PFC .NET – Ingeniería Informática – UOC

Proyecto: Flowerpot

Documento: Memoria

Autor: David García Casado (dgarciacas@uoc.edu)

Tutor: Juan Carlos González Martín

Fecha: 13/06/2011

Índice

Índice de figuras

1.Introducción

El presente documento corresponde a la memoria final del PFC del área .NET desarrollado conforme a la propuesta "Iluminando el escritorio, la web y la nube con Microsoft Silverlight". El nombre de la aplicación es "Flowerpot".

Junto con la presentación en vídeo, conforman la última entrega del PFC.

2.Justificación y objetivos del proyecto

La aplicación consiste en una gestión de plantas hogareñas. El sistema permitirá la gestión de usuarios, especies y ejemplares.

En una primera aproximación a la plataforma Flowerpot, un usuario encontrará una herramienta que le permitirá realizar un inventario de especímenes personal, para después abrirse a la red y posibilitar el intercambio de experiencias y consejos entre usuarios con inquietudes comunes.

El principal objetivo del proyecto es desarrollar la aplicación de gestión de plantas ornamentales Flowerpot. Para ello se necesita:

- Un sistema que permita controlar la evolución de los ejemplares de un usuario.
- Un marco donde exponer esta información en comunidad, de modo que posibilite la compartición de experiencias relacionadas con el cuidado de las mismas.

Esta aplicación pretende ir un paso más allá de los foros y blogs acerca de plantas decorativas (p.ej: plantas.facilisimo.com) ya que proporciona al usuario la utilidad de catalogar sus ejemplares a modo de base de datos personal, incluyendo un histórico con fotos acerca de la evolución de cada una y comentarios acerca de los cuidados dispensados en cada momento.

El sistema dará la posibilidad al usuario de compartir esta información con la comunidad, y una vez dentro, podrá recibir los consejos de otros miembros de Flowerpot.

La catalogación llevada a cabo por cada uno de los usuarios permitirá disponer de una base de datos de especies florales ornamentales vía web a la que el sistema accederá con búsquedas e informes.

3. Planificación

3.1. Planificación inicial

La planificación inicial del proyecto vino definida en su mayor parte por el calendario de la asignatura. Las entregas descritas en el plan docente marcaban los hitos del calendario.

A continuación se muestra la temporalización y el diagrama de Gantt planificado a comienzo del semestre:

	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	PFC .NET	103 días	jue 03/03/11	lun 13/06/11	
2	**Plan trabajo**	**14 días**	**jue 03/03/11**	**mié 16/03/11**	
3	Elección de propuesta	2 días	jue 03/03/11	vie 04/03/11	
4	Introducción a Silverlight	9 días	sáb 05/03/11	dom 13/03/11	3
5	Plan de trabajo	3 días	lun 14/03/11	mié 16/03/11	4
6	**Diseño funcional y técnico**	**26 días**	**jue 17/03/11**	**lun 11/04/11**	
7	Diseño funcional	15 días	jue 17/03/11	jue 31/03/11	5
8	Diseño técnico	11 días	vie 01/04/11	lun 11/04/11	7
9	**Implementación**	**42 días**	**mar 12/04/11**	**lun 23/05/11**	
10	Implementación	35 días	mar 12/04/11	lun 16/05/11	8
11	Pruebas	5 días	mar 17/05/11	sáb 21/05/11	10
12	Manual	2 días	dom 22/05/11	lun 23/05/11	11
13	**Memoria y presentación**	**21 días**	**mar 24/05/11**	**lun 13/06/11**	
14	Memoria	18 días	mar 24/05/11	vie 10/06/11	12
15	Presentación virtual	3 días	sáb 11/06/11	lun 13/06/11	14

Figura 1 – Temporalización inicial

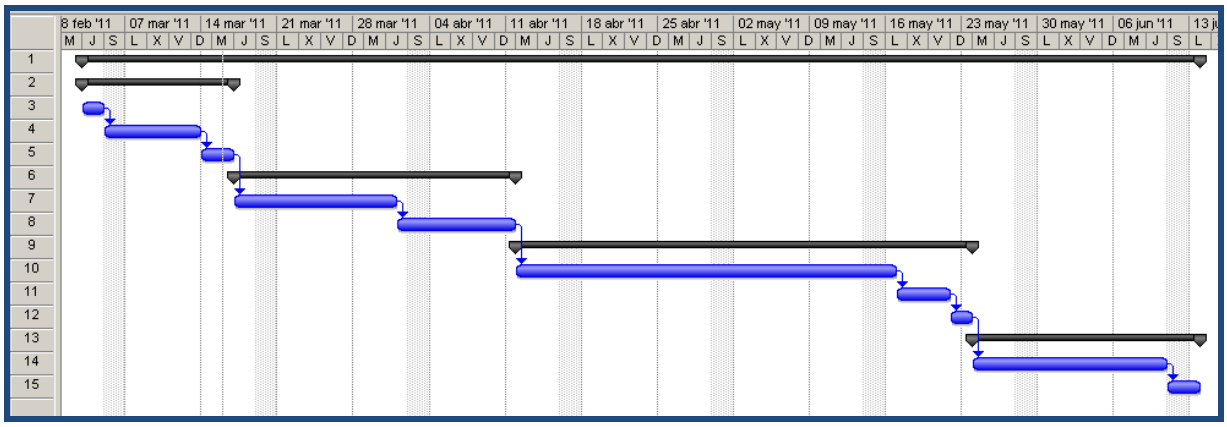

Figura 2 – Diagrama de Gantt inicial

3.2. Planificación real

En la planificación inicial faltaba un elemento muy importante, los hitos. Tan evidentes y documentadas me parecieron las entregas en el calendario de la asignatura, que obvié su inclusión como hitos en el Plan de Proyecto.

En el diagrama correspondiente a la planificación real ya están reflejados y son los siguientes:

1. Entrega Plan de trabajo 16/03/2011
2. Entrega Diseño 11/04/2011
3. Entrega Implementación 23/05/2011
4. Entrega Memoria 13/06/2011

Siguiendo con el Plan de Proyecto, allí señalé como principal riesgo el desconocimiento de gran parte de la tecnología a emplear. En las diferentes fases del Plan, no establecí un periodo concreto para la instalación de los componentes ni para la profundización en Silverlight.

Simplemente, en la fase inicial incluí una introducción a Silverlight que resultó completamente insuficiente. Consistió en seguir el manual proporcionado en los materiales "introducción a .NET" y con el curso on-line facilitado en el tablón

"Introducción a Silverlight". Ambos fueron sencillos y llevaderos aunque ninguno trataba el tema del acceso a datos.

Los problemas vinieron a posteriori. En la fase de análisis y diseño se me complicó la instalación del gestor de base de datos. Visual Studio 2010 permite seleccionar entre los productos a instalar el SQLServer Express. Lo que no instala por defecto es el SQL Server Management Studio.

La elección que tomé en un principio fue instalar el SGBD SQLServer 2008 R2, pero siempre daba un error cuyo origen no conseguía encontrar. Tras varios intentos, opté por desinstalar todo rastro de SQLServer del equipo para seguir con las pruebas, pero la instalación nunca conseguía terminar con éxito.

Como alternativa, instalé VMWare Player y configuré una máquina virtual con Windows 2003 Server. En ella instalé el SQLServer 2008 R2 correctamente. Paralelamente, encontré un artículo, recogido en la biografía, en el que explicaba cómo instalar el Management Studio sobre una instancia de SQLServer Express.

Siguiendo este artículo, volví a instalar SQLServer Express 2008 R2 y a continuación, SQL Server Management Studio 2008 R2, con lo que pude dar por finalizado este punto. En la planificación real he incluido la correspondiente tarea "Instalación SGBD".

Otro problema en el que invertí bastantes horas fue el de la conexión de Silverlight a SQLServer. Este apartado, no tenía tutoriales clarificadores similares a los de la introducción a Silverlight.

En la fase de análisis y diseño tomé la decisión de usar LINQtoSQL basándome en varios ejemplos en los que pude comprobar su funcionamiento. En la planificación real he incluido la correspondiente tarea "Conexión a SQL".

Después, en la fase de implementación y atendiendo las recomendaciones del tutor, modifiqué la estrategia para usar Entity Framework de .NET.

El último inconveniente surgido respecto a la parte de los datos fue la conexión entre las entidades del proyecto web y la aplicación Silverlight. Comencé con un tutorial de Microsoft en el que creaban por un lado la aplicación .NET y en la aplicación Silverlight, se añadía una referencia de servicio.

Después de muchos intentos, no me funcionaba como esperaba y tampoco conseguía entender el motivo. Con este tema estuve casi 2 semanas.

Encontré otro ejemplo en el que se veía más clara la conexión y las pruebas funcionaban correctamente. Se trataba de un proyecto Business Application de Silverlight con una Domain Service Class para la conexión al servicio de datos WCF.

Una vez decidida la parte de la conexión a los datos, vino la tarea de profundizar en el manejo de las entidades, las validaciones y la forma que tiene Silverlight de tratar los datos de forma asíncrona.

Esto último me trajo numerosos quebraderos puesto que perdía mucho tiempo intentando depurar el código para encontrar la causa que impedía que las consultas me devolvieran registros. En la planificación real he incluido las correspondientes tareas "Estudio de Silverlight", "Estudio de acceso a datos" y "Consulta de ejemplos".

Con todos estos comentarios y correcciones, la temporalización del proyecto fue la siguiente:

	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	**PFC .NET**	**103 días**	**jue 03/03/11**	**lun 13/06/11**	
2	**Plan trabajo**	**14 días**	**jue 03/03/11**	**mié 16/03/11**	
3	Elección de propuesta	2 días	jue 03/03/11	vie 04/03/11	
4	Introducción a Silverlight	9 días	sáb 05/03/11	dom 13/03/11	3
5	Elaboración Plan de trabajo	3 días	lun 14/03/11	mié 16/03/11	4
6	Entrega Plan de trabajo	0 días	mié 16/03/11	mié 16/03/11	5
7	**Diseño funcional y técnico**	**26 días**	**jue 17/03/11**	**lun 11/04/11**	
8	Instalación SGBD	14 días	jue 17/03/11	mié 30/03/11	6
9	Conexión a SQL	6 días	jue 31/03/11	mar 05/04/11	8
10	Diseño funcional	3 días	mié 06/04/11	vie 08/04/11	9
11	Diseño técnico	3 días	sáb 09/04/11	lun 11/04/11	10
12	Entrega Diseño	0 días	lun 11/04/11	lun 11/04/11	11
13	**Implementación**	**42 días**	**mar 12/04/11**	**lun 23/05/11**	
14	Estudio de Silverlight	16 días	mar 12/04/11	mié 27/04/11	12
15	Estudio de acceso a datos	13 días	jue 28/04/11	mar 10/05/11	14
16	Consulta de ejemplos	9 días	mié 11/05/11	jue 19/05/11	15
17	Implementación	3 días	vie 20/05/11	dom 22/05/11	16
18	Pruebas, manual y documentación	1 día	lun 23/05/11	lun 23/05/11	17
19	Entrega Implementación	0 días	lun 23/05/11	lun 23/05/11	18
20	**Memoria y presentación**	**21 días**	**mar 24/05/11**	**lun 13/06/11**	
21	Memoria	18 días	mar 24/05/11	vie 10/06/11	19
22	Presentación virtual	3 días	sáb 11/06/11	lun 13/06/11	21
23	Entrega Memoria	0 días	lun 13/06/11	lun 13/06/11	22

Figura 3 – Temporalización real

El diagrama de Gantt correspondiente se muestra a continuación:

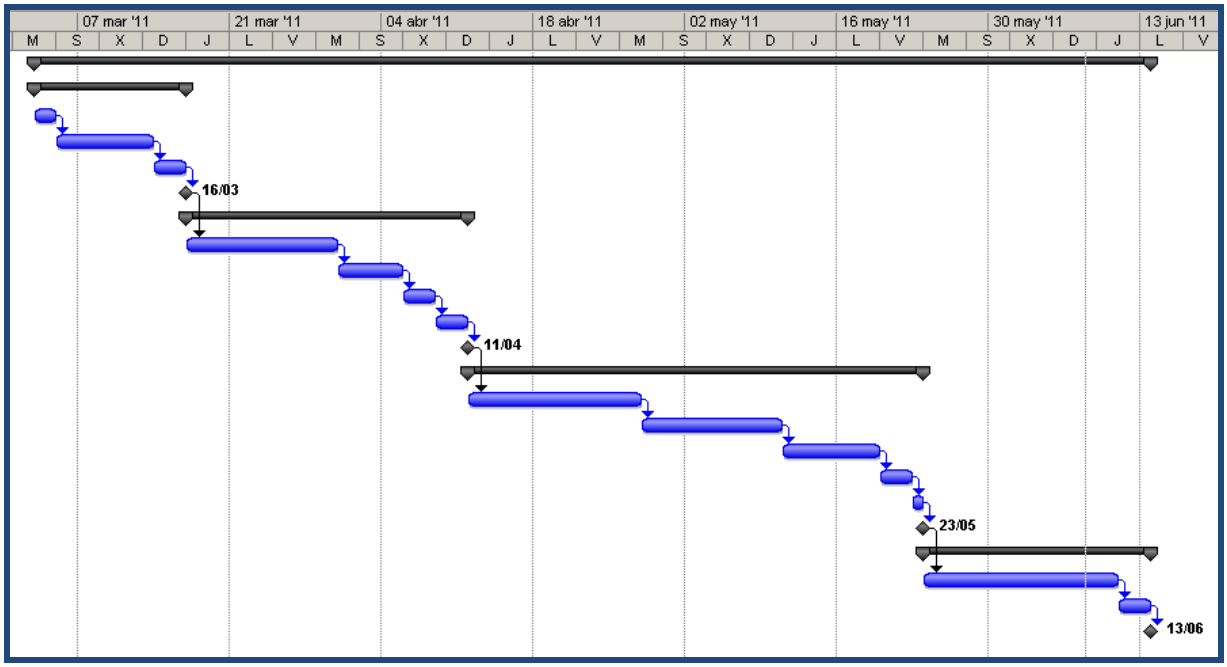

Figura 4 – Diagrama de Gantt real

Se puede apreciar que la parte dedicada a la implementación se redujo drásticamente. De 35 días se pasó a 3.

Las pruebas también se vieron afectadas ya que quedaron en menos de una jornada.

En el apartado "trabajo futuro y recomendaciones de mejora" se analizan todas las funcionalidades que se vieron afectadas.

4.Análisis, diseño e implementación

4.1. Requisitos

4.1.1. Requisitos funcionales:

1. Un usuario, previo registro, podrá introducir un ejemplar en el sistema. El ejemplar, será de una especie concreta. Si ya existe, la seleccionará de una lista, en caso contrario podrá darla de alta.

2. Cada especie, además podrá ser identificada por alguno de sus nombres comunes.

3. Cada ejemplar podrá actualizarse con observaciones o fotos para realizar un seguimiento de su evolución.

4. Si el usuario ha marcado su pertenencia a la comunidad, esta información podrá estar disponible para cualquier usuario.

5. Un usuario registrado podrá añadir comentarios en los ejemplares propios o de otros usuarios en la parte pública de cada ejemplar.

6. La aplicación permitirá a cualquier usuario (registrado o no) acceder a la información de los ejemplares de los usuarios adscritos a la comunidad y a los comentarios asociados.

7. El administrador tendrá la posibilidad de borrar un usuario o un comentario.

4.1.2. Requisito no funcional:

1. La aplicación se ejecutará indistintamente en la web, en el escritorio o podrá subirse a la nube.

4.2. Casos de uso

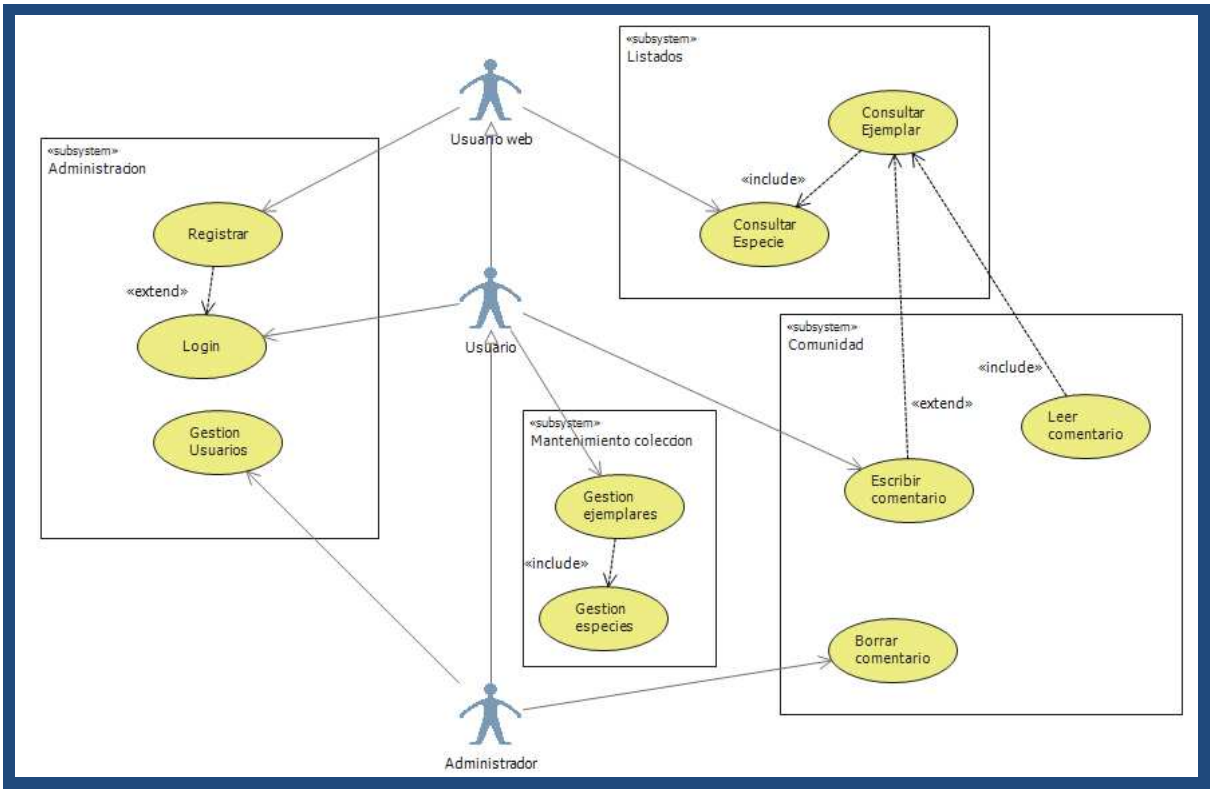

Figura 5 - Casos de uso

A continuación se describen los casos de uso:

4.2.1. Subsistema Administración

Caso de uso Registrar	
Resumen de la funcionalidad	Un usuario registrado quiere acceder al sistema
Actores	Usuario
Casos de uso relacionados	Login
Precondición	El usuario no está autenticado
Postcondición	El usuario queda autenticado
Proceso normal	Desde el apartado de Login, el usuario introduce sus credenciales y accede al sistema
Alternativas y excepciones	Al envíar la solicitud se comprobará que el

	identificador de usuario y la contraseña sean correctos

Caso de uso Login	
Resumen de la funcionalidad	Un usuario quiere darse de alta en el sistema
Actores	Usuario web
Casos de uso relacionados	Registrar
Precondición	El usuario no está registrado
Postcondición	El usuario queda registrado
Proceso normal	Desde el apartado de Login, el usuario accede al formulario para darse de alta. Una vez relleno, envía la solicitud y queda registrado
Alternativas y excepciones	Al envíar la solicitud se comprobará que el identificador no exista en el sistema y que los campos del formulario estén rellenos

Caso de uso Gestión Usuarios	
Resumen de la funcionalidad	El administrador accede al listado de los usuarios y puede consultar y modificar el estado de los mismos.
Actores	Administrador
Casos de uso relacionados	-
Precondición	El administrador accede al sistema
Postcondición	Se muestran los datos de los usuarios y se pueden modificar
Proceso normal	El administrador selecciona la opción Gestionar Usuarios para mostrar la lista de los usuarios dados de alta en el sistema
Alternativas y excepciones	Pinchando sobre el usuario accederá a las opciones de mantenimiento o borrado de los mismos

4.2.2. Subsistema Mantenimiento colección

Caso de uso Gestión Ejemplares	
Resumen de la funcionalidad	El usuario accede a sus ejemplares para dar de alta, consultar o modificarlos.
Actores	Usuario
Casos de uso relacionados	Gestión Especies
Precondición	El usuario accede al sistema
Postcondición	Se muestran los ejemplares del usuario
Proceso normal	El usuario selecciona la opción Gestionar Ejemplares donde se muestran los ejemplares del usuario
Alternativas y excepciones	Al seleccionar un ejemplar se podrá actualizar la información del mismo. Existirá la opción de crear un nuevo ejemplar

Caso de uso Gestión Especies	
Resumen de la funcionalidad	El usuario accede a la lista de las especies
Actores	Usuario
Casos de uso relacionados	Gestión Ejemplares
Precondición	El usuario accede desde la gestión de ejemplares
Postcondición	Se muestran las especies dadas de alta en el sistema
Proceso normal	El usuario puede añadir, modificar o borrar especies
Alternativas y excepciones	Se podrá seleccionar una Especie desde la lista o modificar alguna de ellas

4.2.3. Subsistema Listados

Caso de uso Consultar Especie	
Resumen de la funcionalidad	El usuario accede a la lista de las especies
Actores	Usuario web
Casos de uso relacionados	Consultar Ejemplar
Precondición	El usuario accede desde la pantalla de inicio
Postcondición	Se muestran las especies dadas de alta en el sistema
Proceso normal	El usuario seleccionará una para ver los ejemplares de

	dicha especie
Alternativas y excepciones	-

Caso de uso Consultar Ejemplar	
Resumen de la funcionalidad	El usuario accede a la ficha de un Ejemplar
Actores	Usuario web
Casos de uso relacionados	Consultar Especie, Leer Comentario, Escribir Comentario
Precondición	El usuario accede desde la lista de Especies
Postcondición	Se muestra la información de un ejemplar
Proceso normal	Al usuario se le mostrará la información de una Especie
Alternativas y excepciones	-

4.2.4. Subsistema Comunidad

Caso de uso Leer Comentario	
Resumen de la funcionalidad	El usuario visualiza los comentarios de un Ejemplar
Actores	Usuario web
Casos de uso relacionados	Consultar Ejemplar
Precondición	El usuario accede desde la ficha de un ejemplar
Postcondición	Se muestran los comentarios de un ejemplar
Proceso normal	El usuario podrá visualizar los comentarios de un ejemplar
Alternativas y excepciones	-

Caso de uso Escribir Comentario	
Resumen de la funcionalidad	El usuario escribe un comentario
Actores	Usuario
Casos de uso relacionados	Consultar Ejemplar
Precondición	El usuario tiene que autenticarse en la aplicación
Postcondición	El comentario queda reflejado en el sistema

Proceso normal	Desde la ficha de un ejemplar, el usuario escribe un comentario
Alternativas y excepciones	-

Caso de uso Borrar Comentario	
Resumen de la funcionalidad	El administrador borra un comentario
Actores	Administrador
Casos de uso relacionados	-
Precondición	El administrador tiene que autenticarse en la aplicación
Postcondición	El comentario queda borrado del sistema
Proceso normal	Desde la ficha de un ejemplar, el administrador borra un comentario
Alternativas y excepciones	-

4.3. Modelo Conceptual

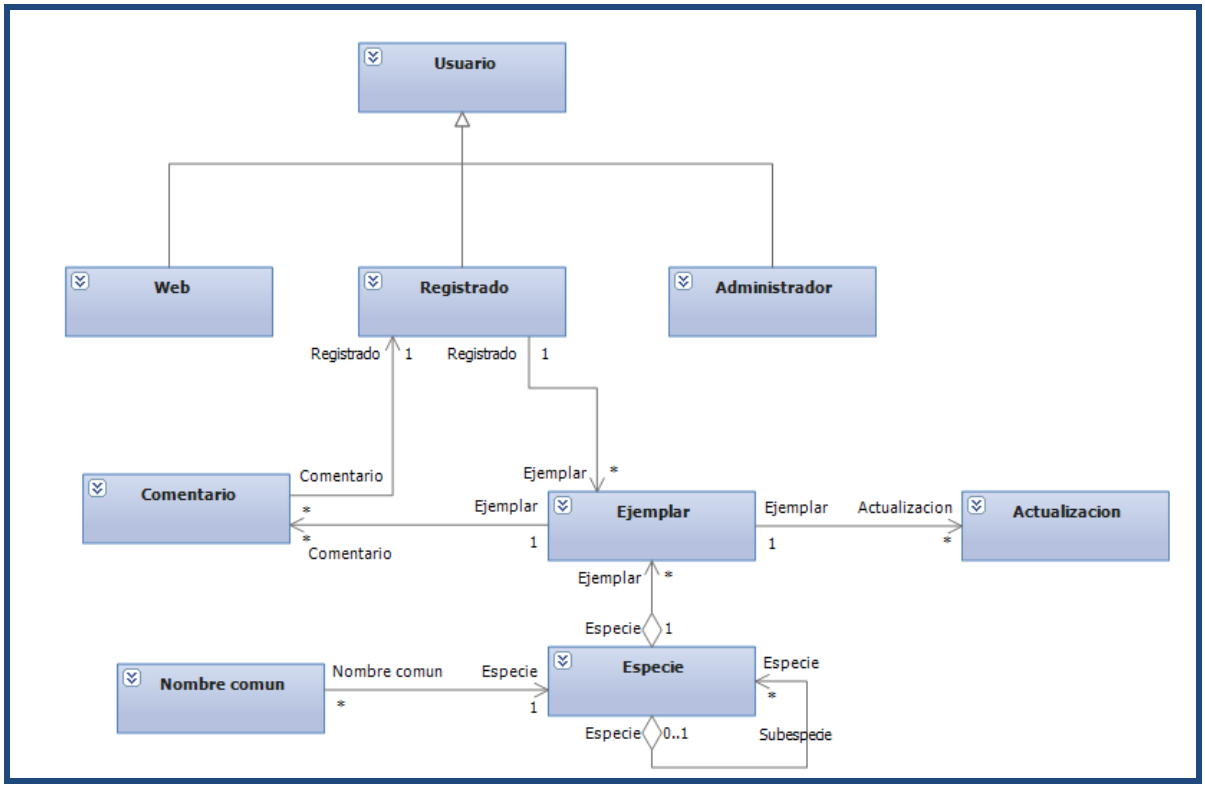

Figura 6 – Diagrama del modelo conceptual

Comentarios al modelo:

- Existirán 3 tipos de usuarios. El único que va a interactuar con alguna clase es el registrado, que podrá disponer de un inventario actualizado de sus ejemplares y podrá participar en la comunidad por medio de comentarios.

- Para facilitar la clasificación de los ejemplares, dentro de las especies podrán existir subespecies. En cualquier caso, todos podrán tener asociado uno o varios nombres comunes.

4.4. Diagrama de Arquitectura

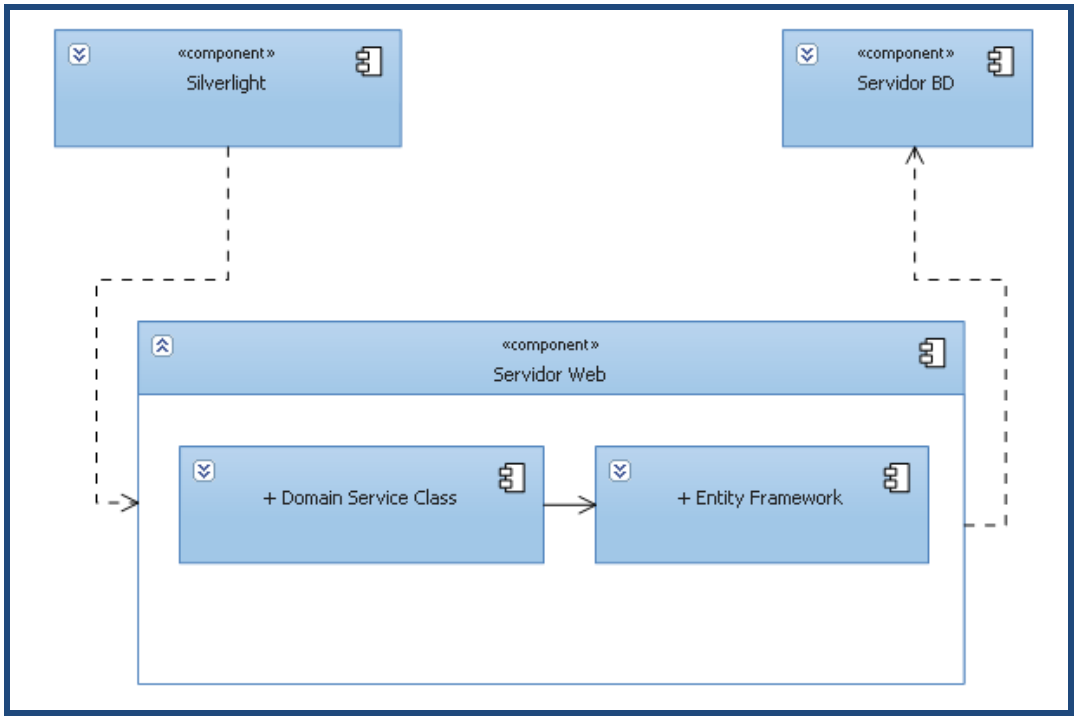

Figura 7 – Diagrama de componentes

Comentarios al diagrama:

- El interfaz gráfico de usuario se implementa por medio de una aplicación Silverlight. Por un lado presenta la información al usuario y por otro, recoge las llamadas del mismo a las clases del programa.

- En el servidor Web, se aloja el modelo de negocio y el modelo de entidades:

  - Para crear el modelo de entidades he usado Entity Framework de .NET, y he generado el modelo a partir de la base de datos.
  - Para el modelo de negocio he usado una Domain Service Class, que automáticamente, genera los métodos esenciales para el acceso a las entidades y contiene el resto de lógica de la aplicación.

- El servidor de base de datos se ocupa de la persistencia de las entidades del modelo conceptual.

4.5. Modelo de clases del diseño

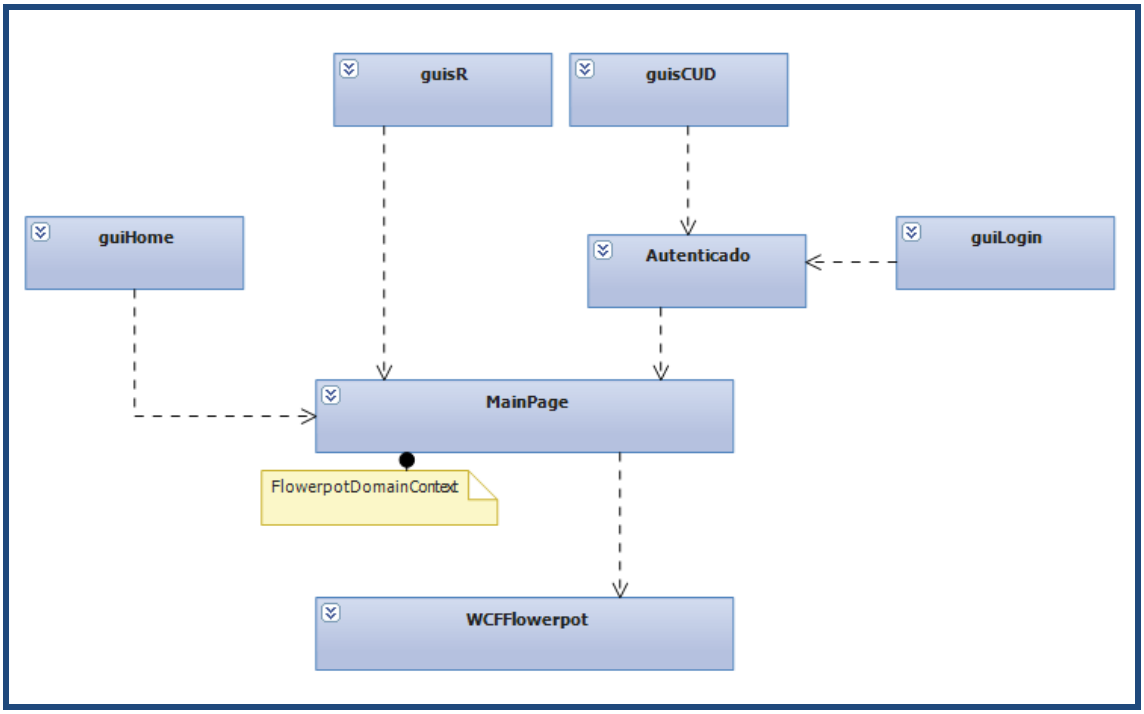

Figura 8 – Diagrama de clases del diseño

Comentarios al diagrama:

- Cada interfaz gráfica de usuario, es representada con una clase 'gui____'.

- El workflow de la aplicación es gestionado por la clase MainPage. En ella se controla la navegación por las distintas pantallas del programa.

- La clase guiHome corresponde a la pantalla de inicio, y la clase guiLogin a la interfaz de autenticación, que actualizará la clase Autenticado.

- A cada clase del modelo conceptual le corresponde un par de clases frontera del modelo de diseño:

  1. Una pantalla de visualización guiR (Read).
  2. Una pantalla de actualización de datos guiCUD (Create, Update y

Delete).

- Estas clases frontera son Especie, Ejemplar, Actualizacion, Comentario y Usuario.

- La herencia de la clase Usuario a las subclases Web, Registrado y Administrador del modelo conceptual, ha quedado reducida a un atributo administrador de tipo boolean en la clase Usuario, ya que de un usuario web (sin registrar) no se guarda ninguna información y un administrador se diferenciará de un usuario registrado en el valor del atributo mencionado.

- Para simplificar el esquema, he agrupado las pantallas de visualización de entidades en la clase guisR (Read) y las de actualización de datos en la clase guisCUD(Create, Update y Delete).

- Estas últimas, comprueban en la clase Autenticado, las credenciales del usuario que intenta actualizar datos.

- La clase WCFFlowerpot contiene la lógica de negocio y la conexión con las clases de entidad. La aplicación cliente la implementa con la clase FlowerpotDomainContext.

4.6. Diseño de la interfaz gráfica

En la fase de diseño se realizó un esbozo de las pantallas de la aplicación con el programa Justinmind Prototyper 4.1.1 (http://www.justinmind.com).

En la presente memoria, no vuelvo a incluir estos prototipos sino que muestro directamente la interfaz gráfica diseñada en Silverlight, en el apartado "Implementación".

4.7. Diseño de la base de datos

En el análisis de la aplicación encontrábamos la organización Especie:Subespecie dando lugar a la creación de la clase Subespecie.

A cada una de estas Especies o Subespecies podíamos asignarles uno o varios nombres comunes, dando lugar en este caso a la clase NombreComun.

En el paso al modelo relacional, a cada clase le correspondía una tabla.

A la hora de implementar la aplicación, he modificado el diseño de la base de datos con el objetivo de eliminar complejidad y reducir el tiempo de desarrollo.

Con los cambios realizados se elimina la gestión de Especie:Subespecie y NombreComun se convierte en un atributo de la clase Especie, por lo que una especie sólo podrá tener uno o ningún nombre común.

Con estas modificaciones, el esquema de la base de datos Flowerpot queda como se muestra a continuación:

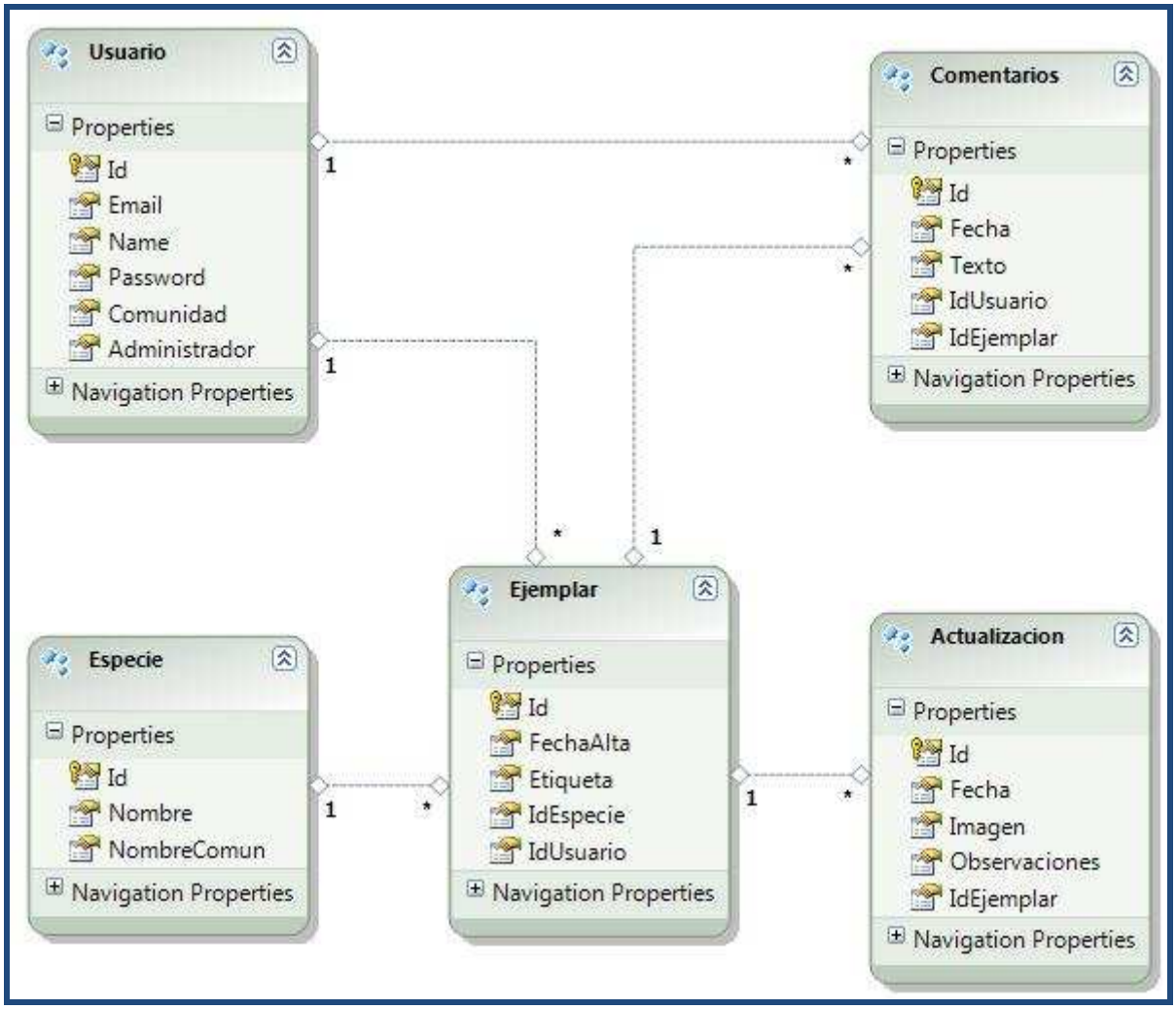

Figura 9 – Diagrama de la base de datos

El script de creación de base de datos de la aplicación Flowerpot para SQLServer es el siguiente:

```
CREATE TABLE Especie
(
Id INT IDENTITY (1,1) NOT NULL,
Nombre VARCHAR(50) NOT NULL ,
NombreComun VARCHAR(50) NULL,
CONSTRAINT PK_Especie PRIMARY KEY (Id)
);

CREATE TABLE Usuario
(
Id INT IDENTITY (1,1) NOT NULL,
Email VARCHAR(50) NOT NULL,
Name VARCHAR(50) NOT NULL,
Password VARCHAR(10) NOT NULL,
Comunidad BIT NOT NULL,
Administrador BIT NOT NULL,
CONSTRAINT pk_Usuario PRIMARY KEY (Id)
);

CREATE TABLE Ejemplar
(
Id INT IDENTITY (1,1) NOT NULL,
FechaAlta DATETIME2 NOT NULL,
Etiqueta VARCHAR(50) NOT NULL,
IdEspecie INT NOT NULL,
IdUsuario INT NOT NULL,
CONSTRAINT pk_Id PRIMARY KEY (Id),
CONSTRAINT fk_EjemplarUsuario FOREIGN KEY (IdUsuario) REFERENCES
Usuario(Id),
CONSTRAINT fk_EjemplarEspecie FOREIGN KEY (IdEspecie) REFERENCES
Especie(Id)
);

CREATE TABLE Comentarios
(
Id INT IDENTITY (1,1) NOT NULL,
Fecha DATETIME2 NOT NULL,
```

```
Texto VARCHAR(500) NOT NULL,
IdUsuario INT NOT NULL,
IdEjemplar INT NOT NULL,
CONSTRAINT pk_Comentarios PRIMARY KEY (Id),
CONSTRAINT fk_ComentariosUsuario FOREIGN KEY (IdUsuario) REFERENCES
Usuario(Id),
CONSTRAINT fk_ComentariosEjemplar FOREIGN KEY (IdEjemplar)
REFERENCES Ejemplar(Id)
);

CREATE TABLE Actualizacion
(
Id INT IDENTITY (1,1) NOT NULL,
Fecha DATETIME2 NOT NULL,
Imagen VARCHAR(500),
Observaciones VARCHAR(500),
IdEjemplar INT NOT NULL,
CONSTRAINT pk_Actualizacion PRIMARY KEY (Id),
CONSTRAINT fk_ActualizacionEjemplar FOREIGN KEY (IdEjemplar)
REFERENCES Ejemplar(Id)
);

INSERT INTO Usuario (Email, Name, Password, Comunidad,
Administrador) VALUES('admin@flower.com', 'admin', 'admin', 0, 1);
```

La última instrucción corresponde al alta de un usuario administrador de la aplicación con nombre admin y contraseña admin para poder comenzar a usar la aplicación.

4.8. Implementación

- Todo el desarrollo está basado en la tecnología .NET de Microsoft (Silverlight y WCF). El sistema gestor de base de datos es Microsoft SQL Server 2008. La plataforma de desarrollo es Visual Studio y el lenguaje de programación es C#.

- La aplicación está basada en el template de Visual Studio 2010 llamado "Silverlight Business Application". La solución se divide en un proyecto Web (flowerpot.Web) y en la aplicación de Silverlight propiamente dicha (flowerpot).

- La base de datos está creada en Microsoft SQLServer 2008 R2. La implementación, en mi caso, ha sido sobre una instancia de SQLEXPRESS. Para ello he usado un script de creación de tablas.

- Para crear el modelo de entidades he usado Entity Framework de .NET, y he generado el modelo a partir de la base de datos.

- Para el modelo de negocio he usado una Domain Service Class, que automáticamente, genera los métodos esenciales para el acceso a las entidades. En este ámbito, he hecho varias modificaciones para como parte de la lógica de mi aplicación.

- En cuanto a recuperar entidades en función de parámetros he añadido en la clase flowerpotDomainService.cs las siguientes funciones:

  - GetActualizacionEjemplar(int idEjemplar): Actualizaciones de un ejemplar.
  - GetComentarioEjemplar(int idEjemplar): Comentarios de un ejemplar.
  - GetEjemplarEspecie(int idEspecie): Ejemplares de una especie.
  - GetEjemplarUsuario(int idUsuario): Ejemplares de un usuario.

- GetLogin(string Nombre, string Password): Verifica el login al sistema.
- GetUsuarioId(int idUsuario): Datos de un usuario en concreto.

- En cuanto a validación de datos he configurado en la clase flowerpotDomainService.metadata.cs las siguientes restricciones, incluyendo el texto que debe mostrar la aplicación en caso de que no se cumplan:

1. Propiedad Observaciones requerida en la entidad Actualizaciones:

```
[Required(AllowEmptyStrings = false, ErrorMessage = "Debe rellenar
el campo Observaciones")]
```

2. Propiedad Texto requerida en la entidad Comentarios:

```
[Required(AllowEmptyStrings = false, ErrorMessage = "Debe
introducir un Comentario")]
```

3. Propiedad Etiqueta requerida en la entidad Especies:

```
[Required(AllowEmptyStrings=false,ErrorMessage="Debe rellenar el
campo Etiqueta")]
```

4. Propiedad email bien formado en la entidad Usuarios:

```
[RegularExpression(@"^([\w-\.]+)@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-
9]{1,3}\.)|(([\w-]+\.)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\]?)$",
ErrorMessageResourceName = "ValidationErrorInvalidEmail",
ErrorMessageResourceType = typeof(ValidationErrorResources))]
```

5. Propiedad Name no repetido de la entidad Usuarios:

```
[CustomValidation(typeof(flowerpot.Web.Services.UserValidator),
"AsyncValidateUser")]
```

Para esta última he usado una validación personalizada implementada

en la clase UserValidator, que hace uso de la función boolean IsUserExist(string userName) definida en flowerpotDomainService.

En la aplicación Silverlight, las principales cuestiones a tener en cuenta son el uso de la clase Autenticado para almacenar si el usuario está validado en el sistema, si es propietario de los ejemplares a los que accede y si es un administrador.

Sobre la base de esta información, se van a habilitar/deshabilitar los botones de modificación/borrado de Especies y Ejemplares (incluyendo los comentarios y actualizaciones) y se va a restringir la gestión de la información de los usuarios.

Los permisos se resumen en la siguiente tabla:

	No Login	**Login**	**Admin**
Especies Consultar	X	X	X
Especies Añadir/Editar/Borrar		X	X
Ejemplares Consultar	X	X	X
Ejemplares Añadir		X	X
Ejemplares Editar/Borrar		Propietario	X
Actualizaciones Consultar	X	X	X
Actualizaciones Añadir/Editar/Borrar		Propietario	X
Comentarios Consultar	X	X	X
Comentarios Añadir		X	X
Comentarios Editar/Borrar		Propietario	X
Usuarios Alta	X	X	X
Usuarios Consultar/Editar/Borrar		Propietario	X

Figura 10 – Permisos de usuarios sobre funciones

- El workflow de la aplicación consiste en que en la pantalla de inicio se selecciona un ejemplar perteneciente a una especie o a un usuario, y desde allí se pasa a la pantalla de Comentarios o Actualizaciones con el ID del ejemplar como parámetro en la url. Si esto lo hace un usuario autenticado, guardo en la clase Autenticado si dicho usuario es el propietario del ejemplar, puesto que hay que tenerlo en cuenta a la hora de mostrar los botones como he indicado en el párrafo anterior.

- La clase Autenticado se actualiza cuando se hace Login o Logout.

- La tabla con todos los usuarios sólo es mostrada a los administradores. Un usuario logueado sólo podrá ver/modificar o borrar sus propios datos. La propiedad Administrador de la ficha de usuario sólo es modificable por los administradores.

- La consulta de la información está basada en grids y los mantenimientos en pantallas ChildWindow que cargo a partir de la entidad seleccionada en cada momento en el grid.

- La única salvedad se hace en el mantenimiento de los ejemplares, en el que antes de abrir la ventana para Editar, cargo las especies y los usuarios para que se muestren en un ComboBox. Si se hace una edición, hay que tener en cuenta que hay que mostrar el ComboBox con la especie y el usuario correspondiente ya seleccionados. Y si se hace un alta, el Identificador del Usuario será el del usuario logueado en la aplicación en ese momento.

4.9. Interface de usuario

A continuación se muestran las pantallas de la aplicación.

Existen ciertas funcionalidades que no requieren autenticación y que se pueden usar al arrancarla. Es el caso, por ejemplo, de la pantalla de inicio:

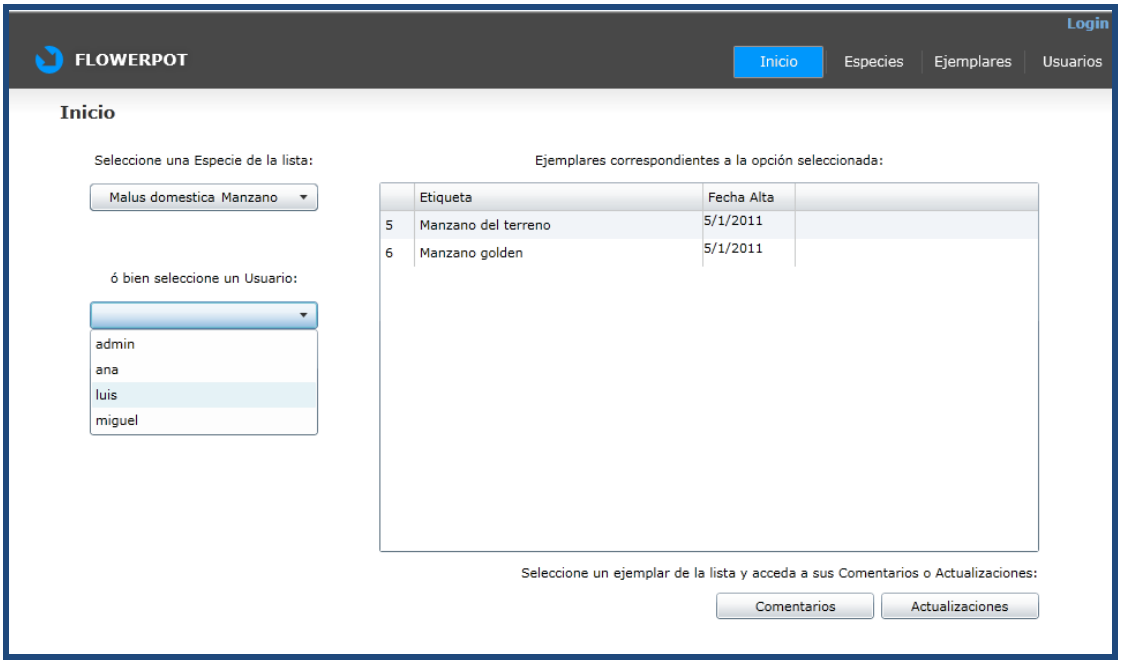

Figura 11 – Pantalla de inicio

El interface de la aplicación está estructurado en 2 partes:

- Una franja superior fija que alberga el logotipo de la aplicación, el menú para navegar por las distintas opciones del programa y la parte de la autenticación.
- Una pantalla inferior donde se muestra la información de las entidades.

Para acceder como usuario registrado, hay que pinchar en la palabra **login** de la franja superior.

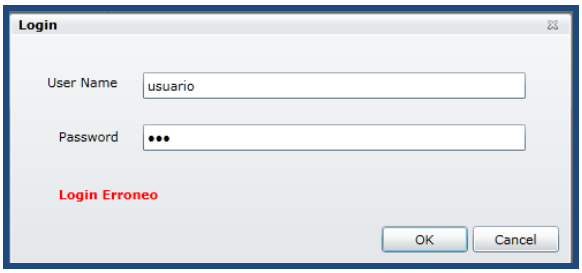

Figura 12 – Pantalla de autenticación

El administrador accede con el usuario *admin* y la contraseña *admin*.

Para registrarse como nuevo usuario hay que pinchar en el enlace *Usuarios* de la franja superior y después en el botón *Registrar*.

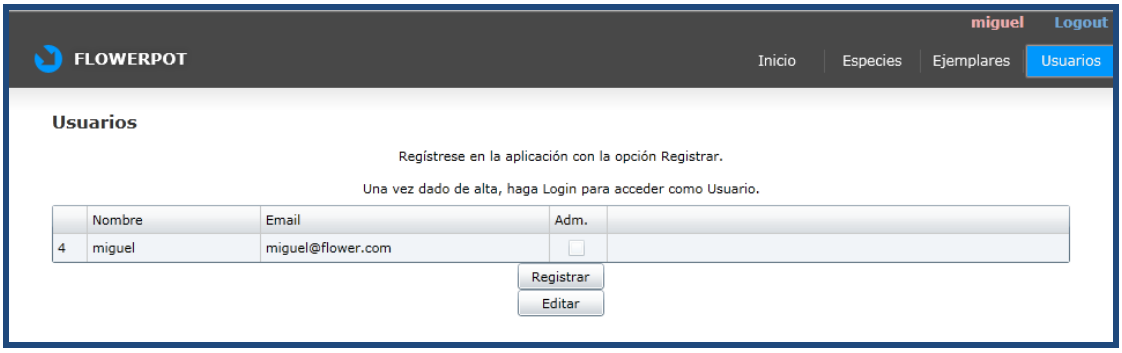

Figura 13 – Pantalla de visualización de usuarios

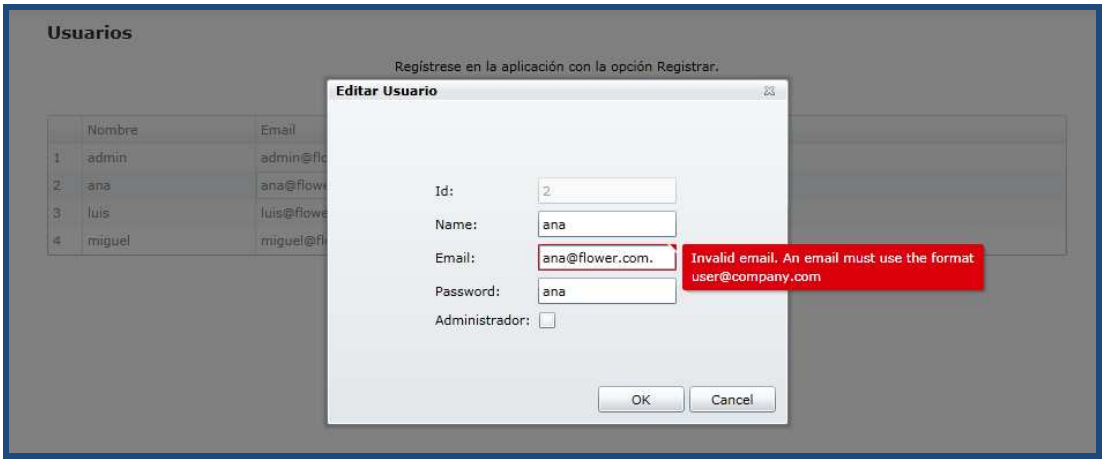

Figura 14 – Pantalla de añadir/editar usuarios

La mecánica de la aplicación consiste en seleccionar una especie o un usuario de alguno de los Combos. Esta acción puebla el DataGrid de los ejemplares con los

correspondientes a la especie o al usuario seleccionado.

Seleccionando uno de los ejemplares del DataGrid de la pantalla de inicio, existen 2 opciones:

1. Pulsando el botón *Comentarios*, se visualizan los Comentarios que cualquier usuario hace de cualquier ejemplar.

Figura 15 – Pantalla de visualización de comentarios

2. Pulsando el botón *Actualizaciones* se consiguen visualizar las Actualizaciones que un usuario hace de sus ejemplares.

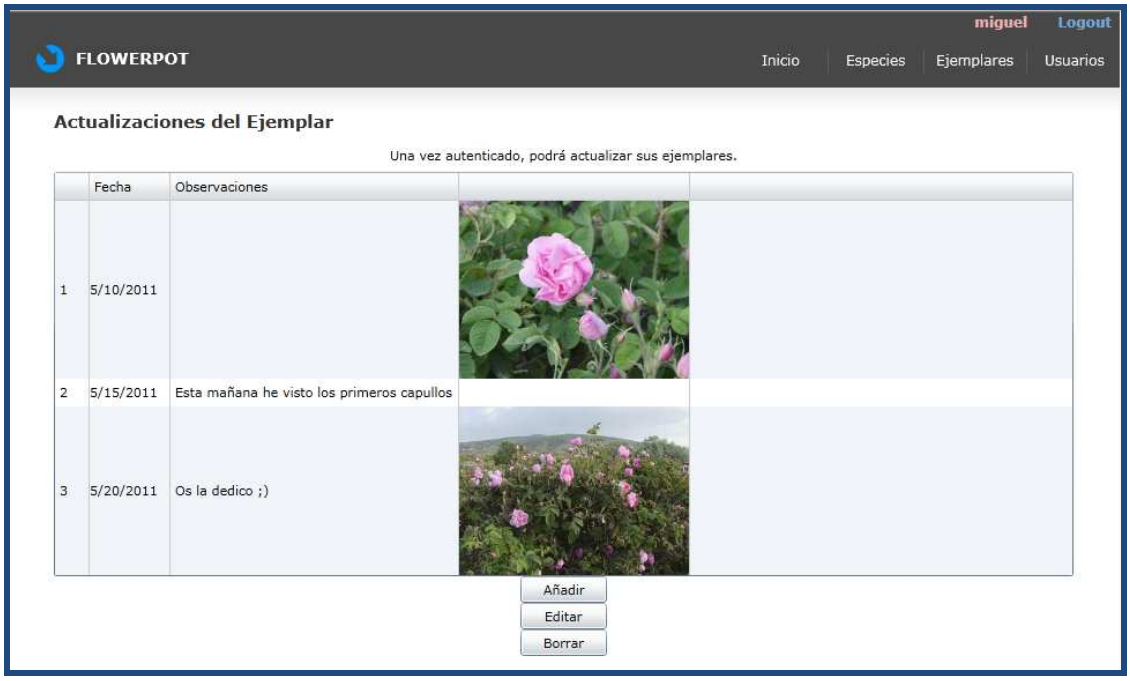

Figura 16 – Pantalla de visualización de actualizaciones

Una vez logueado, un usuario podrá añadir/editar/borrar actualizaciones y comentarios, siguiendo los criterios definidos en el apartado anterior, en concreto en la Figura 10 – Permisos de usuarios sobre funciones

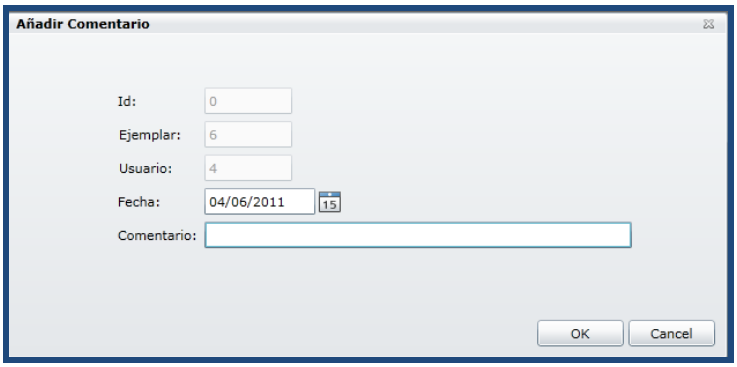

Figura 17 – Pantalla de añadir/editar comentarios

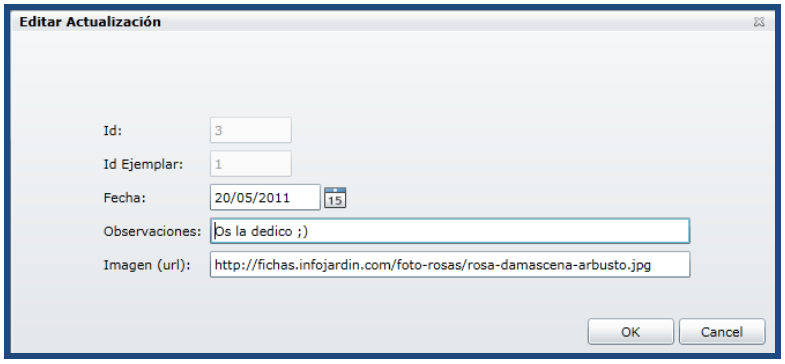

Figura 18 – Pantalla de añadir/editar actualizaciones

Para crear/modificar/borrar una Especie se selecciona el enlace *Especies* de la franja superior:

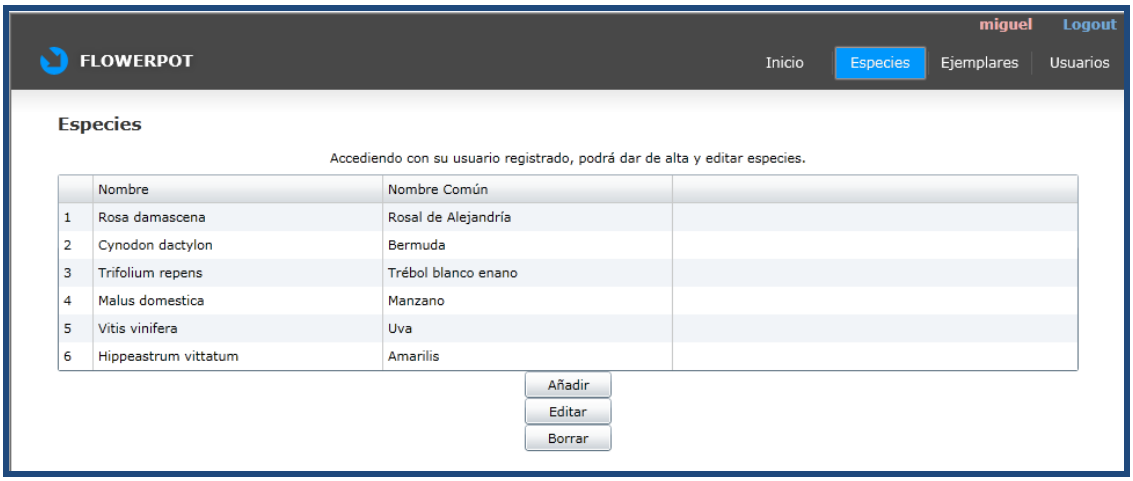

Figura 19 – Pantalla de visualización de especies

Para crear/modificar/borrar un Ejemplar se selecciona el enlace *Ejemplares* de la franja superior:

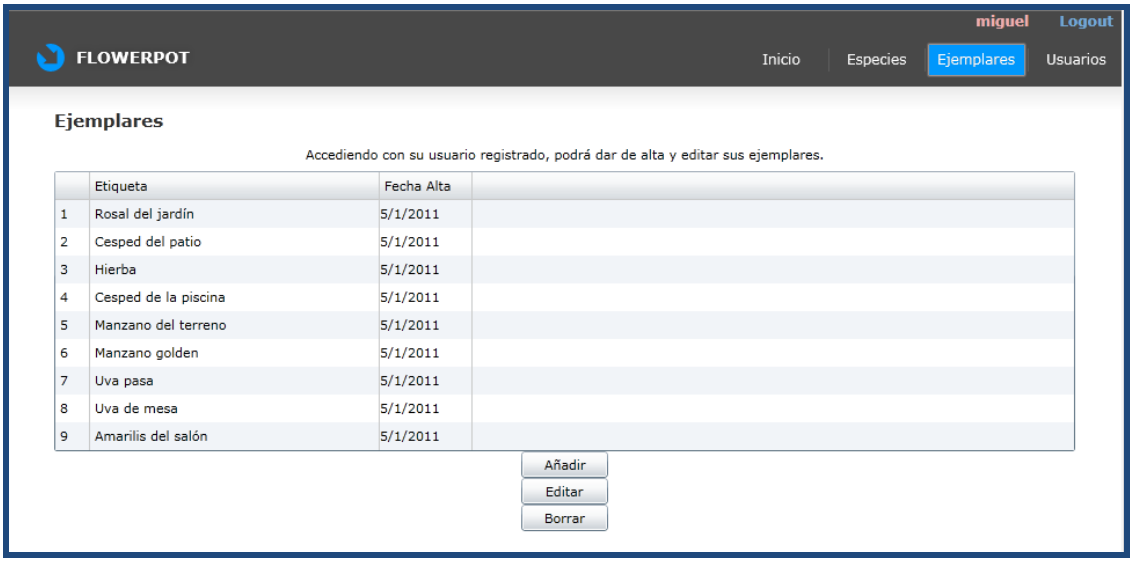

Figura 20 - Pantalla de visualización de ejemplares

En ambas pantallas se ofrecen varias posibilidades:

- Pulsando el botón *Añadir* se accede a la pantalla de alta de una nueva especie o ejemplar.

- Seleccionando uno de los items del DataGrid, se puede editar o borrar dicha selección pulsando sobre el botón *Editar* o *Borrar* respectivamente.

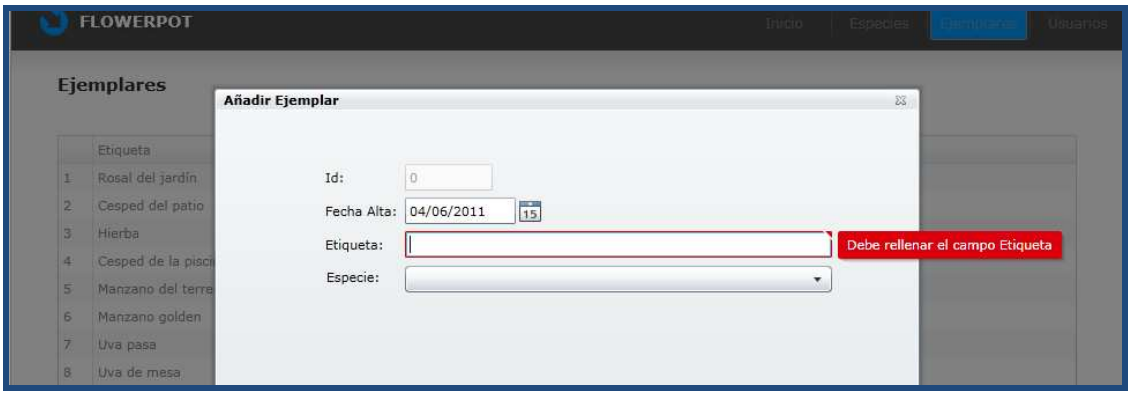

Figura 21 – Pantalla de añadir ejemplares

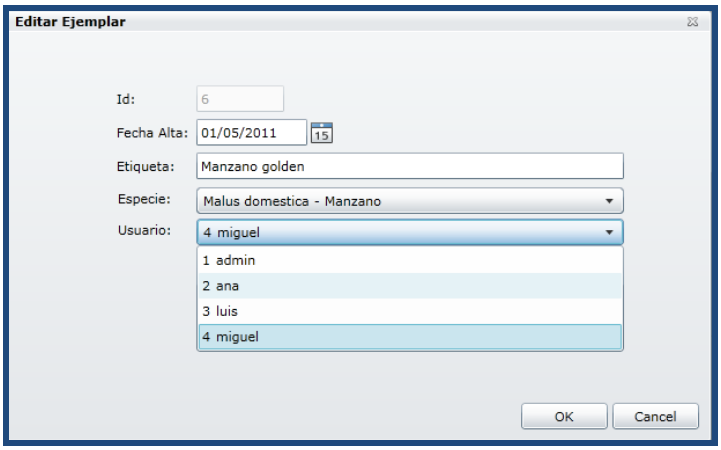

Figura 22 - Pantalla de edición de ejemplares

5.Objetivos conseguidos

El objetivo del proyecto era desarrollar la aplicación de gestión de plantas ornamentales Flowerpot.

Dicho objetivo se ha cumplido, ya que a la finalización de este proyecto, se ha obtenido una aplicación perfectamente funcional llamada Flowerpot que permite la gestión de plantas ornamentales.

Cabe decir, en honor a la verdad, que el objetivo planteado de esa manera admite un amplio espectro de aplicaciones llamadas Flowerpot.

Si entramos en detalle en el plan de trabajo, como he comentado en el apartado "planificación real", han quedado varios aspectos sin tratar. En el apartado "trabajo futuro y recomendaciones de mejora" se hace un análisis de los mismos.

El enunciado de la propuesta de proyecto, recogía además como objetivos, el estudio de la tecnología de Silverlight como plataforma de desarrollo de aplicaciones multi-entorno y el desarrollo de una aplicación, con una arquitectura que ofreciera, escalabilidad y que permitiera la comunicación a bases de datos, WCF o servicios Web.

No creí conveniente incluirlos en el plan de trabajo, puesto que elegí un enfoque de un proyecto real de una empresa en lugar de un trabajo académico. Llegados a este punto cabe reseñar que ambos objetivos se han cumplido; se ha obtenido una aplicación en Silverlight que se comunica con WCF a una base de datos y para ello ha habido que profundizar en la tecnología Silverlight.

6. Evaluación de costes

La estimación del coste de un proyecto en el ámbito académico, no se va a corresponder con la estimación de un proyecto empresarial, pero no deja de ser curioso comprobar las similitudes que pueden llegar a tener en un momento concreto.

Obviamente se trata de un ejercicio teórico. Para comenzar vamos a seleccionar los perfiles que podían haber intervenido en la realización del proyecto. Estaríamos hablando de un Jefe de proyecto, un Consultor y un Desarrollador.

El precio/hora asignado a cada uno de los perfiles ha sido el siguiente:

- Jefe de Proyecto: 60 €/hora.
- Consultor: 50 €/hora.
- Desarrollador: 36 €/hora.

El reparto de actividades a cada perfil se corresponde en su práctica totalidad con el siguiente esquema de asignación:

- Apertura y cierre de proyecto → Jefe de Proyecto
- Análisis y diseño → Consultor
- Implementación → Desarrollador

La semana de trabajo se ha establecido de Lunes a Domingo. Para el cálculo de horas empleadas en cada jornada, se ha establecido una media de duración de 2,5 horas cada una. En la práctica ha habido días que no se ha invertido ninguna hora y otros en los que se han llegado a invertir 12 horas.

Tomando como base la planificación real y siguiendo los criterios comentados en los párrafos anteriores, a continuación se muestra la asignación de recursos a las tareas y el coste de cada tarea:

Tarea	Días	Recurso	Precio/hora	Precio/tarea*
Plan trabajo				
Elección de propuesta	2	Jefe Proy.	60,00 €	300,00 €
Introducción a Silverlight	9	Desarrollador	36,00 €	810,00 €
Elaboración Plan de trabajo	3	Jefe Proy.	60,00 €	450,00 €
Diseño funcional y técnico				
Instalación SGBD	14	Desarrollador	36,00 €	1.260,00 €
Conexión a SQL	6	Consultor	50,00 €	750,00 €
Diseño funcional	3	Consultor	50,00 €	375,00 €
Diseño técnico	3	Consultor	50,00 €	375,00 €
Implementación				
Estudio de Silverlight	16	Desarrollador	36,00 €	1.440,00 €
Estudio de acceso a datos	13	Consultor	50,00 €	1.625,00 €
Consulta de ejemplos	9	Desarrollador	36,00 €	810,00 €
Implementación	3	Desarrollador	36,00 €	270,00 €
Pruebas, manual y comentarios	1	Desarrollador	36,00 €	90,00 €
Memoria y presentación				
Memoria	18	Jefe Proy.	60,00 €	2.700,00 €
Presentación virtual	3	Jefe Proy.	60,00 €	450,00 €
TOTAL	**103**			**11.705,00 €**

Figura 23 – Estimación económica del proyecto

*Horas imputadas a cada jornada: 2,5 horas

La estimación económica asciende a 11.705 €.

Si estuviéramos ante un proyecto real, en este momento habría que comparar la calidad del resultado obtenido, con las características del aplicativo proyectado en el plan de trabajo.

La aplicación entregada, no cumple con los requisitos de integración con elementos ricos que se había proyectado.

En el apartado "planificación real" han quedado reflejados los problemas derivados del desconocimiento de la tecnología a emplear y en el de "trabajo futuro y recomendaciones de mejora" las consecuencias en la aplicación.

Obviamente, una empresa que presupueste una aplicación con Silverlight, no va a tener este tipo de inconvenientes, puesto que tendrá personal experimentado.

Por otro lado, la experiencia con proyectos software también demuestra que en ocasiones las empresas de servicios contratan al personal una vez que se han asegurado que el proyecto está aprobado, por lo que la fase de aprendizaje puede solaparse con alguna de las fases del proyecto.

En la elaboración de este PFC es justamente lo que ha sucedido, la fase de aprendizaje y la de realización del proyecto han coincidido en el tiempo.

7.Trabajo futuro y recomendaciones de mejora

Como he mencionado en el apartado "planificación real", el riesgo del desconocimiento de gran parte de la tecnología a emplear, se ha convertido en la principal contingencia de este proyecto.

En dicho apartado he realizado un repaso pormenorizado de los inconvenientes aparecidos. A continuación hago un análisis de las consecuencias tras la aparición del riesgo.

En el Plan de Proyecto se indicaba la posibilidad de que la aplicación hiciera uso de varios elementos ricos. En concreto se señalaban:

- el control de captura de imágenes por la webcam del ordenador.
- el uso de Bing para buscar información adicional en la web acerca de las especies.
- el uso de Flicker como contenedor de las imágenes.
- el control Bing Maps, para señalar la localización de las plantas.

Todo ello quedaba supeditado a la complejidad del proyecto, y al tiempo requerido para las tareas básicas. En los tutoriales seguidos al inicio del semestre, llegué a comprobar el funcionamiento de los dos primeros controles con sendos ejemplos. Al final no me quedó tiempo de integrarlos en la aplicación.

Por los mismos problemas de tiempo, no logré dar con la documentación ni con los ejemplos para implementar los controles Flickr y Maps. Sólo encontré ejemplos en .NET, pero no en Silverlight. Llegué a integrar la API de Flickr, me hice con las credenciales de acceso a Flickr como desarrollador y conseguí que me funcionara un buscador de fotos con un visor pero no me servía de nada en mi aplicación.

Por último, como he señalado en el apartado "diseño de la base de datos", a la

hora de implementar la aplicación, he modificado dicho diseño con el objetivo de eliminar complejidad y reducir el tiempo de desarrollo.

Los cambios realizados han consistido en eliminar la gestión de Especie:Subespecie, en unificar el trato a los usuarios sin diferenciar si quieren pertenecer o no a la comunidad y en transformar la clase NombreComun en un atributo de la clase Especie, por lo que una especie sólo podrá tener uno o ningún nombre común.

El trabajo futuro inmediato consistiría en incluir estas características en la aplicación, para cumplir con lo acordado en el Plan de Proyecto.

Las pruebas quedaron en menos de una jornada que sirvió para depurar numerosos errores. Seguramente seguirán aflorando fallos cuando se comience a usar la aplicación puesto que no fue tiempo suficiente para realizar un inventario de pruebas más exhaustivo como tenía previsto al planificar el proyecto.

Por ello, la siguiente recomendación para este proyecto y para cualquier otro, sería depurar la aplicación para evitar la aparición de errores a los usuarios.

Otro punto que quedó en el tintero fue el relativo a probar la aplicación en la nube, que era uno de los requisitos del enunciado del proyecto. A este respecto cabe comentar que la aplicación Silverlight es válida para ser usada en Office 365 y el servicio WCF puede ser publicado en Azure.

Con vistas al manejo de la aplicación, se incluyen textos en las páginas para mejorar la experiencia del usuario. Para facilitar la usabilidad, el programa debería incluir tooltips informativos en controles de tipo botón o en imágenes.

Para el despliegue de la BD, sería interesante automatizar el despliegue de la misma al instalar la aplicación ya sea vía script o con un programa instalable.

Dejo para el final un problema de base que requeriría volver a implementar la

aplicación desde cero. En el diseño de la aplicación, se proponía una división en capas que no ha podido ser llevada a la práctica, en gran parte propiciada por desconocimiento personal. La estructura de proyecto, es bastante caótica y resulta difícil distinguir las capas de la aplicación.

Para mejorar este punto, se podrían organizar los componentes por carpetas o bien separando claramente las capas de la aplicación en distintos proyectos: acceso a datos, capa de negocio y capa de presentación.

Siguiendo esta organización, el modelo de entidades debería estar separado del proyecto web, ya que no es buena práctica cuando se trabaja con aplicaciones en n capas.

8.Conclusiones

El desarrollo de este Proyecto Final de Carrera me ha permitido introducirme en la plataforma de desarrollo de aplicaciones .NET de Microsoft. La lástima es que no ha pasado de ser una introducción.

Lo que esperaba en un principio era poder profundizar más en virtud de la experiencia que contaba en Visual Studio desarrollando aplicaciones con Visual Basic 6 y Visual Fox Pro. Fue un periodo entre los años 2000 y 2002 al final de la carrera de I.T. Informática de Sistemas y al comienzo de mi vida laboral como desarrollador en una empresa de servicios.

Aunque he ido alcanzando las diferentes fases, han sido unos meses de bastante estrés debido, principalmente, a mi desconocimiento sobre la arquitectura .NET. La experiencia previa poco tenía que ver con este IDE, con el lenguaje de programación C#, con la programación orientada a objetos y con la novedosa tecnología Silverlight. Además debe mediar un cambio de mentalidad para evolucionar desde la programación estructurada de Visual Basic.

En ocasiones daba la impresión que estaba intentando hacer funcionar una aplicación sin saber muy bien cómo lo hacía. La automatización a la hora de crear una aplicación "Silverlight Business Application" que por un lado crea un proyecto Web y por otro una aplicación Silverlight era un ejercicio de fe, sobre todo al comienzo de la fase de implementación.

Otro aspecto completamente transparente era la conexión a la base de datos y la creación del modelo de entidades por medio del asistente cuando se agregaba el modelo de datos al proyecto Web (ADO .NET Entity Data Model). El último asistente permitía generar los metadatos para el acceso a las entidades, desde el proyecto Web, al agregar la clase para el modelo de negocio (Domain Service Class).

Al comienzo del semestre partía con grandes expectativas. Sobre todo cuando recuerdo el tiempo invertido en idear la aplicación a desarrollar y en el posible resultado una vez implementada. A lo largo del curso, las pretensiones a la hora de abarcar las funcionalidades propuestas en el plan de trabajo van descendiendo en proporción inversa a la curva de aprendizaje de esta tecnología.

No he conseguido aprovechar las posibilidades de diseño de Silverlight. Las aplicaciones de ejemplo que se pueden consultar en internet son francamente espectaculares. Tampoco he llegado a usar Microsoft Expression Blend, a pesar de haber llegado a probar su funcionamiento al comienzo del semestre.

Respecto a la documentación y tras navegar por multitud de artículos técnicos, manuales y ejemplos, no he sido capaz de dar con los tutoriales adecuados para desarrollar una aplicación de calidad en .NET + Silverlight. Todos se centraban en el cliente de Silverlight y daban por supuesta una base en la parte del servidor .NET y en la conexión a los datos, con la que yo no contaba.

Ha resultado una experiencia personal un tanto agridulce.

La parte mala es que ha supuesto mucha dedicación en tiempo y esfuerzo para obtener un resultado poco vistoso. Yo creo que pasará bastante tiempo hasta que decida seguir estudiando. Es evidente que el reciclaje es inherente a nuestra profesión, pero no sé cuánto tardaré en olvidar estos momentos tan duros.

La parte buena es que he aprendido bastante, aunque infinitamente mayor es lo que aún me quedaría por ver y aprender sobre .NET. El entorno de programación me ha causado una grata sensación. Visual Studio 2010 es un IDE preciso, eficiente y fiable; combinado con Silverlight permite crear aplicaciones novedosas y espectaculares.

Otro aspecto positivo del semestre ha sido demostrar la utilidad del Proyecto Fin de Carrera como trabajo personal de investigación, adaptación y resolución de situaciones adversas, progresando desde el inicio hasta el final. Ya sea partiendo

de unos conocimientos previos, consiguiendo así un resultado que permita el lucimiento personal, o partiendo de cero, como en mi caso, y obteniendo un resultado básico.

Agradecer el ánimo del consultor desde el primer momento, ya que ha sido difícil compaginar el trabajo diario con la elaboración del proyecto. En varias ocasiones he estado a punto de abandonar el proyecto a lo largo del semestre. Seguro que con algún otro consultor de la carrera no hubiera pasado de la segunda entrega.

9. Bibliografía

- Materiales de asignaturas UOC:

  - Ingeniería del software:
    - Módulo 5 - Análisis orientado a objetos
    - Módulo 6 - Diseño orientado a objetos

    Benet Campderrich Falgueras y Recerca Informàtica, S.L.

  - Ingeniería del software orientada a objetos:
    - Módulo 1 - Análisis y diseño con patrones
    - Módulo 2 - Catálogo de patrones

    Jordi Pradel i Miquel y José Antonio Raya Martos

  - Metodología de gestión de proyectos TIC:
    - Caso Práctico

    Santiago Codolà Vilahur

  - PFC
    - Módulo 1. Gestión y desarrollo de proyectos.
    - Módulo 2. Redacción de textos científico-técnicos.
    - Módulo 3. Presentación de documentos y elaboración de presentaciones.

    Antoni Pérez Navarro, Alfons Bataller Díaz, Roser Beneito Montagut, Nita Sáenz Higueras, Rut Vidal Oltra
    - Introducción a .NET

    Jordi Ceballos Villach

- MSDN - Cursos Online - http://msdn.microsoft.com/es-es/dd443596.aspx Recursos para el desarrollo en Silverlight.

- Installing SQL Server Management Studio Express 2008 –

http://www.asql.biz/Articoli/SQLX08/Art3_1.aspx

- Silverlight Blog From Manas Patnaik – http://manaspatnaik.com Artículos "Step By Step Guide to WCF RIA enabled SL4 application with Entity Framework", "Data Binding in Silverlight with RIA and Entity Framework.", "Validating Textbox on Lost Focus in RIA Service".

- Tutoriales Microsoft Silverlight – http://www.silverlight.net/ Using WCF RIA Services.

- MSDN Blogs > WPF & Silverlight Designer – http://blogs.msdn.com Create Silverlight Master – Detail UI Using Data Sources Window Object DataSource.

- Recursos para el desarrollo en Silverlight – http://msdn.microsoft.com/es-es/silverlight/gg578537

- Elaboración de documentación – Microsoft Word 2007.

- Creación de prototipos de la interfaz – Justinmind Prototyper 4.1.1 (http://www.justinmind.com).

- Elaboración de diagramas – Microsoft Visual Studio 2010 Ultimate.

- Elaboración de la presentación – Wondershare Demo Creator 3.5.1, WinFF 1.3.2.