



UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAR DE INGENIERÍA EN CIENCIAS APLICADAS

**ESCUELA DE INGENIERÍA EN SISTEMAS
COMPUTACIONALES**

**PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE
INGENIERO EN SISTEMAS COMPUTACIONALES**

**TEMA: ESTUDIO DE APLICACIONES RIA (RICH
INTERNET APPLICATIONS) UTILIZANDO
MICROSOFT SILVERLIGHT**

**APLICATIVO: IMPLEMENTACIÓN DE UN PROTOTIPO
DE SOFTWARE DE E-LEARNING PARA LA
ENSEÑANZA DE SQLSERVER UTILIZANDO
MICROSOFT SILVERLIGHT**

AUTOR: LUIS SANDRO GUEVARA PÉREZ

DIRECTOR: ING. MAURICIO REA

IBARRA, 2011



UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

**AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD
TÉCNICA DEL NORTE**

1. IDENTIFICACIÓN DE LA OBRA

La universidad técnica del norte dentro del proyecto repositorio digital institucional determina la necesidad de disponer de textos completos en formato digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la universidad.

Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto para lo cual pones a disposición la siguiente información.

DATOS DE CONTACTO	
CÉDULA DE IDENTIDAD	1002798641
APELLIDOS Y NOMBRES	GUEVARA PÉREZ LUIS SANDRO
DIRECCIÓN	QUITO, FRANCISCO MATIZ Y XIRONZA
EMAIL	lsandroguevarap@hotmail.com
TELÉFONO FIJO	
TELÉFONO MOVIL	081465610

DATOS DE LA OBRA	
TITULO	ESTUDIO DE APLICACIONES RIA (RICH INTERNET APPLICATIONS) UTILIZANDO MICROSOFT SILVERLIGHT
AUTOR	GUEVARA PÉREZ LUIS SANDRO
FECHA	25 de Julio del 2011
PROGRAMA	Pregrado
TITULO POR EL QUE	INGENIERÍA EN SISTEMAS COMPUTACIONALES
DIRECTOR	ING. MAURICIO REA

2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD

Yo, Luis Sandro Guevara Pérez, con cédula de de identidad Nro. 1002798641, en calidad de autor y titular de los derechos patrimoniales de la obra o trabajo de grado descrito anteriormente, hago entrega del ejemplar respectivo en forma digital y autorizo a la Universidad Técnica del Norte, la publicación de la obra en el Repositorio Digital Institucional y uso del archivo digital en la biblioteca de la universidad con fines académicos, para ampliar la disponibilidad del material y como apoyo a la educación, investigación y extensión, en concordancia con la Ley de Educación Superior Artículo 143.



UNIVERSIDAD TÉCNICA DEL NORTE

CESION DE DERECHO DE AUTOR DEL TRABAJO DE GRADO A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

Yo, Luis Sandro Guevara Pérez, con cédula de identidad Nro. 1002798641, manifiesto mi voluntad de ceder a la Universidad Técnica del Norte los derechos patrimoniales consagrados en la ley de propiedad intelectual del Ecuador, artículos 4,5 y 6 en calidad de autor del trabajo de grado denominado: “ESTUDIO DE APLICACIONES RIA (RICH INTERNET APPLICATIONS) UTILIZANDO MICROSOFT SILVERLIGHT”, que ha sido desarrollado para otra por el título de de: **Ingeniero en Sistemas Computacionales**, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En mi condición de autor me reservo los derechos morales de la obra antes citada.

En concordancia suscribo este documento en el momento en el que hago la entrega del trabajo final en formato impreso y digital a la biblioteca de la Universidad Técnica del Norte.

Firma:

Nombre: Luis Sandro Guevara Pérez

Cédula: 1002798641

Ibarra a los 25 días del mes de Julio del 2011

CERTIFICACIÓN

El Sr. Luis Sandro Guevara Pérez, cédula de identidad Nro. 1002798641, a trabajado en el desarrollo del proyecto: “ESTUDIO DE APLICACIONES RIA (RICH INTERNET APPLICATIONS) UTILIZANDO MICROSOFT SILVERLIGHT”, con el aplicativo: “IMPLEMENTACIÓN DE UN PROTOTIPO DE SOFTWARE DE E-LEARNING PARA LA ENSEÑANZA DE SQLSERVER UTILIZANDO MICROSOFT SILVERLIGHT” previo a la obtención del título de Ingeniero en Sistemas Computacionales, realizándolo con interés profesional y responsabilidad, el cual certifico en honor a la verdad.

Ing. Mauricio Rea
DIRECTOR DE TESIS

AGRADECIMIENTO

*A Dios que colmó de bendiciones mi vida
para poder seguir adelante.*

*A mis padres
que me apoyaron en el transcurso de mi educación*

*A mi esposa e hijo
que me dieron la fuerza necesaria para poder culminar este mi primer objetivo*

SANDRO

DEDICATORIA

*A mi familia por su apoyo incondicional,
cada logro mío fue suyo también.*

*A mi hijo por ser la fuente inagotable
de mi fuerza y esperanza para la terminación del presente proyecto.*

El doble esfuerzo siempre trae sus frutos.

CONTENIDO

CAPÍTULO I. INTRODUCCIÓN	1
1.1 INTERFAZ DE USUARIO	1
1.1.1 DEFINICIÓN.....	1
1.1.2 TIPOS DE INTERFACES.....	1
1.1.3 ELEMENTOS DE LA INTERFAZ DE USUARIO.....	2
1.1.4 MODELOS DE UI SEGÚN EL PUNTO DE VISTA.....	4
1.2 MICROSOFT SILVERLIGHT	8
1.2.1 INTRODUCCIÓN.....	8
1.2.2 ARQUITECTURA.....	10
1.2.2.1 Apartados de la arquitectura.....	11
1.2.2.2 Componentes de presentación básicos.....	12
1.2.2.3 .NET Framework para Silverlight.....	13
1.2.2.4 Características de programación adicionales de Silverlight.....	14
1.2.3 LENGUAJE XAML (Extensible Application Markup Language).....	15
1.2.3.1 Espacios de nombres XAML.....	16
1.2.3.2 Declara objetos.....	17
1.2.3.3 Establecer propiedades.....	18
1.2.3.4 Extensiones de marcado.....	19
1.2.3.5 Convertidores de tipo.....	20
1.2.4 INTEGRACIÓN A BROWSER Y DESKTOP.....	21
1.2.4.1 Integración con el browser.....	21
1.2.4.2 Modelo de objetos de documentos.....	22
1.2.4.3 Puente HTML: interacción entre código HTML y código administrado.....	23
1.2.4.4 Silverlight fuera del navegador.....	23
1.2.4.5 Modelos de ejecución e interoperabilidad en Silverlight 4.....	24
CAPÍTULO II.- RIA (APLICACIONES DE INTERNET ENRIQUECIDAS)	26
2.1. INTRODUCCIÓN	26
2.2. APLICACIONES RIA	27
2.2.1. CARACTERÍSTICAS.....	27
2.2.2. ARQUITECTURA.....	30
2.2.2.1. Interacción entre el cliente y el servidor.....	30
2.2.2.2. Comportamiento del servidor.....	31
2.2.2.3. El “Motor” Cliente.....	33

2.2.3.	VENTAJAS	34
2.1.4	DESVENTAJAS	35
2.3.	TECNOLOGÍAS DE DESARROLLO	36
2.3.1.	ADOBE FLASH.....	36
2.3.2.	MICROSOFT SILVERLIGHT	38
2.3.3.	ADOBE FLEX	40
2.3.4.	ICEFACES	43
CAPÍTULO III.- ESTRUCTURANDO LA APLICACIÓN CON MICROSOFT SILVERLIGHT		46
3.1	CONTROLES PARA LAS INTERFACES DE USUARIO.....	46
3.1.1	BLOQUES DE CONSTRUCCIÓN	46
3.1.1.1.	DependencyObject.....	46
3.1.1.2.	Threading y la interfaz de usuario	48
3.1.1.3.	UIElement	48
3.1.1.4.	La clase Control.....	53
3.1.2.	DISEÑO DE GESTIÓN Y CONTROL DE AGRUPACIÓN	55
3.1.2.1	Canvas	56
3.1.2.2.	StackPanel	57
3.1.2.3.	Grid	58
3.1.2.4.	DockPanel	58
3.1.2.5.	WrapPanel.....	59
3.1.2.6.	TabControl	61
3.1.2.7.	ViewBox.....	63
3.1.3.	Forms Controls.....	64
3.1.3.1.	Button Controls	64
3.1.3.2.	TextBox	65
3.1.3.3.	PasswordBox.....	66
3.1.3.4.	RichTextBox.....	66
3.1.3.5.	AutoCompleteBox	67
3.2.	ACCESO A DATOS	68
3.2.1.	ENLACE CON DATOS	68
3.2.1.1.	Sintaxis del enlace Binding.....	69
3.2.1.2.	Binding en tiempo de ejecución	69
3.2.1.3.	Binding en tiempo de diseño	69
3.2.1.4.	La elección de un modo de unión	70
3.2.2.	LA COMPRENSIÓN DEL ORIGEN DEL ENLACE.....	73
3.2.2.1.	Enlace a una propiedad	73

3.2.2.2.	Enlazar a un objeto.....	74
3.2.2.3.	Binding con elemento UI	75
3.2.2.4.	Binding a un elemento índice	76
3.2.2.5.	Binding a un elemento clave (cadena indexada)	77
3.2.2.6.	Binding a una colección completa	78
3.2.3.	PERSONALIZACIÓN DE LA PANTALLA	78
3.2.3.1.	Los valores de formato.....	78
3.2.3.2.	La conversión de los valores durante el enlace	79
3.2.3.3.	Proporcionar valores de reserva por defecto	81
3.2.3.4.	Manejo de valores nulos.....	82
3.2.4.	CREACIÓN DE PLANTILLAS DE DATOS	82
3.2.4.1.	Una plantilla de datos con un ContentControl	82
3.2.4.2.	Representación ItemsControl con una plantilla de datos.....	84
3.3.	VALIDACIONES	85
3.3.1.	Excepción de propiedad basada en la validación	85
3.3.1.1.	Manipulación de errores en una excepción de validación.....	85
3.3.1.2.	Código de validación personalizada	86
3.3.1.3.	Validación de la pantalla de error	86
3.3.2.	Validación síncrona con IDataErrorInfo.....	87
3.3.2.1.	La interfaz IDataErrorInfo	87
3.3.2.2.	Simple validación con IDataErrorInfo	88
3.3.2.3.	Combinación de las excepciones y IDataErrorInfo	89
3.3.3.	VALIDACIÓN ASÍNCRONA CON INotifyDataErrorInfo.....	90
3.3.3.1.	La interfaz INotifyDataErrorInfo	90
3.3.3.2.	Implementación de la interfaz	91
3.4.	PATRÓN MVVM.....	92
3.4.1.	EL MODELO	92
3.4.2.	LA VISTA	92
3.4.3.	EL MODELO DE VISTA (NUESTRO CONTROLADOR O PRESENTADOR)	93
3.4.4.	LA VISTA Y EL MODELO DE VISTA.....	94
3.4.5.	EL MODELO DE VISTA Y EL MODELO.....	94
3.4.6.	INFRAESTRUCTURA BÁSICA PARA MVVM.....	95
3.5.	WCF RIA SERVICES	95
3.5.1.	WCF INTEGRACIÓN	96
3.5.1.1.	Servicios de dominio	96
3.5.1.2.	Servicios de dominio y sus fuentes de datos	97

3.5.2.	ASEGURAR UNA SOLUCIÓN DE RIA SERVICIOS	97
3.5.2.1.	Seguridad de los servicios RIA WCF	97
3.5.2.2.	Lista de verificación de seguridad	98
3.5.3.	HERRAMIENTAS Y DOCUMENTACIÓN	100
CAPÍTULO IV.- COMPLETANDO LA EXPERIENCIA CON MICROSOFT SILVERLIGHT		101
4.1.	INTEGRACIÓN CON MEDIA(VIDEO Y AUDIO)	101
4.1.1.	INFORMACIÓN GENERAL SOBRE AUDIO Y VÍDEO	101
4.1.1.1.	Propiedades de MediaElement	101
4.1.1.2.	Controlar la reproducción de contenido multimedia de forma interactiva	102
4.1.1.3.	Marcadores de escala de tiempo (puntos de sincronización).....	103
4.1.1.4.	Definir marcadores multimedia.....	104
4.1.1.5.	Listas de reproducción del servidor	105
4.1.1.6.	Registros del servidor.....	106
4.1.1.7.	MediaStreamSource	106
4.1.1.8.	Administración de derechos digitales (DRM).....	107
4.1.2.	FORMATOS MULTIMEDIA, PROTOCOLOS Y CAMPOS DE REGISTRO COMPATIBLES	107
4.1.2.1.	Formatos no compatibles	110
4.1.2.2.	Protocolos compatibles	110
4.1.2.3.	Servidores proxy y transmisión por secuencias en Silverlight	111
4.1.3.	INFORMACIÓN GENERAL SOBRE CÁMARAS WEB Y DISPOSITIVOS	111
4.2.	ANIMACIÓN	113
4.2.1.	INFORMACIÓN GENERAL SOBRE ANIMACIONES	113
4.2.1.1.	Tipos de animación	113
4.2.1.2.	Funciones de entradas y salidas lentas	115
4.2.2.	ANIMACIONES DE FOTOGRAMAS CLAVE	117
4.2.2.1.	Tipos de animación de fotograma clave.....	117
4.2.2.2.	Funciones de entradas y salidas lentas	118
4.2.3.	TRABAJAR CON ANIMACIONES MEDIANTE PROGRAMACIÓN	118
4.3.	GRÁFICOS	118
4.3.1.	FORMAS Y DIBUJOS	118
4.3.1.1.	Objetos de formas	119
4.3.1.2.	Utilizar Trazados y geometrías	119
4.3.1.3.	Pintar Formas	120
4.3.1.4.	Trasformar Formas.....	121
4.3.2.	PINCELES	122

4.3.2.1.	Pintar un área con un color solido	122
4.3.2.2.	Pintar un área con color degradado.....	124
4.3.2.3.	Degradados lineales.....	124
4.3.2.4.	Degradados radiales	125
4.3.3.	INFORMACIÓN GENERAL SOBRE OBJETOS VIDEOBRUSH.....	126
4.3.3.1.	Propiedades de VideoBrush	127
4.3.3.2.	Relación entre VideoBrush y MediaElement.....	127
4.3.4.	GEOMETRÍAS	128
4.3.4.1.	Comparación entre geometrías y formas	128
4.3.4.2.	Propiedades comunes de un objeto Geometry.....	128
4.3.4.3.	Geometrías de Trazado.....	129
4.3.5.	TRANSFORMACIONES	129
4.3.5.1.	Clases de transformación.....	130
4.3.5.2.	Propiedades comunes de transformación.....	131
4.3.6.	IMÁGENES	132
4.3.6.1.	Ajustar una imagen	132
4.3.6.2.	Recortar una imagen.....	133
4.4.	IMPRESIÓN	134
4.4.1.	Mostrar cuadro de dialogo Imprimir.....	134
4.4.2.	Realizar la operación de impresión	135
4.4.3.	CONFIGURAR Y LIMPIAR UNA OPERACIÓN DE IMPRESIÓN.....	137
4.5.	MICROSOFT EXPRESSION BLEND.....	137
4.5.1.	Funcionalidad de Expression Blend	138
4.5.2.	Aplicaciones que funcionan con Expression Blend.....	138
4.5.3.	Características de Expression Blend	139
4.5.4.	Tipos de Aplicaciones	140
4.5.5.	Procedimientos recomendados.....	141
	CAPÍTULO V.- APLICATIVO.....	143
5.1.	ALCANCE	143
5.1.1.	Opciones de Administrador	143
5.1.2.	Opciones de Docentes.....	144
5.1.3.	Opciones de Alumnos.....	145
5.1.4.	Opciones de Invitados	146
5.2 .	PLATAFORMA Y ARQUITECTURA	147
5.2.1.	DISEÑO DE LA ARQUITECTURA	147
5.2.1.1.	Capa de Acceso a Datos.....	148

5.2.1.2.	Capa de Negocio	149
5.2.1.3.	Capa de Servicios(Servicios Web)	150
5.2.1.4.	Capa de Presentación	151
5.2.1.5.	Capa de Infraestructura Transversal.....	151
5.3.	Análisis y diseño	154
5.3.1.	Herramientas utilizadas	154
5.3.1.1.	Visual Studio 2010.....	154
5.3.1.1.1.	Entorno de desarrollo integrado (IDE)	154
5.3.1.1.1.1.	Sistema de proyectos	155
5.3.1.1.1.1.	Editores y diseñadores.....	156
5.3.1.1.1.2.	Herramientas de compilación y depuración	158
5.3.1.1.1.3.	Herramientas de implementación.....	161
5.3.1.1.1.4.	Documentación de productos	162
5.3.1.2.	Expression Studio 4.0 (Expression Blend).....	163
5.3.1.3.	Server Management Studio	163
5.3.1.3.1.	Características de SQL Server Management Studio	163
5.3.2.	DIAGRAMA DE LA BASES DE DATOS.....	165
5.3.3.	DIAGRAMAS DE CASOS DE USO.....	166
5.3.3.1.	Especificación de caso de uso : Iniciar sesión.....	166
5.3.3.3.	Especificación de Caso de Uso: Registrar usuario.....	169
5.3.3.5.	Especificación de Caso de Uso: Administrar publicación de cursos	173
5.3.3.6.	Especificación de Caso de Uso: Mostrar reporte “Resumen de Cursos”	175
5.3.3.7.	Especificación de Caso de Uso: Administrar contenido de cursos	177
5.3.3.8.	Especificación de Caso de Uso: Creación de pruebas	182
5.3.3.9.	Especificación de Caso de Uso: Exponer foros	185
5.3.3.10.	Especificación de Caso de Uso: Mostrar un resumen de cursos por alumno ..	187
5.3.3.11.	Especificación de Caso de Uso: Inscribirse a un curso	189
5.3.3.12.	Especificación de Caso de Uso: Mostrar contenido de un curso para sus alumnos	191
5.3.3.13.	Especificación de Caso de Uso: Aplicar pruebas.....	193
5.3.3.14.	Especificación de Caso de Uso: Guardar comentario de un foro.....	195
5.3.3.15.	Especificación de Caso de Uso: Mostrar reporte de pruebas aplicadas por el alumno	197
5.3.3.16.	Especificación de Caso de Uso: Mostrar cursos publicados.....	199
5.3.3.17.	Especificación de Caso de Uso: Mostrar el temario de un curso seleccionado	200
5.3.4.	DIAGRAMAS DE ACTIVIDADES.....	201

5.3.4.1.	Iniciar sesión (todos los usuarios).....	201
5.3.4.2.	Cambio de contraseña (todos los usuarios)	201
5.3.4.3.	Registrar usuarios.....	202
5.3.4.4.	Administrar información de personas(Inicio de sesión con rol “Administrador”) 202	
5.3.4.5.	Exponer cursos (Rol “Docente”, pantalla cursos).....	203
5.3.4.6.	Publicar cursos (Rol “Administrador”).....	203
5.3.4.7.	Inscribirse a un curso publicado (Rol “Alumno”).....	204
5.3.4.8.	Obtener contenido del curso (Rol “Alumno”).....	204
5.3.4.9.	Exponer foros (Rol Docente).....	205
5.3.4.10.	Comentar foro.....	205
5.3.4.11.	Crear pruebas	206
5.3.4.12.	Aplicar a una prueba.....	206
5.3.4.13.	Reporte del alumno.....	207
5.3.4.14.	Reporte docente	207
5.3.5.	DIAGRAMAS DE SECUENCIA	208
5.3.5.1.	Inicio de sesión.....	208
5.3.5.2.	Cursos y foros	209
5.4.	DOCUMENTACIÓN	210
5.4.1.	DOCUMENTACIÓN TÉCNICA	210
5.4.2.	DOCUMENTACIÓN DE USUARIO	210
5.5.	PRUEBAS FINALES	210
5.5.1.	PRUEBA DE CARGA Y STRESS	210
	CAPÍTULO VI.- CONCLUSIONES Y RECOMENDACIONES	211
6.1.	CONCLUSIONES.....	211
6.2.	RECOMENDACIONES	212
6.3.	POSIBLES TEMAS DE TESIS	213
	BIBLIOGRAFÍA.....	214
	LIBRO Y MANUALES	214
	ARTÍCULOS.....	214
	DIRECCIONES ELECTRÓNICAS.....	214
	DICCIONARIO DE DATOS.....	217

Índice de figuras

Figura 1. 1. Diseño de interfaces de usuario	5
Figura 1. 2. Arquitectura de Silverlight	10
Figura 2. 1. Dominio de las aplicaciones RIA	27
Figura 2. 2. Arquitecturas cliente-servidor	29
Figura 2. 3. Flujo descarga web tradicional	31
Figura 2. 4. Modelo de comunicación web tradicional vs aplicación RIA	32
Figura 2. 5. Ventajas del modelo RIA	34
Figura 2. 6. Adobe Flash	36
Figura 2. 7. Microsoft Silverlight	38
Figura 2. 8. Adobe Flex	40
Figura 2. 9. ICEFACES	43
Figura 3. 1. Clase Jerárquica de la interfaz de usuario en Silverlight 4	46
Figura 3. 2. Colocación de un rectángulo dentro de un control Canvas	56
Figura 3. 3. Colocación de varios rectángulos dentro del StackPanel	57
Figura 3. 4. Grid con tres columnas	58
Figura 3. 5. El DockPanel con anidación interior de los controles	59
Figura 3. 6. WrapPanel con orientación horizontal	60
Figura 3. 7. WrapPanel con orientación vertical	60
Figura 3. 8. TabControl con diferentes fichas seleccionadas	62
Figura 3. 9. Imagen contenedor Viewbox	64
Figura 3. 10. Colección de diferentes controles de botones	65
Figura 3. 11. Control AutoCompleteBox y sus partes	68
Figura 3. 12. Una visión conceptual de enlace de datos. La fuente posee los datos, la función objetivo (exposiciones, ediciones, y así sucesivamente) de los datos	68
Figura 3. 13. Una visión conceptual de la otrora enlace a un origen de datos. El valor es un principio leer desde la fuente y nunca se volvió a actualizar.	70
Figura 3. 14. Una visión conceptual de OneWay enlace a un origen de datos. El valor se actualiza cada vez que cambia la fuente, pero los cambios en el valor en el control de destino no lo hacen de nuevo a la fuente.	71

Figura 3. 15. Una visión conceptual de TwoWay enlace a un origen de datos. El control de destino refleja cambios en el origen y la fuente se actualiza con los cambios realizados en el destino.	72
Figura 3. 16. Uso de elementos de enlace para contar los caracteres a medida que escribe en un cuadro de texto	75
Figura 3. 17. Pantalla de error para el cuadro de texto Apellido, mensaje de error	87
Figura 3. 18. Incorporado en el control basado en excepciones tiene prioridad sobre el código.	90
Figura 3. 19. Miembros INotifyDataErrorInfo	90
Figura 3. 20 Detalle del modelo de vista	94
Figura 3. 21. Se muestra una versión simplificada de una aplicación de n niveles	96
Figura 4. 1. Ilustración de un trazado con el objeto Path	120
Figura 4. 2. Representacion del objeto Ellipse	121
Figura 4. 3. Rectángulo girado 45 grados alrededor del punto (0,0)	122
Figura 4. 4. Degradado lineal diagonal.....	125
Figura 4. 5 Puntos de degradado en un degradado radial	126
Figura 4. 6. VideoBrush y MediaElement	127
Figura 4. 7. Objeto EllipseGeometry utilizado para recortar una imagen	134
Figura 4. 8. Cuadro de diálogo de impresión para Windows	135
Figura 5. 1 Diseño de la arquitectura	147
Figura 5. 2 Entorno de desarrollo integrado (IDE).....	154
Figura 5. 3 Sistema de proyectos.....	155
Figura 5. 4 Diseñador de páginas Web, vista Diseño.....	156
Figura 5. 5 Diseñador de páginas Web, vista Código fuente	157
Figura 5. 6 Diseñador de páginas web y el Diseñador XAML	158
Figura 5. 7 Ventana de salida con información de compilación	159
Figura 5. 8 Formulario de Visual Basic en el modo de interrupción	160
Figura 5. 9 Ventanas de herramientas de depuración.....	160
Figura 5. 10 Asistente para publicación	161
Figura 5. 11 Editor del sistema de archivos	162

Figura 5. 12 Ayuda en la ventana del explorador	162
Figura 5. 13 Diagrama de Bases de Datos	165
Figura 5. 14 Especificación de caso de uso: Iniciar Sesión.....	166
Figura 5. 15 iniciar sesión	167
Figura 5. 16 Especificación de caso de uso: Cambiar clave de usuario.....	168
Figura 5. 17 Cambiar clave	168
Figura 5. 18 Especificación de caso de uso: Registrar usuario	169
Figura 5. 19 Registrar usuario	170
Figura 5. 20 Especificación de caso de uso: Administración de usuarios	171
Figura 5. 21 Administración de personas	172
Figura 5. 22 Especificación de caso de uso: Administrar publicación de cursos	173
Figura 5. 23 Administración de personas	173
Figura 5. 24 Especificación de caso de uso: Mostrar reporte “Resumen de Cursos”	175
Figura 5. 25 Mostrar reporte “Resumen de Cursos”	176
Figura 5. 26 Especificación de caso de uso: Administrar contenido de cursos ...	177
Figura 5. 27 Administrar contenido de cursos	179
Figura 5. 28 Especificación de caso de uso: Creación de pruebas	182
Figura 5. 29 Creación de pruebas	184
Figura 5. 30 Especificación de caso de uso: Exponer foros	185
Figura 5. 31 Exponer foros	186
Figura 5. 32 Especificación de caso de uso: Mostrar un resumen de cursos por alumno	187
Figura 5. 33 Mostrar un resumen de cursos por alumno	188
Figura 5. 34 Especificación de caso de uso: Inscribirse a un curso	189
Figura 5. 35 Inscribirse a un curso	190
Figura 5. 36 Especificación de caso de uso: Mostrar contenido de un curso para sus alumnos	191
Figura 5. 37 Mostrar contenido de un curso para sus alumnos	191
Figura 5. 38 Especificación de caso de uso: Aplicar pruebas	193
Figura 5. 39 Aplicar pruebas	194

Figura 5. 40 Especificación de caso de uso: Guardar comentario de un foro	195
Figura 5. 41 Guardar comentario de un foro	196
Figura 5. 42 Especificación de caso de uso: Mostrar reporte de pruebas aplicadas por el alumno.....	197
Figura 5. 43 Mostrar reporte de pruebas aplicadas por el alumno	197
Figura 5. 44 Especificación de caso de uso: Mostrar cursos publicados	199
Figura 5. 45 Mostrar cursos publicados	199
Figura 5. 46 Especificación de caso de uso: Mostrar el temario de un curso seleccionado	200
Figura 5. 47 Mostrar el temario de un curso seleccionado.....	200
Figura 5. 48 Diagrama de actividades: Iniciar sesión	201
Figura 5. 49 Diagrama de actividades: Cambio de contraseña	201
Figura 5. 50 Diagrama de actividades: Registrar usuarios.....	202
Figura 5. 51 Diagrama de actividades: Administrar información de personas.....	202
Figura 5. 52 Diagrama de actividades: Exponer cursos	203
Figura 5. 53 Diagrama de actividades: Publicar cursos	203
Figura 5. 54 Diagrama de actividades: Inscribirse a un curso publicado.....	204
Figura 5. 55 Diagrama de actividades: Obtener contenido del curso	204
Figura 5. 56 Diagrama de actividades: Exponer foros.....	205
Figura 5. 57 Diagrama de actividades: Comentar foro	205
Figura 5. 58 Diagrama de actividades: Crear pruebas	206
Figura 5. 59 Diagrama de actividades: Aplicar a una prueba.....	206
Figura 5. 60 Diagrama de actividades: Reporte del alumno.....	207
Figura 5. 61 Diagrama de actividades: Reporte docente	207
Figura 5. 62 Diagrama de secuencia: Inicio de sesión	208
Figura 5. 63 Diagrama de secuencia: Cursos y foros.....	209

Índice de tablas

Tabla 1. 1. Componentes de presentación básicos.....	12
Tabla 1. 2. NET Framework para Silverlight.....	14
Tabla 1. 3. Características de programación Silverlight	15
Tabla 1. 4. Modelo de objetos de documentos del explorador	22
Tabla 3. 1. Métodos de la clase System.Windows.DependencyObject.....	47
Tabla 3. 2. Métodos de la clase System.Windows.UIElement.....	50
Tabla 3. 3. Propiedades de la clase System.Windows.UIElement	51
Tabla 3. 4. Eventos de la clase System.Windows.UIElement	53
Tabla 3. 5. Propiedades de la clase System.Windows.Controls.Control	55
Tabla 3. 6. Propiedades de TabControl.....	61
Tabla 3. 7. Propiedades de la clase System.Windows.Controls.Primitives.ButtonBase.....	65
Tabla 3. 8. Propiedades de IDataErrorInfo	87
Tabla 4. 1. Clases para la captura de contenido de audio y de vídeo	112
Tabla 4. 2. Categorías de animación.....	114
Tabla 4. 3. Tipos de animación	115
Tabla 4. 4 Clases de animación de fotograma clave.....	118
Tabla 4. 5. Propiedades comunes de un objeto Geometry	128
Tabla 4. 6. Tipos de segmentos de la geometría de trazados.....	129
Tabla 4. 7. Clases de Transformación.....	130
Tabla 4. 8. Clase de transformaciones más complejas	131
Tabla 4. 9. Propiedades de transformación.....	132
Tabla 4. 10. Valores de Stretch	133
Tabla 5. 1 Registro de auditoría	152
Tabla 5. 2 Evento	153
Tabla 5. 3 Contenido tabla "Evento"	153
Tabla 5. 4 Persona	167
Tabla 5. 5 PerRol	167
Tabla 5. 6 PerRol	172
Tabla 5. 7 Curso.....	175

Tabla 5. 8 TemaCurso.....	177
Tabla 5. 9 ContenidoTema.....	178
Tabla 5. 10 Prueba.....	183
Tabla 5. 11 Pregunta.....	183
Tabla 5. 12 OpcionPregunta.....	184
Tabla 5. 13 Foro.....	186
Tabla 5. 14 Calificacion.....	187
Tabla 5. 15 Inscripcion.....	189
Tabla 5. 16 Comentario.....	195

Resumen

Microsoft Silverlight 4.0 es, una plataforma para la construcción de aplicaciones RIA (Rich Internet Applications), que tiene la característica de conjugar lo mejor del mundo Web con las ventajas de las aplicaciones de escritorio: experiencia de usuario de alta calidad, y distribución remota. Además, a partir de la versión 3.0 de la plataforma, Silverlight permite la ejecución de aplicaciones fuera del navegador.

Esta característica de ejecución tiene importantes implicaciones en la distribución, implantación e instalación en el cliente final, como es la posibilidad de asignar permisos especiales (siempre bajo el consentimiento del usuario de la aplicación), que termina con las limitaciones de la ejecución en sandbox, y permite acceder a ciertas áreas comunes del sistema local de ficheros y a otros recursos del sistema operativo, como la Impresora, el Portapapeles, etc.

Es una implementación multiplataforma de .NET Framework que se puede ejecutar en distintos exploradores para crear y proporcionar la nueva generación de experiencias multimedia y aplicaciones interactivas enriquecidas para la Web. Silverlight unifica las funciones del servidor, la Web y el escritorio, del código administrado y de los lenguajes dinámicos, de la programación declarativa y la tradicional, así como la eficacia de Windows Presentation Foundation (WPF).

Summary

Microsoft Silverlight 4.0 is a platform for building RIAs (Rich Internet Applications), which has the characteristic of combining the best of the Web world with the advantages of desktop applications, user experience quality, and remote deployment. Also, from version 3.0 of the platform, Silverlight enables applications to run outside the browser.

This feature has important performance implications for the distribution, implementation and installation on the client end, as is the ability to assign special permissions (always under the user's consent application), which ends with the limitations of running in sandbox and access to certain areas of the local file system and other operating system resources such as printer, the clipboard, and so on.

It is an implementation platform .NET Framework applications that can run on different browsers to create and deliver the next generation of media experiences and rich interactive applications for the Web. Silverlight unifies the functions of the server, the Web and the desktop, of managed code and dynamic languages, declarative programming and the traditional and the power of Windows Presentation Foundation (WPF).

CAPÍTULO I. INTRODUCCIÓN

1.1 INTERFAZ DE USUARIO

1.1.1 DEFINICIÓN

La interfaz es la conexión física o funcional entre dos aparatos o sistemas independientes. La creación de una página web es posible con la unión de diferentes factores necesarios para su funcionamiento:

- aquello que se ve y es visible para el usuario el diseño web y el contenido y
- aquello que no se ve pero que es igual o más importante.

La programación web, accesibilidad y la interfaz web. Todo junto forma una estructura virtual que tiene como función hacer llegar al usuario o cliente toda la información de manera adecuada y accesible.

1.1.2 TIPOS DE INTERFACES

Hay dos tipos de interfaces de usuario según su manera de interactuar:

- **Alfanuméricas** – intérpretes de pedidos
- **Gráficas de usuarios** – GUI, Graphic User Interface.

Estas últimas permiten comunicarse con el ordenador de una forma rápida, intuitiva y gráfica (ventanas, botones, etc.).

Según su construcción, pueden ser de **hardware** o de **software**:

- En el primer caso se trata de un conjunto de dispositivos que permiten la interacción hombre-máquina, de forma que permiten acceder y coger datos del ordenador.
- Las Interfaces de software son programas o parte de ellos que permiten comunicarse con el ordenador.

1.1.3 ELEMENTOS DE LA INTERFAZ DE USUARIO

Los elementos del diseño de la interface de usuario basado en la tecnología de la Web se los representa por los siguientes sectores:

➤ **La lengua**

El idioma es el medio de comunicación más importante con el usuario. Los contenidos lingüísticos para el programa de software están previstos por el programador. A través de la lengua, los contenidos son presentados al usuario y, a través de la lengua, se realizan muchas interacciones entre el programa y el usuario. Una parte importante de cada programa es la terminología. Ésta se compone de dos campos: del vocabulario técnico del ordenador y del vocabulario técnico del contenido de la aplicación.

Los textos y la información lingüística son una parte importante de cualquier software. Gracias a ellos, el usuario y el software se comunican. Pese a ello, no se les considera un campo independiente. Las inconsistencias que resultan de este procedimiento afectan a la calidad del interfaz de usuario

➤ **La representación de la información**

Otro punto importante es la manera como se presenta la información en la pantalla y la forma en que se organiza. Estas informaciones pueden ser textos, formularios, listas, tablas, fotos, diagramas e ilustraciones. Elementos como la escritura y los colores tienen una función de refuerzo para facilitar al usuario la orientación y la legibilidad de la información.

Para la organización de la interfaz de usuario es importante definir reglas cuando una información o una aplicación aparecen en una nueva ventana o cuando se recomiende un cambio de página.

➤ **La navegación y la interacción**

La conducción del usuario dentro del programa es una tarea fundamental. Se necesita una serie de elementos de navegación que el usuario tiene que conocer y comprender para trabajar con el programa. Estos elementos normalmente tienen aspectos lingüísticos y gráficos.

Un software basado en la tecnología de la Web funciona igual que un sitio Web, como un hipertexto, es decir, informaciones que están representadas en una red dentro de la cual el usuario puede navegar. En general, hay tres posibilidades de organización de la navegación:

- Hiperenlaces
- Menú desplegable de acciones
- Botones

Una particularidad frecuente de un software basado en la tecnología de la Web es la manera de interacción entre el cliente y el servidor. Como no sólo los datos son calculados en la parte del servidor, sino también la presentación de la pantalla, se generan desde allí retrasos temporales que no son infrecuentes.

Esto cuesta mucho tiempo de espera para el usuario que posiblemente se aumenta más por restricciones técnicas. Por eso es importante tener en cuenta que los cambios de página ocurran con la menor frecuencia posible. Evidentemente este hecho coincide con la exigencia de que no debe aparecer demasiada información en una página por razones de legibilidad. Encontrar una solución para este conflicto es otra tarea del concepto de navegación.

➤ **La tramitación de datos**

Las funciones estándares para la tramitación de datos como “nuevo”, “almacenar”, “tramitar”, “eliminar” y “buscar” son realizadas a través de formularios y botones. Se añade a éstas la función "mover", por ejemplo, cuando se intenta mover un documento a otro archivo. Aplicaciones tradicionales de desktop ofrecen al usuario un "drag and drop" (esto es, que un elemento es movido por medio del ratón a otro sitio). Esta función no se realiza tan fácilmente en un browser, así que hay que recurrir a las funciones “cortar” e “insertar”.

1.1.4 MODELOS DE UI¹ SEGÚN EL PUNTO DE VISTA

Existen tres puntos de vista distintos en una UI: el del usuario, el del programador y el del diseñador. Cada uno tiene un modelo mental propio de la interfaz, que contiene los conceptos y expectativas sobre ella, desarrollados a través de su experiencia.

Estos modelos deben estar claros por los participantes en el desarrollo de un producto, de forma que se consiga una interfaz atractiva y a la vez efectiva para el trabajo con el programa.

“Una interfaz no es simplemente una cara atractiva; esta puede impresionar a simple vista pero decepcionar a la larga. Aquello importante es que el programa se adapte bien al modelo del usuario, cosa que se puede comprobar utilizando el programa más allá de la primera impresión”².

➤ **Uso de prototipos**

Se pueden clasificar los principales tipos de prototipos, variando su grado de complejidad:

- **Prototipos Estáticos:** Son aquellos que no permiten la alteración de sus componentes, pero sirven para identificar y resolver problemas de diseño.
- **Prototipos Dinámicos:** Permiten la evaluación de un modelo del sistema sobre una estación de trabajo o una terminal. Estos prototipos involucran aspectos de diseño más detallados que los prototipos estáticos, incluyendo la validación del diseño del sistema en términos de requerimientos no funcionales.
- **Prototipos Robustos:** Deben ser relativamente completos a la simulación de las características dinámicas de la interfaz (presentación de mensajes de error, entrada y edición de datos, etc.). Esta categoría puede ser utilizada para validar los objetivos de diseño.

¹ UI: Interfaz de usuario

² Diseño de Interfaces de Usuario. <http://www.monografias.com/trabajos10/diusuar/diusuar.shtml>

El nivel de sofisticación del prototipo se debería incrementar a lo largo del proceso de diseño de interfaces de usuario. La información recogida durante las tareas de análisis del sistema y la especificación de los requisitos del usuario constituyen los datos claves para el proceso de prototipación.

➤ Puntos de Interfaz

Estos cinco puntos representan una parte pequeña pero importante del diseño de la interfaz de usuario y ayudan a que la usabilidad de la aplicación mejore enormemente.

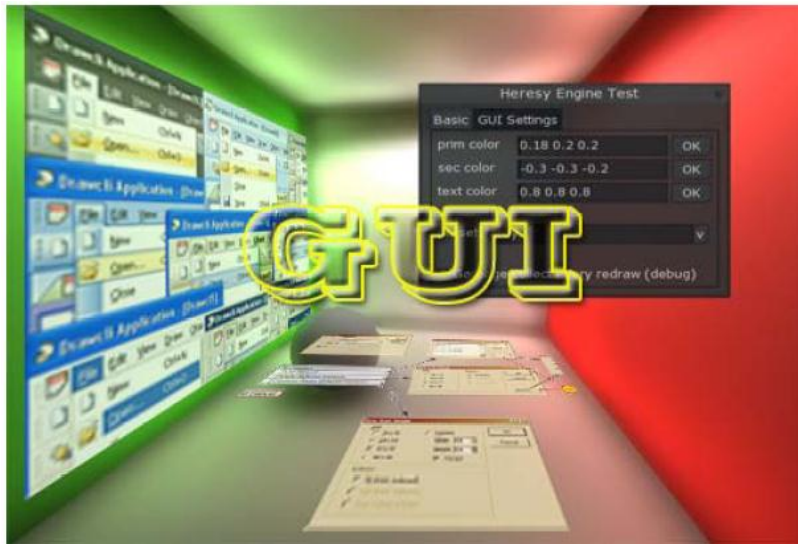


Figura 1. 1 Diseño de interfaces de usuario

- El usuario no está utilizando tu aplicación

La cuestión más básica a considerar en el diseño de interfaz de usuario es que el usuario no quiere utilizar tu aplicación. Quieren hacer su trabajo de la forma más sencilla y rápida posible, y la aplicación no es más que otra herramienta para ayudar a conseguirlo. Cuánto menos estorbe tu aplicación al usuario, mejor.

- **La Ley de Fitt**

“Esta es la ley más básica y más conocida entre las leyes del diseño de interfaces de usuario. Esta ley dice que cuando más grande y más próximo al puntero del ratón es un objeto, más sencillo es hacer click sobre él. Es de sentido común, pero, muchas veces es ignorado completamente en el diseño de interfaces”³.

El tiempo para llegar a un objetivo es en función de la distancia y el tamaño del objetivo. Es por esto, que es conveniente usar objetos grandes para las funciones importantes.

- **Interferencias Innecesarias**

Cuando un usuario está trabajando, su atención está centrada en el trabajo que está realizando. Cada vez que debe concentrarse en la aplicación, le saca tiempo el volver a centrarse en el trabajo. Por lo tanto, se debe minimizar la cantidad de distracción y de interferencias por parte de tu aplicación.

Cada aplicación tiene un elemento clave en la que centrarse: un editor de texto, es el texto; en un navegador web, es la página web, así que se debe de hacer de este elemento clave el centro de la interfaz. Es decir:

- No colocar barreras en el camino del usuario.
- Lanzar una ventana de diálogo solo si esta contiene información útil.
- Si es posible, utilizar indicadores de estado no modales.

- **Utilizar la potencia de la computadora**

En el diseño de interfaces para el usuario, las implicaciones que tiene esta idea son claras: cada vez que haya una decisión a tomar o trabajo a hacer. Los ordenadores son muy potentes así que se usa su potencia para ayudar al usuario. También se debe hacer que se pueda distinguir

³ Diseño de interfaces de usuario 5 pasos básicos
<http://www.alternext.com.bo/articulos/Diseno%20GUI.pdf>

fácilmente entre elementos similares y, sobre todo, recordar las opciones de la aplicación.

- **Haz que sea fácil distinguir los elementos y buscarlos**

Los elementos de la pantalla que hacen cosas distintas deberían ser fácilmente distinguibles unos de los otros y el elemento seleccionado debe ser sencillo de distinguir y de leer. No se debe abrumar al usuario con demasiadas opciones.

1.2 MICROSOFT SILVERLIGHT

1.2.1 INTRODUCCIÓN

Silverlight es un producto desarrollado por Microsoft, que permite crear las aplicaciones Web de última generación, las RIA (Rich Internet Application). Se trata de una combinación entre aplicaciones Web y aplicaciones de escritorio, ofreciendo una buena experiencia de usuario.

El mayor potencial de Silverlight está en el entorno de desarrollo, Silverlight incluye un subconjunto de la plataforma .NET de Microsoft y con esto toda la potencia que ello implica. Como entorno preferido para el desarrollo, bajo Windows, se tiene Visual Studio (desde la versión gratuita, la Express, hasta el entorno de gestión de ciclo de vida Team System). Así como Mac y Linux, MonoDevelop y Eclipse (instalando las recientes Eclipse Tools For Silverlight).

Una aplicación Silverlight está dividida en dos partes: el lenguaje de marcado (XAML⁴), donde está todo su potencial, y luego el lenguaje que se elija dentro de una larga lista: C#, Visual Basic, Python, Ruby, F#, JavaScript, etc.

Desde Silverlight 1.0 hasta el actual Silverlight 4, ha tenido muchas mejoras y nuevas funcionalidades. Dentro de las más importantes se menciona:

➤ **Multimedia:**

- Codecs: H.264/AAC/MP4
- Administración de derechos multimedia
- Motor de animaciones

➤ **Gráficos:**

- Aceleración GPU
- Efectos y Pixel Shaders 2.0
- Perspectiva3D

⁴ XAML: *eXtensible Application Markup Language*, Lenguaje Extensible de Formato para Aplicaciones

- Easing Functions
 - Pixel APIs
 - Codecs audio/video personalizables
 - Comunicación local entre aplicaciones Silverlight
- **Navegación:**
- Navegación a partir de URL
 - Historial del navegador
 - Optimización de Motores de Búsqueda (SEO)
 - Posibilidad de ejecutarse fuera del navegador
 - Pantalla completa
- **Datos:**
- Enlazado a datos
 - Datos de ejemplo durante el diseño
 - Gráficos

Como herramientas orientadas al diseño para Silverlight se cuenta con la suite Expression Studio que contiene: Expression Design, para diseño vectorial; Expression Blend, para diseño interactivo de interfaces de usuario; Expression Encoder, para la codificación de video. Además incluye otras herramientas como Expression Web para el diseño HTML⁵, CSS⁶ y AJAX⁷, con buenas herramientas para accesibilidad o compatibilidad de páginas entre navegadores (como SuperPreview que usa los renders de Firefox e Internet Explorer).

⁵ Html: (Lenguaje de Marcado de Hipertexto), es el lenguaje de marcado predominante para la elaboración de páginas web

⁶ CSS: Cascading Style Sheets - Hojas de Estilo en Cascada - que es un lenguaje que describe la presentación de los documentos estructurados en hojas de estilo para diferentes métodos de interpretación.

⁷ Ajax: JavaScript asíncrono y XML

Por otro lado tanto Expression Blend como Expression Design nos permite importar los diseños de Adobe Photoshop o Adobe Illustrator sin perder calidad.

1.2.2 ARQUITECTURA

Aunque la evolución de la arquitectura ha ido añadiendo elementos y bloques funcionales desde la primera versión, las divisiones fundamentales son las mismas. Por un lado, se tiene la estructura de presentación de la interfaz de usuario, basada en el lenguaje XAML. Por otro, las librerías soportadas y las extensiones de los lenguajes, más todos los añadidos que se han ido incorporando con la evolución de la plataforma. El esquema de la figura 1.1 de la arquitectura de Silverlight 4 muestra los distintos módulos y esa separación (más conceptual que real) en bloques operativos y/o funcionales.

Hay que notar que, respecto a las bibliotecas de MS-AJAX y JavaScript, de lo que se dispone es de la posibilidad de comunicarse con ellas; no se han creado bibliotecas especiales para este propósito.

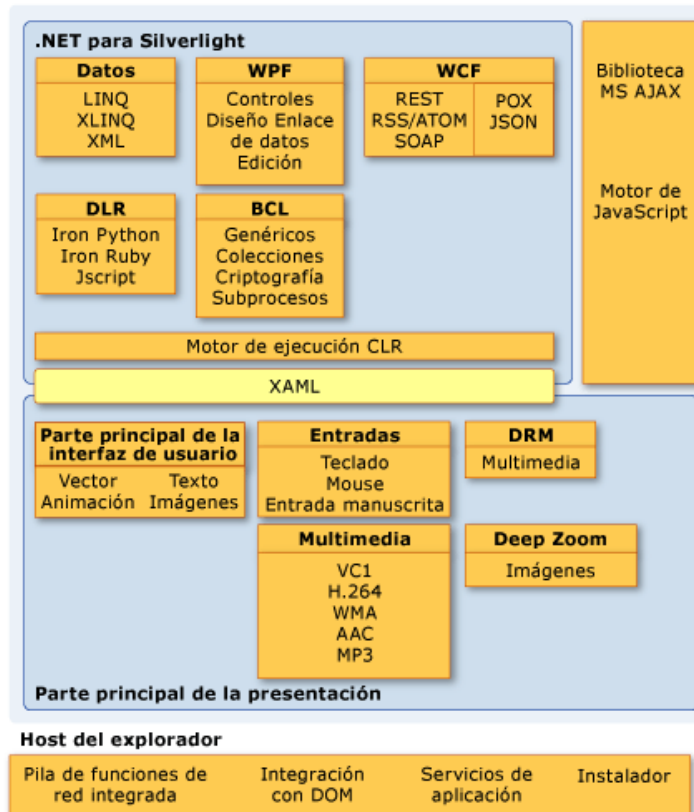


Figura 1. 2. Arquitectura de Silverlight

1.2.2.1 Apartados de la arquitectura

Se puede distinguir en esta arquitectura varias divisiones:

- .NET para Silverlight, que no es más que el conjunto de clases que integran el CLR⁸ del motor de ejecución, con sus principales apartados: Datos, WPF⁹ (XAML, en realidad), WCF¹⁰, el soporte de lenguajes dinámicos y las librerías de clase base (BCL¹¹), que están especialmente adaptadas para el componente Silverlight.
- Ese apartado se conecta gracias al lenguaje XAML como parte principal de la presentación. Esto incluye todos los elementos de entrada/salida: animaciones, tratamiento de texto, tratamiento de imágenes, sistemas diversos de entradas de teclado (incluyendo Ink, o entrada manuscrita y manipulaciones gestuales en Windows Phone 7), los elementos multimedia en sus diversos formatos, la gestión de derechos digitales (DRM¹²) y el motor DeepZoom especializado en tratamiento progresivo y secuencial de imágenes.
- Por otro lado se tiene los vínculos con otros elementos externos, como la biblioteca de AJAX, o el motor de JavaScript
- Y, finalmente, los servicios y funciones que son propias del navegador que alojará el complemento, y a los que se tiene acceso desde éste, como la pila de funciones de Red, la integración con el Modelo de Objetos de Documento (HTML- DOM, por distinguirlo del modelo de objetos que crea el complemento, llamado XML-DOM), los servicios de aplicación y los que dan soporte al proceso de instalación OOB (Out-Of-Browser)

⁸ CLR: (Common Language Runtime) es el motor en tiempo de ejecución del .NET Framework

⁹ WPF: Windows Presentation Foundation (WPF ó Base de Presentación de Windows),

¹⁰ WCF: Windows Communication Foundation (WCF) es un marco de trabajo para la creación de aplicaciones orientadas a servicios

¹¹ BCL: La Librería de Clase Base es una librería incluida en el .NET Framework formada por cientos de tipos de datos que permiten acceder a los servicios ofrecidos por el CLR

¹² DRM: La gestión de derechos digitales es un término genérico que se refiere a las tecnologías de control de acceso usada por editoriales y dueños de derechos de autor para limitar el uso de medios o dispositivos digitales.

1.2.2.2 Componentes de presentación básicos

Las características de presentación básicas de la plataforma Silverlight se describen en la tabla 1.1:

Característica	Descripción
Entrada	Administra datos de entrada procedentes de distintos dispositivos de hardware, como los de dibujo, el teclado y el mouse, y otros.
Representación de la interfaz de usuario	Representa gráficos vectoriales y de mapa de bits, animaciones y texto.
Multimedia	Permite la reproducción y administración de varios tipos de archivos de audio y vídeo, como los archivos .WMP y .MP3.
Deep Zoom	Permite acercar imágenes de alta resolución y realizar un movimiento panorámico alrededor de las mismas.
Controles	Admite controles extensibles que se pueden personalizar aplicando estilos y plantillas.
Diseño	Permite la colocación dinámica de los elementos de la interfaz de usuario.
Enlace de datos	Permite la vinculación de objetos de datos y elementos de la interfaz de usuario.
DRM	Permite la administración de derechos digitales de los recursos multimedia.
XAML	Proporciona un analizador para el marcado XAML.

Tabla 1. 1. Componentes de presentación básicos

1.2.2.3 .NET Framework para Silverlight.

En la tabla 1.2 se describe una lista parcial de las características de .NET Framework para Silverlight.

Característica	Descripción
Data	Admite las características de LINQ (Language Integrated Query) y de LINQ to XML, que facilitan el proceso de integrar datos procedentes de orígenes dispares y trabajar con ellos. También admite el uso de las clases de serialización y XML para administrar los datos.
Biblioteca de clases base	Un conjunto de bibliotecas de .NET Framework que proporcionan las funciones de programación esenciales, como la administración de cadenas, expresiones regulares, entrada y salida, reflexión, colecciones y globalización.
Windows Communication Foundation (WCF)	Proporciona características para simplificar el acceso a los servicios y datos remotos. Esto incluye un objeto de explorador, un objeto de solicitud y respuesta HTTP, compatibilidad con solicitudes HTTP entre dominios, compatibilidad con fuentes de distribución RSS/Atom así como compatibilidad con los servicios JSON, POX y SOAP.
CLR (Common Language Runtime)	Proporciona administración de memoria, recolección de elementos no utilizados, comprobación de seguridad de tipos y control de excepciones.
Controles de WPF (Windows Presentation Foundation)	Proporciona un conjunto enriquecido de controles, como son Button, Calendar, CheckBox, DataGrid, DatePicker, HyperlinkButton, ListBox, RadioButton y ScrollViewer.

DLR (Dynamic Language Runtime)	Admite la compilación y ejecución dinámicas de lenguajes de scripting como JavaScript y IronPython para programar aplicaciones basadas en Silverlight. Incluye un modelo conectable que aporta compatibilidad con otros lenguajes para su uso con Silverlight.
--------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Tabla 1. 2. NET Framework para Silverlight

.NET Framework para Silverlight es un subconjunto de la plataforma .NET Framework completa. Proporciona los fundamentos del desarrollo robusto y orientado a objetos para tipos de aplicaciones (como las de Internet) que tradicionalmente no contaban con este tipo de recursos.

Los desarrolladores pueden interactuar con la capa de .NET Framework para Silverlight escribiendo código administrado en C# y Visual Basic. Los desarrolladores de .NET Framework también pueden tener acceso a la capa de presentación creando sus aplicaciones en Visual Studio o Microsoft Expression Blend.

1.2.2.4 Características de programación adicionales de Silverlight

Silverlight proporciona varias características adicionales que ayudan a los programadores a crear aplicaciones enriquecidas e interactivas, incluidas las descritas en la tabla 1.3.

Característica	Descripción
Almacenamiento aislado	Proporciona acceso seguro del cliente de Silverlight al sistema de archivos del equipo local. Permite el almacenamiento local y el almacenamiento en caché de datos aislados para un usuario determinado.
Programación asincrónica	Un subproceso de trabajo de segundo plano lleva a cabo tareas de programación, mientras la aplicación queda libre para interactuar con el usuario.
Administración de	Proporciona un cuadro de diálogo Abrir archivo seguro, a

archivos	fin de facilitar el proceso de crear cargas de archivo seguras.
Interacción entre HTML y código administrado	Permite a los programadores de .NET Framework manipular directamente los elementos de la interfaz de usuario en el DOM HTML de una página web. Los programadores web también pueden utilizar JavaScript para efectuar llamadas directas al código administrado y tener acceso a los objetos, propiedades, eventos y métodos que admiten el uso de scripts.
Serialización	Proporciona soporte técnico para la serialización de los tipos CLR a JSON ¹³ y XML.
Empaquetar	Proporciona la clase Application y herramientas de compilación para crear paquetes .xap. El paquete .xap contiene la aplicación y el punto de entrada para que se ejecute el control del complemento Silverlight.
Bibliotecas XML	Las clases XmlWriter y XmlReader simplifican el trabajo con los datos XML de los servicios Web. La característica XLINQ permite a los desarrolladores consultar los datos XML directamente en los lenguajes de programación de .NET Framework.

Tabla 1. 3. Características de programación Silverlight

1.2.3 LENGUAJE XAML (Extensible Application Markup Language)

El lenguaje XAML es un lenguaje declarativo. Puede inicializar objetos y establecer propiedades de los objetos, con una estructura de lenguaje que muestra las relaciones jerárquicas entre los diversos objetos y con un sistema de tipos de respaldo que admite la extensión de los tipos. Puede crear elementos de

¹³ JOSN: es un formato de datos muy ligero basado en un subconjunto de la sintaxis de JavaScript: literales de matrices y objetos.

interfaz de usuario visibles en el marcado XAML declarativo. Se puede usar un archivo de código subyacente independiente para responder a los eventos y manipular los objetos declarados en XAML. El lenguaje XAML admite el intercambio de orígenes entre los diferentes roles y herramientas del proceso de desarrollo sin que se pierda información, como el intercambio de orígenes XAML entre Visual Studio y Microsoft Expression Blend.

1.2.3.1 Espacios de nombres XAML

En una definición amplia de la programación, un espacio de nombres determina cómo se interpretan los tokens de cadena que hacen referencia a las entidades de programación. Un espacio de nombres XAML es un concepto de espacio de nombres que tiene esta finalidad para el lenguaje XAML. En cuanto a su finalidad general y su aplicación en Silverlight, el lenguaje XAML se utiliza para declarar objetos, las propiedades de esos objetos y las relaciones entre los objetos y las propiedades que se expresan como jerarquías. Los objetos que se declaran son respaldados por bibliotecas de tipos. Esas bibliotecas podrían ser cualquiera de las siguientes:

- Bibliotecas básicas de Silverlight que están disponibles en cualquier runtime distribuido.
- Bibliotecas distribuidas que forman parte del SDK de Silverlight, que se redistribuye en el paquete (posiblemente con la opción de almacenamiento en caché de biblioteca de aplicaciones).
- Bibliotecas que representan la definición de un control de otro fabricante que la aplicación incorpora y el paquete de aplicación redistribuye.
- Una biblioteca propia creada en un proyecto de Silverlight, que contiene todo o parte del código de usuario de Silverlight de la aplicación.

El concepto de espacio de nombres XAML utiliza las declaraciones de espacio de nombres de estilo XML (xmlns) del marcado y asocia la información de los tipos de respaldo en forma de información de espacio de nombres CLR y de ensamblado a determinados espacios de nombres XAML. De este modo, un procesador XAML

que lee un archivo XAML puede diferenciar los tokens en el marcado y buscar los tipos y miembros de los ensamblados de respaldo asociados a ese espacio de nombres XAML cuando se crea la representación de un objeto en tiempo de ejecución.

Los tipos procedentes de bibliotecas distintas de las bibliotecas básicas de Silverlight requerirán que se declare y se asigne un espacio de nombres XAML con un prefijo para hacer referencia a los tipos de esas bibliotecas. La declaración de un espacio de nombres XAML distinto del predeterminado consta de tres elementos de información:

- Un prefijo, que define el token de marcado que se utiliza para hacer referencia a ese espacio de nombres XAML en el marcado XAML subsiguiente.
- El ensamblado que define los tipos de respaldo de los elementos en ese espacio de nombres XAML, a los que debe obtener acceso un procesador XAML para crear objetos basándose en las declaraciones XAML.
- Un espacio de nombres CLR de ese ensamblado.

1.2.3.2 Declara objetos

Un archivo XAML siempre tiene exactamente un elemento que actúa como su raíz y que declara un objeto que será la raíz conceptual de alguna estructura de programación, como una página, o el gráfico de objetos de toda la definición de una aplicación en tiempo de ejecución.

Por lo que se refiere a la sintaxis de XAML, hay tres maneras de declarar objetos en XAML:

- Directamente, usando la sintaxis de elementos de objeto: se usan etiquetas de apertura y de cierre para crear instancias de un objeto como un elemento con formato XML. Esta sintaxis se puede usar para declarar objetos raíz o crear objetos anidados que establecen valores de propiedad.
- Indirectamente, usando la sintaxis de atributos: se usa un valor de cadena insertado para declarar un objeto. Conceptualmente, esta sintaxis se puede

usar para crear instancias de cualquier objeto distinto de la raíz. Puede usar esta sintaxis para establecer el valor de una propiedad. Se trata de una operación indirecta para el procesador de XAML, puesto que debe haber algún procedimiento que sepa cómo crear un nuevo objeto conociendo la propiedad que se está estableciendo, las características del sistema de tipos de esa propiedad y el valor de cadena proporcionado. Normalmente, esto significa que el tipo o la propiedad en cuestión admite un convertidor de tipos que puede procesar la cadena proporcionada o que el analizador de XAML habilita una conversión nativa.

Esto no significa que siempre se pueda elegir la sintaxis para crear objetos en un vocabulario de XAML determinado. Algunos objetos del vocabulario pueden crearse únicamente usando la sintaxis de elementos de objeto. Un pequeño número de objetos solamente puede crearse estableciéndose inicialmente en un valor de propiedad. De hecho, los objetos que pueden crearse con la sintaxis de elementos de objeto o la sintaxis de atributos son comparativamente inusuales en Silverlight. Incluso cuando ambos tipos de sintaxis son posibles, uno de ellos tiende a ser el tipo dominante o el más apropiado para el escenario en cuestión.

1.2.3.3 Establecer propiedades

Puede establecer propiedades en objetos declarados mediante la sintaxis de elementos de objeto. Existen varias formas de establecer propiedades en XAML:

- Mediante la sintaxis de atributos.
- Mediante la sintaxis de elementos de propiedad.
- Mediante la sintaxis de elementos de contenido.
- Mediante una sintaxis de colección (que suele ser la sintaxis de colección implícita).

Al igual que ocurre con la declaración de objetos, esta lista de técnicas para establecer las propiedades de objetos en XAML no implica que una determinada propiedad pueda establecerse con cualquiera de estas técnicas. Algunas propiedades solamente admiten una de las técnicas.

Algunas propiedades pueden admitir combinaciones de estas técnicas; por ejemplo, una propiedad que admite la sintaxis de elementos de contenido también podría admitir una forma más detallada mediante la sintaxis de elementos de propiedad o una sintaxis de atributos alternativa. Esto depende de la propiedad y del tipo de objeto que use la propiedad. Las posibilidades de la sintaxis XAML se proporcionan en las secciones "Uso de XAML" de las páginas de referencia de cada una de las propiedades que pueden establecerse en XAML. También hay propiedades de objetos de Silverlight que no pueden establecerse en XAML por ningún medio y que pueden establecerse únicamente mediante código.

1.2.3.4 Extensiones de marcado

Las extensiones de marcado son un concepto del lenguaje XAML que se usa primordialmente en la implementación de XAML en Silverlight. En la sintaxis de atributos XAML, las llaves { y } indican el uso de una extensión de marcado. Este uso hace que el procesamiento de XAML se aparte del tratamiento general de valores de atributo como una cadena literal o un valor directamente convertible en cadena. En lugar de ello, un analizador suele llamar al código que respalda dicha extensión de marcado en concreto, lo que ayuda a construir un árbol de objetos a partir del marcado.

Silverlight admite las siguientes extensiones de marcado que se definen en el espacio de nombres XAML predeterminado de Silverlight y que su analizador de XAML entiende:

- **Binding:** admite enlaces de datos, lo que aplaza un valor de propiedad hasta que se interprete en un contexto de datos.
- **StaticResource:** admite referencias a los valores de recursos que se definen en un ResourceDictionary
- **TemplateBinding:** admite plantillas de controles en XAML que pueden interactuar con las propiedades de código del objeto con plantilla.
- **RelativeSource:** habilita una forma concreta de enlace de plantillas.

Silverlight también admite una extensión de marcado muy básica que se define en el espacio de nombres XAML del lenguaje XAML, `x:Null`

Las extensiones de marcado de Silverlight suelen devolver una instancia existente o aplazan un valor al tiempo de ejecución. Mediante el uso de extensiones de marcado, cada propiedad que puede establecerse en XAML también puede establecerse potencialmente en la sintaxis de atributos. Puede usar la sintaxis de atributos para proporcionar valores de referencia para una propiedad, aun cuando dicha propiedad no admita una sintaxis de atributos para la creación de instancias de objeto directas, o puede habilitar un comportamiento concreto que difiera del requisito general que exige que las propiedades de XAML se rellenen con tipos de valores o tipos de referencia creados Just-In-Time.

1.2.3.5 Convertidores de tipo

Un convertidor de tipos es la entidad conceptual que toma un valor de atributo de XAML y utiliza el valor de cadena para crear un valor de resultado que no es literalmente una cadena. Los convertidores de tipos están asociados a los tipos de respaldo de los objetos o, a veces, a sus propiedades específicas¹⁴. El acceso a un convertidor de tipos no requiere ningún uso especial del marcado XAML. Un procesador de XAML lee los tipos de respaldo como parte del comportamiento general de asignación de tipos que presenta un procesador de XAML, determina si esos tipos admiten un convertidor de tipos e invoca ese convertidor de tipos cuando sea necesario para obtener valores.

Los convertidores de tipos habilitan la sintaxis de atributos en los casos en los que, de otra manera, se requeriría la sintaxis de propiedades. Normalmente, el objetivo es la simplificación de los formatos de marcado. Algunos convertidores de tipos tienen un comportamiento de conversión relativamente simple. Por ejemplo, `NullableBoolConverter` permite especificar los tres posibles estados de un valor `Nullable<bool>` como las cadenas `true`, `false` y `{x:Null}`.

Un número reducido de convertidores de tipos admite un "minilenguaje" bastante complejo para el cual se podría necesitar una guía sintáctica o gramatical a fin de

¹⁴ Información general sobre Silverlight. [http://msdn.microsoft.com/es-es/library/cc189036\(v=vs.95\).aspx](http://msdn.microsoft.com/es-es/library/cc189036(v=vs.95).aspx)

comprenderlo perfectamente. Uno de los ejemplos es Sintaxis de las rutas de acceso de propiedad, que documenta la gramática para propiedades como Path que utilizan PropertyPathConverter.

Los convertidores de tipos de Silverlight sirven principalmente como clases de soporte para XAML y como clases base para cualquier persona que cree un convertidor de tipos a fin de admitir un tipo personalizado.

1.2.4 INTEGRACIÓN A BROWSER Y DESKTOP

1.2.4.1 Integración con el browser

La Interfaz de Programación de Aplicaciones (API) administrada y la API de JavaScript permiten utilizar JavaScript en la página HTML en que se hospeda para integrar Silverlight con el Modelo de objetos de documento HTML (DOM¹⁵).

En la API de JavaScript, debe implementar la mayoría de la funcionalidad de la aplicación utilizando JavaScript interpretado por el explorador, que puede interactuar con el DOM directamente. En la API administrada, el código de JavaScript y el código DOM interactúan con el complemento Silverlight y el modelo de aplicaciones utilizando el puente HTML.

Las aplicaciones basadas en Silverlight que utilizan el complemento Silverlight en un host del explorador exponen su funcionalidad a través de un DOM (Document Object Model) basado en explorador, así como un modelo de programación específico de Silverlight que incluye el concepto de árbol de objetos. Cuando se utiliza el modelo de programación de Silverlight, se podrá elegir entre tres variaciones de la API: JavaScript interpretado por el explorador, código administrado o lenguajes dinámicos interpretados por DLR (Dynamic Language Runtime). En este tema se describe la relación entre el DOM y cada posible variación de la API, así como la forma en que las distintas API y modelos de aplicaciones usan el marcado XAML.

¹⁵ DOM: Modelo de Objetos de Documento. Es una plataforma que proporciona un conjunto estándar de objetos a través de la cual se pueden crear documentos HTML y XML,

1.2.4.2 Modelo de objetos de documentos

El DOM (Document Object Model) es un concepto de programación independiente de la plataforma y del lenguaje. Un DOM proporciona una representación estructurada de un documento, como una página web, así como un modo definido de obtener acceso y manipular la estructura, el estilo y el contenido del documento. Los exploradores que pueden hospedar el motor en tiempo de ejecución de Silverlight como complemento implementan un DOM HTML que expone los elementos de la página HTML, incluido el complemento Silverlight.

El estándar relativo al DOM de World Wide Web Consortium (W3C) constituye la base del DOM implementado en los exploradores más utilizados. Sin embargo, la mayoría de los exploradores ofrecen otras extensiones además del estándar de W3C. Por consiguiente, los programadores deben ser conscientes de estas diferencias para crear contenido que sean compatibles con todas las plataformas.

Los DOM de explorador que se muestran en la tabla 1.4 pueden tener acceso a Silverlight.

DOM	Descripción
Gecko DOM (Mozilla, Firefox, versión 6 y posteriores de Netscape y otros exploradores basados en Mozilla)	Gecko es el componente de software que controla el análisis de HTML, el diseño de páginas, el modelo de objetos de documento y la representación de la aplicación completa.
DHTML DOM (Internet Explorer)	El DOM de HTML dinámico (DHTML) ofrece a los autores acceso directo y programable a los componentes individuales de sus documentos web, desde elementos individuales hasta contenedores.

Tabla 1. 4. Modelo de objetos de documentos del explorador

Después de crear el complemento Silverlight (normalmente con una etiqueta HTML object, se puede recuperar una referencia a la instancia del complemento

en el DOM HTML haciendo referencia a su identificador (ID). En el ejemplo de JavaScript siguiente se muestra cómo recuperar los identificadores (ID) del complemento Silverlight utilizando el método `document.getElementById`.

```
var plugin_1 = document.getElementById("SLPlugin_1");
```

1.2.4.3 Puente HTML: interacción entre código HTML y código administrado

En Silverlight, HTML Bridge es un conjunto integrado de tipos y métodos que permiten hacer lo siguiente:

- Exponer los tipos administrados completos a JavaScript para scripting.
- Exponer métodos individuales de tipos administrados a JavaScript para scripting.
- Pasar tipos administrados como parámetros a las funciones y objetos de JavaScript.
- Devolver tipos administrados desde JavaScript.
- Asignar tipos administrados como controladores de eventos, que son invocables desde JavaScript.
- Llamar a los controladores de eventos de JavaScript desde los tipos administrados.
- Controlar varios aspectos de seguridad de la aplicación basada en Silverlight.

Además, HTML Bridge proporciona contenedores administrados para los elementos del DOM (Document Object Model), tales como `window`, `document` y los elementos HTML estándar.

1.2.4.4 Silverlight fuera del navegador

Las aplicaciones en Silverlight ejecutan de manera predeterminada adentro de un navegador, pero no es el único modelo de ejecución que tienen ya que también pueden ser instaladas para ejecutar tal y como si fueran una aplicación de

escritorio tradicional. Gracias a esta característica se puede crear aplicaciones en Silverlight que tengan accesos directos en el equipo del usuario, tengan su propia ventana y además ejecuten de manera desconectada sin necesidad de abrir primero un navegador.

Para obtener una compatibilidad más completa sin conexión, que incluya acceso total al equipo local, puede ser más conveniente utilizar Windows Presentation Foundation (WPF). WPF proporciona un supraconjunto de funcionalidades de Silverlight y permite convertir las aplicaciones basadas en Silverlight en aplicaciones para Windows con todas las características, con mínimas modificaciones del código.

1.2.4.5 Modelos de ejecución e interoperabilidad en Silverlight 4

Los modelos de ejecución del complemento han variado, y pueden considerarse cinco grandes grupos, aparte del modelo clásico de ejecución dentro de una página Web:

- Utilizar Silverlight fuera del explorador.
 - En este caso, se permite la ejecución fuera del explorador en equipos Windows y Macintosh para las aplicaciones Silverlight alojadas en web. Requiere que los usuarios instalen la aplicación desde una página web que lo aloje.
- Utilizar Silverlight en un control de explorador (WebBrowser o equivalente).
 - Desde una aplicación creada con otra tecnología, se puede recurrir a mostrar una página que contenga un complemento Silverlight, siempre y cuando se disponga de un control Browser, que sea capaz de abrir e interpretar la página que aloja el complemento.
 - La comunicación entre el complemento Silverlight y el programa puede establecerse mediante el puente HTML: un mecanismo de interacción entre el modelo de objetos de documento (HTML/DOM) y Silverlight, que funciona en las dos direcciones.
 - Este tipo de alojamiento dispone de varias posibilidades dependiendo del contexto del programa a utilizar:

- Controles **Browser** de Windows Forms, WPF o Win32
 - Componentes MSHTML
 - WebKit de Macintosh OS X.
- Utilizar una interfaz de usuario de Silverlight en otro complemento (add-in) o aplicación.
- Se trata de una opción no demasiado conocida, basada en utilizar una interfaz no administrada llamada ISilverlightViewer, mediante la que se puede recuperar una representación gráfica (mapa de bits) de una interfaz de usuario Silverlight. Esto puede resultar interesante en el caso de que se necesite mostrar toda o parte de la interfaz de Silverlight dentro de otro complemento o una aplicación Win32 que no disponga o utilice un control WebBrowser.
- Utilizar Silverlight en una aplicación basada en Windows a través de COM.
- Dada la cantidad de aplicaciones COM todavía en funcionamiento y soporte, Silverlight suministra varias interfaces COM de forma que una aplicación de este tipo alojar un complemento Silverlight. Se trata de una opción más compleja de programar, pero de gran flexibilidad en ciertos casos puntuales.
- Utilizar Silverlight en otra aplicación a través de la interfaz del explorador.
- Finalmente, se puede utilizar el modelo de programación de add-ins (complementos) de un navegador. La aplicación que hace de host es la encargada en este caso de incrustar el complemento Silverlight igual que si se tratara de un explorador. Es particularmente útil para las aplicaciones basadas en Macintosh que no disponen de acceso a interfaces COM.

En caso de que la arquitectura sea tal que contenga varios complementos Silverlight dentro de una página Web, es posible sincronizar el comportamiento de todos o algunos de ellos a través de un sistema de mensajería entre complementos, que apareció en la versión 3.0.

CAPÍTULO II.- RIA (APLICACIONES DE INTERNET ENRIQUECIDAS)

2.1. INTRODUCCIÓN

Rich Internet Application (RIA) es una aplicación web que tiene muchas de las características de las aplicaciones de escritorio, por lo general entregados ya sea a través de un navegador de sitios específicos, a través de un navegador plug-in, o de forma independiente sandboxes o máquinas virtuales.

Adobe Flash, JavaFx y Silverlight de Microsoft son actualmente las tres plataformas más comunes, con tasas de penetración de alrededor del 99%, 80% y 54% respectivamente (en julio de 2010)¹⁶. A pesar de los nuevos estándares web han surgido, se siguen utilizando los principios detrás de RIAs.

El término “aplicación rica de Internet” se introdujo en un libro blanco de marzo de 2002 por Macromedia (ahora Adobe), aunque el concepto ha existido durante varios años antes bajo nombres tales como:

- Remote Scripting, por Microsoft, circa 1998
- X Internet, por Forrester Research en octubre de 2000
- Rich (web) clientes
- Aplicación web rica

Flash, Silverlight y JavaFx pretenden enriquecer las experiencias de usuario en parte debido a su menor dependencia de las comunicaciones red/servidor. Dado que el 80% del tiempo se dedica a descargar todos los componentes de la página, simplificando el diseño de una página, es también una manera de reducir el tiempo de respuesta. Otra forma de hacer frente a esto es reducir las peticiones HTTP mediante la combinación de todas las peticiones HTTP y hojas de estilo CSS.

Este enfoque permite que el sistema reduzca la cantidad y frecuencia del tráfico cliente-servidor.

¹⁶ <http://www.nohaylimites.com/?p=184>

2.2. APLICACIONES RIA

2.2.1. CARACTERÍSTICAS

Las aplicaciones RIA (Rich Internet Applications) nacen de la convergencia de dos paradigmas, las aplicaciones de escritorio y las aplicaciones web; gracias a la evolución de ambos cada día se hace más difícil distinguir entre uno y otro.

Las tecnologías comprendidas por la definición de RIA, varían en un rango amplio entre las aplicaciones tradicionales de escritorio y las simples páginas estáticas, tal como se muestra en la figura 2.1.

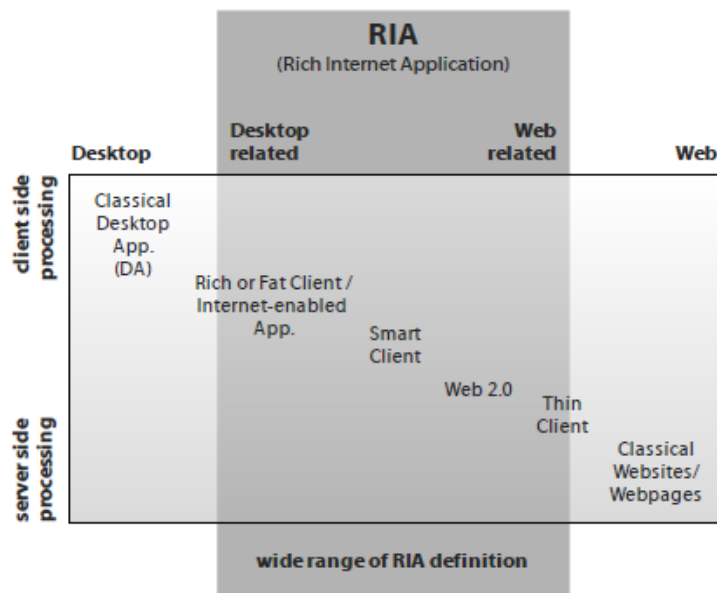


Figura 2. 1. Dominio de las aplicaciones RIA

Se puede notar, de esta manera, que es posible determinar que existen aplicaciones RIA que corren sobre un navegador de Internet y otras no. Las aplicaciones RIA basadas en un navegador son sitios web que mejoran la experiencia de los usuarios aumentando facilidades de personalización, tiempo de respuesta y riqueza multimedia, como ejemplos de estas están Facebook y Google Maps.

Las aplicaciones RIA que no necesitan de un navegador, gracias a su apariencia y gran cantidad de contenidos, hacen que el usuario las perciba como aplicaciones de escritorio, el principal ejemplo de este tipo de aplicaciones es el

reproductor de audio y video Apple iTunes que además incluye iTunes Store.

iTunes puede ser visto como una aplicación de escritorio por la necesidad de una instalación local, pero puede ser vista como una aplicación web que actúa como un sitio de comercio electrónico.

Aunque existen varias tecnologías y aplicaciones que pueden ser consideradas dentro de la categoría de aplicaciones RIA dependiendo del punto de vista, la característica común es que una aplicación RIA junta las bondades de las aplicaciones de escritorio con los beneficios de las aplicaciones web.

De las aplicaciones web se toman las siguientes características:

- Tiempo corto para la obtención de un producto final gracias a la estandarización de scripts y etiquetas
- Las aplicaciones que corren sobre un servidor no necesitan instalación de parches ni actualizaciones.
- Independencia de la plataforma desde la cual son accedidas
- Alta disponibilidad
- Interfaz simple y estandarizada disminuyendo la curva de aprendizaje.

De las aplicaciones de escritorio se toman las siguientes características:

- Gran contenido multimedia (audio, video).
- No existe la necesidad de recargar la aplicación, (recarga de página).
- Soporte online y off line.
- Mayor complejidad en las aplicaciones.
- Gran familiaridad con los usuarios.

Se considerará ahora los aspectos fundamentales que distinguen a las aplicaciones RIA:

1. Proporcionan un gran rendimiento, esto debido a que el procesamiento se puede hacer en el lado del cliente, en lugar que todos los procesos se

realicen en el lado del servidor. De igual manera mientras se usa la aplicación el usuario no experimenta ninguna clase de refresco de página.

Sin embargo la decisión de que procesos se realizan en la parte del cliente y que en la parte del servidor, dependen de varios aspectos, en la figura 2.2 se puede notar la separación de las capas de una aplicación, junto con la distribución de procesamiento.

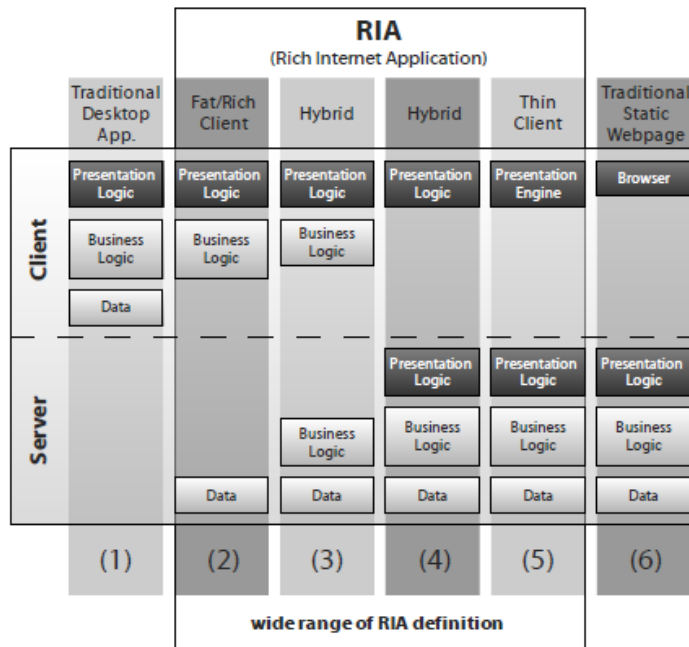


Figura 2. 2. Arquitecturas cliente-servidor

2. Priorizan la interacción con el usuario, por la gran riqueza de recursos proporcionan una experiencia más atractiva y entretenida.

A continuación se describen algunos ejemplos:

- Gracias a las herramientas gráficas para la visualización de datos, permite al nivel gerencial interactuar con datos complejos y realizar análisis más profundos.
 - Permite a los usuarios personalizar un producto en línea para satisfacer sus necesidades específicas.
3. Acoplan de manera natural varias tecnologías para conseguir la aplicación final que cumpla con las necesidades del usuario. Esto se hace sin olvidar la

consigna de independencia de plataforma de acceso del usuario. En este punto es necesario considerar que el uso de varias tecnologías versus el uso de una sola, determina que la complejidad y tiempo de desarrollo sufren un claro incremento.

2.2.2. ARQUITECTURA

Hay tres aspectos muy importantes que influyen en forma directa en la performance de una aplicación RIA.

- El diseño de la aplicación y el contexto o ambiente de uso
- El comportamiento y expectativas del usuario
- El comportamiento de la aplicación cuando se la utiliza

Desde que el usuario inicia una petición a una dirección URL hasta que la respuesta es enviada al usuario, suceden una serie de pasos los cuales se explicarán en los siguientes ítems.

2.2.2.1. Interacción entre el cliente y el servidor

El usuario hace un clic en un link del browser envía un request a un Server. El servidor responde a los request de los clientes, y cuando la cantidad suficiente de los contenidos requeridos llegan al cliente (a la cache del browser), el browser los muestra y el usuario los puede ver. Para que el usuario pueda ver la respuesta deberá esperar el tiempo necesario que tarde el proceso completo en que la respuesta llegue al browser.

La fecha negra del la figura 2.3 muestra el flujo de descarga de una página Web tradicional.

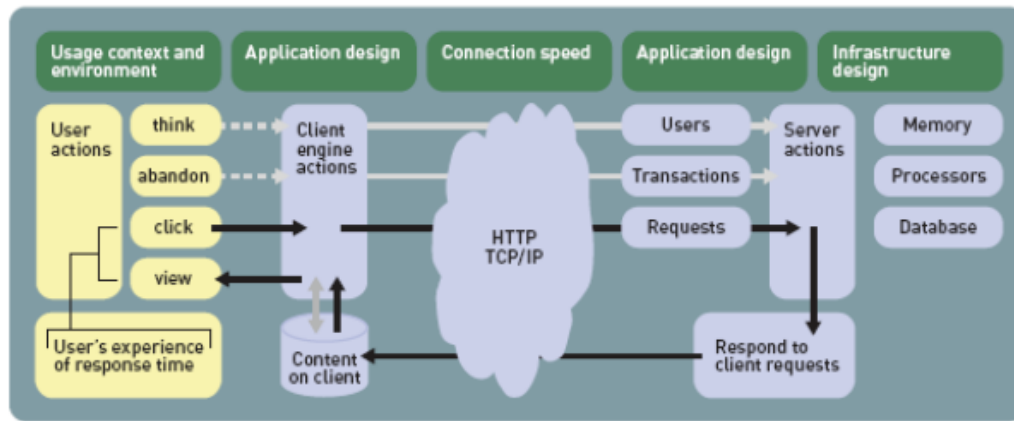


Figura 2. 3. Flujo descarga web tradicional

Una página Web tradicional generalmente está compuesta por hojas de estilo, archivos de scripts, imágenes, objetos flash entre otros, todos estos componentes son descargados, individualmente, varias veces ante los sucesivos request ya que la mayoría de las veces requiere varios intercambios de información entre el servidor y el cliente.

2.2.2.2. Comportamiento del servidor

Los servidores deben dar respuesta a muchos usuarios concurrentemente, no interesa cuan poderoso es el servidor, cada usuario que hace un request al servidor consume una pequeña parte de recursos del mismo, entre los cuales se destaca:

- Memoria
- ciclos de procesador
- recursos de motores de base de datos.

Los servidores Web pueden responder rápidamente a requerimientos de información de usuarios concurrentes, creando caches de browser haciendo mucho más eficiente las respuestas del mismo.

Pero una acción de usuario que involucra cambios en los datos tales como agregar un producto al carrito de compras, consume más recursos de servidor.

El número de transacciones concurrentes, interacciones que actualizan los datos personales de un cliente, juegan un rol crítico en la performance del servidor.

Las flechas grises del diagrama y los recuadros de usuarios y transacciones indican que la performance del servidor es altamente influenciada por estos factores concurrentes.

Los servidores típicamente funcionan de manera coherente hasta un cierto nivel de concurrencia, pero más allá de ese nivel (punto de inflexión), la performance de las transacciones se ven degradadas, transformándose en un cuello de botella.

Como resultado se tiene que con pequeños cambios (refactorings) se puede mejorar los tiempos de respuesta significativamente transformado en un mejor visión del usuario para con la aplicación.

El flujo del envío y recepción de datos de una aplicación rica es marcado en la figura anterior con líneas punteadas.

En la figura 2.4 se observa las diferencias de carga, performance y consumo de recursos entre una aplicación rica y una aplicación convencional

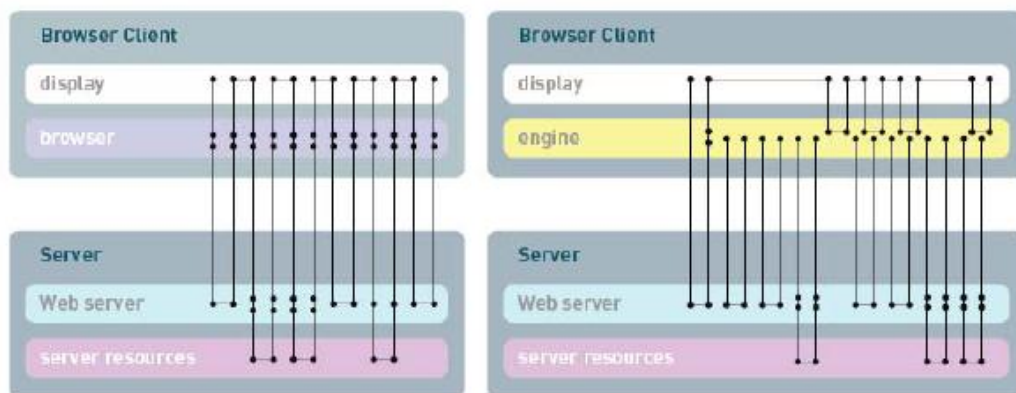


Figura 2. 4. Modelo de comunicación web tradicional vs aplicación RIA

Hay un factor importantísimo que se debe tener en cuenta en las aplicaciones ricas, cualquiera de los frameworks que se utilicen para desarrollar implícitamente proporciona un elemento más a las capas de la aplicación, esa capa también puede ser llamada “Motor Cliente”, esta capa cliente se puede ver en la figura 2.4, en el recuadro amarillo.

2.2.2.3. El “Motor” Cliente

Hoy en día hay varias implementaciones diferentes, todas las RIAs agregan una capa lógica intermedia, el “motor cliente de pedidos”. Esta capa es agregada entre el usuario y el servidor Web. Este “motor” maneja que contenidos a mostrar y los request hacia el servidor.

Este “motor” cliente puede seguir manejando las comunicaciones síncronas como las aplicaciones Web tradicionales, pero también permite al usuario interactuar con la aplicación asincrónicamente (independientemente de forma en que se comunique con el servidor).

La figura 2.4 muestran como es un patrón de comunicación asíncrona usuario-servidor en una aplicación RIA, y como difiere de una comunicación tradicional sincronía de una aplicación Web.

“Dependiendo de la plataforma de desarrollo la implementación difiere, todas las aplicaciones RIA agregan una capa intermedia o lógica, este es el motor del cliente encargado de gestionar los requerimientos de forma background, esta capa es ubicada lógicamente entre el usuario y el servidor Web”¹⁷.

Este “Motor” se descarga en el cliente al inicio de la sesión y maneja los cambios que se producen sobre la pantalla y se los comunica al servidor. Este motor solo produce cambios sobre los elementos necesarios sobre la pantalla, evitando las recargas completas de páginas.

¹⁷ Aplicaciones ricas en Internet (RIA). Un enfoque de refactorización” **Autor:** Eliseo Palacios

2.2.3. VENTAJAS

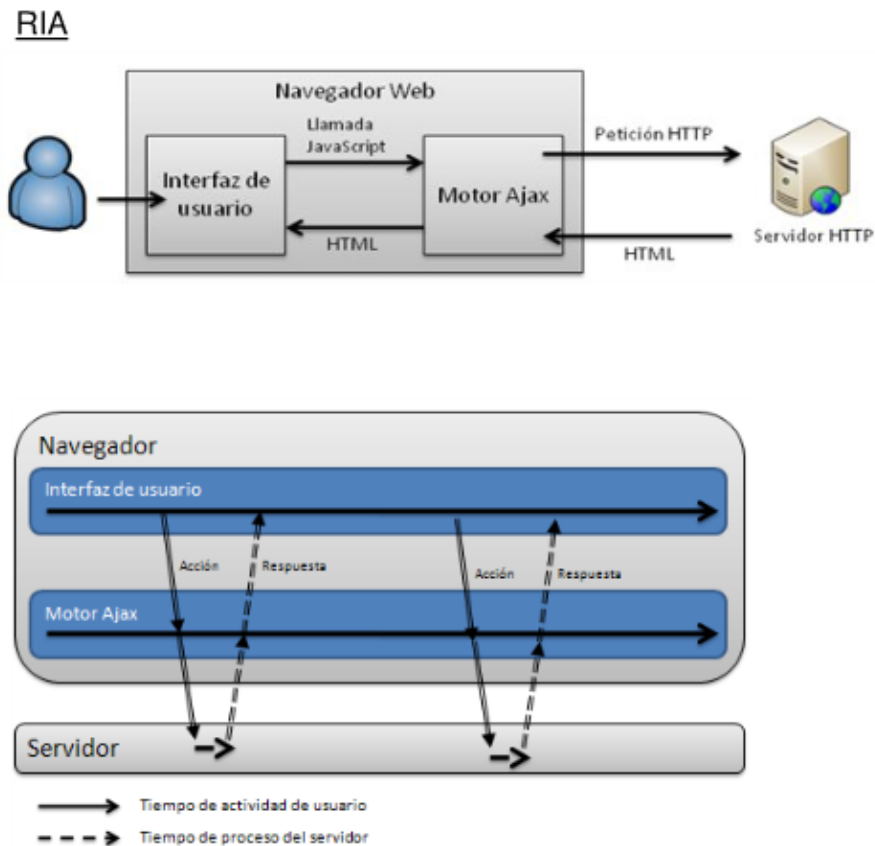


Figura 2. 5. Ventajas del modelo RIA

Ventajas respecto a aplicaciones de escritorio

- Único punto de instalación y mantenimiento en el servidor web
- Los clientes no requieren de ninguna instalación
- Clientes siempre actualizados a la última versión, disponible en el servidor web.
- Multiplataforma, accesible desde Windows, Linux, Mac OS, etc.

Ventajas respecto a aplicaciones web

- Experiencia de uso similar a aplicaciones de escritorio
- Desaparece problema de incompatibilidad entre navegadores.
- Reducción de carga de servidor al estar la capa de presentación en el cliente.
- Potencia en streaming de video

Diferencia entre un desarrollo tradicional y uno RIA

Para desarrolladores de aplicaciones de escritorio

- Adaptarse a trabajar con las restricciones de seguridad del navegador (no podrá guardar información en disco, acceder al registro o hardware local)
- Librerías más limitadas que en aplicaciones de escritorio

Para desarrolladores de aplicaciones web

- Adaptarse a la arquitectura orientada a servicios (SOA¹⁸), ya que la capa de presentación queda en el cliente, mientras que la lógica de negocio está en el servidor.

2.1.4 DESVENTAJAS

Tipología: no todos los tipos de aplicaciones de escritorio se pueden implementar usando RIA's. Por ejemplo programas de diseño, fotografía, videojuegos, etc. Para poder hacer un 3Dstudio online se necesitaría una conexión de banda muy ancha y un servidor con una potencia bastante alta.

Dependencia de la red: absoluta y completa. Si un día que se ejecutar la aplicación se ha cortado la línea no se podrá siquiera acceder a la pantalla de bienvenida.

Contenidos Multimedia: pese a que últimamente se están mejorando a pasos agigantados, los anchos de banda y los materiales utilizados en los cableados hacen que acceder a contenidos de audio o video pueda ser un suplicio, por mucho streaming y técnicas de compresión que se usen.

Luego de haber analizado las desventajas de las RIA también se consideran los siguientes inconvenientes

- Necesidad de instalar plug-ins
- Necesidad de disponer mayor potencia en los dispositivos cliente

¹⁸ SOA: Arquitectura Orientada a Servicios, es un concepto de arquitectura de software que define la utilización de servicios para dar soporte a los requisitos del negocio.

- Riesgo de sobrecargar la interfaz de usuario por querer explotar las posibilidades de RIA
- Problemas relativos a la accesibilidad de la aplicación

2.3. TECNOLOGÍAS DE DESARROLLO

2.3.1. ADOBE FLASH



Figura 2. 6 Adobe Flash

Adobe Flash es la herramienta de desarrollo que mezcla gráficos vectoriales, bitmaps, sonido, animaciones que nos permite crear aplicaciones interactivas y poner las mismas en la web, como si fuese una película.

Hoy en día Flash es usado en la industria gráfica cuya animación es destinada a medios de venta, comunicación y desarrollo de interfaces interactivas.

Utiliza .FLA que es el archivo nativo por defecto del programa y es necesario para poder editar los contenidos de la película, en la que quedarán almacenados todos los archivos que lo compongan, (salvo las tipografías), este archivo no puede ser reproducido sin el software flash instalado.

.SWF son archivos que exporta el programa y son autocontenidos, incluyendo en sí mismos todos los elementos que fueron importados, incluyendo imágenes, audios, videos y tipografías. Estos archivos .SWF se pueden incrustar dentro del HTML y ser subidos a un servidor para ser visualizados en la web o aplicados como proyecto autónomo.

El lenguaje nativo que se utiliza para programar en Adobe Flash es Actionscript.

El proceso de animación en Adobe Flash se denomina interpolación, que equivale a una transición entre al menos dos puntos, creando modificaciones ya sea de posición, tamaño, color, etc entre un punto y otro.

Desde Adobe Flash existen tres tipos de interpolación:

- **de Forma:** modifica vectores
- **de Movimiento:** permite animar por medio del asistente Motion Presets
- **clásica:** permite realizar transformaciones sobre el tamaño, posición, color de un objeto en escena.

Todos los procesos de interpolación se realizan en la línea de tiempo

Ventajas

- Flash está basado en imágenes vectoriales lo que permite que al ajustar sus tamaños éstas no se pixélen o pierdan calidad.
- Una vez que se conozca el software y con la práctica (mucha práctica) se vera que es relativamente sencillo y rápido crear webs.
- Se puede lograr presentaciones excelentes, de altísima calidad, que "cautiven" a nuestros visitantes o que llamen mucho la atención.

Desventajas:

- Puede haber problemas con algunos navegadores ya que si no se tiene instalado el plugin nuestra web "desaparece".
- El contenido de Flash NO ES INDEXADO por los buscadores por lo que difícilmente será fácil encontrar la web en Google, Yahoo, Ask, etc.
- Si la animación no está bien programada puede resultar pesada cargarla.
- Mucha animación o sonido es molesto. Caso típico son aquellas que al cargarlas reproducen un MP3 y no tienen botón desde dónde silenciarlo
- Producir cambios en páginas realizadas con Flash puede significar largo tiempo de espera.

- Para hacer una web flash hay que estudiar Actionscript, un lenguaje de programación propio del sistema.
- Adobe Flash es un programa de pago y si bien los plugin o reproductores de este tipo de archivos son gratuitos, para crearlos se necesita comprar el software.

2.3.2. MICROSOFT SILVERLIGHT



Figura 2. 7 Microsoft Silverlight

Microsoft Silverlight es un complemento para navegadores de Internet basado en la plataforma Windows que agrega nuevas funciones multimedia como la reproducción de vídeos, gráficos vectoriales, animaciones y de entorno de desarrollo; en forma similar a lo que hace Adobe Flash.

Se define como "una tecnología que permite crear aplicaciones de vanguardia", destacando entre sus características más sobresalientes:

- XAML. El subconjunto de WPF que define los elementos en el navegador para crear la interfaz de usuario. Esto supone crear gráficos, animaciones y elementos multimedia, y otras características de cliente enriquecido, permitiendo ir más allá de lo que HTML permite.
- Extensiones de JavaScript. Las extensiones al lenguaje de scripting de explorador que permiten controlar la interfaz de usuario, incluida la posibilidad de trabajar con elementos XAML. La comunicación entre los dos mundos (HTML/DOM y XAML) puede, además, realizarse en los dos sentidos.

- Compatibilidad con otros exploradores y plataformas. Su ejecución es idéntica en todos los exploradores conocidos (y en sus plataformas). Esto incluye varias versiones de Windows y Mac, y gracias a la plataforma MoonLight, Linux y FreeBSD.
- Integración con las aplicaciones existentes. Se integra perfectamente con el código JavaScript y ASP.NET AJAX existente, de modo que complementa la funcionalidad ya creada. De esta forma, Silverlight puede empotrarse como una isla dentro de una página o cubrir toda la interfaz de usuario.
- Modelo de programación similar y compatible con .NET Framework. Basado en la Capa de Adaptación de Plataforma (PAL), permite su ejecución en paralelo junto a otras versiones de .NET. También se pueden utilizar lenguajes dinámicos, como IronPython o IronRuby y enlazar con código de un lenguaje funcional como F#.
- Amplio conjunto de herramientas para desarrollo. Tanto Visual Studio .NET(2008/2010) como Expression Blend, permiten desarrollar para esta plataforma, si bien el soporte de la versión 4 de Silverlight se alinea con el mismo número de versión en Blend.
- Compatibilidad de red. Silverlight incluye compatibilidad con HTTP sobre TCP. Se puede conectar a los servicios WCF, SOAP, REST o ASP.NET AJAX y manejar formatos como Texto plano, XML, JSON o RSS.
- Soporte LINQ. Language Integrated Query, el lenguaje integrado de consultas que permite analizar y manejar orígenes de datos diversos con una sintaxis sencilla, intuitiva y muy potente.

Se mejora el rendimiento de la tecnología DeepZoom, que permite a los usuarios profundizar en el detalle de fotografías o collages de fotografías, permitiendo una transición suave entre ellas, para obtener la sensación de profundidad.

Las imágenes se pueden escalar desde resoluciones de 2 ó 3 megapíxeles hasta un gigapíxel, pero el usuario no tiene que esperar a la descarga completa, gracias a la tecnología subyacente que permite la descarga concreta de las zonas necesarias. Internamente, utiliza el formato de fichero XML.

Se denomina una potente API de Media Stream Source (o streaming adaptativo) que permite la descarga dinámica adaptable de contenidos multimedia. Esto permite a la aplicación que reproduce el contenido multimedia seleccionar la velocidad de descarga basándose en el ancho de banda del cliente y su potencia de CPU. Se implementa mediante una instalación en el servidor donde se aloja el contenido multimedia.

Inconvenientes

- La creación de gráficos vectoriales y animaciones debe realizarse con Expression Design, herramienta con licencia.
- No soportado en todos los navegadores. Soporte para Internet Explorer, Safari y Firefox

2.3.3. ADOBE FLEX



Figura 2. 8 Adobe Flex

Adobe Flex es un framework compuesto por un conjunto de tecnologías encaminadas al desarrollo de aplicaciones RIA. Estas tecnologías son Adobe Flash, ActionScript y AIR (Adobe Integrated Runtime)

El release inicial se dio en Marzo de 2004 de la mano de Macromedia, en este se incluía el SDK (Software Development Kit), el IDE (Integrated Development Environment) y una aplicación para la integración con J2EE a la cual denominaron Flex Data Services. Macromedia mantuvo el producto en el mercado hasta la versión 1.5 con el nombre de Macromedia Flex Server. Desde entonces Flash Player fue el elegido para poder visualizar las aplicaciones en el navegador.

Cuando Adobe compró Macromedia continuó con el proyecto, siendo el primer producto de Macromedia que fue renombrado por Adobe. En la versión 2 del nuevo Adobe Flex se cambió de manera considerable el sistema de licenciamiento así como el IDE para el desarrollo, Flex Builder, el cual se basó en la plataforma de código abierto Eclipse.

El cambio más significativo para la versión 2, fue la inclusión del renovado ActionScript, conocido ahora como ActionScript3, como lenguaje de programación, se mantuvo sin embargo la integración con J2EE mediante Flex Data Services.

El último release en el mercado es la versión Adobe Flex 3, la cual incluye un IDE desarrollado completamente sobre Eclipse y varios cambios importantes, el primero de estos fue la liberación del código del SDK mediante la licencia pública de Mozilla. El segundo fue el reemplazo de Flex Data Service por LiveCycle Data Services. El cambio final es la integración con AIR lo cual permite realizar aplicaciones RIA de manera tal que estas se ejecuten de manera directa sobre el sistema operativo sin la necesidad de un navegador. De esta manera se extendió el campo de acción que estaba limitado al uso de un navegador con Adobe Flash Player.

El servidor Flex también actúa como un gateway permitiendo al cliente comunicarse con servicios Web XML y objetos remotos (tales como Coldfusion CFCs, clases Java, Php, .NET, ASP y cualquiera que soporte el formato de mensajes de acciones).

Por medio del protocolo RTMP, Real Time Messaging Protocol, los clientes pueden establecer una conexión permanente con el servidor que permite una comunicación en tiempo real y la realización de video-chats, juegos multiplayer, etc.

Debido a su madurez y aceptación en el mercado se han creado gran cantidad de frameworks y patrones de diseño entre los cuales se destacan:

- **BlazeDS**: es un proyecto gratuito de código abierto que proporciona Flex Remoting y Messaging a todos los desarrolladores. Flex Remoting

proporciona un formato de transporte de datos binarios y serializados llamado AMF (ActionScript Message Format) para ofrecer un medio rápido y eficaz de transportar datos a las RIA que acelere el rendimiento de las aplicaciones.

- **Bulk Loader:** posee una estructura dinámica y consistente que permite gestionar un pool de conexiones, lanzar eventos, cargar todo tipo de contenidos, establecer prioridades, recopilar información en tiempo real como latencia, velocidad de descarga, ratio, porcentaje particular y de grupo entre otras cosas.
- **Cairngorm:** Cairngorm es un framework estructural para el desarrollo de RIAs que fomenta el uso de determinados patrones y potencia la escalabilidad y el crecimiento de las aplicaciones.
- **Cairngorm:** fue creado por Adobe y distribuido bajo licencia de código abierto.
- **Papervision3D:** es un completo motor de 3d OpenSource para Actionscript 3, permite crear escenas, agregar cámaras, cargar objetos 3D, materiales, luces, sombras, etc. El motor permite además realizar animaciones e interactuar con el escenario 3D

Beneficios

- Desarrollo ágil de aplicaciones RIA cliente-servidor.
- Potencia gráfica 2D y 3D.
- Soporte Multimedia completo, audio, video, streaming.
- Frameworks de desarrollo.
- Soporte multiplataforma, para todos los navegadores Web.

Inconvenientes

- Algunos gráficos vectoriales y animaciones personalizadas requieren Adobe Flash.

2.3.4. ICEFACES



Figura 2. 9 ICEFACES

Es un framework de código abierto para construir aplicaciones web con AJAX tipo RIA (Rich Internet Application).

Desarrollada por IceSoft. Librería de componentes ricos para JSF con un framework avanzado para la integración sencilla de las funcionalidades AJAX dentro del desarrollo de aplicaciones de negocio. Características similares a RichFaces.

Permite al programador incluir una serie de Ajax-tags en sus JSP o xhtml de tal manera que el código Ajax es generado por el propio framework automáticamente.

ICEFaces aísla completamente al desarrollador de AJAX. No hacen falta etiquetas especiales: se ponen los controles en la pantalla e ICEFaces se encarga de enviar sólo la información necesaria entre cliente y servidor. Es decir, ya no se envían los formularios a la antigua usanza, en un POST de HTTP, sino que sólo se envían los cambios que ha hecho el usuario del cliente al servidor, y los cambios en la pantalla del servidor al cliente.

Además, con la inclusión de la librería Scriptaculous en ICEFaces, se dispone de arrastrar+soltar y de efectos (fundidos, parpadeos, apariciones, ...) para los controles.

Las aplicaciones desarrolladas en ICEfaces no necesitan plugins de navegador o applets para ser vistas.

Estas aplicaciones están basadas en JavaServer Faces (JSF), así que permite el desarrollo de aplicaciones Java EE con la posibilidad de utilizar de forma fácil desarrollos basados en JavaScript.

ICEFaces fue una de las más acogidas ya que aísla completamente al desarrollador de AJAX. No hacen falta etiquetas especiales: se ponen los controles en la pantalla e ICEFaces se encarga de enviar entre cliente y servidor sólo la información necesaria.

Ventajas e inconvenientes con otros productos similares

Las ventajas del uso de ICEfaces para enriquecer las aplicaciones web de escritorio son numerosas. Debido a la cantidad de frameworks existentes, se procede a mostrar diferenciadoras respecto a otros frameworks:

- Crea una experiencia de usuario superior además de utilizar las ventajas de aplicaciones Java EE. Esto se consigue gracias a los componentes que vienen incluidos dentro de la distribución de ICEfaces.
- ICEfaces es un framework basado en Ajax bajo licencia de código abierto.
- Basado en estándares: ICEfaces es una solución basada en Java, así que los desarrolladores pueden continuar trabajando de la misma forma que lo hacen. Hay multitud de plugins desarrollados para que ICEfaces sea integrado con multitud de IDEs Java.
- El Ajax es transparente: ICEfaces aporta a los programadores un desarrollo con mínimo esfuerzo en la sección JSF.
- Compatibilidad: ICEfaces soporta todos los servidores de aplicaciones, aporta plugins para los distintos IDEs y efectos javascript de librerías de cualquier empresa que haya desarrollado Ajax del mercado.
- Seguridad: ICEfaces es una de las soluciones Ajax más seguras del mercado. Es compatible con SSL, previene los scripts de cross-site, inyección de código malicioso.
- Escalabilidad y clustering: El servidor asíncrono HTTP (AHS) aporta una alta escalabilidad para aplicaciones ICEfaces y pueden ser utilizadas por un gran número de usuarios concurrentes.

- Carga de páginas incremental con edición de secciones y sin recargas de página completas.
- Se preserva el contexto del usuario durante la actualización de la página, incluyendo posición del foco y scroll.
- En aplicaciones de tiempo real, las recargas de páginas son asíncronas.

CAPÍTULO III.- ESTRUCTURANDO LA APLICACIÓN CON MICROSOFT SILVERLIGHT

3.1 CONTROLES PARA LAS INTERFACES DE USUARIO

3.1.1 BLOQUES DE CONSTRUCCIÓN

Silverlight proporciona muchos controles útiles para mostrar la información y la manipulación de datos de entrada. Antes de llegar a las características específicas de cada control, es importante entender la funcionalidad básica de todos los controles disponibles para Silverlight.

La figura 3.1 muestra un diagrama de clase abreviado con un subconjunto de controles y de paneles de Silverlight 4.

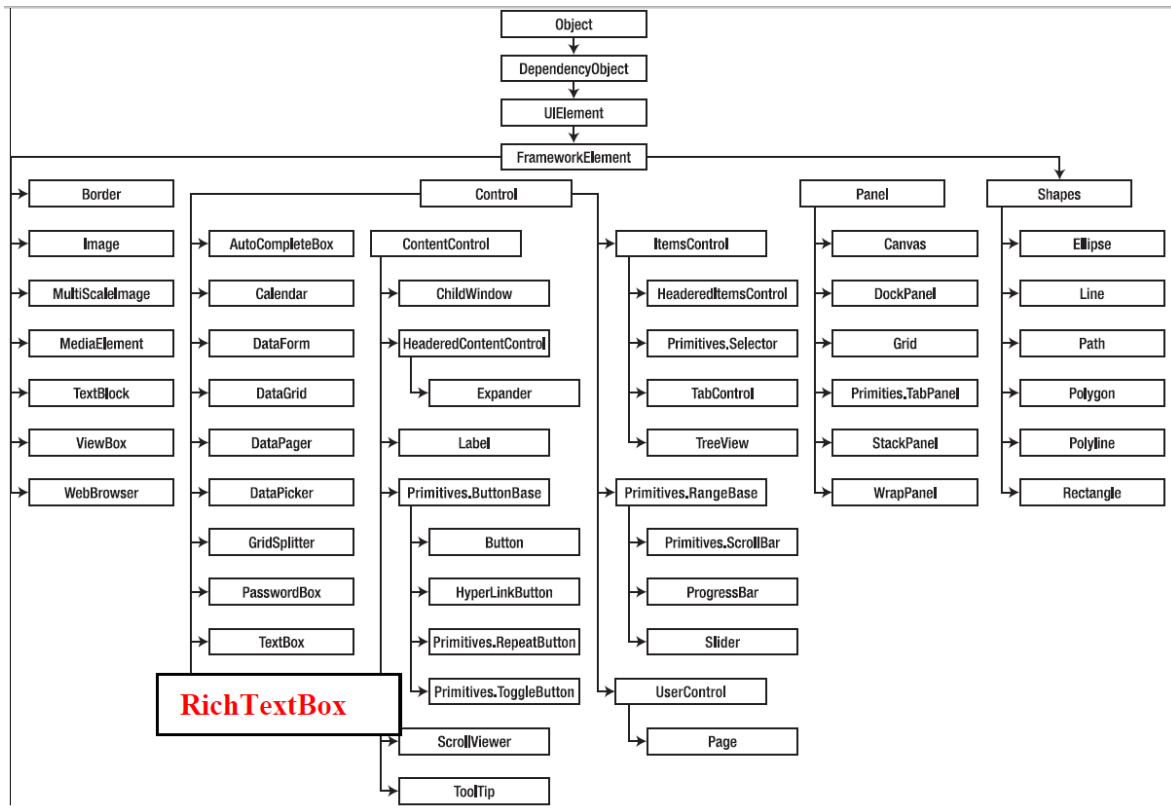


Figura 3. 1. Clase Jerárquica de la interfaz de usuario en Silverlight 4

3.1.1.1. DependencyObject

La clase DependencyObject es sin duda la clase más importante en Silverlight ya que habilita los servicios del sistema de propiedades de dependencia en sus numerosas clases derivadas.

Los servicios y características de DependencyObject incluyen lo siguiente:

- Compatibilidad con el host de propiedades de dependencia.
- Compatibilidad con el host de propiedades de dependencia personalizadas.
- Compatibilidad con el host de propiedades asociadas.
- Compatibilidad con el host de propiedades adjuntas personalizadas.
- Métodos para obtener y establecer valores de las propiedades de dependencia que existan en DependencyObject.
- API del distribuidor para escenarios de subprocesos avanzados.
- Enlace de datos básico y compatibilidad de estilo, permitiendo establecer las propiedades como expresiones que se van a evaluar en algún momento posterior de la duración de un objeto.

Sus características más importantes son los medios que proporciona, que se muestra en la Tabla 3.1.

Método	Descripción
CheckAccess	Determina si el subproceso que realiza la llamada tiene acceso a este objeto.
ClearValue	Borra el valor local de una propiedad de dependencia.
GetAnimationBaseValue	Devuelve cualquier valor base establecido para una propiedad de dependencia de Silverlight, que se aplicaría en los casos en que una animación no esté activa.
ReadLocalValue	Devuelve el valor local de una propiedad de dependencia, si hay un valor local establecido.
SetValue	Establece el valor local de una propiedad de dependencia de un objeto DependencyObject.

Tabla 3. 1. Métodos de la clase System.Windows.DependencyObject

3.1.1.2. Threading y la interfaz de usuario

Silverlight es un entorno multihilo. Usted no puede modificar los elementos de una interfaz de usuario desde un subproceso, ya que puede dar lugar a una serie de problemas. La forma correcta de modificar un interfaz de usuario desde un subproceso diferente es el uso de un despachador. La clase `DependencyObject` proporciona una sola propiedad, `Dispatcher`, que contiene una referencia a la operadora asociada. Si desea establecer el valor de un bloque de texto desde un subproceso diferente, debe utilizar `Dispatcher.BeginInvoke` a la cola de la modificación en el trabajo, de la cola del hilo principal de este tipo de elementos:

```
Dispatcher.BeginInvoke (delegado () {textBlock.Text = "ha cambiado";});
```

3.1.1.3. UIElement

`UIElement` proporciona un punto inicial para las características de diseño de Silverlight.

Gran parte del comportamiento de los elementos visibles en una interfaz de usuario de Silverlight que se define en la clase `UIElement`. Esto incluye los eventos de teclado, mouse y entrada del lápiz óptico así como los eventos de foco.

El tipo `UIElement` expone los siguientes miembros en la tabla 3.2.

Método	Descripción
<code>AddHandler</code>	Agrega un controlador de un evento enrutado especificado y agrega el controlador a la colección de controladores del elemento actual. Especifique <code>handledEventsToo</code> como <code>true</code> de modo que el controlador proporcionado se invoque para el evento enrutado ya marcado como controlado por otro elemento a lo largo de la ruta de evento.
<code>Arrange</code>	Coloca los objetos secundarios y determina el tamaño de <code>UIElement</code> . Los objetos primarios que implementan un diseño personalizado de sus elementos

	secundarios llaman a este método desde las implementaciones de invalidación del diseño para formar una actualización recursiva del diseño.
CaptureMouse	Establece la captura del mouse en UIElement.
InvalidateArrange	Invalida el estado de organización (diseño) de UIElement. Después de la invalidación, UIElement actualizará su diseño, lo que se producirá de forma asincrónica.
InvalidateMeasure	Invalida el estado de medida (diseño) de UIElement.
Measure	Actualiza la propiedad DesiredSize de UIElement. Normalmente, los objetos que implementan el diseño personalizado para sus elementos secundarios del diseño llaman a este método desde sus propias implementaciones de MeasureOverride para formar una actualización recursiva del diseño.
OnCreateAutomationPeer	Cuando se implementa en una clase derivada, devuelve implementaciones AutomationPeer específicas de la clase para la infraestructura de automatización de Silverlight.
ReleaseMouseCapture	Quita la captura del mouse de UIElement. Normalmente, después de esta llamada, ningún objeto tiene la captura del mouse.
RemoveHandler	Quita el controlador de eventos enrutados especificado de UIElement.
TransformToVisual	Devuelve un objeto de transformación que se puede usar para transformar las coordenadas desde el objeto UIElement en el objeto visual especificado.
UpdateLayout	Se asegura de que todas las posiciones de objetos

	secundarios de UIElement se actualizan correctamente para el diseño.
--	----------------------------------------------------------------------

Tabla 3. 2. Métodos de la clase System.Windows.UIElement

Las propiedades del tipo UIElement se describen en la tabla 3.3

Propiedad	Descripción
AllowDrop	Obtiene o establece un valor que determina si UIElement puede ser un destino de colocación en las operaciones de arrastrar y colocar de Silverlight.
CacheMode	Obtiene o establece un valor que indica que el contenido representado debe almacenarse en la memoria caché cuando sea posible.
Clip	Obtiene o establece el objeto Geometry utilizado para definir el contorno del contenido de UIElement.
DesiredSize	Obtiene el tamaño que este UIElement calculó durante el paso de la medida del proceso de diseño.
Dispatcher	Obtiene el objeto Dispatcher al que está asociado este objeto. (Se hereda de DependencyObject).
Effect	Obtiene o establece el efecto de sombreador de píxeles que se va a utilizar para representar esta instancia de UIElement.
IsHitTestVisible	Obtiene o establece si el área contenida de UIElement puede devolver valores true para la prueba de posicionamiento.
Opacity	Obtiene o establece el grado de opacidad del objeto.
OpacityMask	Obtiene o establece el pincel utilizado para modificar la opacidad de las regiones de este objeto.
Projection	Obtiene o establece la proyección en perspectiva

	(efecto 3D) que se va a aplicar al representar esta instancia de UIElement.
RenderSize	Obtiene el tamaño de representación final de un UIElement.
RenderTransform	Obtiene o establece información de transformación que afecta a la posición de representación de UIElement.
RenderTransformOrigin	Obtiene o establece el punto de origen de cualquier posible transformación de representación declarada por RenderTransform, relativo a los límites de UIElement.
UseLayoutRounding	Obtiene o establece un valor que determina si la representación del objeto y su subárbol visual debe utilizar un comportamiento de redondeo que alinee la representación a píxeles enteros.
Visibility	Obtiene o establece la visibilidad de un UIElement. Un UIElement que no está visible no representa ni comunica su tamaño deseado al diseño.

Tabla 3. 3. Propiedades de la clase System.Windows.UIElement

UIElement también define varios eventos importantes, que se muestra en la Tabla 3.4

Evento	Descripción
DragEnter	Tiene lugar cuando el sistema de entrada comunica un evento de arrastre subyacente que tiene este elemento como destino.
DragLeave	Se produce cuando el sistema de entrada comunica un evento de arrastre subyacente que tiene este elemento como origen.
DragOver	Se produce cuando el sistema de entrada comunica un evento de arrastre subyacente que tiene este elemento

	como posible destino de colocación.
Drop	Se produce cuando el sistema de entrada comunica un evento de colocación subyacente que tiene este elemento como destino de la colocación.
GotFocus	Se produce cuando UIElement recibe el foco.
KeyDown	Se produce cuando se presiona una tecla del teclado mientras UIElement tiene el foco.
KeyUp	Se produce cuando se suelta una tecla del teclado mientras UIElement tiene el foco.
LostFocus	Se produce cuando UIElement pierde el foco.
LostMouseCapture	Aparece cuando el objeto UIElement pierde la captura del mouse.
MouseEnter	Se produce cuando el mouse (o un lápiz) entra en el área delimitadora de UIElement.
MouseLeave	Se produce cuando el mouse (o el lápiz) sale del área delimitadora de UIElement.
MouseLeftButtonDown	Se produce cuando se presiona el botón primario del mouse (o cuando la punta del lápiz toca el Tablet PC) mientras el puntero del mouse está sobre un objeto UIElement.
MouseLeftButtonUp	Se produce cuando se suelta el botón primario del mouse (o la punta del lápiz se aparta del Tablet PC) mientras el mouse (o el lápiz) está sobre UIElement (o mientras UIElement tiene la captura del mouse).
MouseMove	Se produce cuando cambia la posición de las coordenadas del mouse (o lápiz) mientras está sobre UIElement (o mientras UIElement tiene la captura del

	mouse).
MouseDown	Se produce cuando se presiona el botón secundario mientras el puntero del mouse está sobre un objeto UIElement.
MouseUp	Se produce cuando se suelta el botón secundario mientras el puntero del mouse está sobre un objeto UIElement. Sin embargo, este evento solo se generará si un llamador marca el evento MouseDown anterior como controlado; vea la sección Comentarios.
MouseWheel	Se produce cuando el usuario gira la rueda del mouse mientras el puntero del mouse está sobre UIElement o mientras UIElement tiene el foco.
TextInput	Se produce cuando un elemento de la interfaz de usuario obtiene el texto de forma independiente del dispositivo.
TextInputStart	Se produce cuando un elemento de la interfaz de usuario obtiene inicialmente el texto de forma independiente del dispositivo.
TextInputUpdate	Se produce cuando el texto continúa componiéndose por medio de un editor de métodos de entrada (IME).

Tabla 3. 4. Eventos de la clase System.Windows.UIElement

3.1.1.4. La clase Control

La clase Control es la clase base de muchos de los controles que se agregan a una aplicación. La clase Control define un comportamiento muy reducido y aunque es posible agregar una clase Control a la aplicación, es más habitual agregar un control que herede de Control, como Button o ListBox.

El tipo Control expone los siguientes miembros que se muestran en la tabla 3.5

Nombre	Descripción
Background	Obtiene o establece un pincel que proporciona el fondo del control.
BorderBrush	Obtiene o establece un pincel que describe el fondo del borde de un control.
BorderThickness	Obtiene o establece el grosor del borde de un control.
DefaultStyleKey	Obtiene o establece la clave que hace referencia al estilo predeterminado del control.
FontFamily	Obtiene o establece la fuente utilizada para mostrar texto en el control.
FontSize	Obtiene o establece el tamaño del texto de este control.
FontStretch	Obtiene o establece el grado en el que se comprime o se expande una fuente en la pantalla.
FontStyle	Obtiene o establece el estilo en que se representa el texto.
FontWeight	Obtiene o establece el grosor de la fuente especificada.
Foreground	Obtiene o establece un pincel que describe el color de primer plano.
HorizontalContentAlignment	Obtiene o establece la alineación horizontal del contenido del control.
IsEnabled	Obtiene o establece un valor que indica si el usuario puede interactuar con el control.

IsTabStop	Obtiene o establece un valor que indica si un control está incluido en la navegación por tabulación.
Padding	Obtiene o establece el relleno interior del control.
TabIndex	Obtiene o establece un valor que determina el orden en el que los elementos reciben el foco cuando el usuario navega por los controles utilizando la tecla TAB.
TabNavigation	Obtiene o establece un valor que modifica el modo en que la tabulación y el control TabIndex funcionan para este control.
Template	Obtiene o establece una plantilla de control.
UseLayoutRounding	Obtiene o establece un valor que determina si la representación del objeto y su subárbol visual debe utilizar un comportamiento de redondeo que alinee la representación a píxeles enteros. (Se hereda de UIElement).
VerticalContentAlignment	Obtiene o establece la alineación vertical del contenido del control.

Tabla 3. 5. Propiedades de la clase System.Windows.Controls.Control

3.1.2. DISEÑO DE GESTIÓN Y CONTROL DE AGRUPACIÓN

Tener una variedad de controles y otros objetos visuales nos da la materia prima para las interfaces de usuario, pero con el fin de formar una interfaz de usuario completa, estos objetos deben ser colocados en la pantalla. Esto se logra a través de la clase Panel la clase base de contenedores de diseño.

3.1.2.1 Canvas

Define un área en la que pueden colocarse explícitamente elementos secundarios utilizando las coordenadas relativas a dicha área.

```
<Canvas Width="640" Height="480" Background="White">
  <Rectangle Canvas.Left="30" Canvas.Top="30"
    Fill="red" Width="200" Height="200" />
</Canvas>
```

Se generan unos resultados similares a los de la figura 3.2.

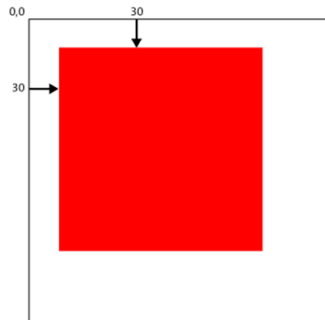


Figura 3. 2. Colocación de un rectángulo dentro de un control Canvas

Canvas puede contener elementos secundarios, que se representan dentro del área del objeto Canvas. Cada elemento secundario debe ser un UIElement. En XAML, los elementos secundarios se declaran como elementos de objeto que son el XML interno de un elemento de objeto Canvas.

En muchos casos, solo se suele utilizar Canvas como contenedor para otros objetos y no tiene ninguna propiedad visible. Un elemento Canvas no está visible si se cumplen una o varias de las condiciones siguientes:

- La propiedad Height es igual a 0.
- La propiedad Width es igual a 0.
- La propiedad Background es null.
- La propiedad Opacity es igual a 0.

- La propiedad Visibility es Collapsed.
- Uno de los objetos de antecesor del Canvas no está visible.

En la API de JavaScript para Silverlight, los eventos de teclado solo están disponibles para el elemento raíz o los controles que pueden recibir el foco (TextBox, PasswordBox). Canvas es un elemento raíz típico, por lo que a menudo es un objeto común con control de eventos de tecla.

3.1.2.2. StackPanel

Organiza los elementos secundarios en una única línea que se puede orientar horizontal o verticalmente

Ejemplo:

```
<StackPanel Margin="20">  
  <Rectangle Fill="Red" Width="50" Height="50" Margin="5" />  
  <Rectangle Fill="Blue" Width="50" Height="50" Margin="5" />  
  <Rectangle Fill="Green" Width="50" Height="50" Margin="5" />  
  <Rectangle Fill="Purple" Width="50" Height="50" Margin="5" />  
</StackPanel>
```

Se generan unos resultados similares a la siguiente ilustración.



Figura 3. 3. Colocación de varios rectángulos dentro del StackPanel

StackPanel es uno de los elementos Panel que habilitan el diseño. StackPanel es útil cuando se desea organizar un conjunto de objetos en una lista vertical u horizontal (por ejemplo, un menú horizontal o vertical de elementos). Establezca la

propiedad Orientation para determinar la dirección de la lista. El valor predeterminado de la propiedad Orientation es Vertical.

3.1.2.3. Grid

Define un área de cuadrícula flexible que está compuesta de columnas y filas. Se pueden colocar objetos en celdas concretas del objeto Grid mediante las propiedades adjuntas Grid.Column y Grid.Row.

```
<Grid ...>
```

```
oneOrMoreUIElements </Grid>
```

En la figura 3.4 se muestran tres columnas mediante el ajuste de tamaño de Auto.

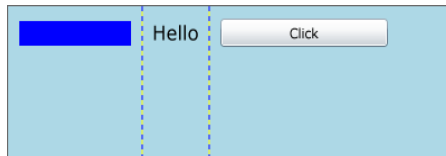


Figura 3. 4. Grid con tres columnas

Las columnas y filas puedan sacar partido del ajuste de tamaño Star¹⁹ para distribuir el espacio restante de forma proporcional. Cuando Star está seleccionado como alto o ancho de una fila o una columna, esa columna o fila recibe una proporción ponderada del espacio restante disponible. El ajuste de tamaño de Star es el comportamiento predeterminado.

3.1.2.4. DockPanel

El DockPanel es uno de los contenedores de diseño nuevo introducido como parte del kit de herramientas de Silverlight. Es necesario agregar una referencia de Systems.Windows.Controls.Toolkit.dll.

El DockPanel está diseñado para colocar el contenido de todo el borde del panel

```
<c:DockPanel Height="200" Width="200"
```

```
Grid.Column="2" Grid.Row="1"
```

```
LastChildFill="False">
```

¹⁹ Proporción ponderada de espacio disponible

```

<Button c:DockPanel.Dock="Top" Content="Top"/>
<Button c:DockPanel.Dock="Bottom" Content="Bottom"/>
<Button c:DockPanel.Dock="Left" Content="Left"/>
<Button c:DockPanel.Dock="Right" Content="Right"/>
<Button c:DockPanel.Dock="Top" Content="Inner Top"/>
<Button c:DockPanel.Dock="Bottom" Content="Inner Bottom"/>
</c:DockPanel>

```

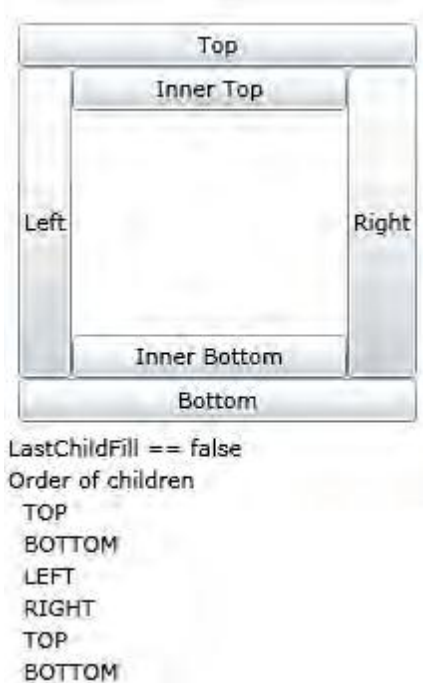


Figura 3. 5. El DockPanel con anidación interior de los controles

3.1.2.5. WrapPanel

El WrapPanel es el contenedor de diseño también disponible como parte del kit de herramientas de Silverlight. Su comportamiento es similar a la StackPanel ya que usted puede poner automáticamente el contenido de uno al lado del otro (de izquierda a derecha o de arriba a abajo), pero añade el comportamiento de ajuste de contenido a la siguiente fila o columna de una cuadrícula invisible cuando el contenido alcanza. Las imágenes de un WrapPanel se muestran en la Figura 3.6

y Figura 3.7 las cuales representan el comportamiento del WrapPanel con la orientación horizontal y vertical.



Figura 3. 6. WrapPanel con orientación horizontal



Figura 3. 7. WrapPanel con orientación vertical

El siguiente fragmento de muestra las imágenes incluidas en el control WrapPanel.

```
<StackPanel>
<TextBlock Text="Horizontally Oriented WrapPanel Example"
Margin="5" FontWeight="Bold"/>
<c:WrapPanel Width="500" Height="250" Orientation="Horizontal">
<Image Width="150" Height="100" Margin="2" Source="res/1.jpg"/>
<Image Width="150" Height="100" Margin="2" Source="res/2.jpg"/>
<Image Width="150" Height="100" Margin="2" Source="res/3.jpg"/>
<Image Width="150" Height="100" Margin="2" Source="res/4.jpg"/>
```

```
<Image Width="150" Height="100" Margin="2" Source="res/5.jpg"/>
```

```
</c:WrapPanel>
```

```
</StackPanel>
```

En el fragmento de código anterior, si cambia la propiedad Orientation de la WrapPanel en Vertical, aumenta el valor de la propiedad Height, y cambiar el texto del TextBlock correctamente, obtendrá un resultado similar a la Figura 3.7

3.1.2.6. TabControl

Representa un control que contiene varios elementos que comparten el mismo espacio en la pantalla

TabControl es útil para minimizar el uso del espacio de pantalla a la vez que permite a una aplicación exponer una gran cantidad de datos. TabControl está compuesto de varios objetos TabItem que comparten el mismo espacio de pantalla. Sólo hay un TabItem en un TabControl visible a la vez. Cuando un usuario selecciona un objeto TabItem, el contenido de TabItem se vuelve visible y se oculta el contenido de los demás objetos TabItem.

La tabla 3.6 describe sus propiedades claves.

Propiedad	Descripción
SelectedContent	Obtiene el contenido del TabItem actualmente seleccionado.
SelectedIndex	Obtiene o establece el índice del primer elemento de la selección actual o devuelve un uno negativo (-1) si la selección está vacía. (Se hereda de Selector).
SelectedItem	Obtiene o establece el primer elemento de la selección actual o devuelve null si la selección está vacía. (Se hereda de Selector).
TabStripPlacement	Obtiene o establece cómo se alinean los encabezados de la ficha en relación con el contenido de la ficha.

Tabla 3. 6. Propiedades de TabControl

Ejemplo

```

<UserControl xmlns:sdk="http://schemas.microsoft.com/winfx/2006/xaml/presentation/sdk"
x:Class="TabControlExample.Page"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

FontFamily="Trebuchet MS" FontSize="11"

Width="400" Height="300">

<Canvas x:Name="LayoutRoot" Background="White">

    <!-- Create a simple TabControl in XAML. -->

        <sdk:TabControl x:Name="tab1" Canvas.Top="20" Canvas.Left="20" Height="200"
Width="300" />

        <sdk:TabControl x:Name="tab2" Canvas.Top="240" Canvas.Left="20" Height="200"
Width="300" />

</Canvas>

</UserControl>

```

En el ejemplo anterior, se generan resultados similares a los de la figura 3.8. Se muestran dos controles TabControl, uno con la primera ficha seleccionada y otro con la segunda ficha seleccionada.

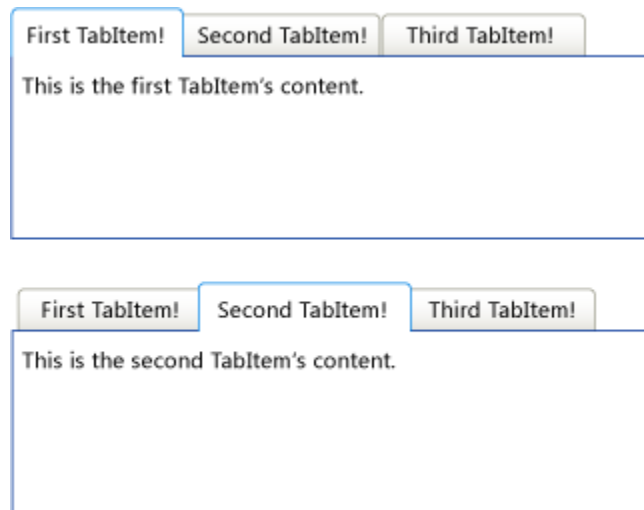


Figura 3. 8. TabControl con diferentes fichas seleccionadas

3.1.2.7. ViewBox

Se define un decorador de contenido que puede ajustar y escalar un elemento secundario único para rellenar el espacio disponible.

Un control Viewbox toma un elemento secundario y lo ajusta o lo escala de acuerdo con el tamaño de Viewbox. Puede controlar los niveles de escala y ajuste usando las propiedades Stretch, StretchDirection, HorizontalAlignment y VerticalAlignment.

Se utiliza normalmente un Viewbox para imágenes gráficas en 2D. No se utiliza normalmente Viewbox para controles ordinarios.

El siguiente ejemplo muestra dos controles Viewbox. La primera 150x200 píxeles Viewbox con la configuración predeterminada muestra la imagen de 1024x768 píxeles se extiende de manera uniforme a llenar el Viewbox todo más pequeño. El segundo 150x200 píxeles Viewbox con StretchDirection muestra sólo la pequeña porción superior de la misma imagen. Esto se debe a la única imagen que se extenderá hasta el tamaño especificado por la dirección del estiramiento, y desde el contenedor Viewbox es más pequeño que la imagen, la imagen no se estira y sólo la parte superior se muestra como se presenta en la figura 3.9.

```
<StackPanel x:Name="LayoutRoot" Background="White">
<TextBlock Text="ViewBox with Stretch=UniformToFill and
StretchDirection=Both" Margin="5" FontWeight="Bold"
HorizontalAlignment="Center"/>
<Viewbox Height="150" Width="200" Stretch="UniformToFill"
StretchDirection="Both">
<Image Margin="2" Source="res/1.jpg"/>
</Viewbox>
<TextBlock Text="ViewBox with StretchDirection=UpOnly"
Margin="2" FontWeight="Bold" HorizontalAlignment="Center"/>
<Viewbox Height="150" Width="200" StretchDirection="UpOnly">
<Image Margin="2" Source="res/1.jpg"/>
```

```
</Viewbox>
```

```
</StackPanel>
```



Figura 3. 9. Imagen contenedor Viewbox

3.1.3. Forms Controls

La entrada del usuario es una parte vital de prácticamente todas las aplicaciones web. La plataforma de Microsoft Silverlight es continuamente mejorada para proporcionar un conjunto bastante sólido de controles que permiten agregar rápidamente ricos elementos de interfaz de entrada y de proceso de usuario con eficacia. Silverlight 4 ha hecho algunas mejoras a los actuales Forms Controls.

3.1.3.1. Button Controls

Existen muchas versiones especializadas en botones, todas heredar directa o indirectamente de la clase `ButtonBase` (en el espacio de nombres `System.Windows.Controls.Primitives`).

Las propiedades de la clase `ButtonBase` se describen en la tabla Tabla 3.7.

Nombre	Descripción
<code>ClickMode</code>	Obtiene o establece cuándo se produce el evento <code>Click</code> .
<code>Command</code>	Obtiene o establece el comando que se invoca cuando se presiona este botón.
<code>CommandParameter</code>	Obtiene o establece el parámetro que se va a pasar a la propiedad <code>Command</code> .
<code>IsFocused</code>	Obtiene un valor que determina si este elemento tiene el

	foco lógico. Se trata de una propiedad de dependencia. (Se hereda de UIElement).
IsMouseOver	Obtiene un valor que indica si el puntero del mouse se encuentra sobre este elemento (incluidos los elementos secundarios del árbol visual). Se trata de una propiedad de dependencia.
IsPressed	Obtiene un valor que indica si un control ButtonBase está actualmente activado.

Tabla 3. 7. Propiedades de la clase System.Windows.Controls.Primitives.ButtonBase

El Command y CommandParameter son propiedades básicas para la aplicación de MVVM (Modelo ViewViewModel) patrón en la aplicación de Silverlight. La clase ButtonBase establece un único evento, la figura 3.10 muestra los distintos botones, HyperlinkButton, RepeatButton y ToggleButton



Figura 3. 10. Colección de diferentes controles de botones

3.1.3.2. TextBox

Representa un control que se puede utilizar para texto multilínea con un único formato.

Puede habilitar el texto multilínea en un control TextBox utilizando la propiedad AcceptsReturn. Utilice la propiedad HorizontalScrollBarVisibility o

`VerticalScrollBarVisibility` para habilitar las barras de desplazamiento horizontales o verticales.

Tal vez desee utilizar un cuadro de texto para mostrar texto, pero sin que el usuario pueda modificar el texto. Para ello, puede usar la propiedad `IsReadOnly`. Si establece la propiedad `IsReadOnly` de un cuadro de texto en `true`, no se admiten los comandos de edición y los eventos `KeyDown` y `KeyUp` se marcan como `handled`. Si necesita texto con formato que no pueda ser modificado por el usuario, utilice el control `TextBlock`.

3.1.3.3. PasswordBox

Representa un control para escribir contraseñas

Puede escribir una línea única de contenido sin ajustar en un control `PasswordBox`. El usuario no puede ver el texto escrito; solo se muestran los caracteres de contraseña que representa el texto. Puede especificar este carácter de contraseña mediante la propiedad `PasswordChar`.

Para aplicar los mismos valores de propiedad a varios controles `PasswordBox`, utilice la propiedad `Style`. Para cambiar la estructura y el comportamiento visuales de `PasswordBox`, copie y modifique su estilo y plantilla predeterminados.

3.1.3.4. RichTextBox

Representa un control de edición de texto enriquecido que admite texto con formato, hipervínculos, imágenes insertadas y otro contenido enriquecido.

Puede usar el control `RichTextBox` para mostrar, escribir, dar formato y modificar texto enriquecido.

`RichTextBox` proporciona características de formato más avanzadas que los controles `TextBox` y `TextBlock`. Puede aplicar carácter y formato de párrafo al texto en `RichTextBox`. Por ejemplo, puede aplicar el efecto a un control `Bold` que a su vez contenga `Italic` y `Underline` a cualquier porción de texto en el control.

La propiedad de contenido de RichTextBox es la propiedad Blocks que se basa en el elemento Paragraph. El contenido puede contener muchos tipos de elementos, que incluyen texto con formato, hipervínculos e imágenes.

Se puede usar el elemento Hyperlink para mostrar hipervínculos en el control RichTextBox. Es preciso establecer la propiedad IsReadOnly de RichTextBox en true para que funcionen los hipervínculos.

3.1.3.5. AutoCompleteBox

Representa un control que proporciona un cuadro de texto para los datos proporcionados por el usuario y una lista desplegable que contiene posibles coincidencias basadas en la entrada del cuadro de texto.

El control AutoCompleteBox permite a los usuarios seleccionar un elemento de la lista desplegable para completar los datos del cuadro de texto en lugar de escribir la entrada completa. La lista desplegable del control AutoCompleteBox se rellenará una vez con posibles coincidencias cuando se cumplan las condiciones establecidas por MinimumPopulateDelay y MinimumPrefixLength.

AutoCompleteBox incluye varias formas de determinar qué elementos se muestran en la lista desplegable. AutoCompleteBox se puede configurar para filtrar elementos mediante métodos de filtrado basados en texto predeterminados o para usar sus propios métodos de filtrado personalizados. Para cambiar el modo en que se filtran los elementos, establezca la propiedad FilterMode y, opcionalmente, las propiedades ItemFilter o TextFilter.

El comportamiento de selección de la lista desplegable se puede personalizar y se determina mediante un adaptador de selección. Un adaptador de selección es un control que suele implementar la interfaz ISelectionAdapter. El adaptador de selección predeterminado es un control ListBox. Puede mostrar los elementos en el adaptador de selección predeterminado con una plantilla de datos estableciendo la propiedad ItemTemplate. También puede utilizar un adaptador de selección personalizado en la parte desplegable del control creando una nueva plantilla para el control que contenga el adaptador de selección personalizado.

En la figura 3.11 se muestra un ejemplo de un control autoCompleteBox y sus partes.

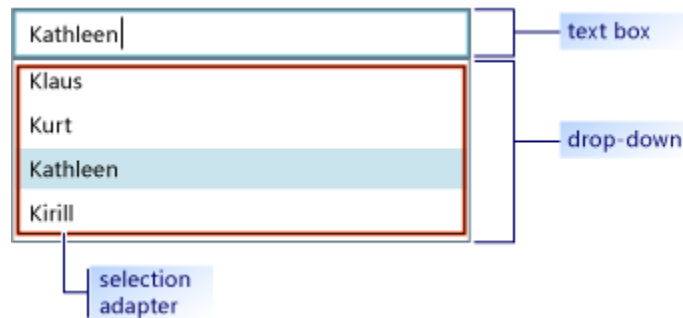


Figura 3. 11. Control autoCompleteBox y sus partes

3.2. ACCESO A DATOS

3.2.1. ENLACE CON DATOS

Binding es una poderosa manera de crear una conexión entre la interfaz de usuario y una fuente de datos. Esta sencilla técnica puede usarse para crear una clara separación entre la interfaz de usuario y sus datos subyacentes y es esencial para la arquitectura, independientemente de la razón, puede utilizar el enlace de datos en su aplicación mediante la creación de una instancia de la clase Binding.

La clase Binding se utiliza para definir una conexión entre un objeto CLR y un componente de interfaz de usuario. Esta conexión se define por tres elementos esenciales: la fuente de los datos (el objeto CLR), el modo de enlace, y el objetivo de los datos. Estos tres elementos forman parte de un marco conceptual que se explica en la figura 3.12.

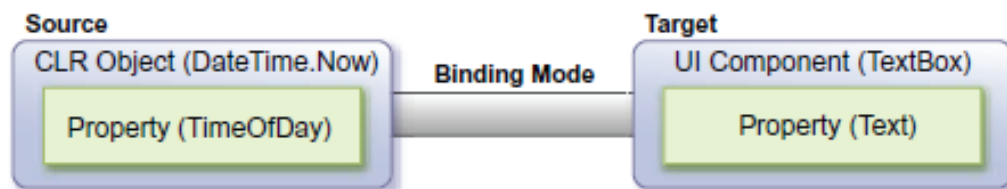


Figura 3. 12. Una visión conceptual de enlace de datos. La fuente posee los datos, la función objetivo (exposiciones, ediciones, y así sucesivamente) de los datos

3.2.1.1. Sintaxis del enlace Binding

Silverlight le da la posibilidad de crear un enlace con dos enfoques diferentes. El primer enfoque permite crear dinámicamente un enlace en tiempo de ejecución. El segundo le da la oportunidad de especificar un enlace en tiempo de diseño. De cualquier manera, el escenario de la figura 3.12 se utilizará para mostrar el resultado de ambos enfoques.

3.2.1.2. Binding en tiempo de ejecución

Enlazar a un origen de datos en tiempo de ejecución es un método común usado en el desarrollo de aplicaciones orientada a eventos, la creación de un enlace en tiempo de ejecución sigue un patrón común. En primer lugar, el XAML en la página:

```
<TextBox X:Name="myTextBox" />
```

A continuación, el código para crear el origen de enlace y de la propia unión:

```
DateTime currentTime = DateTime.Now;
```

```
Binding binding = new Binding("TimeOfDay");
```

```
binding.Source = currentTime;
```

```
binding.Mode = BindingMode.OneWay;
```

```
myTextBox.SetBinding(TextBox.TextProperty, binding);
```

Esto muestra cómo enlazar el valor de una propiedad CLR a un elemento de interfaz de usuario en tiempo de ejecución. El código anterior se une la hora del día para el cuadro de texto creado en XAML. En primer lugar, recuperar el objeto `DateTime` que representa el momento actual en el tiempo. Este objeto se enlaza con el elemento de interfaz de usuario (el cuadro de texto) en sólo cuatro líneas de código. Estas cuatro líneas de código especifican la fuente, el modo de enlace, y el objetivo de un enlace.

3.2.1.3. Binding en tiempo de diseño

Enlazar a un origen de datos en tiempo de diseño es una característica común en los lenguajes de marcado declarativo como XAML. En esencia, este enfoque le permite mantener el código separado de su presentación, para que pueda

aprovechar el desarrollador el flujo de trabajo disponible en Silverlight, como se ve en este código:

```
<TextBox X:Name="myTextBox" Text="{Binding TimeOfDay, Mode=OneWay}" />
```

Esto muestra cómo crear un enlace en tiempo de diseño en XAML. La unión se asocia con un objetivo a través del uso de la sintaxis de extensión de marcado XAML, que utiliza llaves ({}), junto con el uso del nombre de extensión de enlace, informar una propiedad que un origen de datos estará obligado a ello. Esta fuente de datos será un objeto de CLR que tiene una propiedad TimeOfDay, que puede proporcionar o recibir un valor, dependiendo del modo de enlace.

3.2.1.4. La elección de un modo de unión

La clase Binding le da la capacidad para determinar cómo los datos pueden fluir entre la fuente y el destino. Este flujo puede ser controlado por el valor de la propiedad de la instancia de enlace. Esta propiedad representa una de las tres opciones disponibles en el BindingMode enumerador OneTime, OneWay y TwoWay.

ONETIME

La opción ONETIME establece la propiedad de destino a la propiedad de origen cuando un enlace se inicia. Cuando se utiliza este BindingMode, cualquier cambio en el origen de datos no se envía automáticamente a la meta. En cambio, el objetivo se fijará sólo cuando la fuente se ha inicializado, como se muestra en la figura 3.13.

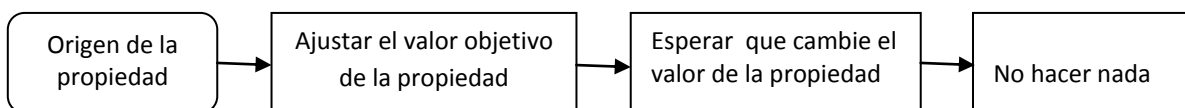


Figura 3. 13. Una visión conceptual de la otrora enlace a un origen de datos. El valor es un principio leer desde la fuente y nunca se volvió a actualizar.

ONEWAY

El BindingMode OneWay es la opción predeterminada cuando se crea un enlace. Esta opción le da la capacidad para recibir automáticamente los cambios de una propiedad de origen, pero la propiedad de origen no va a cambiar si el destino se ve alterado. Este proceso se muestra en la figura 3.14.

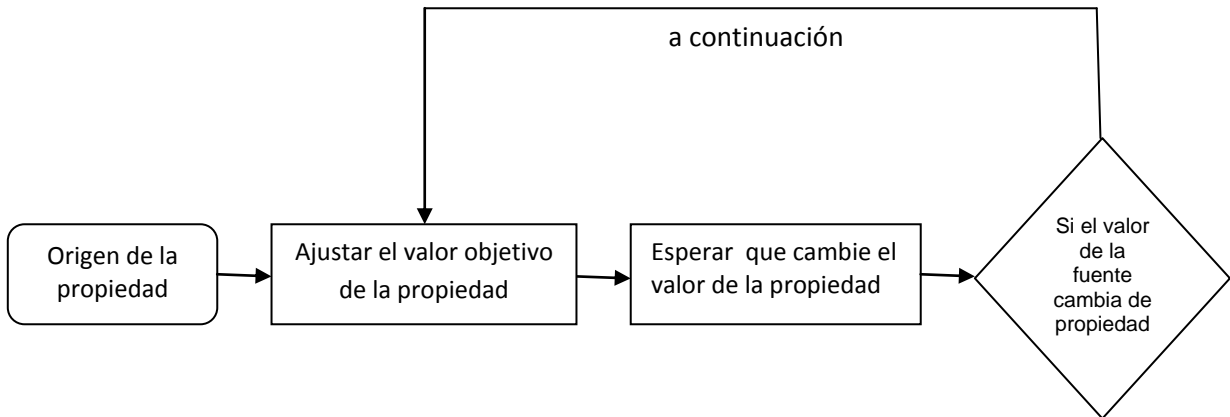


Figura 3. 14. Una visión conceptual de OneWay enlace a un origen de datos. El valor se actualiza cada vez que cambia la fuente, pero los cambios en el valor en el control de destino no lo hacen de nuevo a la fuente.

TwoWay

Binding TwoWay permite que dos propiedades que están obligadas a cambiar al otro. Esto puede parecer recurrente, pero no lo es. Un cambio Binding TwoWay el objetivo cuando se cambia la fuente. Si los cambios de destino, la fuente se actualiza. Este proceso puede verse en la figura 3.15.

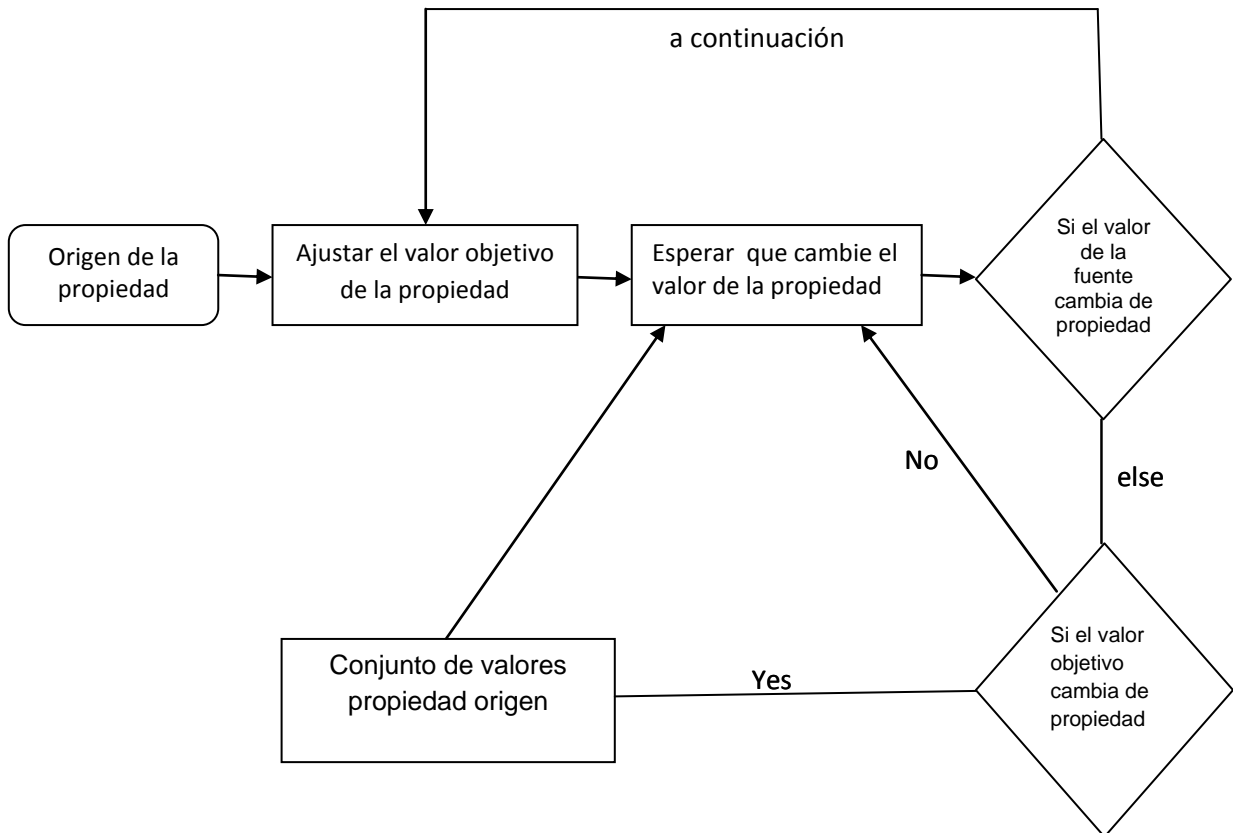


Figura 3. 15. Una visión conceptual de TwoWay enlace a un origen de datos. El control de destino refleja cambios en el origen y la fuente se actualiza con los cambios realizados en el destino.

3.2.2. LA COMPRESIÓN DEL ORIGEN DEL ENLACE

3.2.2.1. Enlace a una propiedad

Silverlight da la flexibilidad de obligar a CLR. Para actualizar se necesita un cambio en la propiedad de CLR, se debe crear un controlador de cambio a la notificación.

Un controlador de cambio a la notificación notifica a un objetivo vinculante de que un cambio se ha hecho. Esto permite que el objetivo responda automáticamente a los cambios. Las propiedades de dependencia ya tienen esta función integrada, pero las propiedades CLR no. Si se quiere que las propiedades CLR puedan difundir los cambios, debe implementar la interfaz `INotifyProperty`, como se muestra en el siguiente código:



3.2.2.2. Enlazar a un objeto

Esta técnica es bastante simple, pero también puede ser algo tediosa si se tiene que obligar a múltiples propiedades de un objeto a una interfaz de usuario. Se puede hacer esta tarea menos pesada utilizando la propiedad DataContext.

La propiedad DataContext le permite compartir un origen de datos a través de una DependencyObject. Esta fuente de datos puede ser utilizado por todos los elementos secundarios de un objeto de dependencia que definen un enlace. Se

puede fácilmente enlazar varias propiedades de un objeto a una interfaz de usuario, con el siguiente código:

```
XAML:
<UserControl x:Class="Chapter011.MainPage"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
  <Grid x:Name="LayoutRoot" Background="White">
    <Grid.RowDefinitions>
      <RowDefinition Height="Auto" />
      <RowDefinition Height="Auto" />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
      <ColumnDefinition />
      <ColumnDefinition />
    </Grid.ColumnDefinitions>
    <TextBlock Text="Name:" />
    <TextBlock Text="Image:" Grid.Column="1" />
    <TextBox Text="{Binding Name, Mode=TwoWay}" Grid.Row="1" />
    <Image Source="{Binding Icon}" Grid.Row="1" Grid.Column="1" />
  </Grid>
</UserControl>
```

← LayoutRoot

```
C#:
Emoticon emoticon =
  new Emoticon("Smiley Face",
    "http://www.silverlightinaction.com/smiley.png");
LayoutRoot.DataContext = emoticon;
```

LayoutRoot's
DataContext

3.2.2.3. Binding con elemento UI

Binding para una o más propiedades de un elemento de interfaz de usuario tiene los valores en una entidad, modelo de vista, o un objeto de negocio es un uso atractivo de la unión. A veces, sin embargo, desea utilizar el enlace de cosas que tradicionalmente no se consideran "datos" las cosas dentro de la interfaz de usuario.

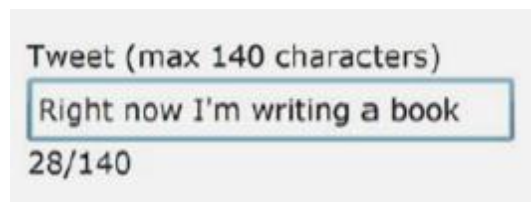


Figura 3. 16. Uso de elementos de enlace para contar los caracteres a medida que escribe en un cuadro de texto

Para producir el cuadro de texto se muestra en la figura 3.16 con el recuento automático de caracteres mediante elemento de enlace, el formato es bastante sencillo y totalmente independiente:

```
<StackPanel Orientation="Vertical" Margin="50">
    <TextBlock Text="Tweet (max 140 characters)" />
    <TextBox x:Name="tweetText"
        MaxLength="140"
        Text="Right now I'm writing a book" />
    <StackPanel Orientation="Horizontal">
        <TextBlock Text="{Binding Text.Length, ElementName=tweetText}" />
        <TextBlock Text="/" />
        <TextBlock Text="{Binding MaxLength, ElementName=tweetText}" />
    </StackPanel>
</StackPanel>
```

Este XAML mostrará un cuadro de texto con un número de caracteres por debajo de ella. El número de caracteres se actualizará en tiempo real para mostrar el número de caracteres escritos en el cuadro de texto. Tenga en cuenta también que el MaxLength que aparece en el cuadro de texto en realidad proviene de la propia TextBox (el 140 en la etiqueta no es). El elemento clave que hace que esto suceda es el parámetro ElementName en la expresión de enlace. ElementName es, como sugiere el nombre de otro elemento de la página XAML.

3.2.2.4. Binding a un elemento índice

Silverlight 3 introdujo la posibilidad de obligar a un elemento numéricamente indexado a una colección. Esto puede ser útil en los casos en que pueda tener propiedades indizadas en una clase o que realmente desea obtener sólo un elemento específico de una colección más grande sin pre filtrado en el código. Una vez que se ha configurado, para hacer referencia a elementos individuales de la colección utilizando la sintaxis de índice dentro de la instrucción obligatoria, como se muestra siguiente código en donde enlaza un elemento específico a una colección, utilizando un índice numérico:

```

C#:
public class Repository
{
    ...
    private ObservableCollection<Emoticon> _emoticons;
    public ObservableCollection<Emoticon> Emoticons
    {
        get { return _emoticons; }
    }
}

XAML:
<UserControl.Resources>
    <local:Repository x:Key="repository" />
</UserControl.Resources>

<Grid x:Name="LayoutRoot"
    DataContext="{StaticResource repository}">
    <TextBlock Text="{Binding Emoticons[2].Name}" />
</Grid>

```

Collection to bind to

Indexed binding

3.2.2.5. Binding a un elemento clave (cadena indexada)

Silverlight 4 introdujo la posibilidad de obligar a este tipo de estructuras mediante la introducción de clave o una cadena indexada a las expresiones de enlace.

```

C#:
public class Repository
{
    ...
    private Dictionary<string, Emoticon> _emoticons;
    public Dictionary<string, Emoticon> Emoticons
    {
        get { return _emoticons; }
    }
}

XAML:
<UserControl.Resources>
    <local:Repository x:Key="repository" />
</UserControl.Resources>

<Grid x:Name="LayoutRoot"
    DataContext="{StaticResource repository}">
    <TextBlock Text="{Binding Emoticons[Smiley].Name}" />
</Grid>

```

Collection to bind to

Keyed/string-indexed binding

3.2.2.6. Binding a una colección completa

Enlazar a una colección es una tarea importante en muchas aplicaciones. Hay muchas veces cuando se necesita mostrar una lista de elementos de una colección. De cualquier manera, estas listas se componen de elementos individuales, por lo que es natural para uso un control derivado de `ItemsControl`. `ItemsControl` es un control básico que se utiliza para mostrar una colección de elementos. Aunque esta técnica es útil en algunas situaciones, el `ItemsControl` proporciona un enfoque más elegante a través de la propiedad `ItemsSource`.

Resultado:



Se enlaza un acoleccion de objetos de Emoticon a un ListBox

XAML:

```
<ListBox x:Name="myListBox" Height="100" />
```

C#:

```
List<Emoticon> emoticons = GetEmoticons();
myListBox.ItemsSource = emoticons;
```

3.2.3. PERSONALIZACIÓN DE LA PANTALLA

Silverlight tiene todo lo necesario para a los valores de formato de pantalla, convertir los valores de entrada y de salida, dar un tratamiento especial para los valores nulos e incluso proporcionar.

3.2.3.1. Los valores de formato

Al escribir código, puede utilizar los valores de formato de la función `String.Format`. Pero hasta Silverlight 4, no hubo buena manera para hacer lo mismo durante una operación de enlace.

Silverlight 4 introdujo la posibilidad de utilizar el formato de cadena cuando se enlaza. La sintaxis es esencialmente la misma que la función `String.Format`. Por ejemplo, esto se establece el valor de `TextBlock` "Fecha de nacimiento: 19 de mayo de 2007" asumiendo que la propiedad `DateOfBirth` en el origen de enlace contiene el valor 19/05/2007:

```
<TextBlock Text="{Binding DateOfBirth, StringFormat=DOB:\{0:D\}}" />
```

Del mismo modo, esta expresión de enlace se establece el valor de `TextBlock` en \$ 1,024.10 asumiendo el decimal `BilledAmount` campo contiene el valor 1024.10m:

```
<TextBlock Text="{Binding BilledAmount, StringFormat=\{0:C\}}" />
```

A veces, simplemente el valor de formato no es suficiente. En esos casos, puede que tenga que realizar una conversión de datos reales y escribir su propio convertidor personalizado.

3.2.3.2. La conversión de los valores durante el enlace

Silverlight le permite convertir dinámicamente valores durante el enlace de datos. Usted puede lograr esto, en primer lugar la creación de una clase personalizada que implementa un convertidor de valores. Se puede hacer referencia directamente en XAML ya que ayuda a mantener el diseño independiente a partir del código.

CREAR UN CONVERTIDOR DE VALORES

Para crear un convertidor de valores, debe crear una clase que implementa la interfaz `IValueConverter`, que le permite crear una lógica personalizada que transforma un valor. Esta transformación puede tener lugar en uno de los dos métodos en función del flujo de datos. El primer método, `Convert`, se utiliza cuando los datos se estén moviendo desde el origen al destino. Si los datos se derivan de la meta de volver a la fuente, por ejemplo, cuando el valor introducido en un cuadro de texto se remonta a su objeto, se utiliza el método llamado `ConvertBack`. Ambos métodos son miembros de la interfaz `IValueConverter`. Esta interfaz y sus métodos se muestran en el siguiente código:

```

public class YesNoValueConverter : IValueConverter
{
    public object Convert(object value, Type targetType,
        object parameter, System.Globalization.CultureInfo culture)
    {
        bool isYes = bool.Parse(value.ToString());
        if (isYes)
            return "Yes";
        else
            return "No";
    }
    public object ConvertBack(object value, Type targetType,
        object parameter, System.Globalization.CultureInfo culture)
    {
        string boolText = value.ToString().ToLower();
        if (boolText == "yes")
            return true;
        else if (boolText == "no")
            return false;
        else
            throw new InvalidOperationException("Please enter 'yes' or 'no'.");
    }
}

```





Este listado muestra un convertidor de valores que convierte entre un bool de Sí o No. Este convertidor utiliza el método Convert cuando los datos están obligados a su interfaz de usuario.

USO UN CONVERTIDOR DE VALORES

El uso de un convertidor de valores consiste en establecer la propiedad convertidor de un objeto Binding, esta propiedad no está establecida (null), pero se puede configurar para hacer referencia a un IValueConverter que se ha creado, se puede hacer referencia a un IValueConverter primero agregando a la colección de Recursos, como se muestra aquí:

```

<UserControl x:Class="Chapter11_9.MainPage"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:local="clr-namespace:Chapter11_9"
Width="400" Height="300">
<UserControl.Resources>
<local:YesNoValueConverter x:Key="myConverter" />

```

```
</UserControl.Resources>
<Grid x:Name="LayoutRoot" Background="White" />
```

```
</UserControl>
```

La propiedad `ConverterCulture` de la clase `Binding` le permite establecer la referencia cultural.

Esta cultura se transmite a lo largo como un objeto `CultureInfo` que puede ser utilizado por los métodos `Convert` y `ConvertBack`. De forma predeterminada, el objeto `CultureInfo` refleja el valor del atributo de `FrameworkElement`. Creación y uso de un convertidor de valores puede ser útil cuando se trabaja con datos. Por ejemplo, Silverlight no tiene soporte para las etiquetas HTML en los controles de texto normal, por lo que puede considerar el uso de un convertidor de valores para fregar las etiquetas HTML de una cadena antes de asociarlo a la interfaz de usuario.

3.2.3.3. Proporcionar valores de reserva por defecto

Las cosas pueden salir mal durante el enlace. La ruta de la propiedad no pueda ser resuelta o una excepción se produce en la obtención del valor. Tal vez el índice o la clave no existe. En esos casos, puede ser útil tener un valor de emergencia previsto en su expresión de enlace. Estos valores se proporcionan mediante la propiedad `FallbackValue`.

En este ejemplo, suponiendo que tiene un campo `ApprovalCode` en su objeto.

```
<TextBlock Text="{Binding ApprovalCode, FallbackValue=Unavailable}" />
```

En muchos casos, que es preferible tener los valores por defecto y retrocesos que se definen en el modelo o modelo de vista, sobre todo debido a que va a hacer que sea más fácil de probar. Pero los valores de reserva en la unión pueden ayudar en caso de apuro o en los casos en que usted necesita para manejar una situación de excepción que ocurre entre el modelo de vista y la vista.

3.2.3.4. Manejo de valores nulos

Similar a los valores de reserva, pero más útil, es la propiedad `TargetNullValue` de la expresión de enlace. `TargetNullValue` le permite mostrar un valor personalizado cuando el valor es nulo.

En muchas aplicaciones, un valor de null realmente significa algo diferente que el valor de vacío o cero. En el primer caso, significa que ningún valor ha sido introducido. Este último indica que un valor se ha introducido, es en blanco o nulo. Para que sea más fácil trabajar con muchas aplicaciones el valor nulo es simplemente reemplazarlo con el valor predeterminado para el tipo. Esto hace que sea más fácil para mostrar en la interfaz de usuario, pero a costa de perder la distinción.

A partir de Silverlight 4, puede conservar el valor nulo y todavía tiene un interfaz de usuario amigable. Basta con proporcionar un `TargetNullValue` en su expresión de enlace:

```
<TextBlock Text="{Binding ApprovalCode, TargetNullValue=(missing)}" />
```

Aunque de gran alcance por su cuenta, a menudo se le quiere mostrar algo más complejo, tal vez contiene múltiples valores enlazados en una lista. Ahí es donde una plantilla de datos entra en juego.

3.2.4. CREACIÓN DE PLANTILLAS DE DATOS

Las plantillas de datos le permiten hacer este cambio fácil, sin afectar el modelo subyacente. Para aprovechar esta función, debe crear un objeto `DataTemplate`. Un objeto `DataTemplate` describe la representación visual de una pieza de información. Este objeto se puede utilizar con dos tipos de controles dentro de la biblioteca de clases de Silverlight. El primero es un `ContentControl`. Más interesante y probablemente utilizada más comúnmente es el `ItemsControl`.

3.2.4.1. Una plantilla de datos con un `ContentControl`

Un `ContentControl` es un tipo de control definido por una sola pieza de contenido. Cada `ContentControl` expone una propiedad denominada de contenido de

plantillas, que especifica la plantilla de datos para utilizar al mostrar el contenido de ContentControl. Este contenido puede ser de estilo con una plantilla de datos utilizando un enfoque similar a la mostrada en el siguiente código:



XAML:

```
<Button x:Name="myButton" Height="70" Width="210">
  <Button.ContentTemplate>
    <DataTemplate>
      <StackPanel Orientation="Horizontal">
        <Image Source="{Binding Icon}" Height="40" Margin="10" />
        <TextBlock Text="{Binding Name}" FontSize="20"
          VerticalAlignment="Center" />
      </StackPanel>
    </DataTemplate>
  </Button.ContentTemplate>
</Button>
```

C#:

```
Emoticon emoticon = new Emoticon("Smiley Face",
  "http://www.silverlightinaction.com/smiley.png");
myButton.Content = emoticon;
```

Esto muestra una plantilla de datos aplicada a un botón. Esta plantilla de datos se aplica a un Emoticon asignado a la propiedad de contenido del objeto Button. Esta propiedad se debe establecer en tiempo de ejecución cuando se utiliza una plantilla de datos. Si la propiedad Content se establece en tiempo de diseño, será superada por la plantilla de datos, lo que no hay datos que se muestran en la interfaz de usuario. Además, si se establece la Propiedad DataContext en tiempo de ejecución en lugar de la propiedad de contenido, sus datos no se muestran.

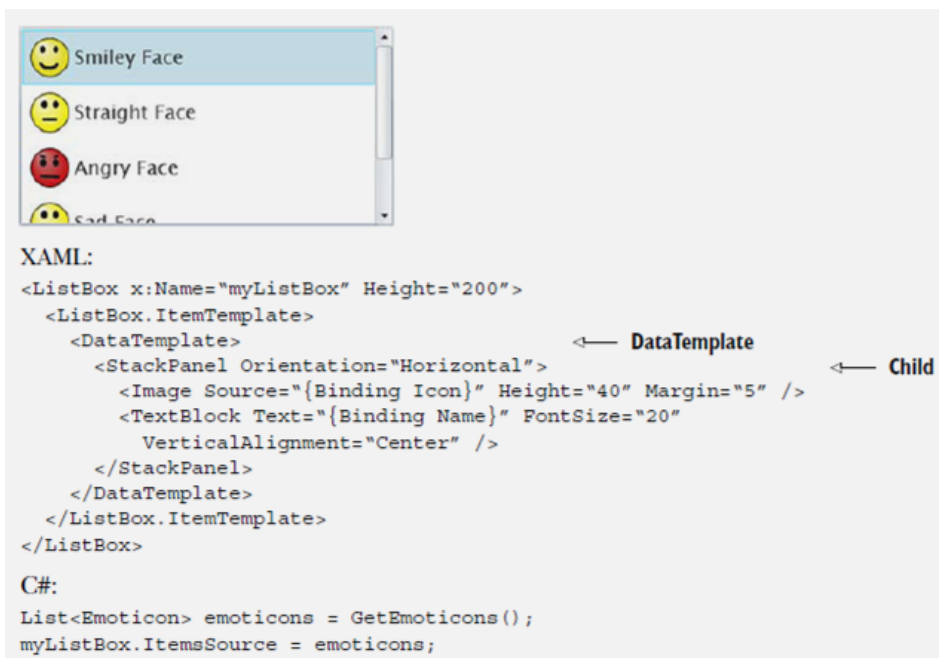
Cuando se enlace datos a un ContentControl, es posible que desee recordar lo siguiente:

- ✓ Al asignar el origen de datos a la propiedad DataContext, utilice la sintaxis de enlace dentro del contenido del control.
- ✓ Al asignar el origen de datos a la propiedad de contenido, utilice una plantilla de datos en su lugar.

3.2.4.2. Representación ItemsControl con una plantilla de datos

El elemento ItemsControl está diseñado para mostrar una colección de elementos, que se enlaza a un control a través de la propiedad ItemsSource. De forma predeterminada, cada elemento se muestra mediante el método ToString de un objeto. Al establecer la propiedad DisplayMemberPath, se puede utilizar una propiedad del CLR para el texto de un artículo

La propiedad ItemTemplate de la clase ItemsControl le permite controlar totalmente cómo cada elemento será mostrado. Esta propiedad utiliza una plantilla de datos para determinar cómo mostrar el resultado de cada elemento de un ItemsControl. Un ItemTemplate básico para una colección de emoticones se muestra en el siguiente código;



The screenshot shows a list box containing four items, each with an emoticon and a text label: a yellow smiley face, a yellow straight face, a red angry face, and a yellow sad face. Below the list box, the XAML code defines the data template for the list box items. The XAML code is as follows:

```
XAML:
<ListBox x:Name="myListBox" Height="200">
  <ListBox.ItemTemplate>
    <DataTemplate>
      <StackPanel Orientation="Horizontal">
        <Image Source="{Binding Icon}" Height="40" Margin="5" />
        <TextBlock Text="{Binding Name}" FontSize="20"
          VerticalAlignment="Center" />
      </StackPanel>
    </DataTemplate>
  </ListBox.ItemTemplate>
</ListBox>
```

The C# code is as follows:

```
C#:
List<Emoticon> emoticons = GetEmoticons();
myListBox.ItemsSource = emoticons;
```

Annotations in the XAML code indicate that the DataTemplate is a child of the ItemTemplate, and the StackPanel is a child of the DataTemplate.

Esto muestra una plantilla de datos básicos asociados con un ItemTemplate. No hay nada complejo acerca de este ejemplo, lo más importante es entender que esta plantilla de datos se utiliza con los elementos de ejecución a través de la propiedad ItemsSource. Además, este comienza a mostrar el poder de utilizar las plantillas de datos.

3.3. VALIDACIONES

Silverlight ofrece varias formas de validar los datos. El enfoque más simple y más antiguo es el uso de validación basado en excepciones.

El `IDataErrorInfo` e interfaces `INotifyDataErrorInfo` son el enfoque más reciente de validar los datos en Silverlight 4. Son un poco más complejas de implementar²⁰.

3.3.1. Excepción de propiedad basada en la validación

La sintaxis de enlace continúa utilizando el parámetro `ValidatesOnExceptions` para permitir la visualización de mensajes de validación cuando un setter de propiedad produce una excepción, pero el control integrado de las plantillas se ha actualizado para ofrecer una visualización adecuada de estado de error.

3.3.1.1. Manipulación de errores en una excepción de validación

Incluso si no se va a tener errores propios de validación basados en excepciones, vale la pena su manipulación con el fin de obtener los beneficios de la comprobación de tipo automático. Enlazar una validación basada en excepciones habilitadas, en su forma más simple, se parece a esto:

```
<TextBox Grid.Row="0" Grid.Column="1"
Text="{Binding LastName, Mode=TwoWay,
    ➤ ValidatesOnExceptions=True}" />
```

El parámetro `ValidatesOnExceptions` informa al sistema de enlace para reportar cualquier excepción vinculante para el cuadro de texto.

Un efecto secundario interesante de esto es que se consigue la validación de tipos de datos de forma gratuita. Por ejemplo, si se intenta introducir letras en una propiedad decimal como el campo Salario, obtendrá un error de coincidencia de tipo de validación.

➤ ²⁰ Silverlight_4_in_Action, Brown Pete; Printed in the United States of America, ©2010 by Manning Publications Co, Capitulo 13, pag 309

3.3.1.2. Código de validación personalizada

En la clase Empleado, se modificar la propiedad LastName para llevar a cabo una validación básica. El apellido de un campo sea necesario y se debe asegurar que tiene una longitud de al menos dos caracteres:

```
public string LastName
{
    get { return _lastName; }
    set
    {
        if (string.IsNullOrEmpty(value))
            throw new Exception("Last Name is a required field.");
        if (value.Trim().Length < 2)
            throw new Exception("Name must be at least 2 letters long.");
        _lastName = value;
        NotifyPropertyChanged("LastName");
    }
}
```

Para mayor brevedad, he utilizado la clase de excepción base para la validación de nuestros errores. En la práctica, se querrá ser menos genérico con sus tipos de excepción, del mismo modo que al lanzar excepciones para los errores de hardware en el código. Para que este código funcione, se necesita ejecutar sin depurar.

3.3.1.3. Validación de la pantalla de error

Validación mediante código de excepción dentro de las propiedades puede ser conveniente, se hace la depuración a veces difícil porque hay excepciones en la pila. Silverlight 4 introdujo las interfaces IDataErrorInfo y INotifyDataErrorInfo. IDataErrorInfo, que se disponía anteriormente en WPF.

Estas interfaces ayudar a eliminar algunos de los problemas actuales con la validación basado en excepciones, ya que tienen una aplicación completamente

diferente sin excepción. Pero las directrices de estilo y características de visualización de error aún se aplican. Además, ofrecen algunas características, como la validación asíncrona, que sería complicado o imposible de aplicar en un modelo basado en excepciones.



Figura 3. 17. Pantalla de error para el cuadro de texto Apellido, mensaje de error

3.3.2. Validación síncrona con IDataErrorInfo

IDataErrorInfo se introdujo con el fin de abordar algunas de las preocupaciones planteadas por la validación basada en excepciones. A diferencia de validación basada en excepciones, no hay excepciones en la pila de llamadas cuando la validación falla. Este enfoque también es más flexible cuando se trata de errores de validación de ajuste de campos individuales.

3.3.2.1. La interfaz IDataErrorInfo

Situado en el espacio de nombres System.ComponentModel, la interfaz IDataErrorInfo está destinado a ser implementado por cualquier clase que desea utilizar como origen de enlace y también quieren tener errores en la superficie de validación.

La interfaz IDataErrorInfo contiene dos propiedades: Error e Item. Estas propiedades se describen en la tabla 3.8.

Propiedad	Descripción
Error	Ponga esto en un único mensaje de error que se aplica a todo el objeto
Item	Una colección de errores, un índice por nombre de la propiedad. Ajusta los dos para tener errores específicos para cada campo

Tabla 3. 8. Propiedades de IDataErrorInfo

Con una simple colección de mensajes, se puede añadir y eliminar utilizando el código en cualquier lugar de nuestra clase. Además, el mensaje de error class-scoped nos permite ofrecer los errores que son difíciles de colocar a la propiedad individual.

Enlazar ValidatesOnDataErrors

Se tiene que modificar el código XAML del formulario que responda a los errores IDataErrorInfo en lugar de los errores basados en excepciones. La declaración vinculante para cada cuadro de texto debe tener el siguiente aspecto:

```
Text="{Binding LastName, Mode=TwoWay, ValidatesOnDataErrors=True}"
```

Como el nombre sugiere, el establecimiento de ValidatesOnDataErrors a true indica al sistema de enlace para ver la interfaz IDataErrorInfo en su clase y responder a los errores reportados.

3.3.2.2. Simple validación con IDataErrorInfo

Lo primero que debe hacer es implementar IDataErrorInfo en la clase. Las modificaciones a la clase de los empleados se visualizan en el siguiente código.

```

public class Employee : INotifyPropertyChanged, IDataErrorInfo
{
    ...
    #region IDataErrorInfo Members
    private string _dataError = string.Empty;
    string IDataErrorInfo.Error
    {
        get { return _dataError; }
    }
    private Dictionary<string, string> _dataErrors =
        new Dictionary<string, string>();
    string IDataErrorInfo.this[string columnName]
    {
        get
        {
            if (_dataErrors.ContainsKey(columnName))
                return _dataErrors[columnName];
            else
                return null;
        }
    }
    #endregion
}

```

Class-level error property

Field-level error property

El Dictionary de string tiene los error-messages, mientras que la propiedad única class-level tiene el mensaje de error de nivel de clase

3.3.2.3. Combinación de las excepciones y IDataErrorInfo

Cuando se transforma en ValidatesOnDataErrors, se ha eliminado el parámetro ValidatesOnExceptions. Por ejemplo, cuando intenta asignar una cadena como "dfdf" a un int, que obtendrá una excepción. Esta excepción se propaga hacia arriba y, si no se manejan por el sistema obligatorio, simplemente desaparece. Por suerte, esto es fácil de arreglar. Basta con modificar la declaración binding system para incluir ambos parámetros:

Text="{Binding Level, Mode=TwoWay,

➤ ValidatesOnDataErrors=True,

➤ ValidatesOnExceptions=True}"

Cuando se ejecuta, el resultado se parecerá a la figura 3.18. Tenga en cuenta que, dado que la excepción se produce antes de que su código de setter de propiedad se ejecute, esta excepción tiene prioridad por encima de su propio código de validación.



Figura 3. 18. Incorporado en el control basado en excepciones tiene prioridad sobre el código.

3.3.3. VALIDACIÓN ASÍNCRONA CON INotifyDataErrorInfo

IDataErrorInfo es un método de validación de la operación síncrona. Aunque usted puede doblar a la superficie de los errores de forma asincrónica, en realidad no es optimizado para eso. Además, IDataErrorInfo no es compatible con múltiples errores de una sola propiedad. INotifyDataErrorInfo resuelve ambos problemas.

3.3.3.1. La interfaz INotifyDataErrorInfo

Al igual que el interfaz IDataErrorInfo, la interfaz INotifyDataErrorInfo se encuentra en el espacio de nombres System.ComponentModel. La interfaz tiene sólo tres miembros, tal como se muestra en la tabla 3.9, y es conceptualmente similar a IDataErrorInfo pero optimizado para la operación asincrónica.

Descripción	Miembros
GetErrors	Este es un método que devuelve todos los errores de validación para un campo específico. Si el parámetro propertyName es nulo o String.Empty, el método devuelve errores de todo el objeto.
HasErrors	Esta es una propiedad que devuelve true si el objeto tiene errores.
ErrorsChanged	Este es un evento similar al evento PropertyChanged en la unión. Siempre que agregar, quitar, cambia errors, debe aumentar este evento para informar el sistema de enlace

Figura 3. 19. Miembros INotifyDataErrorInfo

Una diferencia de IDataErrorInfo es la adición del evento ErrorsChanged. Desde INotifyDataErrorInfo está destinado a ser utilizado en escenarios de validación asíncrono, utiliza un mecanismo basado en eventos para notificar a los oyentes de errores de validación.

3.3.3.2. Implementación de la interfaz

Como fue en el caso de `IDataErrorInfo`, el aumento de la flexibilidad significa un aumento en el código. La propia interfaz es bastante simple, pero detrás de eso, usted debe mantener varias colecciones con el fin mostrar los errores. El código implementando `INotifyDataErrorInfo` se muestra a continuación.

```
public class Employee : INotifyPropertyChanged, INotifyDataErrorInfo
{
    ...
    #region INotifyDataErrorInfo Members
    private Dictionary<string, ObservableCollection<string>>
        _validationErrors;
    private ObservableCollection<string>
        _classValidationErrors;

    public event EventHandler<DataErrorsChangedEventArgs> ErrorsChanged;

    public Employee()
    {
        _validationErrors =
            new Dictionary<string, ObservableCollection<string>>();
        _classValidationErrors =
            new ObservableCollection<string>();
        CreateErrorsCollection("Level");
        CreateErrorsCollection("Salary");
    }

    private void CreateErrorsCollection(string propertyName)
    {
        if (!_validationErrors.ContainsKey(propertyName))
        {
            _validationErrors.Add(propertyName,
                new ObservableCollection<string>());
        }
    }

    IEnumerable INotifyDataErrorInfo.GetErrors(string propertyName)
    {

```



```

if (!string.IsNullOrEmpty(propertyName))
{
    if (_validationErrors.ContainsKey(propertyName))
        return _validationErrors[propertyName];
    else
        return null;
}
else
{
    return _classValidationErrors;
}
}

bool INotifyDataErrorInfo.HasErrors
{
    get
    {
        if (_classValidationErrors.Count > 0)
            return true;

        foreach (string key in _validationErrors.Keys)
        {
            if (_validationErrors[key].Count > 0)
                return true;
        }

        return false;
    }
}
#endregion
}

```

Return errors for field

Check for existing errors

3.4. PATRÓN MVVM

3.4.1. EL MODELO

El modelo representa los datos o información con la que se trabaja, un ejemplo de modelo puede ser un contacto (como su nombre, número de teléfono, dirección y demás) o la descripción de un punto de publicación para un medio audiovisual transmitido en vivo.

La clave para recordar el modelo es que contiene la información, pero no las acciones o servicios que la manipulan. No es responsable de darle forma para que se vea bien en la pantalla, o de obtener una lista de elementos de un servidor remoto (de hecho, en tal lista cada elemento sería a su vez un modelo). La lógica de la aplicación o “reglas empresariales” son generalmente mantenidas en clases separadas del modelo y actúan en él.

3.4.2. LA VISTA

La vista es la parte con la que más se puede familiarizar y la que ve el usuario. Su papel es representar la información, tomándose a veces ciertas libertades con el

fin de hacerla más clara o presentable. Una vista puede también contener ciertos comportamientos, como el aceptar la entrada de datos. La vista se encarga de esta faceta (teclas presionadas, movimientos del ratón, gestos en una pantalla táctil, y así por el estilo) que eventualmente ejerce influencia en las propiedades del modelo.

En MVVM la vista es activa. A diferencia de una vista pasiva sin conocimiento del modelo, y bajo el manejo total de un controlador o presentador, la vista en MVVM contiene comportamientos, eventos y enlaces a datos que, hasta cierto punto, necesitan saber sobre el modelo subyacente y el modelo de vista

Algo digno de recordar sobre la vista es que **no** es responsable de llevar cuenta de su estado. El modelo de vista se encarga de ello y mantiene la vista al tanto de los cambios.

3.4.3. EL MODELO DE VISTA (NUESTRO CONTROLADOR O PRESENTADOR)

El modelo de vista introduce el concepto de separación de la presentación, es decir, mantiene al modelo separado y protegido de los minuciosos detalles de la vista. Por eso es que el modelo de vista es la pieza clave del trío. Esta separación permite que el modelo se limite a contener los datos.

El modelo de vista también hace disponibles métodos, comandos y otros puntos de acceso que ayudan a mantener el estado de la vista, manipular el modelo en respuesta a acciones de la vista y disparar eventos en la misma.

Los ejemplos del patrón generalmente enfatizan el uso de XAML para definir la vista y enlaces de datos para los comandos y propiedades.

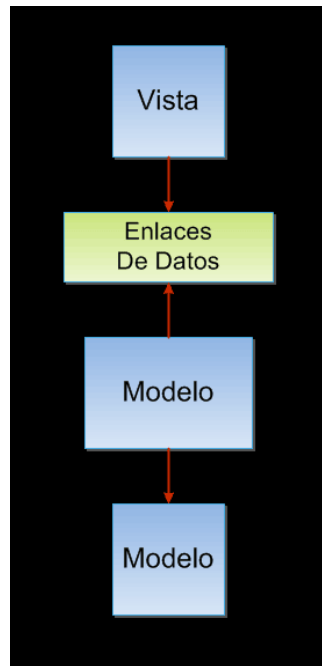


Figura 3. 20 Detalle del modelo de vista

3.4.4. LA VISTA Y EL MODELO DE VISTA

- La vista y el modelo de vista se comunican mediante enlaces de datos, métodos, propiedades, eventos y mensajes
- El modelo de vista expone propiedades (tal como información sobre el estado de la vista, p.ej. el indicador de “ocupado”) y comandos además de modelos
- La vista se encarga de sus propios eventos relacionados con la interfaz al usuario y los pasa al modelo de vista mediante comandos
- Los modelos y propiedades en el modelo de vista son actualizados desde la vista usando enlaces de datos bidireccionales

3.4.5. EL MODELO DE VISTA Y EL MODELO

El modelo de vista es completamente responsable del modelo en este escenario.

Por dicha no está solo:

- El modelo de vista puede que exponga el modelo directamente o mediante propiedades relacionadas para uso en enlaces de datos

- El modelo de vista puede contener interfaces a servicios, datos de configuración y similares, con el fin de obtener y manipular las propiedades que ofrece

3.4.6. INFRAESTRUCTURA BÁSICA PARA MVVM

La infraestructura para MVVM básica requiere sólo dos cosas:

1. Una clase basada en `DependencyObject` o que implemente `INotifyPropertyChanged` de manera que sea utilizable en el enlace de datos, y
2. Algún tipo de mecanismo para comandos

“Silverlight 3 cumple hasta cierto grado con el segundo requisito al proveer la interface `ICommand`. Sin embargo, ahora en Silverlight 4 se tiene algo más completo: los comandos permiten enlazar eventos en la vista al modelo de vista. Es un detalle de implementación, pero facilita el uso del patrón MVVM”²¹.

3.5. WCF RIA SERVICES

“Servicios WCF RIA simplifica el desarrollo de soluciones de n niveles de Rich Internet Applications (RIA), como las aplicaciones de Silverlight”²². Un problema común en el desarrollo de una solución RIA n niveles es la coordinación de la lógica de aplicación entre el nivel medio y nivel de la presentación.

Servicios RIA proporcionan componentes del marco, las herramientas y servicios que hacen que la lógica de aplicación *en* el servidor a disposición del cliente RIA Servicios sin necesidad de duplicar manualmente la lógica de programación.

En la figura 3.20 se muestra una versión simplificada de una aplicación de n niveles. Servicios RIA se centra en la caja entre el nivel de presentación y la capa de acceso a datos (DAL) para facilitar el desarrollo de n niveles con un cliente de Servicios de RIA.

²¹ Patrón Modelo-Vista-Modelo de Vista <http://maromasdigitales.net/2010/05/patron-mvvm-explicado/>

²² Windows Communication Foundation RIA Services;
<http://www.todoroms.com/windows-communication-foundation-ria-services-2>

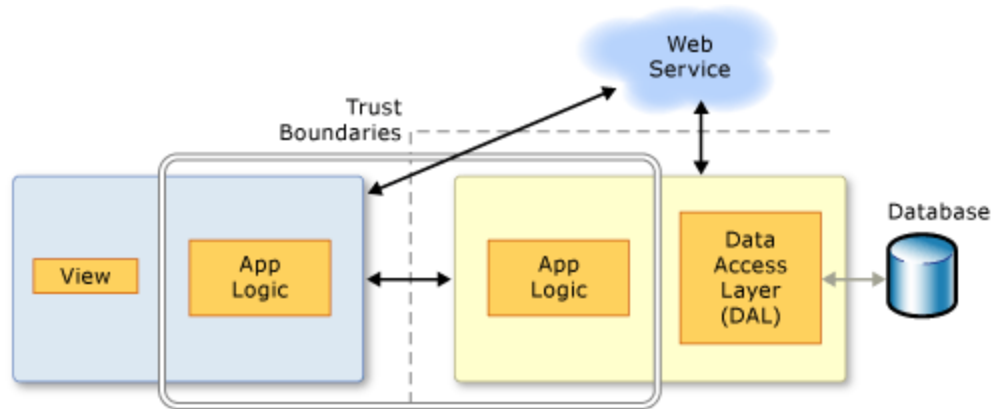


Figura 3. 21. Se muestra una versión simplificada de una aplicación de n niveles

RIA Services agrega herramientas para Visual Studio 2010 que permiten vincular los proyectos de cliente y servidor en una única solución y la generación de código para el proyecto del cliente a partir del código de nivel medio. Los componentes del marco de apoyo a los patrones de prescripción para la escritura lógica de la aplicación para que pueda ser reutilizado en la capa de presentación.

3.5.1. WCF INTEGRACIÓN

En Servicios de RIA, se exponen los datos del proyecto de servidor de proyecto del cliente mediante la adición de los servicios de dominio. El marco RIA Services implementa cada servicio de dominio como un servicio Windows Communication Foundation (WCF).

3.5.1.1. Servicios de dominio

Los servicios son de dominio de Windows Communication Foundation (WCF) que encapsulan la lógica de negocio de una aplicación RIA Servicios WCF. Un servicio de dominio expone un conjunto de operaciones relacionadas en forma de una capa de servicio. Cuando se define un servicio de dominio, debe especificar los datos de las operaciones que se permiten a través del servicio de dominio.

Normalmente, estas tareas implican un pequeño grupo de entidades estrechamente relacionadas. Por ejemplo, en una solicitud de informes de gastos, que podría exponer a las entidades de informes de gastos, las partidas, y los detalles.

3.5.1.2. Servicios de dominio y sus fuentes de datos

El `DomainService` es la clase base para todas las clases que sirven de servicios de dominio. Para crear un servicio de dominio que se une a un objeto de datos personalizada, debe crear una clase que se deriva directamente de `DomainService`, hay clases especiales de lo abstracto se derivan de `DomainService` que utilice en su lugar.

- Para crear un servicio de dominio que se une a una entidad modelo de ADO.NET, se crea una clase que deriva de `LinqToEntitiesDomainService`. RIA Services proporciona la clase `LinqToEntitiesDomainService`.
- Si desea crear un servicio de dominio que expone clases de LINQ to SQL en su aplicación, se crea una clase que deriva de `LinqToSqlDomainService`. Si desea crear un servicio de dominio que expone clases de LINQ to SQL en la aplicación mediante esta clase, se debe descargar el RIA Services Toolkit.

3.5.2. ASEGURAR UNA SOLUCIÓN DE RIA SERVICIOS

Para asegurarse de que su aplicación responde a las preocupaciones de seguridad asociadas con la exposición de un servicio de dominio, debe considerar cuidadosamente cómo se implementa el servicio de dominio.

3.5.2.1. Seguridad de los servicios RIA WCF

Al exponer un servicio de dominio mediante la aplicación de la `EnableClientAccessAttribute` atributo, el servicio de dominio está disponible para todos en la red donde se expone. No se puede suponer que la aplicación cliente es la única aplicación que tendrá acceso al servicio de dominio. Esta consideración es particularmente importante en una red pública. Sin embargo, también es importante en una red restringida, como una red corporativa de datos sensibles cuando se está expuesto.

La siguiente lista es un punto de partida para garantizar el uso seguro de un servicio de dominio.

3.5.2.2. Lista de verificación de seguridad

Para garantizar el uso seguro de un servicio de dominio, se debe tener en cuenta las siguientes orientaciones.

1. Minimizar los datos y las operaciones expuestas por un servicio de dominio. Esta es la primera línea de defensa contra la divulgación de información y denegación de servicio.
 - a. Exponer sólo aquellas entidades que son necesarios por el cliente. Este enfoque puede requerir que la lógica de servidor independiente y validación de la lógica del cliente y de validación, si se le permite reducir el número de entidades expuestas.
 - b. Usted puede utilizar el ExcludeAttribute atributo o modelos de presentación para reducir los datos que está disponible a un cliente
 - c. Métodos requieren de consulta para tomar parámetros que son necesarios en la aplicación, en lugar de confiar en las capacidades de filtrado de datos en LINQ²³.
 - d. Crear los métodos de consulta que sólo proporcionan los datos necesarios para los escenarios específicos de su aplicación. Este enfoque significa que se puede proporcionar múltiples métodos de consulta que las porciones retorno de los datos en lugar de un método de consulta que devuelve todos los datos.
 - e. Filtrar los datos para proporcionar sólo los datos normalmente se requiere para su aplicación.
 - f. Restringir el número de resultados que pueden ser devueltos por una consulta para minimizar la sobrecarga accidental o deliberada del servidor. Se utiliza el ResultLimit propiedad en la QueryAttribute para estrangular el número de resultados que se pueden devolver.

²³ LINQ: Lenguaje Integrado de Consulta que es un conjunto de extensiones de .NET Framework

- g. Minimizar el número de operaciones de cada entidad expuestos.
 - h. Siempre que sea posible, se utiliza el nombre actualizar los métodos que restringen el cual los miembros se pueden actualizar.
2. Restringir el acceso de datos y operación para los usuarios autenticados y usuarios en roles específicos.
- a. Evitar el acceso anónimo siempre que sea posible mediante el RequiresAuthenticationAttribute atributo. Cuando se debe permitir el acceso anónimo, límite que hasta el más mínimo conjunto de servicios de dominio y el conjunto más pequeño de operaciones dentro de los servicios de dominio.
 - b. Añadir la operación específica de RequiresRoleAttribute atributo siempre que sea posible. Considerar la posibilidad de cada operación por separado en un servicio de dominio.
 - c. Considerar el uso de la AuthorizationContext propiedad para proporcionar un modelo de autorización personalizado.
 - d. El tratamiento de los datos enviados por un cliente como sospechoso. Un cliente malicioso (incluso uno que sea autenticado y autorizado) puede proporcionar valores alterados de los valores actuales y originales en un conjunto de cambios. Su lógica de la aplicación no debe asumir que estos valores son dignos de confianza.
3. Utilizar el protocolo https para la autenticación de formularios. El envío de contraseñas en texto claro es una vulnerabilidad significativa, pero puede ser mitigado mediante https.
4. Exponer el número mínimo de puntos finales. De forma predeterminada, RIA Services crear un extremo de binarios para un servicio de dominio. Agregar puntos finales adicionales sólo si tiene clientes que específicamente la necesidad de puntos finales. Deshabilitar todos los extremos que no están en uso.

3.5.3. HERRAMIENTAS Y DOCUMENTACIÓN

La WCF RIA requerirá de varios programas de requisitos previos, tales como Visual Studio 2010 y el tiempo de ejecución de Silverlight para desarrolladores y SDK, ser instalado y configurado correctamente, además de servicios WCF RIA y el Fondo de Operaciones RIA Services Toolkit para trabajar a fondo los tutoriales y cómo temas. También requieren la instalación y configuración de SQL Server 2008 R2 Express con Advanced Services y la instalación de la base de datos OLTP AdventureWorks y LT.

CAPÍTULO IV.- COMPLETANDO LA EXPERIENCIA CON MICROSOFT SILVERLIGHT

4.1. INTEGRACIÓN CON MEDIA(VIDEO Y AUDIO)

4.1.1. INFORMACIÓN GENERAL SOBRE AUDIO Y VÍDEO

Agregar contenido multimedia a una página es tan simple como agregar un objeto `MediaElement` al marcado y proporcionar un identificador uniforme de recursos (URI) al contenido multimedia que se va a reproducir. En el ejemplo siguiente se crea un objeto `MediaElement` y se establece su propiedad `Source` en el URI²⁴ de un archivo de vídeo. La reproducción de `MediaElement` se inicia cuando se carga la página.

```
<MediaElement x:Name="media" Source="xbox.wmv"
    Width="300" Height="300" />
```

El objeto `MediaElement` puede reproducir archivos de vídeo de Windows Media (WMV), audio de Windows Media (WMA) y MP3.

Un objeto `MediaElement` es básicamente una región rectangular que puede mostrar vídeo en su superficie o reproducir audio (en cuyo caso no se muestra vídeo, pero `MediaElement` actúa igualmente como un objeto reproductor con las API correspondientes). Como es un objeto `UIElement`, `MediaElement` admite operaciones de entrada como eventos del mouse y teclado, y puede capturar el foco o el mouse. Puede especificar el alto y ancho de la superficie de presentación del vídeo mediante las propiedades `Height` y `Width`. Sin embargo, para obtener el máximo rendimiento, se debe evitar establecer explícitamente el ancho y alto de `MediaElement`.

4.1.1.1. Propiedades de `MediaElement`

La clase `MediaElement` proporciona varias propiedades específicas del contenido multimedia:

- **AutoPlay** : especifica si la reproducción de `MediaElement` se debe iniciar automáticamente. El valor predeterminado es `true`.

²⁴ URI: identificador uniforme de recursos. También se suele utilizar como sinónimo de URL.

- **IsMuted** : especifica si se ha desactivado el sonido de MediaElement. Un valor de true desactiva el sonido de MediaElement. El valor predeterminado es false.
- **Stretch** : especifica cómo se ajusta el vídeo para rellenar el objeto MediaElement. Los valores posibles son None, Uniform, UniformToFill y Fill. El valor predeterminado es Fill. Para obtener más información.
- **Volume** : especifica el volumen de sonido del objeto MediaElement como un valor de 0 a 1, siendo 1 el valor más alto. El valor predeterminado es 0,5.

4.1.1.2. Controlar la reproducción de contenido multimedia de forma interactiva

Se puede controlar de forma interactiva la reproducción de contenido multimedia mediante los métodos Play, Stop y Pause de un objeto MediaElement. En el ejemplo siguiente se definen un objeto MediaElement y varios botones para controlar la reproducción de contenido multimedia.

El código subyacente crea varios controladores de eventos y utiliza los métodos Stop, Play y Pause para controlar el objeto MediaElement.

```
private void StopMedia(object sender, RoutedEventArgs e)
{
    media.Stop();
}
private void PauseMedia(object sender, RoutedEventArgs e)
{
    media.Pause();
}
private void PlayMedia(object sender, RoutedEventArgs e)
{
    media.Play();
}
```

4.1.1.3. Marcadores de escala de tiempo (puntos de sincronización)

Los marcadores de escala de tiempo son metadatos asociados a un punto determinado en un archivo multimedia. Estos marcadores se crean normalmente por adelantado y se almacenan en el propio archivo multimedia. Se utilizan normalmente para asignar nombres a las distintas escenas de un vídeo, permitir que los usuarios busquen una posición seleccionada o proporcionar claves de scripting.

Cuando el objeto `MediaElement` alcanza un marcador de escala de tiempo durante la reproducción, genera el evento `MarkerReached`. El código puede administrar este evento y utilizar marcadores de escala de tiempo para desencadenar acciones. La propiedad `Markers` de un objeto `MediaElement` permite tener acceso a los marcadores incrustados en los encabezados almacenados en el archivo multimedia que se reproduce actualmente. También puede utilizar esta propiedad para agregar nuevos marcadores de escala de tiempo.

`MediaElement` utiliza objetos `TimelineMarker` para representar la colección de marcadores devuelta por la propiedad `Markers`. El objeto `TimelineMarker` proporciona las propiedades siguientes, que describen su hora, nombre y valor:

- **Time**: estructura `TimeSpan` que especifica la hora a la que alcanza el marcador.
- **Type**: cadena que especifica el tipo del marcador. Este valor puede ser cualquier cadena definida por el usuario.
- **Text** : cadena que especifica el valor del marcador. Este valor puede ser cualquier cadena definida por el usuario.

En el ejemplo siguiente se crea un objeto `MediaElement` y se registra para el evento `MarkerReached`. Se crean también varios objetos `TextBlock`.

```
<StackPanel Margin="40">  
  <StackPanel Orientation="Horizontal">  
    <TextBlock FontSize="12" Foreground="DarkGray">Time:</TextBlock>
```

```

    <TextBlock x:Name="timeTextBlock" FontSize="12" />
</StackPanel>
<StackPanel Orientation="Horizontal">
    <TextBlock FontSize="12" Foreground="DarkGray">Type:</TextBlock>
    <TextBlock x:Name="typeTextBlock" FontSize="12" />
</StackPanel>
<StackPanel Orientation="Horizontal">
    <TextBlock FontSize="12" Foreground="DarkGray">Value:</TextBlock>
    <TextBlock x:Name="valueTextBlock" FontSize="12" />
</StackPanel>
<!-- The MediaElement has the MarkerReached event attached. -->
<MediaElement MarkerReached="OnMarkerReached" HorizontalAlignment="Left"
    Source="thebutterflyandthebear.wmv" Width="300" Height="200" />
</StackPanel>

```

El código subyacente controla el evento MarkerReached. Recupera los valores de Time, Type y Text del marcador de escala de tiempo que activó el evento y muestra esa información en los objetos TextBlock

```

private void OnMarkerReached(object sender, TimelineMarkerRoutedEventArgs e)
{
    timeTextBlock.Text = e.Marker.Time.Seconds.ToString();
    typeTextBlock.Text = e.Marker.Type.ToString();
    valueTextBlock.Text = e.Marker.Text.ToString();
}

```

4.1.1.4. Definir marcadores multimedia

Hay dos maneras de crear marcadores multimedia. La primera consiste en utilizar un editor como el Editor de archivos de Windows Media (que se instala durante la instalación de Windows Media Encoder 9²⁵) o Codificador de Expression Media de

²⁵ Codec que permite la codificación de alta calidad para Windows Media Video

Microsoft para crear marcadores y guardarlos en el propio archivo multimedia o en una secuencia independiente.

El segundo enfoque consiste en generar dinámicamente nuevos objetos TimelineMarker y agregarlos a MediaElement a través de su propiedad Markers. Estos marcadores de escala de tiempo son temporales y no se guardan en el archivo multimedia. Cuando el objeto MediaElement carga un nuevo archivo multimedia, se pierden todos los marcadores de tiempo creados.

4.1.1.5. Listas de reproducción del servidor

Una lista de reproducción del servidor (SSPL), como la de Silverlight, es una secuencia de elementos multimedia (audio o vídeo) que permite a los administradores del servidor controlar la secuencia de transmisión multimedia por secuencias que el usuario ve, así como otros comportamientos. Esta lista de reproducción se puede crear estática o dinámicamente.

Las ventajas de usar listas de reproducción del servidor (SSPL) son las siguientes:

- Dado que el servidor intercambia los archivos multimedia de una lista de reproducción, puede personalizar la experiencia del visor combinando varios archivos multimedia digitales en lo que aparecerá al usuario final como una secuencia de contenido única. Esto minimiza los picos del ancho banda disminuyendo el número de veces que los clientes deben conectarse para recuperar contenido.
- Cuando los usuarios se conectan con un sitio que antes de que se haya iniciado una difusión en directo, puede proporcionar multimedia que se reproduzca en un bucle mientras los usuarios esperan a que comience la difusión en directo.
- Puede utilizar el objeto de servidor para especificar qué archivos multimedia se transmitirán dinámicamente en secuencias. Es decir, puede incluso modificar la lista de reproducción mientras un cliente está viendo una secuencia identificada por la lista de reproducción.

- Puede tener mayor control sobre la reproducción multimedia. Por ejemplo, puede reproducir solo una parte de un archivo multimedia y especificar que se reproduzca cierto archivo si otro archivo da error al cargarse

4.1.1.6. Registros del servidor

Para muchas organizaciones, la transmisión multimedia por secuencias es una fuente de ingresos. Sus clientes transmiten contenido a petición y difusiones en directo pagando una cuota. Estas organizaciones utilizan la información registrada para determinar no solo si un cliente ha visto el contenido, sino durante cuánto tiempo y con qué calidad.

Para entregar el contenido a los usuarios, debe decidir primero lo siguiente:

- Método de entrega: cómo va a entregar el contenido multimedia a los usuarios.
- Contenedor: dónde va a estar incluido el contenido multimedia. Por ejemplo, un MP3, MP4, etc.
- Códec: qué códec desea usar para codificar el contenido multimedia. Por ejemplo, H264-AACL, MP3, etc.

4.1.1.7. MediaStreamSource

Es una parte del runtime de Silverlight que quita la influencia del contenedor de un archivo multimedia, proporcionando a los desarrolladores acceso directo a las API para manipular las secuencias de audio y vídeo codificadas elementales.

Al disponer de acceso a las secuencias elementales, los programadores pueden implementar escenarios que el runtime de Silverlight todavía no ha tenido la oportunidad de implementar. Algunos ejemplos podrían ser la compatibilidad con el protocolo RTSP:T, la compatibilidad con el protocolo SHOUTcast, la repetición de audio ininterrumpida, la compatibilidad con metadatos ID3 v1 e ID3 v2, y muchos otros escenarios.

4.1.1.8. Administración de derechos digitales (DRM)

La integración de Administración de derechos digitales en aplicaciones de Silverlight ayudan a proteger y distribuir de forma segura contenido de descarga de transmisión por secuencias o de descarga progresiva para su reproducción en varias plataformas. Silverlight 4 permite proteger contenido WMA y WMV con PlayReady o WM-DRM a través de MediaElement o MediaStreamSource y permite proteger contenido H264 y AAC-LC a través de MediaStreamSource.

4.1.2. FORMATOS MULTIMEDIA, PROTOCOLOS Y CAMPOS DE REGISTRO COMPATIBLES

Se describen los formatos y protocolos admitidos por el objeto MediaElement

- Contenedor: también conocido como "formato de contenedor", guarda los datos multimedia y la información sobre cómo se almacenan esos datos dentro del contenedor.
- Método de entrega: cómo se entrega el contenido multimedia a los usuarios.
- Códec de vídeo: dispositivo o software que habilita la compresión o descompresión de vídeo digital.
- Códec de audio: dispositivo o software que habilita la compresión o descompresión de audio digital.
- Descarga progresiva: entrega el contenido multimedia reproduciendo la parte descargada de un archivo mientras la descarga todavía está en curso.
- Transmisión por secuencias tradicional: entrega y presenta el contenido multimedia sin haberlo almacenado.
- Transmisión por secuencias suave: es igual que la transmisión por secuencias tradicional con la salvedad de que permite el escalado de la transmisión, en función del hardware y la conexión de red del usuario. La proporciona el servidor IIS 7.0.

- Vídeo sin formato: vídeo sin procesar o en un formato sin codificar. Normalmente es contenido capturado en un formato como RGBA o YV12.
- Audio sin formato: audio sin procesar o en un formato sin codificar. Normalmente es contenido capturado como muestras PCM lineales.
- H.264 (ITU-T H.264 / ISO MPEG-4 AVC): estándar de compresión de vídeo, equivalente a MPEG-4 Parte 10.
- Lista de reproducción del servidor (SSPL): una lista de reproducción del servidor es una secuencia de activos multimedia (audio o vídeo) que permite a los administradores del servidor controlar la secuencia multimedia que ven los usuarios.
- ASX: Listas de reproducción del cliente.

A continuación se ofrece una lista resumida de los formatos admitidos por MediaElement. Estas codificaciones se admiten cualquiera que sea la extensión de nombre de archivo.

Vídeo

- Vídeo sin formato
- YV12 - YCrCb(4:2:0)
- RGBA (alfa rojo, verde y azul de 32 bits)
- WMV1: vídeo de Windows Media 7
- WMV2: vídeo de Windows Media 8
- WMV3: vídeo de Windows Media 9
 - Admite perfiles simples y principales.
 - Solo admite contenido progresivo (no entrelazado).
- WMVA: perfil avanzado de vídeo de Windows Media, distinto de VC-1
- WVC1: perfil avanzado de vídeo de Windows Media, VC-1
 - Admite el perfil avanzado.
 - Solo admite contenido progresivo (no entrelazado).

- H264 (ITU-T H.264 / ISO MPEG-4 AVC)
 - Admite códecs H.264 y MP43.
 - Admite los perfiles base, principal y alto.
 - Solo admite contenido progresivo (no entrelazado).
 - Solo admite perfiles de submuestreo de intensidad de color 4:2:0.
 - Admite DRM PlayReady con Mp4 (H264 y AAC-LC)

Audio

- "1". Modulación por impulsos codificados lineal de 8 o 16 bits. En líneas generales, el formato WAV.
- "353" (Microsoft Windows Media Audio v7, v8 y v9.x Standard (WMA Standard))
- "354" (Microsoft Windows Media Audio v9.x y v10 Professional (WMA Professional))
 - Admite la descodificación de fidelidad completa de los modos de velocidad de bits baja (LBR) de WMA 10 Professional en el intervalo de 32-96 kbps.
 - El contenido de audio multicanal (5.1 y 7.1 Surround) se mezcla automáticamente con el estéreo.
 - El audio de 24 bits devuelve silencio.
 - Las tasas de muestreo superiores a 48000 devuelven un código de error de formato no válido en escenarios de un dominio y un código de error 4001 en escenarios de varios dominios.
- "85" (ISO MPEG-1 Layer III (MP3))
- "255" (Codificación de audio avanzada (AAC) ISO)
 - Admite la descodificación de baja complejidad (AAC-LC) con fidelidad completa (hasta 48 kHz).

- El contenido codificado de eficacia alta (HE-AAC) se descodificará solo con fidelidad media (hasta 24 kHz).
- No se admite el contenido de audio multicanal (5.1 Surround).

4.1.2.1. Formatos no compatibles

Los siguientes formatos no son compatibles:

- Contenido de vídeo entrelazado
- Pantalla de Windows Media
- Windows Media Audio Professional sin pérdida
- Voz de Windows Media
- Combinación de vídeo de Windows Media y MP3 (vídeo WMV + audio MP3)
- Combinación de WMV con AAC-LC
- VC-1 en MP4

4.1.2.2. Protocolos compatibles

Se admiten los siguientes protocolos/esquemas:

- http
- https
- ms-wmisp (puede usar los monikers mms y rtsp, pero retrocederá a http-streaming, también conocido como ms-wmisp).
- UNC (por ejemplo, \\nombre_de_equipo\ubicación_de_recurso_compartido). Observe que no puede establecer **MediaElement.Source = UNC**. Lo que debe hacer en su lugar es obtener un objeto Stream abierto a la UNC y, a continuación, usar `MediaElement.SetSource(stream)`, donde stream es la secuencia abierta de esa UNC. Observe también que puede experimentar degradaciones de rendimiento si intenta reproducir archivos grandes sobre una UNC, porque se debe descargar el archivo completo antes de que pueda comenzar la reproducción.

4.1.2.3. Servidores proxy y transmisión por secuencias en Silverlight

En las redes donde hay un servidor proxy, puede ser necesario realizar una configuración especial en el cliente para que la transmisión por secuencias funcione en Silverlight. En general, Silverlight solamente examina la configuración del sistema para tomar una decisión sobre la configuración del servidor proxy. Por tanto, la configuración de un proxy en Firefox no afectará al modo de funcionamiento de la transmisión por secuencias en las plataformas de Windows o Macintosh.

- **Windows:** los usuarios finales deben asegurarse de que tienen la configuración de proxy correcta en Opciones de Internet. Se puede tener acceso a estas opciones a través de Internet Explorer o directamente desde el Panel de control. Por consiguiente, la transmisión por secuencias en Silverlight debería funcionar correctamente siempre y cuando los usuarios sean capaces de navegar por Internet con Internet Explorer.
- **Macintosh:** los usuarios finales deben asegurarse de especificar manualmente la dirección IP de los servidores proxy en Preferencias de red. Se puede tener acceso a esta configuración a través de Safari o directamente desde Preferencias del sistema. A diferencia de Windows, en Silverlight 1.0 no se permite especificar un script de configuración PAC para el servidor proxy. Como resultado, existe la posibilidad de que un usuario pueda explorar la Web correctamente sin poder transmitir por secuencias el contenido multimedia en Silverlight.

4.1.3. INFORMACIÓN GENERAL SOBRE CÁMARAS WEB Y DISPOSITIVOS

En Silverlight 4, es posible capturar y mostrar contenido de audio y de vídeo de dispositivos multimedia, como cámaras web, sintonizadores de TV y micrófonos. Esto da lugar a diversos escenarios que pueden mejorar la experiencia del usuario con las aplicaciones de Silverlight. Algunos de estos escenarios son la captura y la presentación de imágenes, la carga de imágenes de perfil en aplicaciones de redes sociales, el correo y las agendas de vídeo y de

audio, la toma de notas de audio, las cámaras de seguridad, la realidad aumentada y los movimientos corporales.

Las principales clases que se usan para la captura de contenido de audio y de vídeo de los dispositivos multimedia se muestran en la tabla 4.1:

Clase	Descripción
CaptureDeviceConfiguration	Representa una clase auxiliar para obtener información sobre los dispositivos de captura disponibles (audio o vídeo) y solicitar permiso del usuario del cliente para obtener acceso a las capturas de los dispositivos disponibles.
CaptureSource	Proporciona métodos que funcionan con capturas de audio o vídeo específicas del dispositivo de captura asociado.
VideoCaptureDevice	Describe el formato de vídeo deseado y compatible de un dispositivo de captura de vídeo, como una cámara web.
AudioCaptureDevice	Describe los formatos deseados y compatibles con un dispositivo de captura de audio, como un micrófono.
VideoSink	Expone el gráfico de captura para los dispositivos de vídeo. Derive de esta clase para recibir información de vídeo y obtener el gráfico de captura a través de VideoSink.CaptureSource.
AudioSink	Expone el gráfico de captura para los dispositivos de audio. Derive de esta clase para recibir información de audio y obtener el gráfico de captura a través de AudioSink.CaptureSource.

Tabla 4. 1. Clases para la captura de contenido de audio y de vídeo

4.2. ANIMACIÓN

4.2.1. INFORMACIÓN GENERAL SOBRE ANIMACIONES

Una animación es una ilusión que se crea mediante el cambio rápido entre una serie de imágenes, cada una de las cuales es ligeramente diferente de la anterior. El cerebro percibe el grupo de imágenes como una sola escena cambiante. En las películas, este efecto se obtiene mediante el uso de cámaras que graban muchas fotografías, o fotogramas, cada segundo. Cuando los fotogramas se reproducen en un proyector, la audiencia ve una imagen en movimiento. En Silverlight, los objetos se animan al aplicar animaciones a sus propiedades individuales.

4.2.1.1. Tipos de animación

Existen distintos tipos de animaciones para los diversos tipos de propiedades. Para animar una propiedad que acepta un valor Double, como la propiedad Width de un elemento, se usa una animación que genera valores Double, como DoubleAnimation. Para animar una propiedad que acepta un valor Point, se usa una animación que genera valores Point, como PointAnimation, etc.

En la tabla 4.2 se describen las categorías de animación y sus convenciones de nomenclatura.

Categoría	Descripción	Convención de nomenclatura
Animación From/To/By	<p>Crea una animación entre un valor inicial y final:</p> <ul style="list-style-type: none"> • Para especificar un valor inicial, establezca la propiedad From de la animación. • Para especificar un valor final, establezca la propiedad To de la animación. 	tipo Animation

	<ul style="list-style-type: none"> Para especificar un valor final con respecto al valor inicial, establezca la propiedad By de la animación (en lugar de la propiedad To). <p>En los ejemplos de este tema se incluyen estas animaciones porque son las más fáciles de implementar.</p>	
Animación de fotogramas clave	Crea una animación entre una serie de valores especificados mediante objetos de fotograma clave. Las animaciones de fotogramas clave son más eficaces que las animaciones From/To/By porque se puede especificar cualquier número de valores de destino e incluso controlar su método de interpolación. Las animaciones de fotogramas clave se describen con detalle en Animaciones de fotogramas clave.	tipo AnimationUsingKeyFrames

Tabla 4. 2. Categorías de animación

En la tabla 4.3 se muestran varios tipos de animación comunes y algunas propiedades con las que estos se utilizan.

Tipo de propiedad	Animación básica correspondiente (From/To/By)	Animación de fotogramas clave correspondiente	Ejemplo de uso
-------------------	-----------------------------------------------	-----------------------------------------------	----------------

Color	ColorAnimation	ColorAnimationUsingKeyFrames	Animar la propiedad Color de un objeto SolidColorBrush o GradientStop.
Double	DoubleAnimation	DoubleAnimationUsingKeyFrames	Animar la propiedad Width de un objeto Rectangle o la propiedad Height de un objeto Ellipse (o cualquier objeto FrameworkElement)
Point	PointAnimation	PointAnimationUsingKeyFrames	Animar la posición Center de un objeto EllipseGeometry.
Object	Ninguna	ObjectAnimationUsingKeyFrames	Animar la propiedad Fill de un objeto GradientBrush a otro.

Tabla 4. 3. Tipos de animación

4.2.1.2. Funciones de entradas y salidas lentas

Las funciones de entradas y salidas lentas permiten aplicar fórmulas matemáticas personalizadas a las animaciones.

Además de crear su propia función de entradas y salidas lentas heredando de la clase `EasingFunctionBase`, puede utilizar una de las funciones de entradas y salidas lentas proporcionadas por el motor en tiempo de ejecución a fin de crear efectos comunes.

- **BackEase** : contrae el movimiento de una animación justo antes de que empiece a animarse en el trazado indicado.
- **BounceEase** : crea un efecto de rebote.
- **CircleEase** : crea una animación que acelera o desacelera mediante una función circular.
- **CubicEase** : crea una animación que acelera o desacelera mediante la fórmula $f(t) = t^3$.
- **ElasticEase** : crea una animación que se parece a un resorte que oscila de un lado a otro hasta detenerse.
- **ExponentialEase** : crea una animación que acelera o decelera mediante una fórmula exponencial.
- **PowerEase** : crea una animación que acelera o desacelera mediante la fórmula $f(t) = t^p$, donde p es igual a la propiedad `Power`.
- **QuadraticEase** : crea una animación que acelera o desacelera mediante la fórmula $f(t) = t^2$.
- **QuarticEase** : crea una animación que acelera o desacelera mediante la fórmula $f(t) = t^4$.
- **QuinticEase** : crea una animación que acelera o desacelera mediante la fórmula $f(t) = t^5$.
- **SineEase** : crea una animación que acelera o desacelera mediante una fórmula sinusoidal.

4.2.2. ANIMACIONES DE FOTOGRAMAS CLAVE

Igual que una animación From/To/By, una animación de fotograma clave anima el valor de una propiedad de destino. Crea una transición entre sus valores de destino a lo largo de su valor de Duration. Sin embargo, una animación From/To/By crea una transición entre dos valores, mientras que una animación de fotograma clave única puede crear transiciones entre cualquier número de valores de destino.

Para animar con una animación de fotograma clave, complete los pasos siguientes:

- Declare la animación y especifique su propiedad Duration, igual que haría para una animación From/To/By.
- Para cada valor de destino, se crea un fotograma clave del tipo adecuado, establezca su valor y su propiedad KeyTime se agrega a la colección KeyFrames de la animación.
- Se asocie la animación a una propiedad, igual que haría con una animación From/To/By.

4.2.2.1. Tipos de animación de fotograma clave

Hay distintos tipos de animaciones para los distintos tipos de propiedades. Para animar una propiedad que acepta una estructura Double (tal como la propiedad Width de un elemento), se usa una animación que genera valores Double. Para animar una propiedad que acepta una estructura Point, se usa una animación que genera valores Point, y así sucesivamente.

Las clases de animación de fotograma clave se ajustan a la siguiente convención de nomenclatura:

tipo_ AnimationUsingKeyFrames

donde tipo_ es el tipo de valor que la clase anima.

Silverlight proporciona las siguientes clases de animación de fotograma clave como se muestran en la tabla 4.4.

Tipo de propiedad	Clase de animación de fotograma clave correspondiente	Los métodos de interpolación admitidos
Color	ColorAnimationUsingKeyFrames	Discreta, lineal, spline
Double	DoubleAnimationUsingKeyFrames	Discreta, lineal, spline
Point	PointAnimationUsingKeyFrames	Discreta, lineal, spline
Object	ObjectAnimationUsingKeyFrames	Discreta

Tabla 4. 4 Clases de animación de fotograma clave

4.2.2.2. Funciones de entradas y salidas lentas

Las funciones de entradas y salidas lentas permiten aplicar fórmulas matemáticas personalizadas a las animaciones. Por ejemplo, puede que desee que un objeto bote con realismo, o que se comporte como si tuviera un resorte. Puede utilizar animaciones de fotogramas clave o incluso animaciones From/To/By para conseguir efectos aproximados a estos, pero se necesitaría bastante trabajo y la animación sería menos precisa que utilizando una fórmula matemática. Se puede aplicar funciones de aceleración a las animaciones de fotogramas clave.

4.2.3. TRABAJAR CON ANIMACIONES MEDIANTE PROGRAMACIÓN

Es posible que en algún momento desee cambiar las propiedades de una animación dinámicamente (sobre la marcha). Por ejemplo, tal vez desear ajustar el comportamiento de una animación que se aplica a un objeto, dependiendo de la ubicación actual del objeto en el diseño, del tipo de contenido que incluye, etc. Se puede manipular las animaciones dinámicamente utilizando código de procedimientos (por ejemplo, C# o Visual Basic).

4.3. GRÁFICOS

4.3.1. FORMAS Y DIBUJOS

En Silverlight, Shape es un tipo de UIElement que permite dibujar una forma en la pantalla. Como se trata de elementos de interfaz de usuario, se pueden utilizar

objetos Shape dentro de una variedad de objetos contenedores como Grid y Canvas

4.3.1.1. Objetos de formas

Silverlight proporciona varios objetos Shape listos para usar, como Ellipse, Line, Path, Polygon, Polyline y Rectangle. Los objetos Shape comparten las siguientes propiedades comunes:

- **Stroke** : describe cómo se pinta el contorno de la forma.
- **StrokeThickness** : describe el grosor del contorno de la forma.
- **Fill** : describe cómo se pinta el interior de la forma.

Propiedades de datos para especificar coordenadas y vértices, medidos en píxeles independientes del dispositivo.

Los objetos Shape se pueden utilizar dentro de objetos Canvas que admite posiciones absolutas de sus objetos secundarios mediante el uso de las propiedades adjuntas Canvas.Left y Canvas.Top.

4.3.1.2. Utilizar Trazados y geometrías

La clase Path permite dibujar curvas y formas complejas. Estas curvas y formas se describen mediante objetos Geometry. Para utilizar un Path, se crea una Geometry y se utiliza para establecer la propiedad Data del objeto Path. Hay gran variedad de objetos Geometry entre los que elegir. Las clases LineGeometry, RectangleGeometry y EllipseGeometry describen formas relativamente simples.

Para crear formas más complejas o crear curvas, utilice PathGeometry.

En XAML, se puede utilizar también una sintaxis abreviada especial para describir un objeto Path. En el ejemplo siguiente, se utiliza la sintaxis abreviada para dibujar una forma compleja.

```
<Canvas>
  <Path Stroke="DarkGoldenRod" StrokeThickness="3"
    Data="M 100,200 C 100,25 400,350 400,175 H 280" />
</Canvas>
```



Figura 4. 1. Ilustración de un trazado con el objeto Path

Los parámetros de datos de Path distinguen mayúsculas de minúsculas. La M mayúscula indica una ubicación absoluta para el nuevo punto activo. Una m minúscula indicaría coordenadas relativas.

4.3.1.3. Pintar Formas

Los objetos Brush se utilizan para pintar las propiedades Stroke y Fill de una forma. En el ejemplo siguiente, se especifican el trazo y el relleno de Ellipse. Tenga en cuenta que para las propiedades de pincel se pueden especificar palabras clave o valores de color hexadecimal.

```
<Canvas Width="300" Height="300" Background="LightGray">
```

```
<Ellipse
```

```
Canvas.Top="50"
```

```
Canvas.Left="50"
```

```
Fill="#FFFFFF00"
```

```
Height="75"
```

```
Width="75"
```

```
StrokeThickness="5"
```

```
Stroke="#FF0000FF"/>
```

```
</Canvas>
```

En la siguiente ilustración se muestra el objeto Ellipse representado.

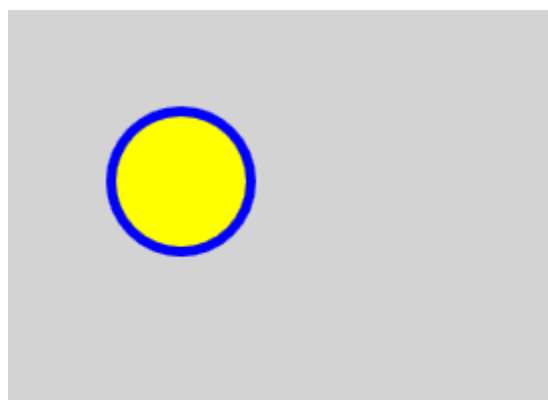


Figura 4. 2. Representacion del objeto Ellipse

4.3.1.4. Transformar Formas

Los objetos Transform se pueden utilizar para transformar las formas (o cualquier otro objeto UIElement) en un plano bidimensional. Los distintos tipos de transformaciones incluyen rotación (RotateTransform), escala (ScaleTransform), sesgo (SkewTransform) y traslación (TranslateTransform).

Una transformación común que se aplica a las formas es el giro. Para girar una forma, cree una RotateTransform y especifique su Angle. Un valor de Angle de 45 gira el elemento 45 grados en el sentido de las agujas del reloj, un ángulo de 90 gira el elemento 90 grados en el sentido de las agujas del reloj, y así sucesivamente. Establezca las propiedades CenterX y CenterY si desea controlar el punto sobre el que se gira el elemento. Estos valores de propiedad se expresan en el espacio de coordenadas del elemento que se transforma. CenterX y CenterY tienen valores predeterminados de 0. Por último, aplique RotateTransform al elemento estableciendo la propiedad RenderTransform de la forma.

En el ejemplo siguiente se muestra una transformación RotateTransform que gira un elemento Rectangle 45 grados sobre el punto (0,0).

```
<Grid x:Name="LayoutRoot" Background="White">
```

```
<Rectangle Width="50" Height="50"
```

```
Fill="RoyalBlue">
```

```
<Rectangle.RenderTransform>
```

```

<RotateTransform Angle="45" />
</Rectangle.RenderTransform>
</Rectangle>
</Grid>

```

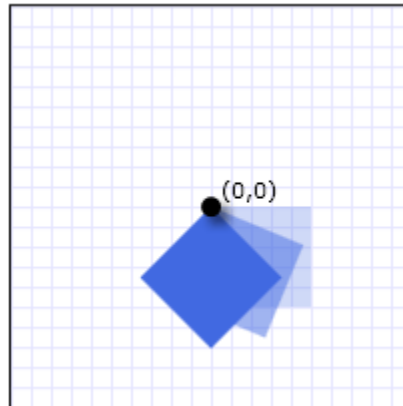


Figura 4. 3. Rectángulo girado 45 grados alrededor del punto (0,0)

Para aplicar varias transformaciones a una forma (o a cualquier otro elemento de la interfaz de usuario), utilice TransformGroup.

4.3.2. PINCELES

Puede utilizar los objetos de pincel de Silverlight para pintar con colores sólidos, degradados lineales, degradados radiales e imágenes.

4.3.2.1. Pintar un área con un color sólido

Una de las operaciones más comunes en cualquier plataforma es pintar un área con un color sólido. Para ello, Silverlight proporciona la clase SolidColorBrush.

Para pintar un área con un color sólido en XAML, use una de las opciones siguientes:

- Seleccionar un SolidColorBrush predefinido por nombre. Por ejemplo, puede establecer la propiedad Fill de un objeto Rectangle en "Red" o "MediumBlue". En el ejemplo se utiliza el nombre de un SolidColorBrush predefinido para establecer la propiedad Fill de un objeto Rectangle.

```
<StackPanel>
```

```

<!-- This rectangle's fill is painted with a red SolidColorBrush,
described using a named color. -->
<Rectangle Width="100" Height="100" Fill="Red" />
</StackPanel>

```

- Elegir un color de la paleta de colores de 32 bits especificando las cantidades de rojo, verde y azul que se van a combinar en un solo color sólido. El formato para especificar un color de la paleta de 32 bits es "#rrggbb", donde rr es un número hexadecimal de dos dígitos que especifica la cantidad relativa de rojo, gg especifica la cantidad de verde y bb, la cantidad de azul. Además, el color se puede especificar como #aarrggbb, donde aa especifica el valor alfa o transparencia del color. Este enfoque permite crear colores parcialmente transparentes. En el ejemplo siguiente, la propiedad Fill de Rectangle se establece en rojo totalmente opaco mediante la notación hexadecimal.

```

<StackPanel>
<!-- This rectangle's background is painted with a red SolidColorBrush,
described using hexadecimal notation. -->
<Rectangle Width="100" Height="100" Fill="#FFFFFF0000" />
</StackPanel>

```

- Usar la sintaxis de elementos de propiedad para describir SolidColorBrush. Esta sintaxis es más prolija pero permite especificar valores adicionales, como la opacidad del pincel. En el ejemplo siguiente, las propiedades Fill de dos elementos Rectangle se establecen en rojo totalmente opaco. El color del primer pincel se describe mediante un nombre de color predefinido y el del segundo, mediante la notación hexadecimal.

```

<StackPanel>
<!-- Both of these rectangles' fills are painted with red
SolidColorBrush objects, described using object element
syntax. -->

```



```

<Rectangle Width="100" Height="100">
  <Rectangle.Fill>
    <SolidColorBrush Color="Red" />
  </Rectangle.Fill>
</Rectangle>
<Rectangle Width="100" Height="100">
  <Rectangle.Fill>
    <SolidColorBrush Color="#FFFF0000" />
  </Rectangle.Fill>
</Rectangle>
</StackPanel>

```

4.3.2.2. Pintar un área con color degradado

Un pincel de degradado pinta un área con varios colores que se mezclan entre sí a lo largo de un eje. Puede utilizarlos para crear impresiones de luz y sombra, dando una sensación tridimensional a los elementos de la interfaz de usuario.

También puede utilizarlos para simular cristal, cromo, agua y otras superficies suavizadas. Silverlight proporciona dos tipos de pinceles de degradado: LinearGradientBrush y RadialGradientBrush.

4.3.2.3. Degradados lineales

LinearGradientBrush pinta un área con un degradado definido a lo largo de una línea, el eje de degradado. Especifique los colores del degradado y su ubicación a lo largo del eje de degradado mediante objetos GradientStop. También puede modificar el eje de degradado, lo que permite crear degradados horizontal y vertical e invertir la dirección del degradado.

En el siguiente ejemplo se crea un degradado lineal con cuatro colores y se utiliza para pintar un objeto Rectangle.

```

<StackPanel>
  <!-- This rectangle is painted with a diagonal linear gradient. -->

```

```

<Rectangle Width="200" Height="100">
  <Rectangle.Fill>
    <LinearGradientBrush StartPoint="0,0" EndPoint="1,1">
      <GradientStop Color="Yellow" Offset="0.0" />
      <GradientStop Color="Red" Offset="0.25" />
      <GradientStop Color="Blue" Offset="0.75" />
      <GradientStop Color="LimeGreen" Offset="1.0" />
    </LinearGradientBrush>
  </Rectangle.Fill>
</Rectangle>
</StackPanel>

```



Figura 4. 4. Degradado lineal diagonal

4.3.2.4. Degradados radiales

Como un LinearGradientBrush, un RadialGradientBrush pinta un área con colores que se mezclan juntos a lo largo de un eje. Un círculo define el eje de un pincel de degradado radial; sus colores irradian desde su origen.

En el ejemplo siguiente, se usa un pincel de degradado radial para pintar el interior de un rectángulo.

```

<StackPanel>
  <!-- This rectangle is painted with a radial gradient. -->
  <Rectangle Width="200" Height="100">
    <Rectangle.Fill>
      <RadialGradientBrush GradientOrigin="0.5,0.5" Center="0.5,0.5"
        RadiusX="0.5" RadiusY="0.5">
        <GradientStop Color="Yellow" Offset="0" />
        <GradientStop Color="Red" Offset="0.25" />
      </RadialGradientBrush>
    </Rectangle.Fill>
  </Rectangle>
</StackPanel>

```

```

<GradientStop Color="Blue" Offset="0.75" />
<GradientStop Color="LimeGreen" Offset="1" />
</RadialGradientBrush>
</Rectangle.Fill>
</Rectangle>
</StackPanel>

```

En la figura 4.5, se muestra el degradado creado en el ejemplo anterior. Se han resaltado los puntos de degradado del pincel.

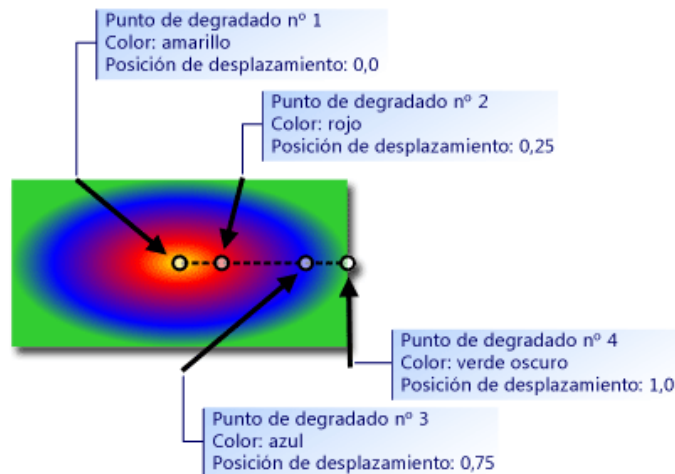


Figura 4.5 Puntos de degradado en un degradado radial

La propiedad `GradientOrigin` especifica el punto de inicio del eje de degradado de un pincel de degradado radial. El eje de degradado irradia desde el origen del degradado hacia el círculo del mismo. Al círculo del degradado de un pincel lo definen sus propiedades `Center`, `RadiusX` y `RadiusY`.

4.3.3. INFORMACIÓN GENERAL SOBRE OBJETOS VIDEOBRUSH

Un objeto `VideoBrush` es un tipo de objeto `Brush` similar a un `LinearGradientBrush` o `ImageBrush`. Sin embargo, en lugar de pintar un área con un degradado o una imagen, la pinta con contenido de vídeo. `MediaElement` proporciona este contenido de vídeo. Al igual que con los demás pinceles, puede utilizar `VideoBrush` para pintar el `Fill` de `Rectangle`, el `Background` de `Canvas` o el `Foreground` de `TextBlock`.

4.3.3.1. Propiedades de VideoBrush

Además de las propiedades que hereda por ser un tipo de Brush (como Opacity y RelativeTransform), VideoBrush proporciona las propiedades adicionales siguientes:

- SourceName : nombre del MediaElement que proporciona el vídeo del pincel.
- Stretch : uno de los valores siguientes que describe cómo el contenido de vídeo se ajusta para adaptarse al área pintada: None, Stretch, Uniform y UniformToFill. El valor predeterminado es Stretch.

4.3.3.2. Relación entre VideoBrush y MediaElement

El fotograma de vídeo actual del objeto VideoBrush viene determinado por su objeto MediaElement. En la figura 4.6., se muestra la relación entre MediaElement y VideoBrush.

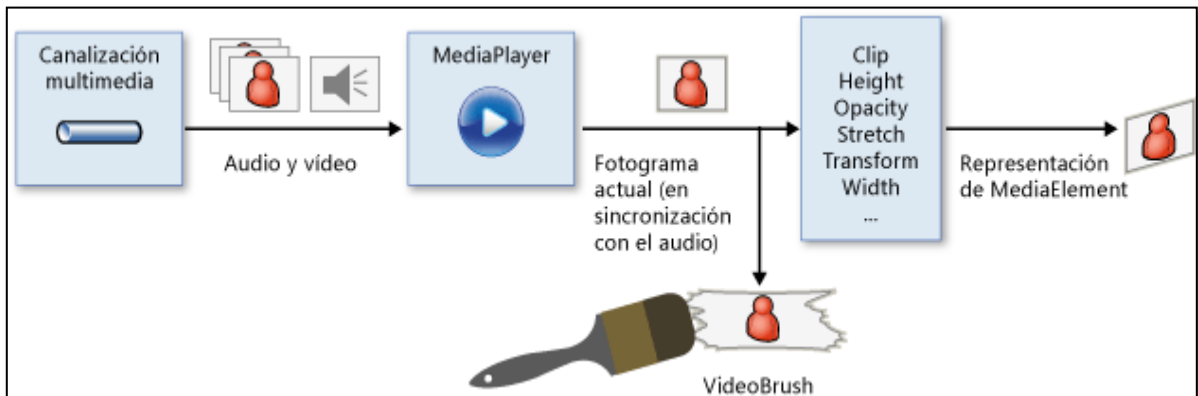


Figura 4. 6. VideoBrush y MediaElement

Cuando MediaElement carga contenido multimedia y VideoBrush utiliza MediaElement como su SourceName, el vídeo al que se hace referencia como Source de MediaElement se descarga y decodifica una sola vez. El rendimiento no resulta gravemente afectado cuando se utilizan MediaElement y VideoBrush a la vez, aunque decida utilizar MediaElement y VideoBrush para mostrar el vídeo.

4.3.4. GEOMETRÍAS

Los objetos Geometry, como EllipseGeometry, PathGeometry y GeometryGroup, permiten describir la geometría de una forma bidimensional (2D). Estas descripciones geométricas tienen varios usos, como definir una forma para pintar en la pantalla o definir regiones de recorte.

Los objetos Geometry pueden ser simples (como rectángulos y círculos) o compuestos (creados a partir de dos o más objetos Geometry). Se pueden crear geometrías más complejas utilizando objetos PathGeometry, que permiten describir arcos y curvas.

4.3.4.1. Comparación entre geometrías y formas

Las clases Geometry y Shape son similares porque ambas describen formas 2D (compare EllipseGeometry y Ellipse, por ejemplo), pero presentan diferencias importantes. Por ejemplo, los objetos Shape son objetos UIElement, pero no ocurre lo mismo con los objetos Geometry. Puesto que son objetos UIElement, los objetos Shape se pueden representar a sí mismos y tener propiedades Opacity, OpacityMask y otras propiedades gráficas inexistentes en los objetos Geometry.

Aunque los objetos Shape se pueden utilizar de un modo más directo que los objetos Geometry, los objetos Geometry son más versátiles.

Una clase Shape, la clase Path, utiliza un objeto Geometry para describir su contenido. Estableciendo la propiedad Data de Path con Geometry y estableciendo sus propiedades Fill y Stroke, puede representar Geometry.

4.3.4.2. Propiedades comunes de un objeto Geometry

En la tabla siguiente se muestran las propiedades que aceptan un objeto Geometry.

Tipo	Propiedad
Path	Data
UIElement	Clip

Tabla 4. 5. Propiedades comunes de un objeto Geometry

4.3.4.3. Geometrías de Trazado

El objeto PathGeometry y el minilenguaje de geometría proporcionan medios para describir varias figuras complejas compuestas de arcos, curvas y líneas.

En el núcleo de PathGeometry hay una colección de objetos PathFigure, que se denominan así porque cada figura describe una forma discreta de PathGeometry. Cada objeto PathFigure, a su vez, está compuesto de uno o varios objetos PathSegment, cada uno de los cuales describe un segmento de la figura. En la tabla 4.6 se muestran los tipos diferentes de segmentos.

Tipo de segmento	Descripción
ArcSegment	Crea un arco elíptico entre dos puntos.
BezierSegment	Crea una curva Bézier cúbica entre dos puntos.
LineSegment	Crea una línea entre dos puntos.
PolyBezierSegment	Crea una serie de curvas Bézier cúbicas.
PolyLineSegment	Crea una serie de líneas.
PolyQuadraticBezierSegment	Crea una serie de curvas Bézier cuadráticas.
QuadraticBezierSegment	Crea una curva Bézier cuadrática.

Tabla 4. 6. Tipos de segmentos de la geometría de trazados

Los segmentos de PathFigure se combinan en una sola forma geométrica donde el punto final de cada segmento se convierte en el punto inicial del segmento siguiente. La propiedad StartPoint de PathFigure especifica el punto desde el que se dibuja el primer segmento. Cada segmento posterior comienza en el punto final del segmento anterior

4.3.5. TRANSFORMACIONES

Puede utilizar las clases Transform bidimensionales (2D) de Silverlight para girar, ajustar a una escala, sesgar y mover (trasladar) objetos.

Un elemento Transform define cómo asignar, o transformar, puntos de un espacio de coordenadas a otro espacio de coordenadas. Esta asignación se describe

mediante una Matriz de transformación, que es una colección de tres filas con tres columnas de valores Double.

4.3.5.1. Clases de transformación

Silverlight proporcionar las siguientes clases Transform bidimensionales para las operaciones de transformación comunes como se describen en la tabla 4.7.

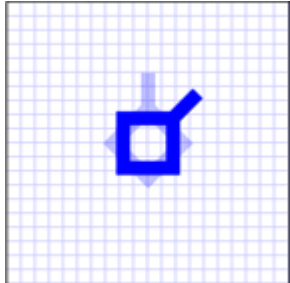
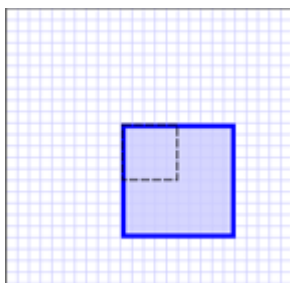
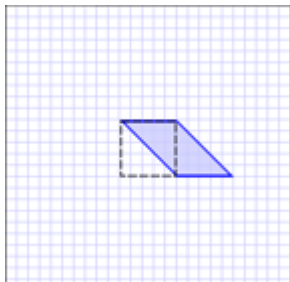
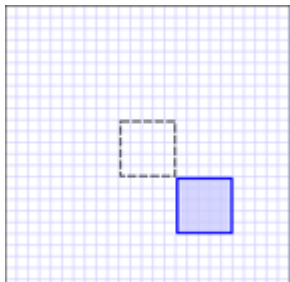
Clase	Descripción	Ilustración
RotateTransform	Gira un elemento el valor de Angle especificado.	
ScaleTransform	Escala un elemento usando los valores de ScaleX y ScaleY especificados.	
SkewTransform	Sesga un elemento los valores de AngleX y AngleY especificados.	
TranslateTransform	Mueve (traslada) un elemento los valores de X y Y especificados.	

Tabla 4. 7. Clases de Transformación

Para crear transformaciones más complejas, Silverlight proporciona las tres clases de la tabla 4.8.

Clase	Descripción
CompositeTransform	Use esta clase para aplicar varias transformaciones al mismo objeto (por ejemplo, sesgos y giros). Esta clase aplica varias transformaciones en un orden preferido y, por tanto, suele ser la mejor manera de aplicar varias transformaciones.
TransformGroup	Al igual que CompositeTransform, puede usar esta clase para aplicar varias transformaciones; sin embargo, la clase CompositeTransform es la mejor forma de hacerlo a menos que desee aplicar las transformaciones en un orden concreto o desee aplicar diferentes puntos centrales para las distintas transformaciones.
MatrixTransform	Crea transformaciones personalizadas que las otras clases Transform no proporcionan. Cuando se usa una clase MatrixTransform, se manipula una matriz directamente.

Tabla 4. 8. Clase de transformaciones más complejas

4.3.5.2. Propiedades comunes de transformación

Una manera de transformar un objeto consiste en declarar el tipo de Transform adecuado y aplicarlo a la propiedad de transformación del objeto. Tipos de objetos diferentes tienen distintos tipos de propiedades de transformación. En la tabla 4.9. se muestran varios tipos comunes de Silverlight y sus propiedades de transformación.

Tipo	Propiedades de transformación
Brush	Transform , RelativeTransform
Geometry	Transform
UIElement	RenderTransform

Tabla 4. 9. Propiedades de transformación

4.3.6. IMÁGENES

4.3.6.1. Ajustar una imagen

Al establecer el valor de Height y Width, se crea un área rectangular contenedora en la que se muestra la imagen. Mediante la propiedad Stretch, se puede especificar cómo la imagen rellena esta área contenedora. La propiedad Stretch acepta los valores siguientes, definidos por la enumeración Stretch:

- **None** : la imagen no se ajusta para rellenar las dimensiones de salida.
- **Uniform** : la imagen se escala para que se ajuste a las dimensiones de salida. Sin embargo, se conserva la relación de aspecto del contenido. Este es el valor predeterminado.
- **UniformToFill** : la imagen se escala para que rellene completamente el área de salida, pero conserva su relación de aspecto original.
- **Fill** : la imagen se escala para que se ajuste a las dimensiones de salida. Dado que la escala del alto y del ancho del contenido se ajusta de manera independiente, puede que no se conserve la relación de aspecto original de la imagen. Es decir, puede que se distorsione la imagen para que rellene completamente el área de salida.

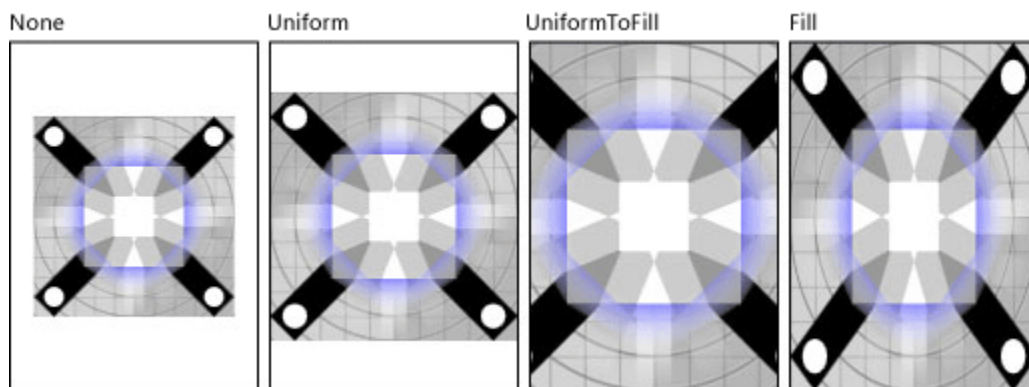


Tabla 4. 10. Valores de Stretch

En el ejemplo siguiente, se muestra un control Image que rellena un área de salida de 300 x 300 píxeles pero conserva su relación de aspecto original porque el valor de la propiedad Stretch está establecido en UniformToFill.

```
<Canvas Width="300" Height="300" Background="Gray">
  <Image Source="myImage.jpg" Stretch="UniformToFill" Width="300" Height="300" />
</Canvas>
```

4.3.6.2. Recortar una imagen

Mediante la propiedad Clip se puede recortar un área de la imagen. Al establecer la propiedad Clip en un objeto Geometry, se puede recortar una gran variedad de formas geométricas (por ejemplo, elipse, línea o un trazado complejo) de la imagen.

En el ejemplo siguiente se muestra cómo utilizar EllipseGeometry como zona de recorte de una imagen. En este ejemplo, se define un objeto Image con el valor de Width establecido en 200 y el valor de Height establecido en 150. La propiedad Clip de la imagen está establecida en un objeto EllipseGeometry con el valor de RadiusX establecido en 100, el valor de RadiusY establecido en 75 y el valor de Center establecido en 100,75. Únicamente se muestra la parte de la imagen que está dentro del área de la elipse.

```
<Grid x:Name="LayoutRoot" Background="White">
  <Image Source="Water_lilies.jpg"
```

```

Width="200" Height="150">
<Image.Clip>
  <EllipseGeometry RadiusX="100" RadiusY="75" Center="100,75"/>
</Image.Clip>
</Image>
</Grid>

```

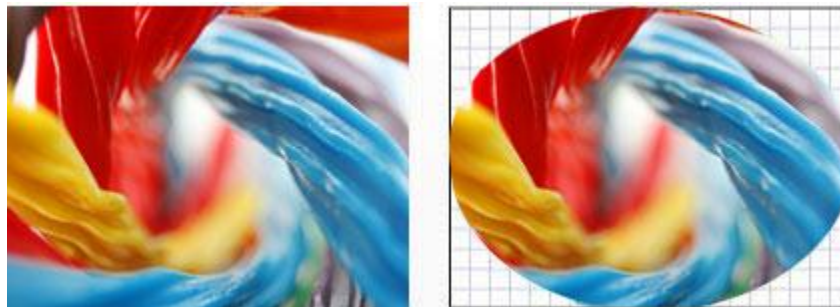


Figura 4. 7. Objeto EllipseGeometry utilizado para recortar una imagen

4.4. IMPRESIÓN

La clase PrintDocument proporciona la función de impresión para una aplicación de Silverlight. En este tema se describe cómo agregar funciones de impresión a sus aplicaciones de Silverlight

4.4.1. Mostrar cuadro de dialogo Imprimir

Para agregar funciones de impresión a su aplicación de Silverlight, agregue primero un objeto PrintDocument a la aplicación. Para mostrar un cuadro de diálogo de impresión, llame al método Print() de PrintDocument. Normalmente, llamará a Print() en el controlador de eventos de un botón que indica que se realizará una operación de impresión si se hace clic en el botón.

Todos los cuadros de diálogo de Silverlight deben estar iniciados por el usuario. Si se intenta mostrar el cuadro de diálogo de impresión cuando la operación no está iniciada por el usuario, se producirá una SecurityException. Por ejemplo, si intenta mostrar el cuadro de diálogo desde un controlador de eventos Loaded, se producirá una excepción de seguridad.

Dentro del cuadro de diálogo de impresión, el usuario puede seleccionar la configuración de impresora que desee y, a continuación, hacer clic en **Imprimir** para continuar con la operación de impresión o en **Cancelar** para cancelar la operación de impresión. En la figura 4.8., se muestra un ejemplo del cuadro de diálogo de impresión para Windows.

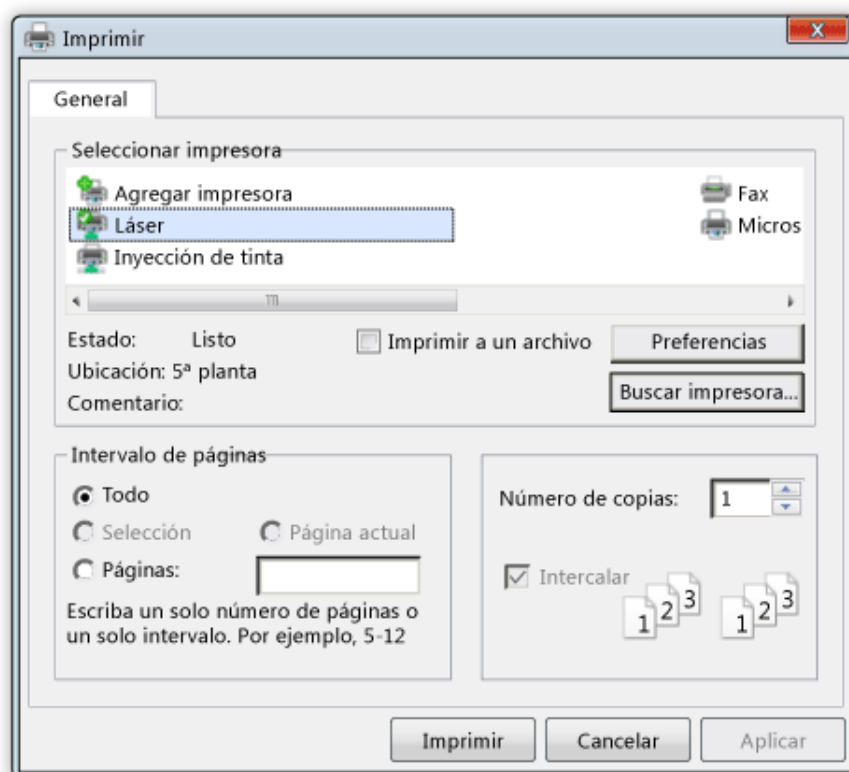


Figura 4. 8. Cuadro de diálogo de impresión para Windows

4.4.2. Realizar la operación de impresión

Para realizar la operación de impresión real y especificar el contenido que se va a imprimir, debe controlar el evento `PrintPage` para `PrintDocument`. En el controlador de eventos `PrintPage`, establezca la propiedad `PrintPageEventArgs.PageVisual` en el `UIElement` con nombre que desee imprimir. El elemento `UIElement` puede ser cualquier `UIElement` con nombre de la aplicación. El `UIElement` especificado no tiene que formar parte del árbol visual. Para imprimir todo el control de Silverlight, puede especificar el elemento `LayoutRoot`.

En el ejemplo de código siguiente se muestra un controlador de eventos de botón que llama al método Print() y un controlador de eventos para el evento PrintPage.

En este ejemplo, se imprime un control Image, que contiene un mapa.

```
public partial class MainPage : UserControl
{
    PrintDocument pd;

    public MainPage()
    {
        InitializeComponent();
        pd = new PrintDocument();
        pd.PrintPage += new EventHandler<PrintPageEventArgs>(pd_PrintPage);
    }

    private void PrintButton_Click(object sender, RoutedEventArgs e)
    {
        pd.Print("My Map");
    }

    void pd_PrintPage(object sender, PrintPageEventArgs e)
    {
        e.PageVisual = mapImage;
    }
}

<StackPanel x:Name="LayoutRoot">
    <Button Margin="5" Width="200" Content="Click to print" x:Name="PrintButton"
        Click="PrintButton_Click" />
    <Image Width="600" Height="600" Source="RedmondMap.jpg" x:Name="mapImage"/>
</StackPanel>
```

La propiedad `PrintPageEventArgs.HasMorePages` se usa para imprimir un documento con varias páginas. El valor predeterminado para `PrintPageEventArgs.HasMorePages` es `false`, por lo que no es necesario establecerla para documentos de una página. Sin embargo, si hay varias páginas para imprimir, establezca el valor de la propiedad `PrintPageEventArgs.HasMorePages` en `true` para indicar que deben imprimirse más páginas. Vuelva a establecer `PrintPageEventArgs.HasMorePages` en `false` en el controlador de eventos `PrintPage` cuando se esté imprimiendo la última página.

4.4.3. CONFIGURAR Y LIMPIAR UNA OPERACIÓN DE IMPRESIÓN

Si tiene una configuración especial que se deba realizar antes de que se produzca la operación de impresión, controle el evento `BeginPrint`. Por ejemplo, si desea configurar un contador de páginas para un documento que tiene varias páginas, puede hacerlo en el controlador de eventos `BeginPrint`.

Si hay que realizar alguna tarea de limpieza después de la operación de impresión, controle el evento `EndPrint`. También puede usar el controlador de eventos `EndPrint` para buscar si se produjeron errores en el proceso de impresión comprobando la propiedad `Error` de `EndPrintEventArgs`.

4.5. MICROSOFT EXPRESSION BLEND

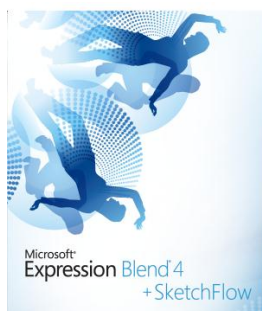


Figura 4. 9 Microsoft Expression Blend

Microsoft Expression Blend es una herramienta visual destinada al diseño y creación de prototipos de aplicaciones web y de escritorio. Puede generar una aplicación dibujando formas y controles como botones y cuadros de lista, haciendo que las distintas áreas de la aplicación respondan a clics del mouse y otras

acciones de usuario y aplicando estilos para lograr una apariencia totalmente personal²⁶.

También puede insertar imágenes, clips de audio y vídeo y personalizar controles de bibliotecas de SDK o terceros. Lo que ve en la superficie de diseño de Expression Blend es exactamente lo que verán los usuarios cuando ejecuten la aplicación.

4.5.1. Funcionalidad de Expression Blend

En Expression Blend, se puede diseñar la aplicación visualmente, se pueden dibujar formas, trazados y controles en la mesa de trabajo y, a continuación, se puede modificar su apariencia y comportamiento. Puede importar imágenes, vídeo y sonido. En las aplicaciones basadas en Windows, también puede importar y cambiar objetos 3D.

Puede crear guiones gráficos que animen los elementos visuales o de audio del diseño y, opcionalmente, activar esos guiones gráficos cuando los usuarios interactúan con la aplicación. Cuando trabaja en aplicaciones basadas en Windows o en Silverlight, puede rediseñar las plantillas que se aplican a controles básicos para que la aplicación tenga un aspecto y un comportamiento exclusivos.

Cuando trabaja en la aplicación, puede actualizar el proyecto en cualquier momento con los archivos de código subyacente o de control personalizado en los que están trabajando los programadores. Si usa Expression Blend, los diseñadores y programadores pueden trabajar en el mismo proyecto simultáneamente y sin molestarse.

4.5.2. Aplicaciones que funcionan con Expression Blend

Puede importar gráficos y recursos del Lenguaje de marcado de aplicaciones extensible (XAML) generados por Microsoft Expression Design en el proyecto de Expression Blend. Además, puede importar proyectos multimedia de Silverlight

²⁶ Acerca de Expression Blend, [http://msdn.microsoft.com/es-es/library/cc296376\(v=expression.30\).aspx](http://msdn.microsoft.com/es-es/library/cc296376(v=expression.30).aspx)

creados en Microsoft Expression Encoder para agregar nuevas características o elementos visuales al proyecto, o para modificar la plantilla del reproductor multimedia que se puede reutilizar en Expression Encoder.

En Microsoft Expression Web, puede importar sitios web de Silverlight y archivos de una aplicación Silverlight compilados en un proyecto nuevo o existente y, a continuación, publicar el trabajo.

Microsoft Visual Studio 2010 funciona perfectamente con Expression Blend para mantener la sincronización al modificar archivos de manera simultánea en Visual Studio 2010 y Expression Blend. En Expression Blend, puede abrir archivos de código subyacente individuales o todo el proyecto en Visual Studio 2010. Asimismo, puede usar las herramientas de implementación de Visual Studio 2010 para implementar las aplicaciones.

Expression Blend genera aplicaciones de Windows Presentation Foundation (WPF), sitios web de Silverlight (.xap y archivos auxiliares) y controles de usuario de Silverlight. El diseño visual se representa mediante XAML. Al igual que HTML es el lenguaje de marcado para las aplicaciones web, XAML es el lenguaje de marcado para WPF.

4.5.3. Características de Expression Blend

Expression Blend incluye lo siguiente:

- SketchFlow, un conjunto de características para crear prototipos que son aplicaciones reales de WPF o Silverlight, se incluye en Expression Studio Ultimate.
- Expression Blend para Windows Phone, disponible al descargar las Windows Phone Developer Tools (Herramientas de desarrollo de Windows Phone).
- Un conjunto completo de herramientas de dibujo vectorial, que incluye herramientas tridimensionales (3D) y de texto.
- Una interfaz visual moderna y fácil de usar con paneles acoplables y menús contextuales en objetos.

- Animación en tiempo real.
- Compatibilidad con elementos en 3D y multimedia para mejorar las experiencias de los usuarios.
- Compatibilidad con efectos y transiciones para mejorar las experiencias de los usuarios.
- Plantillas de proyecto para **Views** y **ViewModels** .
- Opciones de máscara y personalización avanzada, flexible y reutilizable para diversos controles comunes.
- Eficaces puntos de integración de orígenes de datos y recursos externos.
- Vistas de marcado y diseño en tiempo real.
- Capacidades de importación de material gráfico de Expression Design.
- Capacidades de importación de sitios de Expression Encoder.
- Interoperabilidad con Visual Studio 2010 para ayudar a los diseñadores y programadores a colaborar de forma más estrecha y eficaz como un equipo.

4.5.4. Tipos de Aplicaciones

Expression Blend se ha optimizado para generar los siguientes tipos de aplicaciones:

- **Aplicaciones de productividad** Aplicaciones que aumentan la productividad y la eficacia para una base de clientes más amplia, así como aplicaciones de línea de negocio, como Microsoft Office.
- **Aplicaciones de Windows Phone** Aplicaciones diseñadas para ejecutarse en Windows Phone.
- **Aplicaciones para consumidores** Aplicaciones como reproductores multimedia, herramientas de seguridad y gadgets de escritorio.
- **Juegos** Sencillos juegos de escritorio o en línea diseñados exclusivamente para el entretenimiento.

- **Quioscos multimedia** Aplicaciones diseñadas para ejecutarse en quioscos multimedia con los que los usuarios pueden interactuar para obtener información, revisar directorios de productos, facturar en un aeropuerto, etc.
- **Utilidades para profesionales de TI** Herramientas para pequeñas tareas, como el seguimiento de errores, que pueden ser únicas para las necesidades específicas de una empresa o de un cliente.

4.5.5. Procedimientos recomendados

Algunas ideas de diseño son buenas sólo porque mejoran el uso. A continuación se indican algunas formas de mejorar las posibilidades de uso de Expression Blend y Microsoft .NET Framework:

- **Crear modelos del mundo real** Puede usar interacciones y elementos visuales personalizados para crear controles específicos cuya apariencia y comportamiento sean similares a sus equivalentes en el mundo real. Esta técnica es más adecuada cuando los usuarios están familiarizados con el objeto del mundo real y este enfoque del mundo real es la forma más conveniente y eficaz de realizar una tarea. Por ejemplo, utilidades sencillas como las calculadoras funcionan mejor cuando usan como modelo sus equivalentes en el mundo real.
- **Mostrar en lugar de explicar** Puede usar animaciones y transiciones para mostrar relaciones, causas y efectos. Esta técnica es ideal para proporcionar información que, en caso contrario, requeriría texto que explique lo que podrían perderse los usuarios. Por ejemplo, un libro para niños podría realizar una animación del paso de las páginas para mostrar cómo funcionan los controles.
- **Mejorar la captación intuitiva** Captación intuitiva es una propiedad de un objeto que sugiere cómo se usa el objeto (en lugar de usar una etiqueta

para explicarlo). Puede usar animaciones y elementos visuales de control personalizados para sugerir cómo se usan controles no estándar.

- **Usar la asignación natural** La asignación natural es una relación clara entre lo que el usuario desea hacer y cómo hacerlo. Puede usar interacciones y apariencias personalizadas para crear asignaciones naturales cuando los controles comunes estándar no funcionan.
- **Reducir el conocimiento necesario** Puede usar interacciones personalizadas para limitar el número de formas de realizar una operación, así como el conocimiento necesario para realizar una tarea.
- **Mejorar los comentarios** Puede usar animaciones y elementos visuales de control personalizados para proporcionar comentarios que muestren que el usuario ha hecho algo correcta o incorrectamente, o bien para mostrar el progreso. Por ejemplo, en la barra de direcciones de Internet Explorer en Windows Vista y Windows 7, se muestra el progreso de carga de la página en segundo plano.
- **Facilitar la interacción con los objetos** Un modelo de movimiento humano conocido como ley de Fitts indica que el esfuerzo necesario para hacer clic en un objetivo es proporcional a su distancia e inversamente proporcional a su tamaño. Por ejemplo, puede usar animaciones para hacer que los objetos sean más grandes cuando el puntero esté cerca y más pequeños cuando el puntero esté lejos. Esto hará que sea más fácil hacer clic en el objeto. Además, le permitirá usar el espacio en pantalla de forma más eficaz al hacer que los objetos sean más pequeños que lo habitual.
- **Foco** Puede usar un diseño complejo y elementos visuales personalizados para destacar elementos de la pantalla que son necesarios para la tarea y quitar importancia a los elementos secundarios.

CAPÍTULO V.- APLICATIVO

5.1. ALCANCE

El presente prototipo tiene la finalidad de demostrar la tecnología de Microsoft Silverlight para el desarrollo de Aplicaciones RIA (Rich Internet Application).

El aplicativo se encuentra dividido en menús que son los siguientes:

- Administración
- Docentes
- Alumnos
- Reportes

Al ingresar un usuario en función del rol con el que accede al sitio web, se pueden identificar los siguientes grupos de pantallas:

- Opciones de Administradores
- Opciones de Docentes
- Opciones de Alumnos
- Opciones de Invitados

5.1.1. Opciones de Administrador

En el grupo de páginas de Administración cuando el usuario accede con el rol “Administrador” se muestran las siguientes pantallas.

- Menú Administrar
 - Personas

Muestra las opciones necesarias para administrar los registros de las personas que acceden al sitio web.

- Publicar cursos

Sirve para publicar cursos propuestos por los docentes. Luego de que un curso está publicado un alumno puede acceder a ellos.

- Menú reportes
 - Cursos por docentes

Muestra un resumen con los resultados de los cursos por docente.

Las restricciones para un usuario que ingresa al sitio web con un rol de Administrador son:

- No puede ingresar cursos ni su contenido.
- No puede exponer foros de los cursos.
- No puede aplicar pruebas.

5.1.2. Opciones de Docentes

En el grupo de páginas de Docente cuando el usuario accede con el rol “Docente” se muestran las siguientes pantallas.

- Menú Docentes
 - Cursos

Pantalla que muestra las opciones necesarias para administrar los registros de los cursos que se pueden exponer por un docente.

- Crear pruebas

Luego de ingresar un curso, un docente puede ingresar pruebas de evaluación con respuestas para escoger con un mínimo de 2 por pregunta, es decir, que una pregunta debe tener por los menos opciones de respuesta de SI o NO.

- Foros

Sirve para exponer foros de un curso que un docente esté impartiendo.

- Menú reportes
 - Cursos por alumnos

Muestra un resumen con los resultados de las pruebas aplicadas por curso.

Las restricciones para un usuario que ingresa al sitio web con un rol de Docente son:

- No puede aplicar pruebas, es decir, un docente puede crear pruebas pero mientras ingrese al sitio web con rol Docente no puede aplicar a ninguna prueba.

- No puede administrar la información de los registros de las personas que se inscriben a los cursos.

5.1.3. Opciones de Alumnos

En el grupo de páginas de Alumnos cuando un usuario accede con el rol “Alumno” se muestran las siguientes pantallas.

- Menú Alumnos
 - Contenido de cursos

Un alumno luego de inscribirse a un curso puede acceder al contenido del mismo en esta pantalla, en caso que el contenido tenga una extensión que no pueda mostrarse directamente en el browser se dispone para su descarga local.

- Aplicar pruebas

Sirve para acceder a las pruebas que un docente publica, para un determinado curso, las pruebas pueden servir o no para la aprobación final del curso.

- Foros

Un foro expuesto puede ser comentado por medio de esta pantalla, debe escoger el curso y el foro que desea comentar.

- Menú reportes
 - Pruebas

Muestra un resumen con los resultados de las pruebas aplicadas por curso.

Las restricciones para un usuario que ingresa al sitio web con un rol de Alumno son:

- No puede administrar los registros de los usuarios.
- No puede ingresar cursos ni su contenido.
- No puede exponer foros
- No puede visualizar resultados de pruebas de otros alumnos.

5.1.4. Opciones de Invitados

Las opciones de usuario con rol de Invitado o con cualquier otro rol puede visualizar son:

- Pantalla “HOME”

Contiene los cursos que se exponen en el sitio web, haciendo clic en el botón correspondiente a cada curso expuesto se puede visualizar el temario del curso.

- Pantalla “Iniciar sesión”

Sirve para que un usuario pueda acceder con un determinado rol al sitio web, en caso de que el usuario inicie sesión y cuente con varios roles puede escoger con cual desea ingresar al sitio web. También puede registrarse en el sitio web por medio de esta pantalla.

5.2. PLATAFORMA Y ARQUITECTURA

5.2.1. DISEÑO DE LA ARQUITECTURA

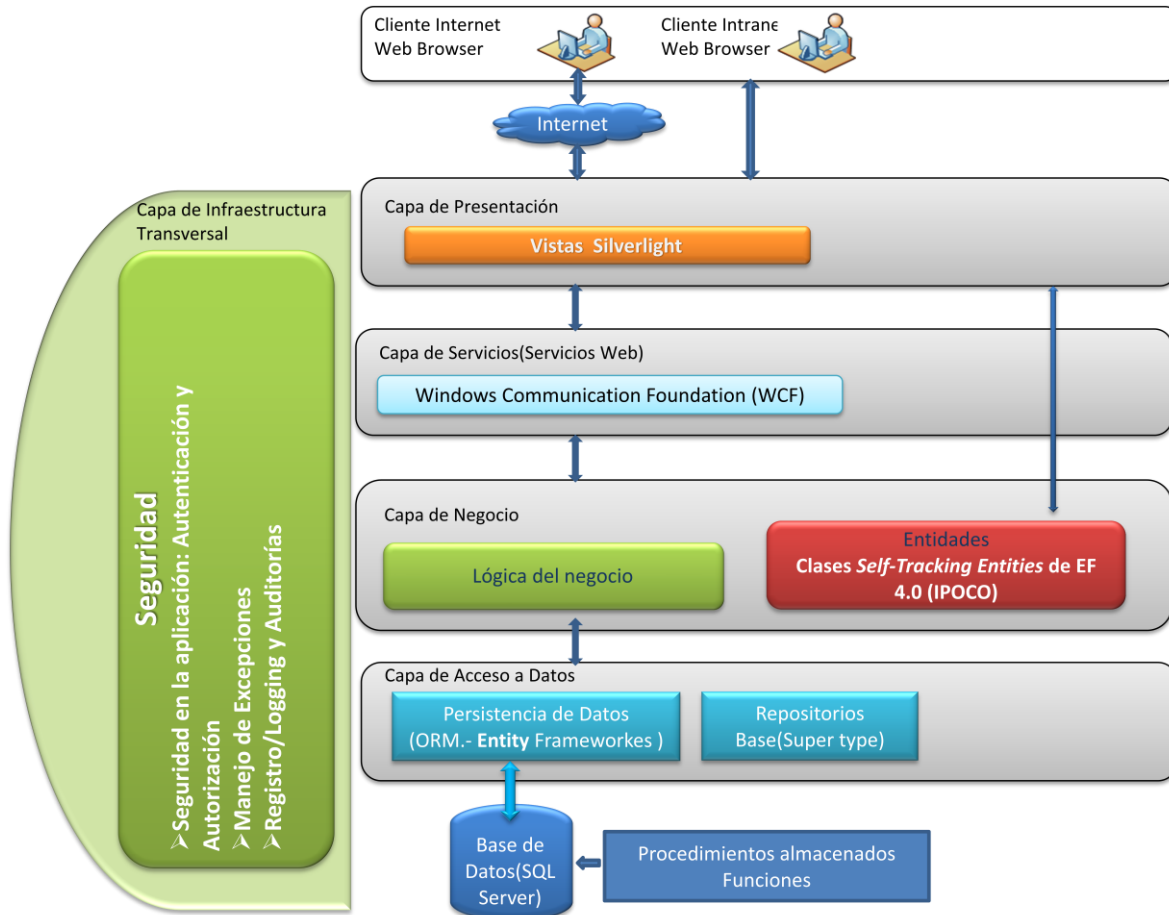


Figura 5. 1 Diseño de la arquitectura

La arquitectura propuesta para la implementación del prototipo del aplicativo se encuentra distribuida en las siguientes capas:

- Capa de Acceso a Datos
- Capa de Negocio
- Capa Servicios(Servicios Web)
- Capa de Presentación
- Capa de Infraestructura Transversal

5.2.1.1. Capa de Acceso a Datos

Los componentes de persistencia de datos proporcionan acceso a datos que están hospedados dentro de las fronteras de nuestro sistema (la base de datos principal).

Esta capa contiene los siguientes elementos:

➤ Repositorios

Son clases/componentes que encapsulan la lógica requerida para acceder a las fuentes de datos de la aplicación. Centralizan por lo tanto funcionalidad común de acceso a datos de forma que la aplicación pueda disponer de un mejor mantenimiento y desacoplamiento entre la tecnología y la lógica de las capas.

Si se hace uso de tecnologías base tipo O/RM (*Object/Relational Mapping frameworks*), se simplifica mucho el código a implementar y el desarrollo se puede focalizar exclusivamente en los accesos a datos y no tanto en la tecnología de acceso a datos (conexiones a bases de datos, sentencias SQL, etc.) que se hace mucho más transparente en un O/RM.

➤ Persistencia de Datos (ORM.- *Entity Framework*)

(El nombre completo es *ADO.NET Entity Framework "EF"*, puesto que está basado sobre la plataforma de ADO.NET). Se crea un modelo de datos mapeado a una base de datos relacional. A nivel superior se mapeará normalmente una clase a múltiples tablas que conformen una entidad compleja. La gran ventaja de EF es que hace transparente la base de datos con la que trabaja, pues el modelo de EF genera las sentencias SQL nativas requeridas para cada SGBD²⁷, así pues, en la mayoría de los casos sería transparente si se está trabajando contra SQL Server, Oracle, DB2, MySql, etc. Solo es necesario hacer uso en cada caso del proveedor de EF relativo a cada SGBD. EF es apropiado cuando se quiere hacer uso de un modelo de desarrollo ORM basado en un modelo de objetos (p.e. el de *Linq to Entities*) mapeado a un modelo relacional mediante un esquema flexible. Si se hace uso de EF, normalmente se hará uso también de:

²⁷ SGBD Sistema de gestión de base de datos, es una agrupación de programas que sirven para definir, construir y manipular una base de datos.

➤ **LINQ to Entities**

Considera “*LINQ to Entities*” si se desea ejecutar consultas contra entidades fuertemente tipadas y haciendo uso del modelo orientado a objetos de la sintaxis de LINQ.

➤ **Modelo de datos.-**

Este concepto sirve para poder definir e, incluso visualizar gráficamente el modelo de datos “entidad-relación” de la aplicación. Este concepto suele ser proporcionado completamente por la tecnología O/RM concreta que se utilice, por lo que está completamente ligado a una infraestructura/tecnología específica (*Entity Framework* proporciona una forma de definir un modelo entidad-relación o incluso de crearlo a partir de una base de datos existente).

5.2.1.2. Capa de Negocio

Esta capa debe ser responsable de representar conceptos de negocio, información sobre la situación de los procesos de negocio e implementación de las reglas del dominio. También debe contener los estados que reflejan la situación de los procesos de negocio, aun cuando los detalles técnicos de almacenamiento se delegan a las capas inferiores de infraestructura (Repositorios, etc.)

Esta capa contiene los siguientes elementos:

Entidades.- Las entidades se usan para contener y gestionar los datos principales de la aplicación. En definitiva, las entidades del dominio son clases que contienen valores y los exponen mediante propiedades, pero también pueden y deben exponer métodos con lógica de negocio de la propia entidad.

➤ **Clases *Self-Tracking Entities* de EF 4.0 (IPOCO)**

“El concepto de clases IPOCO es prácticamente el mismo que el de clases POCO (Significa “*Plain Old Clr Objects*”, es decir, que se implementan las entidades que simplemente son clases sencillas de .NET, con variables miembro y propiedades para los atributos de la entidad”²⁸. Esto puede hacerse manualmente o bien con la

²⁸ Guia_Arquitectura_N-Capas_DDD_NET_4.

ayuda de generación de código de frameworks O/RM, como Entity Framework (EF) o NHibernate, que nos generen estas clases de forma automática, ahorrando mucho tiempo de desarrollo manual para sincronizarlo con el modelo entidad relación que se este usando. La regla más importante de las clases POCO es que no deben tener dependencia alguna con otros componentes y/o clases).

La única diferencia radica en que en las entidades IPOCO se permite implementar interfaces concretos para aspectos que sean necesarios. Por ejemplo, las clases “*Self-Tracking*” de EF 4.0 (para poder realizar gestión de Concurrencia Optimista), son clases IPOCO, porque aunque son clases con código independiente, código de nuestro proyecto, sin embargo implementan un interfaz (o varios) requeridos por el sistema “*Self-Tracking*” de EF 4.0. Concretamente, se implementan los interfaces **IObjectWithChangeTracker** y **INotifyPropertyChanged**. Lo importante es que los interfaces que se implementen sean propios (código nuestro, como **IObjectWithChangeTracker** que es generado por las plantillas T4) o interfaces estándar de .NET Framework (como **INotifyPropertyChanged** que forma parte de .NET). Lo que no sería bueno es que se implementara un interfaz perteneciente a las propias librerías de Entity Framework o de otro O/RM, porque en este último caso se tendría una dependencia directa con una tecnología y versión concreta de framework de persistencia de datos.

5.2.1.3. Capa de Servicios(Servicios Web)

En esta capa tiene una Arquitectura Orientada a Servicios, que se solapa en gran medida con SOA (*Service Oriented Architecture*). La virtud de SOA es precisamente el poder compartir ciertos Servicios/Aplicaciones y dar acceso a ellos de una forma estándar, pudiendo realizar integraciones de una forma interoperable que hace años eran costosísimas.

La Capa de Servicios normalmente incluye lo siguiente:

➤ Interfaces/Contratos de Servicios

Los Servicios exponen un interfaz de servicio al que se envían los mensajes de entrada. En definitiva, los servicios son como una fachada que expone la lógica de

aplicación y del dominio a los consumidores potenciales, bien sea la Capa de Presentación o bien sean otros Servicios/Aplicaciones remotas.

➤ **Mensaje de Tipos**

Para intercambiar datos a través de la capa de Servicios, es necesario hacerlo mediante mensajes que envuelven a estructuras de datos. La capa de servicio también incluirá tipos de datos y contratos que definan los tipos de datos utilizados en los mensajes.

5.2.1.4. Capa de Presentación

Esta capa se desarrollo con Silverlight 4.

La responsabilidades de esta capa son las de presentar al usuario los conceptos de negocio mediante una interfaz de usuario (IU), facilitar la explotación de dichos procesos, informar sobre el estado de los mismos e implementar las reglas de validación de dicha interfaz.

Se debe fijar en un detalle:

Por una parte se define el manejador de eventos desde el propio XAML y se mezclan dos roles. El diseñador, que genera XAML de forma visual con Blend.

Por otra parte, se crea una fuerte dependencia del XAML con la programación de la funcionalidad. Si esta funcionalidad se reutiliza en otros XAMLs se tendrá que sacar a una clase distinta.

5.2.1.5. Capa de Infraestructura Transversal

Actúa sobre cada una de las capas verticales y abarca operaciones como autenticación, autorización, manejo de excepciones, *logging*/registros:

➤ **Seguridad en la aplicación: Autenticación y Autorización**

La **Identidad** y **Autenticación** para identificar a los usuarios que acceden a la aplicación así como el concepto de **Autorización**, para comprobar y en su caso otorgar acceso a los usuarios sobre los recursos (áreas funcionales) de la aplicación.

➤ Manejo de Excepciones

Permitir que las excepciones suban hacia las capas superiores hasta llegar a las capas frontera (como Servicios-Web o Capa de Presentación Web ASP.NET), donde dichas excepciones serán registradas (*logs*) y transformadas según sea necesario antes de pasarlas a la siguiente capa antes de que lleguen a la capa de presentación o interfaz gráfico de usuario.

➤ Registro/Logging y Auditorías

La aplicación puede ser vulnerable a amenazas de repudio, donde los usuarios niegan sus acciones y los ficheros/registros de log pueden requerirse para procedimientos legales que pretendan probar sus acciones. Se debe auditar y registrar la actividad de la aplicación en las diferentes capas en puntos clave que puedan ayudar a detectar actividades sospechosas y proporcionar pronto indicaciones de ataques serios.

Como aplicativo de la presente investigación se implementó prototipo de software el cual no va a ser aplicado en ninguna empresa, razón por la cual no se desarrolló un registro de Loggin de auditoría. Se recomienda en sistemas con media o alta transaccionalidad desarrollar un registro de Loggin de auditoría de la siguiente manera:

Crear una tabla que contenga los siguientes datos:

Id	Campo numérico autoincrementar Identity(1,1)
Id_usuario	Identificador de usuario
Tipo_usuario	Administrador, Docente, Alumno o Cliente
Fecha_Ing	Fecha y hora de conexión
Fecha_Sal	Fecha y hora de desconexión
IP_Terminal	Dirección IP de la PC
Id_evento	Índice referencial, registro de eventos realizados por el usuario
Modulo	Módulo al que accedió el usuario
Opcion_Menu	Menú u submenú a la accedió el usuario

Tabla 5. 1 Registro de auditoría

La tabla Registro de Auditoria deberá incluir un campo clave autonumérico, el cual nos permita hacer pruebas de auditoría de detección de faltantes y repetidos. Así también el campo Evento deberá desagregarse (normalizar) y crear tablas específicas para almacenar los tipos de eventos a las cuales acceden los usuarios.

Id_evento	Identificados del evento , campo numérico autoincrementar Identity(1,1)
Descripción	Eventos ocurridos

Tabla 5. 2 Evento

El contenido de la tabla Evento podría ser el siguiente:

Id_evento	Descripción
1	Acceso exitoso
2	Intento de acceso fallido
3	Error en programación
4	Cambio de configuración
5	Ejecuta reversión de transacciones
6	Creación de registro

Tabla 5. 3 Contenido tabla "Evento"

Se tiene 2 alternativa. La primera alternativa es llenar las tablas mediante el aplicativo, esto significa mucho esfuerzo en programación, especialmente cuando existen cambios en la Base de Datos. La segunda alternativa es llenar las tablas mediante el diseño de triggers, esta opción tiene la ventaja de ser independiente del aplicativo y las tablas se llenarán ya sea cuando se haga modificaciones directamente mediante sentencias SQL o mediante opciones de menú o comandos del aplicativo.

5.3. Análisis y diseño

5.3.1. Herramientas utilizadas

5.3.1.1. Visual Studio 2010

Visual Studio es un conjunto de herramientas de desarrollo basadas en componentes y otras tecnologías para compilar aplicaciones eficaces de alto rendimiento. Además, Visual Studio está optimizado para el diseño, el desarrollo y la implementación en equipo de soluciones empresariales²⁹.

5.3.1.1.1. Entorno de desarrollo integrado (IDE)

La gama de productos de Visual Studio comparte un único entorno de desarrollo integrado (IDE) que se compone de varios elementos: la barra de menús, la barra de herramientas Estándar, varias ventanas de herramientas que se acoplan u ocultan automáticamente a la izquierda, en la parte inferior y a la derecha, así como en el espacio del editor. Las ventanas de herramientas, menús y barras de herramientas disponibles dependen del tipo de proyecto o archivo en el que esté trabajando.

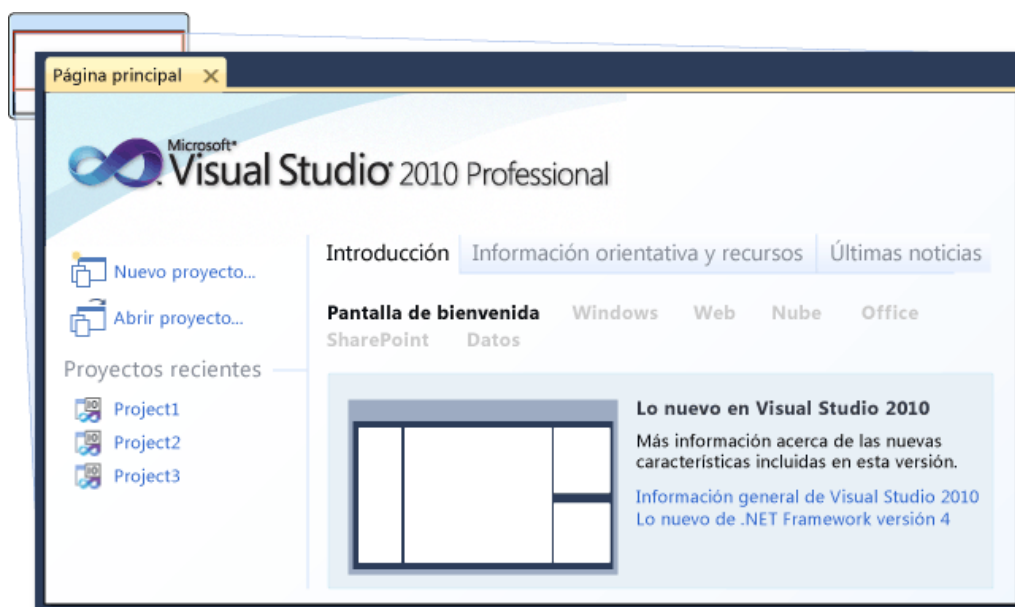


Figura 5. 2 Entorno de desarrollo integrado (IDE)

²⁹ Visual Studio, <http://msdn.microsoft.com/es-es/library/52f3sw5c.aspx>

Dependiendo de la configuración aplicada y de las subsiguientes personalizaciones que haya realizado, variará la colocación de las ventanas de herramientas y de otros elementos en el IDE. Puede cambiar la configuración mediante el Import and Export Settings Wizard. Al seleccionar la opción **Restablecer todas las configuraciones**, se puede cambiar el lenguaje de programación predeterminado.

5.3.1.1.1. Sistema de proyectos

Las soluciones y los proyectos contienen elementos en forma de referencias, conexiones de datos, carpetas y archivos necesarios para crear la aplicación. Un contenedor de tipo solución puede contener varios proyectos y un contenedor de tipo proyecto normalmente contiene varios elementos.

El Solution Explorer muestra soluciones, sus proyectos y los elementos incluidos en dichos proyectos. En el Explorador de soluciones, puede abrir archivos para editar, agregar nuevos archivos a un proyecto y ver las propiedades de las soluciones, proyectos y elementos.

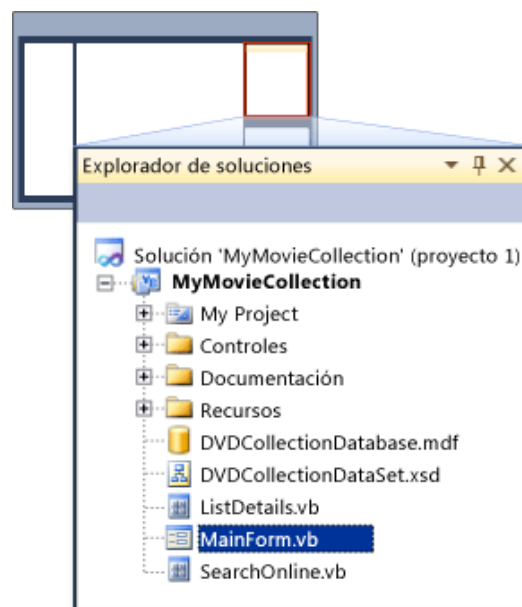


Figura 5. 3 Sistema de proyectos

5.3.1.1.1. Editores y diseñadores

El editor y los diseñadores que se utilice dependerán del tipo de archivo o documento que esté creando. El Editor de texto es el procesador de textos básico del IDE, mientras que el Editor de código es el editor de código fuente básico.

Otros editores, como el Editor CSS, el Diseñador HTML y el Diseñador de páginas Web, comparten muchas de las características del Editor de código, junto con mejoras específicas en el tipo de código o de marcado admitido.

Los editores y diseñadores normalmente tienen dos vistas: una vista de diseño gráfica y la vista de código subyacente o vista de código fuente. La vista de diseño le permite especificar la ubicación de los controles y otros elementos en la interfaz de usuario o la página web. Puede arrastrar controles desde el cuadro de herramientas y colocarlos en la superficie de diseño.

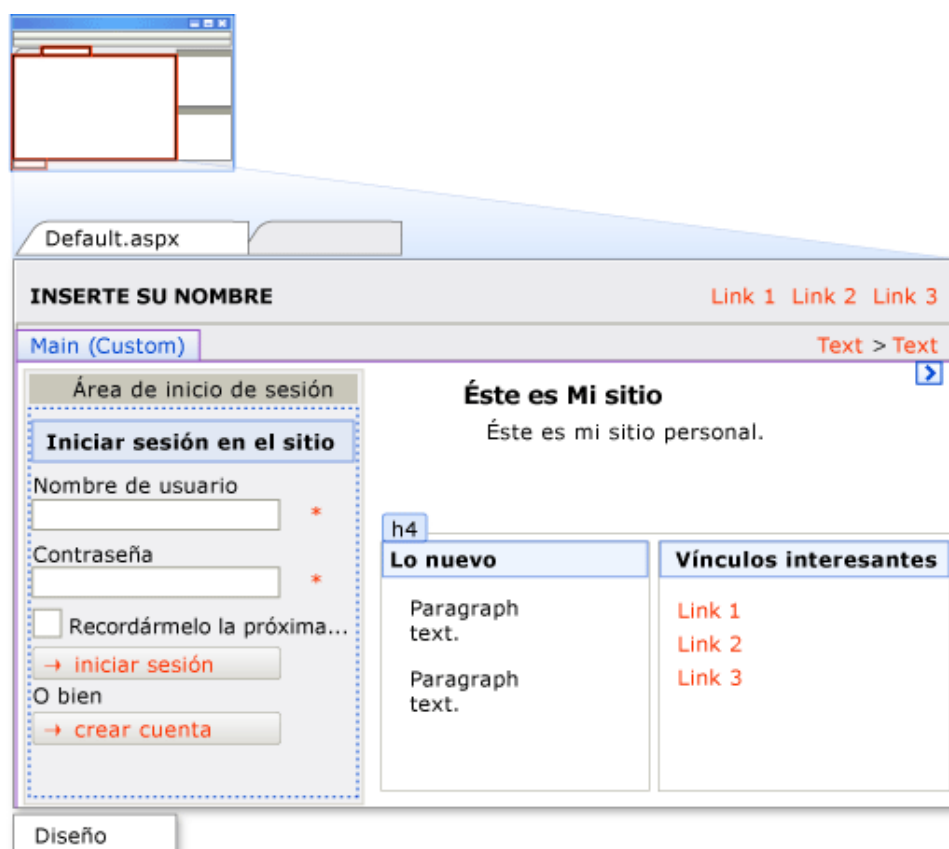


Figura 5. 4 Diseñador de páginas Web, vista Diseño

La vista Código fuente muestra el código fuente del archivo o documento. Esta vista admite ayudas de codificación como IntelliSense, secciones de código plegables, Refactorización (C#) e inserción de fragmentos de código. Otras características incluyen el ajuste automático de línea, los marcadores y la visualización de números de línea, por citar algunos.

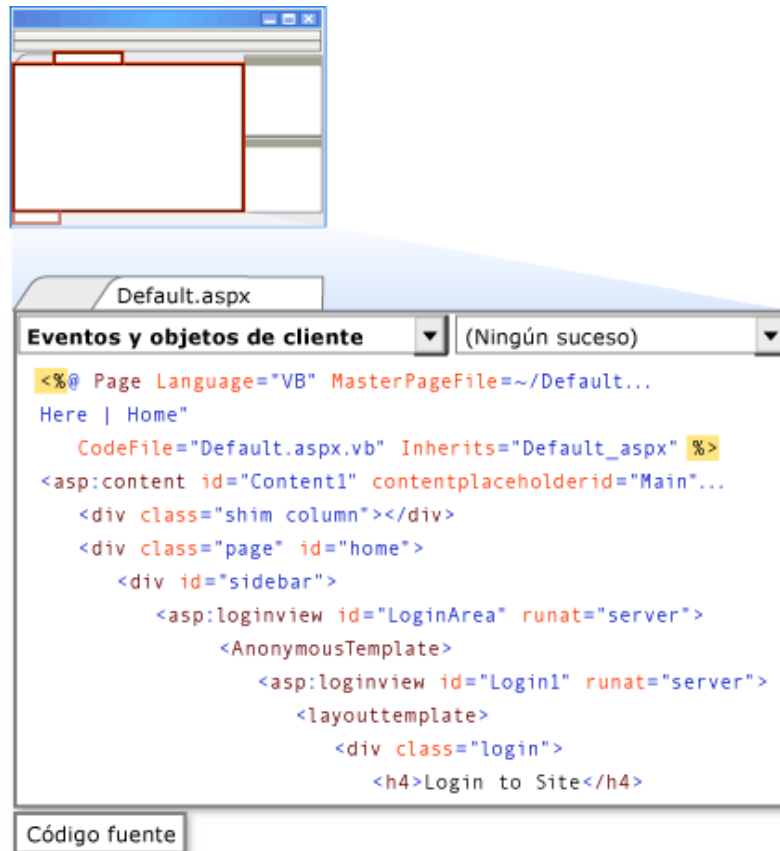


Figura 5. 5 Diseñador de páginas Web, vista Código fuente

Algunos editores, como el Diseñador de páginas web y el Diseñador XAML, también proporcionan una vista híbrida que le permite ver la vista del gráfico y del código de un archivo simultáneamente. Esta vista se llama la Vista dividida.

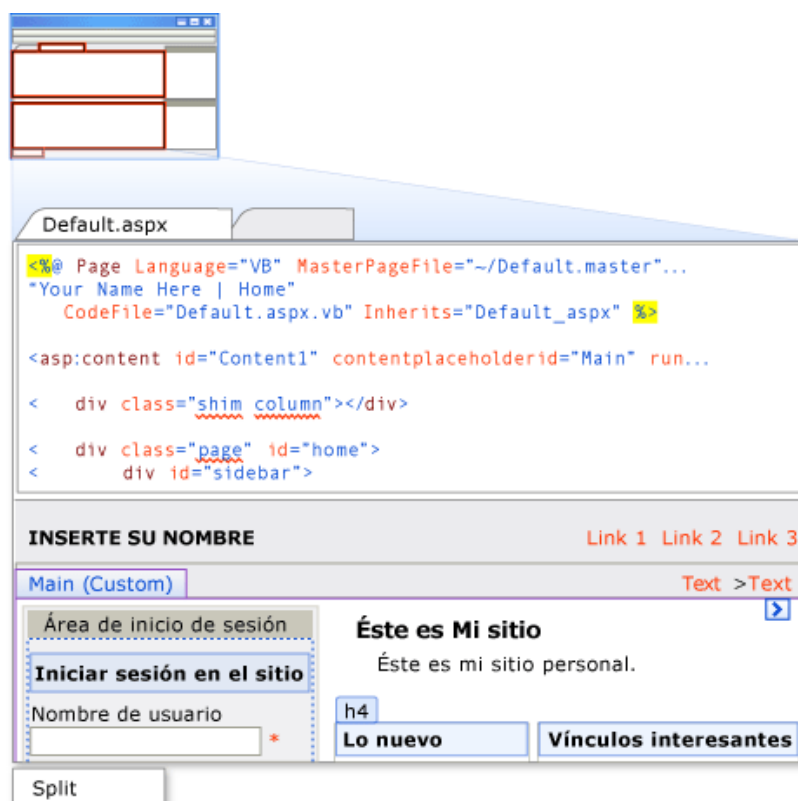


Figura 5. 6 Diseñador de páginas web y el Diseñador XAML

5.3.1.1.2. Herramientas de compilación y depuración

Visual Studio proporciona un sólido conjunto de herramientas de compilación y depuración. Con las configuraciones de compilación puede seleccionar los componentes que se van a generar, excluir los que no se van a generar y determinar cómo se van a generar los proyectos seleccionados y en qué plataforma. Puede tener configuraciones de compilación para soluciones y para proyectos.

Cuando genera, está comenzando el proceso de depuración. La compilación de la aplicación le ayuda a detectar errores de compilación. Estos errores pueden deberse a una sintaxis incorrecta, a palabras clave mal escritas o a divergencias entre los tipos. La Resultados (Ventana) muestra estos tipos de errores.

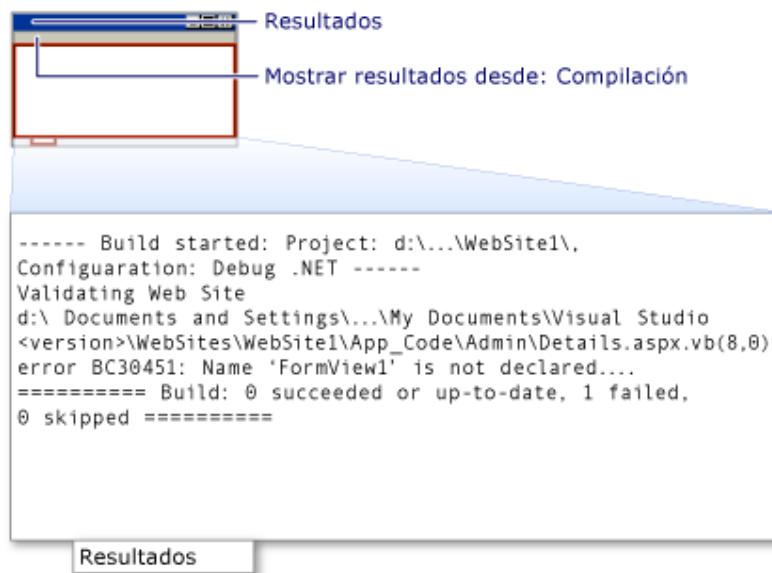


Figura 5.7 Ventana de salida con información de compilación

Después de generar la aplicación, puede utilizar el depurador para detectar y corregir problemas como errores lógicos y semánticos que se descubren en tiempo de ejecución. En el modo de interrupción, puede examinar las variables locales y otros datos pertinentes utilizando herramientas como Ventanas de variables y Memoria (Ventana).

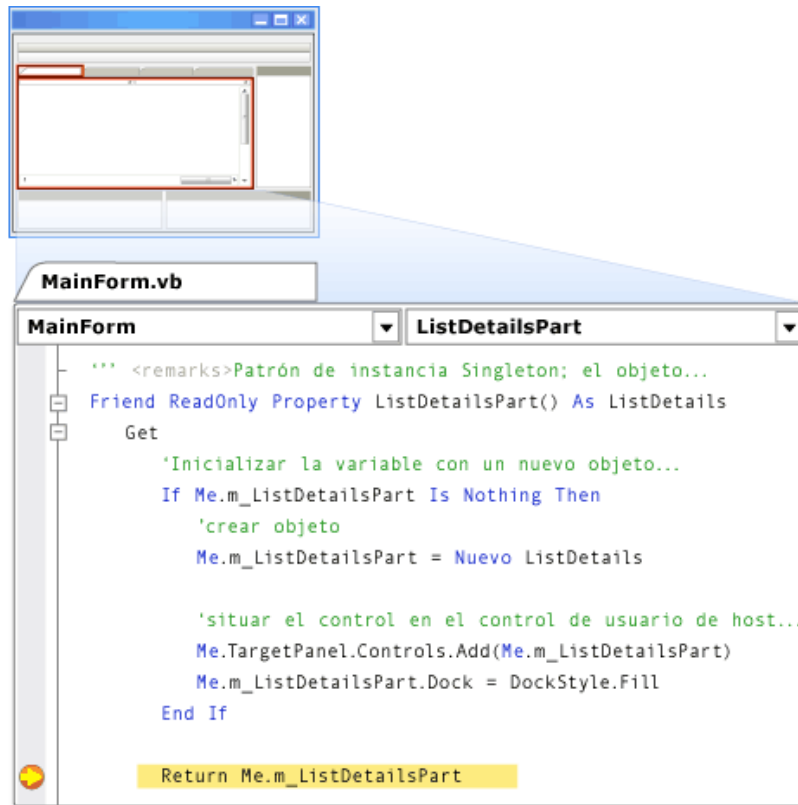


Figura 5. 8 Formulario de Visual Basic en el modo de interrupción



Figura 5. 9 Ventanas de herramientas de depuración

La Lista de errores (Ventana) muestra errores, advertencias y otros mensajes relacionados con la depuración.

5.3.1.1.3. Herramientas de implementación

Visual Studio proporciona dos estrategias de implementación diferentes: ClickOnce y Windows Installer. Con la implementación de ClickOnce, publica la aplicación en una ubicación centralizada y el usuario instala o ejecuta la aplicación desde esa ubicación. La implementación de Windows Installer permite empaquetar la aplicación en un archivo setup.exe y distribuir ese archivo entre los usuarios; ellos ejecutan el archivo setup.exe para instalar la aplicación. Para ver una comparación detallada, consulte Elegir una estrategia de implementación.

ClickOnce permite implementar rápidamente aplicaciones mediante el Asistente para publicación.

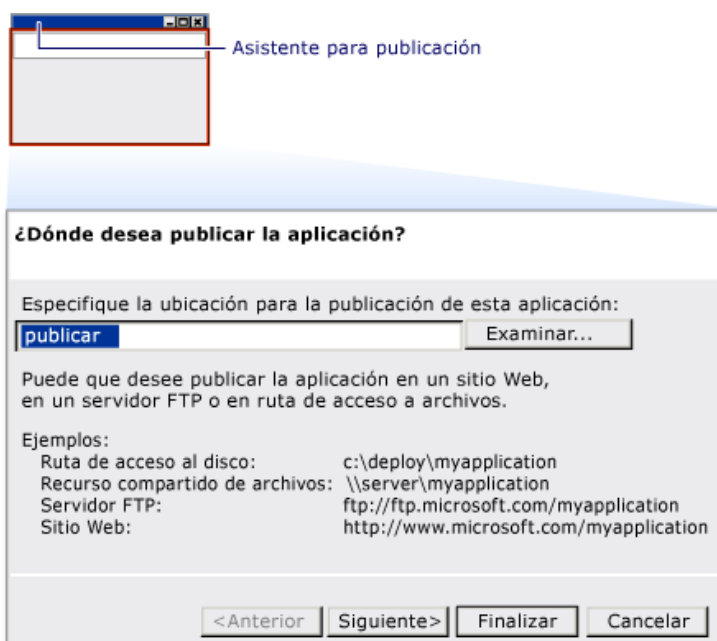


Figura 5. 10 Asistente para publicación

La implementación de Windows Installer proporciona mayor flexibilidad para implementar aplicaciones. Varios editores, como el Editor de acciones personalizadas y el Editor de la interfaz de usuario, permiten personalizar Windows Installer para satisfacer sus necesidades de implementación. Para crear un archivo de instalación básico, utilice el Editor del sistema de archivos, para especificar qué elementos desea implementar.

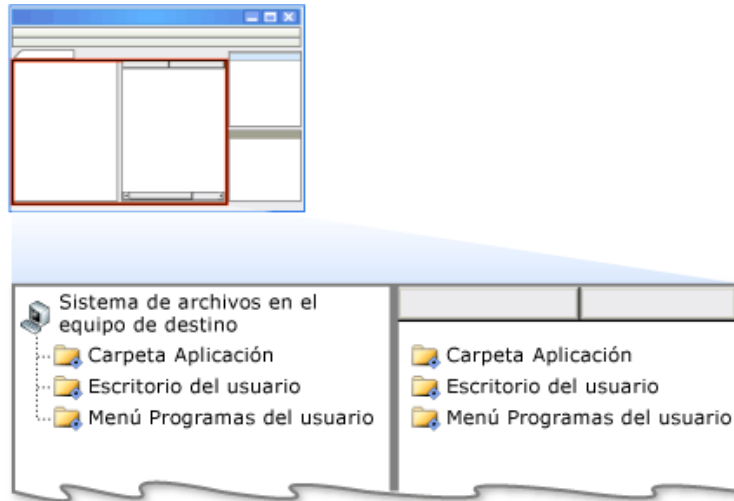


Figura 5. 11 Editor del sistema de archivos

5.3.1.1.4. Documentación de productos

Para obtener acceso a la Ayuda, puede presionar F1 en el IDE o hacer clic en **Documentación de Visual Studio** en el menú **Ayuda**. La documentación de la Ayuda se muestra en el explorador web. Puede utilizar la Ayuda instalada localmente o MSDN en pantalla y otros de recursos en pantalla para obtener ayuda.

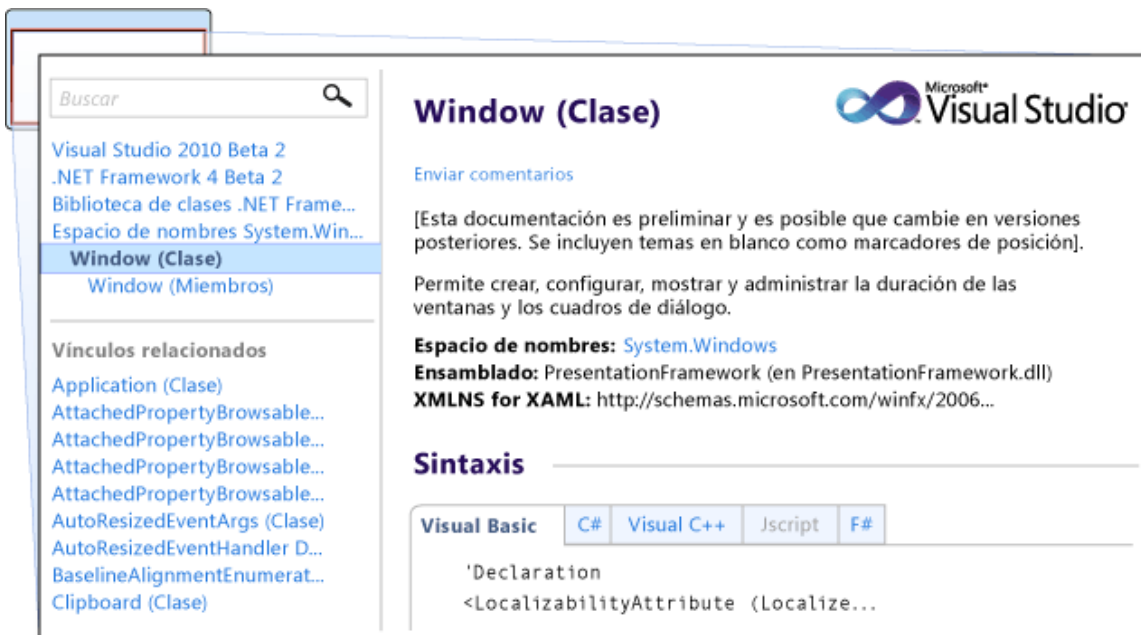


Figura 5. 12 Ayuda en la ventana del explorador

5.3.1.2. Expression Studio 4.0 (Expression Blend)

Posee las siguientes características:

- Crea experiencias web, juegos, aplicaciones de escritorio y mas
- Diseña aplicaciones que usan todo el potencial de Silverlight
- Lleva las ideas desde el concepto hasta la terminación de las mismas
- Trabaja eficientemente con herramientas de diseño

El detalle en resume se encuentra en la sección 4.5 Microsoft Expression Blend

5.3.1.3. Server Management Studio

SQL Server Management Studio es un entorno integrado para obtener acceso a todos los componentes de SQL Server, configurarlos, administrarlos y desarrollarlos. SQL Server Management Studio combina un amplio grupo de herramientas gráficas con una serie de editores de script enriquecidos para ofrecer acceso a SQL Server a desarrolladores y administradores de todos los niveles de especialización.

SQL Server Management Studio combina las características del Administrador corporativo, el Analizador de consultas y Analysis Manager, herramientas incluidas en versiones anteriores de SQL Server, en un único entorno. Además, SQL Server Management Studio funciona con todos los componentes de SQL Server, como Reporting Services, Integration Services y SQL Server Compact 3.5 SP1. Los programadores obtienen una experiencia familiar y los administradores de bases de datos una única herramienta completa que combina herramientas gráficas fáciles de usar con funcionalidad de scripting enriquecida.

5.3.1.3.1. Características de SQL Server Management Studio

SQL Server Management Studio incluye las siguientes características generales:

- Compatibilidad con la mayoría de las tareas administrativas de SQL Server.
- Un entorno único integrado para administración y edición de SQL Server Database Engine (Motor de base de datos de SQL Server).

- Nuevos cuadros de diálogo para la administración de objetos de SQL Server Database Engine (Motor de base de datos de SQL Server), Analysis Services, Reporting Services, Notification Services y SQL Server Compact 3.5 SP1, lo que permite ejecutar las acciones inmediatamente, enviarlas a un editor de código o escribirlas en scripts para ejecutarlas posteriormente.
- Cuadros de diálogo no modales y de tamaño variable que permiten obtener acceso a varias herramientas mientras un cuadro de diálogo está abierto.
- Un cuadro de diálogo común de programación que permite realizar acciones de los cuadros de diálogo de administración en otro momento.
- Exportación e importación del registro de servidor de SQL Server Management Studio desde un entorno de Management Studio a otro.
- Guardado o impresión de archivos de plan de presentación XML o de interbloqueo generados por el Analizador de SQL Server, revisión posterior o envío a los administradores para su análisis.
- Un nuevo cuadro de mensaje de error e informativo que presenta mucha más información, permite enviar a Microsoft un comentario sobre los mensajes, copiar mensajes en el Portapapeles y enviar fácilmente los mensajes por correo electrónico al equipo de soporte.
- Un explorador Web integrado para una rápida exploración de MSDN o la Ayuda en pantalla.
- Integración de la Ayuda de comunidades en línea.
- Un tutorial sobre SQL Server Management Studio para ayudarle a aprovechar las ventajas de las numerosas características nuevas y a que sea más productivo de forma inmediata. Para realizar el tutorial, vaya a Tutoriales.
- Un nuevo monitor de actividad con filtro y actualización automática.
- Interfaces de Correo electrónico de base de datos integradas.

5.3.2. DIAGRAMA DE LA BASES DE DATOS

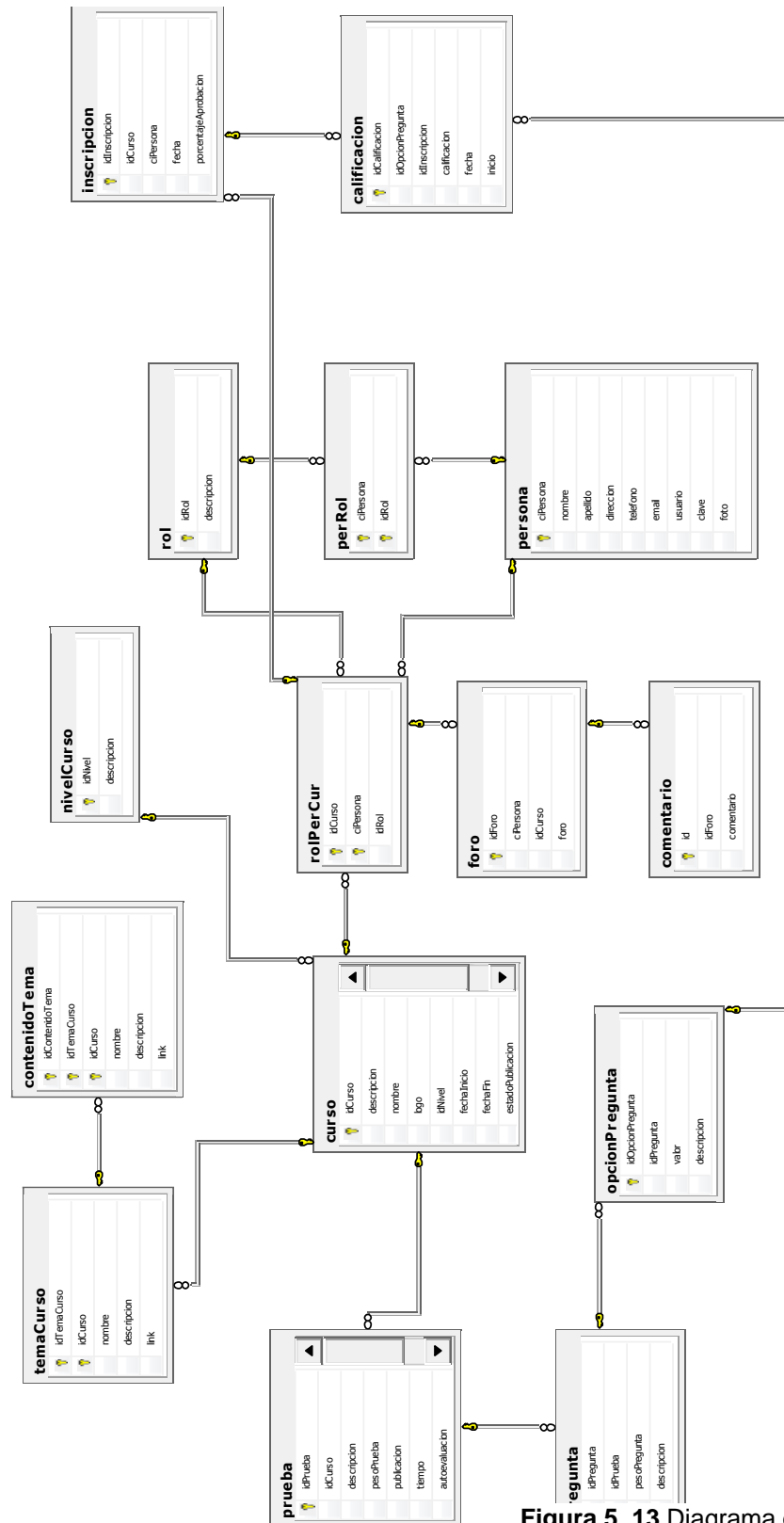


Figura 5. 13 Diagrama de Bases de Datos

5.3.3. DIAGRAMAS DE CASOS DE USO

5.3.3.1. Especificación de caso de uso : Iniciar sesión

→

Figura 5. 14 Especificación de caso de uso: Iniciar Sesión

Descripción

El caso de uso describe como iniciar sesión en el sitio web.

Flujo Básico de Eventos

1. Al acceder a la pantalla “Iniciar sesión”, ingresar “usuario” y “clave”, clic en Aceptar.
2. En caso de que el acceso sea exitoso, muestra una pantalla secundaria que muestra un lista de roles que tiene el usuario. Seleccionar con cual rol desea ingresar al sitio web.

La información consulta la tabla: persona, perRol.

La tabla antes mencionada se encuentra estructurada de la siguiente forma:

Campo	Tipo
ciPersona	varchar(13)
nombre	varchar(50)
apellido	varchar(50)
direccion	varchar(75)
telefono	varchar(25)
email	varchar(100)

usuario	varchar(25)
clave	varchar(25)
foto	image

Tabla 5. 4 Persona

Campo	Tipo
ciPersona	varchar(13)
idRol	int

Tabla 5. 5 PerRol

El sistema presenta al usuario una interfaz con la siguiente estructura:



Figura 5. 15 iniciar sesión

Precondiciones

El usuario debe estar registrado.

5.3.3.2. Especificación de caso de uso: Cambiar clave de usuario

Figura 5. 16 Especificación de caso de uso: Cambiar clave de usuario

Descripción Breve

El caso de uso describe como modificar la clave de un usuario registrado en el sitio web.

Flujo Básico de Eventos

1. Al acceder a la pantalla Iniciar sesión.- “Cambiar clave”, ingresar “Nueva clave” y “Confirmar”.
2. Clic Aceptar, aparece un mensaje de información exitoso.

La información modifica la tabla: persona.

La tabla antes mencionada se encuentra estructurada de la siguiente forma:

Tabla “persona”.- Referencia Tabla 5.4

El sistema presenta al usuario una interfaz con la siguiente estructura:



Figura 5. 17 Cambiar clave

Precondiciones

El usuario debe estar registrado.

5.3.3.3. Especificación de caso de uso: Registrar usuario

Figura 5. 18 Especificación de caso de uso: Registrar usuario

Descripción Breve

El caso de uso describe cómo administrar los registros de los usuarios usando cualquier rol para acceder a los recursos del sitio web.

Flujo Básico de Eventos

1. Al acceder a la pantalla “Iniciar sesión->Registrar”, ingresar cédula, nombre, dirección, e-mail, usuario y clave.
2. En caso de que el usuario ya se encuentre en la base de datos basta con ingresar solamente la cédula para que sus datos se muestren.
3. Clic en guardar

La información se almacena en las tablas persona.

La tabla antes mencionada se encuentra estructurada de la siguiente forma:

Tabla “persona”.- Referencia Tabla 5.4

El sistema presenta al usuario una interfaz con la siguiente estructura:



The image shows a web browser window titled "Registrar usuario" with a close button in the top right corner. The main heading is "Crear su cuenta". Below the heading, there are eight input fields, each with a label and an information icon (i) to its right. The fields are: "Cédula", "Nombre", "Apellido", "Dirección", "Teléfono", "E-mail", "Usuario", and "Clave". To the right of the input fields, there are two buttons: "Limpiar" (Clear) and "Guardar" (Save).

Figura 5. 19 Registrar usuario

Nota.- Al registrar a un alumno automáticamente cuenta con el rol "Alumno".

5.3.3.4. Especificación de caso de uso: Administración de usuarios

Figura 5. 20 Especificación de caso de uso: Administración de usuarios

Descripción Breve

El caso de uso describe cómo administrar la información de los datos personas y sus roles, para acceder al sitio web.

Flujo Básico de Eventos

1. Al acceder a la pantalla “Administración -> Personas” con el rol de usuario “Administrador”, puede hacer lo siguiente:
 - a. Guardar registro persona.- Ingresar los campos: Cédula, Nombre, Apellido, Dirección, e-mail, usuario y clave. Clic en “Guardar”.
 - b. Eliminar registro persona.- Seleccionar en la tabla inferior el registro. Clic en “Eliminar” y aceptar la verificación de eliminación.
 - c. Mostrar registro persona.- En caso de tener usuarios ingresados se muestran al iniciar la pantalla en la tabla posicionada en la parte inferior.

La información se administran de las tablas: persona y rolPerCur.

Las tabla antes mencionada se encuentra estructurada de la siguiente forma:

Tabla “persona”.- Refrencias tabla 5.4

Campo	Tipo
idCurso	int
ciPersona	varchar(13)
idRol	int

Tabla 5. 6 PerRol

El sistema presenta al usuario una interfaz con la siguiente estructura:

Detalle datos personales

Cédula ⓘ

Nombre ⓘ

Apellido ⓘ

Dirección ⓘ

Teléfono ⓘ

E-mail ⓘ

Usuario ⓘ

Clave ⓘ

Escoger Rol

Cédula :

Cédula	Apellido	Nombre	Teléfono	Email
0401389978	GUEVARA	NORMA		
1002798633	Guevara	Andrés		
1002798641	Guevara	Sandro	081465610	sandro.guevara@tcs.com

Figura 5. 21 Administración de personas

Precondiciones

1. Para acceder a la pantalla debe tener un rol "Administrador"

Para mostrar registros en la tabla inferior deben existir usuarios ingresado

5.3.3.5. Especificación de caso de uso: Administrar publicación de cursos

Figura 5. 22 Especificación de caso de uso: Administrar publicación de cursos

Descripción Breve

El caso de uso describe como publicar un curso que está expuesto por un docente.

Flujo Básico de Eventos

1. En la pantalla se puede administrar los datos de personas que pueden usar el sitio web.
2. Al ingresar a la pantalla, se puede navegar en los registros de las personas que están ingresadas en la tabla de la parte inferior.

La información se almacena en las tablas persona, perRol.

La tabla antes mencionada se encuentra estructurada de la siguiente forma:

Tabla “persona”.- Referencia tabla 5.4

Tabla “perRol”.- Referencia tabla 5.5

El sistema presenta al usuario una interfaz con la siguiente estructura:

Publicar curso	Curso	Descripción
<input type="button" value="Publicar"/>	sql basico 2	sld mes 2

Figura 5. 23 Administración de personas

Precondiciones

Administrar curso

Para ingresar a la pantalla necesita permisos de usuario como “Administrador”.

El curso a publicar debe ser expuesto por algún docente.

5.3.3.6. Especificación de caso de uso: Mostrar reporte “Resumen de Cursos”

Figura 5. 24 Especificación de caso de uso: Mostrar reporte “Resumen de Cursos”

Descripción Breve

El caso de uso describe el reporte de curso en función del rol y de que usuarios participan como docentes y alumnos del mismo.

Flujo Básico de Eventos

Al ingresar a la pantalla se exponen los cursos y como detalle los alumnos que se inscribieron para el curso.

La información se obtiene de las tablas persona, cursos.

Las tablas antes mencionadas se encuentra estructurada de la siguiente forma:

Tabla “persona”.- Referencia tabla 5.4

Campo	Tipo
idCurso	int
descripcion	varchar(250)
nombre	varchar(100)
logo	image
idNivel	int
fechaInicio	datetime
fechaFin	datetime
estadoPublicacion	bit

Tabla 5. 7 Curso

El sistema presenta al usuario una interfaz con la siguiente estructura:

Usuarios que participan en cursos						
	Rol	#.Alm.	Cédula	Nombre	Apellido	
▲ nombre: SQL Básico Nivel 1 (3 elementos)						
▲ rol: Docente (1 elemento)						
	Docente	2	1002798641	Sandro	Guevara	
▲ rol: Alumnos (2 elementos)						
	Alumnos	0	0401389978	NORMA	GUEVARA	
	Alumnos	0	0401495221	Miguel	Guevara	

Figura 5. 25 Mostrar reporte "Resumen de Cursos"

Precondiciones

Reporte de cursos

Para ingresar a la pantalla necesita permisos de usuario como "Administrador".

Para visualizar los resultados del curso, los alumnos debieron inscribirse y aplicar las pruebas respectivas.

5.3.3.7. Especificación de caso de uso: Administrar contenido de cursos

Figura 5. 26 Especificación de caso de uso: Administrar contenido de cursos

Descripción Breve

El caso de uso describe la administración de cursos, su temario y el contenido de cada tema.

Flujo Básico de Eventos

1. La pantalla consta del componente TabControl con 3 TabPages, en cada tabPage se describe la administración de los cursos, temas y documentos por tema respectivamente.

La información se almacena en las tablas curso, tema y contenido.

Las tablas antes mencionadas se encuentran estructuradas de la siguiente forma:

Tabla "curso".- Referencia tabla 5.6

Campo	Tipo
idTemaCurso	int
idCurso	int
nombre	varchar(100)
descripcion	varchar(8000)
link	varchar(100)

Tabla 5. 8 TemaCurso

Campo	Tipo
idContenidoTema	int

idTemaCurso	int
idCurso	int
nombre	varchar(100)
descripcion	varchar(250)
link	varchar(100)

Tabla 5. 9 ContenidoTema

El sistema presenta al usuario una interfaz con la siguiente estructura:

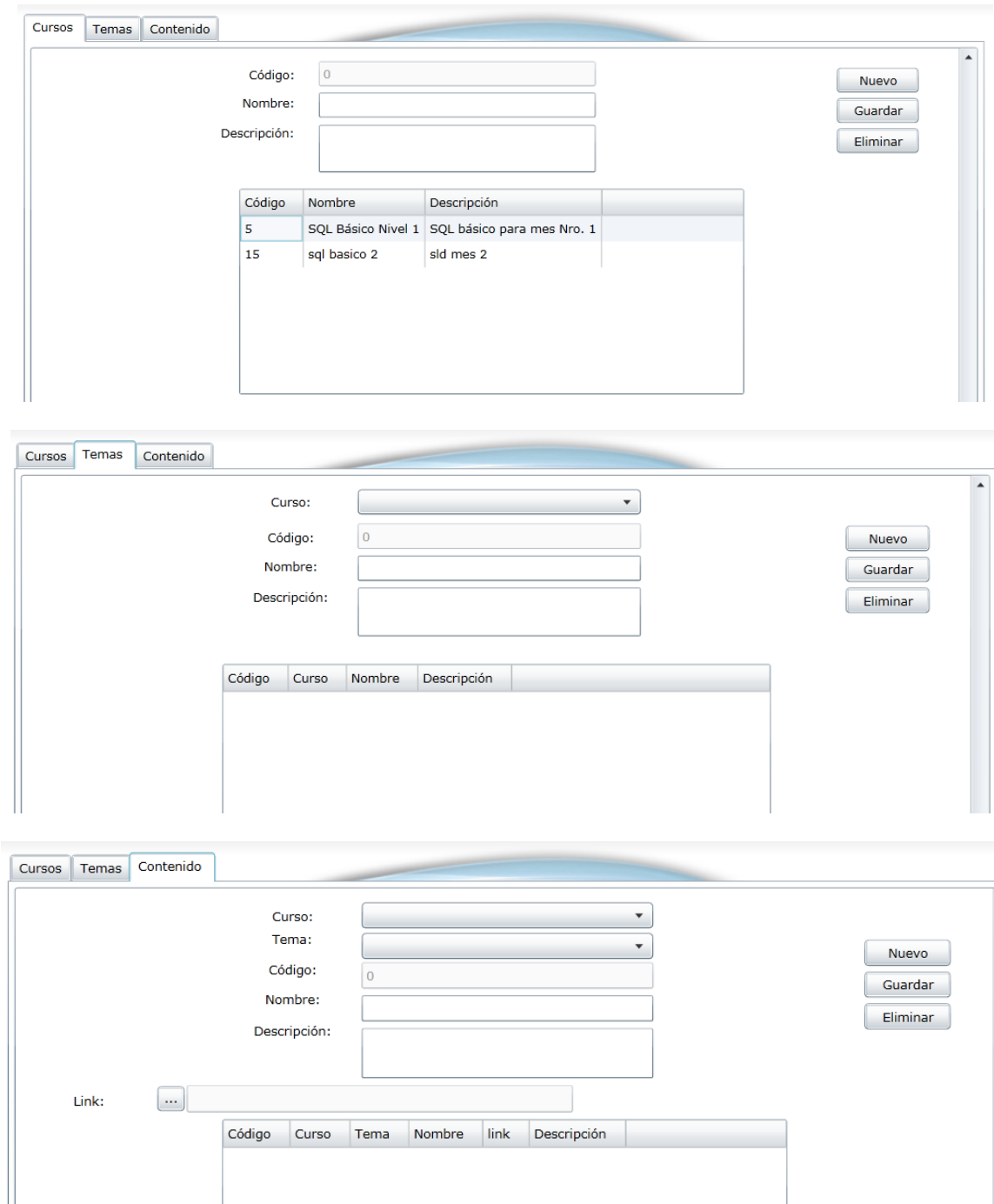


Figura 5. 27 Administrar contenido de cursos

Flujos Alternativos

Mientras los datos son administrados sus resultados son visualizados en la tabla inferior “Contenido de cursos”

Subflujo

En el TabPage de “Cursos”.

En caso que se desee ingresar un nuevo curso, debe ingresar: “Tema del curso”, “Descripción del curso”. Los Id de los cursos son generados al guardar el registro.

Si desea modificar o eliminar un curso debe seleccionar el curso del ComboBox “Cursos”. Se puede modificar los campos de “Tema del curso”, “Descripción del curso”. Clic en Guardar.

En el TabPage de “Temas”.

En caso que se desee ingresar un nuevo tema, debe seleccionar un curso del ComboBox “Cursos” o seleccionarlo en el TabPage de cursos.

Ingresar los campos: “Tema”, “Descripción del tema”. Los Id de los temas son generados al guardar el registro.

Si desea modificar o eliminar un curso, debe seleccionar un curso del ComboBox “Cursos” o seleccionarlo en el TabPage de Cursos, y seleccionar en la tabla “Temario”. Clic en Guardar.

En el TabPage de “Contenido”.

En caso que se desee ingresar un nuevo contenido de algún tema, debe seleccionar un tema del ComboBox “Temas” o seleccionarlo en el TabPage de Temas.

Ingresar los campos: “Contenido”, “Descripción del contenido”. Los Id de los temas son generados al guardar el registro.

Si desea modificar o eliminar un curso, debe seleccionar un curso del ComboBox “Temas” o seleccionarlo en el TabPage de Temas, y seleccionar en la tabla “Contenido”. Clic en Guardar.

Precondiciones

Para ingresar a la pantalla necesita permisos de usuario como “Docente”.

1. En el TabPage de “Temas”.

Un curso debe ser ingresado y seleccionado.

1. En el TabPage de “Contenido”.

Un Tema debe ser ingresado y seleccionado.

5.3.3.8. Especificación de caso de uso: Creación de pruebas

Figura 5. 28 Especificación de caso de uso: Creación de pruebas

Descripción Breve

El caso de uso describe el proceso necesario para hacer pruebas de evaluación del contenido de cursos para los alumnos.

Flujo Básico de Eventos

1. Agregar los campos “Prueba”, “Descripción”, “Ponderación”, seleccionar el CheckBox “Autoevaluación”: si es true la prueba se aplicará a los alumnos pero no se tomará en cuenta en la aprobación del curso, si en false la prueba cuenta como evaluación para el curso.
 - a. En caso de aplicar como true el checkBox de “Publicar” se verifica que las preguntas de la prueba tengan al menos 2 posibles respuestas es decir que el menor tipo posible de pregunta sea de Si/NO.
 - b. Clic en Guardar.
2. Luego de guardar la prueba, para poder ingresar las preguntas se debe hacer clic en el botón “Preguntas”.
 - a. Ingresar los campos “Pregunta”, “Ponderación”. Clic en Guardar
 - b. Si desea eliminar alguna pregunta, seleccionar el botón “Eliminar” de la tabla superior y en el mensaje de confirmación responde “SI”.

3. Luego de guardar la pregunta, para poder ingresar las posibles respuestas se debe hacer clic en el botón “Opciones” en la tabla superior.
 - a. Ingresar los campos “Respuestas”, “Ponderación”. Clic en Guardar
 - b. Si desea eliminar alguna pregunta, seleccionar el botón “Eliminar” de la tabla superior y en el mensaje de confirmación responde “SI”.

La información se almacena en las tablas prueba, pregunta, opcionPregunta.

Las tablas antes mencionadas se encuentran estructuradas de la siguiente forma:

Campo	Tipo
idPrueba	int
idCurso	int
descripcion	varchar(100)
pesoPrueba	float
publicacion	bit
tiempo	int
autoevaluacion	bit

Tabla 5. 10 Prueba

Campo	Tipo
idPregunta	bigint
idPrueba	int
pesoPregunta	float
descripcion	varchar(200)

Tabla 5. 11 Pregunta

Campo	Tipo
idOpcionPregunta	bigint
idPregunta	bigint

valor	bit
descripcion	varchar(100)

Tabla 5. 12 OpcionPregunta

El sistema presenta al usuario una interfaz con la siguiente estructura:

Figura 5. 29 Creación de pruebas

Precondiciones

1. Para ingresar a la pantalla necesita permisos de usuario como “Docente”.
2. Para publicar la prueba el aplicativo verifica que las preguntas de la prueba al menos tenga 2 posibles respuestas.

5.3.3.9. Especificación de caso de uso: Exponer foros

Figura 5. 30 Especificación de caso de uso: Exponer foros

Descripción Breve

El caso de uso describe el proceso necesario para exponer un foro a los alumnos de un determinado curso.

Flujo Básico de Eventos

1. Al ingresar a la pantalla “Foros” debe seguir el siguiente flujo:
 - a. Escoger un curso en el lista situada en la parte superior de la pantalla.
 - b. Con el curso seleccionado en caso de tener foros puede visualizar en la tabla “Foros”. Escoger un foro.
 - c. Con el foro seleccionado en caso de existir comentarios se puede visualizar en la tabla Comentarios.
 - d. En el bloque inferior “Ingreso foros”, puede redactar o pegar texto que puede editarse en la misma pantalla. Clic en el botón en “Enviar Foro”.

La información se almacena en la tabla foro.

Las tablas antes mencionadas se encuentran estructuradas de la siguiente forma:

Campo	Tipo
idForo	bigint
ciPersona	varchar(13)

idCurso	int
foro	varchar(8000)

Tabla 5. 13 Foro

El sistema presenta al usuario una interfaz con la siguiente estructura:

The screenshot shows a web interface for posting a forum post. At the top, there is a dropdown menu labeled "Escojer el curso:" with the selected option "SQL básico para mes Nro. 1". Below this is a section titled "Foros disponibles" containing a table with a header "Temas" and one empty row. Underneath is a "Comentarios" section with a text input field. The main section is "Exponer un foro:", which features a rich text editor with a toolbar containing options for font (Arial), size (12), bold (B), italic (I), underline (U), bulleted list, numbered list, and link. A "Enviar Foro" button is located at the bottom right of the text area.

Figura 5. 31 Exponer foros

Nota.- Al ingresar a la pantalla con rol "Docente" no puede comentar foros, porque los foros son expuestos por el docente a los alumnos de un curso específico.

Precondiciones

1. Para ingresar a la pantalla y poder exponer foros necesita un rol "Docente".

5.3.3.10. Especificación de caso de uso: **Mostrar un resumen de cursos por alumno**

Figura 5. 32 Especificación de caso de uso: **Mostrar un resumen de cursos por alumno**

Descripción Breve

El caso de uso describe como visualizar un resumen de los resultados de los cursos con un detalle por alumno.

Flujo Básico de Eventos

1. Al ingresar a la pantalla de reporte “Cursos por alumno” puede visualizar un reporte de los cursos que un docente está impartiendo.
2. Al hacer clic en el botón “Ver Detalle” muestra en una ventana secundaria la información del alumno seleccionado.

La información se obtiene de la tabla calificación.

Las tablas antes mencionadas se encuentran estructuradas de la siguiente forma:

Campo	Tipo
idCalificacion	bigint
idOpcionPregunta	bigint
idInscripcion	bigint
calificacion	float
fecha	datetime
inicio	bit

Tabla 5. 14 Calificacion

El sistema presenta al usuario una interfaz con la siguiente estructura:



	Publicar curso	Cédula	Nombre	Apellido	
▲	nombre: SQL Básico Nivel 1 (2 elementos)				
▲	ciPersona: 0401389978 (1 elemento)				
	<input type="button" value="Ver Detalle"/>	0401389978	NORMA	GUEVARA	
▲	ciPersona: 0401495221 (1 elemento)				
	<input type="button" value="Ver Detalle"/>	0401495221	Miguel	Guevara	

Figura 5. 33 Mostrar un resumen de cursos por alumno

Precondiciones

1. Para ingresar a la pantalla y visualizar datos de resumen de los resultados de un determinado curso los usuarios deben cumplir con las siguientes condiciones.
 - a. Debe iniciar sesión un usuario con rol “Docente”.
 - b. El docente expuso una evaluación a un grupo de alumnos del curso que el imparte
 - c. Los alumnos de ese curso hayan aplicado la prueba.

5.3.3.11. Especificación de caso de uso: Inscribirse a un curso

Figura 5. 34 Especificación de caso de uso: Inscribirse a un curso

Descripción Breve

El caso de uso describe como un alumno se puede inscribir a un determinado curso.

Flujo Básico de Eventos

1. Al ingresar al sitio web muestra la pantalla “HOME”, donde se muestra una tabla con los cursos publicados, y en cada fila de un curso hay la opción de visualizar el temario y de inscribirse.
2. Al hacer clic en el botón “Inscribir” puede suceder los siguiente:
 - a. En caso de no tener iniciada una sesión muestra un mensaje de error indicando que primero debe Iniciar sesión en el sitio web.
 - b. Si ya Inició la sesión en el sitio web muestra una pantalla secundaria que expone un resumen de la inscripción en caso que fuese exitoso, caso contrario muestra un mensaje de error.

La información se almacena en la tabla inscripcion.

Las tablas antes mencionadas se encuentran estructuradas de la siguiente forma:

Campo	Tipo
idInscripcion	bigint
idCurso	int
ciPersona	varchar(13)
fecha	datetime
porcentajeAprobacion	int

Tabla 5. 15 Inscripcion

El sistema presenta al usuario una interfaz con la siguiente estructura:



The image shows a software window titled "Inscripción" with a close button in the top right corner. The window contains the following text:

Cédula:	0401495221
Nombre:	Miguel Guevara
Curso :	SQL Básico Nivel 1
Nro. Inscripción :	24

At the bottom right of the window is a button labeled "Aceptar".

Figura 5. 35 Inscribirse a un curso

Precondiciones

El alumno debe Iniciar sesión con rol "Alumno".

5.3.3.12. Especificación de caso de uso: Mostrar contenido de un curso para sus alumnos

Figura 5. 36 Especificación de caso de uso: Mostrar contenido de un curso para sus alumnos

Descripción Breve

El caso de uso describe como un alumno puede obtener el contenido de un curso que se inscribió.

Flujo Básico de Eventos

1. Ingresar a la pantalla “Curso”
2. En el bloque “Contenido de cursos”, abrir el árbol agrupado por curso.

La información se obtiene de la tabla contenidoTema.

La tabla antes mencionada se encuentra estructurada de la siguiente forma:

Tabla “contenidoTema”.- Referencia 5.11

El sistema presenta al usuario una interfaz con la siguiente estructura:

Curso	Tema	Contenido	Archivo
▲ curso: SQL Básico Nivel 1 (2 elementos)			
▲ tema: Select simples (2 elementos)			
SQL Básico Nivel 1	Select simples	Select, where	http://localhost/PMV.Web/Uploads/crossdomain.xml
SQL Básico Nivel 1	Select simples	cont2	http://localhost/PMV.Web/Uploads/dir MauricioRea.txt

Figura 5. 37 Mostrar contenido de un curso para sus alumnos

Precondiciones

1. El usuario debe Iniciar sesión con rol "Alumno".
2. Debe estar inscrito en algún curso
3. El curso debe estar publicado y tener contenido.

5.3.3.13. Especificación de caso de uso: Aplicar pruebas

Figura 5. 38 Especificación de caso de uso: Aplicar pruebas

Descripción Breve

El caso de uso describe como un alumno puede aplicar a una prueba de evaluación sea ésta como autoevaluación o como una prueba válida para aprobar el curso.

Flujo Básico de Eventos

1. Ingresar a la pantalla “Aplicar prueba”
2. Seleccionar un curso el bloque superior de “Cursos”.
3. Muestra las pruebas disponibles a aplicar en el bloque inferior de “Pruebas”.
4. Escoger la prueba y hacer clic en el botón “Aplicar”
5. Se muestra una pantalla secundaria con el siguiente funcionalidad:
 - a. En la parte superior muestra la pregunta y en la parte inferior muestra una lista de posibles respuestas con un check vacío en la parte izquierda de cada opción. Escoger la/las respuesta.
 - b. Clic en siguiente.- Cada vez se guarda la respuesta que el usuario escogió y muestra la siguiente pregunta.

La información se almacena en la tabla calificación.

La tabla antes mencionada se encuentra estructurada de la siguiente forma:

Tabla “calificacion”.- Referencia tabla 5.13

El sistema presenta al usuario una interfaz con la siguiente estructura:



Figura 5. 39 Aplicar pruebas

Precondiciones

1. El usuario debe Iniciar sesión con rol "Alumno".
2. Debe estar inscrito en algún curso
3. La prueba debe estar publicada.

5.3.3.14. Especificación de caso de uso: Guardar comentario de un foro

Figura 5. 40 Especificación de caso de uso: Guardar comentario de un foro

Descripción Breve

El caso de uso describe como un alumno puede comentar un foro de un curso específico.

Flujo Básico de Eventos

1. Ingresar a la pantalla “Foros”
2. Seleccionar un curso el bloque superior de “Cursos”.
3. Muestra los foros disponibles en el bloque “Foros”. Seleccionar el foro que desea comentar.
4. Redactar o pegar el comentario y hacer clic en “Enviar comentario”.

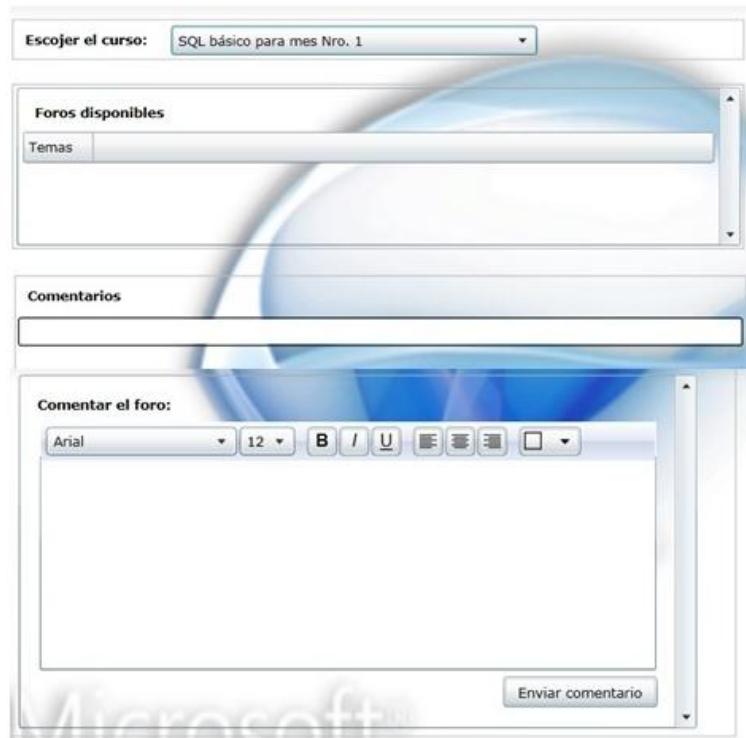
La información se almacena en la tabla comentario.

La tabla antes mencionada se encuentra estructurada de la siguiente forma:

Campo	Tipo
id	bigint
idForo	bigint
comentario	varchar(8000)

Tabla 5. 16 Comentario

El sistema presenta al usuario una interfaz con la siguiente estructura:



The screenshot displays a web interface for posting a forum comment. At the top, there is a dropdown menu labeled "Escojer el curso:" with the selected option "SQL básico para mes Nro. 1". Below this is a section titled "Foros disponibles" containing a list box labeled "Temas". Underneath is a "Comentarios" section with a text input field. The bottom section, "Comentar el foro:", features a rich text editor with a font dropdown set to "Arial", a size dropdown set to "12", and buttons for bold (B), italic (I), underline (U), bulleted list, numbered list, link, and unlink. A large text area for the comment is provided, and an "Enviar comentario" button is located at the bottom right of this section.

Figura 5. 41 Guardar comentario de un foro

Precondiciones

- El usuario debe Iniciar sesión con rol "Alumno".
- Debe estar inscrito en algún curso
- La prueba debe estar publicada.

5.3.3.15. Especificación de caso de uso: Mostrar reporte de pruebas aplicadas por el alumno

Figura 5. 42 Especificación de caso de uso: Mostrar reporte de pruebas aplicadas por el alumno

Descripción Breve

El caso de uso describe como poder visualizar un reporte de las evaluaciones aplicadas en un curso.

Flujo Básico de Eventos

1. Ingresar a la pantalla de Reportes – Evaluaciones aplicadas

La información se obtiene de la tabla calificación.

La tabla antes mencionada se encuentra estructurada de la siguiente forma:

Tabla “calificación”.- Referencia tabla 5.13

El sistema presenta al usuario una interfaz con la siguiente estructura:



Respuestas	Valor prueba	Valor alumno	Calificación Ponderada Prueba(%)
------------	--------------	--------------	----------------------------------

Figura 5. 43 Mostrar reporte de pruebas aplicadas por el alumno

Precondiciones

- El usuario debe Iniciar sesión con rol “Alumno”.

- Debe estar inscrito en algún curso.
- El alumno debió aplicar a alguna evaluación.

5.3.3.16. Especificación de caso de uso: Mostrar cursos publicados

Figura 5. 44 Especificación de caso de uso: Mostrar cursos publicados

Descripción Breve

El caso de uso describe como poder visualizar un reporte de los cursos que están publicados.

Flujo Básico de Eventos

1. Ingresar a la pantalla HOME, se muestra en la tabla Cursos publicados los cursos publicados.

La información se obtiene de la tabla curso.

La tabla antes mencionada se encuentra estructurada de la siguiente forma:

Tabla "curso".- Referencia tabla 5.6

El sistema presenta al usuario una interfaz con la siguiente estructura:

Cursos Disponibles

Temas	Inscripción	Id	Curso	Descripción	
Temas	Inscribirse	5	SQL Básico Nivel 1	SQL básico para mes Nro. 1	
Temas	Inscribirse	15	sql basico 2	sld mes 2	

Figura 5. 45 Mostrar cursos publicados

5.3.3.17. Especificación de caso de uso: Mostrar el temario de un curso seleccionado

Figura 5. 46 Especificación de caso de uso: Mostrar el temario de un curso seleccionado

Descripción Breve

El caso de uso describe como poder visualizar el temario de un curso publicado.

Flujo Básico de Eventos

1. Ingresar a la pantalla HOME, se muestra en una tabla los cursos publicados y un botón en el cual al hacer clic muestra en una pantalla secundaria el temario.

La información se obtiene de la tabla temaCurso.

La tabla antes mencionada se encuentra estructurada de la siguiente forma:

Tabla “temaCurso” .- Referencias tabla 5.7

El sistema presenta al usuario una interfaz con la siguiente estructura:

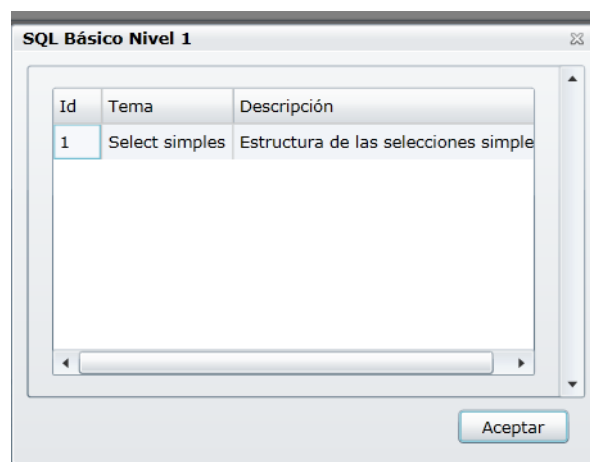


Figura 5. 47 Mostrar el temario de un curso seleccionado

5.3.4. DIAGRAMAS DE ACTIVIDADES

5.3.4.1. Iniciar sesión (todos los usuarios)

Figura 5. 48 Diagrama de actividades: Iniciar sesión

5.3.4.2. Cambio de contraseña (todos los usuarios)

Figura 5. 49 Diagrama de actividades: Cambio de contraseña

5.3.4.3. Registrar usuarios

Figura 5. 50 Diagrama de actividades: Registrar usuarios

5.3.4.4. Administrar información de personas(Inicio de sesión con rol “Administrador”)

Figura 5. 51 Diagrama de actividades: Administrar información de personas

5.3.4.5. Exponer cursos (Rol “Docente”, pantalla cursos)

Figura 5. 52 Diagrama de actividades: Exponer cursos

5.3.4.6. Pubicar cursos (Rol “Administrador”)

Figura 5. 53 Diagrama de actividades: Pubicar cursos

5.3.4.7. Inscribirse a un curso publicado (Rol “Alumno”)

Figura 5. 54 Diagrama de actividades: Inscribirse a un curso publicado

5.3.4.8. Obtener contenido del curso (Rol “Alumno”)

Figura 5. 55 Diagrama de actividades: Obtener contenido del curso

5.3.4.9. Exponer foros (Rol Docente)

Figura 5. 56 Diagrama de actividades: Exponer foros

5.3.4.10. Comentar foro

Figura 5. 57 Diagrama de actividades: Comentar foro

5.3.4.11. Crear pruebas

Figura 5. 58 Diagrama de actividades: Crear pruebas

5.3.4.12. Aplicar a una prueba

Figura 5. 59 Diagrama de actividades: Aplicar a una prueba

5.3.4.13. Reporte del alumno

Figura 5. 60 Diagrama de actividades: Reporte del alumno

5.3.4.14. Reporte docente

Figura 5. 61 Diagrama de actividades: Reporte docente

5.3.5. DIAGRAMAS DE SECUENCIA

5.3.5.1. Inicio de sesión



Figura 5. 62 Diagrama de secuencia: Inicio de sesión

5.3.5.2. Cursos y foros

■

Figura 5. 63 Diagrama de secuencia: Cursos y foros

5.4. DOCUMENTACIÓN

5.4.1. DOCUMENTACIÓN TÉCNICA

La documentación técnica se encuentra detallada en el CD de entrega adjunto.

5.4.2. DOCUMENTACIÓN DE USUARIO

La documentación de usuario se encuentra detallada en el CD de entrega adjunto.

5.5. PRUEBAS FINALES

5.5.1. PRUEBA DE CARGA Y STRESS

La documentación de las pruebas de carga se encuentra detallada en el CD de entrega adjunto.

CAPÍTULO VI.- CONCLUSIONES Y RECOMENDACIONES

6.1. CONCLUSIONES

- Silverlight es una de las plataformas de desarrollo más prometedoras de Microsoft llena un nicho que se encuentra sólidamente entre las aplicaciones tradicionales de escritorio y aplicaciones web, al tiempo que ofrece las capacidades que tanto falta. Funciona en diferentes navegadores y sistemas operativos diferentes.
- Las aplicaciones RIA (Rich Internet Application) integran las ventajas de las aplicaciones web como: facilidad de despliegue y mantenimiento, actualización centralizada, independencia de plataforma y una mejor experiencia para el usuario gracias a: riqueza multimedia y visual, familiaridad con controles y ausencia del efecto de refresco en las páginas.
- Los controles de usuario son ideales para escenarios de reutilización simple, usándose además accesos a datos lo que la convierte en una técnica para crear una clara separación entre la interfaz de usuario y sus datos subyacentes complementando la presentación enriquecida con sus validaciones controlada por excepciones e interfaces.
- El uso seguro de servidores de dominios se maneja con WCF evitando la divulgación de información y denegación de servicio, así como, la restricción del acceso de datos y operación para los usuarios autenticados y usuarios en roles específicos
- La integración multimedia de Silverlight es simple, proporcionando recurso de fácil manipulación para reproducir contenido multimedia

6.2. RECOMENDACIONES

- Para sacar el máximo provecho de la plataforma, se tendrá que aprender a el lenguaje XAML, manteniendo un equilibrio entre lo que se escribe en el código y lo que se pone en el marcado.
- Para complementar el estudio realizado se debería considerar el análisis de frameworks adicionales para RIA, como es Java Fx de Sun Oracle, Adobe Flex o Java Rich Faces que día a día cambian constantemente.
- Una vez seleccionado un framework de desarrollo resulta necesario establecer estándares y patrones de diseño que aseguren una arquitectura sólida que saquen provecho a dicho framework.
- Para establecer una arquitectura solida en aplicaciones RIA utilizando Microsoft Silverlighth, se recomienda usar en la capa de persistencia de datos hacer uso de tecnologías base tipo O/RM (Object/Relational Mapping frameworks), para obtener entidades y contextos necesarios para el modelo que exige Microsoft Silverlight.
- Para mejorar las vistas dentro de la capa de presentación se recomienda usar Microsoft Expression Blend por la mejora en el desarrollo del diseño de una aplicación.

6.3. POSIBLES TEMAS DE TESIS

- Estudio de la Arquitectura N-Capas orientada al dominio utilizando tecnología .Net.

El estilo arquitectural en capas se basa en una distribución jerárquica de los roles y las responsabilidades para proporcionar una división efectiva de los problemas a resolver. Los roles indican el tipo y la forma de la interacción con otras capas y las responsabilidades la funcionalidad que implementan.

- Comparativas de tecnologías O/RM (Object/Relational Mapping frameworks) para el desarrollo de la capa de persistencia de datos en aplicaciones.

Considérese el uso de un O/RM que realice dicha traducción entre las entidades del dominio y la base de datos. Si adicionalmente se está creando una aplicación y un almacén desde cero“, se puede hacer uso del O/RM incluso para generar el esquema de la base de datos a partir del modelo lógico de datos del O/RM (Esto se puede realizar por ejemplo con Entity Framework 4.0). Si la base de datos es una ya existente, se puede utilizar las herramientas del OR/M para mapear entre el modelo de datos del dominio y el modelo relacional.

BIBLIOGRAFÍA**LIBRO Y MANUALES**

- INTRODUCIONG SILVERLIGHT 4; Ghoda Ashish; Springer Science+Business Media ;United States of America;2010
- BEGINNING SILVERLIGHT 4 IN C#; Lair Robert; Springer Science+Business Media ;United States of America;2010
- SILVERLIGHT 4 IN ACTION; Brown Pete; Printed in the United States of America, ©2010 by Manning Publications Co
- FUNDATION EXPRESSION BLEND 4 WITH SILVERLIGHT; Gaudioso Victor; Springer Science+Business Media ;United States of America;2010
- PROGRAMACION EN SILVERLIGTH 4.0; Posadas Mario; Netalia; Netalia, S.L. 2011

ARTÍCULOS

Diseño de aplicaciones en Silverlight 4 usando Servicios RIA, MEF y MVVM Mora David; 2010

De la base de datos al cliente con WCF RIA Service; Mora David; 2010

Expression Blend(5-13); Sanchez Jaime; Febrero 2010

DIRECCIONES ELECTRÓNICAS

- Interfaces Web
<http://www.drauta.com/interfaces-web/>
<http://www.monografias.com/trabajos10/diusuar/diusuar.shtml>
- Especificación de interfaz de usuario
<http://pjmolina.com/papers/TesisPjmolina.pdf>
- Aspectos lingüísticos y comunicativos del interfaz de usuario de un software basado en tecnología de la Web
<http://www.um.es/tonosdigital/znum2/estudios/InterfazdeusuarioUtaTonos2.htm>
- Conozca Silverlight
<http://msdn.microsoft.com/es-es/silverlight/bb187401>

- Qué es Silverlight
<http://www.tucompu.com/2007/10/qu-es-silverlight.html>
- Introduccion a Silverlight
<http://web.ontuts.com/tutoriales/introduccion-a-microsoft-silverlight-parte-i/>
<http://web.ontuts.com/tutoriales/introduccion-a-microsoft-silverlight-parte-ii/>
- Empiece a disfrutar aún más navegando por la Web
<http://msdn.microsoft.com/es-es/magazine/cc163404.aspx>
- Interfaz de usuario
<http://www.fismat.umich.mx/~crivera/tesis/node6.html>
- Introduccion a RIA (Rich Internet Application)
<http://www.nohaylimites.com/?p=184>
- Rich Internet Applications (RIA): A Convergence of User Interface Paradigms of Web and Desktop
<http://www.flomedia.de/diploma/>
- Fundamento a la Arquitectura RIA
<http://www.mailxmail.com/curso-mysql-php/fundamento-arquitectura-ria>
- Rich Internet Application
<http://www.slideshare.net/lisepi09/rich-internet-applications-2615942>
- El mundo de RIA
<http://www.alejandromicheloud.com.ar/blog/?p=33--> revisar
- Adobe Flash
http://help.adobe.com/es_ES/as3/learn/index.html
- ICE Faces
<http://www.juntadeandalucia.es/xwiki/bin/view/MADEJA/ICEFaces>
- Validación de datos con Silverlight 4 y DataForm
<http://msdn.microsoft.com/es-es/magazine/ee335695.aspx>
- Introducción a Silverlight – Parte 2: Diseño de la interfase al usuario y navegación
<http://maromasdigitales.net/2010/02/introduccion-a-silverlight-parte-2-diseno-de-la-interfase-al-usuario-y-navegacion/>
- Patrón Modelo-Vista-Modelo de Vista (MVVM) Explicado

- <http://maromasdigitales.net/2010/05/patron-mvvm-explicado/>
- Una introducción simple al patrón Model View ViewModel para construir aplicaciones Silverlight y Windows Presentation Foundation
<http://fernandomachadopiriz.com/2010/06/09/una-simple-introduccion-al-patron-model-view-viewmodel-para-construir-aplicaciones-silverlight-y-windows-presentation-foundation/>
 - Silverlight MVVM + WCF Ria Services: Una combinación poderosa
<http://www.weboomania.com/2010/01/18/silverlight-mvvm-wcf-ria-services-una-combinacion-poderosa/>
 - ¿Por qué usar MVVM?
<http://blogs.ligasilverlight.com/2010/05/por-que-usar-mvvm/>
 - Patrón de Diseño: Model View ViewModel (I)
<http://livingincoria.wordpress.com/2011/04/13/patron-de-diseo-model-view-viewmodel-i/>
 - Silverlight-4
<http://www.silverlight.net/getstarted/silverlight-4/>
 - DependencyObject
[http://msdn.microsoft.com/es-es/library/system.windows.dependencyobject\(v=vs.95\).aspx](http://msdn.microsoft.com/es-es/library/system.windows.dependencyobject(v=vs.95).aspx)
[http://msdn.microsoft.com/es-es/library/system.windows.uielement_members\(v=vs.95\).aspx](http://msdn.microsoft.com/es-es/library/system.windows.uielement_members(v=vs.95).aspx)

DICCIONARIO DE DATOS

- **Applets.** es un componente de una *aplicación* que se ejecuta en el contexto de otro programa, por ejemplo un navegador web. El *applet* debe ejecutarse en un *contenedor*, que lo proporciona un programa anfitrión, mediante un *plugin*, o en aplicaciones como teléfonos móviles que soportan el modelo de programación por 'applets'
- **CLR.** ("entorno en tiempo de ejecución de lenguaje común") es un entorno de ejecución para los códigos de los programas que corren sobre la plataforma Microsoft .NET.
- **Clustering** El término cluster se aplica a los conjuntos o conglomerados de computadoras construidos mediante la utilización de componentes de hardware comunes y que se comportan como si fuesen una única computadora.
- **DeepZoom** : permite ver imágenes de alta resolución de forma interactiva. Las imágenes se pueden acercar y alejar rápidamente sin afectar al rendimiento de la aplicación
- **Enrutado: direccionador, ruteador o encaminador** es un dispositivo de hardware para interconexión de red de ordenadores que opera en la capa tres (nivel de red) del modelo OSI.
- **Framework:** Es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado.
- **Gateway:** Una **pasarela** o **puerta de enlace** (del inglés *gateway*) es un dispositivo, con frecuencia una computadora, que permite interconectar redes con protocolos y arquitecturas diferentes a todos los niveles de comunicación. Su propósito es traducir la información del protocolo utilizado en una red al protocolo usado en la red de destino.
- **MP3:** Es un formato de compresión de audio digital patentado que usa un algoritmo con pérdida para conseguir un menor tamaño de archivo. Es un

formato de audio común usado para música tanto en ordenadores como en reproductores de audio portátil.

- **iTunes** : es un reproductor de medios y tienda de contenidos multimedia desarrollado por Apple con el fin de reproducir, organizar y sincronizar iPods, iPhones, iPads y comprar música. Es compatible con ordenadores basados en sistemas operativos Mac OS X, Windows 2000, Windows XP, Windows Vista y Windows 7.
- **Just-In-Time:** El método justo a tiempo es un sistema de organización de la producción para las fábricas, de origen japonés. También conocido como *método Toyota* o *JIT*, permite aumentar la productividad. Permite reducir el costo de la gestión y por pérdidas en almacenes debido a stocks innecesarios. De esta forma, no se produce bajo suposiciones, sino sobre pedidos reales.
- **LINQ to SQL:** Es una implementación de O/RM(object relational mapping, mapeador de objetos relacionales) que viene con la versión “Orcas” del .NET Framework, y nos permite modelar bases de datos relacionales con clases de .NET. Se puede consultar bases de datos con LINQ, así como actualizar/añadir/borrar datos de ellas.
- **Entity Framework** : Se parece a una tecnología interesante que es más potente y avanzado que LINQ to SQL. Ambas tecnologías tienen un tipo diferente de la filosofía, pero cuenta con varias implementaciones similares. La EF es mucho más que un ORM (Object Relational Mapping) de Microsoft. Permite a los desarrolladores para consultar y manipular los datos mediante un modelo conceptual en lugar de un modelo de almacenamiento físico.
- **MeasureOverride:** Proporciona el comportamiento para el paso de medida del diseño de Silverlight. Las clases pueden invalidar este método para definir su propio comportamiento paso de medida. Este método tiene una implementación predeterminada que realiza incorporado en el diseño para la mayoría de Silverlight FrameworkElement clases. MeasureOverride

proporciona con eficacia adicional de ejecución de la medida siempre que el método se llama, ya sea por la lógica de diseño interior o el código de aplicación. Si se produce un control de contenido a distribuir, la lógica MeasureOverride define el control de la medida específica pasar la lógica de diseño.

- **Monikers:** La palabra "apodo" se utiliza, principalmente en Microsoft API , para describir los objetos que actúan como identificadores. La información exacta que figura en un apodo depende del contexto, sino que se compone generalmente de uno o varios de los siguientes campos: numéricos / identificador de texto (similar a un GUID), nombre, ruta de acceso en un sistema de archivos, URL, etc El apodo es También supone que es inmutable y la creación de un nuevo identificador suele implicar la creación de un nuevo nombre. La palabra se utiliza en COM , en Microsoft Dynamics , así como SQL Server o ActiveX .
- **MoonLight:** Moonlight es una implementación de código abierto de Silverlight (<http://silverlight.net>), principalmente para Linux y otros sistemas operativos basados en Unix/X11.
- **Null:** El término *null* o **nulo** es a menudo utilizado en la computación, haciendo referencia a la nada.
- **Out-Of-Browser:** Silverlight ofrece un nuevo conjunto de características para la construcción ligera, las experiencias recinto compañero para la Web que se ejecutan en el escritorio. fuera de Silverlight de navegador web permite la construcción de aún más, las relaciones persistentes con los clientes.
- **Sandboxes:** En seguridad informática , una caja de arena es un mecanismo de seguridad para la separación de los programas en ejecución. A menudo se utiliza para ejecutar código no probado, o no son de confianza sin verificar los programas de terceros, los proveedores y los usuarios no son de confianza.

- **Streaming:** consiste en la distribución de audio o video por Internet. La palabra *streaming* se refiere a que se trata de una corriente continua (sin interrupción). El usuario puede escuchar o ver en el momento que quiera. Este tipo de tecnología permite que se almacenen en un búfer lo que se va escuchando o viendo.
- **W3C:** El World Wide Web Consortium (W3C) es una comunidad internacional que desarrolla estándares que aseguran el crecimiento de la Web a largo plazo.
- **WCF: Windows Communication Foundation** o WCF (también conocido como Indigo), es la nueva plataforma de mensajería que forma parte de la API de la Plataforma .NET 3.0 (antes conocida como WinFX, y que no son más que extensiones para la versión 2.0). Se encuentra basada en la Plataforma .NET 2.0 y de forma predeterminada se incluye en el Sistema Operativo Microsoft Windows Vista.
- **WPF: Windows Presentation Foundation (WPF)** es una tecnología de Microsoft, presentada como parte de Windows Vista. Permite el desarrollo de interfaces de interacción en Windows tomando las mejores características de las aplicaciones Windows y de las aplicaciones web.