



Universidad de las Ciencias Informáticas

**Título: Biblioteca de componentes de interfaz de usuario para el
Sistema de Identificación, Inmigración y Extranjería de la
República de Cuba.**

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

Autor: Julio Amado López Palma.

Tutor: Ing. Humberto Salazar Pacios.

Ciudad de la Habana, Junio de 2011.

“Año 53 de la Revolución”

"No te preocupes por ser mejor que tus contemporáneos o predecesores. Intenta ser mejor que tú mismo."

William Faulkner

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor del trabajo titulado:

Biblioteca de componentes de interfaz de usuario para el Sistema de Identificación, Inmigración y Extranjería de la República de Cuba. Y autorizo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Autor: Julio Amado López Palma.

Tutor: Ing. Humberto Salazar Pacios.

DATOS DEL CONTACTO

Autor:

Julio Amado López Palma

Correo: japalma@estudiantes.uci.cu

Universidad de las Ciencias Informáticas, Ciudad de la Habana, Cuba

Tutor:

Ing. Humberto Salazar Pacios

Correo: hsalazar@uci.cu

Universidad de las Ciencias Informáticas, Ciudad de la Habana, Cuba

A mis viejos queridos por brindarme su apoyo incondicional y sus excelentes consejos que me permitieron lograr mis propias metas.

A mis otros padres Nelson y Virgen por ser tan especiales y llenar esos espacios vacíos creados por la ley de la vida, con amor y respeto.

A mis padres blancos Carmen y Carlos por tantos platos de comida y buenos momentos que me brindaron bajo su propio techo.

A mis hermanos Yanet, Chichi, José Daniel, Indiana, Yunior, Javier y Andy por convertirse en mis socorristas en los momentos más difíciles.

A mi abuela querida que aunque no le demuestre todo mi amor, en realidad la quiero muchísimo.

Al mejor tutor de todos que me brindó respeto, apoyo y regaños necesarios para hacer posible la realización de esta tesis, gracias de todo corazón.

Al trío de infladores que me acompañaron durante todo este año la Gorda, el Papá de los Infladores y Fuckęr por apoyarme constantemente y tener que aguantarlos porque no me quedaba más remedio.

Al Feo y a Nairy por ser buenos amigos, por sus consejos, hacerme reír tanto y por ser tan insoportables la mayor parte del tiempo.

A todas aquellas personas que he conocido en la universidad y no he mencionado aquí.

Agradezco a mi padre que ha sido mi ejemplo a seguir, la persona más importante en mi vida.

A mi vieja y a Nelson que me han ayudado inmensamente en hacer posible este día.

*A Helen Susel por ser mi inspiración en los momentos más difíciles, cuando no tuve a nadie
junto a mí, tu recuerdo siempre fue lo que me dio la fuerza para seguir adelante.*

RESUMEN

En los últimos años el Ministerio del Interior de la República de Cuba ha venido desarrollando sistemas para la modernización de la Dirección de Inmigración y Extranjería y la Dirección de Identificación y Registro. Por ello y en aras de una mejora tecnológica el Ministerio del Interior en conjunto con la Universidad de las Ciencias Informáticas deciden realizar el Proyecto de Modernización de los Sistemas de Identificación, Inmigración y Extranjería de Cuba, el cual permite llevar a cabo las funciones de estas direcciones con la calidad requerida. Dicho sistema está compuesto por el Sistema Único de Identificación Nacional y Sistema de Inmigración, Extranjería y Ciudadanía, los cuales cuentan con funcionalidades comunes.

Durante el ciclo de desarrollo de este proyecto se hace necesario el uso de componentes de interfaz de usuario personalizados, los cuales no siempre son provistos por las bibliotecas de componentes de interfaz de usuario existentes; por lo que se hace necesario un equipo especializado en el desarrollo e implementación de las mismas.

El presente trabajo presenta el desarrollo de una biblioteca de componentes comunes de interfaz de usuario con el objetivo de facilitar el trabajo de los desarrolladores, aumentar el tiempo de desarrollo de los módulos proyecto y abstraer a los desarrolladores del proyecto de la implementación de dichos componentes en el Sistema de Identificación, Inmigración y Extranjería de la República de Cuba, garantizando la uniformidad, reutilización y funcionalidad, basada en la tecnología .NET.

Palabras claves: biblioteca, componentes, funcionalidad y uniformidad.

ÍNDICE

RESUMEN	VII
ÍNDICE DE FIGURAS	XIV
ÍNDICE DE TABLAS	XV
INTRODUCCIÓN	1
FUNDAMENTACIÓN TEÓRICA	7
1.1 Introducción	7
1.2 Interfaz de usuario	7
1.2.1 Tipos de interfaz de usuario.....	8
1.3 Biblioteca de componentes de interfaz de usuario	8
1.3.1 Ajax Net Server Control	9
1.3.2 Soluciones existentes	9
1.4 Evaluación de las interfaces de usuario.....	12
1.5 Estándares y tecnologías.....	12
1.5.1 HTML.....	13
1.5.2 CSS	13
1.5.3 JavaScript.....	14
1.5.4 DOM.....	15
1.5.5 AJAX	15
1.5.6 JSON.....	16
1.5.7 JQuery.....	16
1.5.8 Servicios Web.....	17
1.6 Ambiente de desarrollo	18

1.6.1	Metodología de desarrollo	18
	Marco de trabajo de soluciones de Microsoft (MSF, por sus siglas en inglés).....	18
	Modelo de madurez de la capacidad del software (CMMI, por sus siglas en inglés)	19
	MSF for CMMI Process Improvement como base en el desarrollo de la solución	19
1.6.2	Lenguaje de modelado: UML.....	20
1.6.3	Herramienta de Modelado. Altova UModel 2009 Enterprise Edition.....	21
1.6.4	Plataforma de desarrollo.....	22
	Microsoft .NET Framework 4.0.....	22
1.6.5	Entorno de desarrollo integrado (IDE, por sus siglas en inglés).....	23
	Microsoft Visual Studio Team System 2010.....	23
1.6.6	Lenguajes de Programación	24
1.7	Navegador web.....	27
1.7.1	Motor de renderizado de HTML y CSS	27
1.7.2	Intérprete de JavaScript.....	28
1.7.3	Navegadores más usados	29
1.7.4	Tabla comparativa entre navegadores.....	31
1.8	Conclusiones parciales	32
CARACTERÍSTICAS, ANÁLISIS Y DISEÑO DE LA SOLUCIÓN		33
2.1	Especificación de requisitos de software.....	33
2.1.1	Definición de requisitos funcionales	33
2.1.2	Requisitos no funcionales	36
2.3	Arquitectura de la solución.....	38
2.4	Patrones de diseño utilizados	40
2.5	Especificación de clases.....	40

2.5.1 Diagrama de clases de diseño	41
2.6 Descripción de las clases	41
2.7 Conclusiones parciales	45
IMPLEMENTACIÓN Y PRUEBAS	46
3.3 Introducción	46
3.4 Estándares de codificación	46
Capitalización	46
3.5 Implementación	49
3.5.1 Estructura de la biblioteca de componentes.....	49
3.6 Modelo de pruebas	51
3.6.1 Diseño de casos de pruebas.....	51
CPR 1: Incluir componente de entrada de texto.....	52
3.6.2 Pruebas unitarias.....	55
3.6.3 Resultado de las pruebas unitarias.....	56
3.7 Conclusiones parciales	58
CONCLUSIONES	60
RECOMENDACIONES	61
BIBLIOGRAFÍA	62
ANEXOS	64
ANEXO 1. CATÁLOGO DE LOS REQUISITOS FUNCIONALES	64
ANEXO 2. DESCRIPCIÓN DE LOS REQUISITOS FUNCIONALES	67
ANEXO 3. DESCRIPCIÓN DE LOS CLASES	86
ANEXO 4. ITERACIONES DE DESARROLLO.....	99
ANEXO 5. DISEÑO DE CASOS DE PRUEBAS	105

ÍNDICE DE FIGURAS

Figura 1. Fases de MSF para CMMI.20

Figura 2. Vista lógica de la arquitectura de software.38

Figura 3. Diagrama de clases de diseño de la biblioteca de componentes.....41

Figura 4. Estructura de componente de la biblioteca de componentes.....50

Figura 5. Código de prueba unitaria del componente Search.....56

Figura 6. Gráfica de las pruebas realizadas a cada requisito funcional por cada iteración del desarrollo...57

Figura 7. Resultado de la prueba unitaria al componente Search.58

ÍNDICE DE TABLAS

Tabla 1. Navegadores web. Estadísticas de uso mensual.	31
Tabla 2. Comparación entre los navegadores más usados.	31
Tabla 3. Soporte de CSS 1, CSS 2.1 y CSS 3.	32
Tabla 4. Resumen del catálogo de requisitos funcionales.	34
Tabla 5. Descripción del RF4 Incluir componente campo de texto.	36
Tabla 6. Descripción de las propiedades de la clase ILayoutBehavior.	42
Tabla 7. Descripción de los métodos de la clase ILayoutBehavior.	43
Tabla 8. Descripción de las propiedades de la clase IMainInputControl.	43
Tabla 9. Descripción de los métodos de la clase BaseUIToolsBoxControl.	44
Tabla 10. Caso de prueba del requisito funcional Incluir componente TextField.	54
Tabla 11. Pruebas realizadas a cada requisito funcional por cada iteración del desarrollo de la solución.	57

INTRODUCCIÓN

En las aplicaciones web actuales, la calidad de las interfaces de usuario constituye un punto clave para complementar su eficacia. Un indicador para su valoración es la usabilidad que, según Jakob Nielsen¹, se refiere a “la claridad y la elegancia con que se diseña la interacción con un programa de ordenador o un sitio web” (1).

La uniformidad visual de los elementos gráficos favorece a la usabilidad, además de apoyar la consolidación de la imagen empresarial. El uso de pautas para representar un determinado tipo de control o elemento de interfaz contribuye a que el usuario intuya su comportamiento u objetivo independientemente de su contexto, induciendo la tendencia a utilizar los estándares de la web para desarrollar aplicaciones de gestión en medianas y grandes empresas.

Estudios reflejados en el libro “*The Design of Future Things*”² afirman que hasta el 45% del código de una aplicación está dedicado a la interfaz y más de un tercio de los análisis, comparaciones y opiniones de la prensa a su facilidad de uso (1). En el caso específico de los sistemas basados en la web se incrementa la complejidad debido a las notables diferencias a la hora de interpretar los elementos que conforman una vista según el tipo de navegador utilizado por el usuario. Por lo tanto a la hora de desarrollar una aplicación web se hace necesario tener en cuenta desde el primer momento la forma de estandarizar el diseño y la implementación de las interfaces de usuario, con vistas a mantener su uniformidad y eficiencia.

El Ministerio del Interior de la República de Cuba (MININT) en conjunto con la Universidad de las Ciencias Informáticas (UCI), están trabajando en la creación de un conjunto de sistemas con el objetivo de modernizar el trabajo de la Dirección de Inmigración y Extranjería, así como, la Dirección de Identificación y Registro. Con estos propósitos el Centro de Identificación y Seguridad Digital (CISED) de la Universidad de las Ciencias Informáticas y el Ministerio del Interior de la República de Cuba acuerdan crear el Proyecto de Modernización de los Sistemas de Identificación, Inmigración y Extranjería de Cuba.

¹ Ingeniero de Interfaces de Usuario, de origen danés. Obtuvo un Doctorado en Diseño de Interfaces de Usuario y Ciencias de la Computación en la Universidad Técnica de Dinamarca. Ha trabajado en grandes empresas como Bellcore, IBM y Sun Microsystems.

² Escrito por Donald Norman, ha trabajado para Apple como Vice Presidente de Apple's Advanced Technology Group y para Hewlett-Packard.

El sistema deberá ser desplegado en todas las provincias del país. El flujo de información que existe entre las 183 oficinas es muy grande pues se realizan aproximadamente 6000 trámites diarios, que pueden implicar la modificación de más de 6000 personas. Además, la solución que se propone debe soportar el tráfico de imágenes, las cuales consumen un tamaño considerable del ancho de banda disponible. Por las características del sistema y de la información que es gestionada es necesario que el sistema pueda ser accesible en todo momento y en caso de fallas estas deben ser corregidas en el menor tiempo posible. El tiempo de respuesta además debe ser rápido por ser un sistema de vital importancia para el país en aeropuertos y otras entidades que realicen trámites, esto provoca que se generen altos niveles de concurrencia debido a la cantidad de información que es gestionada por el sistema en tiempo real (25).

Dadas estas necesidades se propuso una solución cliente-servidor, con una Arquitectura Orientada a Servicios (SOA, *Service Oriented Architecture*) la cual es una filosofía de diseño que permite un mejor alineamiento de las TIC³ con las necesidades de las empresas, permitiendo a empleados, clientes y socios comerciales responder de forma más rápida y adaptarse adecuadamente a las presiones del mercado. Dicha arquitectura se encuentra representada por 5 capas lógicas: Presentación, Proceso-Servicios, Negocio, Acceso a Datos y Base de Datos. Estas capas cuentan con una independencia lógica entre ellas facilitando la reutilización de la aplicación y disminuyendo en tiempo y esfuerzo los posibles cambios a realizarle a una solución basada en la misma. Además se definió el uso de herramientas y tecnologías de desarrollo basado en .NET debido las facilidades y utilidades que estas proveen.

Se decidió implementar el Sistema de Identificación, Inmigración y Extranjería de la República de Cuba en el cual para dar cumplimiento a las necesidades planteadas se definió el diseño de las interfaces de usuario que se utilizarían en el sistema, pero no se tuvo en cuenta la inclusión de personal especializado a la hora de concebir los equipos de desarrollo para cada uno de los módulos del proyecto. De ahí que fuera necesario buscar una alternativa que facilitara el trabajo de los desarrolladores no especializados y que, al mismo tiempo garantizara la uniformidad, eficiencia y reutilización de los componentes de interfaz de usuario.

Esto provocó la ocurrencia de los siguientes problemas:

³ *Tecnologías de la información y la comunicación.*

- ✓ Poca reutilización de los componentes de interfaz de usuario.
- ✓ Falta de uniformidad en la apariencia y comportamiento de los componentes de interfaz de usuario.
- ✓ Demoras en el proceso de desarrollo por falta de personal especializado, al existir poco dominio de los lenguajes y tecnologías para el desarrollo de las interfaces de usuario.

Por lo tanto se hizo necesario buscar una alternativa para garantizar los parámetros establecidos por el equipo de diseño empleando solamente los miembros del equipo de desarrollo que mayor experiencia tuvieran.

Formulándose entonces el siguiente **problema científico** ¿Cómo garantizar la uniformidad y funcionalidad de los componentes de interfaz de usuario y abstraer a los desarrolladores de la complejidad de la implementación de los mismos en el Sistema de Identificación, Inmigración y Extranjería de la República de Cuba?

El **objeto de estudio** de la presente investigación es el desarrollo de bibliotecas de componentes de interfaz de usuario basado en .NET, y como **campo de acción**: el desarrollo de componentes de interfaz de usuario para aplicaciones web desarrolladas con tecnologías .NET.

En correspondencia con el problema planteado se define el siguiente **objetivo general**: Desarrollar una biblioteca que contenga los componentes comunes para la construcción de las interfaces de usuario del Sistema de Identificación, Inmigración y Extranjería de la República de Cuba, garantizando agilizar el proceso de desarrollo, así como uniformidad y calidad en la presentación.

Se compone de los **objetivos específicos** relacionados a continuación:

- ✓ Desarrollar el marco teórico de la investigación.
- ✓ Describir los requisitos funcionales y no funcionales del sistema para la integración de la biblioteca de componentes de interfaz de usuario.
- ✓ Implementar la biblioteca de componentes de interfaz de usuario.

- ✓ Validar la biblioteca de componentes de interfaz de usuario.

A partir del análisis realizado se plantea como **hipótesis**: El desarrollo de una biblioteca de componentes de interfaz de usuario permitirá garantizar la uniformidad y funcionalidad de los componentes desarrollados, así como abstraer de la implementación de los mismos a los desarrolladores del proyecto.

De la hipótesis anteriormente planteada se obtienen como variables de la investigación:

- ✓ **Variable Independiente**: biblioteca de componentes de interfaz de usuario para aplicaciones web basadas en .NET.
- ✓ **Variables dependientes**: uniformidad, funcionalidad, abstracción de su implementación.

Como **posibles resultados** de esta investigación se espera:

- ✓ La obtención de una biblioteca de componentes comunes para la creación de interfaces de usuarios del Sistema de Identificación, Inmigración y Extranjería de la República de Cuba, abstrayendo a los desarrolladores de la complejidad asociada a la implementación de la misma garantizando la uniformidad y funcionalidad del sistema.

Para garantizar el cumplimiento de los objetivos específicos de la investigación se identifican las siguientes **tareas investigativas**:

- ✓ Estudio de las herramientas, tecnologías y metodologías definidas para el desarrollo del proyecto.
- ✓ Análisis de soluciones semejantes existentes.
- ✓ Análisis de las tecnologías para el enriquecimiento de funcionalidades nativas del navegador web.
- ✓ Determinar los requisitos funcionales y no funcionales sobre la base de las necesidades de desarrollo en el proyecto.
- ✓ Definir la arquitectura a utilizar para la biblioteca de componentes de interfaz de usuario.
- ✓ Desarrollar la biblioteca de componentes de interfaz de usuario.

- ✓ Realizar pruebas unitarias a la biblioteca de componentes de interfaz de usuario.
- ✓ Realizar pruebas de caja negra a la biblioteca de componentes de interfaz de usuario.

Para asistir las tareas de la investigación se emplearán los siguientes **métodos científicos**:

Métodos teóricos:

- ✓ **Histórico – lógico:** Se empleará durante la revisión de la bibliografía utilizada para el estudio de los diferentes estándares y protocolos.
- ✓ **Analítico – sintético:** Se utilizará para la definición de las funcionalidades y caracterización de los mecanismos necesarios para el desarrollo de la biblioteca.

Métodos empíricos:

- ✓ **Entrevista:** Se realizan entrevistas a los miembros del equipo de desarrollo para comprender las necesidades comunes de los componentes de interfaz de usuario.

El presente documento está estructurado en capítulos, enumerados del 1 al 3, los cuales han sido desarrollados partiendo de los estudios realizados y de la observación de las tareas específicas de los procesos que se despliegan.

A continuación se describe el contenido que se abordará en cada uno:

- ✓ **Capítulo 1 Fundamentación Teórica:** Cuenta con un respaldo teórico de los conceptos fundamentales para el entendimiento de la solución. Presenta el estudio realizado de las tecnologías y herramientas para el desarrollo de la biblioteca de componentes de interfaz de usuario.
- ✓ **Capítulo 2 Características, Análisis y Diseño de la solución:** Se realiza un análisis del dominio actual para su comprensión. Se presentan los requisitos funcionales y no funcionales de la solución conjuntamente con sus descripciones, además de una vista general de la propuesta. Se muestran detalladamente los elementos utilizados en el proceso de desarrollo de los componentes de interfaz de usuario de la biblioteca. Se realiza una descripción detallada de la arquitectura, por la cual se rige el desarrollo haciendo énfasis en los puntos cruciales de esta.

- ✓ **Capítulo 3 Implementación y Pruebas:** Se analizan los componentes indispensables para el desarrollo de la solución, diagramas de clases y las relaciones entre los distintos componentes que conforman la aplicación. Se presenta el diagrama de componentes y las pruebas de software aplicadas a la solución.

FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

La necesidad de implementar una solución que resuelva los problemas relacionados con el diseño e implementación de interfaces de usuario en el equipo de desarrollo del Sistema de Inmigración y Extranjería de la República de Cuba, queda reflejada en el presente capítulo; el cual brinda una visión general de los aspectos vinculados con los conceptos necesarios para el desarrollo de una biblioteca de componentes de interfaz de usuario para la plataforma .NET.

Presenta la información obtenida a partir del estudio realizado de los sistemas existentes para el desarrollo de bibliotecas de componentes con el objetivo de obtener conclusiones parciales sobre posibilidades de desarrollo en esta área. Además se exponen los resultados obtenidos a partir del estudio bibliográfico realizado con el objetivo de fundamentar las herramientas y métodos de desarrollo que se emplearán en la solución propuesta.

1.2 Interfaz de usuario

La definición de interfaz de usuario se define como un conjunto de componentes empleados por los usuarios para comunicarse con las computadoras. El usuario dirige el funcionamiento de la máquina mediante instrucciones, denominadas genéricamente entradas. Las entradas se introducen mediante diversos dispositivos, y se convierten en señales electrónicas que pueden ser procesadas por la computadora. Estas señales se transmiten a través de circuitos conocidos como *bus*, y son coordinadas y controladas por la unidad de proceso central y por un soporte lógico conocido como sistema operativo. Una vez que el CPU⁴ ha ejecutado las instrucciones indicadas por el usuario, puede comunicar los resultados mediante señales electrónicas, o salidas, que se transmiten por el bus a uno o más dispositivos de salida (impresora, monitor, etc.) (2).

⁴ *Microprocesador que realiza las funciones aritméticas y lógicas de un ordenador.*

1.2.1 Tipos de interfaz de usuario

Interfaz de usuario web

Con la aparición de la web se hizo posible que cualquier persona pudiera ofrecer información particularizada a los demás y encontrar documentos interactivos sobre cualquier tema, relacionados unos con otros mediante enlaces que permitían saltar de página en página alrededor del mundo.

Las páginas web supusieron la aparición de las interfaces web, interfaces gráficas de usuario con unos elementos comunes de presentación y navegación que pronto se convirtieron en estándares de facto. Este tipo de interfaces sirven de intermediarias entre usuarios genéricos, no acostumbrados generalmente al uso de aplicaciones informáticas, y unos sistemas de información y procesos transaccionales que se ejecutan en un segundo plano, permitiendo la localización de la información deseada, el entendimiento claro de las funcionalidades ofrecidas, la realización práctica de tareas específicas por parte de los usuarios y la navegación intuitiva por las diferentes páginas que forman el sitio web.

Buscando una homogeneidad entre los millones de páginas web que existen actualmente en Internet, el diseño de las mismas ha evolucionado con el tiempo hacia un esquema general perfectamente definido, ofreciendo interfaces bien definidas, con un conjunto de componentes gráficos y funcionales similares que hacen posible que sea cual sea el usuario que accede a un sitio web cualquiera, la comunicación entre ellos sea posible y efectiva.

De esta forma se han definido elementos y agrupaciones de estos que han demostrado su utilidad y su comprensión por los usuarios, entre los se pueden destacar los sistemas de navegación, los dinteles, los pies de página, los formularios de entrada de datos, etc., que normalmente se encuentran en las páginas web, cuyo diseño y funcionalidad son similares en todas ellas (2).

1.3 Biblioteca de componentes de interfaz de usuario

En ciencias de la computación, una biblioteca (o *library*, en idioma inglés) es un conjunto de subprogramas utilizados para desarrollar software. Las bibliotecas de componentes de interfaz de usuario contienen código y datos, que proporcionan servicios a programas independientes, es decir, pasan a

formar parte de estos; permitiendo que el código y los datos se compartan y puedan modificarse de forma modular. Algunos programas ejecutables pueden ser a la vez programas independientes y bibliotecas, pero la mayoría de estas no son ejecutables (3).

Ejecutables y bibliotecas hacen referencias (llamadas enlaces) entre sí a través de un proceso conocido como enlace, que por lo general es realizado por un software denominado enlazador. La mayoría de los sistemas operativos modernos proporcionan bibliotecas que implementan la mayoría de los servicios del sistema. De esta manera, estos servicios se convierten en una "materia prima" que cualquier aplicación moderna espera que el sistema operativo ofrezca. Como tal, la mayor parte del código utilizado por las aplicaciones modernas se ofrece en estas bibliotecas.

1.3.1 Ajax Net Server Control

Los controles de servidor de .NET (*server control*, en inglés) son controles que pueden ser enlazados a un control de origen de datos para facilitar la operación de mostrar y modificar datos en la aplicación web. Están compuestos por la combinación de otros controles web propios de ASP.NET, como los controles *Label* y *TextBox*, en un diseño único (3).

Además de permitir enlazar el control a un conjunto de resultados de datos, permiten personalizar el diseño del control mediante la utilización de plantillas. También proporcionan un modelo cómodo para controlar y cancelar los eventos haciendo uso de *javascript* y de la tecnología AJAX.

1.3.2 Soluciones existentes

En la actualidad existen varias bibliotecas de componentes de interfaz de usuario, cada una de ellas con particularidades. Por lo que se realizó un estudio detallado de las más empleadas (debido a que son las que ofrecen el uso de la tecnología AJAX, la cual es de vital importancia para la interoperabilidad y eficiencia del sistema del proyecto), con el objetivo de analizar sus características y comprobar si ofrecen las funcionalidades necesarias que requiere el Sistema de Inmigración y Extranjería de la República de Cuba.

Telerik

La biblioteca de componentes *Telerik* contiene controles para *Microsoft ASP.NET AJAX* utilizando AJAX como principal tecnología de desarrollo. Presenta beneficios para el aprovechamiento de la industria estándar de la tecnología de *Microsoft AJAX*.

Entre sus principales características se encuentran:

- ✓ Provee mejoras en el rendimiento con respecto a los controles de ASP.NET debido al almacenamiento en caché de los recursos.
- ✓ Es compatible con .Net Framework 3.x y 4.0.
- ✓ Se integra con Microsoft Visual Studio 2010.
- ✓ Proporciona dos tipos de licencia: Una *Open Source* (código abierto) que está diseñada para el público en general, en la que se puede visualizar el código fuente de los controles pero no modificarlo y, por otro lado, existe una licencia comercial para recibir más beneficios y soporte (4).

Dicha licencia comercial posibilita dos variantes, con costos de adquisición como se muestra a continuación:

- ✓ La licencia comercial para *Telerik Premium Collection for .NET* está valorada en \$6.442 USD.
- ✓ La licencia comercial para *Telerik Ultimate Collection for .NET* está valorada en \$8.190 USD.

ASP.NET AJAX Control Toolkit

ASP.NET AJAX Control Toolkit es una biblioteca de componentes de descarga gratuita que contiene más de 40 controles AJAX y componentes que aportan funcionalidad AJAX a ASP.NET con soporte para Framework .NET 3.5. Es desarrollada por un grupo de desarrollo de Microsoft, y por docenas de colaboradores que aportan múltiples controles. Su última versión consta del año 2009, la cual provee soporte para .NET Framework 3.5 usando Microsoft Visual Studio 2008 (5).

Su principal deficiencia es respecto a la carga de datos debido al mal uso que hace del control *UpdatePanel* de ASP.NET el cual hace posible la representación parcial de contenido de una página en el

cliente. Para su utilización hay que tener en cuenta dos elementos importantes que si no se conoce su correcto funcionamiento podrían arruinar la funcionalidad esperada de dicho control. Durante el proceso de la petición asíncrona que desencadena un control *UpdatePanel*, en el servidor se ejecuta todo el ciclo de vida de la página que alberga dicho control. Es decir, que aunque el resultado de la petición sólo sea la actualización de una parte de la página del cliente, en realidad la petición realiza la carga total de los datos de la página web provocando problemas en cuanto a términos de rendimiento y eficiencia debido al volumen de información transmitida en cada petición (5).

1.3.2.1 Necesidad de implementar biblioteca de componentes de interfaz de usuario

A partir del análisis realizado a las diferentes bibliotecas de componentes de interfaz de usuario más empleadas en la actualidad se concluye:

- ✓ No resulta provechoso el uso de las bibliotecas de componentes antes mencionadas pues no cumplen con las necesidades y características principales que se requieren para el Sistema de Inmigración y Extranjería de la República de Cuba.
- ✓ En el caso de la biblioteca de componentes *Telerik* cuenta con licencia con costos de adquisición muy elevados y no es posible extender sus funcionalidades.
- ✓ En cuanto a la biblioteca de componentes *ASP.NET AJAX Control Toolkit*, esta presenta problemas de eficiencia en la carga de datos y con el volumen de información transmitida en cada petición que realiza al servidor.

Debido a lo expuesto anteriormente, y a la carencia de una biblioteca de componentes que cumpla con los controles necesarios para el Sistema de Inmigración y Extranjería de la República de Cuba, se hace inminente el desarrollo de la misma que garantice dichas necesidades y agilice el proceso de desarrollo así como la uniformidad del sistema.

1.4 Evaluación de las interfaces de usuario

“La usabilidad es un concepto complejo, debido a la naturaleza compleja de los seres humanos. Sin llevar a cabo algún tipo de evaluación es imposible saber si el sistema satisface las necesidades de los usuarios. No importa cuánto se enfatice la realización de actividades de usabilidad en el proceso de desarrollo de las interfaces de usuario, no se puede predecir con exactitud el nivel de usabilidad por adelantado. Por esta razón es necesario realizar actividades de usabilidad a lo largo de todo el desarrollo, especialmente al final de cada ciclo iterativo, para conocer qué nivel de usabilidad ha alcanzado el producto, y cuánta mejora será necesaria para cumplir los objetivos de usabilidad establecidos (6).”

Se necesita realizar evaluaciones de usabilidad de las interfaces de usuario en todas las etapas de desarrollo, aunque el grado de formalidad necesaria varía, según los productos de diseño disponibles para evaluación de cada momento, y según las limitaciones de tiempo y recursos.

La evaluación se puede usar según el estándar ISO 13407 [ISO13407, 99] para:

- ✓ Proporcionar retroalimentación que sirva para la mejora del diseño de las interfaces de usuario.
- ✓ Evaluar si los objetivos del usuario y organizacionales se han alcanzado.
- ✓ Monitorizar el uso a largo plazo del producto o sistema.

La evaluación de usabilidad de las interfaces de usuario, es una parte fundamental del enfoque iterativo del desarrollo, porque las actividades de evaluación pueden producir soluciones de diseño para su aplicación en el próximo ciclo de desarrollo o, al menos, un mayor conocimiento sobre la naturaleza del problema de interacción detectado. Por tanto, la evaluación de usabilidad de las interfaces de usuario, no es una prueba con un resultado apto/no-apt, sino que es parte inherente del proceso de desarrollo de las interfaces de usuario (6).

1.5 Estándares y tecnologías

El uso de tecnologías permite que los sistemas sean más interactivos, dinámicos y eficientes. Los estándares web son especificaciones técnicas que definen y describen aspectos de la *World Wide Web*. El término ha sido frecuentemente asociado con la tendencia de aprobar un conjunto de mejores prácticas

estandarizadas para construir sitios web y a la filosofía del diseño y desarrollo de sistemas basados en la web que incluyen esos métodos. Posibilitan aspectos como la interoperabilidad, la accesibilidad y la usabilidad de páginas y sitios web (7).

A continuación se muestran los estándares y tecnologías asociados a la solución propuesta.

1.5.1 HTML

HTML, siglas de *HyperText Markup Language* (Lenguaje de Marcas de Hipertexto), es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de "etiquetas", rodeadas por corchetes angulares (<>). HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un *script*⁵ (por ejemplo JavaScript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML (7).

HTML también es usado para referirse al contenido del tipo de MIME⁶ text/html o todavía más ampliamente como un término genérico para el HTML, ya sea en forma descendida del XML⁷ (como XHTML 1.0 y posteriores) o en forma descendida directamente de SGML (*Standard Generalized Markup Language* o Lenguaje de Marcado Generalizado), como HTML 4.01 y anteriores.

1.5.2 CSS

“CSS (*Cascading Style Sheets* u Hojas de Estilo en Cascada) es la tecnología desarrollada por la *World Wide Web Consortium*⁸ (W3C) con el fin de separar la estructura de la presentación.”

“CSS se utiliza para dar estilo a documentos *HTML* y *XML*, separando el contenido de la presentación. Permite a los desarrolladores controlar el estilo y el formato de múltiples páginas web al mismo tiempo.

⁵ Es un guión o conjunto de instrucciones que permiten la automatización de tareas, creando pequeñas utilidades.

⁶ *Multipurpose Internet Mail Extension* (Extensiones de Correo de Internet Multipropósito), son una serie de especificaciones dirigidas a que se puedan intercambiar a través de Internet todo tipo de archivos (texto, audio, vídeo, etc.) de forma transparente para el usuario.

⁷ *Extensible Markup Language*, por sus siglas en inglés.

⁸ *World Wide Web Consortium*: Comunidad internacional que desarrolla estándares que aseguran el crecimiento de la Web a largo plazo.

Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento.” (3)

Existen varias versiones: CSS1 y CSS2, con CSS3 en desarrollo por el *World Wide Web Consortium* (W3C). Los navegadores modernos implementan CSS1 de manera eficiente, aunque existen pequeñas diferencias de implementación según marcas y versiones de los navegadores. Sin embargo CSS2, está solo parcialmente implementado en los más usados en la actualidad (*Internet Explorer, Mozilla Firefox, Google Chrome, Safari y Opera*).

Las ventajas de utilizar CSS son:

- ✓ Control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.
- ✓ Los navegadores web permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad.
- ✓ El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño (siempre y cuando no se utilice estilo en línea).

1.5.3 JavaScript

“*JavaScript* es un lenguaje de secuencias de comandos desarrollado por *Netscape Communications*⁹ y diseñado para desarrollar aplicaciones de Internet de cliente y de servidor. *Netscape Navigator*¹⁰ está diseñado para interpretar el código *JavaScript* incrustado en las páginas web. *JavaScript* es independiente del lenguaje Java de *SunMicrosystem*¹¹” (8).

El código *JavaScript* puede enlazarse o añadirse a las páginas web proporcionando un control total y dinámico sobre ellas. Además, también permite controlar hasta cierto punto las aplicaciones que lo ejecutan, habitualmente navegadores.

⁹ Es una empresa de software precursora de la naciente *World Wide Web*.

¹⁰ Es un navegador web desarrollado por compañía *Netscape Communications*.

¹¹ Empresa informática de *Silicon Valley*, fabricante de semiconductores y software.

Entre las acciones típicas que se pueden realizar en *JavaScript* existen dos vertientes. Por un lado los efectos especiales sobre páginas web, para crear contenidos dinámicos y elementos de la página que tengan movimiento, cambien de color o cualquier otro dinamismo. Por el otro, permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que se pueden crear páginas interactivas con programas como calculadoras, agendas, o tablas de cálculo.

1.5.4 DOM

DOM es esencialmente una interfaz de programación de aplicaciones que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos.

DOM es una API (Interfaz de Programación de Aplicaciones) para acceder, añadir y cambiar dinámicamente contenido estructurado en documentos con lenguajes como *JavaScript*. Además define la manera en que los objetos y elementos se relacionan entre sí en el navegador y en el documento. Cualquier lenguaje de programación adecuado para el desarrollo de página web puede ser utilizado (7).

1.5.5 AJAX

AJAX son las siglas de *Asynchronous JavaScript And XML*. No es un lenguaje de programación sino un conjunto de tecnologías (*HTML-JavaScript-CSS-DHTML-PHP/ASP.NET/JSP-XML*) que permite hacer páginas web más interactivas. Es una tecnología asíncrona, en el sentido de que los datos adicionales se requieren al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página.

La característica fundamental de AJAX es permitir actualizar parte de una página con información que se encuentra en el servidor sin tener que refrescar completamente la página. De modo similar es posible enviar información al servidor (8).

La complejidad de su uso se encuentra en que se deben dominar varias tecnologías:

- ✓ HTML o XHTML.
- ✓ CSS.

- ✓ DHTML, básicamente se debe poseer un dominio de todos los objetos que proporciona el DOM.
- ✓ XML Para el envío y recepción de los datos entre el cliente y el servidor.
- ✓ ASP.Net/JSP o algún otro lenguaje que se ejecute en el servidor (PHP).

1.5.6 JSON

Conocida por el acrónimo en inglés de *JavaScript Object Notation*, es un formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML. La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a XML en AJAX (8).

JSON se emplea habitualmente en entornos donde el tamaño del flujo de datos entre cliente y servidor es de vital importancia (de aquí su uso por *Yahoo* y *Google*, que atienden a millones de usuarios) cuando la fuente de datos es explícitamente de fiar y donde no es importante el no disponer de procesamiento XSLT¹² para manipular los datos en el cliente, debido a las cuestiones de seguridad que plantea el uso de la función *eval()* y el auge del procesamiento nativo de XML incorporado en los navegadores modernos.

1.5.7 JQuery

JQuery es una biblioteca o *framework* de JavaScript, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas web.

El *framework* JQuery aporta las siguientes ventajas:

- ✓ Ofrece una serie de funcionalidades basadas en *JavaScript* que de otra manera requerirían de mucho más código.
- ✓ Hace transparente el soporte de la aplicación para los navegadores principales.
- ✓ Provee de un mecanismo para la captura de eventos.

¹² Transformaciones XSL es un estándar de la organización W3C que presenta una forma de transformar documentos XML en otros formatos.

- ✓ Provee un conjunto de funciones para animar el contenido de la página en forma sencilla.
- ✓ Integra funcionalidades para trabajar con AJAX (9).

1.5.8 Servicios Web

Son un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos. Las organizaciones OASIS (*Organization for the Advancement of Structured Information Standards*) y W3C son los comités responsables de la arquitectura y reglamentación de los servicios web. Para mejorar la interoperabilidad entre distintas implementaciones de servicios web se ha creado el organismo WS-I, encargado de desarrollar diversos perfiles para definir de manera más exhaustiva estos estándares (10).

Los servicios web ofrecen varias ventajas como son:

- ✓ Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen.
- ✓ Fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.
- ✓ Al apoyarse en HTTP, los servicios web pueden aprovecharse de los sistemas de seguridad *firewall* sin necesidad de cambiar las reglas de filtrado.
- ✓ Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados para proveer servicios integrados.
- ✓ Permiten la interoperabilidad entre plataformas de distintos fabricantes por medio de protocolos estándar y abiertos. Las especificaciones son gestionadas por una organización abierta, la W3C, por tanto no hay secretismos por intereses particulares de fabricantes concretos y se garantiza la plena interoperabilidad entre aplicaciones (10).

1.6 Ambiente de desarrollo

En la informática, obtener un software de calidad implica la utilización de metodologías, tecnologías y herramientas que permitan uniformar la filosofía de trabajo. Para dar solución al problema planteado, se utiliza el ambiente de desarrollo definido por el arquitecto del proyecto Identificación, Inmigración y Extranjería de la República de Cuba de la Universidad de las Ciencias Informáticas, al cual pertenece la presente investigación.

1.6.1 Metodología de desarrollo

Una metodología de desarrollo de software es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información. Se basa en una combinación de los modelos de proceso. Adicionalmente una metodología debería definir con precisión los artefactos, roles y actividades involucradas, junto con prácticas y técnicas recomendadas, guías de adaptación de la metodología al proyecto, guías para uso de herramientas de apoyo, etc. (11).

1.6.1.1 MSF for CMMI.

Marco de trabajo de soluciones de Microsoft (MSF¹³, por sus siglas en inglés)

MSF versión 4.0 es una combinación de un meta-modelo o marco en el que los procesos y dos instancias de ese meta-modelo se construyen. Una de las instancias es un proceso de desarrollo de software ágil y la otra se dirige a un medio ambiente más formal.

Ambos son escenarios impulsados por el desarrollo de software basado en el contexto de los procesos de construcción y mejora de .NET y otras aplicaciones orientadas a objetos. Cada uno incorpora directamente prácticas para el manejo de calidad del servicio, tales como el rendimiento y la seguridad. Ellos usan un enfoque basado en el contexto para guiar la operación del proyecto. Este enfoque ayuda a crear una adaptación ágil o perfeccionamiento formal, proceso que rige la totalidad del proyecto. MSF

¹³ *Microsoft Solution Framework*

proporciona un marco flexible y escalable para satisfacer las necesidades de cualquier organización grande o equipo del proyecto (11).

La orientación de MSF se compone de principios, modelos y disciplinas de la gestión del personal, procesos y elementos de tecnología que la mayoría de los proyectos posee.

Modelo de madurez de la capacidad del software (CMMI¹⁴, por sus siglas en inglés)

CMMI puede ser usada para guiar la mejora de procesos a través de un proyecto, una división o una organización entera. Este modelo ayuda a integrar las funciones tradicionalmente separadas de una organización, establecer objetivos de mejora de procesos y prioridades, proporcionar una guía para los procesos de calidad, y proporcionar un punto de referencia para evaluar los procesos actuales. CMMI clasifica las empresas en niveles de madurez a partir de los procesos que se realizan para producir software. Posee 5 niveles: Inicial, Repetido, Definido, Gestionado y Optimizado (12).

MSF for CMMI Process Improvement como base en el desarrollo de la solución

Con la creación del producto *Visual Studio Team System*, se ha dotado al mundo informático de una potente herramienta para el trabajo con dos metodologías encaminadas a guiar y mejorar el proceso de desarrollo de software en los proyectos productivos: *MSF¹⁵ for Agile Software Development* para el trabajo en entornos que emplean metodologías ágiles y *MSF for CMMI Process Improvement* para el trabajo en entornos con el modelo CMMI.

Es precisamente esta última - *MSF for CMMI Process Improvement* (Marco de trabajo de soluciones de Microsoft para el Modelo de Integración de Madurez de las Capacidades) - la metodología propuesta para guiar el desarrollo de la biblioteca de componentes de interfaz de usuario *UIToolsBox*; pues brinda un entorno flexible y perfectamente ajustable al desarrollo del proyecto permitiendo que los miembros del equipo puedan asumir uno o varios roles en función de las necesidades del ambiente de desarrollo, asumiendo actividades que producen resultados de valor observable como son la documentación, el código fuente o el plan del proyecto.

¹⁴ *Capability Maturity Model Integration*

¹⁵ *Siglas del inglés Microsoft Solutions Framework.*

MSF para CMMI define cinco fases para el ciclo de vida del proyecto que encapsulan conjuntos de *workstreams*¹⁶ y actividades. Cada fase concluye con un punto de control y cada punto de control brinda una oportunidad para autorizar la continuación o suspensión del proyecto. El trabajo en cada fase debe responder una serie de preguntas que irán guiando el paso de una fase a otra (12). (Ver Figura 1. Fases de MSF para CMMI)

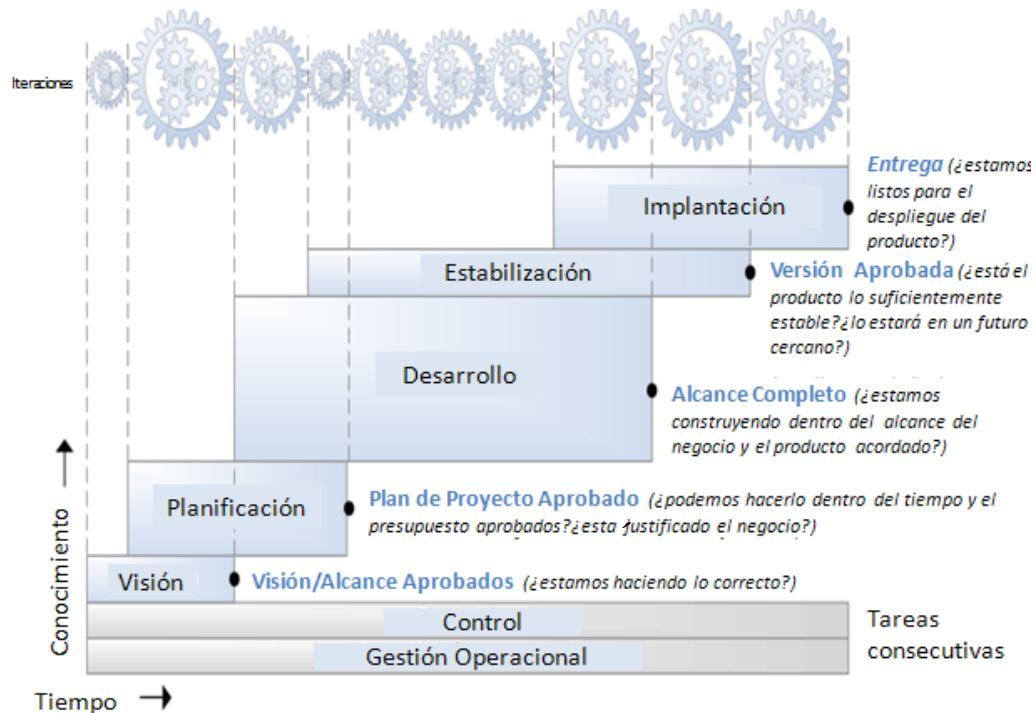


Figura 1. Fases de MSF para CMMI.

1.6.2 Lenguaje de modelado: UML

“UML (*Unified Modeling Language*¹⁷) es un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema. Proporciona una forma estándar de escribir los planos de un sistema, tales como procesos del negocio, funciones del sistema, clases escritas en un lenguaje de programación específico, esquemas de bases de datos y componentes software reutilizables. Es la

¹⁶ Secuencia de actividades de trabajo.

¹⁷ Unified Modeling Language: Lenguaje Unificado de Modelado.

sucesión de una serie de métodos de análisis y diseño orientados a objetos. Está consolidado como el lenguaje estándar en el análisis y diseño de sistemas de cómputo. Mediante UML es posible establecer la serie de requisitos y estructuras necesarias para plasmar un sistema de software previo al proceso intensivo de escribir código” (13).

“UML es apropiado para modelar desde sistemas de información en empresas hasta aplicaciones distribuidas basadas en la Web, e incluso para sistemas empotrados de tiempo real muy exigentes. Es un lenguaje muy expresivo, que cubre todas las vistas necesarias para desarrollar y luego desplegar tales sistemas. Aunque sea expresivo, UML no es difícil de aprender ni de utilizar. UML es sólo un lenguaje y por tanto es tan sólo una parte de un método de desarrollo de software.” (13)

1.6.3 Herramienta de Modelado. Altova UModel 2009 Enterprise Edition.

AltovaUModel es una herramienta completa de desarrollo UML. Soporta todo tipo de diagramas, además de la generación de código y documentación, ingeniería inversa, y características avanzadas de usabilidad. Es compatible con la última especificación de UML 2.3. Es una herramienta de modelado que posee capacidad de compatibilidad con sistemas de control de versiones además de una estrecha integración con *Microsoft Visual Studio*.

Esta herramienta *Case*¹⁸ combina una amigable interfaz visual con funciones de usabilidad que favorecen a los usuarios con las más completas ventajas en el modelado de software. Las características de UModel 2009 para el desarrollo de software basado en las capacidades de modelado avanzado son:

- ✓ Soporte para los 14 tipos de diagramas UML.
- ✓ Generación de código fuente en lenguajes Java, C#, y VB.NET.
- ✓ Crea diagramas de secuencia desde el código fuente de la ingeniería inversa.
- ✓ Generación de documentación personalizable de proyecto.

¹⁸ Siglas del inglés *Computer Aided Software Engineering*.

- ✓ Integración con sistemas de control de versiones.
- ✓ Estrecha integración con *Microsoft Visual Studio* y *Eclipse*¹⁹ (14).

1.6.4 Plataforma de desarrollo

Microsoft .NET Framework 4.0

La plataforma .NET de *Microsoft* es un componente de software que puede ser añadido al sistema operativo *Windows*. Provee un extenso conjunto de soluciones predefinidas para necesidades generales de la programación de aplicaciones, y administra la ejecución de los programas escritos específicamente con la plataforma.

.NET Framework se incluye en *Windows Server 2008*, *Windows Vista* y *Windows 7*. De igual manera, la versión actual de dicho componente puede ser instalada en *Windows XP*, y en la familia de sistemas operativos *Windows Server 2003*. Una versión "reducida" de *.NET Framework* está disponible para la plataforma *Windows Mobile*, incluyendo teléfonos inteligentes (15).

Este *framework* brinda nuevas mejoras y características que lo hacen diferente a sus versiones anteriores, aunque posee la capacidad de funcionar en paralelo con ellas:

- ✓ Mejoras en *Common Language Runtime (CLR)* y la *Biblioteca de Clases Base (BCL)*.
- ✓ Innovaciones en los lenguajes *Visual Basic* y *CSharp*.
- ✓ Servicios de datos de *Windows Communication Foundation (WCF)*, componente que permite crear servicios y aplicaciones que usen protocolo de datos abierto para exponer y usar datos a través de la web.
- ✓ Mejoras en *ASP.NET*.
- ✓ Compatibilidad con formularios para nuevas mejoras de *AJAX Library*.
- ✓ Nuevos controles de línea de negocio.

¹⁹ Es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar proyectos.

- ✓ Mejor funcionamiento del diseñador.
- ✓ Mejoras en *Windows Communication Foundation* (WCF).
- ✓ Detección de servicios y servicio de enrutamiento (16).

1.6.5 Entorno de desarrollo integrado (IDE²⁰, por sus siglas en inglés).

Microsoft Visual Studio Team System 2010

Microsoft Visual Studio 2010 es un conjunto de herramientas que simplifica el desarrollo de aplicaciones para desarrolladores individuales o equipos, haciendo posible la creación de aplicaciones escalables y de alta calidad. Proporciona la capacidad de optimizar su entorno de desarrollo con compatibilidad para varios monitores y permite soportar varias versiones de .NET Framework con una sola herramienta (17).

Presenta novedosas herramientas de pruebas que automatizan la realización de pruebas de la interfaz de usuario en aplicaciones basadas en web, así como pruebas manuales, *Test Professional*, pruebas de rendimiento de web, pruebas de carga, cobertura de código, generación de datos de pruebas; además incluye nuevas características de depuración y diagnóstico.

Incluye una versión renovada del servidor *Team Foundation Server*, el cual ofrece un gestor de pruebas mejorado y nuevas herramientas para el servidor; además de elementos nuevos como *Lab Management*, que permite crear y gestionar entornos virtuales.

1.6.5.1 Control de Versiones

Una de las más importantes cuestiones a las que debe mantenerse atento el equipo de desarrollo de cualquier sistema informático es el control de versiones. Para facilitar el trabajo del equipo en este sentido, existen diversos sistemas automatizados que combinan tecnologías y prácticas a seguir para supervisar la planificación del proyecto y mantener informado al equipo de desarrollo sobre los últimos cambios realizados en los ficheros del proyecto, en particular en el código fuente, la documentación o las páginas web. Este ayuda virtualmente en todos los aspectos al dirigir un proyecto: comunicación entre los

²⁰ *Integrated Development Environment*

desarrolladores, manejo de los lanzamientos, administración de fallos, estabilidad entre el código y los esfuerzos de desarrollo experimental, atribución y autorización en los cambios de los desarrolladores.

1.6.5.1.1 Team Foundation Server 2010

Microsoft Visual Studio Team Foundation Server 2010 es la plataforma de colaboración central en la solución de administración del ciclo de vida de las aplicaciones de *Microsoft*, que automatiza el proceso de entrega de software y permite a las organizaciones administrar proyectos de desarrollo de software de forma eficiente durante su ciclo de vida. *Team Foundation Server 2010* permite al equipo ser más ágil, colaborar de un modo más eficaz y ofrecer software con mayor calidad, así como adquirir y compartir conocimiento institucional.

Los datos y artefactos de proyecto del seguimiento de elementos de trabajo, control de código fuente, compilaciones y herramientas de pruebas se acumulan en un almacén de datos y eficaces informes y paneles proporcionan tendencias históricas, seguimiento completo y visibilidad en tiempo real de la calidad y el progreso respecto al propósito empresarial (18).

1.6.6 Lenguajes de Programación

Un lenguaje de programación es un lenguaje artificial para expresar acciones que pueden ser llevadas a cabo por los ordenadores, así como para la creación de sistemas que controlen el comportamiento físico y lógico de un ordenador en sí. Existen diferentes lenguajes de programación y cada uno con sus características propias que permiten especificar de manera precisa sobre qué datos debe operar una computadora, cómo deben ser almacenados o transmitidos estos datos y qué acciones deben tomar bajo una variada gama de circunstancias.

1.6.6.1 CSharp (C#)

C# es un lenguaje orientado a objetos²¹, diseñado por *Microsoft* para ser utilizado en la plataforma *.NET*. Permite crear aplicaciones cliente de *Windows* tradicionales, servicios web XML, componentes distribuidos, aplicaciones cliente-servidor y aplicaciones de base de datos.

Como lenguaje orientado a objetos admite los conceptos de encapsulación, herencia y polimorfismo. Todas las variables y métodos, incluido el método *main* que es el punto de entrada de la aplicación, se encapsulan dentro de definiciones de clase. La sintaxis de *C#* basada en signos de llave es muy expresiva, pero también es sencilla y fácil de aprender (19).

Las ventajas que ofrece *C#* frente a otros lenguajes de programación son:

- ✓ Declaraciones en el espacio de nombres: al empezar a programar algo, se puede definir una o más clases dentro de un mismo espacio de nombres.
- ✓ Tipos de datos: existe un rango más amplio y definido de tipos de datos que los que se encuentran en C, C++ o Java.
- ✓ Atributos: cada miembro de una clase tiene un atributo de acceso del tipo público, protegido, interno, interno protegido y privado.
- ✓ Pase de parámetros: aquí se puede declarar a los métodos para que acepten un número variable de parámetros. De forma predeterminada, el pase de parámetros es por valor, a menos que se use la palabra reservada "*ref*", la cual indica que se pasa por referencia.
- ✓ Métodos virtuales y redefiniciones: antes de que un método pueda ser redefinido en una clase base, debe declararse como virtual.
- ✓ Propiedades: un objeto tiene intrínsecamente propiedades, y debido a que las clases en *C#* pueden ser utilizadas como objetos, se permite la declaración de propiedades dentro de cualquier clase.

²¹ Se basa en la idea de un objeto, que es una combinación de variables locales y procedimientos llamados métodos que juntos conforman una entidad de programación.

- ✓ Inicializador: un inicializador es como una propiedad, con la diferencia de que en lugar de un nombre de propiedad, un valor de índice entre corchetes se utiliza en forma anónima para hacer referencia al miembro de una clase.
- ✓ Control de versiones: permite mantener múltiples versiones de clases en forma binaria, colocándolas en diferentes espacios de nombres. Esto permite que versiones nuevas y anteriores de software puedan ejecutarse en forma simultánea.

1.6.6.2 ASP.NET

ASP.NET es un *framework* para aplicaciones web desarrollado y comercializado por *Microsoft*. Es usado por programadores para construir sitios web dinámicos, aplicaciones web y servicios web XML con el código mínimo. ASP.NET forma parte de *.NET Framework* y al codificar las aplicaciones ASP.NET tiene acceso a las clases en *.NET Framework*.

El código de las aplicaciones puede escribirse en cualquier lenguaje compatible con el *Common Language Runtime* (CLR), entre ellos Microsoft Visual Basic, C#, JScript.NET y J#. Estos lenguajes permiten desarrollar aplicaciones ASP.NET que se benefician del *Common Language Runtime*, seguridad de tipos, herencia, entre otros.

ASP.NET incluye:

- ✓ Marco de trabajo de página y controles.
- ✓ Compilador de ASP.NET.
- ✓ Infraestructura de seguridad.
- ✓ Funciones de administración de estado.
- ✓ Configuración de la aplicación.
- ✓ Supervisión de estado y características de rendimiento.
- ✓ Capacidad de depuración.
- ✓ Marco de trabajo de servicios Web XML.
- ✓ Entorno de host extensible y administración del ciclo de vida de las aplicaciones.

- ✓ Entorno de diseñador extensible (20).

1.7 Navegador web

Un navegador web es el software o programa que permite ver la información que contiene una página web. Traduce el código HTML en el que está escrita la página y lo muestra en la pantalla permitiendo al usuario interactuar con su contenido y navegar hacia otras páginas o sitios de la red mediante enlaces o hipervínculos. El seguimiento de los enlaces de una página a otra, se llama navegación; que es de donde se origina el nombre de navegador.

1.7.1 Motor de renderizado de HTML y CSS

Un motor de renderizado es un software que toma contenido marcado (como HTML, XML, archivos de imágenes, etc.) e información de formateo (como CSS, XSL²², etc.), lo interpreta de manera visual y luego muestra el contenido ya formateado en la pantalla. Todos los navegadores web incluyen algún motor de renderizado (21).

El término motor de renderizado, (*layout engine* o *rendering engine* en inglés), se hizo popular cuando la *Fundación Mozilla*²³, desarrolló el suyo de manera libre y diferenciado del propio navegador, siendo posible de esta manera reutilizarlo para otros navegadores. Entre los motores de renderizado más utilizados en la actualidad se encuentran:

Gecko

Es un motor multiplataforma y libre originalmente desarrollado por *Netscape*²⁴. Actualmente su desarrollo es gestionado por la *Fundación Mozilla*. Solía ser criticado por su enorme complejidad y uso de memoria. La mayor parte de estas debilidades se subsanaron en la versión 3 de *Mozilla Firefox* con el nuevo motor Gecko 1.9. Con este motor se logró disminuir drásticamente el uso de memoria, incluso superando a los navegadores web *Safari* y *Opera*. Estos cambios permitieron a Gecko pasar el test

²² *Extensible Stylesheet Language, por sus siglas en inglés.*

²³ *Corporación Empresarial que coordina e integra el desarrollo de aplicaciones informáticas relacionadas con Internet como el navegador web Mozilla Firefox.*

²⁴ *Es una empresa de software precursora de la World Wide Web.*

Acid2²⁵ y soportar algunos elementos de CSS 3. Con la llegada de *Mozilla Firefox 3.1*, se implementó Gecko 1.9.1 que incluía *TraceMonkey*, el nuevo intérprete *JavaScript* (8).

Trident

El motor de renderizado *Trident* de *Microsoft* es el motor de renderizado menos compatible con los estándares, lo que durante mucho tiempo ha supuesto que un gran número de páginas web no estén diseñadas para ser compatibles con los estándares, sino para ser compatibles con el navegador web *Internet Explorer*. Es también conocido como MSHTML (8).

La versión 4.0 de *Trident*, que acompaña a *Internet Explorer 8*, fue la primera versión en pasar el test Acid2, (casi 5 años después que *Safari*), cuando todos los demás motores pasan sin problemas el test Acid3²⁶.

WebKit

Es una plataforma para aplicaciones que funciona como base para los navegadores web: *Safari*, *Google Chrome*, *Epiphany*, *Midori* entre otros. Está basado originalmente en el motor de renderizado KHTML del navegador web del proyecto KDE, *Konqueror*. WebKit logra 100/100 en la prueba Acid3 desde el 26 de marzo de 2008 (8).

1.7.2 Intérprete de JavaScript

También conocido como motor de *JavaScript* o implementación de *JavaScript*, es un intérprete informático²⁷ que interpreta el código fuente en lenguaje *JavaScript* y ejecuta un script acorde a las instrucciones dadas.

²⁵ Es una página de prueba publicada para exponer las fallas de la renderización de páginas web en navegadores web y otras aplicaciones que renderizan HTML.

²⁶ Es un sitio hecho por WaSP que pone a prueba los navegadores con los estándares web, especialmente los de DOM y ECMAScript.

²⁷ Un intérprete informático es un programa informático capaz de analizar y ejecutar otros programas, escritos en un lenguaje de alto nivel.

1.7.3 Navegadores más usados

Actualmente el navegador web más utilizado en el mundo es *Internet Explorer*, esta ventaja se debe a que viene integrado con el sistema operativo *Windows*, detrás de este se encuentra *Mozilla Firefox*, a continuación *Google Chrome* y *Safari*. Existen también otros navegadores, *Opera* y *Netscape Navigator* los cuales tienen un uso de menos del 6% en el mercado (8).

A continuación se muestra una breve descripción de cada uno de ellos:

Internet Explorer



El navegador más utilizado actualmente en internet, tiene más del 50% de los usuarios de todo el mundo. Es un complemento indispensable y por defecto del sistema operativo *Windows*, gracias al cual ha logrado su alto nivel de uso. Ha sido criticado por su desempeño, conformidad con estándares y seguridad (22).

Mozilla Firefox



En la actualidad es el navegador más popular de la red, es utilizado por más del 35% de los usuarios de todo el mundo, cada vez se hace más conocido, no se considera un navegador común, sino como una plataforma para aplicaciones que interactúen con Internet. *Mozilla Firefox* es un navegador multiplataforma y está disponible en varias versiones de *Microsoft Windows*, *Mac OS X*²⁸, *GNU/Linux*²⁹ y algunos sistemas basados en Unix. Su código fuente es software libre, publicado bajo una triple licencia GPL/LGPL/MPL (23).

Google Chrome



Es el navegador más joven de la red, para el poco tiempo que lleva (desde septiembre del 2008) se considera un gran logro que ya lo usen más de 20% de los usuarios en el mundo. *Google Chrome* es un navegador web desarrollado por *Google Inc.*³⁰ y compilado con base en componentes de código abierto como el motor de renderizado de *WebKit* y su estructura de desarrollo de aplicaciones (*framework*). Es el tercer navegador más utilizado en Internet, y

²⁸ Es una línea de sistemas operativos computacionales desarrollada, comercializada y vendida por *Apple Inc.*

²⁹ Sistema operativo libre similar a *Unix* que usualmente utiliza herramientas de sistema *GNU*.

³⁰ Es la empresa propietaria de la marca *Google*, cuyo principal producto es el motor de búsqueda del mismo nombre.

actualmente posee una cuota de mercado del 4,63% y está disponible gratuitamente bajo condiciones de servicio específicas. El nombre del navegador deriva del término usado para el marco de la interfaz gráfica de usuario ("*chrome*") (22).

Safari



Es el complemento indispensable para los usuarios de *Mac OS X*, para donde fue ideado inicialmente que iba a ejecutarse y donde están la gran mayoría de usuarios que lo utilizan, aproximadamente el 4% de usuarios existentes en el mundo. Es un navegador que se ha distinguido por su desempeño, velocidad y soporte de los estándares. *Safari* es un navegador web de código cerrado desarrollado por *Apple Inc.*, es un multiplataforma y está disponible para *Microsoft Windows* y *Mac OS X* (22).

Opera



Es reconocido por su velocidad, seguridad, soporte de estándares (especialmente CSS), tamaño reducido, internacionalidad y constante innovación. Implementó ya desde sus primeras versiones la navegación por pestañas, el *Speed Dial*³¹, los movimientos del ratón en la navegación, personalización por sitio, y la vista en miniatura por pestaña, tiene su versión para teléfonos inteligentes y computadores de bolsillo (22).

1.7.3.1 Comparación entre los navegadores más usados

Las estadísticas de uso de cada navegador difieren, en la Tabla 1 se muestran datos recolectados de los *logs*³² en los servidores de más tráfico en la web, por la *World Wide Web Consortium*³³.

2011	Internet Explorer	Firefox	Google Chrome	Safari	Opera
Abril	24.3 %	42.9 %	25.6 %	4.1 %	2.6 %

³¹ Es una modalidad para albergar marcadores en cuando se muestra una nueva pestaña en el navegador mediante la inclusión de capturas y el linkado rápido.

³² Es un registro de actividad producida en un sistema.

³³ La *World Wide Web Consortium*, abreviado *W3C*, es un consorcio internacional que produce recomendaciones para la *World Wide Web*.

Marzo	25.8 %	42.2%	25.0%	4.0%	2.5%
Febrero	26.5 %	42.4%	24.1%	4.1%	2.5%
Enero	26.6 %	42.8%	23.8%	4.0%	2.5%
2010	Internet Explorer	Firefox	Chrome	Safari	Opera
Diciembre	27.5 %	43.5%	22.4%	3.8%	2.2%
Noviembre	28.6 %	44.0%	20.5%	4.0%	2.3%
Octubre	29.7 %	44.1%	19.2%	3.9%	2.2%
Septiembre	31.1 %	45.1%	17.3%	3.7%	2.2%
Agosto	30.7 %	45.8%	17.0%	3.5%	2.3%
Julio	30.4 %	46.4%	16.7%	3.4%	2.3%
Junio	31.0 %	46.6%	15.9%	3.6%	2.1%
Mayo	32.2 %	46.9%	14.5%	3.5%	2.2%
Abril	33.4 %	46.4%	13.6%	3.7%	2.2%
Marzo	34.9 %	46.2%	12.3%	3.7%	2.2%
Febrero	35.3 %	46.5%	11.6%	3.8%	2.1%
Enero	36.2 %	46.3%	10.8%	3.7%	2.2%

Tabla 1. Navegadores web. Estadísticas de uso mensual.

1.7.4 Tabla comparativa entre navegadores.

Los navegadores web presentan estructuras comunes que posibilitan realizar comparaciones entre ellos para obtener resultados que den una idea clara de la compatibilidad, funcionalidad y rapidez con que procesan los datos. A continuación se muestran estadísticas comparativas en las tablas 2 y 3 .

Navegador Web	Versión más usada	Motor HTML	Motor JavaScript
Internet Explorer	IE8	Trident	V8
Internet Explorer	IE9	Trident	Chakra
Firefox	3.5	Gecko	TraceMonkey
Google Chrome	10	WebKit	V8
Safari	4	WebKit	Javascript Nitro
Opera	12	Presto	Carakan

Tabla 2. Comparación entre los navegadores más usados.

Navegador	CSS 1	CSS 2.1	CSS 3
Internet Explorer	Completo desde la versión 6.0	Completo desde la versión 8.0	Prácticamente nulo
Firefox	Completo	Casi completo	Parcial
Safari	Completo	Casi completo	Parcial
Opera	Completo	Casi completo	Parcial
Google Chrome	Completo	Casi completo	Parcial

Tabla 3. Soporte de CSS 1, CSS 2.1 y CSS 3.

1.8 Conclusiones parciales

El estudio de la metodología, tecnologías y herramientas utilizadas para el desarrollo del proyecto, así como las relacionadas con el desarrollo de la biblioteca de componentes de interfaz de usuario *UIToolsBox* permitió profundizar en los principales elementos y los aspectos novedosos a tener en cuenta durante el proceso de desarrollo. Además se intuyó con el análisis de soluciones semejantes disponibles se determinó que no existe ninguna que cumpla con los requisitos establecidos para el desarrollo de las interfaces de usuario del proyecto.

CARACTERÍSTICAS, ANÁLISIS Y DISEÑO DE LA SOLUCIÓN

En el presente capítulo se presenta la solución que se propone para dar respuesta al problema planteado. Se describen los requisitos funcionales y no funcionales con los que debe cumplir la biblioteca de componentes propuesta y se muestra una vista general de la misma.

Además se muestran detalladamente los elementos utilizados en el proceso de desarrollo de cada uno de los componentes de la solución. Se realiza una descripción detallada de la arquitectura, por la cual se rige el desarrollo de la biblioteca de componentes haciendo énfasis en los puntos cruciales de la misma. Además se explican los patrones empleados y se realiza una descripción de los diferentes tipos de clases que son implementadas.

2.1 Especificación de requisitos de software

La especificación de los requisitos del software permite clasificar los requerimientos en su correspondiente categoría y la descripción de cada uno de ellos sin ambigüedades en su relación con el resto.

2.1.1 Definición de requisitos funcionales

Los requisitos funcionales de un software definen qué es lo que el sistema debe hacer. Funcionalidades requeridas y restricciones que debe presentar el sistema que son aprobados en mutuo acuerdo con el usuario final. En el siguiente epígrafe se expone un resumen del catálogo de requisitos funcionales de la biblioteca de componentes de interfaz de usuario *UIToolsBox* como se muestra en la Tabla 4. Para más detalles se puede consultar el *Anexo 1* donde se muestran las especificaciones de cada uno de ellos.

RF1.	Incluir componente para la división virtual de un área en columnas.
RF2.	Incluir componente para la agrupación de elementos.
RF3.	Incluir componente acordeón.
RF4.	Incluir componente campo de texto.

RF5.	Incluir componente para el autocompletamiento de datos.
RF6.	Incluir componente <i>ComboBox</i> .
RF7.	Incluir componente para listar elementos.
RF8.	Incluir componente <i>CheckBox</i> .
RF9.	Incluir componente <i>RadioButton</i> .
RF10.	Incluir componente botón.
RF11.	Incluir componente para la entrada de fechas.
RF12.	Incluir componente para la búsqueda de personas.
RF13.	Incluir componente capturador de imágenes.
RF14.	Incluir componente para la carga de archivos.
RF15.	Incluir componente etiqueta.
RF16.	Incluir componente para mostrar información de una persona.
RF17.	Incluir componente de pestañas.

Tabla 4. Resumen del catálogo de requisitos funcionales.

En el presente epígrafe se muestra la descripción del requisito funcional *Incluir componente campo de texto*, como ejemplo de la tarea de descripción de requisitos realizada, en el *Anexo 2* se muestran las descripciones de los restantes. Para el cual se diseñó la tabla que se muestra a continuación con los siguientes elementos:

Nombre: Nombre de la clase que identifica al componente.

Clasificación: Puede ser de 2 tipos: básico (no presentan comportamiento del lado del cliente) o avanzado (presentan comportamiento del lado del cliente).

Tipo: Puede ser de 3 tipos: entrada de datos (son aquellos componentes que su funcionamiento se centra en la entrada de datos), contenedores (son aquellos componentes que su funcionalidad se basa en que son componentes de salida) o complejos (son aquellos componentes que están formados por otros componentes, tanto de entrada de datos como contenedores).

Descripción: Breve descripción de la funcionalidad del componente.

Propiedades: Propiedades de comportamiento que presenta el componente.

Comportamientos: Comportamientos que contiene al componente.

Relación con otros componentes: Componentes con que se relaciona el componente para su implementación y funcionamiento.

Dependencias: Clases de las cuales el componente dependa para su correcto funcionamiento.

Requisitos no funcionales: Cualidades que el componente debe tener.

Prototipo: Se muestra una vista visual del componente.

RF4. Incluir componente campo de texto.

Nombre	TextField	Clasificación	Básico	Tipo	Entrada de Datos
Descripción	Componente para la entrada texto.				
Propiedades			Comportamientos		
Nombre	Descripción		✓ LayoutBehavior		
Caption	Define el título que identifica el contenido del mismo.				
ReadOnly	Define si está habilitado el control para ser modificado o no.				
Text	Define el texto contenido en el control.				
TextMode	Define el comportamiento del control con respecto a su contenido, puede ser de 3 tipos: <ul style="list-style-type: none"> ✓ <i>SingleLine</i>: el control muestra una línea con el texto contenido. ✓ <i>MultiLine</i>: el control muestra varias líneas en correspondencia con el texto contenido. 				

	<ul style="list-style-type: none"> ✓ <i>Password</i>: el control visualiza el texto contenido mediante el uso de asteriscos. 	
Relación con otros componentes		
-		
Dependencias		
<ul style="list-style-type: none"> ✓ UIToolsBox.Common.Behaviors.StyleBehavior. ✓ UIToolsBox.Common.Behaviors.BaseUIToolsBoxControl. ✓ UIToolsBox.Common.IMainInputControl. 		
Requisitos no funcionales		
El título debe mostrarse acorde a las pautas definidas en el documento de pautas de diseño del proyecto (24) [página 18].		
Prototipo		
<p>Correo:</p> <input style="width: 100px; height: 20px;" type="text"/>		

Tabla 5. Descripción del RF4 Incluir componente campo de texto.

2.1.2 Requisitos no funcionales

Según la IEEE *Standard Glossary of Software Engineering Terminology* los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto. A continuación se muestran los requerimientos no funcionales definidos para la biblioteca de componentes de interfaz de usuario *UIToolsBox*.

Usabilidad

RnF1. La biblioteca de componentes de interfaz de usuario *UIToolsBox* podrá ser utilizada por cualquier desarrollador con las siguientes características:

a) Conocimientos básicos de ASP.NET y C#.

RnF2. Los nombres de las propiedades de los componentes de interfaz de usuario se establecerán de manera que resulte de fácil comprensión para los desarrolladores.

Interfaz

RnF1. Interfaz accesible e intuitiva, el manejo de los componentes de interfaz de usuario debe ser lo más intuitivo posible de manera que sean muy claras las acciones que se puedan realizar con ellos y la manera de hacerlas.

RnF2. Interfaz consistente con las pautas de diseño definidas para los sistemas del proyecto.

RnF3. Internacionalización de la interfaz.

RnF4. Consistencia de la aplicación entre los distintos navegadores. Debe visualizarse y manejarse de la misma forma en los navegadores más comunes agrupados a continuación según su motor/intérprete de HTML, CSS y *JavaScript*:

a) Trident (*Internet Explorer* a partir de la versión 8)

b) WebKit (*Safari*, *Google Chrome*)

c) Gecko (*Mozilla Firefox* superiores a la versión 3.6)

2.3 Arquitectura de la solución

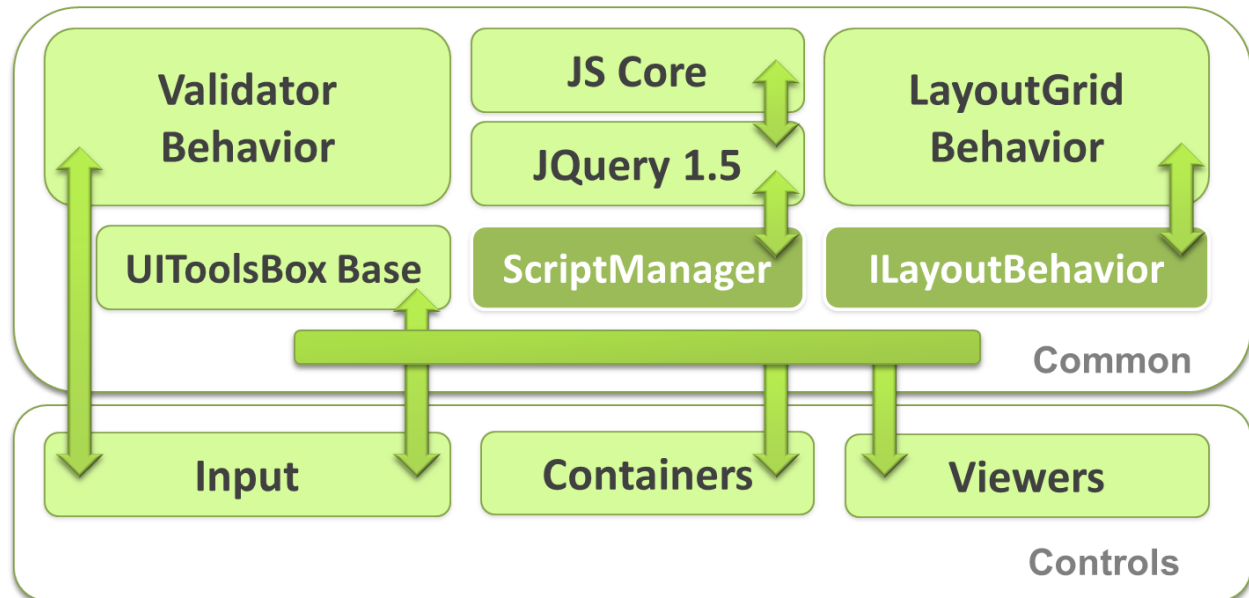


Figura 2. Vista lógica de la arquitectura de software.

La arquitectura de la solución está constituida por dos capas. Emplea el uso de la tecnología AJAX para .NET que consta de controles de interfaz de usuario. En la primera de las capas denominada *Common* se agrupan los siguientes elementos:

- ✓ *Validator Behavior*: Se definen las validaciones comunes para todos los componentes de interfaz de usuario.
- ✓ *JSCore*: Fortalece la implementación de código *JavaScript* para el desarrollo de los componentes de interfaz de usuario añadiendo nuevas funcionalidades *JavaScript*.
- ✓ *JQuery 1.5*: Contiene la biblioteca *JavaScript* de *JQuery* en su versión 1.5.
- ✓ *ScriptManager*: Funciona como el núcleo *JavaScript* que permite el uso del *framework* de *JQuery* en su versión 1.5.
- ✓ *LayoutGrid Behavior*: Es una clase donde se definen las propiedades comunes de posicionamiento de los componentes de interfaz de usuario.

- ✓ *UIToolBox Base*: Contiene un conjunto de funcionalidades *JavaScript* comunes para los componentes de interfaz de usuario con el objetivo de facilitar y servir como base al desarrollo de los mismos.
- ✓ *ILayoutBehavior*: Es una interfaz donde se cargan los estilos visuales necesarios para la biblioteca de componentes de interfaz de usuario.

En la segunda capa llamada *Controls* se agrupan los siguientes elementos:

- ✓ *Input*: Formado por los componentes de interfaz de usuario de tipo entrada de datos, entre los que se encuentran:
 - ✓ Componente campo de texto (*TextField*).
 - ✓ Componente de autocompletamiento de datos (*AutoComplete*).
 - ✓ Componente *ComboBox*.
 - ✓ Componente *CheckBox*.
 - ✓ Componente *RadioButton*.
 - ✓ Componente botón (*Button*).
 - ✓ Componente para la entrada de fechas (*DatePicker*).
 - ✓ Componente para la carga de archivos (*FileUpload*).
 - ✓ Componente para la búsqueda de personas (*Search*).
 - ✓ Componente para la captura de imágenes (*PhotoCapture*).
- ✓ *Containers*: Formado por los componentes de interfaz de usuario utilizados como contenedores de datos, entre los que se encuentran:
 - ✓ Componente para la división virtual de un área en columnas (*LayoutGrid*).
 - ✓ Componente para la agrupación de elementos (*LayoutPanel*).
 - ✓ Componente acordeón (*Accordion*).

- ✓ **TabControlViewers:** Formado por los componentes de interfaz de usuario que se utilizan para mostrar datos, entre los que se encuentran:
 - ✓ Componente para listar elementos (*ListView*).
 - ✓ Componente para mostrar información de una persona (*PersonStripe*).
 - ✓ Componente etiqueta (*Label*).

2.4 Patrones de diseño utilizados

Encapsulación: propone esconder algunos componentes, permitiendo sólo accesos estilizados al objeto. Se hace uso de este patrón en casi todas las clases que componen la solución, permitiendo que estas solo posean como elementos públicos aquellos que son exclusivamente necesarios (24).

Subclase: propone heredar miembros por defecto de una superclase, seleccionando la implementación correcta a través de resoluciones sobre qué implementación debe ser ejecutada. Se puede encontrar este patrón con más fuerza en las entidades de negocio que por su conceptualización las funciones y la información que almacenan pueden estar diferenciadas en cierta medida (24).

Fábrica: provee de una interfaz para crear familias de objetos relacionados o dependientes sin especificar los tipos concretos de clases. Su uso se encuentra centrado a la creación de los conectores correspondientes al acceso a datos que se esté utilizando, así como en la obtención de los servicios a utilizar (24).

2.5 Especificación de clases

Para el desarrollo de los componentes de interfaz de usuario se hace necesaria la implementación de las clases que corresponden a cada capa de la arquitectura propuesta. Estas son fundamentalmente las clases comunes e interfaces.

2.5.1 Diagrama de clases de diseño

El diagrama de clases del diseño representa la interacción de las clases de los componentes y las clases comunes que fortalecen dichos componentes mediante el uso del lenguaje *JavaScript*. (Ver Figura 3 Diagrama de clases del diseño)

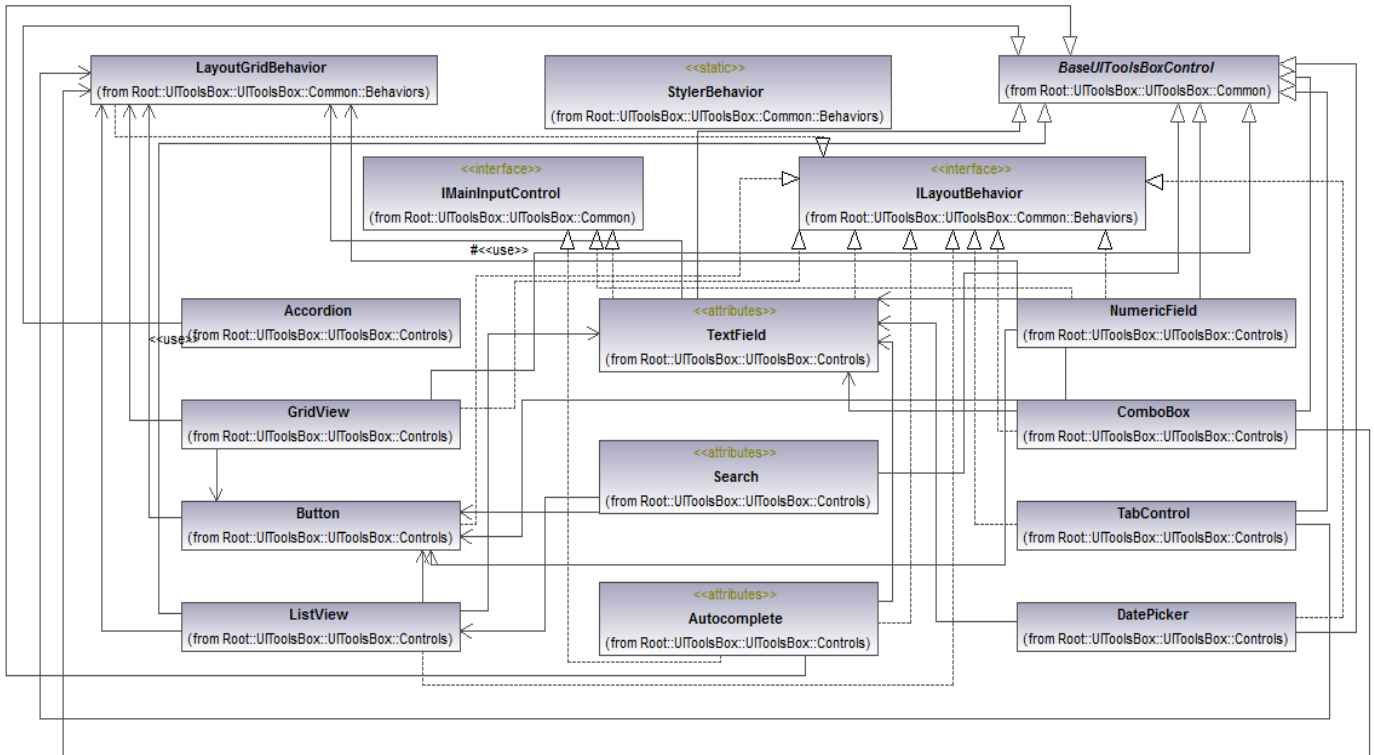


Figura 3. Diagrama de clases de diseño de la biblioteca de componentes.

2.6 Descripción de las clases

“Una clase es una construcción que permite crear tipos personalizados propios mediante la agrupación de variables de otros tipos, métodos y eventos” (24).

Para la propuesta de solución se definieron disímiles tipos de clases, estas son las clases comunes e interfaces. A continuación se muestra las descripciones de las clases `ILayoutBehavior`, `IMainInputControl`

y *BaseUIToolsBoxControl* así como las propiedades comunes para el posicionamiento de los controles en el área de diseño (Ver *Tabla 6, 7, 8 y 9*). Además del tipo de dato que manejan y una breve descripción de los mismos.

Descripción de la clase *ILayoutBehavior*

Nombre		ILayoutBehavior
Descripción		Es una clase interfaz donde se definen las propiedades para el posicionamiento de los controles en el área de diseño.
Propiedad	Tipo	Descripción
ShowGrid	bool	Muestra columnas del área de diseño facilitando el posicionamiento de los controles.
Columns	int	Define el ancho del control, similar a la propiedad width de ASP.NET donde cada columna representa 20 píxeles.
Append	int	Permite espacios a la izquierda del componente al que se le ha definido.
Prepend	int	Permite espacios a la derecha del componente al que se le ha definido.
SpaceTop	bool	Define un espaciado por encima del control de 20 píxeles.
SpaceBottom	bool	Define un espaciado por debajo del control de 20 píxeles.

*Tabla 6. Descripción de las propiedades de la clase *ILayoutBehavior*.*

Nombre	ILayoutBehavior
Descripción	Es una clase interfaz donde se declaran los métodos necesarios para el posicionamiento de los controles en el área de diseño.

Visibilidad	Método	Argumentos	Descripción
Público	LayoutGridBehavior	-	Es el constructor de la clase que por defecto oculta las columnas del área de diseño.
Público	GetBehavior	string	Acceder al componente en base de los estilos definidos.

Tabla 7. Descripción de los métodos de la clase ILayoutBehavior.

Descripción de la clase IMainInputControl

Nombre		IMainInputControl	
Descripción		Es una clase interfaz donde se declaran propiedades comunes para los campos de entrada de datos.	
Propiedad	Tipo	Descripción	
Caption	string	Texto a mostrar para identificar al control.	
Text	string	Texto a mostrar como valor del campo de entrada de datos.	
TextMode	enum	Define los comportamientos para el campo de entrada de datos (<i>Multiline, Password, Singleline</i>)	
ReadOnly	bool	Define si es un campo de sólo lectura o no.	

Tabla 8. Descripción de las propiedades de la clase IMainInputControl.

Descripción de la clase BaseUIToolsBoxControl.

Nombre	BaseUIToolsBoxControl
---------------	-----------------------

Descripción		Es una clase abstracta donde se definen los métodos comunes que emplean utilizados por los componentes de interfaz de usuario.	
Visibilidad	Método	Argumentos	Descripción
Protegido	Render	System.Web.UI.HtmlTextWriter	Carga los estilos visuales del componente.
Protegido	OnPreRender	-	Permite cambiar valores de la variable que contendrá los estilos visuales antes de la carga de los datos del componente.
Público	GetScriptDescriptors	-	Devuelve uno o más objetos de tipo ScriptDescriptor donde cada objeto tiene la información sobre el componente del lado del cliente que se requiera.
Público	GetScriptReferences	-	Permite el acceso a las referencias a los archivos script para que puedan ser accedidos por los Ajax server controls.

Tabla 9. Descripción de los métodos de la clase *BaseUIToolsBoxControl*.

El resto de las descripciones de las clases se muestran en el *Anexo 3* de este documento.

2.7 Conclusiones parciales

Se analizaron las necesidades del proyecto para el desarrollo de las interfaces de usuario, a partir de las cuales se determinaron los requisitos a tener en cuenta para la implementación. Además se definió una arquitectura basada en servicios que garantiza la flexibilidad de la ejecución de la propuesta de solución, lográndose una independencia entre las capas definidas y extensibilidad necesaria en la propuesta.

IMPLEMENTACIÓN Y PRUEBAS

3.3 Introducción

En el presente capítulo se hace alusión al estándar de código seleccionado para su uso en la implementación de la solución, se muestra el diagrama de componentes para la implantación de la solución. Además, se describirá la validación y verificación del cumplimiento de los requerimientos convenidos mediante la aplicación de métodos de pruebas que garanticen la calidad de la solución haciendo uso de las pruebas unitarias que en gran medida garantizan la corrección de errores en la codificación de la aplicación.

3.4 Estándares de codificación

Para lograr una buena calidad en el software es necesario que en su realización se tengan presentes técnicas de codificación sólidas apoyadas por buenas prácticas de programación. Para una mejor comprensión del código se definieron varios estándares basados en diversas reglas, que junto a las revisiones, logra alcanzar un mejor rendimiento para determinada aplicación. Con el uso de estándares se logra elevar la capacidad de mantenimiento del código, sirve como punto de referencia para los programadores manteniendo un estilo de programación y ayuda a mejorar el proceso de codificación haciéndolo en gran medida eficiente y en muchos casos reutilizables.

A continuación se muestran ejemplos de estándares que se tuvieron en cuenta durante el desarrollo de la biblioteca de componentes de interfaz de usuario.

Capitalización

Se podrán utilizar 3 convenios para la capitalización de los identificadores:

Pascal: La primera letra en el identificador y la primera letra de cada subsiguiente palabra concatenada se capitalizan. Puede utilizar los identificadores de *Pascal Case* en caso de 3 o más caracteres.

Ejemplo: BackColor.

Camello: La primera letra en el identificador está en minúscula y la primera letra de cada subsiguiente palabra concatenada en mayúscula.

Ejemplo: backColor.

Mayúscula: Todas las letras en el identificador se capitalizan. Esta convención se utilizará sólo para los identificadores que tienen 2 o menos caracteres.

Ejemplo: System.IO

System.Web.UI

Sensibilidad a mayúsculas

- ✓ No se deberá crear 2 *namespaces* que se diferencien sólo en el uso de las mayúsculas.

Ejemplo: namespace Ao.Entities.

Namespace ao.entities.

- ✓ No crear funciones con nombres de parámetros que se diferencian sólo en el uso de la mayúscula.

Ejemplo: void MyFuntion(string a, string A)

- ✓ No se deberá crear *namespaces* con nombres de clases que se diferencien sólo en el uso de las mayúsculas.

Ejemplo: System.Windows.Form.Point p

System.Windows.Form.POINT p

- ✓ No crear clases con propiedades que se diferencien sólo en el uso de las mayúsculas.

Ejemplo: int Color {get, set}

- ✓ No crear clases con métodos que se diferencien sólo en el uso de las mayúsculas.

*Ejemplo: voidcalculate()
voidCalculate()*

Abreviaturas

- ✓ No se deberá usar abreviaturas o contracciones como parte del nombre de un identificador.

Ejemplo: Se deberá usar GetWindows en lugar de GetWin.

- ✓ Cuando sea apropiado se utilizarán las siglas para remplazar frases o nombres largos.

Ejemplo: UI por User Interface.

Evitando confusión de nombre y tipo

- ✓ Utiliza nombres que describan a sus identificadores en vez de nombres que describen el tipo de identificador.

Ejemplo en C# no utilizar: voidWrite(floatfloatValue)

Regla para declarar las clases

- ✓ Hacer uso económico de las abreviaturas.
- ✓ No hacer uso del *underscore* (_).

Regla para declarar las interfaces

- ✓ Declarar una clase con la letra "I" al inicio para indicar que es una clase de tipo interfaz.
- ✓ Utilizar el mismo nombre cuando defina una pareja de clase/interface donde la clase es un estándar de implementación de la interface.

3.5 Implementación

La Implementación es el flujo de trabajo donde se desarrolla la solución en términos de componentes, los cuales pueden ser ejecutables, ficheros de código fuente, *scripts*, etc. En este flujo de trabajo se desarrolla la arquitectura y la solución como un todo, definiendo y aplicando patrones y estándares para la codificación con el fin de alcanzar una óptima organización del código. El diagrama de componentes conforma el modelo de implementación; este describe la organización y dependencia de cada uno de los componentes implementados.

3.5.1 Estructura de la biblioteca de componentes

Los diagramas de componentes se encargan de mostrar las organizaciones y dependencias lógicas que existen entre los componentes de software, sean estos componentes de código fuente, binarios o ejecutables. Este diagrama contiene componentes y relaciones entre ellos que a su vez son utilizados para agrupar los elementos de modelado. A continuación se muestra el diagrama de componentes de la biblioteca de componentes de interfaz de usuario, el cual presenta una estructura que es común a cada uno de los componentes del paquete (24).

A partir del siguiente diagrama que muestra la estructura de la biblioteca de componentes se describen las diferencias dependencias y funcionalidades de la misma.

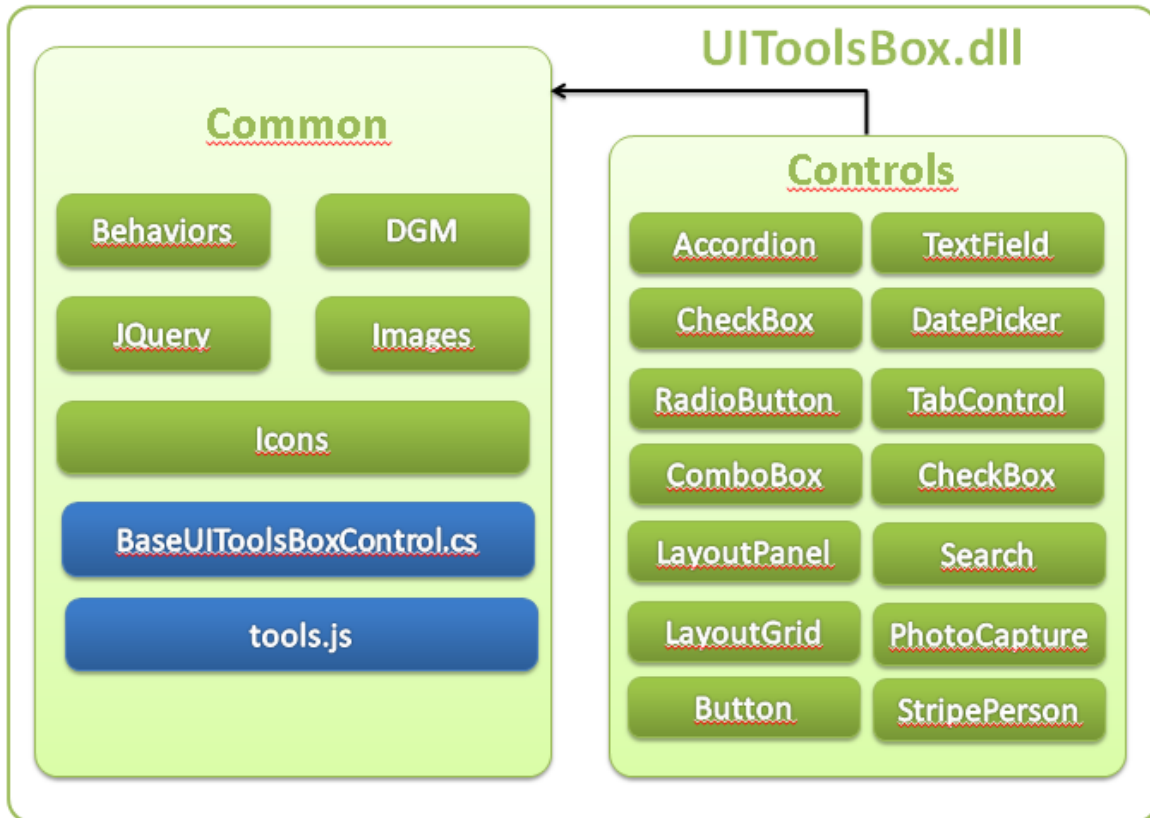


Figura 4. Estructura de componente de la biblioteca de componentes.

En el mismo se muestran los paquetes que forman parte de la arquitectura de la solución, anteriormente mencionada. En el paquete *Controls* están contenidos todos los componentes que conforman la biblioteca *UIToolsBox* los cuales implementan los comportamientos y funcionalidades de las clases incluidas en el paquete *Common*, a continuación se realiza una breve descripción de las mismas:

Behaviors: Es donde se definen los diferentes comportamientos de posicionamiento comunes para los componentes, entre los que se encuentran: *ShowGrid*, *Columns*, *Append*, *Prepend*, entre otros.

DGM: Funciona como una biblioteca de comunicación de dispositivos empleada fundamentalmente para los componentes *PhotoCapture* y el capturador de huellas.

JQuery: Contiene el *framework JavaScript*, *JQuery* en su versión 1.5, el cual fue descrito con anterioridad.

Images: Contiene las imágenes comunes para su reutilización por los diferentes componentes.

Icons: Contiene los iconos comunes para su reutilización por los diferentes componentes.

BaseUIToolsBoxControl.cs: Es la clase base para los componentes que tienen comportamiento del lado del cliente.

tools.js: Contiene funcionalidades comunes para los componentes de interfaz de usuario, facilitando el tiempo de implementación de los mismos.

3.6 Modelo de pruebas

Durante todas las etapas del desarrollo de software las pruebas son fundamentales, a partir de ellas es posible controlar que los productos cumplan con los requisitos de operabilidad, además de verificar y revelar la calidad de un producto de software. Las pruebas constituyen una actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan o almacenan los resultados y se realiza una evaluación de algún aspecto de los mismos.

El objetivo de esta actividad es verificar que los requerimientos de software han sido satisfechos a través de pruebas de unidad, carga y aceptación.

3.6.1 Diseño de casos de pruebas

El diseño de casos de prueba permite la confección de los distintos casos de prueba según la técnica o técnicas identificadas previamente. Debe ir acompañado del resultado que ha de producir el software al ejecutar dicho caso para detectar un posible fallo en el programa. Los casos de prueba determinan un conjunto de entradas, condiciones de ejecución y resultados esperados para un objetivo particular. Cada técnica de pruebas proporciona unos criterios distintos para generar estos casos o datos de prueba (27).

A continuación se muestra el caso de prueba aplicado al requerimiento funcional *Incluir componente campo de texto*. (Ver Tabla 10) El resto de los casos de prueba aplicados a la biblioteca de componentes se encuentran en el Anexo 4.

CPR 1: Incluir componente de entrada de texto.

Condiciones de ejecución

- ✓ Se debe establecer el valor de la propiedad *ValidationTypes* por el correspondiente a cada escenario (*NoValidation, Text, Number, Email, IntNumber, Date*).

Escenario	Descripción	Text	Respuesta del sistema	Flujo central
EC 1.1 Introducir valor en el campo de datos de tipo texto.	El usuario introduce valor de cualquier tipo en el campo de datos del componente.	V/ oficial2011. V/ 15/12/Cuba.	Se muestra el valor introducido en el campo de texto.	-Establecer propiedad ValidationTypes con el valor: NoValidation .
EC 2.1 Introducir valor en el campo de datos de tipo texto correctamente.	El usuario introduce datos de tipo texto correctamente en el campo de datos del componente.	V/ oficial interno.	Se muestra el valor introducido en el campo de texto.	-Establecer propiedad ValidationTypes con el valor: Text .
		V/ Ciudadanía.		
EC 2.2 Introducir valor en el campo de datos de tipo texto incorrectamente.	El usuario introduce datos incorrectos en el componente.	I/ emigrante55 I/ T5oficina	Se muestra un mensaje de error: Campo no válido .	-Establecer propiedad ValidationTypes con el valor: Text .
EC 3.1 Introducir valor en el campo de datos de tipo numérico y enteros correctamente.	El usuario introduce datos de tipo numérico correctamente en el campo de datos del componente.	V/ 545 V/ 8789	Se muestra el valor introducido en el campo de texto.	-Establecer propiedad ValidationTypes con el valor: Number .

<p>EC 3.2 Introducir valor en el campo de datos de tipo numérico y enteros incorrectamente.</p>	<p>El usuario introduce datos incorrectos en el componente.</p>	<p>I/ 878,5</p>	<p>Se muestra un mensaje de error: Campo no válido.</p>	<p>-Establecer propiedad ValidationTypes con el valor: Number.</p>
<p>EC 4.1 Introducir valor en el campo de datos de tipo correo correctamente.</p>	<p>El usuario introduce datos con el formato estándar del correo correctamente en el campo de datos del componente.</p>	<p>V/ rperez@uci.dev V/ pedro75@gmail.com</p>	<p>Se muestra el valor introducido en el campo de texto.</p>	<p>-Establecer propiedad ValidationTypes con el valor: Email.</p>
<p>EC 4.2 Introducir valor en el campo de datos de tipo correo incorrectamente.</p>	<p>El usuario introduce datos incorrectos en el componente.</p>	<p>I/ 25rperez@uci.dev I/ pedro.uci.dev</p>	<p>Se muestra un mensaje de error: Campo no válido.</p>	<p>-Establecer propiedad ValidationTypes con el valor: Email.</p>
<p>EC 5.1 Introducir valor en el campo de datos de tipo numérico correctamente.</p>	<p>El usuario introduce datos de tipo numérico correctamente en el campo de datos del componente.</p>	<p>V/878.5 V/ 940,48</p>	<p>Se muestra el valor introducido en el campo de texto.</p>	<p>-Establecer propiedad ValidationTypes con el valor: IntNumber.</p>
<p>EC 5.2 Introducir valor en el campo de datos de tipo numérico incorrectamente.</p>	<p>El usuario introduce datos incorrectos en el componente.</p>	<p>I/número878.5 I/878^5</p>	<p>Se muestra un mensaje de error: Campo no válido.</p>	<p>-Establecer propiedad ValidationTypes con el valor: IntNumber.</p>
<p>EC 6.1 Introducir valor en el campo</p>	<p>El usuario introduce datos con el formato</p>	<p>V/15/12/2010</p>	<p>Se muestra el valor introducido en el</p>	<p>-Establecer propiedad ValidationTypes con el</p>

de datos de tipo fecha correctamente.	fecha correctamente en el campo de datos del componente.	V/10/02/2011	campo de texto.	valor: Date .
EC 6.2 Introducir valor en el campo de datos de tipo fecha incorrectamente.	El usuario introduce datos incorrectos en el componente.	I/12/15/2010 I/20/01/201	Se muestra un mensaje de error: Campo no válido .	-Establecer propiedad ValidationTypes con el valor: Date .

Descripción de las variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Text	Campo de texto	Sí	<p>Las reglas dependen de la propiedad ValidationTypes que se describen a continuación:</p> <ul style="list-style-type: none"> -NoValidation: permite la entrada de cualquier valor. -Text: sólo permite entrada de texto, no incluye cadenas de texto con valores de tipo numérico. -Number: sólo permite valores de entrada de tipo numérico. Estos números deben ser enteros, no permite números decimales. -IntNumber: sólo permite valores de entrada de tipo numérico. Acepta números enteros y decimales. Permite para los valores decimales el uso de la coma o punto. -Date: sólo permite valores de entrada de tipo fecha con el formato DD/MM/AAAA. -Email: sólo permite valores de entrada con el formato estándar de tipo email (correo).

Tabla 10. Caso de prueba del requisito funcional Incluir componente campo de texto.

3.6.2 Pruebas unitarias

Las pruebas unitarias constituyen una herramienta importante para el personal de pruebas, sobre todo para los desarrolladores. Aunque el tamaño de la "unidad" que se prueba puede variar, la herramienta para pruebas del *Microsoft Visual Studio 2010* permite generar pruebas unitarias específicas para métodos y funcionalidades.

A continuación se muestran ilustraciones con el código de los métodos que se le realizaron pruebas unitarias para verificar su correcto funcionamiento. Las pruebas se le aplican al componente buscador de personas. A cada funcionalidad de este se le generan los resultados esperados y se comparan con los resultados arrojados de la ejecución del componente.

Se realizaron tres iteraciones de pruebas en las cuales se encontraron y corrigieron errores llegando a un resultado positivo en todas las funcionalidades. El resultado de la pruebas se muestran en la *Figura 4*. Para ver el código de las restantes pruebas unitarias realizadas a otras funcionalidades referirse al *Anexo 4*.

```
[TestMethod()]
[HostType("ASP.NET")]
[AspNetDevelopmentServerHost("D:\\Proyecto\\repo\\SUIN\\SUIN\\DevTarjetaMenor\\source\\Suin", "/")]
[UrlToTest("http://localhost:53040/")]

public void GetPersonByIdentityNumberTest()
{
    SearchService target = new SearchService();
    string NumberIdentity = Convert.ToString(87100915465);
    string expected = "DUEÑAS";
    string actual;
    actual = target.GetPersonByIdentityNumber(NumberIdentity).PFirstSurname;
    Assert.AreEqual(expected, actual);
}

[TestMethod()]
[HostType("ASP.NET")]
[AspNetDevelopmentServerHost("D:\\Proyecto\\repo\\SUIN\\SUIN\\DevTarjetaMenor\\source\\Suin", "/")]
[UrlToTest("http://localhost:53040/")]
public void GetNationalPersonByIdTest()
{
    SearchService target = new SearchService();
    Guid PersonId = new Guid("9910ebc0-97cd-7643-ad8c-0ef3e8c57f0c");
    string expected = "RAFAEL";
    string actual;
    actual = target.GetNationalPersonById(PersonId).PSecondName;
    Assert.AreEqual(expected, actual);
}
```

Figura 5. Código de prueba unitaria del componente para la búsqueda de personas.

3.6.3 Resultado de las pruebas unitarias

En las 5 iteraciones de desarrollo se realizaron un total de 7 iteraciones de pruebas en las cuales se encontraron y corrigieron errores llegando a un resultado positivo en todas las funcionalidades (Ver Anexo 4 que muestra las iteraciones de desarrollo realizadas). El resultado de la prueba se muestra a continuación en la Tabla 11 y mediante el uso de una gráfica (Ver Figura 6). Para ver el código de las restantes pruebas unitarias realizadas a otras funcionalidades referirse a Anexo 5.

ITERACIONES	1	1,1	2	3	4	4,1	5
Requisitos No Implementados	9	9	8	6	4	4	0
Requisitos que No Pasaron	1	0	0	0	2	0	0
Requisitos que Pasaron	7	8	9	11	11	13	16

Tabla 11. Pruebas realizadas a cada requisito funcional por cada iteración del desarrollo de la solución.

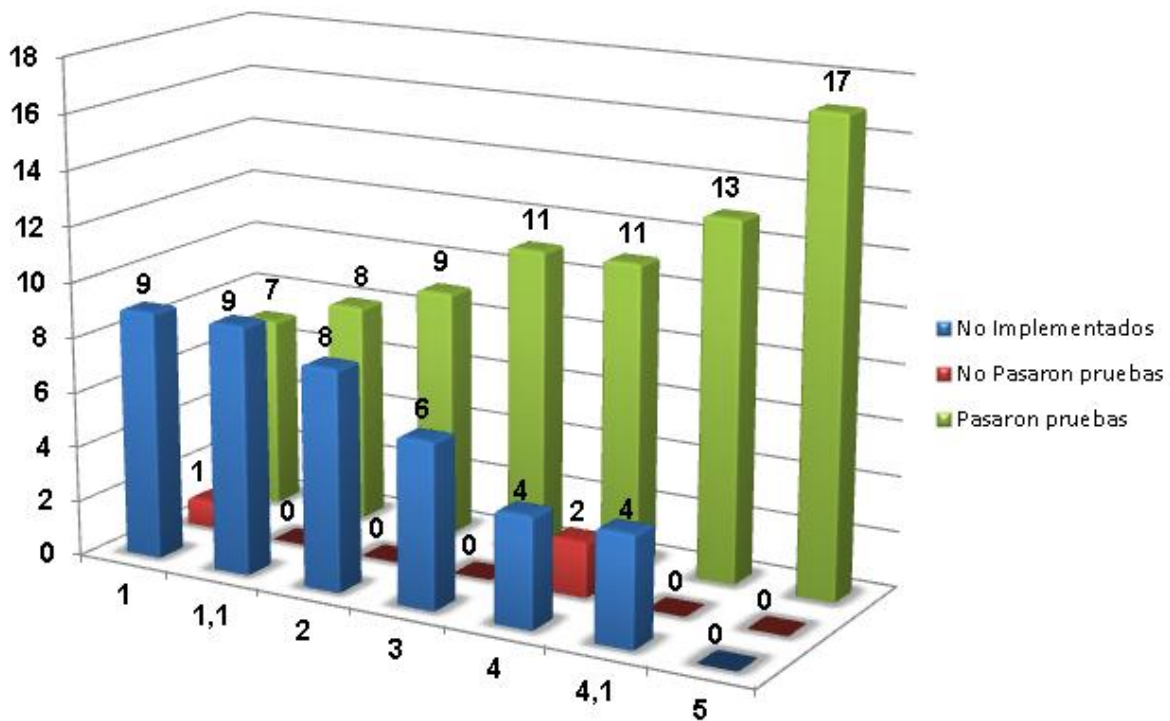


Figura 6. Gráfica de las pruebas realizadas a cada requisito funcional por cada iteración del desarrollo.

Luego de realizar un análisis estadístico de los resultados de las pruebas en cada iteración se obtuvo como resultado que la biblioteca de componentes de interfaz de usuario cumple con todas las funcionalidades y requerimientos del cliente, teniendo resultados correctos y un adecuado rendimiento.

```

[TestMethod()]
[HostType("ASP.NET")]
[AspNetDevelopmentServerHost("D:\\Proyecto\\repo\\SUIN\\SUIN\\DevTarjetaMenor\\source\\Suin", "/")]
[UrlToTest("http://localhost:53040/")]

public void GetPersonByIdentityNumberTest()
{
    SearchService target = new SearchService();
    string NumberIdentity = Convert.ToString(87100915465);
    string expected = "DUEÑAS";
    string actual;
    actual = target.GetPersonByIdentityNumber(NumberIdentity).PFirstSurname;
    Assert.AreEqual(expected, actual);
}

[TestMethod()]
[HostType("ASP.NET")]
[AspNetDevelopmentServerHost("D:\\Proyecto\\repo\\SUIN\\SUIN\\DevTarjetaMenor\\source\\Suin", "/")]
[UrlToTest("http://localhost:53040/")]
public void GetNationalPersonByIdTest()
{
    SearchService target = new SearchService();
    Guid PersonId = new Guid("9910ebc0-97cd-7643-ad8c-0ef3e8c57f0c");
    string expected = "RAFAEL";
    string actual;
    actual = target.GetNationalPersonById(PersonId).PSecondName;
    Assert.AreEqual(expected, actual);
}

```

Test run completed Results: 2/2 passed; Item(s) checked: 0				
	Result	Test Name	Project	Error Message
<input type="checkbox"/>	Passed	GetNationalPersonByIdTest	BirthTest	
<input type="checkbox"/>	Passed	GetPersonByIdentityNumberTest	BirthTest	

Figura 7. Resultado de la prueba unitaria al componente para la búsqueda de personas.

3.7 Conclusiones parciales

Se desarrolló la biblioteca de componentes de interfaz de usuario con los controles que se mencionan a continuación: *LayoutGrid*, *LayoutPanel*, *Accordion*, *TextField*, *AutoComplete*, *ComboBox*, *ListView*,

CheckBox, *RadioButton*, *Button*, *DatePicker*, *Search*, *PhotoCapture*, *FileUpload*, *StripImage*, *Label*, *TabControl*, que es actualmente usada en todos los subsistemas en desarrollo del proyecto.

Se realizaron pruebas unitarias usando el *Microsoft Visual Studio 2010*, así como 3 iteraciones de pruebas de caja negra garantizándose que la biblioteca tiene un nivel adecuado de funcionalidad y usabilidad.

CONCLUSIONES

- ✓ El estudio de la metodología, tecnologías y herramientas utilizadas para el desarrollo del proyecto, así como las relativas al desarrollo de la biblioteca de componentes de interfaz de usuario *UIToolsBox* permitió profundizar en los principales elementos y los aspectos novedosos a tener en cuenta durante el proceso de desarrollo.
- ✓ Con el análisis de soluciones semejantes disponibles se determinó que no existe ninguna que cumpla con los requisitos establecidos para el desarrollo de las interfaces de usuario del proyecto.
- ✓ Se analizaron las necesidades del proyecto para el desarrollo de las interfaces de usuario, a partir de las cuales se determinaron los requisitos a tener en cuenta para la implementación.
- ✓ Se definió una arquitectura basada en servicios que garantiza la flexibilidad de la ejecución de la propuesta de solución, lográndose una independencia entre las capas definidas y extensibilidad necesaria en la propuesta.
- ✓ Se desarrolló la biblioteca de componentes de interfaz de usuario con los controles que se mencionan a continuación: *LayoutGrid, LayoutPanel, Accordion, TextField, AutoComplete, ComboBox, ListView, CheckBox, RadioButton, Button, DatePicker, Search, PhotoCapture, FileUpload, PersonStripe, Label, TabControl*, que es actualmente usada en todos los subsistemas en desarrollo del proyecto.
- ✓ Se realizaron pruebas unitarias usando el Microsoft Visual Studio 2010, así como X iteraciones de pruebas de caja negra garantizándose que la biblioteca tenga un nivel adecuado de funcionalidad y usabilidad.

RECOMENDACIONES

- ✓ Realizar pruebas periódicas a la biblioteca para garantizar la compatibilidad con nuevas versiones de los navegadores web.
- ✓ Realizar revisiones periódicas de las soluciones con el objetivo de determinar elementos comunes que existan y generalizarlos a través de nuevos componentes en la biblioteca de componentes de interfaz de usuario *UIToolsBox*.
- ✓ Continuar perfeccionando el desarrollo de los componentes para mejorar su funcionamiento.
- ✓ Realizar un mayor número de pruebas basadas en ambientes reales para el análisis de la eficiencia, con el objetivo de optimizar las soluciones brindadas.

BIBLIOGRAFÍA

1. Norman, Donald. *The Design of Future Things*. 2007. p. 245.
2. Barcia, Diego. Maestros de la Web. ¿Qué es CSS? [Online] [Cited: Julio 21, 2010.] <http://www.maestrosdelweb.com/editorial/introcscs/>.
3. Todo RIA. [Online] [Cited: Enero 20, 2011.] <http://www.todo-ria.blogspot.com>.
4. Sitio Web Oficial Telerik. [Online] 2011. [Cited: enero 25, 2011.] <http://www.telerik.com>.
5. Sitio Web Oficial ASP.NET Ajax Control Toolkit. [Online] 2009. [Cited: febrero 4, 2011.] <http://www.asp.net/ajax/ajaxcontroltoolkit>.
6. Grau, Xavier Farré. *Marco de integración de la usabilidad en el proceso de desarrollo de software*. Madrid : s.n., 2005, p. 97.
7. World Wide Web Consortium. *World Wide Web Consortium*. [Online] mayo 9, 2010. <http://www.w3.org>.
8. Introducción al DHTML. *DHTML Ya*. [Online] 2007. [Cited: noviembre 10, 2010.] <http://www.dhtml-ya.com.ar>.
9. Desarrollo Tecnologías Web. [Online] 2011. [Cited: diciembre 15, 2010.] <http://sabia.tic.udc.es/gc/Contenidos%20adicionales/trabajos/Interfaces/enlightment/index.html>.
10. Documentación JQuery. *JQuery*. [Online] 2009. [Cited: noviembre 3, 2010.] http://docs.jquery.com/Main_Page.
11. Granville G., Miller. Visual Studio Team System. *Innovation and integration across the entire application development life cycle*. [Online] 2010. [Cited: febrero 25, 2011.] <http://blogs.msdn.com/b/askburton/archive/2004/12/22/330974.aspx>.
12. Digital Persona. *Digital Persona*. [Online] [Cited: marzo 10, 2011.] <http://www.digitalpersona.com>.
13. Carnegie Mellon, University. CMMI. *Software Engineering Institute*. [Online] 2010. [Cited: marzo 10, 2011.] <http://www.sei.cmu.edu/cmmi/index.cfm>.
14. L., Ing. Maria Eugenia Arevalo. Maria Eugenia Arevalo Blog. [Online] enero 7, 2010. <http://arevalomaria.wordpress.com/2010/01/07/ejemplo-de-documento-vision-y-alcance/>.
15. Altova Corporation. Altova. [Online] Altova, 2005-2011. [Cited: noviembre 27, 2010.] http://www.altova.com/agile_umodel.html.

16. Microsoft. MSDN Library . *MSDN Library*. [Online] <http://www.msdn.microsoft.com/en-us..>
17. Microsoft Corporation. MSDN. *MSDN*. [Online] 2011. [Cited: diciembre 10, 2010.] <http://msdn.microsoft.com/es-ES/library/z1zx9t92.aspx>.
18. —. Microsoft Download Center. [Online] 2011. [Cited: diciembre 3, 2010.] <http://www.microsoft.com/downloads/details.aspx?familyid=9CFB2D51-5FF4-4491-B0E5-B386F32C0992&displaylang=es..>
19. —. Soporte Microsoft. *Microsoft*. [Online] 2011. [Cited: noviembre 20, 2010.] <http://support.microsoft.com/kb/154585>.
20. Microsoft Visual Studio. Microsoft Visual Studio. *Microsoft Visual Studio*. [Online] 2011. [Cited: diciembre 10, 2010.] <http://msdn.microsoft.com/es-es/vstudio/ff431702.aspx>.
21. Microsoft Corporation. Microsoft. [Online] 2010. [Cited: diciembre 2, 2010.] <http://www.microsoft.com/visualstudio/en-us/products/2010-editions/team-foundation-server>.
22. IEE. [Online] 1990 . <http://ieeexplore.ieee.org/>.
23. Mozilla developer center. [Online] [Cited: febrero 2, 2010.] <https://developer.mozilla.org/>.
24. Mozilla Addons. *Mozilla Addons*. [Online] marzo 16, 2010. <https://addons.mozilla.org/>.
25. UCI. *Documentos Proyecto Identidad Cuba*. 2009.
26. Ingeniería de software orientada a objetos con Java e Internet. [book auth.] A WEITZENFELD. 2006.
27. Vegas, Sira, Juristo, Natalia and Moreno, Ana M. Técnicas de Evaluación de Software. [Online] http://is.ls.fi.upm.es/docencia/erdsi/Documentacion_Evaluacion_7.pdf.
28. Corporation, Microsoft. *Manual de la metodología CMMI for MSF*. 2009.
29. Proyecto Identidad Cuba. UCI. *Arquitectura de Software*. [Documento] 2010.

ANEXOS

ANEXO 1. CATÁLOGO DE LOS REQUISITOS FUNCIONALES

RF1. Incluir componente para la división virtual de un área en columnas.

1.1 Permitir dividir el área de trabajo en 28 columnas de 25px de ancho por 20px de alto.

RF2. Incluir componente para la agrupación de elementos.

2.1 Definir el título que identifique, en forma de encabezado el área contenida por el componente.

2.2 Permitir especificar si se debe dejar un espacio equivalente a una Línea (20px) entre el borde superior del control y el contenido del mismo.

2.3 Definir el alto del componente.

RF3. Incluir componente acordeón.

3.1 Definir una variable de sesión para el menú que sea seleccionado.

3.2 Definir una variable de sesión que determine la carga de los datos de la vista lógica.

RF4. Incluir componente campo de texto.

4.1 Definir el título que identifique el contenido del componente.

4.2 Permitir definir si está habilitado el componente para ser modificado o no.

4.3 Definir el texto contenido en el componente.

4.4 Define el comportamiento del control con respecto a su contenido puede ser de 3 tipos:

- a) *SingleLine*: el control debe mostrar una línea con el texto contenido.
- b) *MultiLine*: el control debe mostrar varias líneas en correspondencia con el texto contenido.
- c) *Password*: el control debe visualizar el texto contenido mediante el uso de asteriscos.

RF5. Incluir componente para el autocompletamiento de datos.

5.1 Definir la ubicación de los datos a cargar por el control.

5.2 Definir un identificador del elemento seleccionado por el control.

5.3 Definir un listado de elementos que se muestren en el control (opcional).

RF6. Incluir componente ComboBox.

6.1 Definir la ubicación de los datos a cargar por el control.

6.2 Definir el identificador del elemento seleccionado por el control.

6.3 Permitir definir si al seleccionar un elemento en el control, se realiza el eventoPostBack del mismo.

6.4 Permitir definir si es posible modificar el contenido del control.

6.5 Definir un listado de elementos que se muestren en el control (opcional).

RF7. Incluir componente para listar elementos.

7.1 Definir un listado de elementos que se muestran en el control (opcional).

7.2 Definir un valor del elemento seleccionado.

7.3 Permitir añadir código HTML a la funcionalidad del control.

7.4 Permitir predeterminar el alto del componente.

7.5 Permitir definir si es posible usar el filtrado de datos o no.

7.6 Permitir define si es posible seleccionar datos listados o no.

RF8. Incluir componente CheckBox.

8.1 Permitir definir si el elemento fue seleccionado o no.

8.2 Permitir definir el texto que identifica el control.

RF9. Incluir componente RadioButton.

9.1 Permitir definir si el elemento fue seleccionado o no.

9.2 Permitir definir el texto que identifica el control.

RF10. Incluir componente botón.

10.1 Definir el texto que identifica el control.

RF11. Incluir componente para la entrada de fechas.

- 11.1 Definir una restricción del valor de fecha mínima del control.
- 11.2 Definir una restricción del valor de fecha máxima del control.
- 11.3 Definir el tipo de hora según el uso horario del ordenador utilizado.

RF12. Incluir componente para la búsqueda de personas.

- 12.1 Definir la dirección URL del servicio que brinda la fuente de datos para la obtención de información para el control.

RF13. Incluir componente capturador de imágenes.

- 13.1 Definir la captura de imágenes en formato de imagen en base 64 bits.
- 13.2 Definir la captura de imágenes en formato de un arreglo de imágenes.
- 13.3 Definir la captura de imágenes en formato de imagen jpeg/png.

RF14. Incluir componente para la carga de archivos.

- 14.1 Definir el valor de la ubicación del archivo seleccionado.
- 14.2 Definir un identificador para el valor de la ubicación del archivo seleccionado.

RF15. Incluir componente etiqueta.

- 15.1 Definir el texto que muestra el control.

RF16. Incluir componente para mostrar información de una persona.

- 16.1 Definir el nombre de la persona a mostrar.
- 16.2 Definir el número de identificación de la persona a mostrar.
- 16.3 Definir el identificador de la imagen a mostrar de la persona deseada.

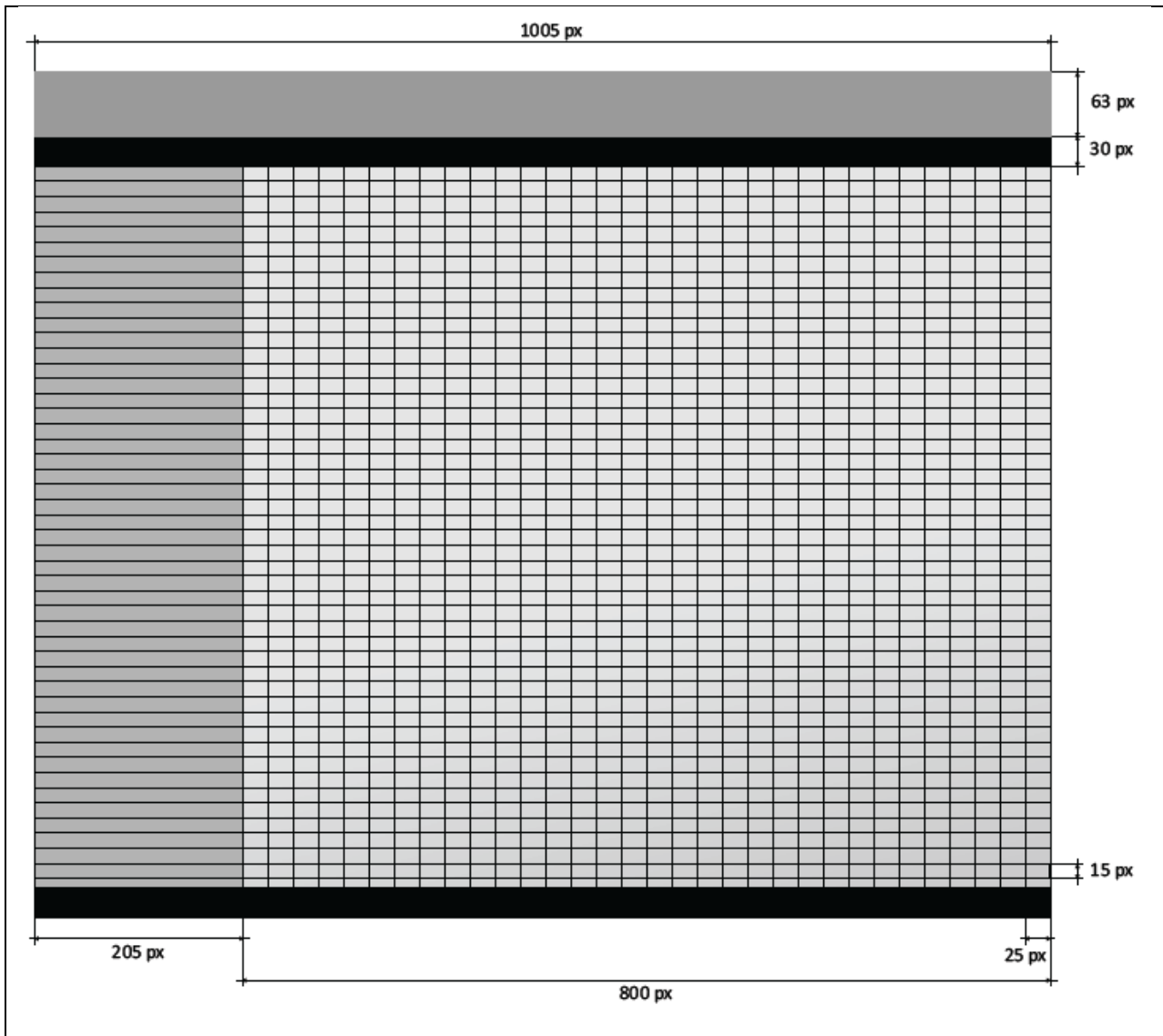
RF16. Incluir componente de pestañas.

- 16.1 Listar los elementos que contienen las pestañas del control.
- 16.2 Definir la pestaña que se encuentra seleccionada.

ANEXO 2. DESCRIPCIÓN DE LOS REQUISITOS FUNCIONALES

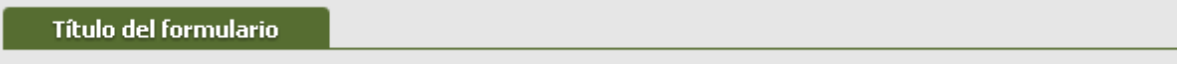
RF1. Incluir componente para la división virtual de un área en columnas.

Nombre	LayoutGrid	Clasificación	Básico	Tipo	Contenedor
Descripción	Componente para la división virtual de un área en columnas. Contiene los estilos necesarios				
Propiedades			Comportamientos		
-			-		
Relación con otros componentes					
System.Web.UI.WebControls.WebControl.Panel					
Dependencias					
UIToolsBox.Common.Behaviors.StyleBehavior					
Requisitos no funcionales					
Divide el área de trabajo en 28 columnas de 25px de ancho por 20px de alto.					
Prototipo					



RF2. Incluir componente para la agrupación de elementos.

Nombre	LayoutPanel	Clasificación	Básico	Tipo	Contenedor
Descripción	Componente para agrupar controles en un área determinada.				
Propiedades				Comportamientos	

Nombre	Descripción	
Title	Define el título que identifica, en forma de encabezado el área contenida por el mismo.	✓ LayoutBehavior
InnerTopSpace	Especifica si se debe dejar un espacio equivalente a una Línea (20px) entre el borde superior del control y el contenido del mismo	
Lines	Predetermina el alto del componente.	
Relación con otros componentes		
System.Web.UI.WebControls.WebControl.Panel		
Dependencias		
UIToolsBox.Common.Behaviors.StyleBehavior		
Requisitos no funcionales		
El título debe mostrarse acorde a las pautas definidas en el documento [página 5].		
Prototipo		
		

RF3. Incluir componente acordeón.

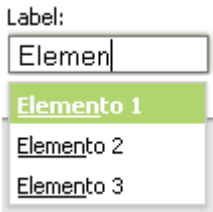
Nombre	Accordion	Clasificación	Básico	Tipo	Contenedor
Descripción	Componente listar un conjunto de elementos agrupados en dos niveles en forma de acordeón.				

Propiedades		Comportamientos
Nombre	Descripción	✓ LayoutBehavior
MenuActiveItem	Define la variable de sesión del menú seleccionado.	
MenuSourceVar	Define la variable de sesión que determina la carga de los datos de la vista lógica.	
Relación con otros componentes		
-		
Dependencias		
<ul style="list-style-type: none"> ✓ UIToolsBox.Common.Behaviors.StyleBehavior. ✓ UIToolsBox.Common.IMainInputControl. 		
Requisitos no funcionales		
El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 10].		
Prototipo		




RF5. Incluir componente para el autocompletamiento de datos.

Nombre	Autocomplete	Clasificación	Avanzado	Tipo	Entrada de Datos
Descripción	Componente para la entrada texto con sugerencias.				
Propiedades				Comportamientos	
Nombre	Descripción				✓ LayoutBehavior
DataSource	Define la ubicación de los datos a cargar por el control.				
Value	Identificador del elemento seleccionado por el control.				
Items	Define un listado de elementos que se muestran en el control (opcional).				
Relación con otros componentes					

UIToolsBox.Controls.TextField (Hereda)
Dependencias
<ul style="list-style-type: none"> ✓ UIToolsBox.Common.Behaviors.StyleBehavior.
Requisitos no funcionales
El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 18]
Prototipo
 <p>Label: Elemen Elemento 1 Elemento 2 Elemento 3</p>

RF6. Incluir componente *ComboBox*.

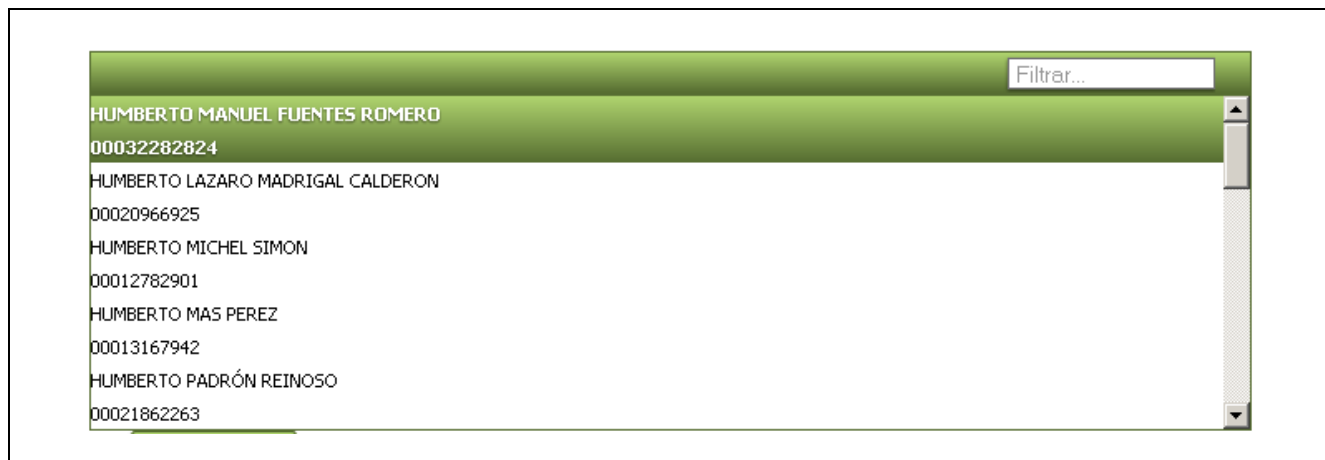
Nombre	ComboBox	Clasificación	Básico	Tipo	Entrada de Datos
Descripción	Componente para la entrada texto a partir de la selección de una opción.				
Propiedades			Comportamientos		
Nombre	Descripción		<ul style="list-style-type: none"> ✓ LayoutBehavior 		
DataSource	Define la ubicación de los datos a cargar por el control.				
Value	Identificador del elemento seleccionado por el control.				

AutoPostBack	Define si al seleccionar un elemento en el control, se realiza el evento <i>PostBack</i> del mismo.	
Write	Define si es posible modificar el contenido del control.	
Items	Define un listado de elementos que se muestran en el control (opcional).	
Relación con otros componentes		
UIToolsBox.Controls.TextField (Hereda)		
Dependencias		
<ul style="list-style-type: none"> ✓ UIToolsBox.Common.Behaviors.StyleBehavior. ✓ System.Web.UI.IPostBackDataHandler. ✓ System.Web.UI.IPostBackEventHandler. 		
Requisitos no funcionales		
El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 18]		
Prototipo		
<p style="text-align: center;">Sexo:</p> 		

RF7. Incluir componente para listar elementos.

Nombre	ListView	Clasificación	Avanzado	Tipo	Contenedor
--------	----------	---------------	----------	------	------------

Descripción	Componente para listar un conjunto de elementos.	
Propiedades		Comportamientos
Nombre	Descripción	✓ LayoutBehavior
Items	Define un listado de elementos que se muestran en el control (opcional).	
SelectedItem	Define el valor del elemento seleccionado.	
TemplateDefinition	Permite añadir código HTML a la funcionalidad del control.	
Lines	Predetermina el alto del componente.	
Filterable	Define si es posible usar el filtrado de datos o no.	
Selectable	Define si es posible seleccionar datos listados o no.	
Relación con otros componentes		
-		
Dependencias		
UIToolsBox.Common.Behaviors.StyleBehavior		
Requisitos no funcionales		
El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 19]		
Prototipo		



RF8. Incluir componente *CheckBox*.

Nombre	CheckBox	Clasificación	Básico	Tipo	Entrada de Datos
Descripción	Componente que permite realizar selecciones múltiples a partir de un conjunto de opciones.				
Propiedades			Comportamientos		
Nombre	Descripción		✓ LayoutBehavior		
Checked	Define si el elemento fue seleccionado o no.				
Text	Define el texto que identifica el control.				
Relación con otros componentes					
System.Web.UI.WebControls.CheckBox (Hereda)					
Dependencias					
UIToolsBox.Common.Behaviors.StyleBehavior.					
Requisitos no funcionales					

El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 18]
Prototipo
<input checked="" type="checkbox"/> Nacional <input type="checkbox"/> Extranjero

RF9. Incluir componente *RadioButton*.

Nombre	RadioButton	Clasificación	Básico	Tipo	Entrada de Datos
Descripción	Componente que permite realizar selecciones únicas a partir de un conjunto de opciones.				
Propiedades				Comportamientos	
Nombre	Descripción				✓ LayoutBehavior
Checked	Define si el elemento fue seleccionado o no.				
Text	Define el texto que identifica el control.				
Relación con otros componentes					
System.Web.UI.WebControls.RadioButton (Hereda)					
Dependencias					
UIToolsBox.Common.Behaviors.StyleBehavior.					
Requisitos no funcionales					

El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 18]

Prototipo

Nacional Extranjero

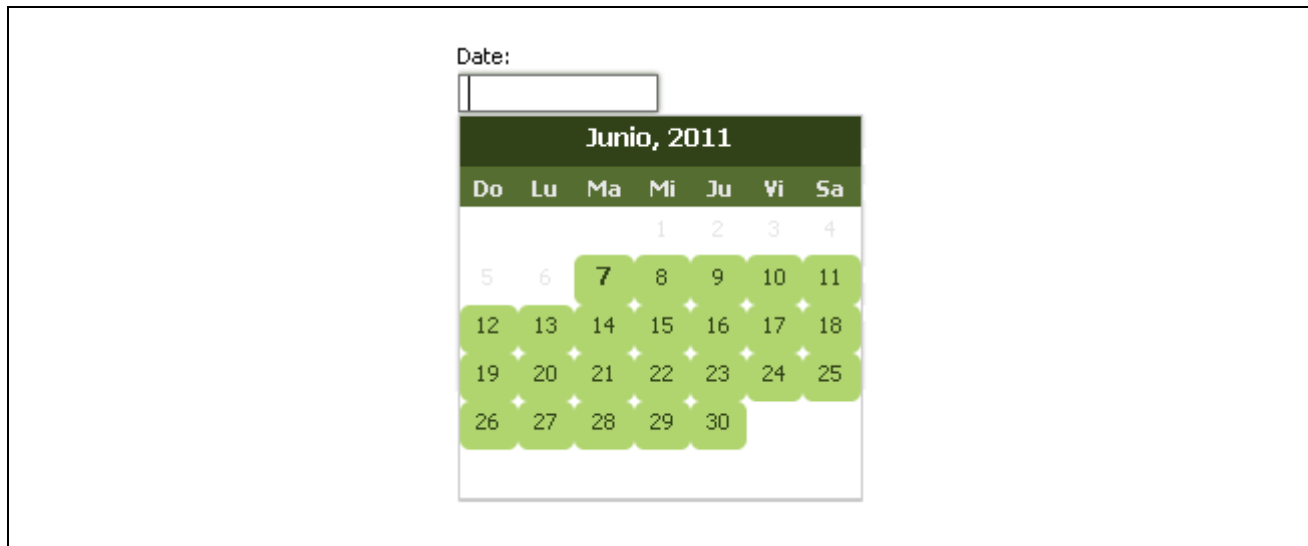
RF10. Incluir componente botón.

Nombre	Button	Clasificación	Básico	Tipo	Entrada de Datos
Descripción	Componente que genera una acción o evento a partir de su pulsación.				
Propiedades					
Comportamientos					
Nombre	Descripción			✓ LayoutBehavior	
Text	Define el texto que identifica el control.				
Relación con otros componentes					
System.Web.UI.WebControls.Button (Hereda)					
Dependencias					
UIToolsBox.Common.Behaviors.StyleBehavior.					
Requisitos no funcionales					
El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 15]					
Prototipo					

Buscar

RF11. Incluir componente para la entrada de fechas.

Nombre	DatePicker	Clasificación	Avanzado	Tipo	Entrada de Datos
Descripción	Componente que permite la entrada de fechas.				
Propiedades			Comportamientos		
Nombre	Descripción		✓ LayoutBehavior		
DataBegin	Define una restricción del valor de fecha mínima del control.				
DataEnd	Define una restricción del valor de fecha máxima del control.				
DateTime	Define el tipo de hora según el uso horario del ordenador utilizado.				
Relación con otros componentes					
UIToolsBox.Controls.TextField (Hereda)					
Dependencias					
UIToolsBox.Common.Behaviors.StyleBehavior.					
Requisitos no funcionales					
El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 18]					
Prototipo					



RF12. Incluir componente para la búsqueda de personas.

Nombre	Search	Clasificación	Avanzado	Tipo	Complejo
Descripción	Componente que permite la búsqueda de una persona a partir del filtrado de datos.				
Propiedades			Comportamientos		
Nombre	Descripción		✓ LayoutBehavior		
SearchServiceUrl	Define la dirección <i>URL</i> del servicio que brinda la fuente de datos para la obtención de información para el control.				
Relación con otros componentes					
UIToolsBox.Controls.TabControl (Hereda)					
Dependencias					
UIToolsBox.Common.Behaviors.StyleBehavior.					
Requisitos no funcionales					

El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 14].

Prototipo

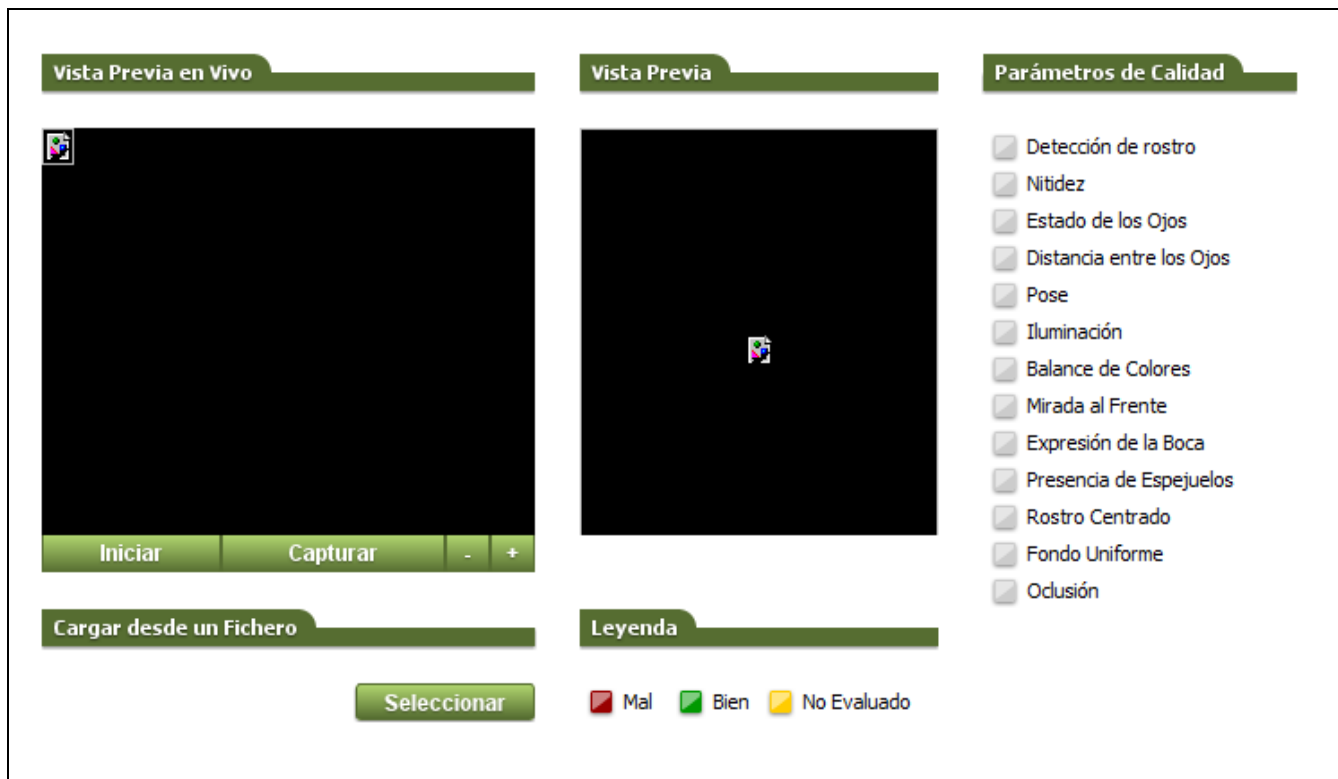
Número CI: Primer Nombre: Segundo Nombre: Primer Apellido: Segundo Apellido: Fonética

Fallecido Residente en el Exterior

RF13. Incluir componente para la capturar de imágenes.

Nombre	PhotoCapture	Clasificación	Avanzado	Tipo	Complejo
Descripción	Componente que permite la captura y procesamiento de imágenes.				

Propiedades		Comportamientos
Nombre	Descripción	✓ LayoutBehavior
ImageBase64	Define la captura de imágenes en formato de imagen en base 64 bits.	
ImageByte	Define la captura de imágenes en formato de un arreglo de imágenes.	
Image	Define la captura de imágenes en formato de imagen jpeg/png.	
Relación con otros componentes		
-		
Dependencias		
<ul style="list-style-type: none"> ✓ UIToolsBox.Common.Behaviors.StyleBehavior. ✓ UIToolsBox.Common.Behaviors.BaseUIToolsBoxControl. 		
Requisitos no funcionales		
El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 14].		
Prototipo		



RF14. Incluir componente para la carga de archivos.

Nombre	FileUpload	Clasificación	Avanzado	Tipo	Entrada de Datos
Descripción	Componente que permite la carga de archivos.				
Propiedades			Comportamientos		
Nombre	Descripción		✓ LayoutBehavior		
Text	Define el valor de la ubicación del archivo seleccionado.				
Value	Identificador del valor de la ubicación del archivo seleccionado.				
Relación con otros componentes					

UIToolsBox.Controls.TextField (Hereda)
Dependencias
UIToolsBox.Common.Behaviors.StyleBehavior.
Requisitos no funcionales
El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 18]
Prototipo
<input type="button" value="Cargar Archivo"/> <input type="button" value="Seleccionar"/>


RF15. Incluir componente etiqueta.

Nombre	Label	Clasificación	Básico	Tipo	Entrada de Datos
Descripción	Componente que permite introducir una cadena de texto para identificar o describir otros componentes.				
Propiedades			Comportamientos		
Nombre	Descripción		✓ LayoutBehavior		
Text	Define el texto que muestra el control.				
Relación con otros componentes					
System.Web.UI.WebControls.Label (Hereda)					
Dependencias					

UIToolsBox.Common.Behaviors.StyleBehavior.
Requisitos no funcionales
El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 7]
Prototipo
Label:

RF16. Incluir componente para mostrar información de una persona.

Nombre	PersonStripe	Clasificación	Avanzado	Tipo	Complejo
Descripción	Componente que permite mostrar información dada una persona.				
Propiedades			Comportamientos		
Nombre	Descripción		✓ LayoutBehavior		
FullName	Define el nombre de la persona a mostrar.				
IdNumber	Define el número de identificación de la persona a mostrar.				
ImageId	Define el identificador de la imagen a mostrar de la persona deseada.				
Relación con otros componentes					
System.Web.UI.WebControls.CompositeControl (Hereda)					
Dependencias					

UIToolsBox.Common.Behaviors.StyleBehavior.
Requisitos no funcionales
El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 13].
Prototipo


RF17. Incluir componente de pestañas.

Nombre	TabControl	Clasificación	Avanzado	Tipo	Contenedor
Descripción	Componente estructurado por varios paneles con datos que están contenidos dentro de una sola ventana principal, usando pestañas para alternar entre ellos.				
Propiedades			Comportamientos		
Nombre	Descripción		✓ LayoutBehavior		
Tabs	Listado que contiene todas las pestañas del control.				
TabActive	Define la pestaña que se encuentra seleccionada.				
Relación con otros componentes					
System.Web.UI.WebControls.Label (Hereda)					
Dependencias					

- ✓ UIToolsBox.Common.Behaviors.StyleBehavior.
- ✓ UIToolsBox.Common.Behaviors.BaseUIToolsBoxControl.

Requisitos no funcionales

El título debe mostrarse acorde a las pautas definidas en el documento (...) [página 11]

Prototipo



ANEXO 3. DESCRIPCIÓN DE LOS CLASES

2.1 Descripción de la clase Button.

Nombre	Button	
Descripción	Es un componente con un funcionamiento muy similar al componente tipo botón que contiene la tecnología ASP.NET. Implementa el comportamiento de las propiedades de la clase interfaz ILayoutBehavior. Corresponde al RF11.	
Propiedad	Tipo	Descripción
PreventPostBack	bool	Define si se ejecutó la acción de <i>PostBack</i> , si fue presionado el botón.

Tabla 12. Descripción de las propiedades de la clase Button del componente Button.

Nombre	Button
Descripción	<p>Es un componente con un funcionamiento muy similar al componente tipo botón que contiene la tecnología ASP.NET.</p> <p>Es un componente de entrada de datos que hereda de la clase BaseUIToolsBoxControl y de la clase del sistema <i>WebControl.Button</i>.</p> <p>Corresponde al RF11.</p>

Tabla 13. Descripción de los métodos de la clase Button del componente Button.**2.2 Descripción de la clase DatePicker.**

Nombre	DatePicker	
Descripción	<p>Es un componente de entrada de datos que permite seleccionar dinámicamente un valor de tipo fecha. Contiene propiedades personalizables para validar y procesar dicho valor seleccionado. Corresponde al RF4.</p>	
Propiedad	Tipo	Descripción
DateBegin	string	Define el valor de la fecha de inicio usando el formato de la fecha común (dd/mm/aa) o con la cadena "hoy" para referirse a la fecha actual.
DateEnd	string	Define el valor de la fecha de inicio usando el formato de la fecha común (dd/mm/aa).

Tabla 14. Descripción de las propiedades de la clase DatePicker del componente DatePicker.

Nombre	DatePicker		
Descripción	Es un componente de entrada de datos que permite seleccionar dinámicamente un valor de tipo fecha. Contiene propiedades personalizables para validar y procesar dicho valor seleccionado. Corresponde al RF4.		
Visibilidad	Método	Argumentos	Descripción.
Protegido	CreateChildControls		Construye la estructura de cada server control, aquí es donde se definen las propiedades de cada uno de ellos.

Tabla 15. Descripción de los métodos de la clase DatePicker del componente DatePicker.

2.3 Descripción de la clase Autocomplete.

Nombre	Autocomplete		
Descripción	Es un componente de entrada de datos que muestra sugerencias a medida que introducimos texto mediante un filtrado de datos permitiendo agilizar el proceso de selección de datos. Implementa el comportamiento de las propiedades de la clase interfaz ILayoutBehavior y hereda de la clase <i>TextField</i> , para la entrada de datos. Corresponde al RF5.		
Propiedad	Tipo	Descripción	
DateSource	string	Define la dirección del servicio web que emplea dicho control para la carga dinámica de datos filtrar por el componente.	

Items	object[]	Define un arreglo de elementos para serán mostrado en el campo de entrada de datos del control.
-------	-----------	---

Tabla 16. Descripción de las propiedades de la clase Autocomplete del componente Autocomplete.

Nombre	Autocomplete		
Descripción	Se definen las propiedades y métodos básicos del componente Autocomplete. Es un componente de entrada de datos que hereda de la clase BaseUIToolsBoxControl. Corresponde al RF5.		
Visibilidad	Método	Argumentos	Descripción.
Protegido	CreateChildControls		Construye la estructura de cada <i>server control</i> , aquí es donde se definen las propiedades de cada uno de ellos.

Tabla 17. Descripción de los métodos de la clase Autocomplete del componente Autocomplete.

Nombre	ResponseObject		
Descripción	Es una clase cuya funcionalidad es la comunicación con el servicio web para el intercambio de datos con el componente de autocompletamiento. Corresponde al RF5.		
Visibilidad	Método	Argumentos	Descripción.
Público	ResponseObject		Constructor de la clase donde se definen los valores iniciales del componente como vacíos.

Tabla 18. Descripción de la clase ResponseObject del componente Autocomplete.

2.4 Descripción de la clase Accordion.

Nombre	Accordion		
Descripción	Un control que despliega elementos de un menú personalizado permitiendo la navegabilidad por el sistema. Posibilita la carga dinámica de los datos a mostrar en los elementos del componente. Corresponde al RF6.		
Propiedad	Tipo	Descripción	
MenuSourceVar	string	Define el nombre que identifica el menú del control.	

Tabla 19. Descripción de las propiedades de la clase Accordion del componente Accordion.

Nombre	Accordion		
Descripción	Un control que despliega elementos de un menú personalizado permitiendo la navegabilidad por el sistema. Hereda de la clase BaseUIToolsBoxControl. Corresponde al RF6.		
Visibilidad	Método	Argumentos	Descripción.

Tabla 20. Descripción de los métodos de la clase Accordion del componente Accordion.

Nombre	Module
---------------	--------

Descripción	Gestiona los elementos del componente Accordion. Corresponde al RF6.		
Visibilidad	Método	Argumentos	Descripción.
Público	Activeltem	string	Verifica y cambia el estado de los elementos del componente Accordion.
Private	HaveUrl	string	Retorna si existen elementos con la dirección url especificada.
Público	Module		Constructor de la clase donde se crea un listado con los elementos del componente Accordion.

Tabla 21. Descripción de la clase Module del componente Accordion.

Nombre	Resource		
Descripción	Contiene los atributos para un elemento del componente Accordion. Corresponde al RF6.		
Visibilidad	Método	Argumentos	Descripción.
Público	Resource		Constructor de la clase donde se asigna por defecto un valor false a la propiedad Active que define el estado para un elemento del componente Accordion.

Tabla 22. Descripción de la clase Resource del componente Accordion.

2.5 Descripción de la clase ListView.

Nombre	ListView		
Descripción	<p>Un control que permite mostrar elementos (imágenes y textos) en forma de listado. Contiene propiedades para definir una adecuada personalización.</p> <p>Implementa el comportamiento de las propiedades de la clase interfaz ILayoutBehavior. Corresponde al RF7.</p>		
Propiedad	Tipo	Descripción	
DateSource	string	Define la dirección del servicio web que emplea dicho control para la carga dinámica de datos por el componente.	
DataFields	ArrayList	Define un arreglo de elementos para serán mostrados en el componente.	
Filtrable	bool	Define si los datos del control son filtrables o no.	

Tabla 23. Descripción de las propiedades de la clase ListView del componente ListView.

Nombre	ListView		
Descripción	<p>Un control que permite mostrar elementos (imágenes y textos) en forma de listado. Contiene propiedades para definir una adecuada personalización. Hereda de la clase BaseUIToolsBoxControl. Corresponde al RF7.</p>		
Visibilidad	Método	Argumentos	Descripción.
Protegido	CreateChildControls		Construye la estructura de cada <i>server control</i> , aquí es donde se

			definen las propiedades de cada uno de ellos.
--	--	--	---

Tabla 24. Descripción de los métodos de la clase ListView del componente ListView.

2.6 Descripción de la clase TabControl.

Nombre	TabControl		
Descripción	<p>Un control que permite ubicar mediante pestañas otros controles posibilitando agrupar mayor cantidad de elementos en una página del sistema.</p> <p>Implementa el comportamiento de las propiedades de la clase interfaz ILayoutBehavior. Corresponde al RF8.</p>		
Propiedad	Tipo	Descripción	
tabBody	Panel	Define un panel para cada pestaña que contendrá el control.	
tabName	string	Define el nombre que tendrá la pestaña.	

Tabla 25. Descripción de las propiedades de la clase TabControl del componente TabControl.

Nombre	TabControl		
Descripción	<p>Un control que permite ubicar mediante pestañas otros controles posibilitando agrupar mayor cantidad de elementos en una página del sistema. Hereda de la clase BaseUIToolsBoxControl. Corresponde al RF8.</p>		
Visibilidad	Método	Argumentos	Descripción.
Protegido	CreateChildControls		Construye la estructura de cada server

			control, aquí es donde se definen las propiedades de cada uno de ellos.
Público	TabControl		Constructor de la clase donde se crea un objeto de tipo Panel y se declara una variable con un identificador para el componente.

Tabla 26. Descripción de los métodos de la clase TabControl del componente TabControl.

2.7 Descripción de la clase NumericField.

Nombre	NumericField		
Descripción	<p>Un control de entrada de datos numéricos que permite definir propiedades de validación para restringir el valor máximo y mínimo del mismo, definir valores de incremento, así como otras propiedades comunes para un componente de entrada datos de tipo texto.</p> <p>Implementa el comportamiento de las propiedades de la clase interfaz ILayoutBehavior. Corresponde al RF9.</p>		
Propiedad	Tipo	Descripción	
incremento	int	Define el valor de incremento para el control.	
Max	int	Define el valor máximo que puede tomar el control.	
Min	int	Define el valor mínimo que puede tomar el control.	
ValidationTypes	enum	Define tres comportamientos para el campo de entrada de datos (NoValidation, Min, Max, Both), descritos a	

		<p>continuación:</p> <ul style="list-style-type: none"> ✓ NoValidation: el valor del control puede tomar cualquier valor. ✓ Min: el valor del campo estará restringido por el valor definido en la propiedad <i>Min</i>. ✓ Max: el valor del campo estará restringido por el valor definido en la propiedad <i>Max</i>. ✓ Both: el valor del campo estará restringido por el valor mínimo definido en la propiedad <i>Min</i> y por el valor máximo definido en la propiedad <i>Max</i>.
--	--	--

Tabla 27. Descripción de las propiedades de la clase *NumericField* del componente *NumericField*.

Nombre	NumericField		
Descripción	Un control de entrada de datos numéricos que permite definir propiedades de validación para restringir el valor máximo y mínimo del mismo, definir valores de incremento, así como otras propiedades comunes para un componente de entrada datos de tipo texto. Hereda de la clase <i>BaseUIToolsBoxControl</i> . Corresponde al RF9.		
Visibilidad	Método	Argumentos	Descripción.
Protegido	CreateChildControls		Construye la estructura de cada server control, aquí es donde se definen las propiedades de cada uno de ellos.

Tabla 28. Descripción de los métodos de la clase *NumericField* del componente *NumericField*.

2.8 Descripción de la clase ComboBox.

Nombre	ComboBox	
Descripción	<p>Un control que permite definir la fuente de datos para los elementos que cargará de forma automática y serán mostrados en forma de lista, permitiendo la selección de los mismos.</p> <p>Implementa el comportamiento de las propiedades de la clase interfaz ILayoutBehavior. Corresponde al RF10.</p>	
Propiedad	Tipo	Descripción
Write	bool	Define si aparece el valor del elemento seleccionado en el campo de entrada de texto del control.
Items	object[]	Define una lista de objetos con los elementos que cargará y mostrará el control.
DataSource	string	Define una cadena donde se especifica el servicio web que empleará el control para la carga dinámica de los datos.
AutoPostBack	bool	Define automáticamente si fue seleccionado algún elemento del control por el usuario.

Tabla 29. Descripción de las propiedades de la clase ComboBox del componente ComboBox.

2.9 Descripción de la clase Search.

Nombre	Search
Descripción	Un control que permite buscar y mostrar información haciendo uso del filtrado de datos. Corresponde al RF1.

Propiedad	Tipo	Descripción
SearchServiceUrl	string	Define la dirección <i>url</i> del servicio web que debe consumir el componente para acceder a la información que se desea buscar.

Tabla 30. Descripción de las propiedades de la clase Search del componente Search.

2.10 Descripción de la clase PhotoCapture.

Nombre	PhotoCapture	
Descripción	<p>Un control que permite la captura de imágenes haciendo uso de un dispositivo de captura de vídeo (<i>webcam</i>) y procesar para la gestión de la misma.</p> <p>Implementa el comportamiento de las propiedades de la clase interfaz <i>ILayoutBehavior</i>. Corresponde al RF9.</p>	
Propiedad	Tipo	Descripción
Image	Image	Define un objeto de tipo Image de ASP.NET para la manipulación de imágenes.
ImageByte	byte[]	Define un arreglo de las imágenes que debe capturar el componente.
DataSource	string	Define la ubicación de dónde cargará información del control.
ImageBase64	string	Define la propiedad para el procesamiento de imágenes de tipo base 64 dígitos.

Tabla 31. Descripción de las propiedades de la clase PhotoCapture del componente PhotoCapture.

Nombre	PhotoCapture		
Descripción	Un control de captura de imágenes que permite definir propiedades para procesar imágenes. Hereda de la clase BaseUIToolsBoxControl. Corresponde al RF2.		
Visibilidad	Método	Argumentos	Descripción.
Protegido	CreateChildControls		Construye la estructura de cada server control, aquí es donde se definen las propiedades de cada uno de ellos.

Tabla 32. Descripción de los métodos de la clase PhotoCapture del componente PhotoCapture.

2.11 Descripción de la clase GridView.

Nombre	GridView	
Descripción	Un control que definir las propiedades de posicionamiento de los controles en el modo de diseño. Implementa el comportamiento de las propiedades de la clase interfaz ILayoutBehavior. Corresponde al RF13.	
Propiedad	Tipo	Descripción
DataColumns	ArrayList	Define una lista de elementos que contendrán la información referente a la correspondencia entre columna y píxeles para el posicionamiento de elementos en la pantalla.

Tabla 33. Descripción de las propiedades de la clase GridView del componente GridView.

Nombre	GridView		
Descripción	Un control que definir las propiedades de posicionamiento de los controles en el modo de diseño. Hereda de la clase BaseUIToolsBoxControl. Corresponde al RF13.		
Visibilidad	Método	Argumentos	Descripción.
Protegido	CreateChildControls		Construye la estructura de cada server control, aquí es donde se definen las propiedades de cada uno de ellos.

Tabla 34. Descripción de los métodos de la clase GridView del componente GridView.**ANEXO 4. ITERACIONES DE DESARROLLO**

Iteración	Componentes	Descripciones, correcciones y nuevas funcionalidades
Primera iteración UIToolsBox.dll v0.5.4123.19385	<ul style="list-style-type: none"> ✓ Accordion ✓ AutoComplete ✓ Button ✓ TabControl ✓ DatePicker ✓ ListView ✓ TextField ✓ LayoutGrid 	Componente Autocomplete <ul style="list-style-type: none"> ✓ Mejorado el embebido de los estilos básicos. ✓ Permite cargar datos desde una fuente configurable. ✓ Permite filtrar los resultados. ✓ Permite navegar por los resultados usando el teclado. ✓ Resaltar el elemento activo.

		<p>Componente DatePicker</p> <ul style="list-style-type: none">✓ Permite restringir rango de fechas.✓ Permite seleccionar valor de fecha. <p>Componente GridView</p> <ul style="list-style-type: none">✓ Establece ubicar y organizar los componentes en el sistema. <p>Componente Search</p> <ul style="list-style-type: none">✓ Permitir la configuración de la ruta al servicio de búsqueda.✓ Permitir configurar tipos de búsqueda. <p>Componente Button</p> <ul style="list-style-type: none">✓ Permite definir propiedades comunes a la clase <i>Button</i> de <i>ASP.NET</i>. <p>Componente TabControl</p> <ul style="list-style-type: none">✓ Permite establecer cantidad de pestañas a mostrar.✓ Permite añadir elementos a cada pestaña. <p>Componente ListView</p> <ul style="list-style-type: none">✓ Muestra listado de elementos. <p>Componente TextField</p> <ul style="list-style-type: none">✓ Contiene las propiedades comunes de la clase <i>TextField</i> de <i>ASP.NET</i>.✓ Permite establecer validaciones para el tipo de
--	--	--

		dato a introducir.
Segunda iteración UIToolsBox.dll v0.6.3952.26297	<ul style="list-style-type: none"> ✓ Accordion ✓ AutoComplete ✓ Button ✓ LayoutGrid ✓ TabControl ✓ DatePicker ✓ ListView ✓ TextField ✓ Search 	<p>Generales</p> <ul style="list-style-type: none"> ✓ Añadido nuevo componente: <i>Search</i>. ✓ Correcciones de Estilos para el navegador <i>Internet Explorer</i>. <p>Componente Autocomplete</p> <ul style="list-style-type: none"> ✓ Mejorado el embebido de los estilos básicos. <p>Componente DatePicker</p> <ul style="list-style-type: none"> ✓ Nueva propiedad que permite seleccionar rango válido de fechas. ✓ Mejorado el embebido de los estilos básicos. <p>Componente Accordion</p> <ul style="list-style-type: none"> ✓ Nueva propiedad que permite cargar dinámica los datos.
Tercera iteración UIToolsBox.dll v0.7.4070.21930	<ul style="list-style-type: none"> ✓ Accordion ✓ AutoComplete ✓ Button ✓ TabControl ✓ DatePicker ✓ LayoutGrid ✓ ListView ✓ TextField ✓ Search ✓ RadioButton 	<p>Generales</p> <ul style="list-style-type: none"> ✓ Añadidos nuevo componente: <i>RadioButton</i> y <i>CheckBox</i>. ✓ Correcciones de Estilos para el navegador <i>Internet Explorer</i>. ✓ Actualización de la versión del framework JQuery 1.4 a JQuery 1.5. <p>Componente Layout Behavior</p> <ul style="list-style-type: none"> ✓ Posibilidad de dejar un espacio sobre el control.

	<ul style="list-style-type: none"> ✓ CheckBox 	<ul style="list-style-type: none"> ✓ Posibilidad de dejar un espacio debajo del control.
<p>Cuarta iteración</p> <p>UIToolsBox.dll v0.7.4072.19704</p>	<ul style="list-style-type: none"> ✓ Accordion ✓ AutoComplete ✓ Button ✓ ComboBox ✓ DatePicker ✓ LayoutGrid ✓ ListView ✓ LayoutPanel ✓ TextField ✓ Search ✓ TabControl ✓ RadioButton ✓ CheckBox 	<p>Generales</p> <ul style="list-style-type: none"> ✓ Añadidos nuevos componentes: <i>ComboBox</i> y <i>LayoutPanel</i>. ✓ Correcciones de estilos generales. <p>Componente LayoutPanel</p> <ul style="list-style-type: none"> ✓ Nueva propiedad opcional <i>Title</i>. ✓ Corregido espacio fijo al final del elemento. <p>Componente ListView</p> <ul style="list-style-type: none"> ✓ Posibilidad de agregarle elementos manualmente en el momento de generarlo. Propiedad <i>Items</i>. ✓ Posibilidad de especificar el alto del elemento. Propiedad <i>Lines</i> de tipo entero (cada línea equivale a 20 píxeles). ✓ Corregido bug, de selección de elemento cuando convivían varios <i>ListView</i> en la misma página.
<p>Quinta iteración</p> <p>UIToolsBox.dll v0.9.4157.22101</p>	<ul style="list-style-type: none"> ✓ Accordion ✓ AutoComplete ✓ Button ✓ ComboBox ✓ DatePicker ✓ GridView ✓ ListView ✓ LayoutPanel ✓ PersonStripe ✓ PhotoCapture 	<p>Generales</p> <ul style="list-style-type: none"> ✓ Añadidos nuevos componentes: <i>PersonStripe</i> y <i>PhotoCapture</i>, <i>Label</i>, <i>FileUpload</i> . ✓ Aplicado estilos CSS para resaltar <i>TextField</i>, <i>Autocomplete</i>, <i>ComboBox</i>, <i>DatePicker</i>, y demás componentes, en el estado <i>ReadOnly</i>. <p>Componente Search</p> <ul style="list-style-type: none"> ✓ [Estructura Modificada] Botones <i>Buscar</i> y <i>Limpiar</i>

	<ul style="list-style-type: none"> ✓ Search ✓ TabControl ✓ TextField ✓ RadioButton ✓ CheckBox ✓ FileUpload ✓ Label 	<p><i>Intercambiados.</i></p> <ul style="list-style-type: none"> ✓ [Propiedad Modificada] Posibilidad de especificar la <i>url</i> relativa en el <i>SearchServiceUrl</i> (~/<i>Search.asmx</i>) <p>Componente <i>Autocomplete</i></p> <ul style="list-style-type: none"> ✓ [Propiedad Modificada] Posibilidad de especificar la <i>url</i> relativa en el <i>DataSource</i> (~/<i>Servicio.asmx/Funcion</i>). ✓ [Propiedad Agregada] Posibilidad de especificar dependencia de otro control de tipo (<i>Autocomplete</i>). ✓ [Bug Corregido] Luego de hacer una selección, si se modifica el texto por algo que no coincida con uno de los posibles resultados no se limpia el <i>Value</i>. ✓ [Comportamiento Agregado] Pasa a modo <i>ReadOnly</i> automáticamente si se le especificó un padre y este no contiene datos. ✓ [Comportamiento Agregado] Se actualiza automáticamente cuando se producen cambios en el padre. ✓ [Comportamiento Corregido] Al estar en modo <i>ReadOnly</i> no responde al evento <i>Click</i>; <p>Componente <i>ComboBox</i></p> <ul style="list-style-type: none"> ✓ [Propiedad Modificada] Posibilidad de especificar la <i>url</i> relativa en el <i>DataSource</i>
--	---	---

		<p>(~/Servicio.asmx/Funcion).</p> <ul style="list-style-type: none">✓ [Propiedad Agregada] Posibilidad de especificar dependencia de otro control de tipo (<i>Autocomplete</i>).✓ [Bug Corregido] Luego de hacer una selección, si se modifica el texto por algo que no coincida con uno de los posibles resultados no se limpia el <i>Value</i>.✓ [Comportamiento Agregado] Pasa a modo <i>ReadOnly</i> automáticamente si se le especificó un padre y este no contiene datos.✓ [Comportamiento Agregado] Se actualiza automáticamente cuando se producen cambios en el padre.✓ [Comportamiento Corregido] Al estar en modo <i>ReadOnly</i> no responde al evento <i>Click</i>. <p>Componente ListView</p> <ul style="list-style-type: none">✓ [Propiedad Modificada] Posibilidad de especificar la <i>url</i> relativa en el <i>DataSource</i> (~/Servicio.asmx/Funcion).✓ Agregada plantilla general para listar personas (<i>LVPersonTemplate</i>).
--	--	---

ANEXO 5. DISEÑO DE CASOS DE PRUEBAS

CPR2. Incluir componente para el autocompletamiento de datos.

Condiciones de ejecución

- Se debe establecer el valor de la propiedad **ValidationTypes** por el correspondiente a cada escenario (**NoValidation, Text, Number, Email, IntNumber, Date**).

- Se debe establecer el valor de la propiedad **DataSource** con la ubicación de la base de conocimientos que contiene los elementos.

Escenario	Descripción	Text	Respuesta del sistema	Flujo central
EC 1.1 Introducir cadena de texto del elemento que se desea buscar.	El usuario introduce una cadena de texto con el valor del elemento que desea buscar entre datos existentes.	V/ ofi	Se muestra una lista desplegable filtrada por los caracteres de la cadena introducida por el usuario.	
		V/ Cub		
EC 1.2 Seleccionar elemento.	El usuario selecciona un elemento de la lista de elementos.	NA	Se muestra el valor del elemento seleccionado.	

Descripción de las variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
----	-----------------	---------------	------------	-------------

1	Text	Campo de texto	Sí	<p>Las reglas dependen de la propiedad ValidationTypes que se describen a continuación:</p> <p>-NoValidation: permite cualquier valor.</p> <p>-Text: sólo permite entrada de texto, no incluye cadenas de texto con valores de tipo numérico.</p> <p>-Number: sólo permite valores de entrada de tipo numérico. Estos números deben ser enteros, no permite números decimales.</p> <p>-IntNumber: sólo permite valores de entrada de tipo numérico. Acepta números enteros y decimales. Permite para los valores decimales el uso de la coma o punto.</p> <p>-Date: sólo permite valores de entrada de tipo fecha con el formato DD/MM/AAAA.</p> <p>-Email: sólo permite valores de entrada con el formato estándar de tipo email (correo).</p>
---	------	----------------	----	---

CPR3. Incluir componente ComboBox.

Condiciones de ejecución

- Se debe establecer el valor de la propiedad **ValidationTypes** por el correspondiente a cada escenario (**NoValidation, Text, Number, Email, IntNumber, Date**).

Escenario	Descripción	Text	Respuesta del sistema	Flujo central
EC 1.1 Cargar datos en el componente al seleccionarlo desde una fuente de datos externa.	El usuario selecciona el componente para visualizar los datos que contiene el mismo.	V/ Nacional Extranjero	Se muestra una lista desplegable con los elementos que obtuvo el componente de la fuente de datos externa definida en la propiedad DataSource .	-Establecer propiedad DataSource con la ubicación de la fuente de datos externa.

<i>EC 2.1 Cargar datos en el componente a través de la propiedad onselectchange.</i>	<i>El usuario selecciona el componente para visualizar los datos que contiene el mismo.</i>	<i>V/ Oficial interno. Oficial externo. Otro funcionario.</i>	<i>Se muestra una lista desplegable con los elementos que obtuvo el componente de la propiedad onselectchange.</i>	<i>-Implementar funcionalidad para cargar datos usando la propiedad onselectchange.</i>
---	---	---	---	--

Descripción de las variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Text	Campo de texto	Sí	<p>Las reglas dependen de la propiedad ValidationTypes que se describen a continuación:</p> <p>-NoValidation: permite cualquier valor.</p> <p>-Text: sólo permite entrada de texto, no incluye cadenas de texto con valores de tipo numérico.</p> <p>-Number: sólo permite valores de entrada de tipo numérico. Estos números deben ser enteros, no permite números decimales.</p> <p>-IntNumber: sólo permite valores de entrada de tipo numérico. Acepta números enteros y decimales. Permite para los valores decimales el uso de la coma o punto.</p> <p>-Date: sólo permite valores de entrada de tipo fecha con el formato DD/MM/AAAA.</p> <p>-Email: sólo permite valores de entrada con el formato estándar de tipo email (correo).</p>

CPR4. Incluir componente para la entrada de fechas.

Condiciones de ejecución

- Establecer propiedad **ReadOnly** con el valor **True**.
- Establecer propiedad **ValidationTypes** con el valor **Date**.

Escenario	Descripción	Fecha	Respuesta del sistema	Flujo central
EC 1.1 Seleccionar fecha correctamente.	El usuario selecciona de una ventana desplegable la fecha deseada.	V/ 12/5/2011	Se muestra el valor de la fecha seleccionada en el campo de entrada de datos.	

Descripción de las variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Fecha	campo de texto	No	Sólo debe permitir formato de fecha: DD/MM/AAAA

CPR5. Incluir componente para mostrar información de una persona.

Condiciones de ejecución

- Se debe establecer la propiedad **ValidationTypes** con el valor **Number**.

Escenario	Descripción	Text	Respuesta del sistema	Flujo central
EC 1.1 Introducir valor en el campo de texto del componente correctamente.	El usuario introduce un valor numérico entero en el campo de texto.	V/ 80	Se muestra el valor introducido por el usuario en el campo de texto.	
EC 1.2 Introducir valor en el campo de texto del componente incorrectamente.	El usuario introduce datos incorrectos.	V/ número	Se muestra un mensaje de error: Campo no válido .	

<p>EC 1.3 Introducir valor en el campo de texto del componente correctamente que supere el valor de la propiedad Max.</p>	<p>El usuario introduce un valor numérico entero en el campo de texto que supere el valor de la propiedad Max.</p>	<p>V/ 45</p>	<p>Se muestra el valor establecido en la propiedad Max en el campo de texto.</p>	<p>Establecer para la propiedad ValidationTypes con el valor Max. Establecer para la propiedad Max el valor 40.</p>
<p>EC 1.4 Introducir valor en el campo de texto del componente correctamente que sea inferior al valor de la propiedad Min.</p>	<p>El usuario introduce un valor numérico entero en el campo de texto que es inferior al valor de la propiedad Min.</p>	<p>V/ 4</p>	<p>Se muestra el valor establecido en la propiedad Min en el campo de texto.</p>	<p>Establecer para la propiedad ValidationTypes con el valor Min. Establecer para la propiedad Min el valor 5.</p>
<p>EC 2.1 Realizar acción sobre botón de incremento sin restricción de incremento.</p>	<p>El usuario presiona el botón de incremento.</p>	<p>V/ 2</p>	<p>Se muestra el valor numérico en el campo de texto, incrementado según el valor establecido en la propiedad Incremento.</p>	<p>Establecer para la propiedad Incremento el valor 2. Establecer para la propiedad ValidationTypes con el valor NoValidation.</p>
<p>EC 2.2 Realizar acción sobre botón de incremento con restricción de incremento máximo.</p>	<p>El usuario presiona el botón de incremento.</p>	<p>V/ 40</p>	<p>Se muestra el valor numérico en el campo de texto, incrementado según el valor establecido en la propiedad Incremento y</p>	<p>Establecer para la propiedad Incremento el valor 5. Establecer para la propiedad</p>

<p>EC 2.3 Realizar acción sobre botón de incremento con restricción de incremento máximo y mínimo.</p>	<p>El usuario presiona el botón de incremento.</p>	<p>V/ 45</p>	<p>cuando el próximo incremento supera el valor establecido por la propiedad Max no permite modificar dicho valor.</p> <p>Se muestra el valor numérico en el campo de texto, incrementado según el valor establecido en la propiedad Incremento y cuando el próximo incremento supera el valor establecido por la propiedad Max no permite modificar dicho valor.</p>	<p>ValidationTypes con el valor Max.</p> <p>Establecer para la propiedad Max el valor 40.</p> <p>Establecer para la propiedad Incremento el valor 4.</p> <p>Establecer para la propiedad ValidationTypes con el valor Both.</p> <p>Establecer para la propiedad Max el valor 55.</p>
<p>EC 3.1 Realizar acción sobre botón de decremento sin restricción de decremento.</p>	<p>El usuario presiona el botón de decremento.</p>	<p>V/ -10</p>	<p>Se muestra el valor numérico en el campo de texto, disminuyendo según el valor establecido en la propiedad Incremento.</p>	<p>Establecer para la propiedad Incremento el valor 5.</p> <p>Establecer para la propiedad ValidationTypes con el valor NoValidation.</p>

<p>EC 3.2 Realizar acción sobre botón de decremento con restricción de decremento mínimo.</p>	<p>El usuario presiona el botón de decremento.</p>	<p>V/ -25</p>	<p>Se muestra el valor numérico en el campo de texto, disminuyendo según el valor establecido en la propiedad Incremento y cuando el próximo decremento es inferior al valor establecido por la propiedad Min no permite modificar dicho valor.</p>	<p>Establecer para la propiedad Incremento el valor 10. Establecer para la propiedad ValidationTypes con el valor Min. Establecer para la propiedad Min el valor -25.</p>
<p>EC 3.3 Realizar acción sobre botón de decremento con restricción de decremento máximo y mínimo.</p>	<p>El usuario presiona el botón de decremento.</p>	<p>V/ -30</p>	<p>Se muestra el valor numérico en el campo de texto, disminuyendo según el valor establecido en la propiedad Incremento y cuando el próximo decremento es inferior al valor establecido por la propiedad Min no permite modificar dicho valor.</p>	<p>Establecer para la propiedad Incremento el valor 5. Establecer para la propiedad ValidationTypes con el valor Both. Establecer para la propiedad Min el valor 30.</p>

Descripción de las variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
----	-----------------	---------------	------------	-------------

1	Text	Campo de texto	Sí	Como la propiedad establecida ValidationTypes fue Number , sólo permite valores de entrada de tipo numérico. Estos números deben ser enteros, no permite números decimales.
---	------	----------------	----	---

CPR6. Incluir componente etiqueta.

Condiciones de ejecución

- Se le debe agregar a su estructura el componente **etiqueta**.

Escenario	Descripción	TabBody	Respuesta del sistema	Flujo central
EC 1.1 Navegar entre varias pestañas.	El usuario selecciona diferentes pestañas para acceder al contenido de cada una de ellas.		Se muestra el contenido de cada pestaña cuando cada una de ellas es seleccionada.	