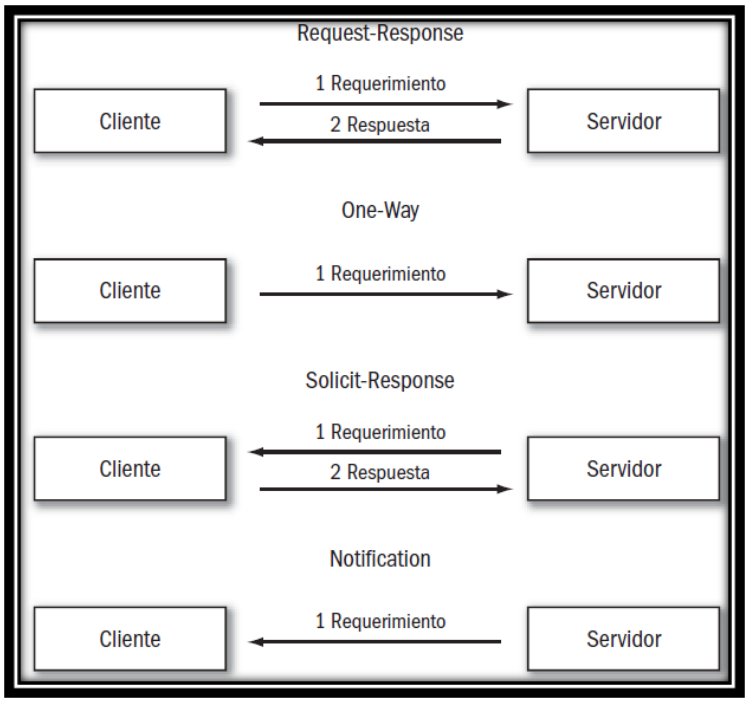
WSDL (1), es un documento XML que se utiliza para describir los mensajes SOAP y como éstos mensajes son intercambiados, es decir, se crea un Web Service y se quiere que otras aplicaciones lo consuman, las otras aplicaciones deben acceder a un documento WSDL en donde pueden conocer los métodos que expone el Web Service y como acceder a ellos, es decir, cuales son los nombres de los métodos y que tipos de parámetros espera cada uno de ellos.

Todo documento WSDL está compuesto por un elemento raíz llamado definitions que a su vez está compuesto de los siguientes elementos:

- *Types*: define el tipo de esquema a ser utilizado, usualmente XML
- *Message:* en este elemento se definen los mensajes de entrada y salida en forma abstracta entre el servidor y el cliente. Normalmente habrá varios elementos de este tipo para cada protocolo (HTTP, SOAP) y para cada Web Method.
- *PortType:* define los tipos de mensajes a intercambiar entre el cliente y el servidor. Puede ser de los tipos:
  - *Request-response*
  - *One-Way*
  - *Solicit-Responde*
  - *Notification*
- *Binding*: en este elemento se establecen las definiciones de los vínculos de los protocolos como SOAP a un tipo de vínculo en particular. Por el protocolo SOAP tendremos en atributo soapAction del elemento soap:operation la URL que tendremos que utilizar para invocar un Web Method

La ilustración 4 - Tipos de porType, muestra Distintos tipos de portType con sus órdenes y direcciones.

Ilustración 4 – Tipos de porType



Fuente:http://img.redusers.com/imagenes/libros/lpcu104/capitulogratis.pdf

El la ilustración 5 – Documentos WSDL, se muestra la relación de los distintos elementos que componen un documento WSDL:

Ilustración 5 - Documento WSDL

```
<definitions>
<types>Los tipos de datos...</types>
<message>Las definiciones del mensaje...</message>
<portType>Las definiciones de operación...</portType>
<binding>Las definiciones de protocolo...</binding>
</definitions>
```

Fuente:http://www.cibernetia.com/manuales/servicios_web/4_wsdl.php

4.2.5. Simple Object Access Protocol – SOAP

SOAP (5) es el protocolo base de los Web Services. Este protocolo está basado en XML y no se encuentra atado a ninguna plataforma o lenguaje de programación. Además es el protocolo mas aceptado por la mayoría de las plataformas.

Los mensajes SOAP se componen por un tag principal llamado Envelope, que está dividido en una cabecera o Header y en un cuerpo o Body.

4.3. ENTITY FRAMEWORK

Entity Framework (7) es un conjunto de tecnologías de ADO.NET que permiten el desarrollo de aplicaciones de software orientadas a datos.

En el desarrollo de software arquitectos y programadores de aplicaciones orientadas a datos tienen dos objetivos muy diferentes.
Por una parte modelar las entidades, las relaciones y la lógica de los problemas empresariales que resuelven, además de trabajar con los motores de datos que se usan para almacenar y recuperar los datos, ya sea en uno o varios sistemas de almacenamiento, con sus respectivos protocolos y requisitos del sistema de almacenamiento y por otro con respecto a los requisitos de escribir un código de aplicación eficaz y fácil de mantener.

4.3.1. Generalidades y alcance

Entity Framework permite a los programadores trabajar con datos en forma de objetos y propiedades específicos del dominio, por ejemplo, con clientes y direcciones, sin tener que pensar en las tablas de las bases de datos subyacentes y en las columnas en las que se almacenan estos datos. Con Entity Framework, los desarrolladores de software pueden trabajar en un nivel más alto de abstracción cuando tratan con datos, y puede crear y mantener aplicaciones orientadas a datos con menos código que en las aplicaciones tradicionales. Dado que Entity Framework es un componente de .NET Framework, las aplicaciones de Entity Framework se pueden ejecutar en cualquier equipo en el que esté instalado .NET Framework a partir de la versión 3.5 SP1.

4.3.2. Modelo de datos conceptual

La creación de un modelo relacional extendido, llamado Entity Data Model (EDM), que engloba las entidades y las relaciones, se maneja con un lenguaje de consultas para EDM, un motor de mapeado completo que traduce del nivel conceptual al lógico (relacional), y un conjunto de herramientas guiadas por modelos que ayudan a crear los transformadores entidad-objeto, objetoxml y entidad-xml.

El primer paso es definir un modelo conceptual propio. EDM representa una expresión formal, de diseño y ejecución de un modelo. Con EDM se puede describir el modelo en términos de las entidades y las relaciones, definir el modelo explícitamente de forma manual escribiendo el XML o incluso a través de una herramienta gráfica de diseño.

Lo que hace ADO.NET EntityFrameWork es mapear los objetos de Negocio a tablas relacionales atacando así directamente a los datos entre un modelo entidad relación y un modelo orientado a objetos. ADO.NET EntityFrameWork almacena

metadatos EF (CSDL,SSDL,MSLcontent) , en un archivo *.edmx estos están separados en los siguientes elementos XML:

- Archivo de lenguaje de definición de esquemas conceptuales (.csdl): defineel modelo conceptual.
- Archivo de lenguaje de definición de esquemas de almacenamiento (.ssdl): define el modelo de almacenamiento, que también se denomina modelo lógico.
- Archivo de lenguaje de especificación de asignaciones (.msl): define la asignación entre los modelos conceptuales y de almacenamiento.

Como algo más que otra solución de asignación objeto-relacional, Entity Framework permite que las aplicaciones obtengan acceso y cambien los datos que están representados como entidades y relaciones en el modelo conceptual.
Los resultados de las consultas en EDM se materializan en los objetos que servicios de objeto administran. Entity Framework proporciona las maneras siguientes de consultar un EDM y devolver objetos:

- *LINQ toEntities*: proporciona compatibilidad con Language - IntegratedQuery (LINQ) para consultar los tipos de entidad que se definen en un modelo conceptual.

- *Entity SQL*: dialecto independiente del almacenamiento de SQL que funciona directamente con las entidades del modelo conceptual y que admite características del EDM como la herencia y las relaciones. Se utiliza con las consultas de objeto y con las consultas que se ejecutan con el proveedor de EntityClient.

- *Métodos del generador de consultas*: permite construir consultas de Entity SQL utilizando los métodos de consulta del estilo de LINQ.

El Entity Framework incluye el proveedor de datos de EntityClient. Este proveedor administra las conexiones, traduce las consultas de entidad en consultas específicas del origen de datos y devuelve un lector de datos que Servicios de objeto usa para materializar los datos de la entidad en los objetos. Cuando no se requiere la materialización de los objetos, el proveedor de EntityClient también se puede utilizar como un proveedor de datos ADO.NET estándar habilitando las aplicaciones para ejecutar las consultas de Entity SQL y usar el lector de datos de solo lectura devuelto.

4.4. SEGURIDAD INFORMÁTICA

La seguridad informática se enfoca en la protección de la infraestructura computacional y todo lo relacionado con esta, incluyendo la información privilegiada o confidencial contenida en ésta. La seguridad informática comprende software, bases de datos, metadatos, archivos y todo lo que se valore como activo y signifique un riesgo si ésta llega a personas no autorizadas.

Para garantizar la seguridad informática existen una serie de estándares, protocolos, métodos, reglas, herramientas y leyes concebidas para minimizar los posibles riesgos a la infraestructura o a la información.

4.4.1. Técnicas de Encriptación

Existen técnicas para asegurar el sistema como codificar la información para ocultar contraseñas difíciles de averiguar a partir de datos personales del individuo. Otra técnica es la de respaldo remoto o servicio de backup remoto

4.4.2. Técnica Hashing

Los datos pueden cifrarse con un algoritmo criptográfico y transmitirse en un estado cifrado a una tercera persona, que posteriormente los descifrará. Si un tercero intercepta los datos cifrados, le resultará difícil.

Los algoritmos hash asignan valores binarios de longitud arbitraria a valores binarios más pequeños de longitud fija, que se denominan valores hash. Un valor hash es una representación numérica de un segmento de datos. Si aplica un algoritmo hash a un párrafo de texto simple y cambia solo una letra del párrafo, el valor hash subsiguiente producirá un valor distinto.

4.4.3. Algoritmo SHA1

SHA1 (8) es un algoritmo criptográfico perteneciente a la familia de algoritmos criptográficos SHA (Secure Hash Algorithm o Algoritmo de Hash Seguro) de la Agencia Nacional de Seguridad de Estados Unidos y desarrollada por el NIST (NationalInstitude of Standards and Technology).

El primero de estos algoritmos fue desarrollado en el año 93 y fue el SHA (ahora conocido como SHA-0), 4 años después, fue sustituido por el SHA1 y posteriormente han aparecido SHA-224, SHA-256, SHA-384 y SHA-512 y todos son conocidos como SHA-2.
En 1998, se encontró una vulnerabilidad en SHA-0 pero esta no afectaba a SHA1 del cual no se ha encontrado ningún ataque efectivo. En 2004, fueron publicados una serie de ataques sobre hash parecidos al que genera SHA-1 planteándose por lo tanto dudas sobre la seguridad que este aporta.

5. DISEÑO METODOLÓGICO

Un hallazgo importante para la dirección del presente trabajo consiste en poder usar un servicio web el cual invoca un procedimiento almacenado en PL/SQL de Oracle para la autenticación de un usuario final. Dicho servicio fue desarrollado en java y es el de autenticación del sistema de elecciones diseñado por la dirección del programa de Ingeniería de Sistemas en el año 2010.

Se parte de la premisa que la tecnología necesaria para desarrollar el Web Service es .NET y se definen 3 requisitos principales: acceso-login, defínción-roles y obtención-datos (docentes/estudintes), se plantea la estructura de análisis que permita al servicio en java ser consumido por aplicaciones desarrolladas en .NET.

Entre los modelos que parten de la base de datos se tomó Entity framework, que permite partir de una base de datos para construir un modelo funcional descrito en el marco teórico, dicho modelo sólo necesita de las tablas directamente relacionadas con la validación de los requisitos funcionales del sistema, ya que son las aplicaciones las que necesitan definir el nivel de acceso por medio del presente servicio.

5.1. HERRAMIENTAS Y METODOLOGÍAS A UTILIZAR

La herramienta Microsoft Visual Studio 2010 (MVS-2012) se empleó para la construcción del WS y previamente se definen los objetos de sincronización a través de SQL Server Management Studio.MVS-2012 utiliza como marco de trabajo a Windows CommunicationFoundation (WCF) por lo que a partir de ahora será llamado al paquete principal como wcfSIRE.

Se crea un proyecto del Tipo WCF Service Library, en el que se debe definir previamente como se llevará la comunicación entre el proveedor y un consumidor del servicio: definición del Contrato.

5.2. CRONOGRAMA DE ACTIVIDADES

MES - SEMANA	MARZO			ABRL				MAYO					JUNIO			
ACTIVIDAD	SEM 1	SEM 2	SEM 3	SEM 4	SEM 5	SEM 6	SEM 7	SEM 8	SEM 9	SEM 10	SEM 11	SEM 12	SEM 13	SEM 14	SEM 15	SEM 16
Configuración OLE-DB Oracle																
Creación de Vistas desde Banner																
Construcción de Tablas en SIRE																
Creación SP para actualización de tablas																
Creación de Jobs																
Análisis del Servicio Web (WS)																
Definición de tablas base para el WS																
Codificación del servicio																
Verificación y validación del servicio																
Documentación del Servicio Web																

6. RESULTADOS Y DISCUSIÓN

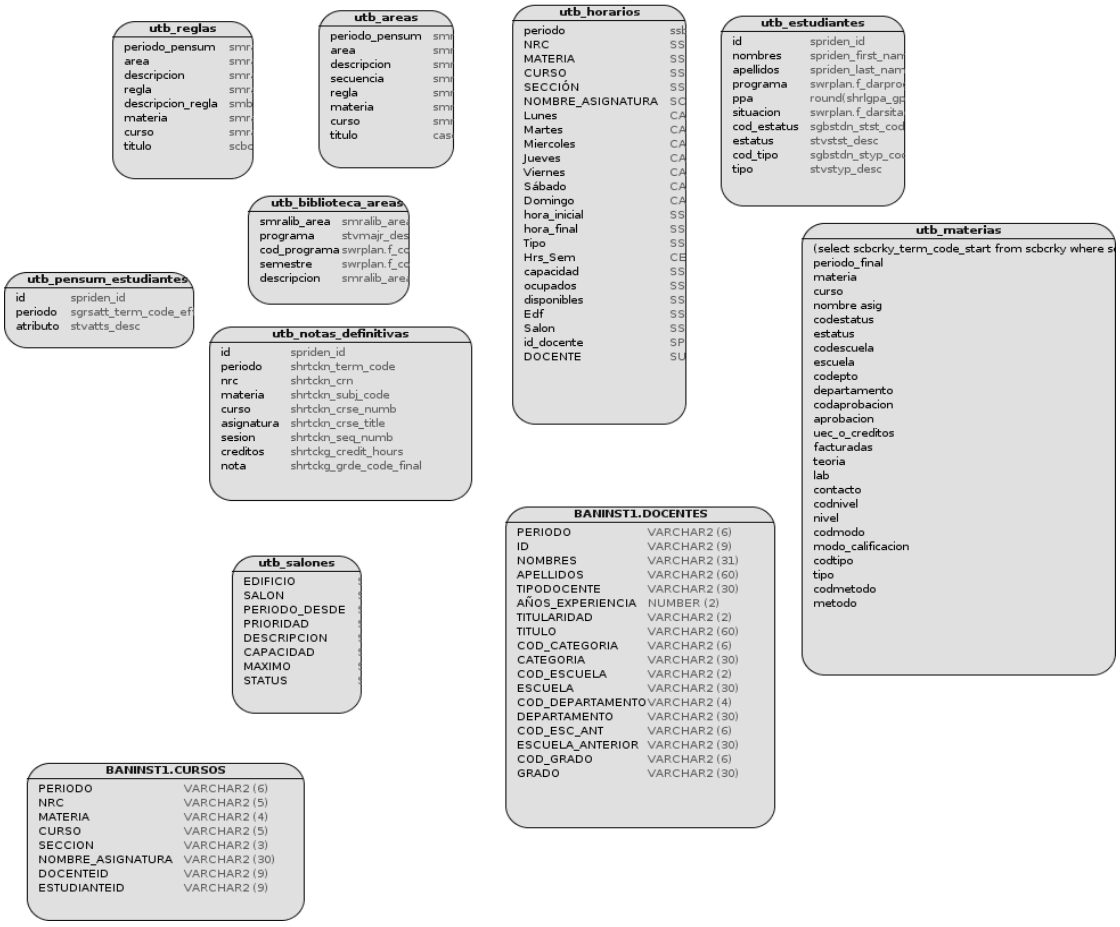
Con el conocimiento del servicio de acceso, desarrollado en Java en 2010 para las elecciones de representantes de estudiantes y docentes, que accede a Banner de forma segura se enfocó el desarrollo del servicio “I-SIRE” de interconexión que envíe algunos datos actualizados a SIRE, e interconectar aplicaciones .Net y otras frameworks de forma segura.

Dicho servicio I-SIRE debe garantizar entre otras cosas la encriptación de datos de usuarios proporcionados por Banner, la navegación entre servicios de acceso a las tablas de forma segura.

Para la descripción de los resultados se han definido dos etapas, una primera denominada en el presente documento como PRELIMINARES se centra en la construcción de la base de datos SQL Server y la interconexión con la base de datos Oracle. Y una segunda etapa denominada WEB SERVICE comprende la construcción de los servicios web para ser consumidos por las aplicaciones que se conectan a la base de datos SIRE.

Desde Banner – UTB, se sacan las vistas de las tablas originales como lo muestra la Ilustración 6 - Modelo de datos de Banner:

Ilustración 6 - Modelo de datos de Banner



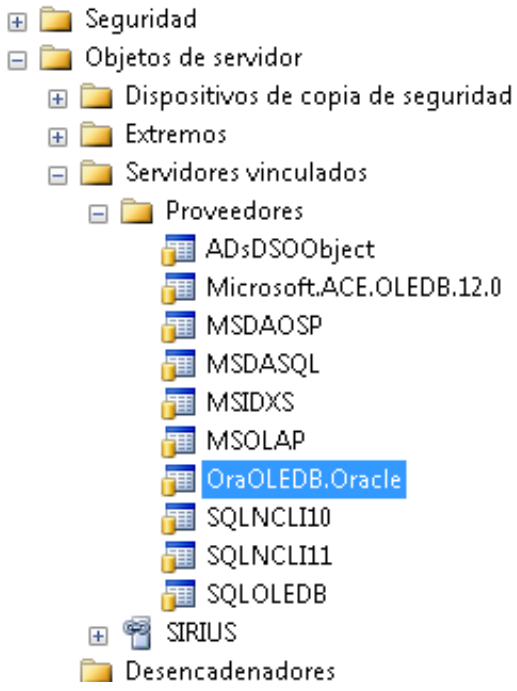
Fuente: Elaboración propia

6.1.1. Configuración Provider OLE DB

Para interconectar los SGBD se hace necesario utilizar el proveedor OLE DB de oracle que permite interconectar este sistema gestor con otro como Microsoft, se utilizó la siguiente ruta para la configuración del conector en SQL Server Management Studio (lustración 6 - Configure provider in SQL Server):

Servidor → DBName → Objetos de servidor → Servidores vinculados → Proveedores → OraOLEDB.Oracle → Propiedades → Habilitar “Permitir inProcess”. (4)

Ilustración 7 - configure provider in SQL Server

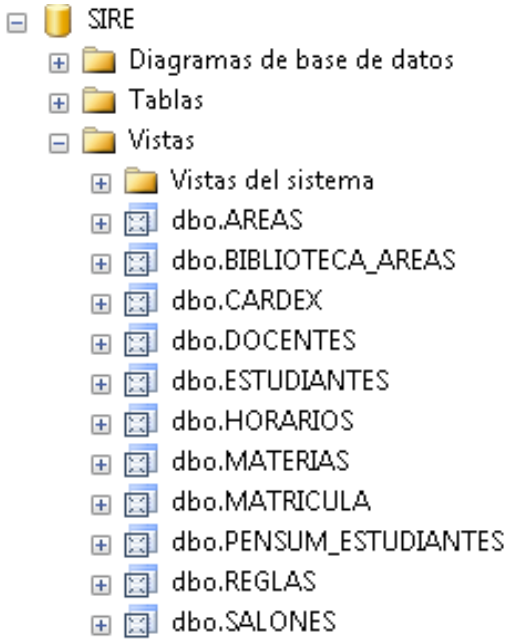


Fuente: elaboración propia

6.1.2. Construcción de tablas en la Base de Datos SIRE

Con el proveedor de se pueden llamar a las vistas directamente de Banner – UTB, las vistas configuradas en SQL Server a partir de Oracle son las mostradas en la Ilustración 7 – Vistas a partir de Banner:

Ilustración 8 - Vistas a partir de Banner



Fuente: Elaboración propia

- Vista AREAS: lista de las materias incluidas en cada semestre de los planes de estudio (pensum) de carrera.

```
 [SIRE].[dbo].[AREAS]
([PERIODO_PENSUM],[AREA],[DESCRIPCION],[SECUENCIA],[REGLA],[MATERIA],[CUR
SO],[TITULO])
```

- Vista BIBLIOTECA_AREAS: contiene la lista de semestres por planes de estudio.

```
 [SIRE].[dbo].[BIBLIOTECA_AREAS]
([AREA],[PROGRAMA],[COD_PROGRAMA],[SEMESTRE],[DESCRIPCION])
```

- Vista CARDEX: lista de notas definitivas de un estudiante.

```
[SIRE].[dbo].[CARDEX]
([ID],[PERIODO],[NRC],[MATERIA],[CURSO],[ASIGNATURA],[SESION],[CREDITOS],
[NOTA])
```

- Vista DOCENTES: datos de un docente discriminados por periodo.

```
 [SIRE].[dbo].[DOCENTES]
([PERIODO],[ID],[NOMBRES],[APELLIDOS],[TIPODOCENTE],[AÑOS_EXPERIENCIA],[T
ITULARIDAD],[TITULO],[COD_CATEGORIA],[CATEGORIA],[COD_ESCUELA],[ESCUELA],
[COD_DEPARTAMENTO],[DEPARTAMENTO],[COD_ESC_ANT],[ESCUELA_ANTERIOR],[COD_G
RADO],[GRADO])
```

- VISTA ESTUDIANTES: datos académicos de un estudiante

```
[SIRE].[dbo].[ESTUDIANTES]
([ID],[NOMBRES],,[APELLIDOS],[PROGRAMA],[PPA],[SITUACION],[COD_ESTATUS],[
ESTATUS],[COD_TIPO],[TIPO])
```

- Vista HORARIOS: horarios de clase disponibles en el periodo vigente.

```
[SIRE].[dbo].[HORARIOS]
([PERIODO],[NRC],[MATERIA],[CURSO],[SECCIÓN],[NOMBRE_ASIGNATURA],[LUNES],
[MARTES],[MIERCOLES],[JUEVES],[VIERNES],[SÁBADO],[DOMINGO],[HORA_INICIAL]
,[HORA_FINAL],[TIPO],[HRS_SEM],[CAPACIDAD],[OCUPADOS],[DISPONIBLES],[EDF]
,[SALON],[ID_DOCENTE],[DOCENTE])
```

- Vista MATERIAS: materias disponibles para matricular en la universidad.

```
[SIRE].[dbo].[MATERIAS]
([PERIODO_CREACION],[PERIODO_FINAL],[MATERIA],[CURSO],[CODESTATUS],[ESTAT
US],[CODESCUELA],[ESCUELA],[CODEPTO],[DEPARTAMENTO],[CODAPROBACION],[APRO
BACION],[UEC_O_CREDITOS],[FACTURADAS],[TEORIA],[LAB],[CONTACTO],[CODNIVEL
],[NIVEL],[CODMODO],[MODO_CALIFICACION],[CODTIPO],[TIPO],[CODMETODO],[MET
ODO])
```

- Vista MATRICULA: materias matriculadas por estudiante en un periodo específico.

```
[SIRE].[dbo].[MATRICULA]
([PERIODO],[NRC],[MATERIA],[CURSO],[SECCION],[NOMBRE_ASIGNATURA],[DOCENTE
ID],[ESTUDIANTEID])
```

- Vista PENSUM_ESTUDIANTES: relaciona el respectivo plan de estudios por estudiante.

```
[SIRE].[dbo].[PENSUM_ESTUDIANTES]
([ID],[PERIODO],[COD_PROGRAMA])
```

- Vista REGLAS: listado de cursos electivos definidos en el plan de estudios.

```
[SIRE].[dbo].[REGLAS]
([PERIODO_PENSUM],[AREA],[DESCRIPCION],[REGLA],[DESCRIPCION_REGLA],[MATER
IA],[CURSO],[TITULO])
```

- Vista SALONES: lista de aulas y laboratorios de los campus.

```
[SIRE].[dbo].[SALONES]
([EDIFICIO],[SALON],[PERIODO_DESDE],[PRIORIDAD],[DESCRIPCION],[CAPACIDAD]
,[MAXIMO],[STATUS])
```

Como vistas pueden ser consideradas tablas, previamente definida una tabla, se llenan los campos con las vistas asociadas. Para crear las tablas se decide diferenciarlas de las vistas con un guión de piso al final y copiar datos procedemos así:

- Crear tablas y definir campos

```
CREATE TABLE Tabla (<campo1, tipodato_campo1(longitud),>,<campo2,
tipodato_campo2(longitud),>, …)
```

- Insertamos datos:

```
insert into Horarios_ select * from HORARIOS
```

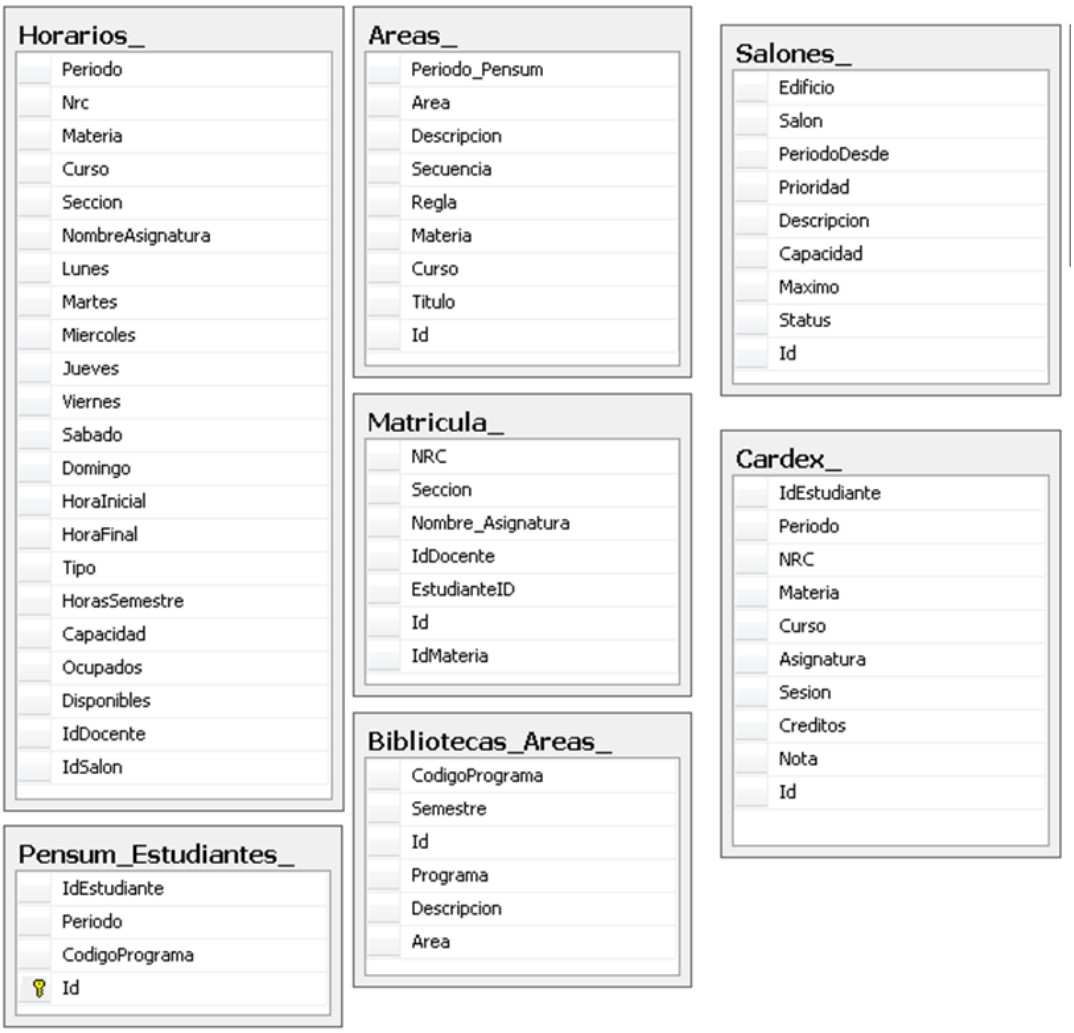
Posterior a la creación de las tablas se procede a normalizar la información sustraída de las vistas, algunos campos se suprimen de las tablas y se crean identificadores para cada una, llevando la tablas a 2NF como lo muestra la

Ilustración 9 - Tablas que se crearon a partir de las vistas extraídas desde BD Oracle Banner:

Ilustración 9 - Tablas que se crearon a partir de las vistas extraídas desde BD Oracle Banner

Fuente: Elaboración propia

Ilustración 9 – Continuación.



Fuente: Elaboración propia

6.1.3. Procedimientos de almacenado para Sincronización

Los procedimientos de almacenado se construyeron para verificar si la vista tiene datos y solo en este caso borra los datos de las tablas creadas, y copia los datos traídos desde Banner. El código genérico de un procedimiento de almacenado es así:

Ilustración 10 - Procedimiento de almacenado

```
ALTER procedure [dbo].[NOMBRE_PROCEDIMIENTO] AS
IF OBJECT_ID('identificador_VISTA') IS NOT NULL
BEGIN
SET NOCOUNT ON
-- Eliminamos los datos de la Tabla_
DELETE FROM [Tabla_];
-- Copiamos los datos de la VISTA a la Tabla_
INSERT INTO [Tabla_]
           ([campo1]
           ,[campo2])
SELECT [campo1]
           ,[campo2] FROM [VISTA];
END
```

Fuente: Elaboración propia

6.1.4. JOB en SIRE

Los trabajos – Jobs, permiten ejecutar procedimientos de almacenado de manera periódica, dichos jobs los ejecuta el servicio SQLServerAgent (SQL Server Agente) por lo que este servicio debe estar iniciado, para iniciarlo solo se requiere ir al Administrador corporativo y expandir los niveles por medio dela siguiente ruta: Servidores Microsoft SQL Server -> Grupo de SQL Server -> Administración -> SQL Server Agent nos permite iniciar la configuración de un job través de transact-sql con storedprocedures propias.

Los jobs e información relacionada con ellos se guardan en tablas propias del SQL Server en concreto en la base de datos msdb y en las tablas:

La periodicidad de la evaluación en los JOBs se índice es la siguiente lista:

- Vista AREAS: semestral
- Vista BIBLIOTECA_AREAS: semestral

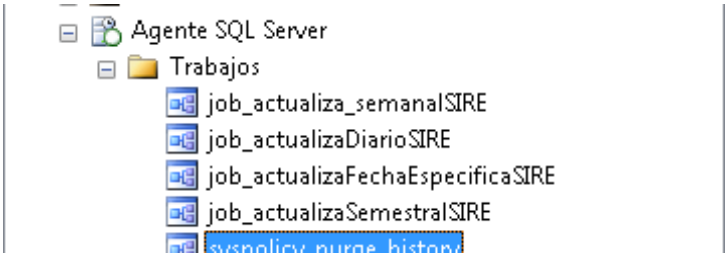
- Vista CARDEX: enero, junio, julio, diciembre (2 veces al mes)
- Vista DOCENTES: diario
- Vista ESTUDIANTES: diario
- Vista HORARIOS: semanal
- Vista MATERIAS: semestral (finales feb - jul)
- Vista MATRICULA:semanal.
- Vista PENSUM_ESTUDIANTES: semestral
- Vista REGLAS: semestral
- Vista SALONES: semestral

De esta manera se crearon 4 Jobs que agrupan los anteriores:

- job_actualizaDiarioSIRE: docentes y estudiantes
- job_actualiza_semanalSIRE: horarios y materias
- job_actualizaSemestralSIRE: areas, biblioteca_areas, pensum_estudiantes, reglas y salones
- job_actualizaFechaEspecificaSIRE: cardex y materiasl

Lo anterior se muestra en la ilustración 11 - jobs creados para actualizar SP:

Ilustración 11 – jobs creados para actualizar SP



Fuente: Elaboración propia

6.2. WEB SERVICES

MVS-2012 utiliza como marco de trabajo a Windows Communication Foundation (WCF) por lo que a partir de ahora será llamado al paquete principal como wcfSIRE.

Se crea un proyecto del Tipo WCF Service Library, en el que se debe definir previamente como se llevará la comunicación entre el proveedor y un consumidor del servicio: definición del Contrato.

6.2.1. Especificación de requerimientos para construcción de WS

Se desarrollará un servicio web que permita interconectar las bases de datos Oracle (Banner) con SQL Server (SIRE), dicho servicio permitirá validar el acceso de un usuario a SIRE identificando su existencia en Banner, devolver el rol de un docente que ingresa en el sistema y obtener los últimos datos actualizados de docentes o estudiantes en Banner.

El servicio debe ser construido en el framework .NET, utilizar un manejador que soporte el lenguaje SQL. La aplicación debe correr en background.

El servicio debe tener en cuenta los siguientes aspectos:

6.2.1.1. Usuarios del sistema

Aplicaciones en .NET: obtiene la identificación o permisos de acceso de sus usuarios.

6.2.1.2. Requerimientos funcionales

Requisito funcional 1 - RF1: el sistema permite verificar que un usuario tiene o no los privilegios de acceso a los datos de la base de datos a través de un login en el sistema Web.

Requisito funcional 2 - RF2: el sistema permite retornar el rol de un docente en el sistema

Requisito funcional 3 - RF3: el sistema permite devolver los datos actuales en Banner de los usuarios (Docente - Estudiantes).

6.2.1.3. Casos de uso del sistema

En los casos de uso del sistema se considera una solicitud inicial y una repuesta inmediata con acceso, rol o datos, por lo que no se consideran pasos de interacción usuario-sistema:

RF1 - Verifica privilegios de usuario – LOGIN	
Objetivo	El sistema permite verificar que un usuario tiene o no los privilegios de acceso en Banner para utilizar los datos de la base de datos SIRE a través de un login en el sistema Web.
Actor	Aplicaciones en .NET: - Aplicativo para Sombreado de pensum - Aplicativo para verificar prueba académica de estudiantes - Aplicativo para llevar el control de las actividades de los docentes y directivos docentes - Aplicativo para evaluar la intención de matrícula.
Salida	Permiso concedido – denegado

Precondición	ID-Usuario, Clave de acceso
Post-condición	Éxito: Permiso concedido Fallo: Permiso denegado

RF2–Retorna rol de docente	
Objetivo	El sistema retornar el rol de un docente en el sistema
Actor	Aplicaciones en .NET: - Aplicativo para llevar el control de las actividades de los docentes y directivos docentes
Salida	Rol del docente: decano, director, docente
Precondición	Usuario con permiso concedido
Postcondición	Éxito: Rol devuelto

RF2–Devolver datos actuales	
Objetivo	El sistema permite devolver los datos actuales en Banner de los usuarios (Docente - Estudiantes).
Actor	Aplicaciones en .NET: - Aplicativo para Sombreado de pensum - Aplicativo para verificar prueba académica de estudiantes - Aplicativo para llevar el control de las actividades de los docentes y directivos docentes - Aplicativo para evaluar la intención de matrícula.
Salida	Datos de docente/estudiante
Precondición	Usuario con permiso concedido
Postcondición	Éxito: datos devueltos

6.2.1.4. Requerimientos no funcionales

El servicio debe ser construido en el framework .NET. La clave entregada por los usuarios (aplicaciones) debe ser encriptada. Y finalmente el acceso a la base de datos en ORACLE debe ser restringido a la conexión login en JAVA previamente definida.

6.2.2. Modelo de análisis y diseño

Estructura del comportamiento del servicio web: En esta parte se determinaran los objetivos y límites del sistema a implementar haciendo énfasis en su estructura y funcionamiento. Se establecerán normas que permitan alcanzar los objetivos propuestos. También se creará el diseño conceptual de la información que se manejará, los componentes que se encargaran de su funcionamiento y las relaciones que existen entre uno y otro. Se mostrará una vista estática del sistema por medio de un diagrama de componente y una vista de la relación Hardware/Software usando diagrama de despliegue, detallando en su totalidad la comunicación entre los distintos objetos en tiempo de ejecución dando una idea de cómo será el funcionamiento del sistema.

Ilustración 12 - Diagrama de componente: se observa una aplicación (Explorador Web) que accede a un puerto o interfaz por medio de un HTTP a Web_Service_Autentication, en el cual se encuentran los 4 servicios prestados: Get_Docente, Get_Estudiante y Login, quienes acceden por un puerto a una interfaz de los elementos del componente, esto indica que los mensajes enviados al componente se administran en el elemento WS banner o sencillamente que los mensajes enviados desde el elemento se envían fuera del componente primario.

SireEntities por su parte representa un grupo de mensajes o llamadas que un componente implementa y que otros componentes o sistemas externos pueden utilizar.

Ilustración 13 – Diagrama de despliegue: se modela el hardware usado (dos servidores Sirius y Sistemas) cuya comunicación entre las bases de datos se da por un conector desarrollado en .NET que permite a SQL-SIRE conectarse a Oracle-Banner, y el WebServiceSIRE que hace llamadas específicas a artefactos en WebServiceBanner.

Ilustración 14 – Diagramas de procesos de negocio: está el despliegue de actividades donde primero se autentica, luego se verifica la autenticación a través de un subproceso en *Login Banner* descrito en la Ilustración 15, sólo si la respuesta es "Acceso Concedido" se sigue al subproceso *Obtener rol* para docentes descrito en la Ilustración 16, que remite un mensaje con el rol, este mensaje permite ingresar a una u otra aplicación del servidor sistemas.unitecnológica.edu.co

Ilustración 17 – Modelo del servicio: este modelo de datos se definió a partir de 4 tablas que son las necesarias para los requerimientos funcionales:

7. Docentes, que viene desde Banner
8. Estudiantes, , que viene desde Banner
9. Rol, que se encuentra en SIRE
10. Docente en Rol, que es una tabla resultado del cruce Docente-Rol para asignar roles a los docentes del sistema.

Se observó en las consultas de profesores y estudiantes que muchos campos duplicados limitaron la creación de llaves primarias y por ende la normalización de

la base de datos por fuera de la 2NF, sin embargo esta situación se controló desde el servicio tomando de IDs duplicadas la primera opción.

Ilustración 12 - Diagrama de componente

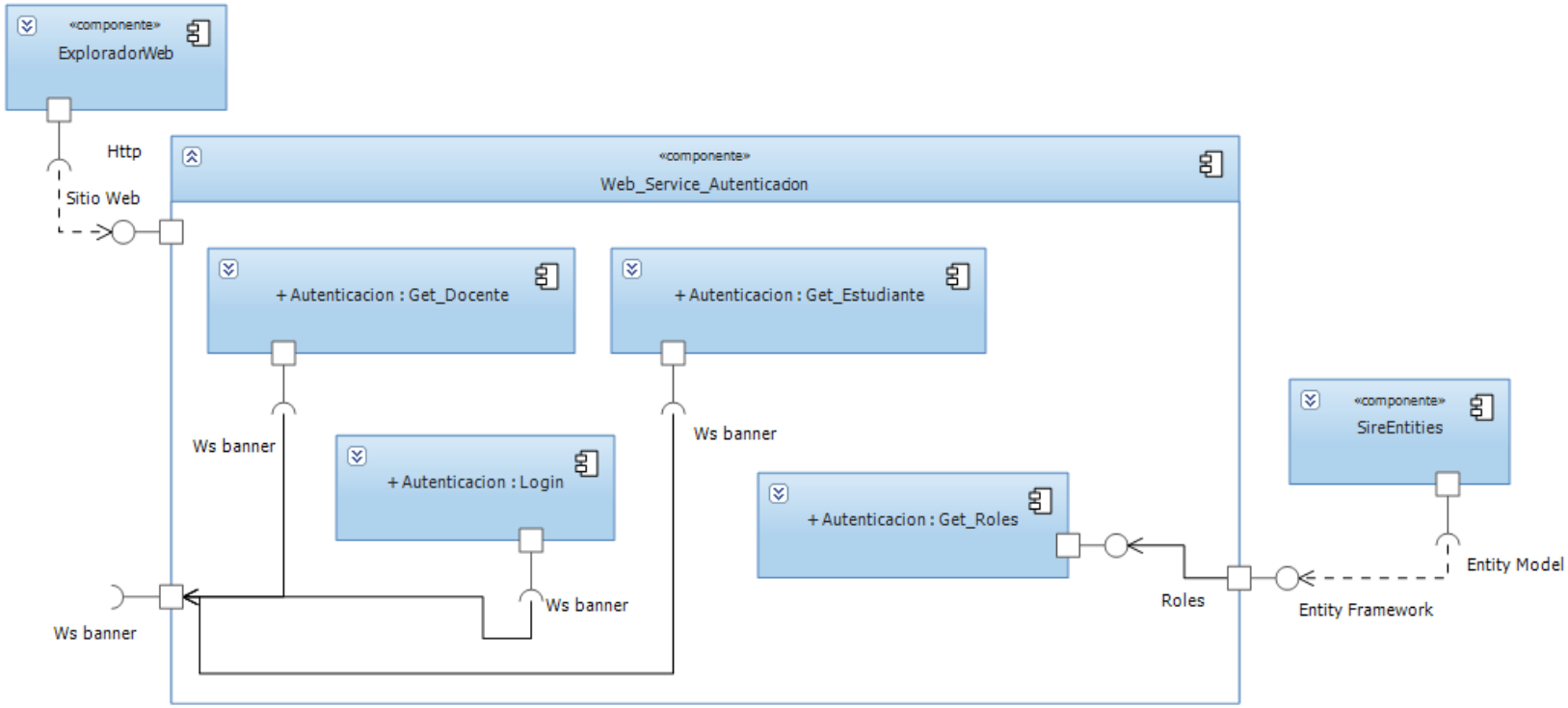


Ilustración 13 - Diagrama de Despliegue

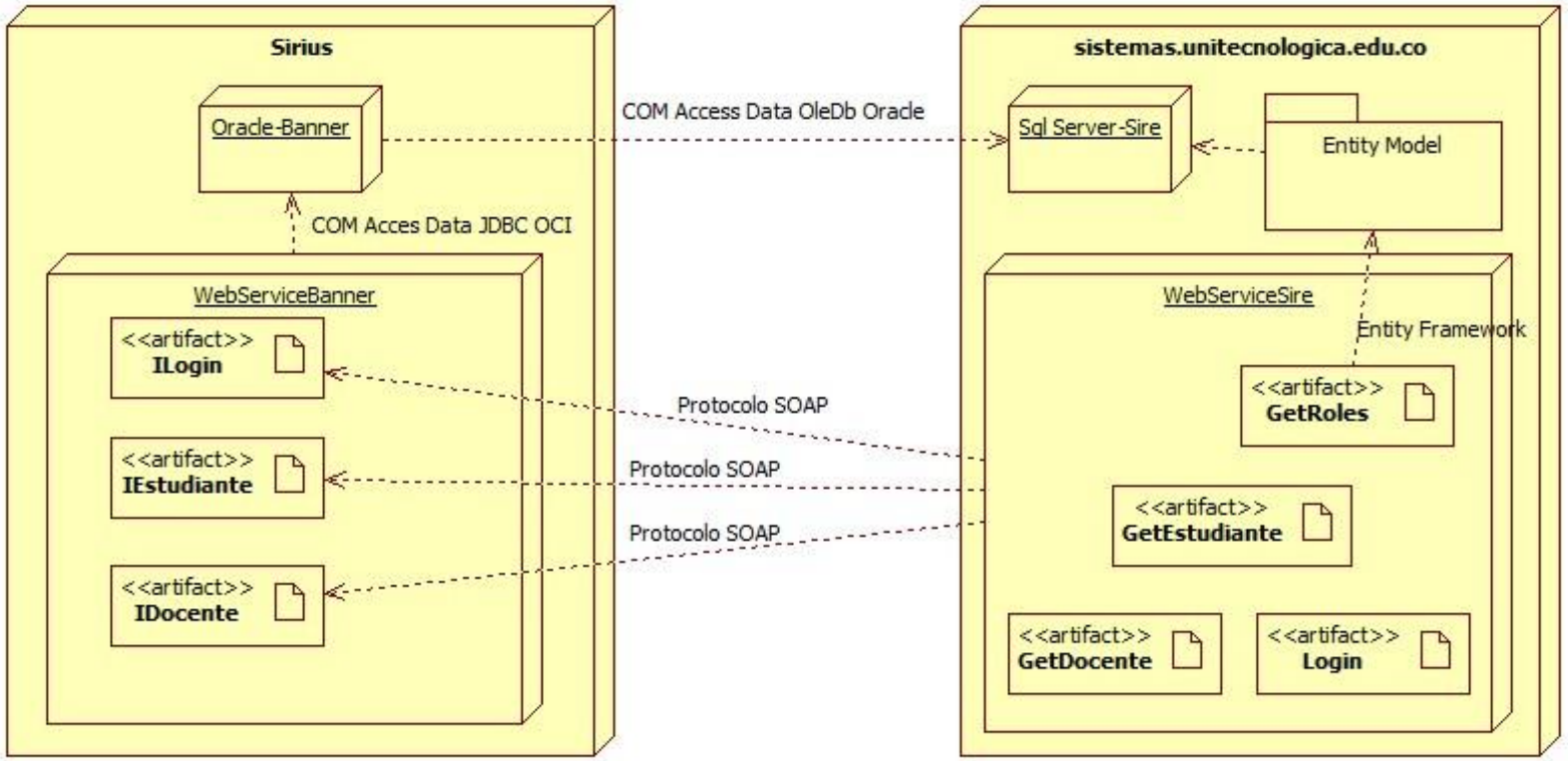


Ilustración 14 - Diagrama De Proceso De Negocio

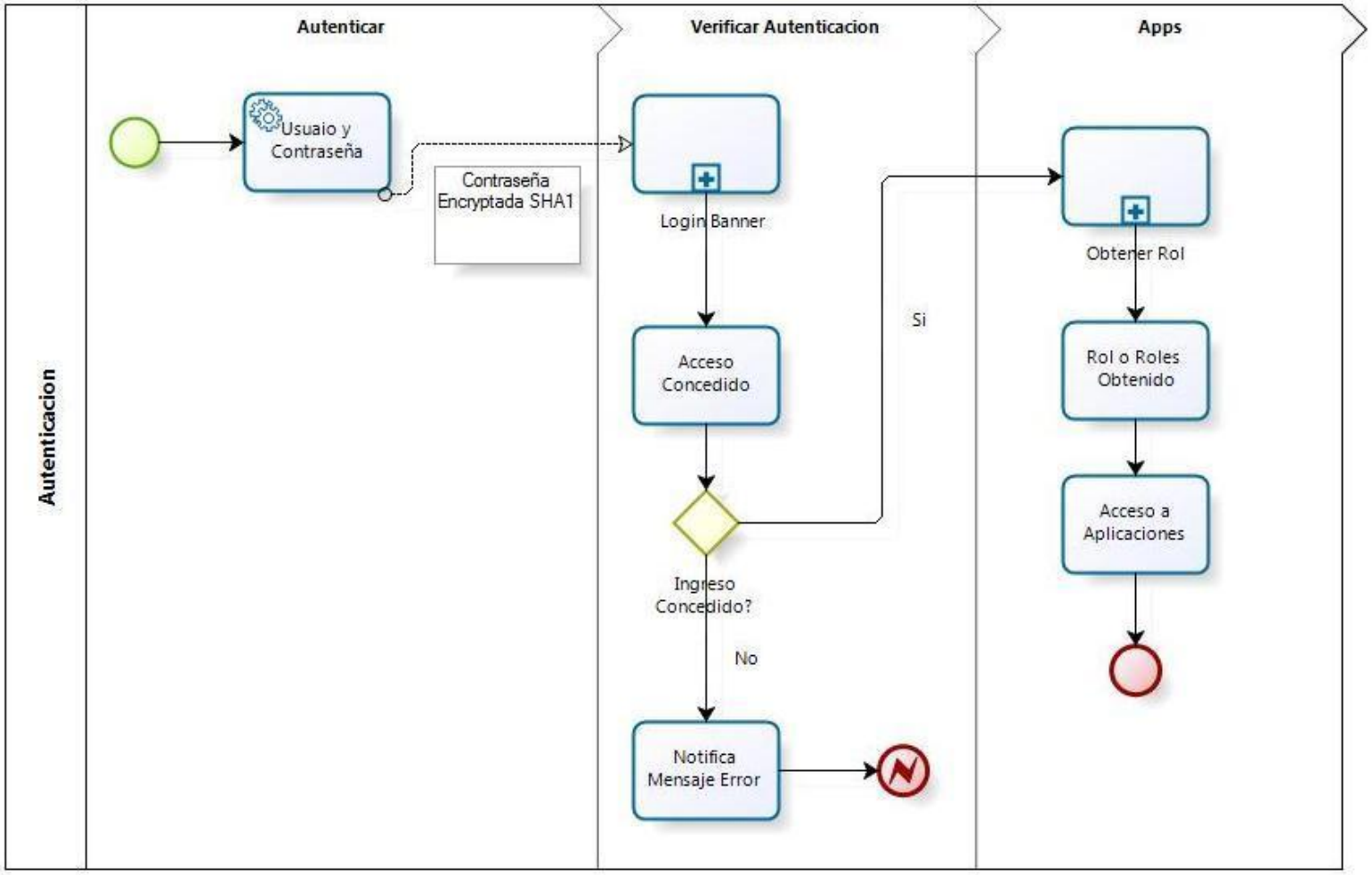


Ilustración 15 - Subproceso Login Banner

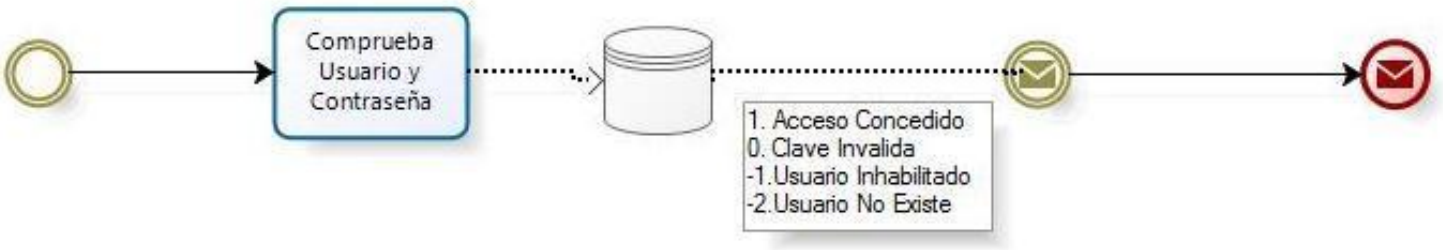


Ilustración 16 - Subproceso Obtener Rol

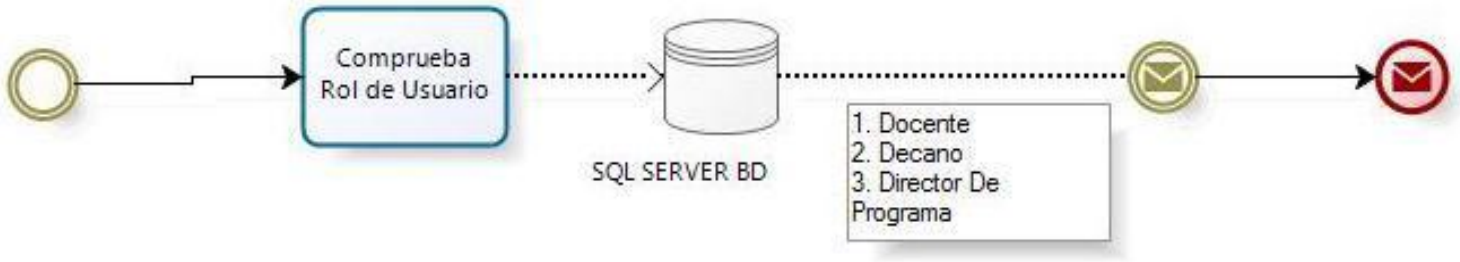
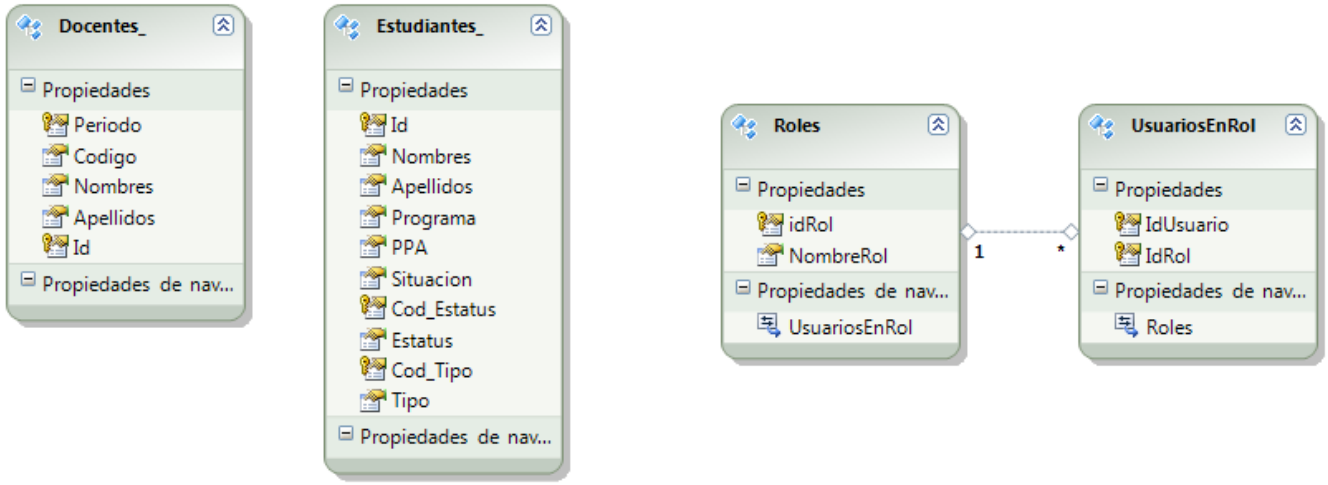


Ilustración 17- Modelo del servicio



6.2.3. Validación del sistema

Posterior a una revisión de algunas opciones para validar (9), la verificación y buen funcionamiento del sistema se prueba a través de una aplicación cliente creada en Visual Studio 2010 (8) por consola denominada TESTER: ApplicationConsole o Asp.Net obteniendo unas pruebas unitarias a cada uno de los subsistemas.

Para estas pruebas se realizan pruebas funcionales que consisten en el ingreso de un usuario y contraseña para obtener acceso al sistema, información del estudiante, docente, el rol o roles que obtiene cada usuario del sistema y poder acceder a las aplicaciones que están consumiendo el servicio planteado, así:

Ilustración 18 - Código del Tester para pruebas unitarias

```
[DataContract]
    public class CompositeType
    {
        bool boolValue = true;
        string stringValue = "Hello ";

        [DataMember]
        public bool BoolValue
        {
            get { return boolValue; }
            set { boolValue = value; }
        }

        [DataMember]
        public string StringValue
        {
            get { return stringValue; }
            set { stringValue = value; }
        }
    }
```

Fuente: Elaboración propia

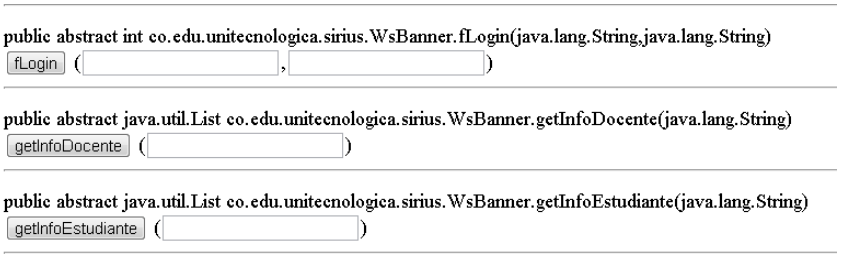
6.2.4. Construcción Web Service wcf Sire En La Plataforma De Desarrollo .NET

WCF (11) nos brinda un canal seguro de comunicación (a través de contratos) no solo entre aplicaciones de una misma máquina, sino a través de una red que puede estar conformada por servidores Windows o Linux (Interoperabilidad).

Primero se define el contrato, para establecer cómo se llevará la comunicación entre estos dos entes. Esto incluye: operaciones que se pueden realizar, firmas y los tipos de datos que se enviarán y/o recibirán durante las operaciones, protocolos y formatos de serialización que se utilizan para llevar a cabo la comunicación.

Posterior a la definición del contrato, y una vez creadas la interface y clase, es necesario publicarlo para que pueda ser consumido, esto lo hacemos colocando el nombre del servicio, donde esta publicado, y el tipo de enlace que permitirá. Esto implica que se incorpore la URI del servicio el cual se utilizo Web Service Banner realizado en la plataforma de desarrollo JAVA, el cual contiene los métodos: por medio del protocolo SOAP que permite obtener permisos para consultas en el sistema SIRIUS.
Los métodos disponibles para utilizar en Banner UTB se observan en la Ilustración 12 - métodos disponibles desde Banner en Java:

Ilustración 19 - métodos utilizados desde Banner en Java



Fuente: Elaboración propia

Tal como lo muestra la figura, fLogin recibe 2 cadenas y devuelve un entero (desde -2 a 1), getInfoDocente recibe un string y entrega una Lista (datos de los campos asociados al código del docente) y finalmente getInfoEstudiante recibe un string y entrega una Lista (datos de los campos asociados al código del estudiante).

Existen otros métodos a los que no se accede directamente para abrir, cerrar y verificar conexión cerrada.

A continuación se muestra el grafico en donde se implemento uno de los métodos que componen el Web Service de wcfSire para obtener la información del docente.

Ilustración 20 - ejemplo implementación método *getDocente* del servicio wcfSire

```
[OperationContract]
//Metodo para obtener la informacion del docente a traves de un ID de tipo string
public Docentes_ getDocente(string Id)
{
    //Se instancia la entidad por medio de un objeto context
    //context se asigna a un objeto de tipo List que filtra por medio de us id o codigo asignado en el sistema academico
    SireEntities context = new SireEntities();
    List<Docentes_> List = context.Docentes_.Where(t => t.Codigo == Id).ToList();
    if (List.Count > 0)// se pregunta si la cantidad es mayor que Zero, si es verdadero retorna el primer elemento
    {                  // de lo contrario valor nulo
        return List.First();
    }
    else
    {
        return null;
    }
}
```

Fuente: Elaboración propia

6.2.5. Modelo de Acceso a Datos ADO.NET Entity Framework

Visual Studio 2010 tiene incorporado Entity Framework el cual se utilizo para mapear la tablas que se encuentran realizadas en SQL SERVER 2008 R2 ya que .NET crea una clase entidad que contiene todo el esquema Orientado a Objetos.
A continuación se muestra de forma grafica como Entity Framework define su esquema de entidades orientado a objeto.

6.2.6. Seguridad con SHA1

La plataforma de desarrollo de Microsoft Visual Studio .NET contiene un namespace o biblioteca de clase llamada *System.Security.Cryptography* utilizada para encriptar la contraseña que viaja a través del protocolo SOAP para autenticarse en el sistema académico Sirius. Grafico de *Ilustración 15 - namespaces*:

Ilustración 21–namespaces

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;
using wcfSire.wsBannerService;
using wcfSire.Datos;
using System.Security.Cryptography;
```

Fuentes: Elaboración propia

A continuación se muestra el grafico con la implementación del método (7) que compone el Web Service de Sire para encryptar la clave de usuario.

Ilustración 22 - ejemplo del código del método de *generarClaveSHA1*

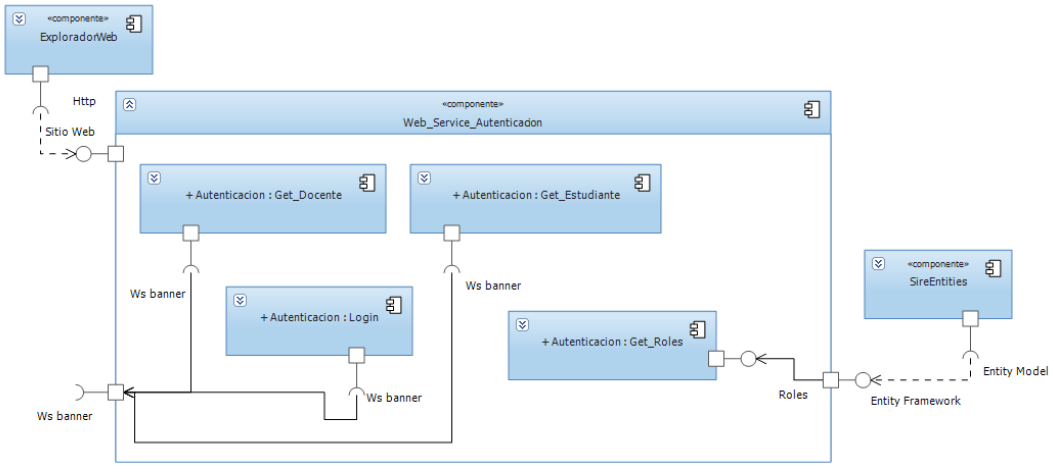
```
private string generarClaveSHA1(string nombre)
{
    // Crear una clave SHA1 como la generada por
    // FormsAuthentication.HashPasswordForStoringInConfigFile
    // Adaptada del ejemplo de la ayuda en la descripción de SHA1 (Clase)
    UTF8Encoding enc = new UTF8Encoding();
    byte[] data = enc.GetBytes(nombre);
    byte[] result;

    SHA1CryptoServiceProvider sha = new SHA1CryptoServiceProvider();
    // This is one implementation of the abstract class SHA1.
    result = sha.ComputeHash(data);
    //
    // Convertir los valores en hexadecimal
    // cuando tiene una cifra hay que rellenarlo con cero
    // para que siempre ocupen dos dígitos.
    StringBuilder sb = new StringBuilder();
    for (int i = 0; i < result.Length; i++)
    {
        if (result[i] < 16)
        {
            sb.Append("0");
        }
        sb.Append(result[i].ToString("x"));
    }
    //
    return sb.ToString().ToUpper();
}
```

Fuente: Elaboración propia

Los diagramas de componentes muestran elementos de diseño de un sistema de software por lo que permiten visualizar con facilidad la estructura general del sistema y el comportamiento del servicio que estos componentes proporcionan y utilizan a través de las interfaces. En la ilustración 17 - diagrama de componentes, se ilustra el diagrama de componentes del servicio web implementado:

Ilustración 23 - diagrama de componentes



Fuente: Elaboración propia

7. CONCLUSIONES

El conector OLE DB proporcionó la conectividad esperada en el entorno de servicios para interconectar las vistas con las tablas en SQL Server, por lo que el diseño de la base de datos SIRE pudo realizarse como fiel copia de las vistas provenientes de Banner.

Las tablas se crearon como fiel copia de las vistas, cabe destacar que en este aspecto la normalización de las bases de datos a 2NF no es necesaria ya que Entity solo extrajo los modelos de las tablas que directamente dependen se los servicios y aunque se obtienen de Banner campos duplicados de profesores y estudiantes, esto pudo ser controlado en la descripción de los servicios.

Ahora bien, se tienen algunas limitantes en lo concerniente a las actualizaciones en las vistas de Banner ya que pueden generar inconsistencias en los datos, este aspecto del diseño se controló limitando la actualización de tablas a la existencia de las vistas. En un futuro se pueden ampliar estas opciones verificando nuevos campos y alterando las tablas en la ejecución de jobs. De igual forma la inclusión de nuevas vistas se puede prever ante la creación de una nueva tabla al modelo.

El servicio en Java desarrollado por el departamento de sistemas facilitó el proceso de consultas en PL/SQL y a partir de éste se configuraron los servicios haciendo un llamado a 3 de sus métodos utilizando el algoritmo de encriptación SHA1 para la comunicación Banner – SIRE.

Por otra parte, la creación de la tabla ROL es un indicador de necesidades en la ampliación de algunos campos en Banner y de hecho en este mismo repositorio de datos centralizado se puede escalar a nuevas necesidades, previo análisis de

la seguridad en Banner, ya que un esquema en Oracle DB e incluso el manejo de vistas permite acceso a la colección de objetos directamente y las consultas a gran escala tienen un mayor nivel de rendimiento, como el control de usuarios de forma centralizada.

Finalmente se sugiere el estudio de la modificación de tablas y Stored Procedures de forma dinámica partiendo de actualización, inserción y/o eliminación de campos en el modelo de vistas, ya que el sistema actual no contempla estas opciones.

BIBLIOGRAFÍA

[1]. **W3C.** Web Services Description Language (WSDL) 1.1. [Online] 2001. http://www.cin.ufpe.br/~redis/intranet/bibliography/standards/wsdl-01.pdf.

[2]. —. World Wide Web Consortium (W3C). [Online] Mayo 2012. www.w3.org.

[3]. **Marston, Tony.** The Relational Data Model, Normalisation and effective Database Design. *Normalisation and effective Database Design.* [Online] Agosto 2005. http://www.tonymarston.co.uk/php-mysql/database-design.html.

[4]. **Pelzer, Peter Gulutzan & Trudy.** SQL-99 Complete, Really. [Online] 2010. http://kb.askmonty.org/en/sql-99-complete-really/.

[5]. **Wyllys, R. E.** DATABASE-MANAGEMENT PRINCIPLES AND APPLICATIONS - Steps in Normalization. [Online] 2003. http://www.gslis.utexas.edu/~wyllys/DMPAMaterials/normstep.html..

[6]. **Gerring, Taylor.** Connecting to Oracle from SQL Server. [Online] Enero 5, 2009. http://www.ideaexcursion.com/2009/01/05/connecting-to-oracle-from-sql-server/.

[7]. **Erl, Tomas.** *Service-Oriented Architecture Concepts, Technology, and Design. l.* s.l. : Prentice Hall, 2005. 978-0131858589.

[8]. **MSDN Library.** Información general de Entity Framework. [Online] 2012. http://msdn.microsoft.com/es-es/library/bb399567.aspx.

[9]. —. SHA (Clase). [Online] 2012. http://msdn.microsoft.com/es-us/library/system.security.cryptography.sha1.aspx.

[10]. **Serrano, Segundo.** ASP.NET - Herramientas para testear servicios web. [Online] Marzo 12, 2011. http://www.neuronasoft.net/2011/03/aspnet-herramientas-para-testear.html.

[11]. **Albahari, Joseph.** *C# 4.0 in a Nutshell.* s.l. : O'Reilly, 2010. 978059680095-6.

[12]. **Parasi, Victor.** Creando un proyecto WCF. [Online] Julio 20, 2010. http://copstone.com/2010/07/creando-un-proyecto-wcf/.