Universidad de las Américas

Ingeniería en Sistemas de Computación Informática

Desarrollo de soluciones inteligentes utilizando la plataforma puente de información y Visual Studio .Net 2003

Diego Alfonso Vega González

2007

Universidad de las Américas

Ingeniería en Sistemas de Computación Informática

Desarrollo de soluciones inteligentes utilizando la plataforma puente de información y Visual Studio .Net 2003

Trabajo de titulación presentado en conformidad a los requisitos
Para obtener el título de Ingeniero en Sistemas de Computación Informática

Ing. Santiago Albuja

Diego Alfonso Vega González

2007

Declaración del Profesor Guía

Yo, Ing. Santiago Albuja, con cédula de identidad No. 171024571-1, certifico haber dirigido el trabajo de titulación del alumno Diego Alfonso Vega González, bajo el título de "Desarrollo de soluciones inteligentes utilizando la plataforma puente de información y Visual Studio .Net 2003".

Ing. Santiago Albuja

Dedicatoria

A mis padres por el apoyo brindado
a lo largo de mi carrera profesional.

AGRADECIMIENTOS

Al Ing. Santiago Albuja por la ayuda prestada para la elaboración de este trabajo de titulación, y su apoyo incondicional a lo largo de mi carrera.

Índice

1. CAPÍTULO I 8
1.1. Planteamiento del problema 8
1.2. Formulación del problema 9
1.3. Sistematización del problema 12
1.4. Objetivos 12
1.4.1. Objetivo general 12
1.4.2. Objetivos específicos 12
1.5. Justificación 13
1.6. Introducción al IBF 13
1.7. Requisitos para el desarrollo de una aplicación IBF. 14
1.8. Arquitectura 15
1.8.1. Componente del Cliente 16
1.8.2. Componente de servidor 17
1.8.3. Diseñador de Metadata (MetaData Designer) 17
1.9. Flujo de Información y Datos 18
1.10. Metadata 19
1.10.1. Jerarquía de la Metadata 19
1.10.2. Metadata Scopes 19
1.10.3. Entities 20
1.10.4. Ports 20
1.10.5. Schemas 20
1.10.6. Operations 21
1.10.7. Transformations 22
1.10.8. Translations 22
1.10.9. Groups 23
1.10.10. Operations Types 23
1.10.11. OperationBrowse 23
1.10.12. OperationCallComponent 23
1.10.13. OperationExecuteAction 24
1.10.14. OperationNavigate 24
1.10.15. OperationQueryMenu 24
1.10.16. OperationReturnActionResult 24
1.10.17. OperationSoapRequest 25
1.10.18. OperationStatusCheck 25
1.10.19. OperationTransformationAggregation 25
1.10.20. OperationCustom 25
1.11. Planificación de una solución IBF 26
1.11.1. Planificación de las entidades. 27
1.11.2. Planificación de los servicios Web 27
1.11.3. Planificación de la Metadata. 27
1.11.4. Planificación de la Interfase de Usuario 28
1.11.5. Planificación de seguridad 28
1.11.6. Planificación de liberación 28
1.12. Instalación del IBF 29
1.12.1. Instalación del Servidor 29

1.12.2. Instalación de componentes de desarrollo 30
1.12.3. Instalación del componente del Cliente 30
2. CAPÍTULO II - Conectando los servicios de la línea de negocio 31
2.1. Análisis 31
2.2. Definición de las entidades 33
2.3. Definición de las vistas 34
2.4. Definición de Referencias: 36
2.4.1. Definición de operaciones 36
2.4.2. Operaciones del tipo Get. 37
2.4.3. Operaciones del tipo Put. 37
2.4.4. Operaciones del tipo Act. 38
2.5. Desarrollo del Servicio Web 38
2.5.1. Construcción de un servicio Web para una operación del tipo Get 39
2.5.2. Creación de los Objetos de Negocio 41
2.5.3. Creación del Objeto Estudiante 41
2.5.4. Creación de la colección Estudiantes 45
2.5.5. Creación de la capa de datos. 46
2.6. Creación del Servicio Web 48
3. CAPÍTULO III - MetaData 51
3.1. Que es la MetaData? 51
3.2. Desarrollo de la MetaData. 51
3.2.1. Creación del servicio de la metadata 54
3.2.2. Importando la Metadata 54
3.2.3. Creación de Entidades, Vistas, y localizadores 58
3.3. Creación de la solución de MetaData 63
3.3.1. Creación de Regiones 64
3.4. Creación de regiones del tipo lista de Referencia. 72
3.5. Definición de menús 77
3.6. Definición de búsquedas. 85
4. CAPÍTULO IV - Ejecución de la solución IBF y Etiquetas Inteligentes. . 92
4.1. Configuración de ensamblados seguros 92
4.2. Ejecución de la solución IBF 95
4.3. Desarrollo de Etiquetas Inteligentes (Smart Tags - IBF) 98
4.3.1. Implementación del Reconocedor 101
4.3.2. Definición de Referencias 101
4.3.3. Reconociendo las etiquetas inteligentes 102
4.4. Otras configuraciones 107
CONCLUSIONES 108
RECOMENDACIONES 109

Índice de Figuras

Figura 1.1 Requisitos para el desarrollo de una aplicación IBF..... 15
Figura 1.2 Representación gráfica de la arquitectura del IBF..... 16
Figura 1.3 Flujo de información y datos en la petición al IBF..... 18
Figura 3.1 Creación del nuevo proyecto de metadata..... 52
Figura 3.2 Proyecto de metadata creado en el explorador de solución..... 53
Figura 3.3 Pantalla correspondiente a la guía de metadata..... 53
Figura 3.4 Pantalla para la importación de la metadata desde un servicio Web 55
Figura 3.5 Pantalla de la selección de operaciones..... 56
Figura 3.6 Pantalla para adjuntar o remplazar la metadata importada..... 56
Figura 3.7 Pantalla para cambiar el nombre de la metadata..... 57
Figura 3.8 Asistente de creación del servicio de metadata..... 58
Figura 3.9 Creación de una entidad..... 59
Figura 3.10 Pantalla para la definición de vistas..... 60
Figura 3.11 Selección del identificador o localizador de la vista..... 61
Figura 3.12 Resumen de los elementos creados por el asistente..... 62
Figura 3.13 Elementos creados por el asistente en formato de jerarquía de metadata..... 62
Figura 3.14 Representaciones en el explorador de metadata..... 63
Figura 3.15 Guia de metadata, asistentes para la creación de la solución de metadata..... 64
Figura 3.16 Inicio del asistente para la creación de regiones..... 65
Figura 3.17 Definición de esquemas..... 66
Figura 3.18 Creación de acciones..... 67
Figura 3.19 Tipos de región..... 68
Figura 3.20 Propiedades de la región..... 69
Figura 3.21 Resumen del asistente..... 70
Figura 3.22 Configuración avanzada..... 70
Figura 3.23 Control de usuario creado por el asistente..... 71
Figura 3.24 Acciones creadas en la metadata..... 72
Figura 3.25 Inicio del asistente para la creación de lista de referencia..... 73
Figura 3.26 Definición de esquema..... 74
Figura 3.27 Propiedades de la lista de referencia..... 75
Figura 3.28 Propiedades de la región..... 75
Figura 3.29 BizTalk en IBF..... 76
Figura 3.30 Región lista de referencia..... 77
Figura 3.31 Pantalla para la creación de transformaciones..... 79
Figura 3.32 BizTalk para transformación de esquemas..... 80
Figura 3.33 Pantalla para añadir una relación..... 81
Figura 3.34 Pantalla inicial del asistente de creación de menús..... 82
Figura 3.35 Definición de la solución de metadata..... 83
Figura 3.36 Definición de contexto..... 83
Figura 3.37 Propiedades del menú..... 84

Figura 3.38 Resumen de las propiedades creadas por el asistente 84
Figura 3.39 Finalización del asistente 85
Figura 3.40 Pantalla de inicio del asistente de definición de búsqueda 86
Figura 3.41 Definición de metadata 87
Figura 3.42 Definición de localizador y acciones para búsqueda 88
Figura 3.43 Esquemas de entrada 88
Figura 3.44 Tipo de búsqueda 89
Figura 3.45 Propiedades de texto 90
Figura 3.46 Definición del criterio de búsqueda 90
Figura 3.47 Finalización del asistente 91
Figura 4.1 Pantalla de inicio del asistente de configuración de .NET framework 1.1 93
Figura 4.2 Configuración del alcance de la seguridad 94
Figura 4.3 Pantalla para establecer el ensamblado en cuestión 94
Figura 4.4 Pantalla donde se establece el nivel de confianza 95
Figura 4.5 Ejecución de la solución IBF 96
Figura 4.6 Parametro de referencia XML 97
Figura 4.7 Pantalla del panel IBF 97
Figura 4.8 Ejemplo de la activación de una etiqueta inteligente 98
Figura 4.9 Acciones por defecto de una etiqueta inteligente 99
Figura 4.10 Reconocedor creado 105
Figura 4.11 Elemento reconocido como matricula de un estudiante 106
Figura 4.12 Despliegue de la vista estudiante en el IBF mediante una etiqueta inteligente 106

1. CAPÍTULO I

1.1. Planteamiento del problema

Las soluciones inteligentes fueron creadas en un inicio para facilitar el uso de las herramientas Office mediante ayudas y motores inteligentes, ahora estas tecnologías están siendo usadas por grandes empresas como Google y Amazon para interconectar sus productos y servicios con las herramientas Office System 2003.

Actualmente es posible para las empresas programar sus propias soluciones inteligentes con herramientas Microsoft y conectarlos a sus fuentes de datos. Hoy en día es más importante una información de calidad que simples conjuntos de datos.

Imagine tener a mano la información de la empresa, de los empleados, de las ventas, compras, etc., en un mismo entorno con el cual ya está familiarizado como Microsoft Word, Excel o Microsoft Outlook. Esto implica menores costos para el desarrollo o compra de aplicaciones que den la misma información, mayor productividad, ya que la búsqueda y obtención de la información se hace prácticamente de forma directa por parte del usuario.

Hoy en día mucha gente emplea grandes cantidades de tiempo en las suites de oficina siendo una de las más utilizadas los Sistemas Microsoft Office. Empresarios, Financieros, Educadores, etc. pueden estar un día entero utilizando herramientas como Word, Excel, PowerPoint o Outlook,

las cuales les brindan soluciones genéricas para sus respectivas necesidades, aun siendo Microsoft Office una gran herramienta no deja de ser genérica. Para algunos usuarios esto no es suficiente y requieren aspectos específicos que se puedan integrar como una extensión de los Sistemas Office.

Este trabajo investigativo se enfocará en explicar una tecnología que puede ser integrada a los Sistemas Office 2003, y que al utilizarlos darán mayor productividad a los usuarios.

Uno de los beneficios de esta tecnología es permitir a los usuarios el acceso a la información de diferentes sistemas sin necesidad de salirse de su entorno de trabajo y de esta forma facilitar la toma de decisiones.

1.2. Formulación del problema

Actualmente con la evolución del desarrollo de software el uso múltiples sistemas en las organizaciones es común.

Normalmente un solo empleado utiliza varios programas para realizar su trabajo. El crecimiento constante que ha tenido Microsoft llevo a las organizaciones a emplear el uso de los sistemas Office como parte fundamental en el desempeño de funciones en una empresa.

Esto lleva a que los usuarios busquen información en los sistemas propios de la empresa y los lleven a su entorno de trabajo (Sistemas Office) ya sea

para realizar un informe, realizar cálculos o enviar esa información vía correo electrónico.

Para tener una mejor visión se propone el siguiente escenario:

El decano de la universidad recibe un correo electrónico por parte de un estudiante, en el cual explica un problema que ha tenido al querer revisar sus notas mediante la página web de la universidad, el estudiante adjunta su número de matricula al final del correo electrónico. El decano de la universidad sabe que las razones por la cual un estudiante no tendría acceso a revisar sus notas son las siguientes:

- No esta matriculado
- Tiene pagos pendientes
- Tiene un libro prestado por la biblioteca de la universidad
- No ha rendido un último examen.

El sistema en cual el decano de la universidad tiene acceso, solo respondería la última pregunta, solo podría ver si el estudiante no ha rendido un último examen.

Para responder las otras interrogantes, el decano tendría que comunicarse con los encargados de las otras áreas en este caso contabilidad y biblioteca respectivamente, para verificar si el estudiante ha faltado en alguno de estos requisitos.

Esto implica una baja de productividad, tanto del decano como de las personas a las cuales solicita información sobre el estudiante.

Ahora con este mismo escenario se plantea lo siguiente:

El decano de la universidad recibe el mismo correo electrónico con la misma petición. Cuando el estudiante escribió su número de matricula al final del documento, este es reconocido automáticamente por Outlook, advirtiendo al decano que ese es un número válido de matrícula de cierto estudiante.

El decano procede a activar las opciones de ese texto reconocido y a desplegar la información completa de ese estudiante en el panel de tareas del Outlook, teniendo a la mano información tanto académica, financiera y conectando al sistema de prestamos en biblioteca.

Y de esta manera dar una respuesta ágil al estudiante, sin necesidad de pedir información a otros funcionarios y sin necesidad de salirse de su entorno de trabajo, en este caso Microsoft Outlook.

Con esta realidad Microsoft ha desarrollado una plataforma llamada Information Bridge Framework (IBF) que ofrece todo un conjunto de herramientas de programación para conectar la información de línea de negocio sin importar la fuente de datos con los sistemas Microsoft Office permitiendo exponer datos relevantes al negocio. Permitiendo así tener la información disponible desde cualquier programa de Office.

¿Cómo y de qué forma aportaría una guía práctica para el desarrollo de tecnologías que integren la información del negocio de una forma más eficiente?

1.3. Sistematización del problema

¿Cómo aportaría para un desarrollador el tener acceso a una guía práctica de desarrollo de esta nueva tecnología?

¿En qué influiría el hacer una demostración piloto de lo que sería un aumento en la productividad y el fácil acceso a datos por parte de altos directivos?

¿Cómo ayudaría este trabajo de titulación como una guía para desarrollar potentes herramientas de línea de negocio?

1.4. Objetivos

1.4.1. Objetivo general

Crear una guía práctica para mostrar como desarrollar, implementar y aprovechar las características de la plataforma de puentes de negocio (IBF – Information Bridge Framework) de Office System 2003 para integrarlo con los datos del negocio de las organizaciones para obtener una mayor productividad.

1.4.2. Objetivos específicos

- Desarrollar una guía para el programador del desarrollo de soluciones de inteligencia de negocio utilizando IBF.

- Mostrar como se puede aumentar la productividad en ciertas tareas con el uso de IBF.
- Analizar las ventajas de conectar la línea de negocio a los sistemas Office 2003.
- Desarrollar una aplicación para la demostración de una solución IBF, utilizando la guía práctica expuesta.

1.5. Justificación

El uso de la plataforma – puentes de información (IBF) propone una interesante y útil forma de integrar los datos del negocio a las aplicaciones de uso común de Office System 2003, esta integración supone un aumento de productividad al usuario ya que puede acceder a los datos pertinentes desde un mismo entorno. El uso de esta tecnología es poco conocida y difundida en el medio, tanto la parte de desarrollo, integración y uso, es por tal razón que el presente tema de titulación busca cubrir estos aspectos, y ser una guía para empezar a conocer esta tecnología y aumentar la productividad de los usuarios del Sistema Office 2003.

1.6. Introducción al IBF

La plataforma – puentes de información (IBF por sus siglas en inglés Information Bridge Framework), es una de las respuestas de Microsoft para resolver los problemas de los negocios inteligentes. IBF es una tecnología que provee una puerta de enlace entre la Línea de Negocio (LOB por siglas en inglés line-of-business) y los documentos Office. IBF

es un conjunto de herramientas, aplicaciones, servicios e interfases que permiten al usuario tener acceso a la información del negocio desde documentos Office, incluyendo documentos Word, Excel, InfoPath, Outlook y paginas Web en Internet Explorer.

1.7. Requisitos para el desarrollo de una aplicación IBF.

- Servicios de Internet IIS 6.0
- Visual Studio .NET 2003 Edición Profesional.
- .NET Framework 1.1 SP1
- Microsoft Office 2003 SP1
- SQL Server 2000 SP3a
- Servicio de IBF 1.0
- Cliente IBF 1.5
- MSXML (Microsoft XML Core Services)
- Diseñador de Metadata del IBF 1.5

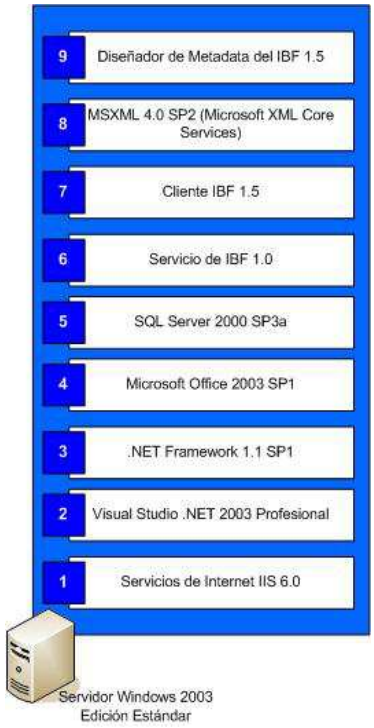

Figura 1.1 Requisitos para el desarrollo de una aplicación IBF

1.8. Arquitectura

La arquitectura del IBF se divide en los siguientes componentes:

- Cliente
- Servidor
- Diseñador de Metadata.

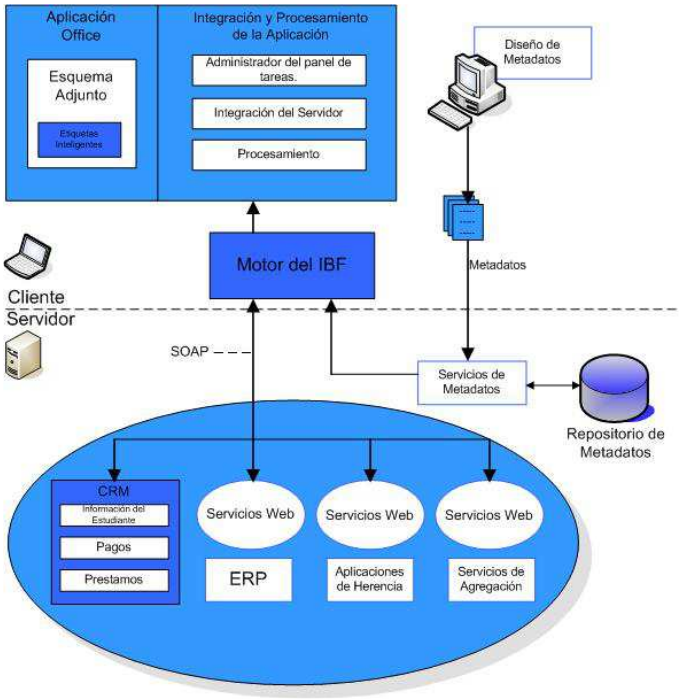

Figura 1.2 Representación gráfica de la arquitectura del IBF

1.8.1. Componente del Cliente

El componente del cliente es el encargado de mostrar al usuario final la información contextual sobre los datos del negocio, conectándose a la línea de negocio a través de los servicios Web o mediante CLR (Common language runtime).

El componente del cliente es representado como una ventana en el panel de acciones del documento de Office, este accede a los servicios de línea del negocio invocando los servicios de Metadata. Este es un servicio Web XML que retorna la información necesaria que describe la forma en que el componente del cliente debe conectarse y desplegar la información al usuario final.

El servicio Web usa directivas de autorización para la seguridad. Estas directivas son simplemente el nivel de seguridad que se le da a un ensamblado. Para conocer como modificar el nivel de seguridad en un ensamblado ver capitulo IV.

1.8.2. Componente de servidor

El componente de servidor es el que proporciona la Metadata que permiten que el componente cliente pueda acceder a los servicios Web. Estos presentan los datos, las vistas, y las acciones de los servicios de línea de negocio.

1.8.3. Diseñador de Metadata (MetaData Designer)

El diseñador de Metadata es un complemento de Visual Studio que ayuda a los desarrolladores a crear la Metadata.

La Metadata describe que acciones puede tomar el componente cliente con respecto a los datos de línea de negocio, por ejemplo: consulta, manipulación y vistas de los datos.

Una vez creada la Metadata, puede publicarse mediante un servicio Web hacia la base de datos de Metadata y así el componente del cliente puede acceder a este servicio Web para determinar que elementos de las interfases de usuario mostrar para cada acción y realizar las llamadas hacia los servicios “backend” cuando una acción es requerida.

El desarrollador también tiene la responsabilidad de crear las interfases de usuario para el componente del cliente, y debe acoplar la interfaz con la acciones de la Metadata.

1.9. Flujo de Información y Datos

Una solicitud hacia el IBF normalmente comienza con el reconocimiento de texto etiquetado en un documento de Office, ha este texto etiquetado se lo conoce como Etiquetas Inteligentes (Smart Tags). Sin embargo un vínculo en el Explorador de Internet también puede levantar una solicitud hacia el IBF.

En la figura 1.3 se muestra el flujo de información y datos después de una solicitud al IBF. Este será explicado en detalle en un capítulo posterior.

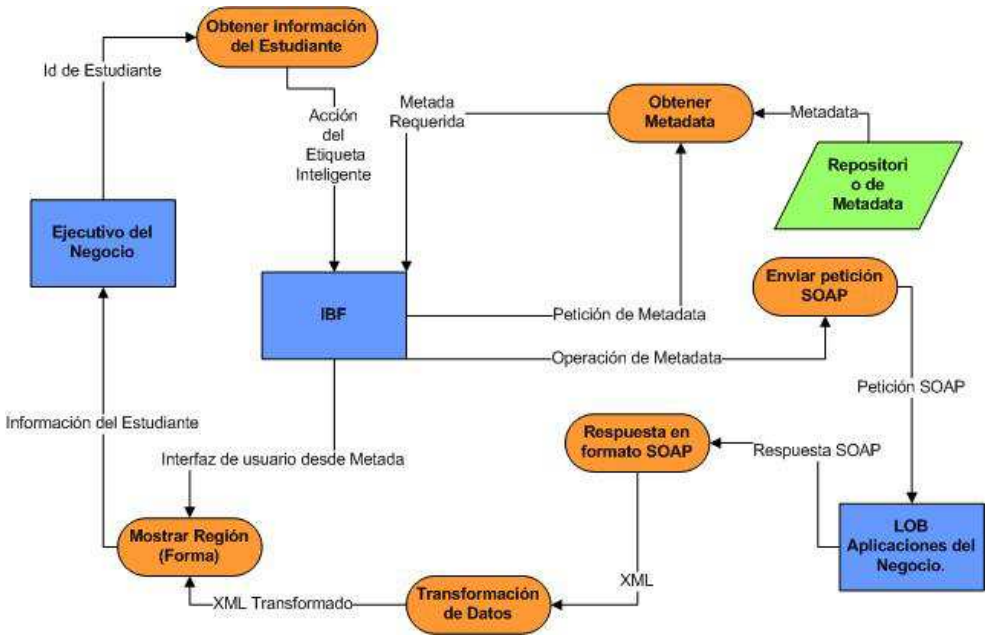

Figura 1.3 Flujo de información y datos en la petición al IBF

.

1.10. Metadata

La parte más importante de las soluciones IBF es la Metadata. La Metadata se divide en dos tipos:

- Servicios de Metadata.
- Soluciones de Metadata.

Los servicios de Metadata describen como el IBF deber conectarse con el “backend” de los servicios de línea de negocios. Por ejemplo, cuales operaciones SOAP llamar.

Las soluciones de Metadata describen como IBF debe usar los datos recibidos de los servicios de línea de negocio para ser desplegados en la interfase de usuario final.

1.10.1. Jerarquía de la Metadata

La jerarquía de la Metadata son elementos que se pueden usar para la creación de una solución IBF. Por tratarse de elementos de palabras reservadas estas se encuentran en idioma Inglés. Se definirá uno a uno estos elementos manteniendo sus nombres.

1.10.2. Metadata Scopes

El elemento Metadata Scopes o Alcance de Metadata provee la separación lógica de las áreas del negocio. Por ejemplo, se puede

separar el área de Recursos Humanos, Admisiones, Decanato, etc. Cada área de alcance tendrá sus propios elementos de jerarquía de la Metadata.

1.10.3. Entities

Entities o entidades son el equivalente de los objetos del negocio en una organización. Estas representan personas, lugares o cosas que son usadas en el contexto de las aplicaciones de la línea de negocios.

Las entidades son creadas para representar objetos del negocio a un alto nivel. Por ejemplo, el decanato requiere información detallada del Estudiante y las asignaturas que cursa, en cambio el departamento de contabilidad solo requiere información de pago del Estudiante.

Una entidad puede ser creada con múltiples datos de la línea de negocio.

1.10.4. Ports

Ports o puertos, este elemento es el encargado de especificar la fuente de los datos, como por ejemplo, archivos XML, XSLT, ensamblados .net, o servicios Web.

1.10.5. Schemas

Schemas o Esquemas, definen la estructura que es usada en la solución IBF. Esta incluye la estructura tanto de los datos de la línea de negocio como los datos del IBF. Los esquemas definen los parámetros de entrada y salida para las operaciones. Cuando la validación del esquema es encendida, los esquemas también validan los datos que son recibidos

y retornados por las operaciones. Cada esquema es enlazado al elemento “Port” que contiene los datos donde se define realmente el esquema.

1.10.6. Operations

El elemento Operations u Operaciones, representa una tarea individual y atómica que se levanta como una parte de una acción en particular. Por ejemplo, si el Decano de la facultad quiere ver información de un estudiante en particular, este realizará una sola acción la cual consiste en las siguientes operaciones en secuencia:

- Invoca un método en un servicio Web para consultar la información.
- Despliega la información en la ventana del IBF.

Cuando exista una operación como parte de una acción, esta operación es considerada como una instancia. En otras palabras, las operaciones por si mismas son abstractas.

Así como las operaciones definen los esquemas de entrada y salida para los parámetros y el valor de retorno, una instancia de la operación puede definir exactamente cuales datos son los que van a ser pasados como entrada a una operación, y también las transformaciones a ser usadas para los datos.

Los datos de entrada pueden ser, datos de vista, datos de referencia, datos de acción, o datos de salida de otra operación existente.

1.10.7. Transformations

Transformations o transformaciones, es el elemento que permite transformar los datos antes de ser usados. Algunos tipos de transformaciones incluyen expresiones regulares como XPath y XSLT.

XPath, es un lenguaje que sirve básicamente para navegar entre los elementos y atributos de un documento XML.

XSLT, es un lenguaje que permite transformar un documento XML en otro utilizando XPath.

Las trasformaciones también son abstractas como las operaciones. Las instancias de las trasformaciones existen cuando una transformación ha sido aplicada a una instancia de una operación.

En si las transformaciones definen los esquemas de entrada y salida afectados después de una transformación, las instancias de una transformación definen los datos a ser transformados. Una instancia de una transformación puede ser aplicada a una instancia de operación por cada parámetro de entrada que la operación acepte.

1.10.8. Translations

Translations o traductores, permiten traducir los datos de un lenguaje a otro. Ej. Un esquema que transforme del idioma Inglés al español.

<Translations>
<Translation Key = "Student Name" Modifier = "en-us" Value = " Student Name" IsFallback = "true"></Translation>
<Translation Key = " Student Name" Modifier = "es" Value = "Nombre del estudiante" IsFallback = "false"></Translation>
</Translations>

1.10.9. Groups

El elemento Groups o Grupos, permiten asignar permisos a las operaciones. Estos existen como definición de tareas en el Manejador de Autorización (“Authorization Manager”). Se pueden asignar estas tareas a roles y dar el permiso a usuarios de Windows y cuentas de grupo.

1.10.10. Operations Types

El elemento Operation Types o Tipos de Operación, debe ser asociado para cada una de las operaciones definidas anteriormente. Esto se debe a que el IBF es el encargado de invocar las operaciones, este debe saber como hacerlo. A continuación se explica cada uno de estos tipos.

1.10.11. OperationBrowse

Este tipo de operación permite recopilar datos desde una ubicación HTTP. Por ejemplo, se puede recopilar información desde archivos XML, archivos de texto, o archivos binarios.

1.10.12. OperationCallComponent

Las operaciones de este tipo pueden invocar los métodos de un ensamblado .net a un nivel de clases. El IBF usa reflexión CLR para invocar estos métodos. Para realizar estas operaciones se debe especificar el puerto que contiene la ubicación del ensamblado, la clase que contiene al método, y el nombre del método a invocar. Los parámetros y el valor de retorno deben ser serializados.

1.10.13. OperationExecuteAction

La operación de este tipo puede ejecutar las acciones definidas en la Metadata. Para esta operación, se debe especificar el destino del alcance, entidad, vistas, esquema de referencia y el nombre de la acción. El esquema de referencia determina como los datos van a ser recolectados.

1.10.14. OperationNavigate

Permite la navegación de una vista a otra usando las relaciones definidas en la Metadata. Las relaciones describen como las vistas están asociadas unas con otras.

1.10.15. OperationQueryMenu

Este tipo de operación recopila las definiciones de un menú para las acciones, vistas, y relaciones. Cada elemento del menú es asignado a un tipo de menú. Una operación OperationQueryMenu es también asignado a un tipo de menú. La definición del menú que es recopilada corresponde a aquellos que son asignados al mismo tipo de menú como la operación.

1.10.16. OperationReturnActionResult

Las operaciones de este tipo devuelven los resultados de las acciones que han sido ejecutadas después de una operación OperationExecuteAction.

1.10.17. OperationSoapRequest

Las operaciones de este tipo invocan a métodos en servicios Web. Para estas operaciones, se debe especificar el puerto que define el punto final del SOAP, al igual que la acción del SOAP identifica cual método Web invocar.

1.10.18. OperationStatusCheck

Este tipo de operación es usada en conjunto con la operación OperationQueryMenu para determinar que elementos del menú deben tener la propiedad de ocultos o habilitados. Una operación del tipo OperationStatusCheck es básicamente una operación del tipo OperationCallComponent que retorna un valor booleano.

1.10.19. OperationTransformationAggregation

Las operaciones de este tipo permiten la transformación de uno o varios esquemas de entrada a un solo esquema de salida. Estas operaciones utilizan transformaciones XSL para su realización.

1.10.20. OperationCustom

Este tipo de operación es usada cuando no son suficientes las operaciones creadas originalmente por el IBF, y se requiere desarrollar

operaciones propias. Estas pueden aceptar cero, o un esquema de entrada y salida.

1.11. Planificación de una solución IBF

Uno de los más comunes requerimientos es cuando el usuario necesita acceder a varias aplicaciones del negocio, o diferentes módulos de una misma aplicación para obtener la información necesaria. IBF permite agrupar todas esas operaciones en una sola operación, la cual puede ser ejecutada en un simple documento de Office.

Una vez identificado la necesidad de una solución IBF, se puede comenzar con su implementación. El concepto básico es empezar con una solución pequeña y hacerla lo suficientemente flexible para su crecimiento.

Es importante recordar que las soluciones IBF no fueron diseñadas para remplazar las aplicaciones de negocio existentes, sino para dar una solución adicional ha este conjunto de herramientas.

La planificación de una solución IBF se divide en las siguientes etapas:

- Planificación de las entidades
- Planificación de los servicios Web
- Planificación de la Metadata.
- Planificación de la interfase de usuario.

- Planificación de seguridad.
- Planificación de liberación.

1.11.1. Planificación de las entidades.

La planificación de las entidades consiste en determinar las entidades del negocio. Por ejemplo, personas, lugares, o cosas que serán utilizadas en la solución IBF.

1.11.2. Planificación de los servicios Web

La planificación de los servicios Web consiste en determinar las operaciones de la lógica del negocio que serán representados como métodos Web. Los servicios Web deben ser diseñados de manera que puedan acceder a los datos del negocio desde cualquier cliente y no solo por la solución IBF. En general un servicio Web debe apuntar a la clase de una operación de la línea del negocio, mientras que una operación de la línea del negocio debe apuntar a un método Web.

1.11.3. Planificación de la Metadata.

La planificación de la Metadata puede dividirse en planificación de los servicios de Metadata y planificación de la solución de Metadata. Los servicios de Metadata consisten en la descripción de cómo acceder a uno o varios servicios Web, en cambio, la solución de Metadata consiste

en la descripción de cómo se usará la Metadata para el despliegue de la información retornada por el servicio de línea del negocio.

1.11.4. Planificación de la Interfase de Usuario

La planificación de la interfase de usuario consiste en describir como el usuario final interactuará con la solución, como será la estructura de despliegue de información y la accesibilidad de la información por parte del usuario.

1.11.5. Planificación de seguridad

La planificación de seguridad consiste en determinar que usuarios necesitan acceder a que secciones de la línea de negocio. La seguridad necesita ser incorporada en la planificación de los servicios Web, Metadata, e Interfase de Usuario.

En general la seguridad de las soluciones IBF es gestionada mediante el Manejador de Autorización. Sin embargo se puede implementar una seguridad adicional en los métodos Web.

1.11.6. Planificación de liberación

La liberación consiste en determinar que Servicio Web de Metadata deber ser liberado, y la implementación de la solución en cada usuario de Office de la organización.

1.12. Instalación del IBF

La instalación del IBF requiere tres tipos de instalaciones diferentes.

- Instalación del Servidor
- Instalación de la herramienta de desarrollo
- Instalación del Cliente

1.12.1. Instalación del Servidor

La instalación se debe realizar en un equipo que este bajo Windows 2003 Server y SQLserver 2000 Service Pack 3, para ahí almacenar el servicio Web de la Metadata.

Nota: Si se desea se puede instalar una instancia de SQLserver 2000 en un equipo diferente, solo se necesita configurarlo de tal modo que el servicio Web pueda acceder a la información.

Una vez designado el equipo servidor, se necesita establecer dos puertos HTTP que serán usados por el IBF para acceder al servicio Web de Metadata.

El servicio Web de Metadata consiste en dos puntos finales, un servicio de lectura y uno de escritura: IBFReadService.asmx y el IBFWriteService.asmx utilizando por defecto los puertos 8081 y 8082 respectivamente.

El asistente de instalación del servidor se encargará de las configuraciones correspondientes.

1.12.2. Instalación de componentes de desarrollo

Después de instalar los componentes de servidor, se procede a instalar los componentes de desarrollo ejecutando el instalador del IBF MetaData Designer.

Este instalador crea una nueva plantilla de proyecto en Visual Studio .Net y así poder diseñar y publicar los servicios Web de Metadata.

1.12.3. Instalación del componente del Cliente

Una vez instalado los componentes de servidor y de desarrollo se puede proceder a instalar los componentes del cliente. Estos se pueden instalar en el equipo de desarrollo para pruebas y liberación.

Con todos los componentes instalados se puede iniciar con el desarrollo de una solución IBF.

2. CAPÍTULO II - Conectando los servicios de la línea de negocio

El objetivo principal de IBF es permitir que los usuarios tengan acceso a la información del negocio dentro del contexto de los documentos de Office.

El IBF puede comunicarse y tener acceso a los servicios de línea de negocio mediante solicitudes SOAP.

Este capítulo se enfocará a dar una guía de cómo construir servicios Web acordes con el IBF

2.1. Análisis

El objetivo de conectar la información es extraer los datos que se pueden relacionar entre sí, así estos estén en diferentes módulos inclusive en muchos casos en diferentes repositorios de datos. Generalmente los módulos de los sistemas informáticos extraen la información que es exclusiva para ese módulo. La línea de negocio va mas allá, extrayendo la mayor cantidad de datos relacionados.

Antes de realizar una solución IBF es necesario saber si realmente al usuario le sería útil una aplicación de esta categoría como apoyo a sus sistemas existentes.

Por ejemplo, si el usuario final no tiene ningún contacto con los sistemas Office, o raramente recibe solicitudes vía correo electrónico, una aplicación IBF no sería necesaria.

Hay que recordar que las soluciones IBF fueron creadas con el fin de que el usuario final no tenga que salir de su entorno de trabajo, en este caso los sistemas Office, para consultar información en diferentes aplicaciones.

La extracción de los datos se realiza mediante peticiones de tipo SOAP es decir, se necesita un servicio Web para poder acceder a los datos de la línea del negocio.

El primer paso para el análisis es identificar los artefactos. Los artefactos son: entidades, vistas, referencias, y relaciones que existen en la aplicación del negocio las cuales pueden ser fácilmente identificadas realizando las siguientes preguntas:

¿Qué información es relevante para que el usuario final consuma desde los sistemas Office?

Ej. Información del estudiante, pagos, préstamos, etc.

¿Qué acciones serán permitidas por parte del usuario?

Ej. Extraer información del estudiante, Mostar pagos por estudiante, etc.

¿Qué tipo de relación existen entre las entidades?

Ej. El número de matrícula del estudiante.

¿Es necesaria la búsqueda de información?

Esto permitirá que se añada la opción de buscar por algún tipo de referencia.

Ej. Búsqueda por número de Matrícula del estudiante.

¿Está la información almacenada en diferentes repositorios de datos?

En algunos casos los sistemas existentes tienen diferentes repositorios de datos, en este caso el servicio Web debería extraer la información necesaria de las diferentes bases de datos.

¿Deben los datos ser normalizados?

Se debe verificar si para completar la información el servicio Web debe acceder a diferentes tablas en un mismo sistema o a diferentes bases de datos para completar la misma.

¿Tendrá el usuario final la posibilidad de actualizar la información o ingresar datos?

En algunos casos es posible que el usuario de una solución IBF necesite ingresar o actualizar datos, sin embargo esto no es recomendable ya que las soluciones IBF fueron creadas para dar un apoyo a los sistemas existentes mas no a remplazarlos en su lógica de negocio.

2.2. Definición de las entidades

Para la creación de una solución inteligente se necesita la definición de las entidades. Estas son simples objetos conceptuales del negocio, los cuales pueden representar personas, lugares o cosas que estén dentro del esquema de la línea de negocio. Estas entidades deben ser objetos

del negocio a un alto nivel, deben ser distintos entre si, y deben ser el punto de referencia hacia otras entidades.

Las entidades pueden ser entendidas involuntariamente por el usuario, y con mayor facilidad por parte del desarrollador. Contestando la primera pregunta del análisis se pude identificar fácilmente las entidades. Si se quiere extraer información del estudiante claramente la entidad resultante sería: “Estudiante”. Otros ejemplos de entidades podrían ser: Préstamos, Pagos, Notas, Materias, etc.

Nota: No necesariamente las entidades se ven reflejadas en un modelo entidad relación. Una entidad podría ser también “Universidad”, la cual no es identificada como un objeto en la base de datos pero si tiene significado en el contexto del negocio.

2.3. Definición de las vistas

Las vistas son diferentes representaciones de cada entidad. Por ejemplo el resumen de Pagos por estudiante puede ser representada como una vista, y una colección de los préstamos del estudiante se obtiene mediante una vista diferente.

Una entidad puede tener múltiples vistas, pero es obligatorio para cada entidad poseer una vista por defecto.

Las vistas como tal son esquemas básicos de XML los cuales tienen etiquetados los campos que serán desplegados hacia el cliente.

Por ejemplo, si se quiere extraer información de la línea del negocio de la entidad Estudiante el usuario lo verá de la siguiente manera:

Matrícula: 100038
Nombre del Estudiante: Diego
Apellido del Estudiante: Vega
E-mail: diego.a.vega@gmail.com
Cédula: 1716726110
Fecha de Nacimiento: 03/03/82
Dirección: Calle 1 Transversal 2
Teléfono: 2417705

Para que la información anterior pueda ser desplegada por el IBF está debe estar en un esquema XML el cual se representa a continuación.

```
<GetEstudiantePorIDResponse xmlns="http://UDLA.SolucionIBF">
    <Estudiante xmlns="urn-InfoEstudiantes">
      <Matrícula >string</Matrícula >
      <Cedula>string</Cedula>
      <Nombre>string</Nombre>
      <Apellido>string</Apellido>
      <Email>string</Email>
      <Fecha_Nacimiento>dateTime</Fecha_Nacimiento>
      <Telefono>string</Telefono>
    </Estudiante>
</GetEstudiantePorIDResponse>
```

Y que a su vez será llenado con información de la siguiente manera:

```
<Estudiante xmlns="urn-InfoEstudiantes">
      <Matrícula >100039</Matrícula >
      <Cedula>1716726110</Cedula>
      <Nombre>Diego</Nombre>
      <Apellido>Vega</Apellido>
      <Email>diego.a.vega@gmail.com</Email>
      <Fecha_Nacimiento>03/03/82</Fecha_Nacimiento>
      <Telefono>2417789</Telefono>
</Estudiante>
```

2.4. Definición de Referencias:

Una vez definidas las vistas se procede a definir las referencias. Las referencias vienen a ser análogas a las claves primarias en una base de datos. Estas serán utilizadas como criterios de búsqueda de la información y así tener acceso a las diferentes entidades. Por ejemplo el número de matrícula del estudiante: “100039”

La misma referencia puede devolver la misma vista, ya que se puede buscar la información del estudiante mediante el número de Matrícula o el correo electrónico.

2.4.1. Definición de operaciones

Las operaciones son las acciones que serán ejecutadas por parte del usuario hacia la línea de negocio. En el concepto de un servicio Web las operaciones son directamente representadas por los métodos de este. Estos métodos son usados para extraer, actualizar o ingresar información de y hacia la base de datos.

Existen tres tipos de operaciones en el contexto de IBF:

Get, Put y Act

2.4.2. Operaciones del tipo Get.

Estas operaciones son las encargadas de recuperar las instancias de las entidades en un formato de vistas. Normalmente se utilizan las referencias para obtener las vistas en las operaciones de tipo Get.

Un ejemplo de una operación de tipo Get sería:

“Extraer información del Estudiante” la cual podría ser representada por el método Web GetInformacionEstudiante

2.4.3. Operaciones del tipo Put.

Las operaciones del tipo Put son aquellas que permitirán al usuario actualizar información desde el IBF. Se puede usar operaciones de tipo Put por ejemplo si se desea actualizar información no crítica como la dirección de correo electrónico (e-mail) de un estudiante.

Las operaciones del tipo Put requieren dos tipos de parámetros.

- La referencia a una instancia de la entidad
- Los datos que serán cambiados en esa entidad.

Desde la perspectiva de la metadata las operaciones del tipo Put son similares a las del tipo Get con diferencias como:

- Se debe definir en la metadata que la acción es del tipo actualizable.
- La operación debe tener un esquema de entrada.

Al igual que las operaciones del tipo Get estas serán representadas como métodos Web.

2.4.4. Operaciones del tipo Act.

Las operaciones del tipo Act son recomendadas cuando es necesario ejecutar algún procedimiento. Como por ejemplo: Rechazar una solicitud o cambiar el estado de alguna entidad. Los requerimientos para la metadata son los mismos que para las operaciones del tipo Put.

Al igual que las operaciones del tipo Put las operaciones de tipo Act en ciertos casos no es recomendable usarlas, todo depende de las necesidades de los usuarios al realizar este tipo de requerimientos.

Estas solo deberían usarse en casos que las decisiones que tenga que tomar el usuario sean críticas y puedan acoplarse a los procedimientos establecidos.

Estas operaciones también son representadas como métodos Web.

2.5. Desarrollo del Servicio Web

"Un servicio Web o WebService es un servicio ofrecido por una aplicación que expone su lógica a clientes de cualquier plataforma mediante una interfaz accesible a través de la red utilizando protocolos estándar de Internet."[1]

A continuación se procederá a la creación de un servicio Web para una operación del tipo Get.

[1] Desarrollo de Aplicaciones .Net con Visual C#, Miguel Rodríguez Gómez-Stern y Marco Antonio Besteiro Gorostizaga, 1era Edición, McGraw-Hill, 2002, Pág. 480.

2.5.1. Construcción de un servicio Web para una operación del tipo Get

Como se explicó anteriormente es necesario definir los artefactos que serán utilizados, para ello se deberá responder las siguientes preguntas:

¿Qué información es relevante para que el usuario final consuma desde los sistemas Office?	• Información personal del estudiante.
¿Qué acciones serán permitidas por parte del usuario?	• Extracción de los datos del estudiante
¿Qué tipo de relación existen entre las unidades?	• Matrícula del Estudiante
¿Es necesaria la búsqueda de información?	• Si.
¿Se encuentra la información almacenada en diferentes repositorios de datos?	• No
¿Deben los datos ser	• No

normalizados?	
¿Tendrá el usuario final la posibilidad de actualizar información o ingresar datos?	• No

El siguiente paso es identificar los artefactos a usar para la creación del servicio Web.

Entidades	Estudiante
Operaciones	**Get:** GetInfoEstudiante
Referencias	Matrícula del Estudiante

En la creación de un servicio Web los artefactos son representados como objetos, métodos y propiedades.

A continuación se procederá a la creación del servicio Web acorde con el IBF.

En este ejemplo, se creará un servicio Web que extraiga la información personal de un estudiante desde un repositorio de datos el cual se encuentra almacenado en SQL Server, el cual contiene información relevante del estudiante como: Nombre, Apellido, correo electrónico, teléfono, etc.

Los pasos a seguir en este caso serán los siguientes:

- Abrir **Microsoft Visual Studio.NET 2003**
- Seleccionar **Nuevo Proyecto | Proyecto Visual C# | Servicio Web ASP.NET**
- Ubicación http://localhost/SolucionIBF y Abrir

2.5.2. Creación de los Objetos de Negocio.

Bajo las condiciones del IBF, los servicios Web deben ser construidos de tal manera que permitan al IBF interactuar con cualquier fuente de datos y que los elementos creados puedan ser reconocidos por el IBF. Para lograr esto es necesario que el servicio Web utilice serialización XML al crear los objetos del negocio y los servicios. En este ejemplo se creará un objeto denominado Estudiante el cual será identificado por el IBF como una "Entidad". Además se creará un objeto denominado EstudianteID que será identificado como el "View Locator" por el IBF y también usado como referencia.

Por último, se realizará la creación de una colección de Estudiantes que retornará uno o varios estudiantes.

2.5.3. Creación del Objeto Estudiante

- **Archivo | Añadir Proyecto | Nuevo Proyecto**
- **Proyecto Visual C# | Class Library**
- Nombre: "**Objetos**"
- Renombrar la clase creada por defecto "Class1.cs" por "Estudiante.cs"

- Añadir al principio de la clase la siguiente referencia using System.Xml.Serialization para habilitar la serialización XML.

```
using System;
using System.Xml.Serialization;


namespace UDLA.InfoEstudiantes.Objetos
{
	/// <summary>
	/// Clase Estudiante
	/// </summary>
	///

	[XmlRoot("EstudianteID",Namespace="urn-InfoEstudiantes")]
	public class EstudianteID
	{
		private string id;

		[XmlAttribute]
		public string ID
		{
			get{return this.id;}
			set{this.id = value;}
		}

	}


	[XmlRoot("Estudiante",Namespace="urn-InfoEstudiantes")]
	public class Estudiante
	{
		private string _Matrícula ;
		private string _Nombre;
		private string _Apellido;
		private string _Email;
		private string _Cedula;
		private DateTime _Fecha_Nacimiento;
		private string _Direccion;
		private string _Telefono;

		[XmlElement]
		public string Matrícula
		{
			get{return _Matrícula ;}
			set{_Matrícula  = value;}
		}


		[XmlElement]
		public string Nombre
		{
			get{return _Nombre;}
			set{_Nombre = value;}
		}

		[XmlElement]
		public string Apellido
		{
```

```
                get{return _Apellido;}
                set{_Apellido = value;}
            }

            [XmlElement]
            public string Email
            {
                get{return _Email;}
                set{_Email = value;}
            }

            [XmlElement]
            public string Cedula
            {
                get{return _Cedula;}
                set{_Cedula = value;}
            }

            [XmlElement]
            public DateTime Fecha_Nacimiento
            {
                get{return _Fecha_Nacimiento;}
                set{_Fecha_Nacimiento = value;}
            }
            [XmlElement]
            public string Direccion
            {
                get{return _Direccion;}
                set{_Direccion = value;}
            }

            [XmlElement]
            public string Telefono
            {
                get{return _Telefono;}
                set{_Telefono = value;}
            }
        }

}
```

Se ha creado el archivo Estudiante.cs, que contiene dos objetos:

- EstudianteID
- Estudiante

El objeto EstudianteID es la representación de una referencia en este caso del ID del Estudiante. Para que el IBF reconozca este objeto como parte de una entidad es necesario agregar la siguiente línea.

[XmlRoot("EstudianteID",Namespace="urn-InfoEstudiantes")]

En el que EstudianteID es el nombre por el cual el IBF identificará esta referencia y urn-InfoEstudiantes es el nombre único que identifica a la solución.

Además para que el IBF reconozca que se trata de una referencia se debe especificar que los atributos de este objeto están en formato XML, esto se logra encabezando el atributo con la siguiente propiedad [XmlAttribute]

De igual forma el debe ser creado el objeto Estudiante cuyo encabezado debe ser:

```
[XmlRoot("Estudiante",Namespace="urn-InfoEstudiantes")]
        public class Estudiante
{
…..
}
```

Incluyendo en este cada uno de los elementos que contiene el objeto. Debe notarse que cada elemento de esta clase ha sido denotado con la propiedad [XmlElement]. Esto se realiza con la finalidad de que se aplique la serialización de los datos y sean retornados como elementos nativos XML y así poder ser identificados por el IBF.

Por tratarse de una misma entidad y tener un orden acorde con los conceptos IBF se ha creado los dos objetos EstudianteID y Estudiante en una misma clase. No obstante por prácticas de programación se pueden representar los objetos en dos clases independientes.

2.5.4. Creación de la colección Estudiantes

La creación de la colección Estudiante es necesaria bajo los parámetros del IBF para que los elementos de la entidad sean almacenados en estas colecciones y así tener una mejor integridad de los tipos de datos y una mejora en el rendimiento.

- Añadir | Nueva Carpeta
- Añadir | Añadir Clase
- Nombre del la clase: ColeccionEstudiantes.cs

Para habilitar el uso de colecciones se debe agregar la siguiente referencia:

```
using System.Collections;

using System;
using System.Collections;

namespace UDLA.InfoEstudiantes.Objetos
{
        /// <summary>
        /// Clase ColeccionEstudiantes.
        /// </summary>
        public class ColeccionEstudiante : System.Collections.CollectionBase
        {
                public Estudiante this[int indice]
                {
                        get{return (Estudiante) List[indice];}
                        set{List[indice] = value;}
                }

                public int Add(Estudiante item)
                {
                        return List.Add(item);
                }
```

```
        public int IndexOf(Estudiante item)
        {
            return List.IndexOf(item);
        }

        public void Insert(int indice, Estudiante item)
        {
            List.Insert(indice,item);
        }

        public void Remove(Estudiante item)
        {
            List.Remove(item);
        }

        public bool Contains(Estudiante item)
        {
            return List.Contains(item);
        }

        public void CopyTo(Estudiante[] destino, int indice)
        {
            List.CopyTo(destino,indice);
        }
    }
}
```

Esta clase será utilizada cuando se requiera extraer varios elementos de una misma entidad.

Como por ejemplo, si se requiere extraer la información de varios estudiantes bajo un mismo parámetro de búsqueda.

2.5.5. Creación de la capa de datos.

- **Archivo | Añadir Proyecto | Nuevo Proyecto**
- **Proyecto Visual C# | Class Library**
- Nombre: "**AccesoDatos**"
- Renombrar la clase creada por defecto "Class1.cs" por "Datos.cs"
- Añadir al principio de la clase using UDLA.InfoEstudiantes.Objetos para poder tener acceso a la clase Objetos creada anteriormente.

```
using System;
using UDLA.InfoEstudiantes.Objetos;
```

```
using System.Data;
using System.Data.SqlClient;


namespace UDLA.InfoEstudiantes
{
        /// <summary>
        ///
        /// </summary>
        ///

        public class AccesoDatos
        {

                public                                          UDLA.InfoEstudiantes.Objetos.Estudiante
GetEstudiantePorID(UDLA.InfoEstudiantes.Objetos.EstudianteID estudianteId)
                {

                UDLA.InfoEstudiantes.Objetos.Estudiante  EstuItem = null;
                System.Data.SqlClient.SqlConnection             conn            =           new
System.Data.SqlClient.SqlConnection("server=DNET\\Local;database=UDLA;User id=sa;");
                        string sSQL;
sSQL = "SELECT * FROM ESTUDIANTE WHERE MATRÍCULA  = " +          estudianteId.ID + "";
                        System.Data.SqlClient.SqlCommand cmd = new
System.Data.SqlClient.SqlCommand(sSQL,conn);
                        conn.Open();
                        System.Data.SqlClient.SqlDataAdapter sqlAdapter = new
System.Data.SqlClient.SqlDataAdapter(cmd);
                        DataTable dtEstudiante = new DataTable();
                        sqlAdapter.Fill(dtEstudiante);

        if (dtEstudiante.Rows.Count >= 0)
        {

                foreach (DataRow Fila in dtEstudiante.Rows)
                {

                EstuItem = new Estudiante();
                EstuItem.Matrícula  = (string)Fila[0].ToString();
                EstuItem.Nombre =(string)Fila[1].ToString();
                EstuItem.Apellido = (string)Fila[2].ToString();
                EstuItem.Cedula =(string)Fila[3].ToString();
                EstuItem.Email = (string)Fila[4].ToString();
                EstuItem.Fecha_Nacimiento = (DateTime)Fila[5];
                EstuItem.Direccion = (string)Fila[6].ToString();
                EstuItem.Telefono  = (string)Fila[8].ToString();
                                                                                        }
                }

                        conn.Close();
                        conn.Dispose();

                        return EstuItem;

                }
```

La función GetEstudiantePorID tiene como propósito consultar a la base de datos y extraer la información del estudiante mediante un parámetro de entrada en este caso, el objeto EstudianteID el cual retorna el objeto

Estudiante. Si se requiere realizar una consulta que extraiga varios estudiantes el valor de retorno seria una colección de Estudiantes.

2.6. Creación del Servicio Web

El paso final, es la creación del servicio Web como tal.

- Renombrar archivo por defecto “Service1.asmx” por “ServicioWeb.asmx”

El servicio Web será el que exponga la información del estudiante, este consumirá las funciones creadas en la capa de datos.

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Web;
using System.Web.Services;
using System.Xml;
using System.Xml.Serialization;


namespace SolucionIBF
{

        [WebService(Namespace = "http://UDLA.SolucionIBF")]
        public class ServicioWeb : System.Web.Services.WebService
        {
                public ServicioWeb()
                {
                        InitializeComponent();
                }


                [WebMethod]
                public                                  UDLA.InfoEstudiantes.Objetos.Estudiante
GetEstudiantePorID(UDLA.InfoEstudiantes.Objetos.EstudianteID IDEstudiante)
                {
                        UDLA.InfoEstudiantes.AccesoDatos        acceso        =        new
UDLA.InfoEstudiantes.AccesoDatos();
                        return acceso.GetEstudiantePorID(IDEstudiante);
                }
```

Se ha creado un Servicio Web que representa una operación del tipo Get acorde con el IBF. Ahora se puede compilar el proyecto y ejecutarlo.

El resultado de la ejecución de este servicio Web es solamente la estructura de entrada y salida de este. Ya que para tener los resultados de los datos extraídos seria necesario pasar como parámetro el EstudianteID en un formato de XML serializado.

```
POST /SolucionIBF/ServicioWeb.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://UDLA.SolucionIBF/GetEstudiantePorID"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetEstudiantePorID xmlns="http://UDLA.SolucionIBF">
      <EstudianteID ID="string" xmlns="urn-InfoEstudiantes" />
    </GetEstudiantePorID>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetEstudiantePorIDResponse xmlns="http://UDLA.SolucionIBF">
      <Estudiante xmlns="urn-InfoEstudiantes">
        <Matrícula >string</Matrícula >
        <Nombre>string</Nombre>
        <Apellido>string</Apellido>
        <Email>string</Email>
        <Cedula>string</Cedula>
        <Fecha_Nacimiento>dateTime</Fecha_Nacimiento>
        <Direccion>string</Direccion>
        <Telefono>string</Telefono>
      </Estudiante>
    </GetEstudiantePorIDResponse>
  </soap:Body>
</soap:Envelope>
```

El código anterior demuestra cual es la entrada y salida de este servicio Web en particular.

El código fuente completo de acceso a datos, y servicios Web se encuentra en el anexo #2.

3. CAPÍTULO III - MetaData

3.1. Que es la MetaData?

Básicamente la metadata es un archivo XML el cual contiene toda la información sobre el funcionamiento de la solución IBF.

La metadata define todo lo que se refiere a operaciones, entidades, acciones y los elementos visuales (Interfaz de usuario), todos estos elementos se agrupan en dos conceptos de desarrollo denominados: creación del servicio de la metadata y solución de la metadata.

3.2. Desarrollo de la MetaData.

Para el desarrollo de la metadata Microsoft creó un componente adicional para el Visual Studio 2003, denominado herramienta de diseño de metadata.

Los componentes claves del diseñador de metadata son: el explorador de metadata y la guía de metadata.

Este componente es de gran ayuda ya que permite navegar por los diferentes elementos XML e interactuar de una forma amigable con dichos elementos.

Además el diseñador de metadata cuenta con asistentes para la creación de entidades, vistas, acciones, etc. De esta forma agilitar el proceso de desarrollo.

En este capítulo se creará un proyecto de metadata y se usará las funcionalidades que presta el diseñador de la metadata.

A la solución existente se añade un nuevo proyecto del tipo Information Bridge Metedata Proyects

Se selección la plantilla MetaData como lo muestra la figura 3.1

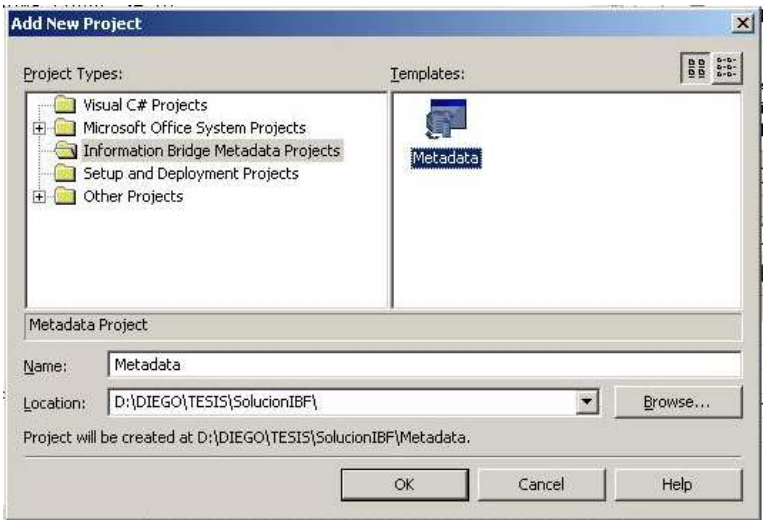

Figura 3.1 Creación del nuevo proyecto de metadata

Como resultado de la operación anterior, se creará un nuevo proyecto denominado Metadata, el cual contiene un archivo XML, dicho archivo contendrá toda la información necesaria para el funcionamiento de la solución.

Figura 3.2 Proyecto de metadata creado en el explorador de solución

Adicionalmente se desplegará una pestaña similar a la conocida en Visual Studio como pestaña de herramientas, la cual contiene los asistentes para la creación de la metadata, se denomina la guía de metadata.

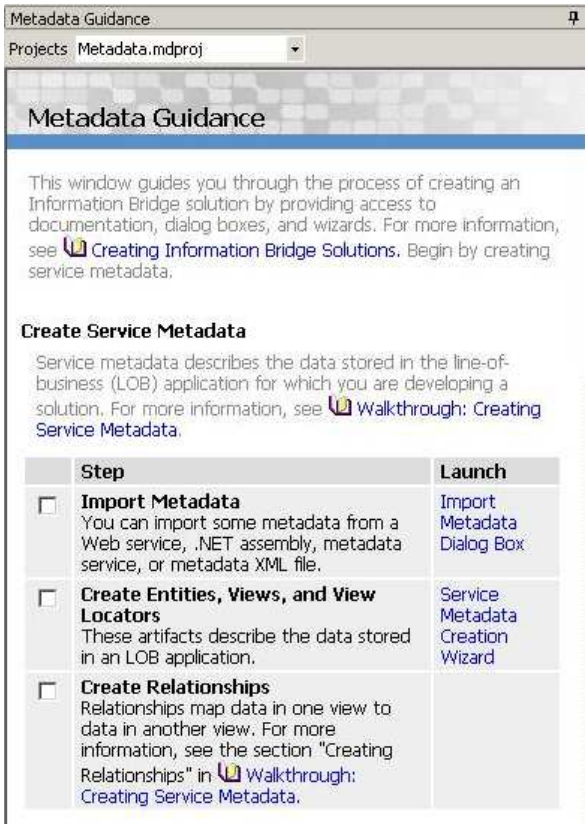

Figura 3.3 Pantalla correspondiente a la guía de metadata

La guía de metadata se divide en dos importantes secciones:

Servicio de la metadata

Solución de la metadata.

3.2.1. Creación del servicio de la metadata

Esta sección es un conjunto de asistentes que sirven para describir todos los elementos que permiten la conexión con la línea del negocio, incluyendo los servicios Web, los artefactos, relaciones etc.

3.2.2. Importando la Metadata

El primer paso para la creación del servicio de metadata es la importación de la metadata, esto se realiza desde el servicio Web creado anteriormente.

Pulsando en la opción Import Metadata Dialog Box de la guía de metada, se despliega el siguiente cuadro.

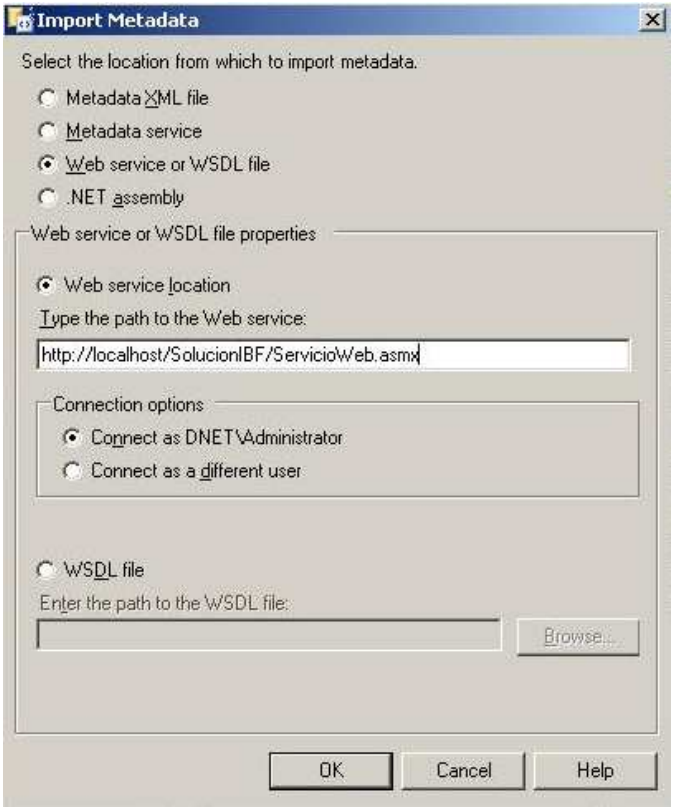

Figura 3.4 Pantalla para la importación de la metadata desde un servicio Web

En este cuadro se selecciona desde donde se quiere importar la metadata, en este caso se selecciona “Web Service or WSDL file”.(Servicio Web o archivo WSDL)

Y se escribe el URL en el cual se publico el servicio Web. En este caso es:

http://localhost/SolucionIBF/ServicioWeb.asmx

Realizada esta acción la siguiente pantalla desplegará, todos los métodos publicados en el servicio Web, sus diferentes esquemas tanto de entrada como de salida. Como se explicó anteriormente, las

operaciones en el concepto del IBF son representados como métodos en los servicios Web.

Es por eso que en el cuadro de operaciones se ve reflejado el método Web “GetEstudiantePorID”. Al cual le corresponden el esquema de entrada “EstudianteID” y como esquema de salida “Estudiante”.

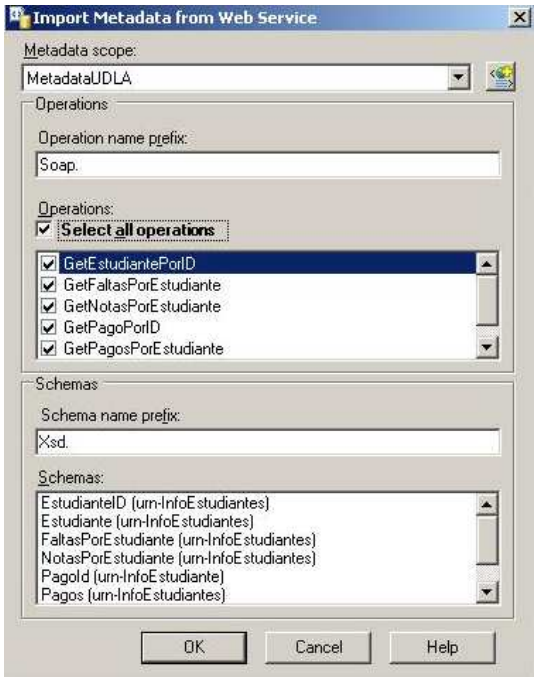

Figura 3.5 Pantalla de la selección de operaciones

Figura 3.6 Pantalla para adjuntar o remplazar la metadata importada

Se adjunta la metadata existente, seleccionando la opción “Merge”

Se cambia el nombre de la metadata creada pulsando el botón derecho en el nombre creado por defecto (Enter your scope name in here), el cual desplegará la pantalla siguiente:

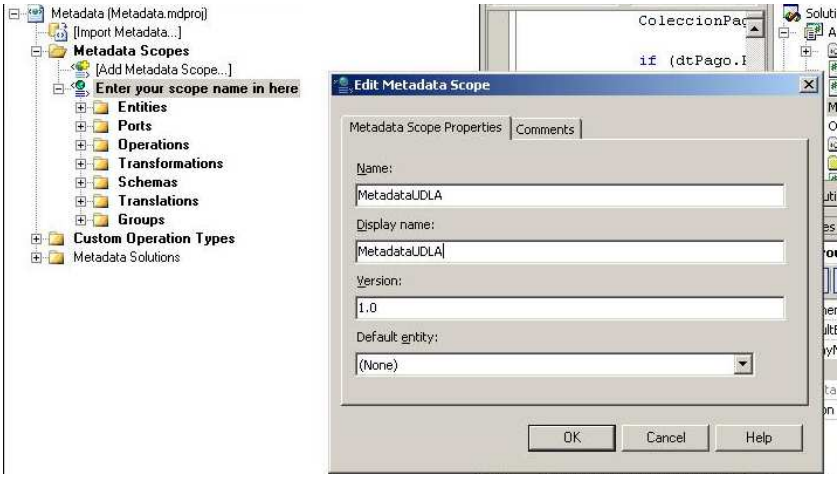

Figura 3.7 Pantalla para cambiar el nombre de la metadata

Se procede a cambiar el nombre, para este caso en particular se nombró: MetaData.UDLA

Nota: Se recomienda usar nombres separados por puntos para obtener un identificador único.

Ej. Proyecto.Institucion.Departamento

Hasta este punto se ha generado un archivo XML con las operaciones y esquemas que serán utilizados por la solución IBF.

3.2.3. Creación de Entidades, Vistas, y localizadores

La parte más importante en el desarrollo del servicio de MetaData es la creación de las Entidades, Vistas, y localizadores (Identificadores Ej. EstudianteID)

En la guía de MetaData se encuentra la opción “Service MetaData Creation Wizard” (Asistente de Creación de Servicio de Metadata)

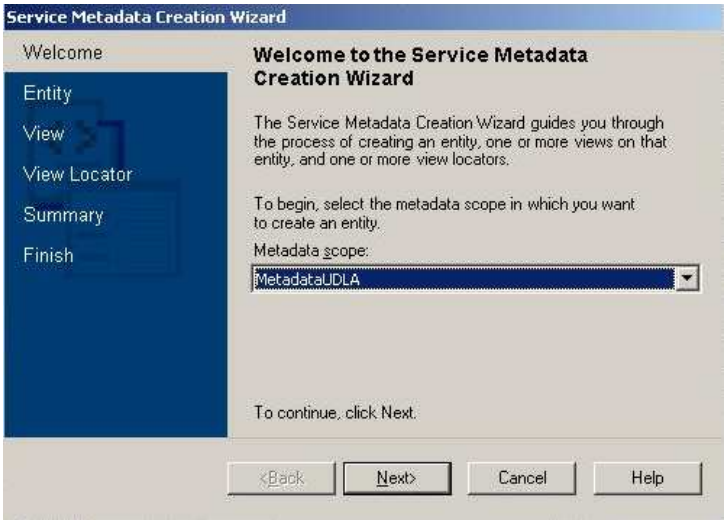

Figura 3.8 Asistente de creación del servicio de metadata

Esta opción desplegará la pantalla de inicio del asistente el cual, guiará para la creación de las entidades, vistas e identificadores.

El primer paso es escoger el alcance de la MetaData, recuerde que el alcance de la metadata es el nombre en el cual irán agrupados todos los

artefactos, operaciones y los diferentes elementos en conforman la MetaData.

En este caso es MetaData.UDLA.

El segundo paso es crear la entidad, y crear un esquema de referencia.

Hay que recordar que el esquema de referencia corresponde a la salida expuesta por el Servicio Web.

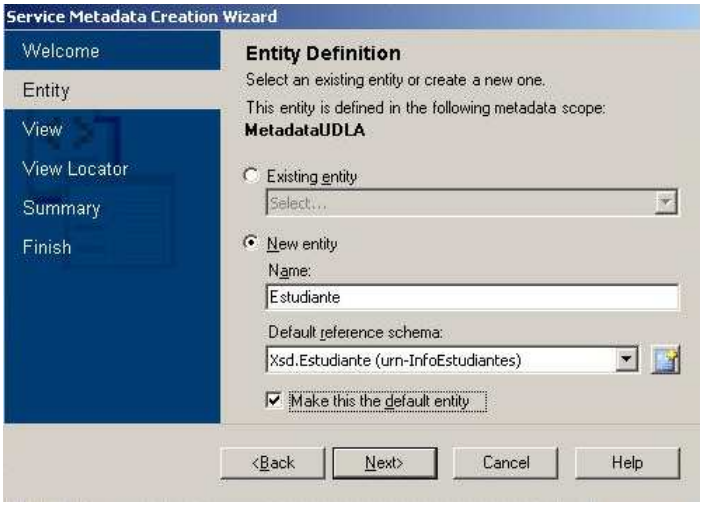

Figura 3.9 Creación de una entidad

A continuación se procede a crear la Vista de la entidad, es importante saber que se puede crear varias vistas por una entidad.

En este caso el nombre de la vista es VistaEstudiante y se debe asignar un esquema para cada vista. En este caso el esquema es Xsd.Estudiante.

Se puede observar en la figura 3.10 que existe la opción para definir si la vista contiene varios resultados.

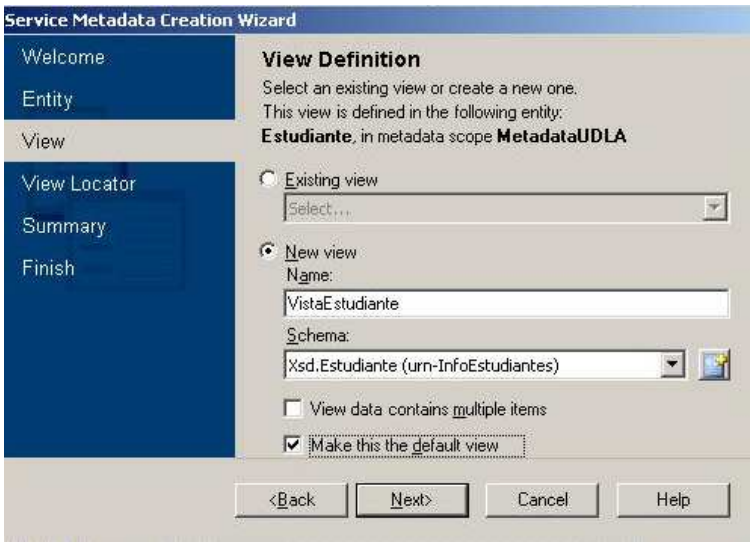

Figura 3.10 Pantalla para la definición de vistas

Después de concluir con la definición de las vistas, es necesario establecer el identificador o conocido como el localizador de la vista, este elemento es la referencia que permite desplegar la vista, y se realiza mediante las operaciones (métodos) expuestos por el servicio Web. En este caso la operación que es utilizada y es detectada automáticamente por el asistente es Soap.GetEstudiantePorID donde su identificador o parámetro de entrada es el esquema Xsd.EstudianteID.

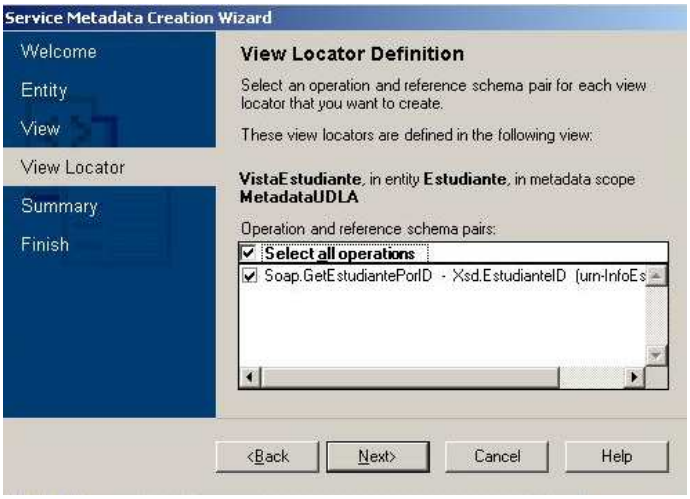

Figura 3.11 Selección del identificador o localizador de la vista

Finalmente el asistente despliega un resumen de las selecciones que se han realizado en el.

Se puede observar en la figura 3.12 los elementos creados por el asistente.

- Entidad: Estudiante
- Vista: VistaEstudiante
- Identificado: EstudianteID

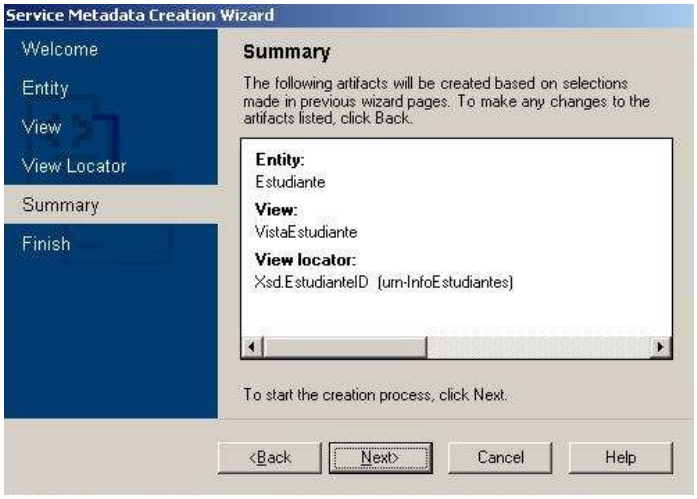

Figura 3.12 Resumen de los elementos creados por el asistente

La última pantalla del asistente, muestra los elementos en un formato de árbol en un esquema de jerarquía de metadata.

Figura 3.13 Elementos creados por el asistente en formato de jerarquía de metadata

En la figura 3.14 se puede observar los elementos creados por el asistente, y que bajo de la entidad Estudiante se encuentran las vistas,

acciones, relaciones e identificadores. Algunos de estos elementos se verán más adelante.

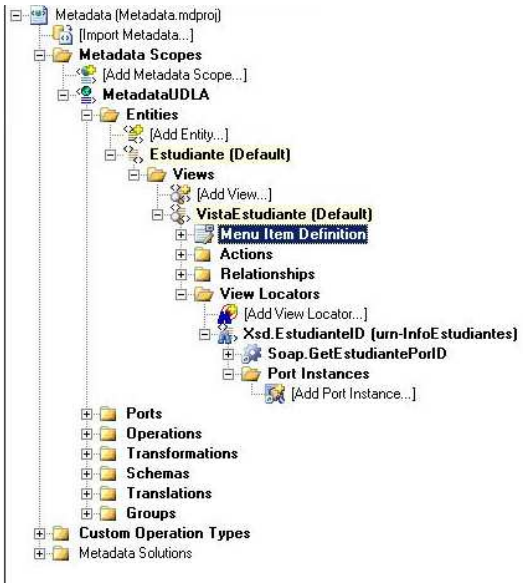

Figura 3.14 Representaciones en el explorador de metadata

3.3. Creación de la solución de MetaData

Al igual que para la creación de los servicios de metada, existen varios asistentes que ayudan a la creación de la solución de metada. Estos son:

- Asistente para la creación de regiones.
- Asistente para la creación de regiones de referencia.
- Asistente para la creación de menús.
- Asistente para la definición de búsqueda.

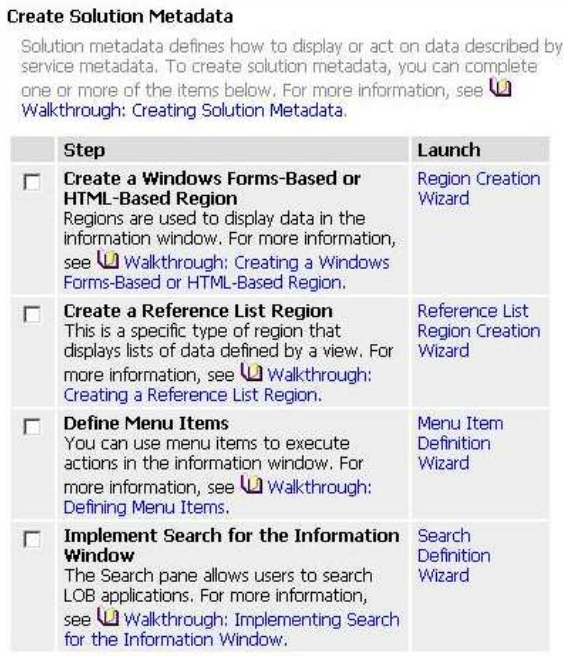

Figura 3.15 Guia de metadata, asistentes para la creación de la solución de metadata

3.3.1. Creación de Regiones

El IBF ofrece muchas opciones para el despliegue de información hacia los usuarios. Se puede construir interfaces que sean contenidas por el panel del IBF.

El panel del IBF esta diseñado para soportar varios elementos de interfaz conocidos como regiones, y cada región es independiente en el panel del IBF.

Este asistente ayuda a crear las regiones (Formularios Windows) o regiones de tipo HTML, para que sean mostradas por el IBF.

Adicionalmente también creará las acciones necesarias para el despliegue de dicha región. Una vez creada la región se puede modificar

tanto el código como la apariencia del control creado para acoplarlo a necesidades específicas.

Figura 3.16 Inicio del asistente para la creación de regiones

El segundo paso sería crear un nombre para la solución de metadata, este proceso se encargará de agrupar todos los artefactos creados anteriormente. En este caso no se necesita crear un nombre, ya que anteriormente se creó un alcance de metadata.

Crear un nombre es recomendable solo cuando se desarrollan soluciones grandes, ya que agrupa los elementos y es más fácil identificarlos cuando se quiere realizar modificaciones.

Para que la región pueda leer los datos, es necesario que los datos sean leídos primero en forma de un esquema. Es por eso que se debe escoger los esquemas creados por las vistas de cierta entidad.

Hay que recordar que una vista no contiene los datos, es simplemente un esquema a llenar con datos.

Para esta pantalla se realizan los siguientes pasos: (Figura 3.17)

- Seleccionar el alcance de metadata – seleccionar MetaData.UDLA
- Seleccionar la fuente del Esquema – seleccionar Vista.
- Entidad – Estudiante
- Vista – Vista Estudiante.
- Siguiente

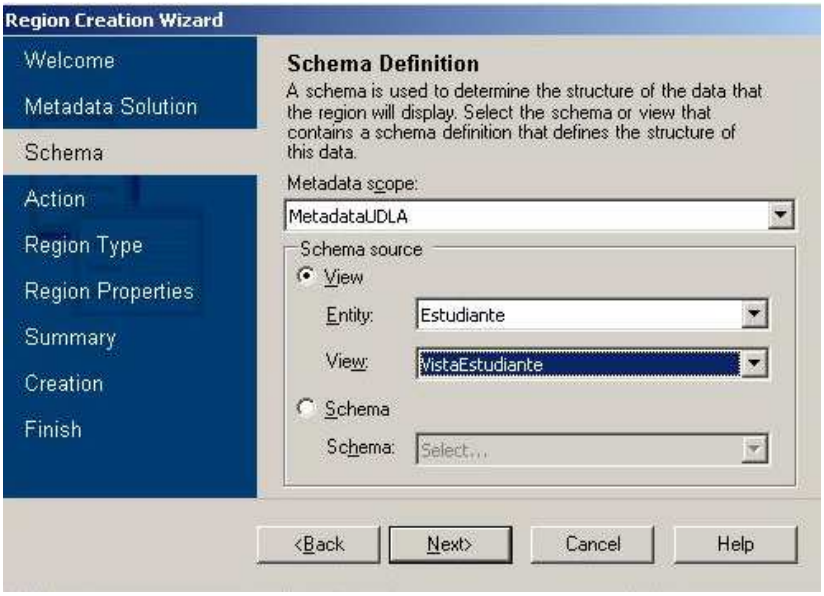

Figura 3.17 Definición de esquemas

Para que una región sea desplegada necesita de una acción. La acción se encadena con la operación respectiva para realizar la petición correspondiente.

Estas operaciones incluyen menús, transformación de datos y la llamada a que la región sea desplegada.

En el siguiente paso se creará una acción que despliegue la región y propiedades al panel.

- Activar la opción – crear acción para desplegar la región (Figura 3.18)
- Darle un nombre a la acción – MostrarEstudiante
- Establecer un nombre para el panel de referencia.
- Darle una descripción al panel de referencia.

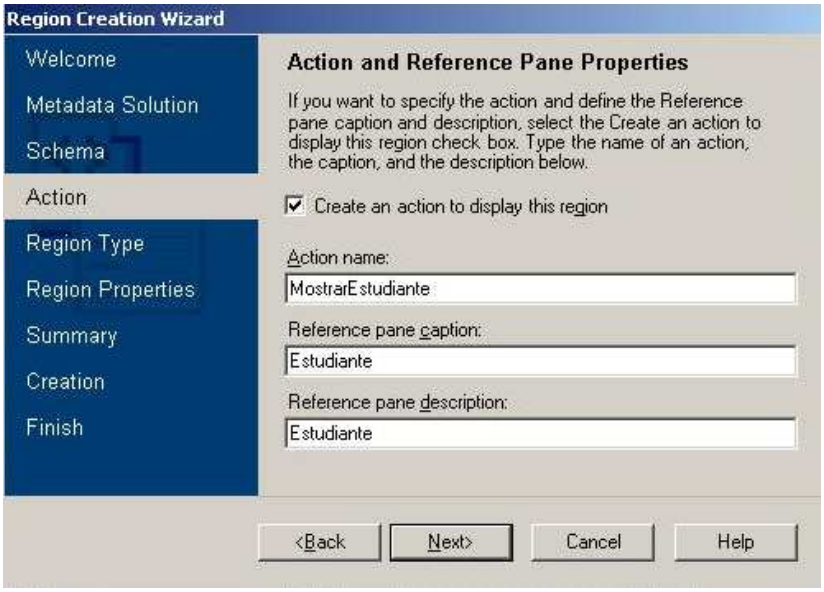

Figura 3.18 Creación de acciones

A este punto, el asistente brinda la opción de escoger el tipo de región que se desea desplegar. Se puede escoger entre una región basada en HTML o

una región del tipo formulario de Windows, además se puede escoger el lenguaje de programación.

- Seleccionar el tipo de región – Formulario de Windows (Figura 3.19)
- Lenguaje – Csharp
- Proyecto Visual Studio – Nuevo proyecto – Interfaz.

El asistente crea un nuevo proyecto del tipo formulario Windows con el nombre Interfaz.

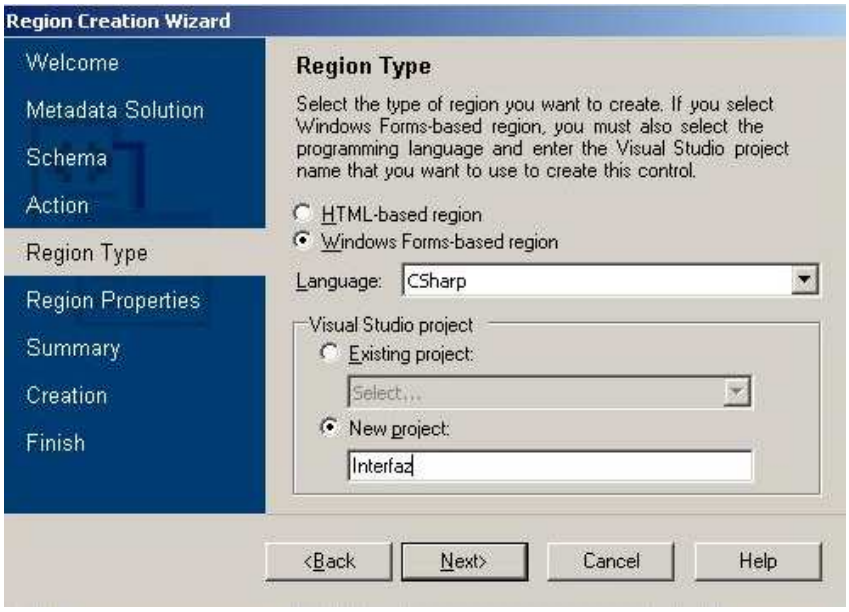

Figura 3.19 Tipos de región

El siguiente paso es establecer las propiedades de la región, en esta pantalla se especifica el nombre de la clase que será creada, además el

nombre que será desplegado en la región, descripción y modo de despliegue (colapsada, abierta y que no pueda ser colapsada)

- Establecer el nombre de la clase – ClassEstudiante. (Figura 3.20)
- Nombre a desplegar – Estudiante
- Descripción – Estudiante
- Modo de despliegue – Abierta (ExpandedRegion)

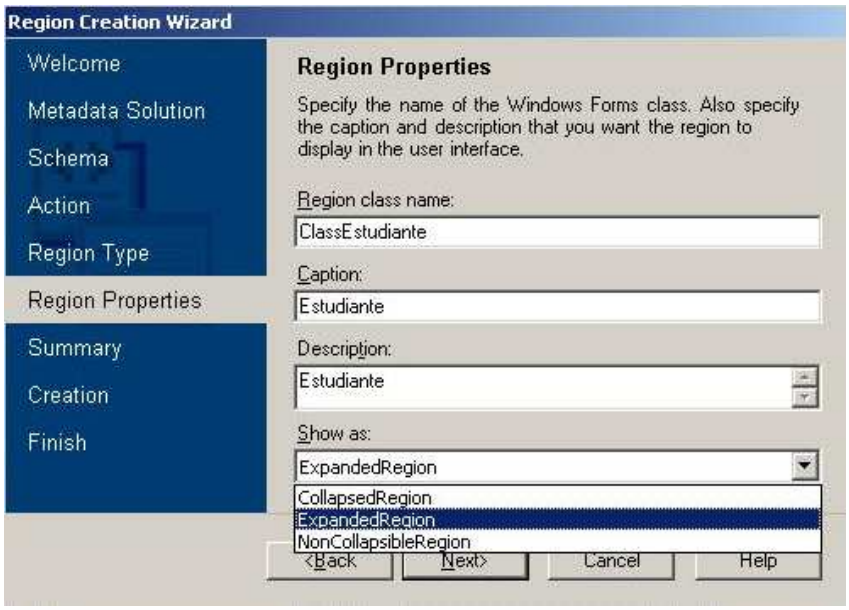

Figura 3.20 Propiedades de la región

En la Figura 3.21 se presenta el resumen de la creación de la región por parte del asistente. En el botón Avanzado (Advanced) se puede acceder a todas las propiedades creadas por el asistente. Si se desea se puede cambiar estos nombres. Ver Figura 3.22

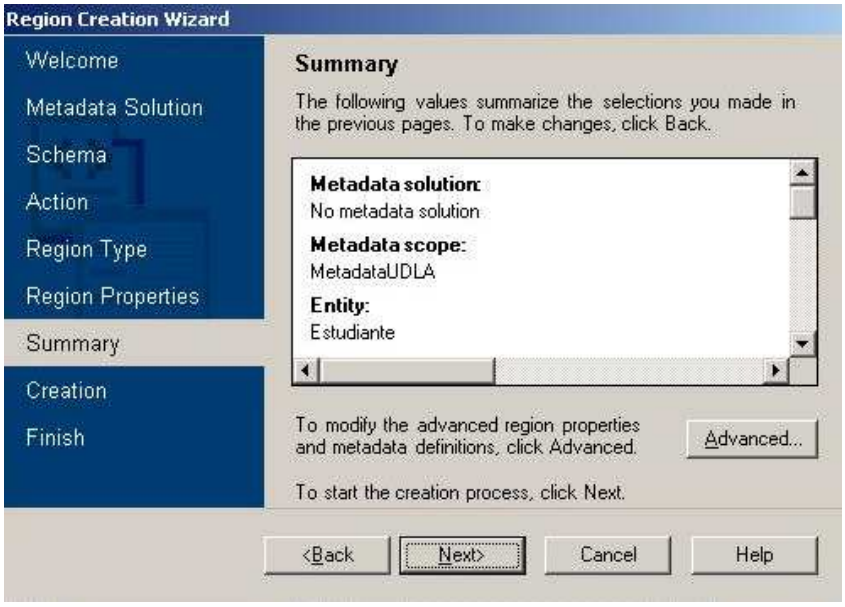

Figura 3.21 Resumen del asistente

Luego de lo cual se muestra una pantalla en la cual se puede cambiar los nombres de las diferentes propiedades creadas por el asistente.

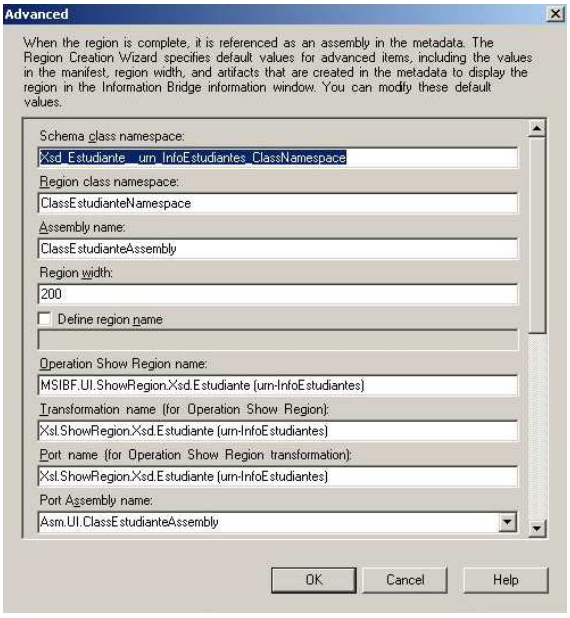

Figura 3.22 Configuración avanzada

Finalizado el asistente, este creará un proyecto adicional a la solución.

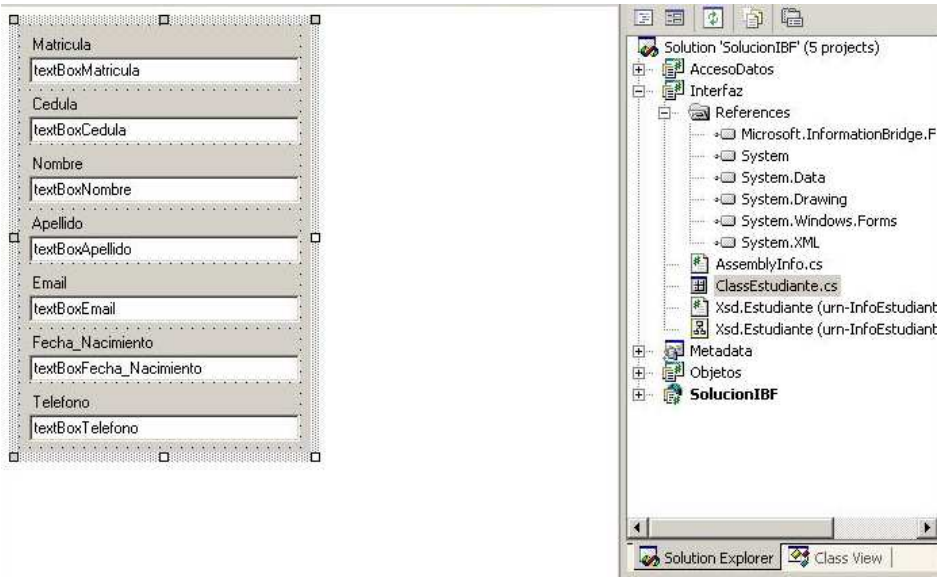

Figura 3.23 Control de usuario creado por el asistente

Como se puede ver en la figura 3.23, el asistente ha creado un control de usuario, con los elementos necesarios para el despliegue de los datos, esto es gracias a la importación de los esquemas, ya que el asistente crea un cuadro de texto por cada elemento del esquema importado.

Como se trata de cualquier control de usuario se puede modificar su parte visual y lógica de acuerdo a las necesidades de la solución.

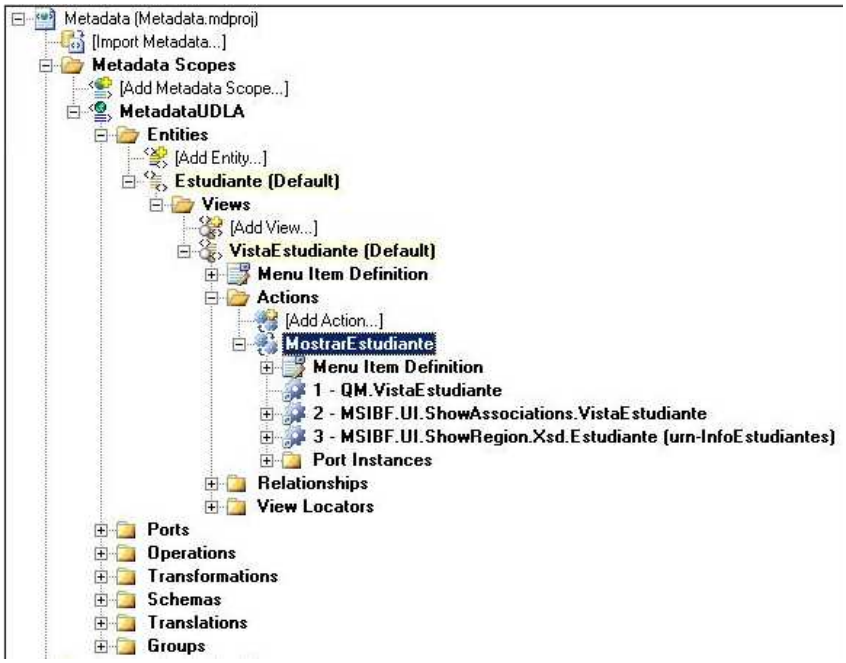

Figura 3.24 Acciones creadas en la metadata

Se puede observar en la Figura 3.24 que bajo la entidad Estudiante – Vistas – VistaEstudiante- se han creado las acciones correspondientes a la región.

3.4. Creación de regiones del tipo lista de Referencia.

Las regiones del tipo lista de referencia fueron creadas exclusivamente para el IBF, estas regiones tienen la funcionalidad de desplegar una lista de datos que están contenidos en cierta vista.

Cada uno de estos ítems puede ser expandido mediante la función estándar “Show Details” o mostrar detalles, para acceder a la información completa de dicho ítem.

Para poder acceder a esta funcionalidad lo único que se requiere es que la solución IBF contenga un método Web que extraiga varios ítems. Por ejemplo. Pagos del Estudiante.

En la guía de metadata existe el asistente denominado: Reference List Creation Wizard.

Pulsando en esta opción se desplegará la pantalla siguiente: Figura 3.25

Al igual que los asistentes anteriores para este caso no se utilizará una solución de metadata prederterminada.

Pulse en Next

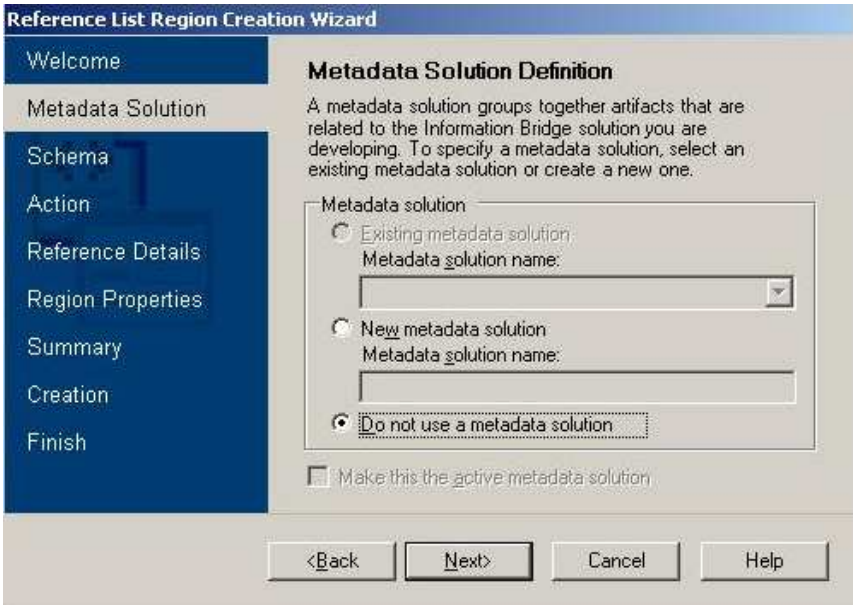

Figura 3.25 Inicio del asistente para la creación de lista de referencia

El siguiente paso es definir el esquema fuente para la región.

Alcance de Metadata – Metadata.UDLA

Entidad – Pagos

Vista – VistaPagosPorEstudiante.

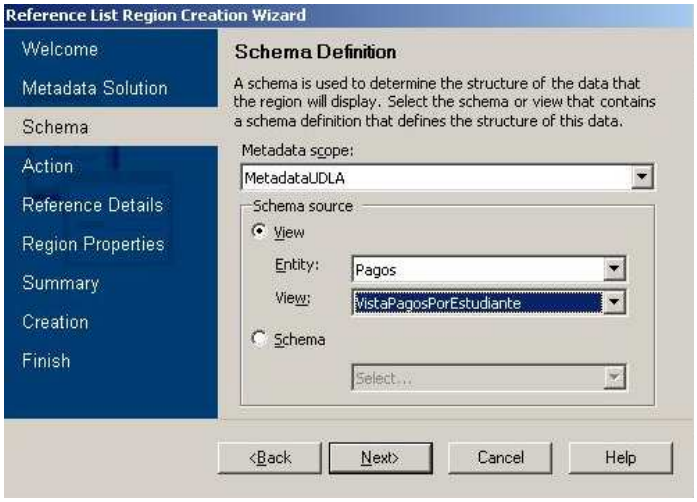

Figura 3.26 Definición de esquema

Como se mencionó anteriormente cada región debe tener una acción asociada para que esta pueda ser mostrada en el IBF. El siguiente paso es definir esta acción y darle al IBF las propiedades de texto a mostrar. Figura 3.27

Nombre de la acción – MostrarListaPagos

Nombre a desplegar – Pagos

Descripción de referencia - Pagos

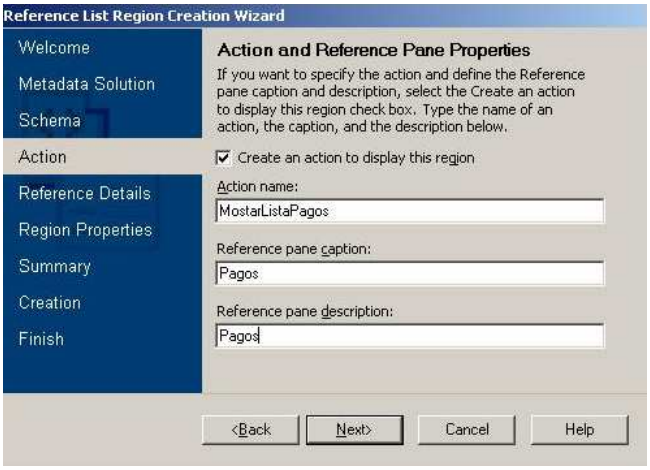

Figura 3.27 Propiedades de la lista de referencia

Es necesario establecer las propiedades de la región como tal. (Figura 3.28)

Nombre a desplegar – Lista de Pagos

Descripción - Lista de Pagos

Forma de despliegue – Región expandida (ExpandedRegion)

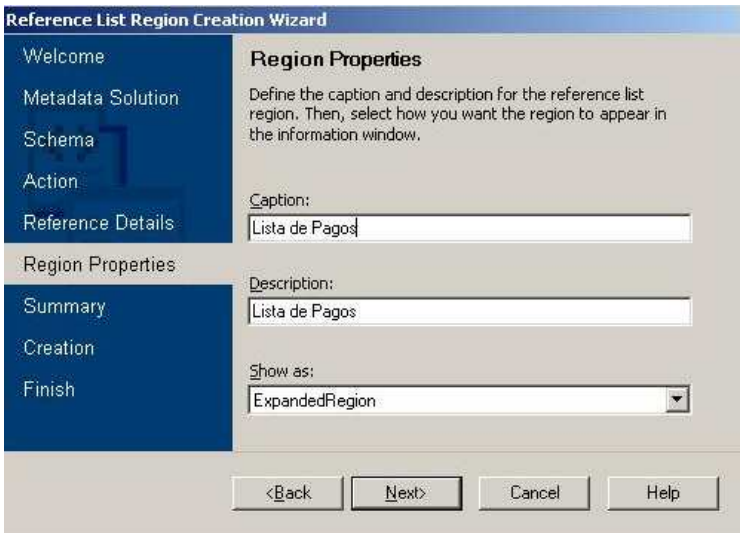

Figura 3.28 Propiedades de la región

Al igual que el asistente para crear regiones este muestra un resumen de las diferentes propiedades creadas por el asistente.

Al final del asistente se activará el programa BizTalk el cual permite un mapeo entre los elementos XML, arrastrando los elementos que se muestran en la pantalla de la izquierda se puede conectar con los elementos de la pantalla derecha. De esta forma haciendo una transformación directa entre los esquemas de entrada y de salida.

La región de lista de referencia tiene 3 elementos importantes: el texto a mostrar, la descripción, y el identificador del elemento a mostrar.

Para este ejemplo se concateno los siguientes elementos: (Figura 3.29)

IDPago – ID

Razón – Texto a desplegar (Caption)

Fecha_Pago – Descripción (Description)

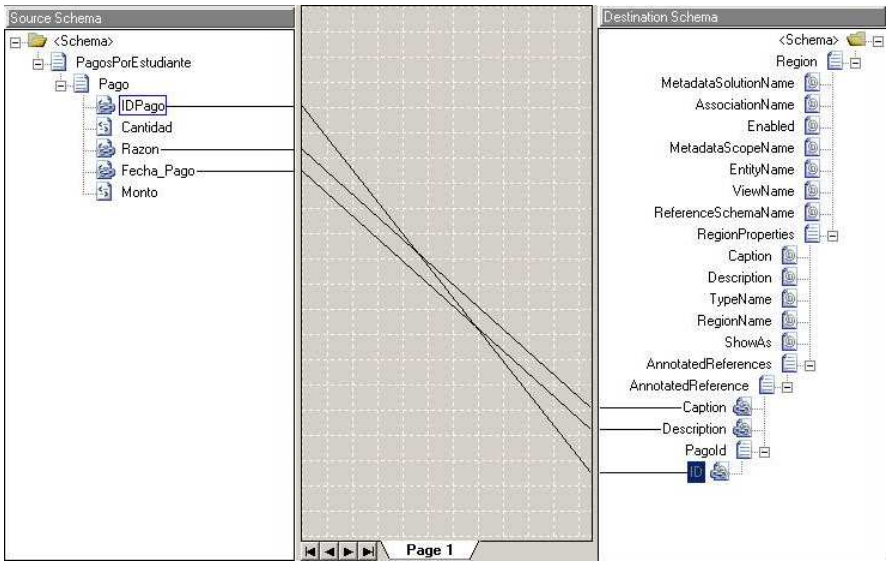

Figura 3.29 BizTalk en IBF

Como resultado se obtiene una región creada por el IBF en donde se obtiene los elementos agrupados en forma de lista y con acciones independientes.

En la Figura 3.30 se puede observar como están agrupados los resultados y con la opción “Show Details”, la cual mostrará la región correspondiente a la información del ítem solicitado.

Figura 3.30 Región lista de referencia

3.5. Definición de menús.

Una de las ventajas del IBF es la posibilidad de crear menús dentro de la solución, de esta manera poder interactuar con los datos expuestos en el IBF.

Por ejemplo si se esta exponiendo la información del estudiante, se puede crear un menú que ejecute la operación para que despliegue los pagos realizados. De esta manera obtener una navegación entre entidades y a su vez entre vistas.

Para poder crear menús se necesitan ciertos pasos previos antes de poder ejecutar el asistente.

- Crear una transformación de esquemas
- Crear una relación entre las dos vistas.

3.5.1. Crear una transformación de esquemas.

Este tipo de transformación es esencial para que los datos expuestos en el IBF puedan ser pasados como parámetros.

Por ejemplo:

Se tiene expuesto el esquema siguiente:

```
<Estudiante xmlns="urn-InfoEstudiantes">
    <Matrícula >100039</Matrícula >
    <Cedula>1716726110</Cedula>
    <Nombre>Diego</Nombre>
    <Apellido>Vega</Apellido>
    <Email>diego.a.vega@gmail.com</Email>
    <Fecha_Nacimiento>03/03/82</Fecha_Nacimiento>
    <Telefono>2417789</Telefono>
</Estudiante>
```

El cual corresponde a la vista “VistaEstudiante”. Para poder visualizar la lista de Pagos del estudiante, es necesario el parámetro de entrada Matrícula.

La transformación se encarga de trasformar el esquema inicial en:

<EstudianteID ID="100039" xmlns:"urn-InfoEstudiantes" />

El cual será tomado como parámetro de entrada para la acción “MostrarPagosDelEstudiante”

Para crear una transformación se debe desplegar el nivel “Transformations” que se encuentra en el explorador de la metadata.

Pulsando en [Add Transformation]

Se desplegará la pantalla correspondiente a la Figura 3.31

Se escoge un nombre para la transformación:

TransformaciónEstudiantesPagos

Esquema de entrada: Xsd.Estudiante

Esquema de salida: Xsd.EstudianteID

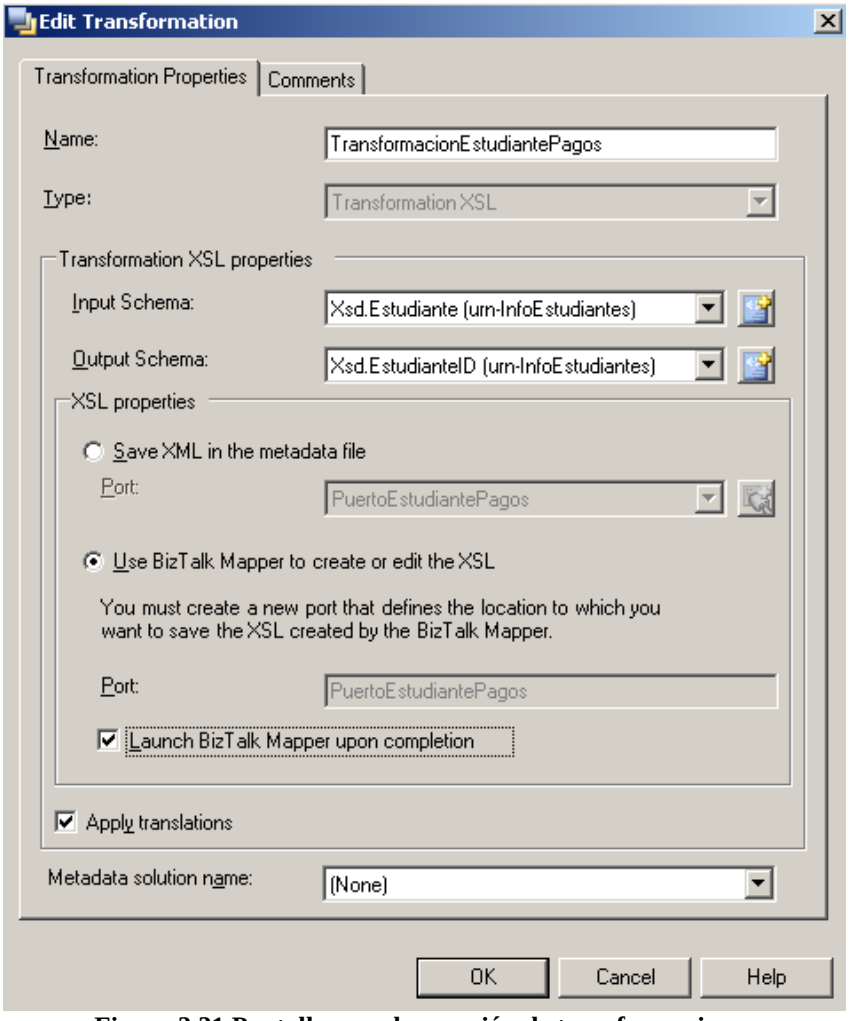

Figura 3.31 Pantalla para la creación de transformaciones

Se puede activar la opción de ejecutar el programa BizTalk para el mapeo de los elementos XML.

En este caso se desea relacionar el elemento Matrícula con el elemento EstudianteID (Figura 3.32).

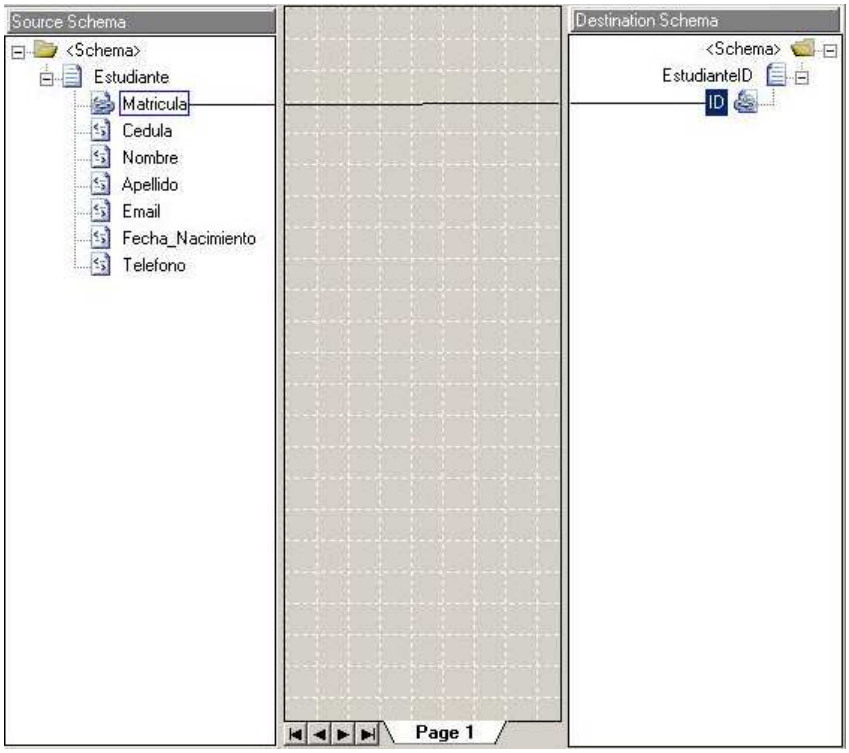

Figura 3.32 BizTalk para transformación de esquemas

3.5.2. Crear una relación entre las dos vistas.

Una vez creada la transformación se puede realizar una relación entre las dos vistas.

Para construir una relación entre dos vistas se debe especificar primero cual es la vista principal. Si se quiere tener un menú dentro de la

VistaEstudiante que despliegue los pagos, la vista principal sería VistaEstudiante, y por lo tanto debería estar dentro de la entidad Estudiante.

En el explorador de MetaData, debajo de Entidades – Vistas, existe la carpeta Relaciones.

Pulsando dos veces en la opción [Add Relationship] Añadir Relación, aparecerá la siguiente pantalla (Figura 3.33)

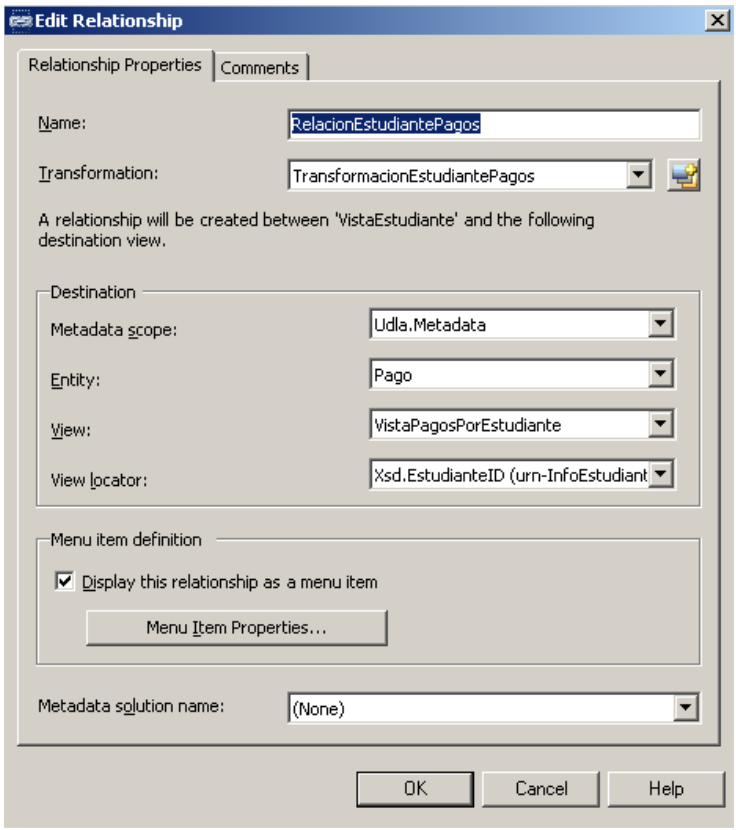

Figura 3.33 Pantalla para añadir una relación

Para construir esta relación se debe especificar lo siguiente:

Nombre: RelacionEstudiantePagos

Trasformación: Seleccionar – TrasnformaciónEstudiantePagos

Esquemas de destino:

Esquema de metadata: UDLA.Metadata

Entidad: Pago

Vista: VistaPagosPorEstudiante

Localizador: Xsd.EstudianteID

Una vez creada la relación en la vista respectiva es posible crear un menú para dicha vista.

- En la guía de metadata, ejecutar el asistente “Menu Item Definition Wizard” o Asistente para definición de menús. (Figura 3.34)

Figura 3.34 Pantalla inicial del asistente de creación de menús

- Pulse en Next.

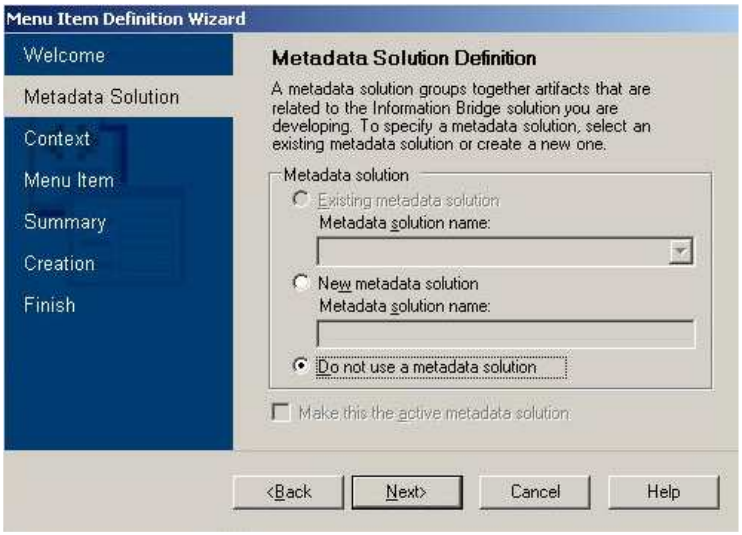

Figura 3.35 Definición de la solución de metadata

- No utilizar una solución metadata predefinida.(Figura 3.35)

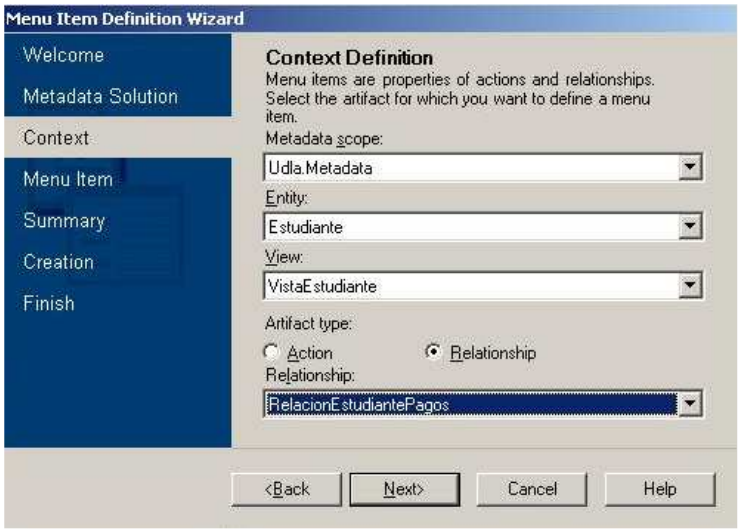

Figura 3.36 Definición de contexto

- Alcance de metadata – UDLA.Metadata (Figura 3.36)
- Entidad – Estudiante
- Vista – VistaEstudiante
- Relacion – RelacionEstudiantePagos

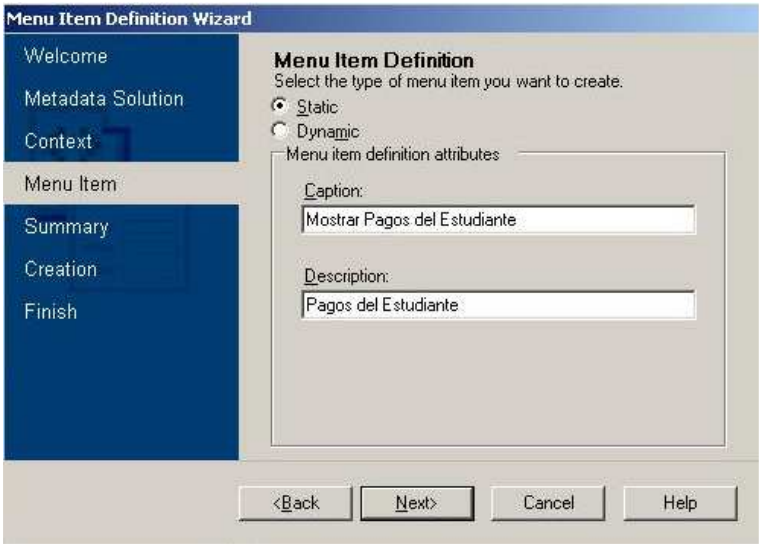

Figura 3.37 Propiedades del menú

- Texto a desplegar para el menú – Mostrar Pagos del Estudiante (Figura 3.37)
- Descripción – Pagos del Estudiante

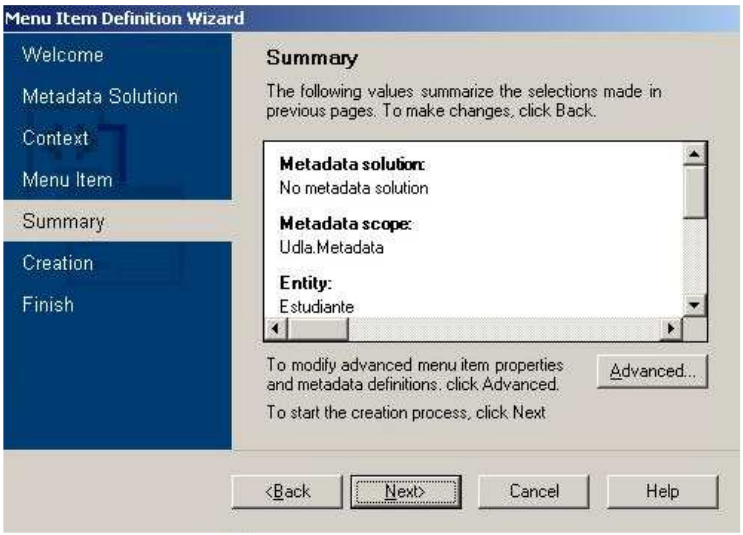

Figura 3.38 Resumen de las propiedades creadas por el asistente

- Pulse Next

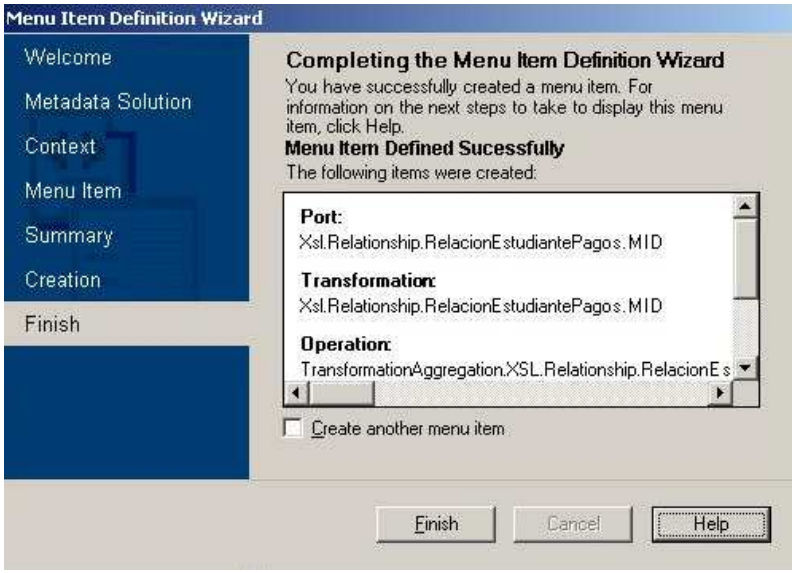

Figura 3.39 Finalización del asistente

- Pulse Finish

3.6. Definición de búsquedas.

La funcionalidad de búsqueda en el IBF ofrece a los usuarios una alternativa de poder acceder a los datos sin la necesidad de utilizar etiquetas inteligentes.

Esta funcionalidad es muy útil en escenarios en donde el usuario necesita observar datos dentro del IBF.

La pantalla del IBF puede ser desplegada desde el menú de Office Herramientas – Information Bridge.

Para el desarrollo de cualquier solución IBF es recomendable incluir la funcionalidad de búsqueda.

Para esto la guía metadata cuenta con un asistente para la creación de búsqueda.

La creación de un sistema de búsqueda, puede ser el paso más sencillo de una solución, ya que el IBF cuenta con un conjunto de componentes y regiones predefinidas las cuales hacen el sistema de búsqueda casi automático.

Para la creación del sistema de búsqueda, el primer paso es ejecutar el asistente que se encuentra en la guía de metadata.

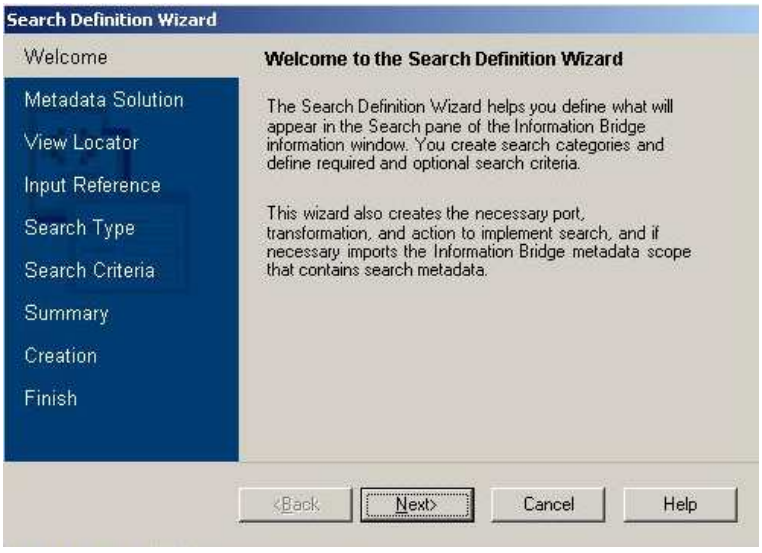

Figura 3.40 Pantalla de inicio del asistente de definición de búsqueda

Elegir la opción – No utilizar solución metadata predefinida (Do not use a metadata solution)

Pulsar en Next

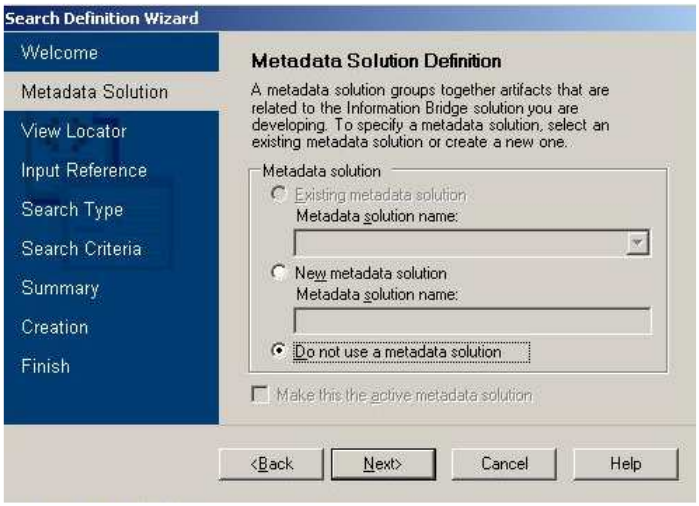

Figura 3.41 Definición de metadata

La búsqueda se realizará mediante un criterio en especial, en este caso sería el ID de Estudiante, el cual despliega la VistaEstudiante.

En la siguiente pantalla se debe especificar:

Alcance de Metadata – UDLA.Metadata

Entidad – Estudiante

Vista – VistaEstudiante

Localizador – Xsd.EstudianteID

Acción – Contexto de Entrada (Enter Context).

Pulse en Next

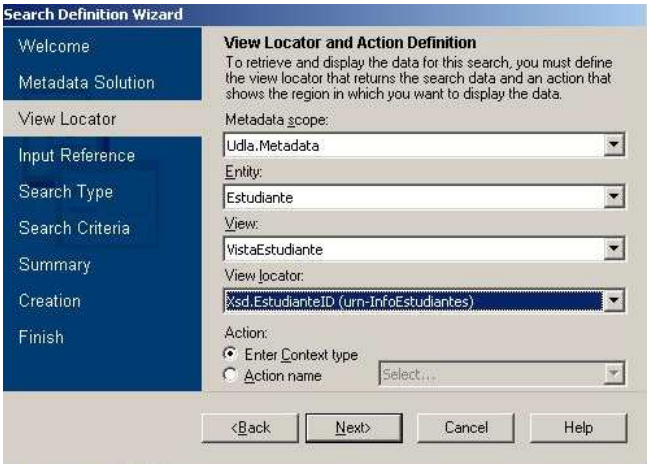

Figura 3.42 Definición de localizador y acciones para búsqueda

El siguiente paso es definir el esquema de entrada, por defecto el asistente crea una plantilla acorde con el esquema de entrada que recibe el servicio Web.

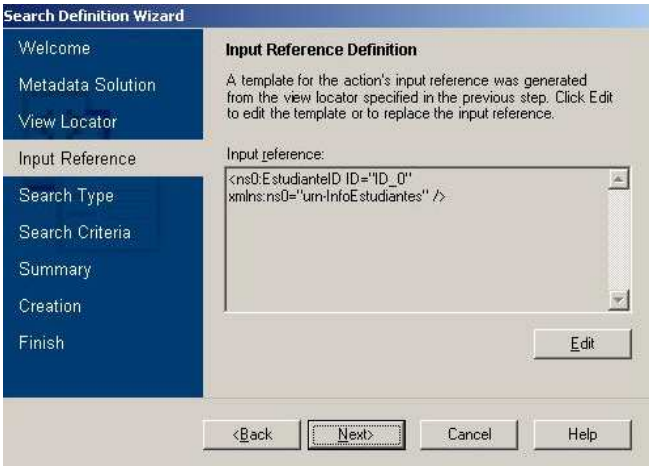

Figura 3.43 Esquemas de entrada

Ya que se pueden crear varios tipos de búsqueda con diferentes criterios por cada vista creada, es necesario agrupar estas búsquedas en categorías.

Se pueden tener varias categorías en el mismo panel de búsqueda.

Por ejemplo se puede especificar una categoría: Búsqueda de estudiantes o tener otra categoría denominada: Búsqueda de pagos por estudiante.

En este caso se define la categoría:

Estudiante por Matrícula.

Se puede desplegar los resultados en regiones personalizadas, a modo de ejemplo se utilizará la región creada por defecto.

(SearchPage)

Pulse en Next.

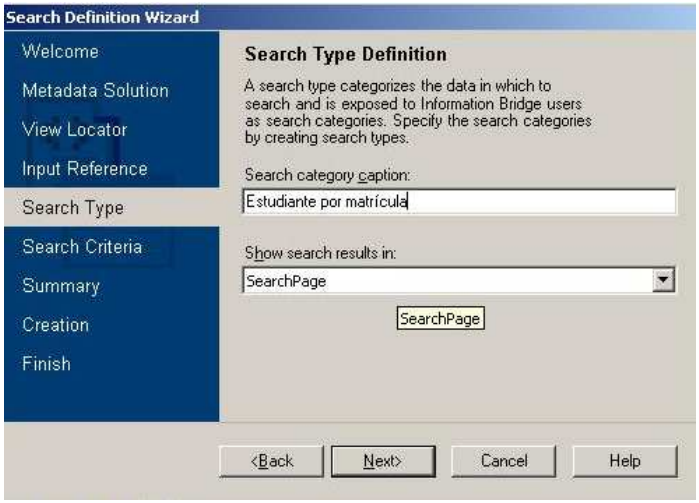

Figura 3.44 Tipo de búsqueda

El siguiente paso para crear la búsqueda es establecer el criterio a usar, normalmente corresponde al utilizado como localizador en la solución.

La información a llenar será:

Criterio – Matrícula

Descripción – Matrícula del Estudiante

Referencia - El esquema es creado automáticamente por el asistente apuntando al localizador inicial EstudianteID.

Tipo de control – Cuadro de texto.

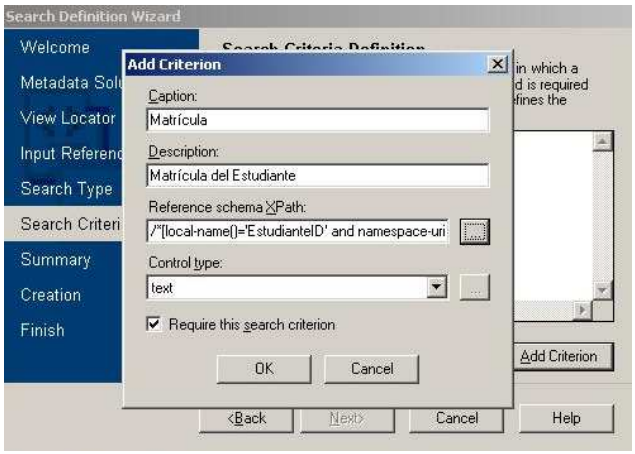

Figura 3.45 Propiedades de texto

El asistente identificará el localizador. En este caso es el localizador Matrícula el que servirá para desplegar la VistaEstudiante.

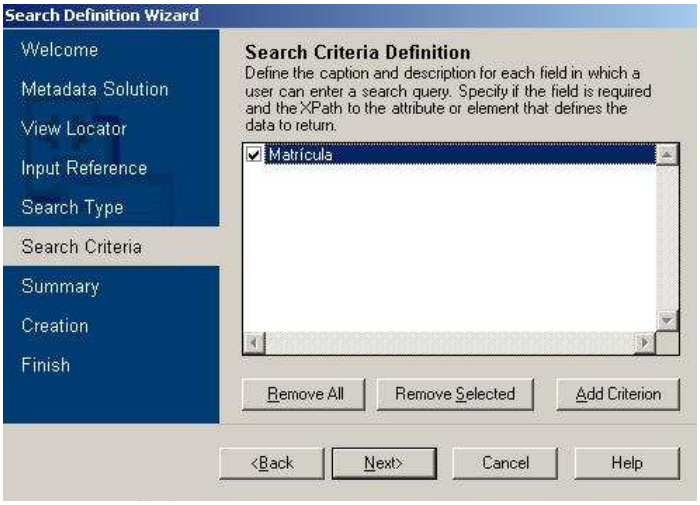

Figura 3.46 Definición del criterio de búsqueda

Pulse en Next.

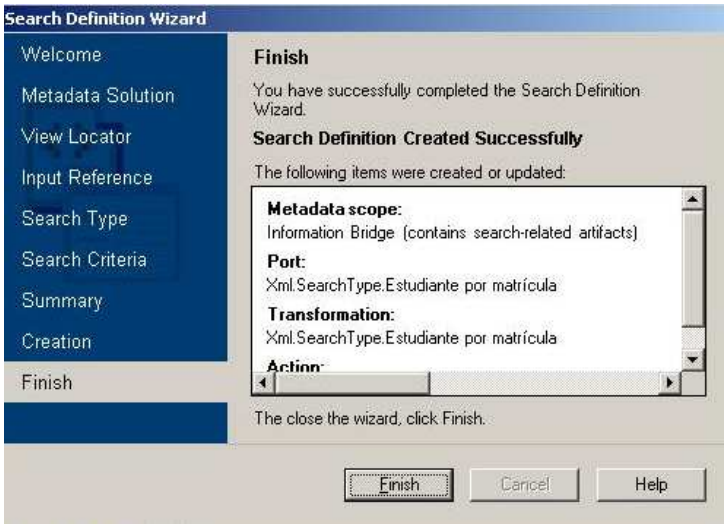

Figura 3.47 Finalización del asistente

A este punto se ha terminado con una solución completa IBF, en el siguiente y último capítulo se realizará una demostración del funcionamiento de la solución, para ello se han creado entidades y vistas adicionales con el objeto de darle mayor funcionalidad y así brindar un mejor entendimiento sobre el potencial del Information Bridge Framework.

4. CAPÍTULO IV - Ejecución de la solución IBF y Etiquetas Inteligentes.

El lado del cliente del Information Bridge Framework carga los ensamblados que se encuentran en algún servidor de Internet. Esto significa que los ensamblados no son seguros y el cliente no tendrá acceso a estos.

Ya que el IBF interactúa con procesos y documentos, los ensamblados creados por cada uno de los proyectos deben ser seguros.

4.1. Configuración de ensamblados seguros

El primer paso para configurar que un ensamblado sea seguro es crear una firma de seguridad.

Visual Studio 2003 cuenta con una herramienta específica para crear estas firmas de seguridad, se denomina Strong Name Tool (sn.exe).

Para crear una firma de seguridad se debe ejecutar el siguiente comando en la consola de Visual Studio 2003:

sn –k NombreFirmaSegura.snk

Al elemento creado se le denomina “Strong Name Key”.

Una vez creado el Strong Name Key se debe especificar en cada uno de los proyectos modificando el código por defecto del archivo AssemblyInfo.cs de la siguiente manera.

```
[assembly: AssemblyDelaySign(false)]
[assembly: AssemblyKeyFile(@"..\\..\\key.snk")]
[assembly: AssemblyKeyName("")]
```

Se debe compilar el proyecto y guardar cambios en el archivo AssemblyInfo.cs.

El siguiente paso es registrar el ensamblado y darle permisos de confianza, esto se realiza en la “Configuración del .Net Framework 1.1” Este aplicativo se ubica en **Herramientas Administrativas – Asistente de Configuración del .Net Framework 1.1 (Figura 4.1)**

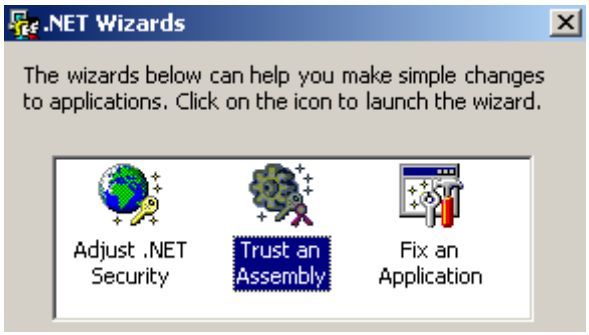

Figura 4.1 Pantalla de inicio del asistente de configuración de .NET framework 1.1

En la primera pantalla de la configuración del .net framework 1.1 se debe escoger la opción “Trust an Assembly” o en español “Configurar seguridad de ensamblados”.

En la siguiente pantalla se puede configurar el alcance de la seguridad, se puede establecer la seguridad del ensamblado tanto a nivel de máquina como a nivel de usuario. (Figura 4.2)

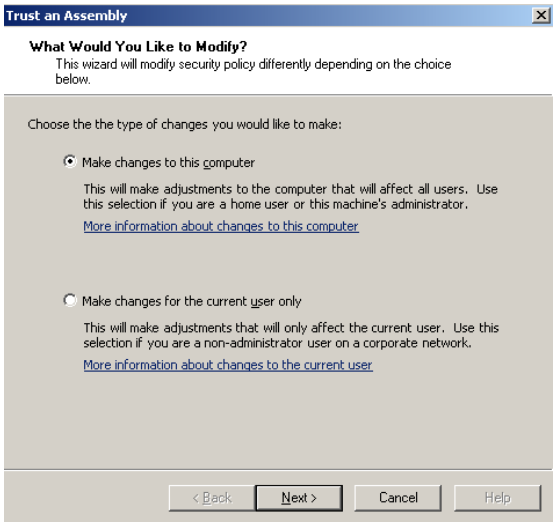

Figura 4.2 Configuración del alcance de la seguridad

Pulse Next

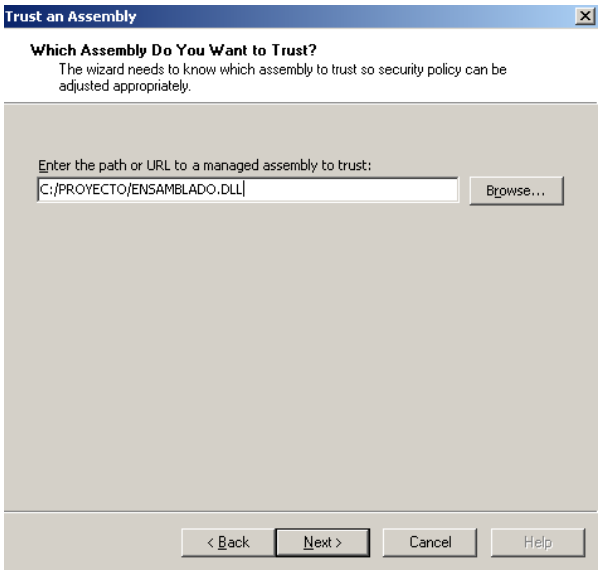

Figura 4.3 Pantalla para establecer el ensamblado en cuestión

Luego de lo cual podemos establecer el nivel de confianza del ensamblado en cuestión, se debe escoger el nivel mayor de confianza (Full Trust).

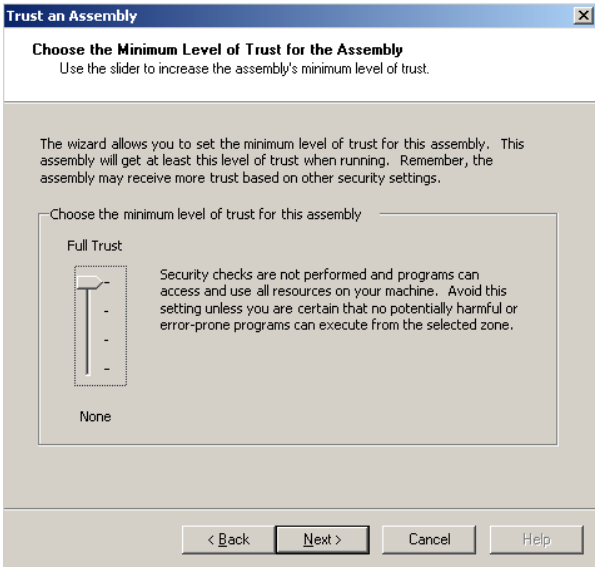

Figura 4.4 Pantalla donde se establece el nivel de confianza

Pulse en Next.

A este punto el ensamblado del proyecto podrá ser accedido por el cliente del IBF sin dar un error de seguridad en la aplicación.

4.2. Ejecución de la solución IBF

El último paso del desarrollo de una solución IBF es comprobar que esta se ejecuta sin ningún error en el entorno de desarrollo.

Para ello el explorador de metadata permite ejecutar y depurar la solución sin necesidad de estar en el entorno Office utilizando para ello el navegador de la metadata hasta llegar al elemento:

Entidad – Vista – Acciones.

- Pulse el botón derecho en la acción a ejecutar.
- Ejecutar. (Figura 4.5)

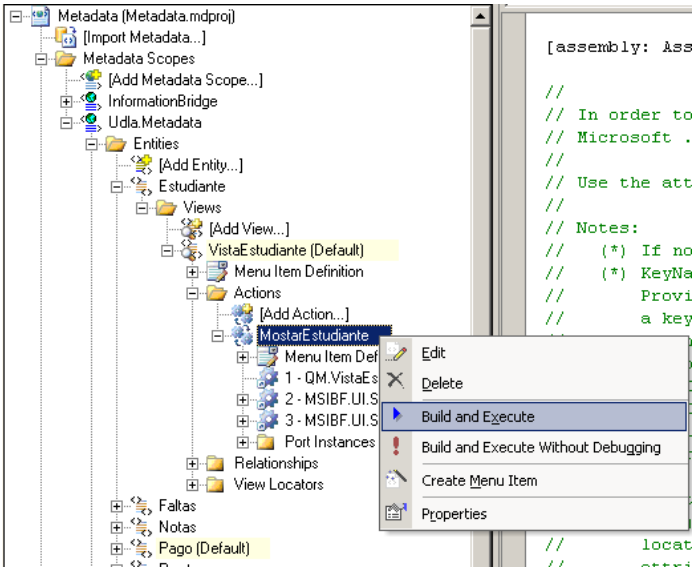

Figura 4.5 Ejecución de la solución IBF

Si necesitamos que el entorno de desarrollo pueda simular la ejecución como si estuviera en un entorno de Office es necesario establecer un parámetro de referencia XML el cual contenga el localizador, en este caso en especial el localizador será EstudianteID.

<ns0:EstudianteID ID="100029" xmlns:ns0="urn-InfoEstudiantes" />

En otras palabras el parámetro simulará que el IBF sea ejecutado por una etiqueta inteligente. (Figura 4.6)

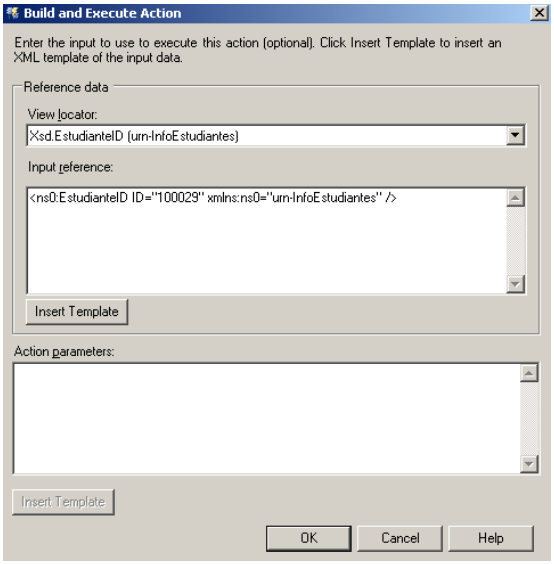

Figura 4.6 Parametro de referencia XML

Como resultado de la ejecución se puede ver el panel del IBF desplegando la información del estudiante. (Figura 4.7)

Figura 4.7 Pantalla del panel IBF

Dado que una solución IBF debe ser ejecutada tanto manual como automáticamente por el usuario en un entorno de Office, se debe desarrollar etiquetas inteligentes acordes con la plataforma.

4.3. Desarrollo de Etiquetas Inteligentes (Smart Tags - IBF)

Una etiqueta inteligente son datos reconocidos que permite invocar acciones dependiendo del contexto de los datos. Los reconocedores de las etiquetas inteligentes son módulos que implementan una funcionalidad para reconocer dichas etiquetas, mientras que las acciones de las etiquetas inteligentes son módulos que usan las etiquetas para realizar cierta operación. Por ejemplo, en los Sistemas Office se usan reconocedores de etiquetas inteligentes para encontrar direcciones, nombres, números de teléfono, direcciones de correo electrónico, etc., con lo que se podrían utilizar acciones de etiquetas inteligentes para cada uno de estos datos. Por ejemplo si se reconoce una dirección de correo electrónico una acción sería: Enviar correo o Añadir dirección a contactos en Outlook.

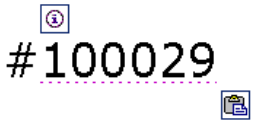

Figura 4.8 Ejemplo de la activación de una etiqueta inteligente

Las etiquetas inteligentes del IBF encapsulan estos datos transformándolos en un elemento XML llamado ContextInformation. Los

datos de contexto son simples elementos que son concernientes a una etiqueta inteligente. En el esquema del IBF esta es la manera de invocar una acción y ejecutar la llamada de la plataforma para desplegar datos referenciados.

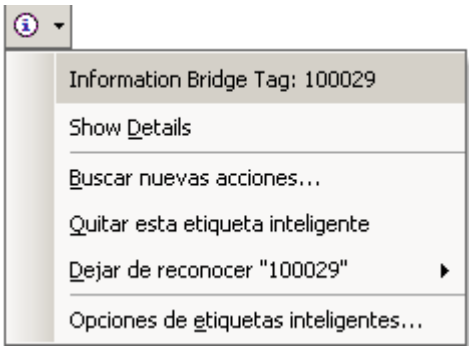

Figura 4.9 Acciones por defecto de una etiqueta inteligente

El elemento XML ContextInformation soporta cuatro atributos y un elemento hijo, los cuales están descritos en la siguiente tabla:

Elementos y Atributos	Descripción
Atributo: ViewName	Este atributo debe especificar la vista que debe ser desplegada para los datos de Contexto
Atributo: ReferenceSchemaName	Este atributo debe especificar el esquema de referencia de la referencia utilizada para desplegar una vista específica. No es necesario incluir este atributo si se requiere tener una vista por defecto.

Atributo: EntityName	Este atributo debe especificar la entidad la cual contiene la vista.
Atributo: MetadataScopeName	Este atributo debe especificar el nombre de la Metadata que contiene la entidad.
Elemento: Reference	Este elemento encapsula el texto XML que representa una referencia.

Un ejemplo del elemento XML ContextInformation para desplegar la vista de cierto estudiante seria así:

```
<ContextInformation
        http://schemas.microsoft.com/InformationBridge/2004/ContextInformation"
        MetadataScopeName="Udla.InfoEstudiante"
        EntityName="Estudiante"
        ViewName="VistaEstudiante"
        ReferenceSchemaName="Xsd.ReferenciaMatrícula (urn-InfoEstudiante)">
<Reference>
        <ReferenciaMatrícula
        xmlns="urn-InfoEstudiante"
        xmlns:ibf=http://schemas.microsoft.com/InformationBridge/2004
        ibf:MetadataScopeName="InfoEstudiante"
        ibf:EntityName="Estudiante"
        ibf:ViewName="VistaEstudiante"
        ibf:ReferenceSchemaName="xsd.ReferenciaMatrícula (urn-OrderFulfillment-Data)"
        ID="1000">
        </CustomerIDReference>
</Reference>
</ContextInformation>
```

En el atributo de referencia del ContextInformation se debe incluir el elemento que representa la conexión con la información en este caso es la Matrícula del Estudiante.

4.3.1. Implementación del Reconocedor

Un reconocedor es simplemente una Clase con referencias específicas. Para crear el proyecto debe incluir las siguientes referencias: Microsoft.InformationBridge.Framework.Interfaces, Microsoft.InformationBridge.Framework.UI.Interop, y Microsoft.Office.Interop.SmartTag.

El paso siguiente es definir las referencias de las etiquetas inteligentes. En resumen esto es crear una lista de términos que deben ser reconocidos, esta lista de términos debe corresponder a varias referencias que pueden ser pasadas hacia las acciones del Information Bridge.

Estas referencias se pueden obtener directamente de una base de datos o mediante un servicio Web.

4.3.2. Definición de Referencias

Antes de que el reconocedor pueda ser usado, se debe definir las referencias o términos que necesitan ser reconocidos. La lista de términos a utilizar debe corresponder a las referencias que se pasan

hacia las acciones del Information Bridge. En si estas referencias son un listado de elementos XML que van a ser leídos por el Information Bridge.

Un ejemplo del archivo XML generado sería

```
<?xml version = “1.0” encoding=”utf-8”?>
<References>
        <Estudiante>
                <Matrícula Estudiante Matrícula =”10323”>
                <Matrícula Estudiante Matrícula =”15623”>
                <Matrícula Estudiante Matrícula =”19750”>
        </Estudiante>
</References>
```

Mediante estos elementos XML que serán generados a través de una consulta a un servicio Web, las etiquetas inteligentes se activarán automáticamente si el texto ingresado corresponde a alguno de los términos mencionados.

4.3.3. Reconociendo las etiquetas inteligentes

El método reconocedor acepta cuatro parámetros. Parámetro de texto, el cual es el texto que será escrito en la aplicación office ej. Word, que será enviado hacia el reconocedor para determinar si es o no una etiqueta inteligente. Normalmente una aplicación como Word hace el reconocimiento de las etiquetas inteligentes línea a línea, para esto es necesario el segundo parámetro el cual determina si el parámetro de texto es una palabra, un párrafo, un carácter, una celda de Excel, etc. El tercer parámetro es el lenguaje en el cual esta escrito el parámetro de

texto. Y finalmente el cuarto parámetro resulta ser la llamada a los diferentes métodos que el reconocedor debe invocar por cada parámetro de texto.

Desarrollar etiquetas inteligentes para IBF es similar a crear etiquetas inteligentes simples, la diferencia es que no es necesario crear manejadores de acciones, ya que el IBF cuenta con su propio creador de acciones meditante la lectura de los elementos XML del ContextInformation. (Por defecto Show Details).

Microsoft ofrece las librerías e interfaces necesarias para la creación simple de etiquetas inteligentes.

Las interfaces a utilizar para el desarrollo de las etiquetas inteligentes que deben ser implementadas por la clase son:

ISmartTagRecognizer

ISmartTagRecognizer2.

Sin embargo, aunque estas interfaces son las encargadas de levantar los servicios y componentes para el reconocimiento de texto, se necesita programar dos métodos importantes miembros de dichas interfaces.

- SmartTagInitialize
- Recognize2

Cada vez que la aplicación de etiquetas inteligentes es cargada dentro del entorno Office, el método SmartTagInitialize es disparado. Este

evento permite insertar código, el cual será el encargado de crear o extraer datos en formato XML los cuales serán los elementos a reconocer.

Nota. Los datos pueden ser extraídos tanto de un archivo XML predefinido o mediante el consumo de un servicio Web.

El método Recognizer2 de la interfaz ISmartTagRecognizer2 es disparado por la aplicación Office cada vez que lee el documento en búsqueda de texto que potencialmente puede ser una etiqueta inteligente. Este método recibe como parámetro una cadena de texto la cual es comprobada contra los elementos XML cargados en el método SmartTagInitialize, si encuentra concordancia con algún elemento este es adjuntado hacia la variable ContextInformation.

La cadena de texto XML completa es pasada hacia el IBF mediante la acción por defecto Show Details. Como se explico anteriormente esta cadena le dice al IBF cual entidad, vista, y acción debe invocar.

Para ver el código completo de estos métodos ver Anexo #

Finalmente para que la clase reconocedor de Matrícula sea reconocida como parte del conjunto de etiquetas inteligentes se debe registrar el ensamblado en el registro del sistema mediante el siguiente archivo de registro.

Windows Registry Editor Version 5.00

[HKEY_CURRENT_USER\Software\Microsoft\Office\Common\Smart Tag\Recognizers\SmartTag.Reconocedor]

"Filename"="C:\SmartTag.dll"

"Managed"=dword:00000001

En el cual Filename es la ruta en donde se encuentra almacenado el ensamblado creado.

Una vez realizada esta acción se puede ejecutar cualquier aplicación de Office Ej. Microsoft Word

En las opciones de auto corrección – Etiquetas Inteligentes, se puede observar que se ha creado un nuevo reconocedor denominado Information Bridge Tag. (Figura 4.10)

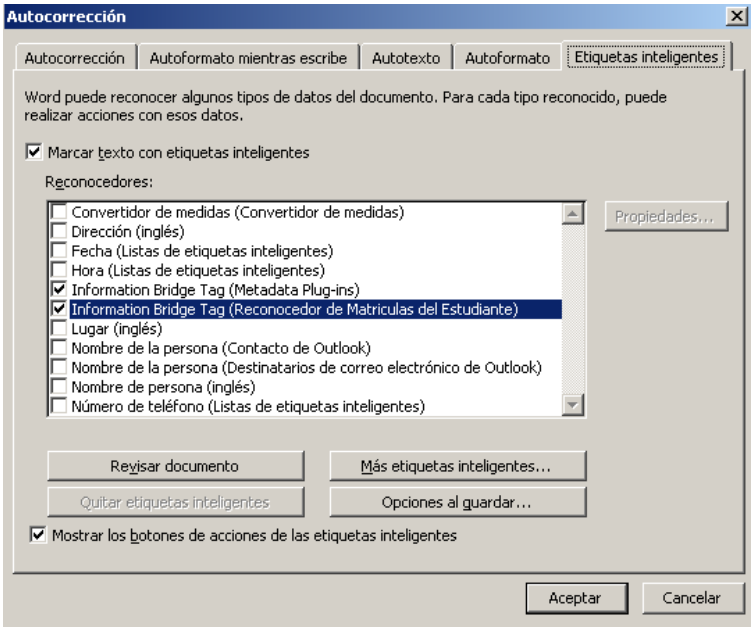

Figura 4.10 Reconocedor creado

Para poder probar el funcionamiento se puede escribir en el documento cualquier elemento creado ya sea en XML o llamado desde un servicio Web. En este caso un número de Matrícula de un estudiante:

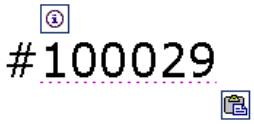

Figura 4.11 Elemento reconocido como matricula de un estudiante

Si se accede a las acciones de las etiquetas inteligentes se encontrará la acción por defecto del IBF “Show Details” la cual levantara el IBF desplegado la vista creada.

Figura 4.12 Despliegue de la vista estudiante en el IBF mediante una etiqueta inteligente

.

Se ha terminado con el desarrollo de una solución IBF desde la creación de los servicios Web a consumir hasta el despliegue de la información en el panel del IBF mediante etiquetas inteligentes.

Cabe recalcar que esta solución puede ser ampliada según los requerimientos del negocio.

4.4. Otras configuraciones

El IBF cuenta con una herramienta para configuraciones adicionales del entorno. Estas se dividen en dos:

- Configuración del cache
- Manejo del cache

La configuración del cache permite administrar el tiempo de expiración y el espacio de disco a utilizar para el cache de la metadata.

Además de poder establecer el URL del servicio de metadata.

Por otro lado el manejo de cache, es aquel que permite la importación, eliminación y actualización del cache de la metadata.

Para acceder a las figuras del administrador del IBF ver anexo 1

CONCLUSIONES

- La plataforma puentes de información o Information Bridge Framework, ofrece una gama muy amplia de herramientas y asistentes para la creación de soluciones que integren la información de la línea de negocio para que sea expuesta en los sistemas Office.
- El IBF ayuda a los desarrolladores a crear dichas soluciones disminuyendo considerablemente la línea de aprendizaje y de tiempo de desarrollo.
- En el contexto del cliente, el IBF cuenta con una plataforma completamente integrada hacia los sistemas Office, lo cual intuye un aumento considerable en la productividad y en agilidad en la toma de decisiones ya que el usuario no tendrá que salir de su entorno de trabajo (Sistemas Office) para acceder a la información.
- Por último, el uso de etiquetas inteligentes alertando al usuario en tiempo real la existencia de elementos referente a su negocio es de gran ayuda en situaciones laborales.

RECOMENDACIONES

Como se mencionó a lo largo de este trabajo de titulación, el IBF debe ser cuidadosamente analizado en su funcionamiento ya que no todos los escenarios son favorables para la implementación de una solución de este tipo.

A continuación se mencionan algunos puntos en donde el desarrollo de una solución IBF no es necesaria:

- En el caso que los usuarios finales no utilicen los sistemas Office muy frecuentemente.
- Cuando exista un sistema que permita la integración de información de las diferentes líneas de negocio.
- Si uno de los requerimientos es la actualización de información dentro de la solución.

Hay que recordar que el IBF fue diseñado exclusivamente para la exposición de datos, con esto no se quiere decir que no es posible realizar actualización de información.

Se debe tener mucha cautela y analizar si es estrictamente necesaria la implementación de dichas funcionalidades y tomar las medidas de seguridad respectivas.