

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE
TELECOMUNICACIÓN
UNIVERSIDAD POLITÉCNICA DE CARTAGENA



**Proyecto Fin de
Carrera**

“Diseño Y desarrollo de una aplicación en .net para el Control De Empleados Y Gestión De Tareas “



AUTOR: Widad Saib
DIRECTOR: Antonio Javier García Sánchez

Junio/2012



Autor	Widad Saib
E-maildelAutor	Widadsaib@hotmail.com
Director(es)	Antonio Javier García Sánchez
E-maildelDirector	antoniojavier.garcia@upct.es
Codirector(es)	
TítulodelPFC	Diseño Y desarrollo de una aplicación en .net para el Control De Empleados Y Gestión De Tareas “LuppApp“
Descriptor	
<p>Resumen</p> <p>Realizar una aplicación escritorio mediante C#, que permita gestionar las tareas asignadas a los empleados de cualquier empresa. De esta forma, se conseguirá realizar un seguimiento sobre el tiempo destinado a desarrollar cada proyecto, además, la aplicación tendrá la capacidad de controlar las ausencias del personal así como las incidencias que puedan surgir a lo largo del horario laboral.</p> <p>El programa permitirá disponer de información detallada de los tiempos empleados para cada tarea, proyecto y usuario. La representación podrá ser tanto gráfica como de tipo texto. Además, lo más importante de esta será la capacidad de estar instalada en cualquier cliente para que de esta forma los datos sean enviados constantemente a un servidor que recogerá toda la información.</p> <p>El administrador además, dispondrá de una aplicación móvil que le mostrará toda la información referente a sus trabajadores y proyectos.</p> <p>Mediante el Kinect de Microsoft, la aplicación podrá detectar tanto la presencia física del empleado así como su postura, lo cual nos permitirá controlar el tiempo destinado al desarrollo real de las tareas.</p>	
Titulación	Ingeniería de Telecomunicaciones
Intensificación	Sistemas y Redes de Telecomunicación
Departamento	Tecnologías de la Información y las Comunicaciones
Fecha de Presentación	2012

Agradecimientos

A mis padres, Driss et Rachida, y hermanos por confiar en mi y apoyarme en todo lo que me propongo, por saber darme cariño en la distancia durante todos estos años. Por dar me toda la libertad de actuar y elegir... *“Merci Cher Père et Mère”*.

A mi familia española”Paqui, Pepe, la abuela...” gracias por todo el apoyo y la ayuda incondicional, gracias por hacerme un hueco en vuestra familia sin pedir nada en cambio.

A mi gran Amigo Miguel por toda la ayuda y las ideas tan brillantes, sin ti este trabajo no habría sido posible.

A mi director del proyecto Antonio Javier García Sánchez, por la libertad que me ha brindado al planificar y desarrollar el proyecto.

Como no a Domingo, gracias por hacer me la vida tan agradable y sencilla ¡Eres mi vida!

Índice

1. - Introducción	7
1.1.- Motivación	7
1.2.- Descripción del sistema a desarrollar	8
1.3.- Objetivos	9
2. Análisis de requerimientos del sistema	11
2.1 Requerimientos funcionales	11
2.2.- Requerimientos no funcionales	15
3. Especificación.....	16
3.1.- Introducción	16
3.2.- Modelo de casos de uso.....	16
3.2.1.- Identificar los actores	16
3.2.2.-Especificación de casos de uso.....	16
3.3.- Diagrama de secuencia.....	21
3.4.- Diagrama de flujo.....	25
4. Diseño del sistema	28
4.1.- Arquitectura del sistema.....	28
4.1.1.- Servidor Web	29
4.1.2.- El servidor de BBDD	31
4.1.3.- Cliente de Escritorio.....	31
4.1.4.- Cliente móvil	31
4.2.- Diseño interfaz	32
4.3.- Diseño BBDD	37
4.3.1.- Introducción	37
4.3.2.- Descripción del modelo.....	37
4.3.3.- Relación entre tablas	43
4.3.4.- Procedimientos almacenados	45
5. Implentación	48
5.1.- Introducción	48
5.2.- Herramientas y tecnología utilizada.....	48
5.2.1.- Microsoft Visual Studio .NET	48
5.2.2.- C#	51
5.2.3.- XML	54
5.2.4.- Internet Information Services (IIS)	54
5.2.5.- Servicios web	57
5.2.6.- Microsoft SQL Server	61
5.2.7.- Kinect	62
5.3.-Arquitectura de software	65
5.4.- Desarrollo Kinect	69
6.-Aplicación Móvil	73
6.1.-Introducción	73
6.2.-Titanium Appcelerator: JavaScript	73
6.3. -Servicios Web de la aplicación móvil	75
6.4.-Modelo Caso de uso	76
6.5.-Diseño e implementación.....	76
8. Conclusiones.....	82
8.1. - Conclusiones	82
8.2.-Lineas futuras	83
Bibliografía.....	85

<i>Anexo I: Manual usuario aplicación escritorio.....</i>	<i>86</i>
<i>Anexo II: Aplicación móvil.....</i>	<i>103</i>

1. - Introducción

1.1.- Motivación

Después de observar desde mi propia experiencia profesional la ineficiencia en la gestión de los tiempos y tareas desarrolladas en distintas empresas, he llegado a la conclusión de que es difícil hacer un seguimiento y control por parte de la dirección de los pymes de algunos aspectos esenciales para el buen desarrollo de la actividad empresarial. Dichos aspectos influirán sustancialmente en la toma de decisiones y por consiguiente en el resultado final de los proyectos y funcionamiento general de la empresa.

En una época de difícil crisis económica la búsqueda de la eficiencia está siendo uno de los valores más perseguidos por los directivos y sin duda, la falta de información en la toma de decisiones es un lastre difícil de gestionar que penaliza dicha eficiencia en las empresas.

El control directo, en tiempo real y con informes precisos también es una herramienta muy valorada por los gestores de personal, pero muy difícil de llevar a cabo, al menos tal y como esta concebida tradicionalmente.

Por otro lado, tanto empresas como particulares, somos conscientes de que estamos viviendo en plena revolución tecnológica. El modelo de los Smartphones y tablets ha entrado con fuerza en el mundo desarrollado y numerosas empresas dedican sus mejores esfuerzos para convertirlos en auténticas herramientas de trabajo.

Es por todo esto que vi clara la necesidad de bucear en este mundo para vincular lo más eficientemente posible las necesidades o carencias de la gestión y control tradicionales con el mundo de la tecnología, generando así una herramienta potente, cómoda y fácil de usar con la que poder tomar decisiones y hacer tareas de control de una forma más sencilla y eficiente, lo cual siempre va aparejado a una mejora en los costes de la empresa y por tanto en el beneficio de la misma.

No solo se trata de crear un producto útil, que nos podemos meter dentro del bolsillo de cualquier gestor mediante la aplicación móvil. Esto, además de la comodidad que supone, fomenta el trabajo desde cualquier punto y a cualquier hora y libertad de movimientos, haciendo a dichos gestores tan eficientes como si estuvieran en el lugar físico donde se encuentran habitualmente sus trabajadores.

La intención del estudio es generar eficiencia y que ésta redunde en el beneficio final de la empresa, es mi motivación y objetivo para este proyecto.

1.2.- Descripción del sistema a desarrollar

El proyecto se divide principalmente en dos módulos de trabajo:

Por un lado disponemos de una aplicación escritorio, más completa y funcional desde la cual los trabajadores pueden acceder a sus tareas diarias siendo el administrador el principal actor encargado del buen funcionamiento del programa. De esta forma, se conseguirá realizar un seguimiento sobre el tiempo destinado a desarrollar cada proyecto, además, la aplicación tendrá la capacidad de controlar las ausencias del personal así como las incidencias que puedan surgir a lo largo del horario laboral.

Como complemento de la mencionada aplicación de escritorio, añadimos mediante el gadget Kinect de Microsoft una solución a la aplicación para poder detectar tanto la presencia física del empleado así como su postura, lo cual nos permitirá controlar el tiempo destinado al desarrollo real de las tareas en tiempo real.

El programa permitirá disponer de información detallada de los tiempos empleados para cada tarea, proyecto y usuario mediante informes que hemos intentado que sean tanto gráficos como de tipo texto.

Por otro lado, el administrador además, dispondrá de una aplicación móvil que le mostrará toda la información referente a sus trabajadores y proyectos de forma más compacta y sencilla.

Resumiendo

Aplicación de escritorio

Mediante esta aplicación los usuarios con perfil de administrador podrán acceder a toda la información para la gestión tanto de sus empleados como las tareas que tiene cualquier proyecto, modificarlas según sus necesidades. Más concretamente podrán realizar altas, bajas y modificaciones de los empleados así como sobre las tareas. El administrador a su vez puede obtener cualquier información deseada mediante los informes creados. Estos informes podrán darnos información precisa de la productividad de cada empleado, las incidencias que tiene, las tareas que ha realizado a lo largo de la fecha que seleccionamos, así como el tiempo trabajado en cada tarea.

Aplicación para dispositivos móviles

Esta es la aplicación que maneja el administrador para consultar de manera rápida los datos de la aplicación tanto de usuario como el tiempo trabajado en las tareas asignadas.

Esta aplicación permite a los gestores tener una herramienta de control en muchos aspectos de su negocio. Pretendemos acercar el control diario al administrador, pudiendo observar de cerca los detalles de trabajo de una manera cómoda. Es por esto que hemos llamado *LuppApp* a nuestra aplicación, y es así como nos vamos a referir a ella de aquí en adelante.

1.3.- Objetivos

El proyecto puede tener varios objetivos entre ellos podemos citar:

- Mejorar la gestión de tareas de manera eficiente y real.

- Controlar la presencia de empleados, así como los tiempos invertidos en cada desarrollo.
- Aumentar la información referente a la empresa, tiempos, incidencias, tareas, usuarios mediante informes que son reportados al administrador/gerente.
- Fomentar el Teletrabajo de manera eficaz y controlada.
- Aumentar el rendimiento de los trabajadores y los desarrollos en grupo.

Como objetivos secundarios podemos hablar de una introducción a un aprendizaje personal en las diferentes posibilidades que ofrece desarrollar aplicaciones móviles.

2. Análisis de requerimientos del sistema

2.1 Requerimientos funcionales

Los requerimientos funcionales de nuestro proyecto han de satisfacer las necesidades de cualquier usuario que utiliza la aplicación. Dentro de los requerimientos funcionales hemos listado los puramente funcionales y los requerimientos de los datos, sin hacer ningún tipo de distinción. El grupo de requerimientos puramente funcionales hace que cada uno de los requerimientos vaya inherentemente ligado a un caso de uso.

Configuración del sistema: Esta funcionalidad debe dejar al usuario cambiar la configuración de la cadena de conexión de la aplicación, ya que si instalamos el programa en otro servidor, nos permitiría cambiar el nombre o la dirección de este.

Salir: Solo podemos salir de la aplicación mediante salir, en caso contrario el programa seguirá ejecutándose.

Gestión de empleados: esta funcionalidad debe permitir al administrador dar de alta baja o editar el perfil de los empleados. Así mismo el usuario también tendrá la posibilidad de consultar o modificar los datos sobre estos. El objetivo de tener una base de datos con este tipo de información, es la de poder utilizar esos datos de manera corporativa, por ejemplo para asignar tareas a los diferentes usuarios.

Los datos que se deberán guardar de los empleados son:

- *Personales: Nombre, apellido,*
- *Domicilio: Dirección.*
- *De contacto: Teléfono, mail*

Gestión de Proyectos: En este apartado, se contempla la opción de guardar en el sistema los datos relacionados con los proyectos y sobre los cuales se realiza las tareas del proyecto. En esta funcionalidad del sistema será posible dar de alta y de baja a proyectos, así como consultar o modificar los datos referentes a los proyectos guardados en la BBDD.

Sobre los proyectos se deberán conservar los siguientes datos:

- *Descripción del proyecto.*
- *Estado del proyecto.*
- *Horas estimadas.*
- *Horas trabajadas.*

Gestión de Tareas: Esta funcionalidad permitirá dar de alta, baja, consultar o editar las tareas. De esta manera, en cualquier momento se podrá obtener información acerca de las tareas actuales. Además, en este menú encontraremos una de las más importantes funcionalidades del software, seleccionar tarea, ya que a partir de esta, el sistema empieza a contabilizar el tiempo trabajado.

En este mismo apartado, está previsto que el administrador pueda asignar o quitar tareas a los distintos empleados.

Búsqueda: El usuario de la aplicación puede necesitar en cualquier momento acceder a consultar datos. En muchos de estos casos los usuarios no poseen mucha información completa sobre el elemento que desean consultar, y puede que en ocasiones tan solo conozcan una característica del elemento que se quiere buscar. Por este motivo, se considera importante que la aplicación tenga un buen motor de búsqueda para cada menú (empleados, proyectos, tareas), donde se permita realizar filtros por diferentes campos, dejando la decisión al usuario de los campos utilizados para buscar la entrada de la base de datos que deseamos.

Los diferentes campos que se podrán usar para buscar serán los siguientes:

- *Empleados: Apellido, nick, teléfono, mail.*
- *Proyectos: Descripción del proyecto, estado.*
- *Tarea: Descripción del proyecto, descripción de tarea, estado.*

Gestión de Incidencias: Esta Funcionalidad permitirá al usuario ingresar las incidencias que le vayan pasando a lo largo del día.

Generación de Informes: Esta Funcionalidad permitirá al administrador la generación completa de cualquier report, de esta manera podrá consultar de manera rápida, los informes que se pueden generar y que son:

- *El tiempo dedicado a cada tarea.*
- *Incidencias por empleado.*
- *Horas trabajadas al día para cada empleado.*
- *Las horas trabajadas por empleado.*
- *Tarea cerrada entre fechas.*
- *Tarea abierta entre fechas.*
- *Proyectos abiertos entre fechas.*
- *Proyectos cerrados entre fechas*

Sistema automático (Demonio): Creación de un sistema automático que cada cierto tiempo (cada minuto) realizará algunas acciones como actualizar la barra de estado, enviar mails o mostrar globos de alertas.

Icono: Creación de un icono que permanecerá de forma permanente en la barra de estado. Además de mostrar la aplicación o hacerla invisible, mediante globos emergentes indicará el estado y la progresión de las distintas tareas.

Ayuda: Esta funcionalidad permitirá al usuario consultar la ayuda del sistema en cualquier momento durante la utilización de la aplicación.

En el menú de ayuda encontraremos todos los pasos a seguir para conseguir realizar cualquier paso en el software.

Como resumen gráfico de todos los requisitos tenemos las tablas citadas a continuación:

R1. Configuración del sistema: Cambiar cadena de conexión.

R2. Salir: Única salida de la aplicación.

R3. Gestión de empleados: Alta, baja, edición y consulta de los empleados.

R4. Gestión de proyectos: Alta, baja, edición y consulta de los proyectos.

R5. Gestión de tareas: Alta, baja, edición, consulta y selección de las tareas.

R6. Búsqueda: Filtrar para poder buscar empleados, proyectos, tareas.

R7. Gestión de incidencias: Anotar incidencias ocurridas.

R8. Icono: Facilitar entrada/salida a la aplicación

R9. Sistema automático: Generar avisos o mandar mails automáticamente.

R10. Icono: Generar avisos o mandar mails automáticamente.

R11. Ayuda: Manual descriptivo de la aplicación.

2.2.- Requerimientos no funcionales

El sistema también debe cumplir las siguientes condiciones o requisitos no funcionales:

- **Seguridad:** el proyecto tiene que garantizar cierta privacidad de datos a los usuarios y proteger el acceso para proteger las contraseñas.
- **Entorno amigable:** el sistema tiene que tener un entorno fácil de usar para que cualquier usuario pueda utilizar los servicios sin dificultades.
- **Rápido:** el sistema tiene que ser lo más eficiente posible en cuanto a velocidad.
- **Escalable:** el sistema no debería colapsarse al ampliar el número de usuarios o recursos.
- **Ampliable:** la aplicación tiene que dar facilidades para poder ampliar el número de secciones o servicios o incluso un cambio de aspecto de la interfaz gráfica.
- **Concurrencia:** el sistema puede ser utilizado por más de una persona de manera concurrente.
- **Control de errores:** el sistema debe disponer de un sistema de excepciones para intentar identificar el mayor número de errores posibles y con rapidez.

3. Especificación

3.1.- Introducción

En esta parte del proyecto se explicará el trabajo realizado sobre la especificación utilizando UML (*Unified Modeling Language*) que es el lenguaje más utilizado para la especificación y el diseño de la programación orientada a objetos. En este capítulo se desarrollara los diferentes tipos del modelo de casos de uso.

3.2.- Modelo de casos de uso

En el modelo de Casos de uso se definen los casos de uso del sistema y la relación existente entre ellos. Un caso de uso es un diagrama que describe una secuencia de acontecimiento que realiza un actor (agente externo al sistema) que usa el sistema para llevar a cabo un proceso que tiene algún valor para él. Este modelo principalmente se compone de:

3.2.1.- Identificar los actores

Un actor es una entidad externa al sistema que participa en la historia de los casos de uso. Puede ser una persona, un conjunto de personas, un sistema hardware, un sistema software o un reloj.

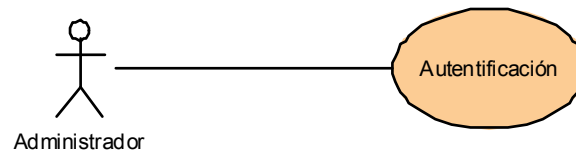
3.2.2.-Especificación de casos de uso

Es una breve descripción de las acciones del caso de uso.

Administrador

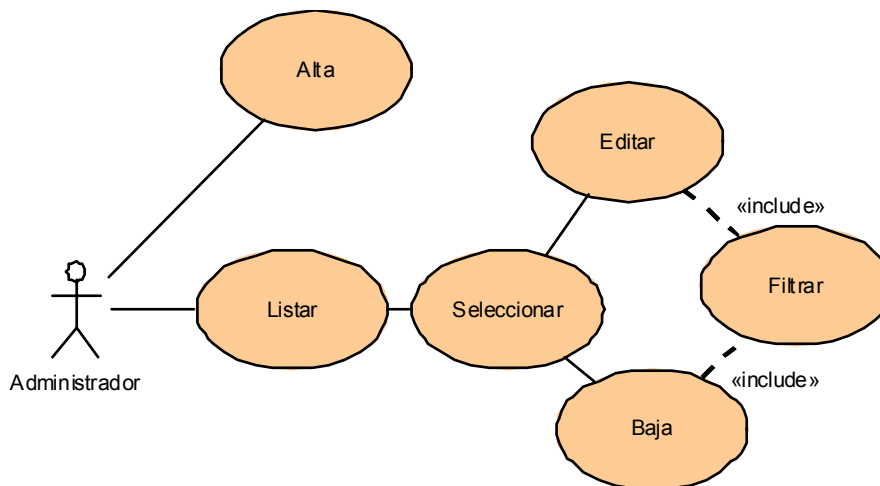
- Autenticación

El administrador introduce su nombre de usuario y contraseña. El sistema carga diferentes opciones de menú según el perfil de usuario en este caso cargará la de admin.



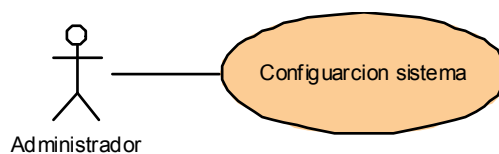
- Mantenimientos

Vamos a hacer un diagrama de uso general en el caso de mantenimientos ya que su funcionamiento es igual para empleados, proyectos como tareas, se trata de dar alta, baja o editar.

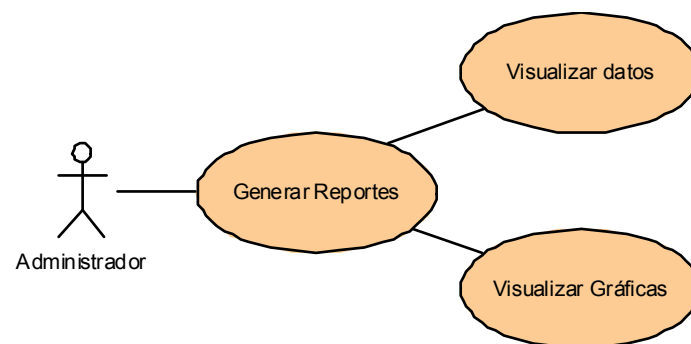


- Parámetros

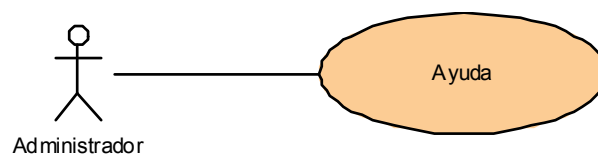
El administrador quiere realizar unos cambios en el ajuste del sistema y puede modificar los parámetros del sistema.



- Report



- Ayuda

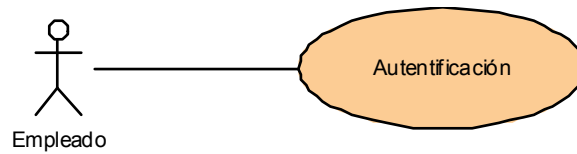


Empleado

A continuación vamos a describir las acciones correspondientes al empleado.

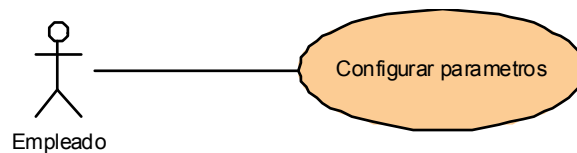
- Autenticación de usuarios

El empleado introduce su nombre de usuario y contraseña. El sistema carga diferentes opciones de menú según el perfil de usuario, en este caso cargara el menú de empleado.



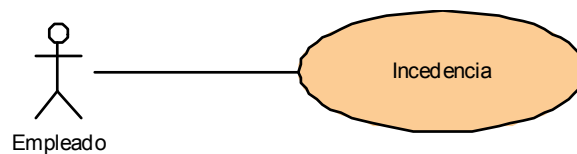
- Parámetros

El empleado quiere realizar unos cambios en el ajuste del sistema y por eso decide modificar los parámetros del sistema.



- Incidencia

El empleado introduce las incidencias que le pueden surgir.

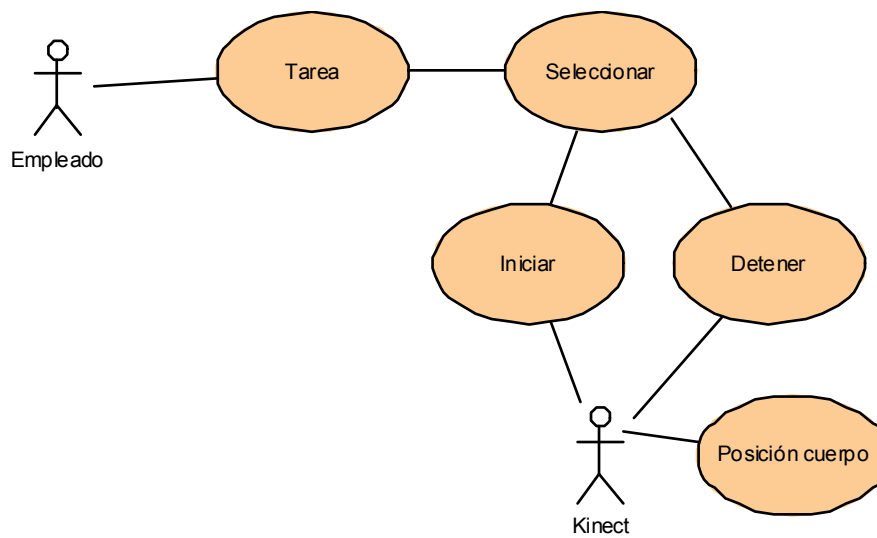


- Tarea

El empleado selecciona una tarea y empieza a trabajar en ella, a partir de aquí puede cambiarla, hacer una pausa o continuar con ella.

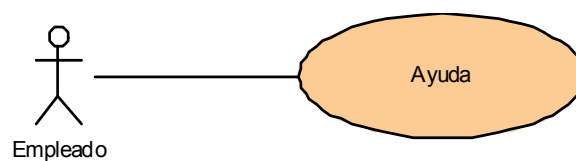
El sistema le permite dos maneras de interactuar; una automática mediante el Kinect y otra manual.

La configuración mediante Kinect permite a LupApp determinar la posición del cuerpo del empleado, de esta manera, detiene la aplicación si esta de pie, y la reanuda si se sienta.



- **Ayuda**

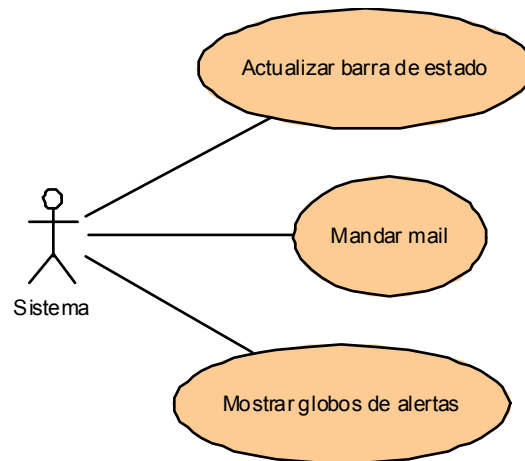
Consultar la ayuda de la aplicación.



Sistema

Es el corazón de la aplicación, es el responsable de que se lleven acabo la totalidad de las tareas como:

Comprobar que las tareas han sido excedidas, y si es el caso manda un mail al admin avisándole que se ha excedido el tiempo estimado para esta tarea. Actualizar la barra de estado con la información adecuada en cada momento. Mostrar las alertas que vayan saltando.



3.3.- Diagrama de secuencia

Un diagrama de secuencia es de los diagramas más útiles para el modelo de interacción entre los objetos de un sistema de información. Mientras que el diagrama de un caso de uso permite crear una visión del escenario, el diagrama de secuencia contiene características de implementación del escenario, puede incluir objetos y clases que se necesitan para implementar el escenario, así como los mensajes entre objetos.

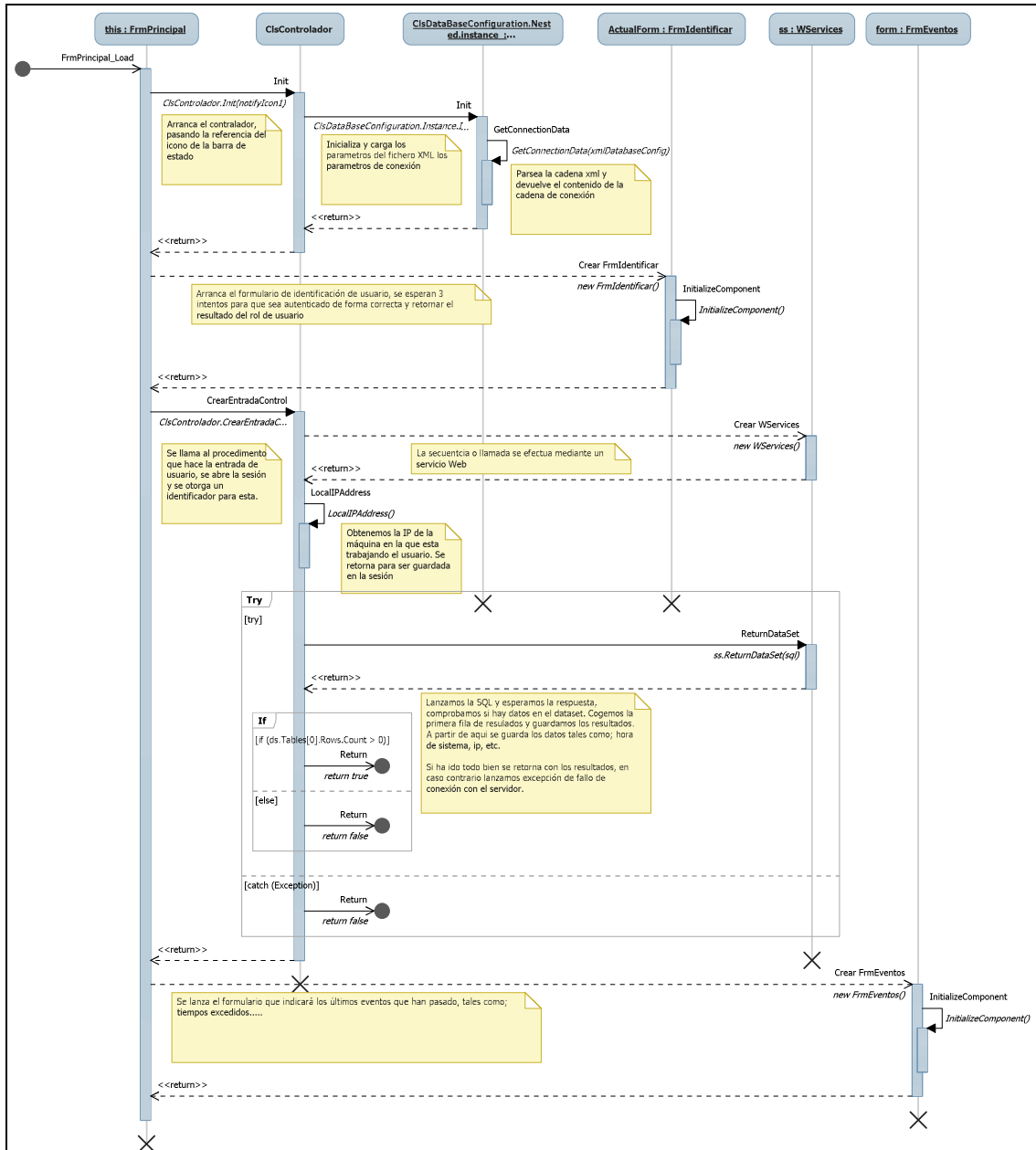
Por lo tanto en un diagrama de secuencia se muestra los objetos que actúan en el escenario con líneas discontinuas verticales, llamadas líneas de vida, mientras que los mensajes pasados entre los objetos como vectores horizontales. Los mensajes se

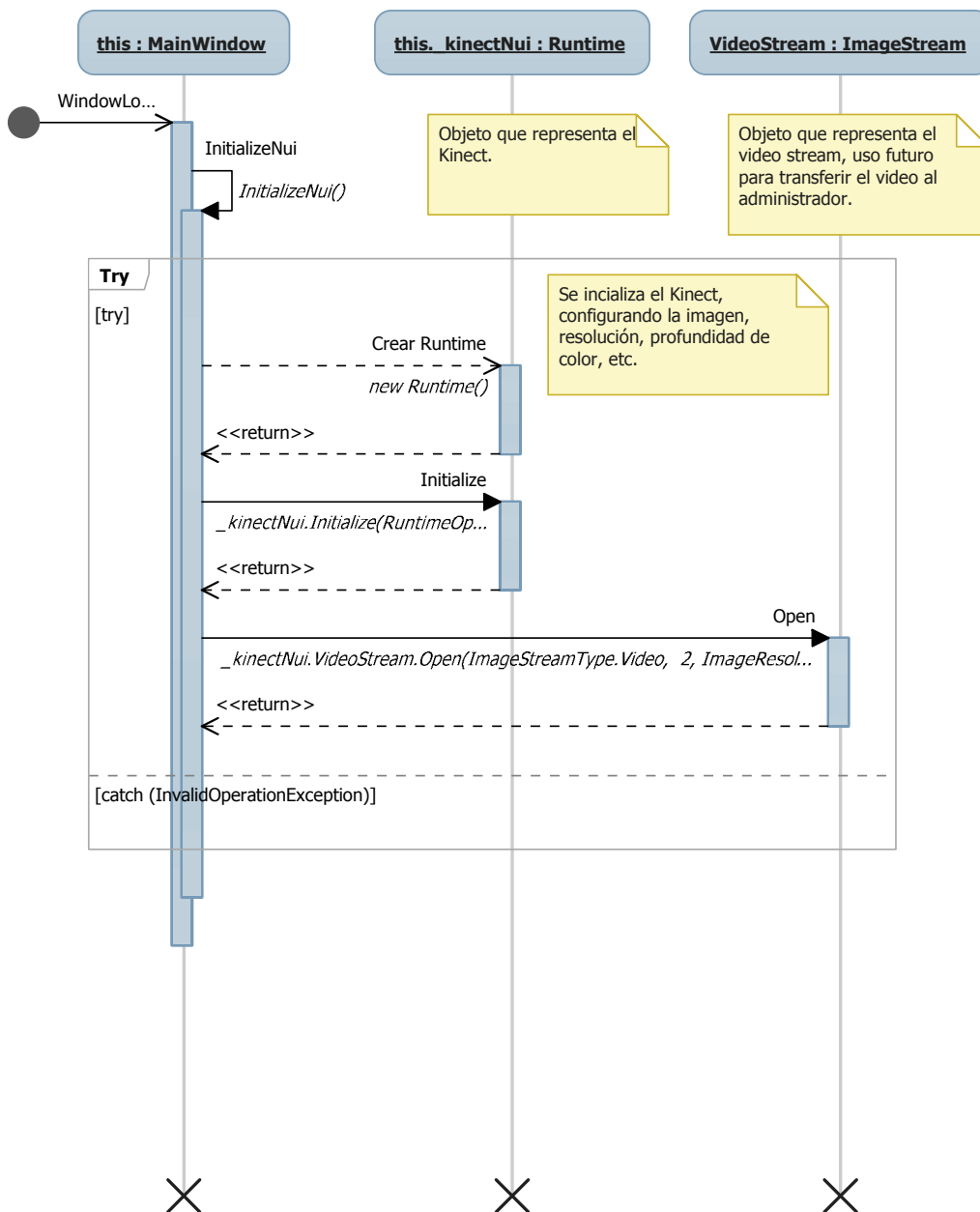
muestran cronológicamente desde la parte superior del diagrama a la parte inferior, la distribución horizontal de los objetos no tiene orden.

Mediante la secuencia de pasos que define se descubren los objetos que intervienen en el escenario así como los eventos dirigidos hacia el sistema y desde el sistema a los actores.

Se han creado los diagramas de secuencia automáticamente con una herramienta de visual estudio que permite crearlos a partir del código. Pueden resultar más complejos, al llevar muchos detalles, es por esto que hemos incluido comentarios para una mejor comprensión.

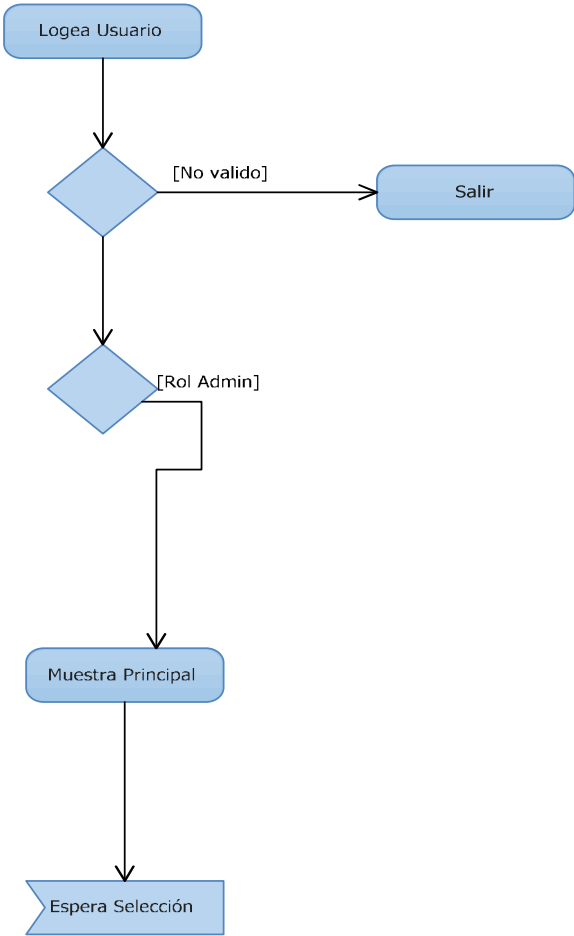
A continuación se muestran varios escenarios del proyecto:

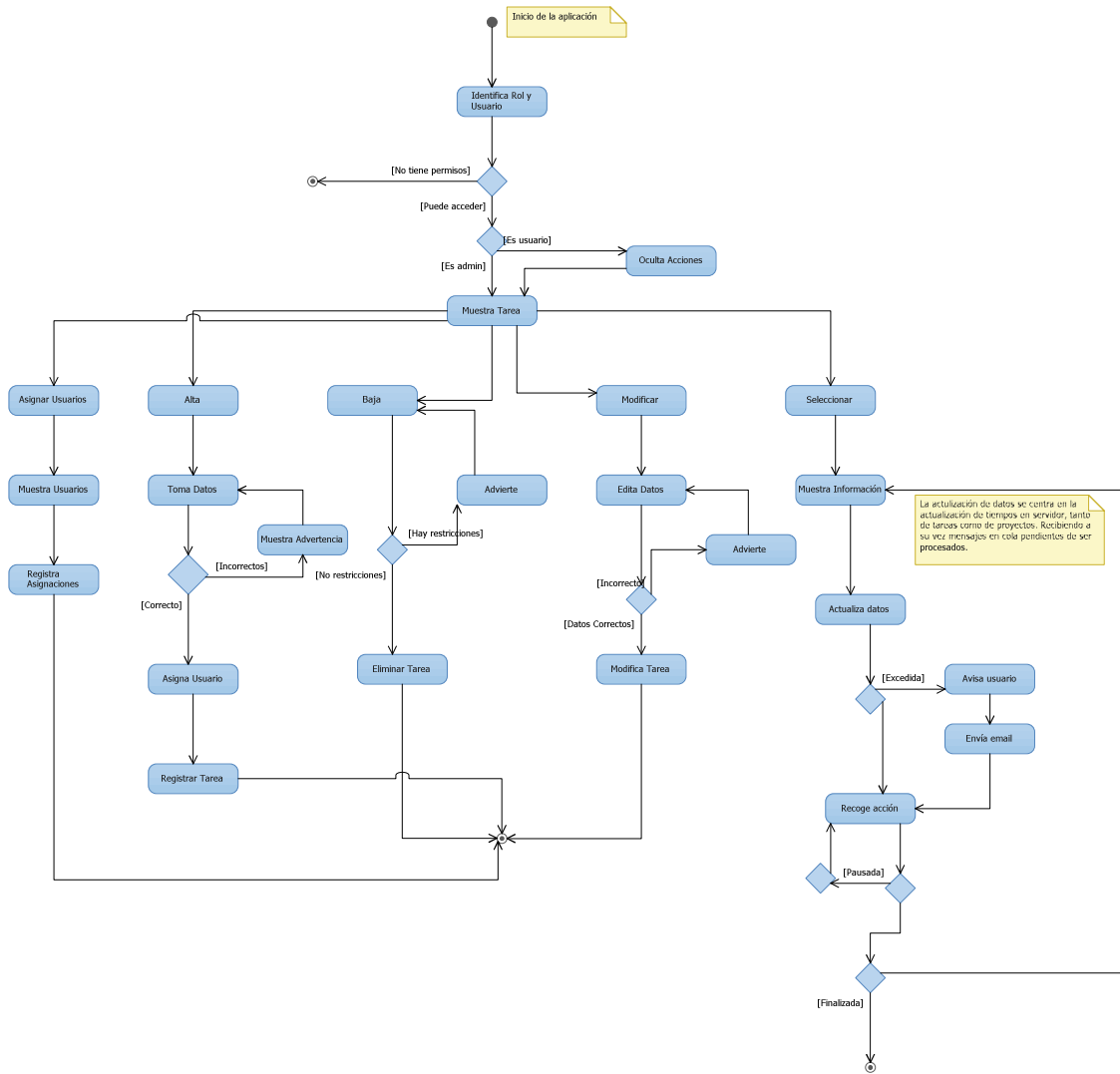




3.4.- Diagrama de flujo

Hemos incluido solo dos diagramas de flujo para el principio de la aplicación y para la función más importante de la aplicación que es seleccionar el menú de tareas.





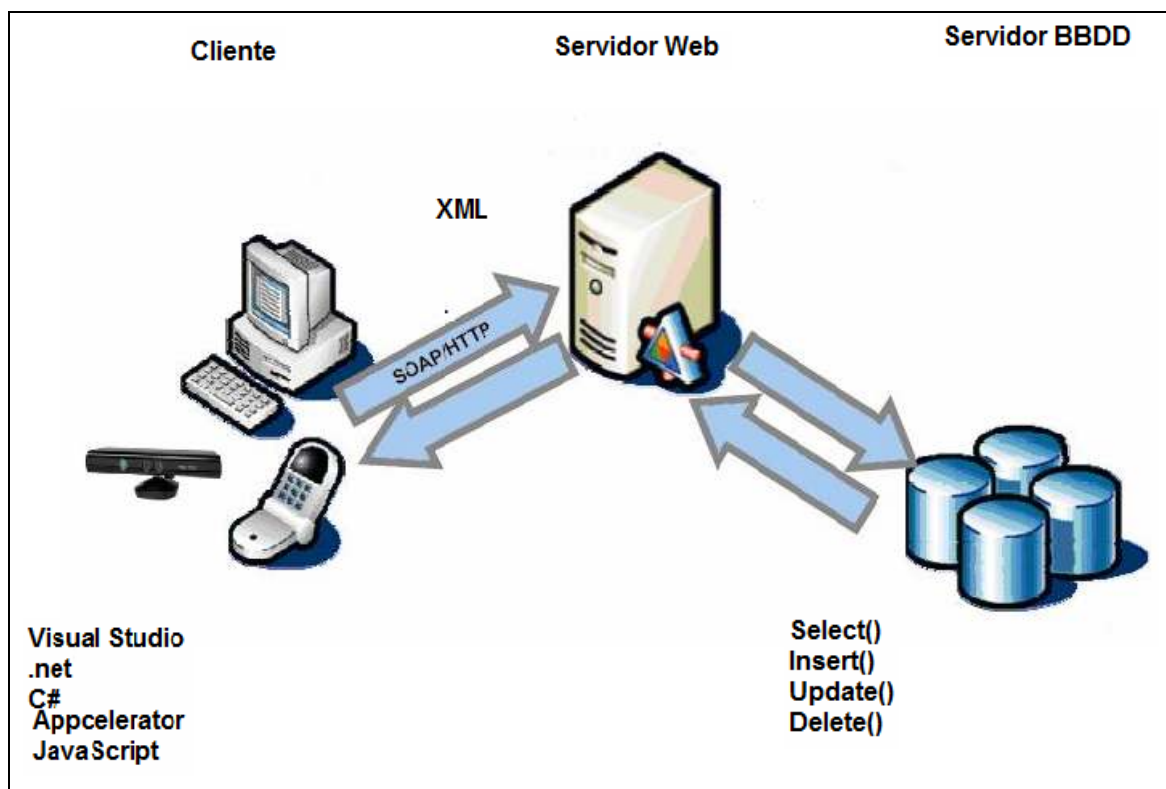
4. Diseño del sistema

4.1.- Arquitectura del sistema

La solución propuesta consiste en instalar la aplicación en puestos de escritorio o dispositivos móviles como se muestra a continuación. En el siguiente gráfico queda reflejada la arquitectura general del sistema y las tecnologías usadas en cada parte.

La arquitectura del sistema está formada por los siguientes elementos:

- Servidor Web.
- Servidor BBDD.
- Cliente de escritorio.
- Cliente Móvil.



4.1.1.- Servidor Web

Contenedor de los distintos Servicios Web y otras funcionalidades de que se compone la aplicación. El servidor de aplicaciones (Internet Information Server v.6) dispone de la lógica del acceso a los Servicios Web.

En el presente proyecto se creará un sitio Web específico con un directorio particular llamado servicios.asmx.

```
<%@ WebService Language="C#" Class="WServices" %>

using System;
using System.Data;
using System.Data.SqlClient;
using System.Web.Services;
using System.Collections.Generic;

[WebService(
    Namespace="ServicesWeb",
    Description="Servicios Web para el proyecto")]

public class WServices {
    private SqlDataAdapter da;
    private SqlCommand command;

    [WebMethod(Description="Devuelve el recordset.")]
    public DataSet ReturnDataSet(string sql)
    {
        da = new SqlDataAdapter(sql,
            "data source=localhost\\SQLEXPRESS; initial
catalog=db_pfc;User ID=sa;Password=QuieroMesaDB");

        DataSet ds = new DataSet();
        try
        {
            da.Fill(ds);
        }
        catch(Exception ex)
        {
            throw ex;
        }
        return ds;
    }
}
```

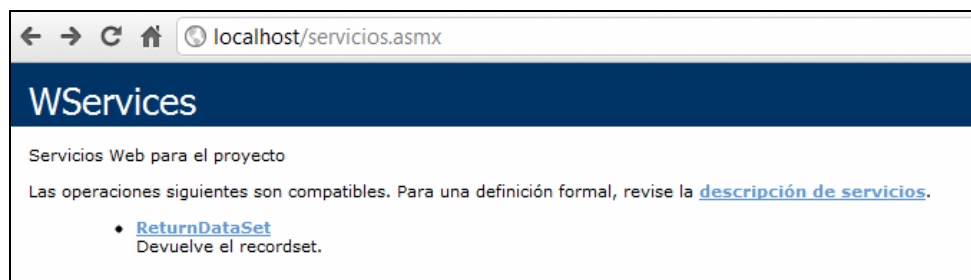
Para que *.NET Framework* sepa que este archivo es un servicio Web, tenemos que indicárselo mediante la directiva de ASP.NET @ WebService, en esa directiva se indicará el lenguaje en el que estará el código, así como el nombre de la clase.

Incluimos la directiva *System.Web.Services*, la cual contiene las clases que nos permite crear servicios Web y clientes de servicios Web.

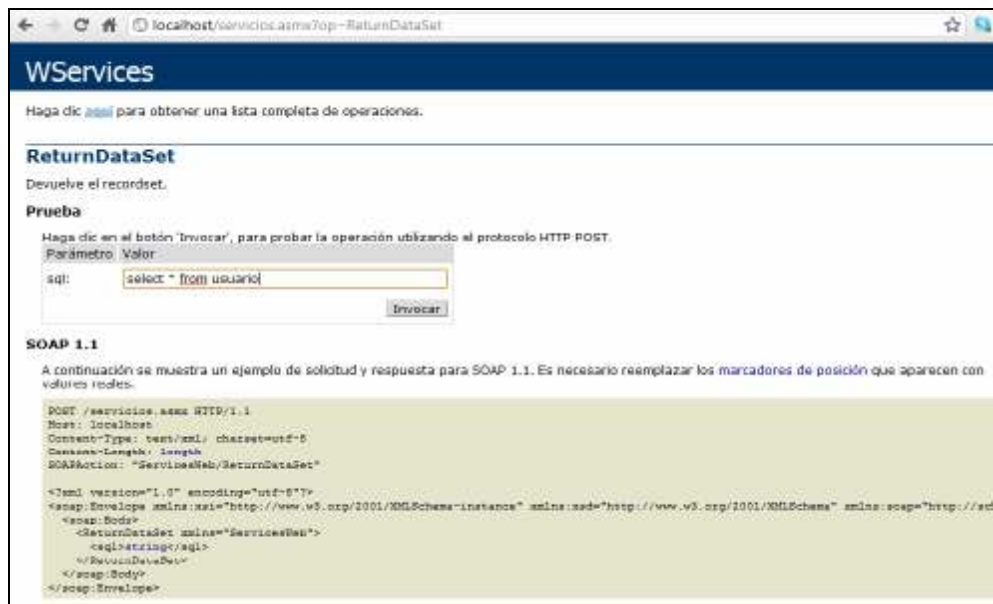
Se incluirá el código que contendrá esta clase, en especial la declaración de las funciones que nuestra clase expondrá desde el servicio Web, para ello debemos aplicar a

el atributo *WebMethod* a cada uno de los métodos que queramos que el servicio Web exponga. Si no indicamos este atributo, el método no será visible (o accesible) desde el servicio Web. Añadimos una pequeña descripción, dependiendo de nuestro método

Para poder usar este servicio Web desde el explorador simplemente tendremos que indicar la dirección Web en la que tengamos alojado el servicio Web. En nuestro caso como estamos utilizando el servidor local (localhost) tendremos que escribir: <http://localhost/servicios.asmx>.



EL sitio Web contendrá el Servicio Web con el que se comunica la aplicación cliente.



4.1.2.- El servidor de BBDD

Permite el acceso a la bases de datos que utiliza el Sistema cuyo nombre es “Pfc” contiene todos las tablas que maneja el sistema.

4.1.3.- Cliente de Escritorio

Es la aplicación cliente instalada en un ordenador, se comunica con el Servidor Web para realizar todas las operaciones que sean necesarias para proporcionar los servicios. Realiza el servicio de autenticación de los empleados mediante un proceso de autenticación utilizando para ello los servicios Web que le permiten acceder a nuestra base de datos de situada en el Servidor de BBDD. Una vez se ha realizado esta autenticación y el usuario es autenticado de forma satisfactoria, se le otorgan unos permisos dependiendo del perfil que tenga.

4.1.4.- Cliente móvil

Aplicación cliente instalada en un dispositivo móvil, precisamente en el del administrador o el gerente de la empresa y, que se comunica con el Servidor Web para realizar todas las operaciones que sean necesarias para proporcionar los servicios.

Los servicios Web utilizados son GetProyectos, GetUsuarios, GetTareas.

WServices

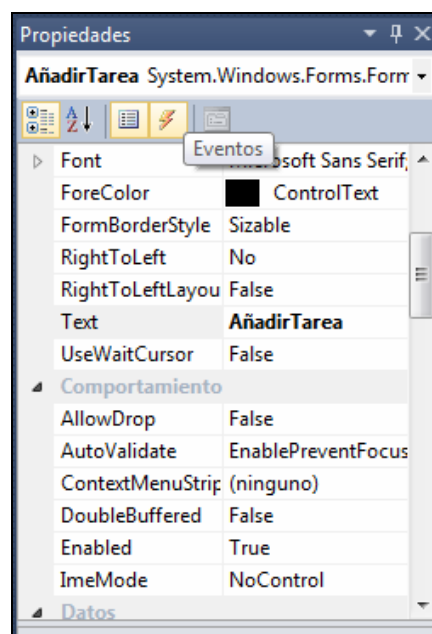
Servicios Web para el proyecto

Las operaciones siguientes son compatibles. Para una definición formal, revise la [descripción de servicios](#).

- [GetConectados](#)
Devuelve el recordset con los Usuarios Conectados
- [GetProyectos](#)
Devuelve el recordset con los proyectos.
- [GetUsuarios](#)
Devuelve el recordset con los usuarios.
- [GetTareas](#)
Devuelve el recordset con las tareas.

se convierte en un objeto de interfaz de usuario programable en nuestra aplicación. Estos objetos pueden ser visibles o no; depende del rol del usuario que entra al sistema, y funcionan como los objetos estándares de cualquier aplicación basada en Windows.

La ventana de propiedades (ventana Propiedades) muestra una lista de las propiedades que pueden configurarse para el formulario o el control seleccionado y que podemos modificar mientras creamos o editamos la interfaz. Una propiedad describe una característica de un objeto, como el tamaño, título o color.



La ventana Propiedades nos permite visualizar las propiedades de un formulario o control en una vista ordenada por categorías. De esta manera, se puede modificar las propiedades seleccionando los objetos y cambiando su configuración en la ventana propiedades.

A continuación, se van a comentar brevemente los controles más utilizados con algunos ejemplos de nuestra aplicación LuppApp:

Button

Presenta un botón estándar en el que el usuario puede hacer clic para realizar acciones.

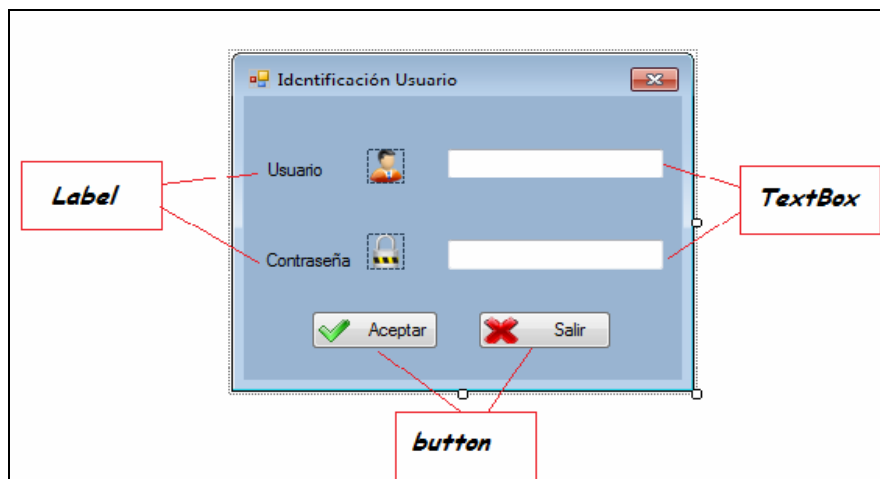
Label

Muestra un texto informativo al usuario. Su utilización puede ser conjunta con otro, control. Por ejemplo, podremos colocar un Label cerca de una TextBox que informe acerca del contenido de éste control.

Se trata de un control estático por lo que el usuario no podrá interactuar con él ya que simplemente se limita a mostrar un determinado texto.

TextBox

Muestra una caja capaz de almacenar un texto introducido por el usuario y cuyo contenido puede cambiar a lo largo del programa.



ListView

Este control es capaz de almacenar una lista de valores, y a la vez permite al usuario seleccionar uno o varios de estos valores. Las propiedades más importantes de este control son:

- Items: Contiene la lista de valores que almacena el ListView. Se trata de un objeto Collection, el cual proporciona una serie de métodos para poder trabajar sobre ese conjunto de valores, en nuestro caso hemos hecho uso de :
 - Add(item) : Permite añadir un nuevo elemento a la lista.

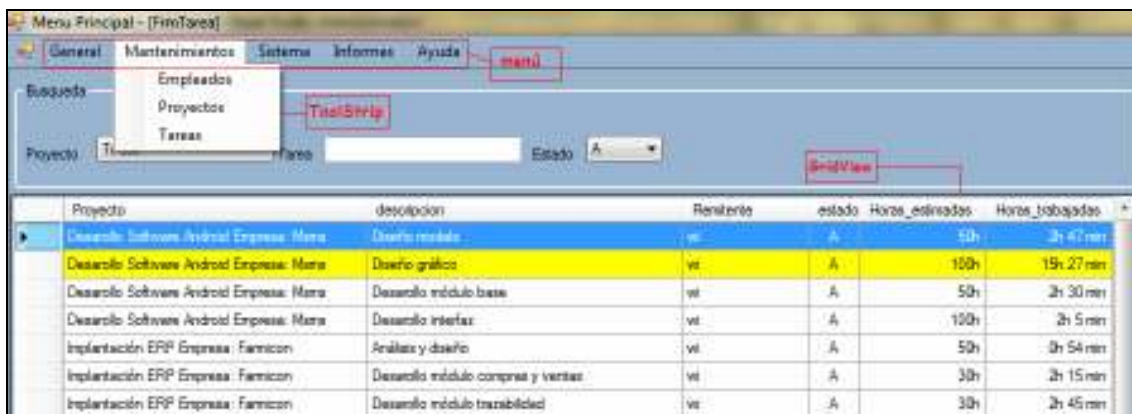
El control GridView proporciona una tabla que se puede personalizar para mostrar los datos. GridView habilita la personalización de celdas, filas, columnas y bordes.

Menú

El menú es un control muy utilizado en todo tipo de aplicaciones.

ToolStrip

Crea barras de herramientas y menús personalizados en la aplicación.

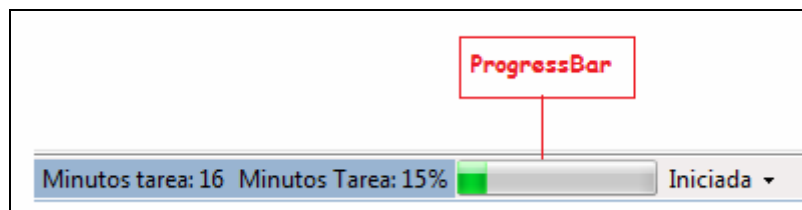


Timer

Provoca un evento a intervalos regulares.

ProgressBar

Muestra el progreso actual de una operación al usuario



4.3.- Diseño BBDD

4.3.1.- Introducción

Para realizar la base de datos hemos utilizado el modelo de entidad-relación.

Este es el modelo que más se utiliza para el diseño de base de datos. El modelo está formado por las siguientes características:

Entidad: Es cualquier objeto sobre el que se almacena información. Hay dos tipos de entidades. Está la entidad débil que significa que su existencia depende de otra, y la entidad fuerte que es una entidad que no es débil.

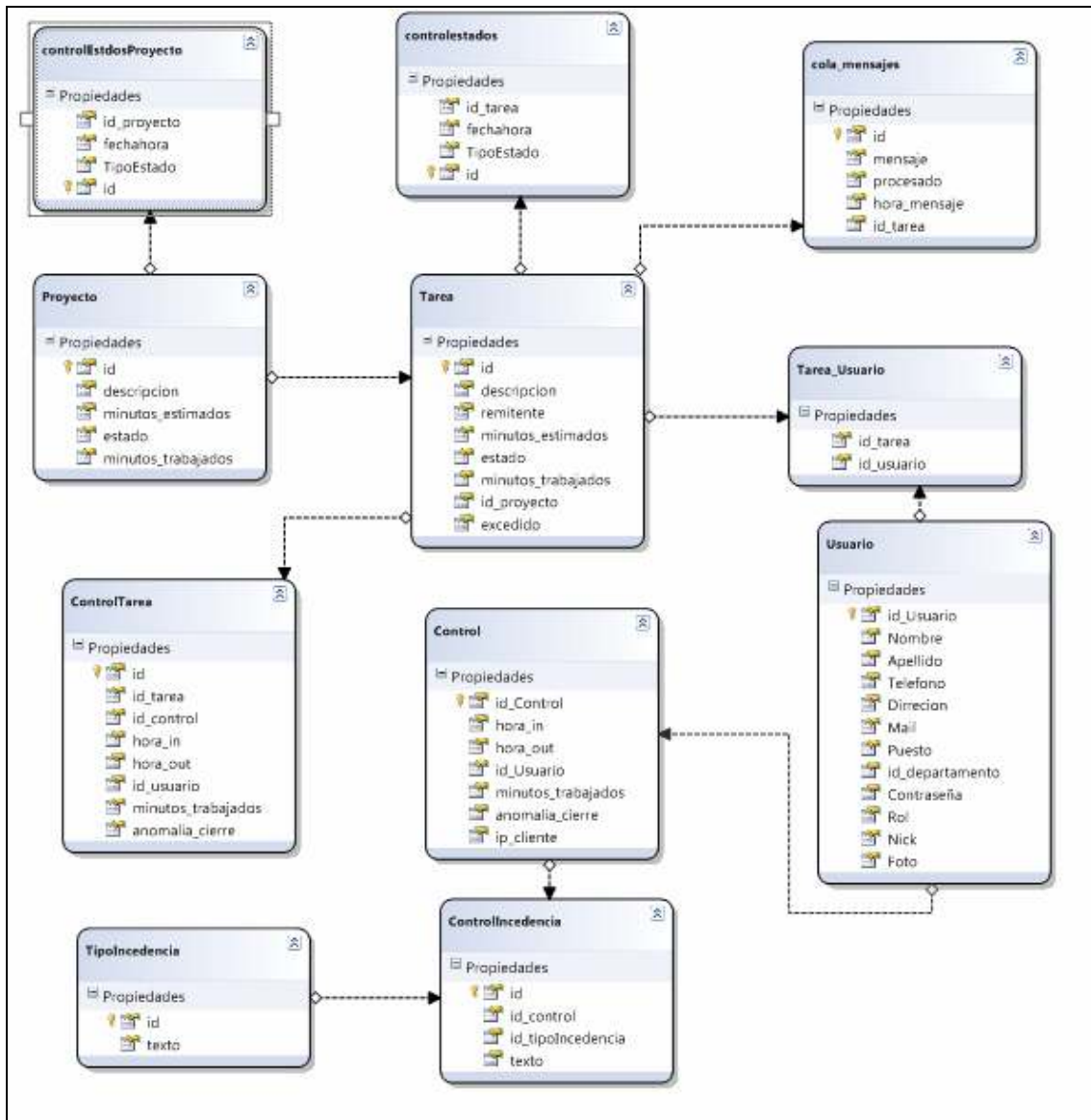
Relación: Es una asociación entre dos entidades. Cada relación tiene un nombre que define la función. Existen varios tipos de relación pero lo más común es el de asociación. La asociación puede ser de 1-1, que significa una entidad está asociada con otra entidad, de 1-muchos, que significa que una entidad está relacionada con muchas de la segunda y de esta forma se construye la de muchos-muchos, 0-muchos.

Atributo: Es una característica de una entidad o sobre una relación. Los atributos definen las propiedades más básicas. Por cada atributo se tiene un conjunto de valores.

Identificador: Es un atributo o varios atributos que determina de forma única la entidad. Para esto el identificador ha de cumplir dos reglas. Una es que no pueden existir dos identificadores iguales, y la otra es que si se elimina un identificador del conjunto la regla anterior deja de cumplirse.

4.3.2.- Descripción del modelo

En la siguiente imagen se muestra la composición de la base de datos. Para mayor legibilidad se muestra primero como están conectadas las tablas y después de forma independiente los atributos de cada tabla.



Ahora describimos los detalles de cada tabla:

Cola mensajes

Esta tabla alojará los mensajes por enviar al administrador, según se están enviando se indicará en el procesado si ha sido enviado o no.

Campo	Tipo / Relación	Descripción
Id	campo primario y clave única, no nulo, de tipo entero	Es clave única.
Mensaje	campo de tipo carácter, tamaño 50 caracteres, no nulo	Contiene el texto que se mostrará.
Procesado	campo entero, no nulo	Indica con 1 que ha sido procesado.
Hora_mensaje	indica la hora del mensaje creado	campo datetime, no nulo
Id_tarea	campo relación , no nulo, uno a varios	Relaciona con la tarea correspondiente.

Control

Tabla que contiene la monitorización de las sesiones, entradas y salidas de cada usuario.

Campo	Tipo / Relación	Descripción
Id_control	Primaria, entero, no nulo	Es la clave única
Hora_in	Datetime, admite nulo	Indica la hora de apertura de la sesión
Hora_out	Datetime, admite nulo	Indica la hora de cierre de la sesión
Id_Usuario	Entero, no nulo, campo relación uno a varios	Indica el usuario que ha abierto la sesión
Minutos_Trabajados	Decimal, null	Contiene el número de minutos transcurridos en la sesión
Anomalia_Cierre	Bit, nulo	Indica si hubo algún tipo de anomalía en la sesión, por estar mal cerrada. Si es true es que lo hubo.
Ip_Cliente	Carácter, nulo	Contiene la Ip del cliente.

ControlEstados

Tabla que contiene los distintos cambios de estado que se suceden en las tareas, ejemplo: abierto, pendiente, cerrado.

Campo	Tipo / Relación	Descripción
Id	Primaria, entero, nulo	Es la clave única
FechaHora	Datetime, nulo	Contiene la hora en la cual se cambió el estado
TipoEstado	Carácter, nulo	Contiene un carácter que identifica el estado al cual ha cambiado la tarea.
Id_Tarea	Entero, nulo, relación uno a varios	Enlaza con el id de la tarea correspondiente

ControlEstadosProyecto

Tabla que contiene los distintos cambios de estado que se suceden en los proyectos.

Campo	Tipo / Relación	Descripción
Id	Primaria, no nulo, entero	Es la clave única
Id_Proyecto	Entero, nulo, relación uno a varios	Relaciona con la Id del Proyecto
FechaHora	Datetime, nulo	Indica la hora en la que se ha producido el cambio de estado del proyecto
TipoEstado	Carácter, nulo	Indica el estado al cual se ha cambiado el proyecto

ControlIncidencia

Muestra las incidencias que se producen, pueden ser de tipo; “llego tarde”, “salgo”, “Fallo en la conexión”.

Campo	Tipo / Relación	Descripción
Id	Primaria, entero, no nulo	Clave Primaria
Id_Control	Entero, no nulo, relación uno a varios	Contiene el id de la tabla Control, la sesión que se ha abierto
Id_TipoIncidencia	Entero, no nulo, relación uno a varios	Relaciona al tipo de incidencia
Texto	Carácter, nulo	En el caso de que sea otros, contiene el mensaje personalizado

ControlTarea

Esta tabla contiene la información de las sesiones que se abren para cada tarea, en ella se especifica y controla el tiempo dedicado.

Campo	Tipo / Relación	Descripción
Id	Primaria, no nulo	Clave Primaria
Id_Tarea	Entero, no nulo, relación uno a varios	Indica el id de la tarea relacionada
Hora_in	Datetime, no nulo	Indica la hora en la que fue abierta la sesión de la tarea
Hora_out	Datetime, nulo	Indica la hora en la cual se cerró la sesión de la tarea
Id_usuario	Entero, nulo	Indica el usuario que al cual pertenece la sesión de la tarea
Id_control	Entero, nulo	Indica el id de la tabla control al cual se ha relacionado
Minutos_Trabajados	Entero, nulo	Contiene el tiempo transcurrido en la sesión de la tarea
Anomalia_Cierre	Bit, nulo	Contiene si hubo alguna anomalía en el cierre de la tarea

Proyecto

Tabla que contiene toda la información relacionada con los proyectos.

Campo	Tipo / Relación	Descripción
Id	Clave Primaria, entero, no nulo	Clave Primaria
Descripcion	Carácter, no nulo	Contiene la descripción del proyecto
Minutos_Estimados	Decimal, no nulo	Indica los minutos que se han estimado para la tarea
Estado	Carácter, no nulo	Indica el estado en el cual se encuentra el proyecto
Minutos_Trabajados	Decimal, no nulo	Contiene el totalizado de minutos trabajados hasta el momento en el proyecto

Tarea

Tabla que contiene la información concerniente a cada una de las tareas.

Campo	Tipo / Relación	Descripción
Id	Clave Primaria, entero, no nulo	Clave Primaria
Descripción	Carácter, no nulo	Describe la tarea
Remitente	Entero, no nulo	El id del usuario que ha creado la tarea
Minutos_Estimados	Entero, no nulo	El total de minutos estimados para la tarea
Estado	Carácter, no nulo	El estado en el cual se encuentra el proyecto
Minutos_Trabajados	Entero, no nulo	El total de minutos acumulado para la tarea hasta el momento actual
Id_Proyecto	Entero, no nulo, relación uno a varios	Relaciona con el proyecto correspondiente
Excedido	Bit, nulo	Indica si la tarea ha sido excedida con un true

Tarea Usuario

Establece los usuarios que podrán trabajar para cada una de las tareas.

Campo	Tipo / Relación	Descripción
Id_Tarea	Entero, no nulo, relación uno a varios	Relaciona con la tarea
Id_Usuario	Entero, no nulo, relación uno a varios	Relaciona con el usuario

TipoIncidencia

Tabla que contiene el listado de todas las incidencias posibles para el usuario.

Campo	Tipo / Relación	Descripción
Id	Primaria, entero, no nulo	Clave Primaria
Texto	Carácter	Contiene el texto de la incidencia

Usuario

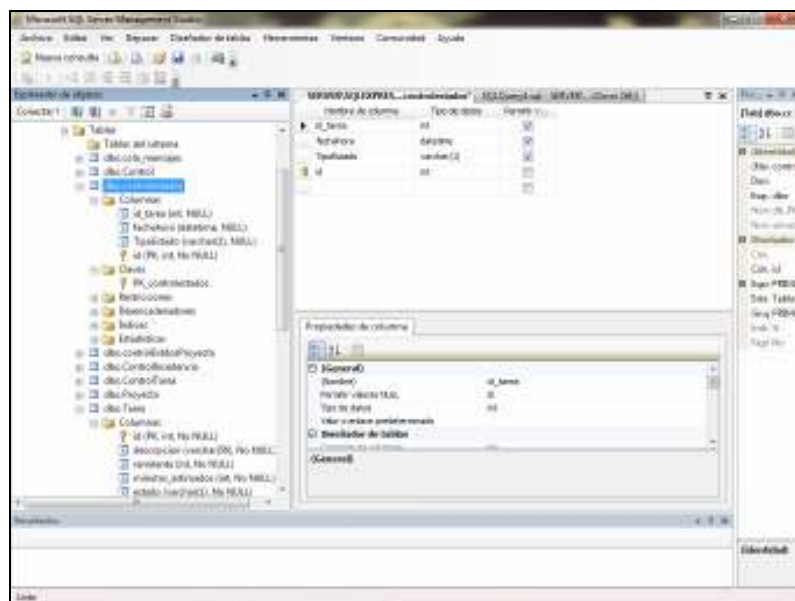
Contiene la información del usuario.

Campo	Tipo / Relación	Descripción
Id_Usuario	Primaria, entero, no nulo	Clave Primaria
Nombre	Carácter, no nulo	Contiene el nombre
Apellido	Carácter, no nulo	Contiene el apellido
Telefono	Carácter, no nulo	Contiene el teléfono
Direccion	Carácter, no nulo	Contiene la dirección
Mail	Carácter, no nulo	Contiene el email
Puesto	Carácter, nulo	Contiene el puesto

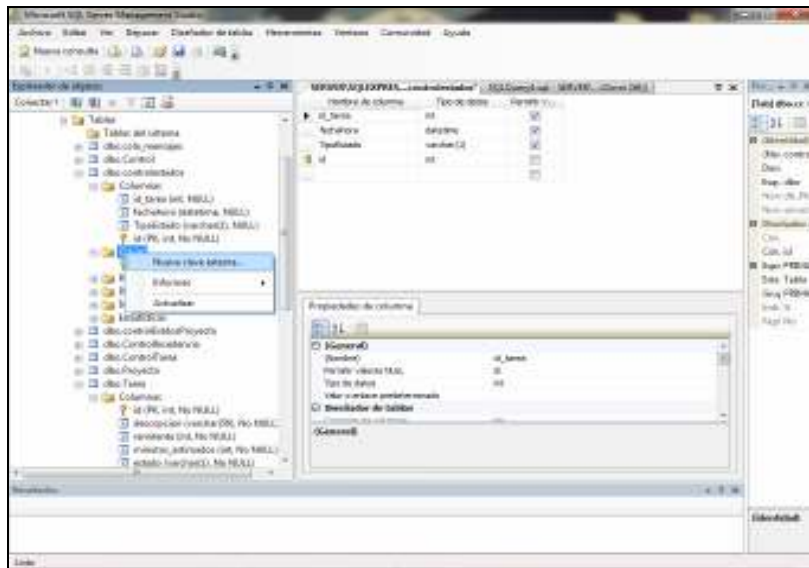
Contraseña	Carácter, nulo	Contiene la contraseña
Rol	Entero, nulo	Indica Rol – 0 admin, 1 usuario
Nick	Carácter, nulo	Contiene el nick
Foto	Carácter, nulo	Uso futuro, campo foto

4.3.3.- Relación entre tablas

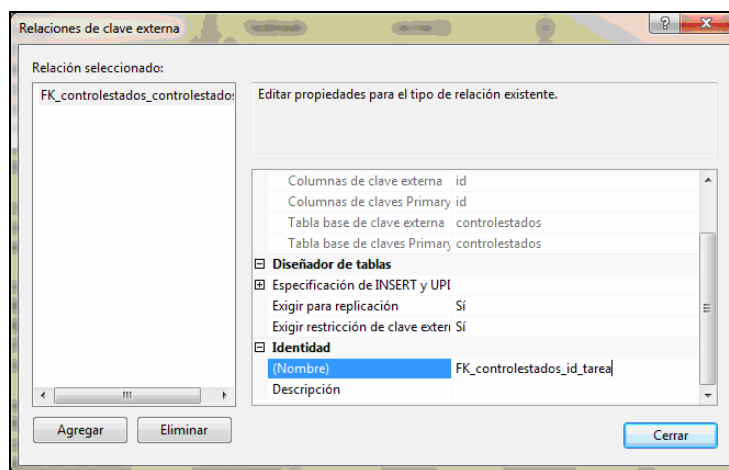
Se ha creado la relación entre tablas, de la siguiente manera, en primer lugar seleccionamos la tabla en la cual queremos crear la relación.



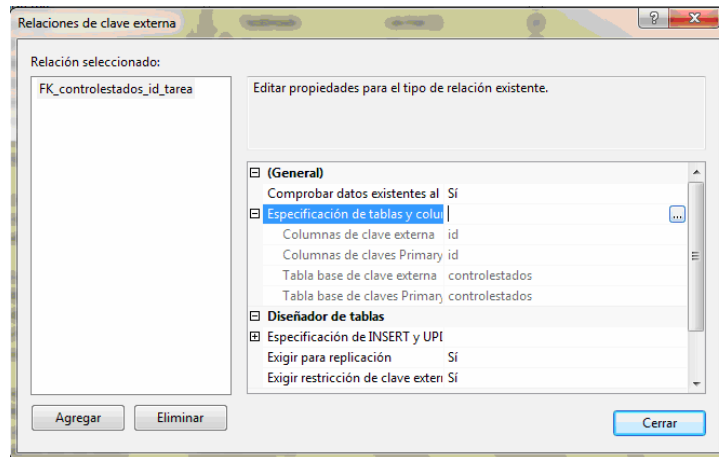
En segundo lugar, mediante el botón derecho conseguimos hacer aparecer un menú emergente que nos permite crear una nueva clave.



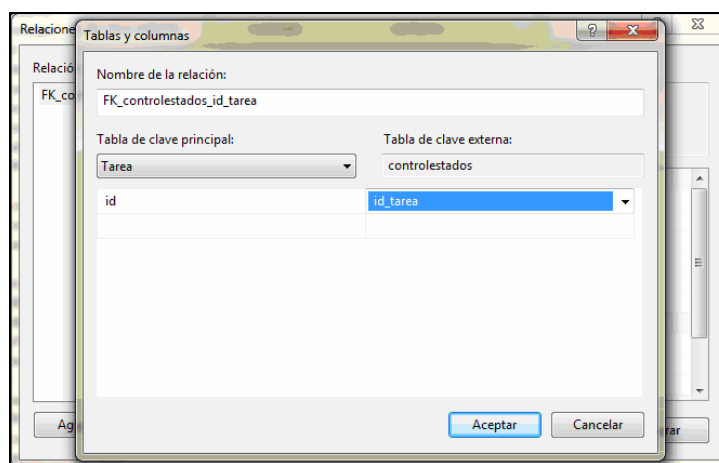
Nos aparece una ventana en la cual introducimos el nombre de la clave o relación.



Ahora especificamos la relación entre las distintas tablas.



Para esta tabla seleccionamos como tabla principal la tabla “Tarea” y el campo “id” de la tabla, como clave de externa pondremos la tabla “controlados” y como campo el “id_tarea”.



4.3.4.- Procedimientos almacenados

Dos de las cuestiones más importantes para el usuario de bases de datos son la velocidad y la eficiencia. En SQL Server la herramienta diseñada principalmente para optimizar la obtención de datos, es el procedimiento almacenado.

Un procedimiento almacenado es una consulta que se almacena en una base de datos en SQL Server en lugar de almacenarse en el código cliente (C# o Java) en el equipo cliente.

Creación de procedimientos almacenados

La instrucción general para crear procedimientos almacenados es la siguiente:

```
CREATE PROC nombre_proc    parametros
AS
    INSTRUCCION SQL
```

Es necesario aclarar, que un procedimiento almacenado puede recibir parámetros de entrada y devolver parámetros de salida.

Así que, para asegurar la integridad y mejorar la seguridad, la aplicación cuenta con varios procedimientos almacenados capaces de gestionar las entradas de datos en la BBDD.

Insertar Entrada Control

Procedimiento encargado de hacer la entrada del usuario en la sesión, si encuentra una sesión que fue no cerrada con anterioridad, la cierra indicando que hubo anomalía en esta.

Insertar Entrada Control Tarea

Procedimiento encargado de hacer la entrada de en la sesión de la tarea, si encuentra una sesión de tarea que fue no cerrada con anterioridad, la cierra indicando que hubo anomalía en esta.

Insertar Salida Control

Procedimiento encargado de cerrar la sesión del usuario.

Insertar Salida Control Tarea

Procedimiento encargado de cerrar la sesión de la tarea.

Vivo y Actualizar

Procedimiento encargado de actualizar los minutos transcurridos en la tarea, además realiza el acumulativo tanto en tarea como en la sesión de control. Es llamado constantemente por el demonio, con lo cual se obtiene además un vivo del usuario.

A continuación se muestra un ejemplo del: insertar_Entrada_Control_Tarea:

```
USE [db_Pfc]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[Insertar_Entrada_Control_Tarea]
(
    @Id_Usuario int,
    @Id_Control int,
    @Id_Tarea int
)
AS
declare @Hora datetime

SET NOCOUNT OFF;

SET @Hora=GETDATE();
// Aquí obtenemos la hora actual

Execute Insertar_Salida_Control_Tarea @Id_Usuario
// Llamamos al procedimiento que se encarga de comprobar si hubo una //
entrada sin cerrar anterior, si la hay la cierra

INSERT INTO ControlTarea (id_tarea, id_usuario, id_control, hora_in,
minutos_trabajados)
VALUES (@Id_Tarea, @Id_Usuario, @Id_Control, @Hora, 0);
// Inserta el registro correspondiente a la sesión de tarea actual

SELECT ControlTarea.id, ControlTarea.hora_in, (Select Tarea.descripcion from
Tarea Where Tarea.Id=@Id_Tarea), (Select Tarea.minutos_estimados from Tarea
Where Tarea.Id=@Id_Tarea), (Select Tarea.minutos_trabajados from Tarea Where
Tarea.Id=@Id Tarea) from ControlTarea where
ControlTarea.id=SCOPE_IDENTITY();
// Devuelve el resultado del registro creado a la aplicación.
```

5. Implantación

5.1.- Introducción

Para el desarrollo de este proyecto se van a utilizar las últimas herramientas, estándares y protocolos disponibles en desarrollo de aplicaciones proporcionadas por Microsoft y por la industria de desarrollo de software.

Se trabajará con una máquina con sistema operativo Windows 7, con la versión 6 de Internet Information Server y con la versión gratuita del gestor de base de datos SQL Server. Para el diseño de la aplicación se utilizará Lenguaje Unificado de Modelado (UML) y para el desarrollo se utilizará el entorno proporcionado por Visual Studio 2010 con el lenguaje de programación C#.

Se realiza a continuación una introducción a todas estas herramientas.

5.2.- Herramientas y tecnología utilizada

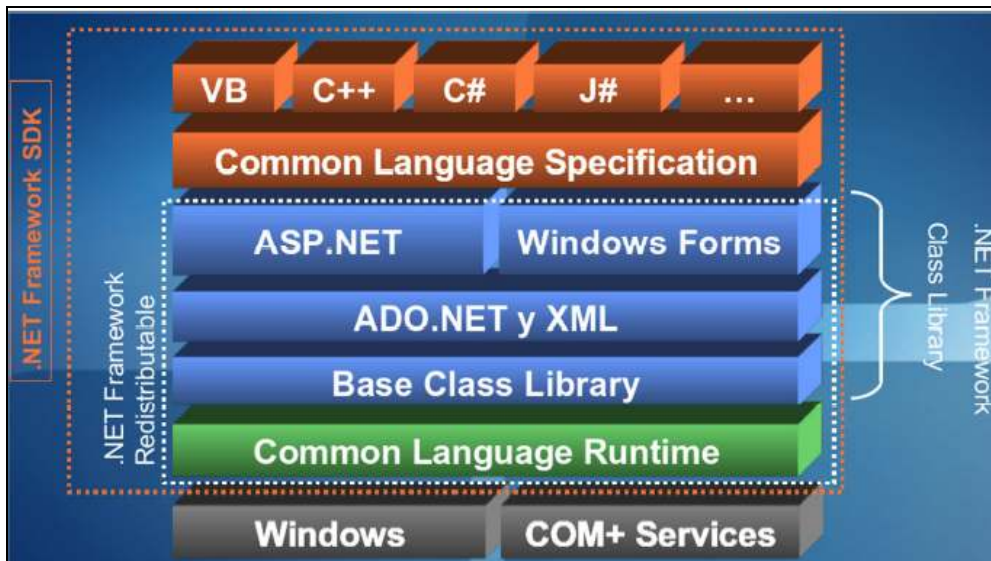
5.2.1.- Microsoft Visual Studio .NET

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros. Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones Web, así como servicios Web en cualquier entorno que soporte la plataforma .NET (a partir de la versión net 2002). Así se pueden crear aplicaciones que se intercomunican entre estaciones de trabajo, páginas Web y dispositivos móviles.

Visual Studio 2010 es la versión más reciente de esta herramienta, acompañada por .NET Framework 4.0.

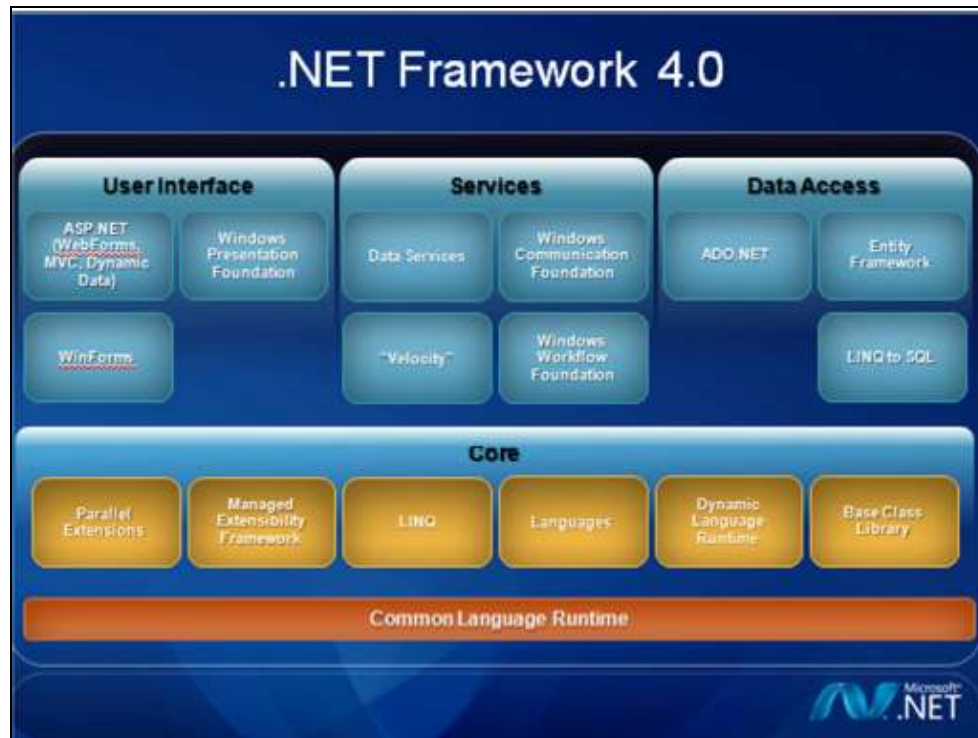
Hasta ahora, uno de los mayores logros de la versión 2010 de Visual Studio ha sido el de incluir las herramientas para el desarrollo de aplicaciones para Windows 7, tales como herramientas para el desarrollo de las características de Windows (System.Windows.Shell) y la Ribbon Preview para WPF.

Entre sus más destacables características, se encuentran la capacidad para utilizar múltiples monitores, así como la posibilidad de desacoplar las ventanas de su sitio original y acoplarlas en otros sitios de la interfaz de trabajo. Además de esto, aparece una edición que compila las características de todas las ediciones comunes de Visual Studio: Professional, Team Studio, Test, conocida como Visual Studio Ultimate. En la siguiente figura se muestra la arquitectura de .Net.



.Net Framework 4

.NET Framework es un componente integral de Windows que admite la compilación y la ejecución de la siguiente generación de aplicaciones y servicios Web. Los componentes clave de .NET Framework son Common Language Runtime (CLR) y la biblioteca de clases .NET Framework, que incluye ADO.NET, ASP.NET, formularios Windows Forms y Windows Presentation Foundation (WPF). .NET Framework proporciona un entorno de ejecución administrado, un desarrollo e implementación simplificada y la integración con una gran variedad de lenguajes de programación. En la figura se puede observar la arquitectura de .Net Framework 4.



Las características y mejoras que se han incluido en .NET Framework 4 son:

- **Compatibilidad y e implementación de aplicaciones:** .NET Framework 4 es muy compatible con las aplicaciones que se han compilado con versiones anteriores de .NET Framework, salvo con respecto a algunos cambios que se han realizado para mejorar la seguridad, el cumplimiento de normas, la exactitud, la confiabilidad y el rendimiento.
- **Diagnósticos y rendimiento:** Puede supervisar el uso de la CPU y de la memoria de cada dominio de aplicación. La supervisión de recursos del dominio de aplicación está disponible a través de las API de hospedaje administrado y nativo, y de Seguimiento de eventos para Windows (ETW). Cuando esta característica está habilitada, recopila estadísticas para todos los dominios de aplicación del proceso mientras dure el proceso.
- **Globalización:** proporciona nuevas referencias culturales neutras y específicas, valores de propiedad actualizados, mejoras en el tratamiento de cadenas y otras mejoras.

- **Recolección de elementos no utilizados:** proporciona recolección de elementos no utilizados en segundo plano. Esta característica reemplaza la recolección simultánea de elementos no utilizados de las versiones anteriores y proporciona un mayor rendimiento.
- **Dynamic Language Runtime (DLR):** es un nuevo entorno en tiempo de ejecución que agrega un conjunto de servicios para lenguajes dinámicos a CLR. Con DLR es más fácil desarrollar lenguajes dinámicos para su ejecución en .NET Framework y agregar características dinámicas a lenguajes con tipos estáticos.
- **Informática en paralelo:** presenta un nuevo modelo de programación para escribir código multiproceso y asíncrono que simplifica considerablemente el trabajo de los desarrolladores de aplicaciones y de bibliotecas. El nuevo modelo permite a los desarrolladores escribir código paralelo eficaz, específico y escalable en una locución natural sin tener que trabajar directamente con subprocesos o el bloque de subprocesos.
- **Redes:** Mejoras de seguridad para la autenticación de Windows en varias clases, compatibilidad con NAT, nuevos contadores de rendimiento de la red que proporcionan información sobre los objetos `HttpRequest`.

5.2.2.- C#

C# se trata de un nuevo lenguaje de programación orientada a objetos creado por Microsoft, que forma parte de la plataforma .NET. En este proyecto se ha trabajado con la versión 4.0 de C#, aunque actualmente ya existe una versión 4.5 en fase beta.

Aunque desde un principio fue incluido en esta plataforma, C# podría utilizarse para desarrollar aplicaciones en otras plataformas, sólo sería necesario un compilador específico de C# para la plataforma requerida.

Posee una sintaxis similar a C++ o Java, y fue diseñado, según Microsoft, para reunir las virtudes de Java, C++ y Visual Basic en un solo lenguaje, ampliando el concepto de lenguaje orientado a objetos al de lenguaje orientado a componentes.

Al ser un lenguaje completamente nuevo, que no ha sido adaptado para trabajar en .NET, sino que se ha desarrollado para trabajar específicamente en esta plataforma, C# es el lenguaje más adecuado para desarrolladores que quieran trabajar con la nueva plataforma de Microsoft, ya que el lenguaje está optimizado para .NET. Es por tanto, el lenguaje nativo de la plataforma .NET.

ECMA, el estándar que validó C# para que fuera admitido en la lista de lenguaje perteneciente a la plataforma .NET afirma que C# debe poseer las siguientes características:

- Lenguaje de propósito general, orientado a objetos y simple en su sintaxis.
- Debe poder usarse para desarrollar componentes software en aplicaciones distribuidas.
- Debe incluir mecanismos de revisión y detección de errores, como por ejemplo comprobación de variables usadas sin inicialización previa, tipos de datos y límites de un array, etc.
- Portabilidad del código fuente.
- Debe servir para desarrollar cualquier proyecto software.
- Soporte para internacionalización.

A continuación se presentarán las características más destacadas de C#, aunque algunas de ellas son características generales de cualquier lenguaje incluido en la plataforma .NET.

- Código autocontenido. C# no necesita ficheros adicionales al propio código fuente, como DLLs para que las aplicaciones escritas en este lenguaje funcionen correctamente.
- Tamaño de los tipos de datos básicos del lenguaje: Presentan un tamaño independiente (no depende del tipo de máquina o del sistema operativo que se esté usando) y fijo, lo que permite la portabilidad del código.
- Inclusión de sentencias originales de otros lenguajes, como `foreach`, o el tipo básico decimal. Estos elementos han demostrado ser útiles para los desarrolladores y por este motivo han sido incluidos en C#.
- C# no admite el uso de funciones o variables globales, para evitar confusiones y facilitar la comprensión del código fuente. Todas las definiciones de variables y funciones deben estar incluidas en definiciones de tipos de datos.

- Se trata de un lenguaje orientado a objetos, soportando el polimorfismo, la herencia (no soporta herencia múltiple) y la encapsulación.
- C# es un lenguaje orientado a componentes. Se puede, por tanto, definir una clase, que actúe como componente, definiendo sus propiedades, sus eventos y sus atributos de una manera muy simple.
- C# incluye el concepto de delegados. Un delegado es algo parecido a un puntero a una función, salvo que el delegado tiene el comportamiento de un objeto. Un delegado puede almacenar referencias de varios métodos simultáneamente.
- Control de instrucciones. Para evitar errores comunes y hacer más seguro el código fuente, en C# se han incluido una serie de comprobaciones en el código.
- De esta forma, la evaluación de una condición será lógica, evitando confusiones con el operador = y el operador ==. Además, cada caso de la instrucción switch deberá finalizar con break o goto.
- C# es un lenguaje que usa el concepto de sistema de tipos de datos unificado. Esto quiere decir que todas las clases de objetos que se definen en este lenguaje derivan de una clase general, llamada System.Object. Así se evitan confusiones, aumenta el rendimiento en la ejecución y permite el desarrollo de colecciones que almacenen tipos de datos genéricos.
- En C#, los tipos de datos básicos pueden actuar como objetos o no. Este concepto se llama boxing y unboxing. Es decir, los tipos de datos básicos sólo serán tratados como objetos cuando verdaderamente se requiera, mientras tanto, se los considerará como tipos de datos básicos.
- C# incorpora un nuevo mecanismo, llamado indizador. Un indizador, permite acceder de una forma rápida a un elemento almacenado en una colección, utilizando el operador [], como si la colección se tratara de un array.
- C# permite utilizar punteros de una manera muy parecida a C++, aumentando la eficiencia y la velocidad de procesamiento. Sin embargo, C# no permite el uso de punteros ya que es un lenguaje muy estricto en materia de seguridad, y para poder usar punteros, hay que incluir la palabra reservada unsafe, marcando esa región de código como insegura.
- En C# se pueden incluir fragmentos de código escritos en Java, C y C++.
- Además, C# incluye la posibilidad de acceder a código nativo escrito como funciones no orientadas a objetos. Estas funciones normalmente se refieren a

DLLs y al API de Windows. Para acceder a código nativo se debe utilizar la sentencia PInvoke.

5.2.3.- XML

XML es una tecnología que tiene a su alrededor otras tecnologías que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores.

XML es interesante en el mundo de Internet y el e-bussiness, ya que existen muchos sistemas distintos que tienen que comunicarse entre sí. Pero interesa por igual a todas las ramas de la informática y el tratamiento de datos, ya que permite muchos avances a la hora de trabajar con ellos.

XML se puede usar para infinidad de trabajos y aporta muchas ventajas en amplios escenarios. Algunas ventajas del XML en algunos campos prácticos son:

- **Comunicación de datos:** Si la información se transfiere en XML, cualquier aplicación podría escribir un documento de texto plano con los datos que estaba manejando en formato XML y otra aplicación recibir esta información y trabajar con ella.
- **Migración de datos:** Si tenemos que mover los datos de una Base de Datos a otra sería muy sencillo si las dos trabajasen en formato XML.
- **Aplicaciones web:** Hasta ahora cada navegador interpreta la información a su manera y los programadores de páginas Web tienen que hacer unas cosas u otras en función del navegador del usuario. Con XML se puede tener una sola aplicación que maneja los datos y para cada navegador o soporte se tendrá una hoja de estilo o similar para aplicarle el estilo adecuado.

5.2.4.- Internet Information Services (IIS)

Para la construcción del sistema se requiere de un servidor de Internet que de servicio a los subsistemas del presente proyecto, concretamente, a las aplicaciones instaladas en los dispositivos móviles y como a las aplicaciones instaladas en los puestos de escritorio

instalados. El servidor habilitado a tal efecto es el *Internet Information Server*, con lo cual será necesario adaptarse a sus posibilidades. Se deberá de configurar un sitio Web que contenga los servicios Web que den soporte a los subsistemas y, asimismo, establecer el sistema de autenticación para controlar el acceso a los mismos. IIS, es una serie de servicios para los ordenadores que funcionan con Windows. Originalmente era parte del *Option Pack para Windows NT*. Luego fue integrado en otros sistemas operativos de Microsoft destinados a ofrecer servicios, como Windows 2000 o Windows Server 2003. Windows XP Profesional incluye una versión limitada de IIS. Los servicios que ofrece son: *FTP, SMTP, NNTP y HTTP/HTTPS*. Este servicio convierte a un ordenador en un servidor de Internet o Intranet es decir que en las maquinas que tienen este servicio instalado se pueden publicar páginas Web tanto local como remotamente (servidor Web).

El servidor Web se basa en varios módulos que le dan capacidad para procesar distintos tipos de páginas, por ejemplo Microsoft incluye los de *Active Server Pages (ASP)* y *ASP.NET*. También pueden ser incluidos los de otros fabricantes, como PHP o Perl. La versión actual de IIS es la 6.0 para *Windows Server 2003* e IIS 5.1 para Windows XP Profesional. IIS 5.1 para Windows XP es una versión compacta del IIS que soporta solo 10 conexiones simultaneas y solo un sitio Web, aunque puede ser extensible mediante el manejo de AdminScripts instaladas en la ruta del servidor. En el presente proyecto se trabajará con las versiones actuales mencionadas, IIS 5.1 para el entorno de desarrollo e IIS 6.0 para el entorno de producción.

IIS gestiona sitios Web. Un sitio Web es un conjunto de páginas Web, típicamente comunes a un dominio en Internet o subdominio en la World Wide Web en Internet. Una página Web es un documento HTML/XHTML accesible generalmente mediante el protocolo HTTP de Internet. A las páginas de un sitio Web se accede desde una URL raíz común llamada portada, que normalmente reside en el mismo servidor físico. Las URLs organizan las páginas en una jerarquía, aunque los hiperenlaces entre ellas controlan cómo el lector percibe la estructura general y cómo el tráfico Web fluye entre las diferentes partes de los sitios. Los sitios Web están escritos en HTML (Hyper Text Markup Language), o dinámicamente convertidos a éste y se acceden usando un cliente http, navegador Web o cualquier otro. Los sitios Web pueden ser visualizados o accedidos desde un abanico de dispositivos con disponibilidad de Internet como computadoras personales, computadores portátiles, PDAs y teléfonos móviles. En IIS se pueden configurar los sitios Web para que se acceda a ellos a través de un puerto TCP

diferente al puerto TCP por defecto para los servidores Web, los tipos MIME que son reconocidos, la versión del ASP.NET que se utiliza así como una serie de parámetros propios de un sitio Web: (Control de acceso, seguridad, configuración ssl...). Al configurar los sitios Web debe indicar los directorios que contienen los documentos que desea publicar. El servidor Web no puede publicar documentos que no están en los directorios especificados.

Cada sitio Web o FTP debe tener un directorio particular. El directorio particular es la ubicación central de las páginas publicadas. En este caso, por seguridad del sitio Web, conviene crear un directorio particular diferente al que viene por defecto. Uno de los parámetros configurables corresponde a la seguridad de acceso a un sitio Web o a un directorio particular del mismo. Se accede (IIS 7.0) desde las propiedades del sitio Web o directorio y ofrece varias posibilidades:

- **Autenticación anónima:** cuando se activa el acceso anónimo, no se requieren credenciales de usuario autenticado para tener acceso al sitio. El uso más adecuado de esta opción es para conceder acceso público a la información que no requiere seguridad. Cuando un usuario intenta conectarse al sitio Web, IIS asigna la conexión a la cuenta IUSER_nombreDeEquipo, donde nombreDeEquipo es el nombre del servidor en el que se está ejecutando IIS. De forma predeterminada, la cuenta IUSER_nombreDeEquipo es miembro del grupo Invitados. Este grupo tiene restricciones de seguridad, impuestas por los permisos del sistema de archivos NTFS, que indican el nivel de acceso y el tipo de contenido que está disponible para los usuarios públicos. Se puede configurar la cuenta de Windows que se utiliza para el acceso anónimo.
- **Autenticación básica:** la autenticación básica requiere un Id. de usuario y una contraseña, y proporciona un nivel bajo de seguridad. Las credenciales del usuario se envían en texto sin cifrar a través de la red. Este formato proporciona un nivel bajo de seguridad, porque casi todos los analizadores de protocolo pueden leer la contraseña. Sin embargo, es compatible con el número más amplio de clientes Web. El uso más adecuado de esta opción es para conceder acceso a información con poca o ninguna necesidad de privacidad.
- **Autenticación de Windows:** anteriormente se denominaba NTLM o autenticación por desafío/respuesta de Windows NT. Este método envía la

información de autenticación del usuario por la red en forma de vale de Kerberos y proporciona un alto nivel de seguridad. La autenticación de Windows integrada utiliza la versión 5 de Kerberos y la autenticación NTLM.

Para emplear este método, los clientes deben utilizar Microsoft Internet Explorer 2.0 o posterior. Adicionalmente, la autenticación de Windows integrada no se admite sobre conexiones de proxy HTTP. El uso más adecuado de esta opción es para una intranet, donde el usuario y el servidor Web están en el mismo dominio, y los administradores pueden asegurarse de que todos los usuarios utilizan Internet Explorer 2.0 o posterior.

Autenticación mediante formularios: La autenticación de formularios hace referencia a un sistema en el que la solicitudes no autenticadas se redirigen a un formulario de Lenguaje de marcado de hipertexto (HTML) en el que los usuarios escriben sus credenciales. Una vez que el usuario proporciona las credenciales y envía el formulario, la aplicación autentica la solicitud y el sistema emite un vale de autorización en el formulario de una cookie. Esta cookie contiene las credenciales o una clave para readquirir la identidad. Las solicitudes subsiguientes del explorador automáticamente incluyen la cookie.

Suplantación de ASP.NET: Corresponde más que a un método alternativo de autenticación o un añadido en el proceso de autenticación de aplicaciones Web. El escenario de suplantación se basa en la autenticación de Servicios de Microsoft Internet Information Server (IIS) y en la seguridad de acceso a archivos de Microsoft Windows para minimizar la programación de la seguridad en la propia aplicación ASP.NET. El flujo de datos se muestra en la ilustración siguiente.

5.2.5.- Servicios Web

Los servicios Web son una colección de protocolos y estándares, que se usan para el intercambio de información entre sistemas. Como principal característica, estos servicios están publicados, definidos y localizados en Internet. Las características que tienen los Servicios Web son las siguientes:

- Los Servicios Web son accesibles por la Web de Internet.

- La comunicación con los Servicios Web es independiente de la plataforma, ya que se usa protocolos con un lenguaje neutral.
- El Servicio Web esta registrado y puede ser localizado por medio del registro de Servicios Web.
- El Servicio Web permite la conexión entre sistemas bajamente acoplados.

Los aspectos importantes de los Servicios Web son los siguientes:

Protocolo Estándar: El Servicio es expuesto funcionalmente vía interfaces usando los principales protocolos de Internet como son HTTP, SMTP, FTP, etc.... Aunque en la mayoría de casos el protocolo es el HTTP.

Descripción del Servicio: Los Servicios Web necesitan describir todas sus interfaces en detalle para que el cliente que quiera consumirlo disponga de toda la funcionalidad dispuesta por el Servicio. Esta descripción se realiza en XML, mediante un documento llamado WSDL (Web Services Description Language).

Búsqueda de Servicios: Los clientes necesitan saber que Servicios existen y donde esta ubicados para poder consumirlos. Para que los clientes conozcan y encuentren los Servicios disponen de un repositorio UDDI (Universal Discovery, Description and Integration), donde se muestran el listado de Servicios disponibles.

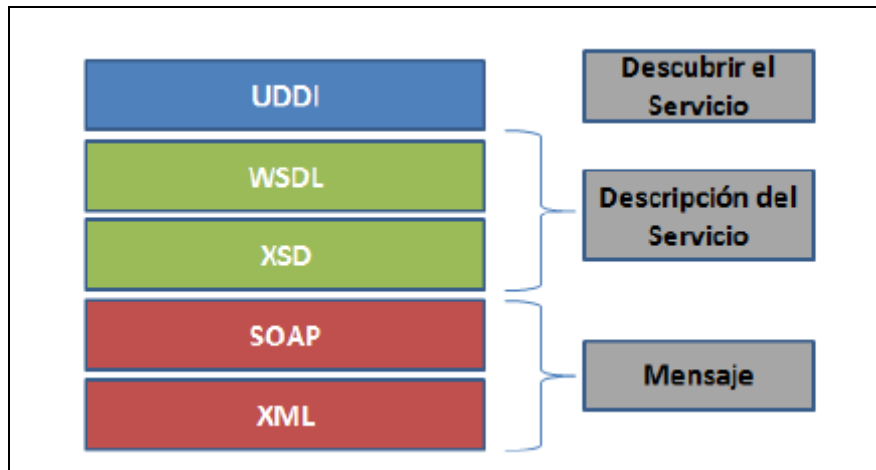
Por lo tanto los protocolos que utiliza el Servicio Web son las siguientes:

- XML (eXtensible Markup Language): XML es un metalenguaje extensible de etiquetas desarrollado por W3C (World Wide Web Consortium). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos. Los Servicios Web utilizan este protocolo para representar la información, esta información se estructurará mediante los esquemas de SOAP.
- SOAP (Simple Object Access Protocol): SOAP es un protocolo de comunicaciones para los Servicios Web que se basa en el estándar XML. Este

protocolo se usa para el intercambio de información de forma estructurada y tipificada entre los sistemas. SOAP permite la invocación de métodos en diferentes sistemas sin tener que conocer los detalles de su plataforma o aplicaciones que se están ejecutando. La comunicación SOAP se realiza mediante los Mensajes SOAP que contienen la siguiente información: - Primera Parte: SOAP envuelve el documento XML que encapsula el mensaje que se está comunicando.

- Segunda Parte: Se usa para definir tipos de datos personalizados que existen en el Servicio.
 - Tercera Parte: Define el patrón RPC (Remote Procedure Call) que va a ser usado.
 - Cuarta Parte: Define como SOAP se une a HTTP.
-
- WSDL (Web Services Description Language): WSDL es un lenguaje basado en XML para describir el comportamiento de un Servicio Web, conteniendo los métodos y los tipos de datos necesarios para definir dichas operaciones. Es necesario para que el cliente o consumidor pueda invocar el Servicio Web.
 - UDDI (Universal Description, Discovery, and Integration): UDDI es un registro independiente de la plataforma basado en XML para registrar los Servicio Web y permitir posteriormente descubrirlos.

En la siguiente figura se puede ver la pila de protocolos para los Servicios Web.



Para consumir un Servicio Web primero se debe descubrir el Servicio y ver que ofrece, esta operación se realiza mediante el protocolo UDDI. Una vez, se haya descubierto el Servicio se debe entender la interfaz, es decir, los métodos y parámetros que acepta el servicio para ser consumido. Esta operación se realiza mediante los protocolos WSDL y XSD. Finalmente el consumo del Servicio Web se realiza mediante mensajes los cuales están en formato XML y encapsulados mediante el protocolo SOAP. De esta forma y en este orden, se utilizaría la pila de protocolos de un Servicio Web. En la siguiente figura se muestra el ciclo de vida del Servicio Web:



5.2.6.- Microsoft SQL Server

Microsoft SQL Server 2008 es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional. Sus lenguajes para consultas son T-SQL y ANSI SQL.

Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle, Sybase ASE, PostgreSQL, Interbase, Firebird o MySQL.

Microsoft SQL Server se encuentra en los primeros lugares en cuanto a SGBD se trata, debido a que muchas empresas optaron por la utilización de un producto creado por una empresa puntera como Microsoft y que con el paso del tiempo han seguido confiando en este software que cada vez se intenta aproximar mas al usuario y reduciendo sustancialmente la dificultad de las tareas que conlleva la gestión de una base de datos.

A continuación se exponen algunas de sus principales características:

- Soporte de transacciones.
- Escalabilidad, estabilidad y seguridad.
- Soporta procedimientos almacenados.
- Incluye también un potente entorno grafico de administración, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y las terminales o clientes de la red solo acceden a la información.
- Además permite administrar información de otros servidores de datos.

T-SQL (Transact-SQL) es el principal medio de programación y administración de SQL Server. Expone las palabras clave para las operaciones que pueden realizarse en SQL Server, incluyendo creación y modificación de esquemas de la base de datos, introducir y editar datos en la base de datos, así como supervisión y gestión del propio servidor.

Las aplicaciones cliente, ya sea que consuman datos o administren el servidor, aprovechan la funcionalidad de SQL Server mediante el envío de consultas de T - SQL y declaraciones que son procesadas por el servidor y los resultados (o errores) regresan a la aplicación cliente.

5.2.7.- Kinect



Recientemente ha salido al mercado un interesante sensor, comercialmente llamado Kinect. En la Wikipedia encontramos que Kinect se puede definir como:

“un controlador de juego libre y entretenimiento desarrollado por Microsoft para la videoconsola Xbox 360. Kinect permite a los usuarios controlar e interactuar con la consola sin necesidad de tener contacto físico con un controlador de videojuegos tradicional, mediante una interfaz natural de usuario que reconoce gestos, comandos de voz, y objetos e imágenes “.

Microsoft Research invirtió veinte años de desarrollo en la tecnología de Kinect. Fue anunciado por primera vez el 1 de junio de 2009 en la Electronic Entertainment Expo 2009 como "Project Natal". El sensor de Kinect es una barra horizontal de aproximadamente 23cm conectada a una pequeña base circular con un eje de articulación de rótula, y está diseñado para ser colocado longitudinalmente por encima o por debajo del televisor.

Kinect se compone principalmente de:

- Una cámara tradicional (Resolución 640x480 RGB 30fps VGA).
- Un emisor de infrarrojos.

- Una cámara de infrarrojos.
- 4 Micrófonos (16bit sampling rate: 16Hz).
- Un motor.

¿Como funciona?

Kinect tiene las mismas características que una cámara normal, lo que la hace tan especial es la integración del sensor de profundidad junto a su procesador.

Para detectar la profundidad Kinect utiliza un emisor de infrarrojos para emitir una nube de puntos, si utilizáramos un visor nocturno se podría observar este campo, como vemos en la siguiente imagen.



Esta multitud de puntos marca la distancia, es decir, Kinect *calcula la distancia entre donde estaba el punto al proyectarlo a donde está en la proyección.*

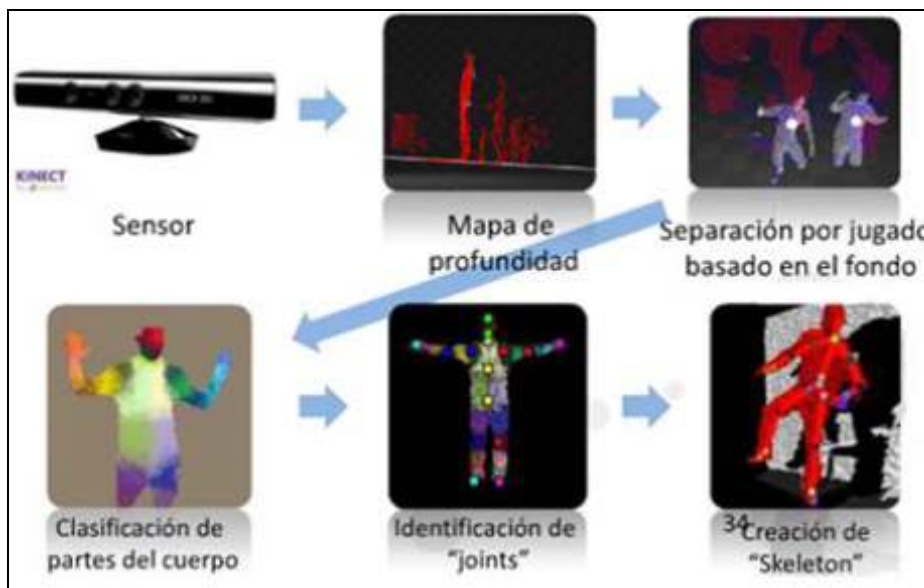
Pero como decíamos, esto sería una simple cámara con un sensor de profundidad si no fuese por la tecnología que lleva integrada que es capaz de reconocer el cuerpo humano.

Skeleton Tracking

Este proceso es el método por el que Kinect es capaz de detectar la figura humana cuando se sitúa delante.

Consta de 6 pasos:

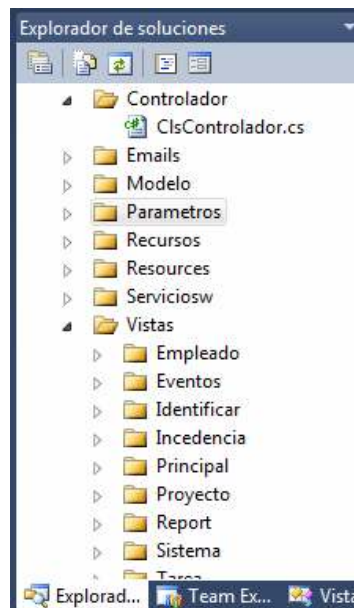
- 1- El sensor lanza la nube de puntos
- 2- Kinect crea el mapa de profundidad a partir del sensor.
- 3- Detecta el suelo y separa los objetos del fondo para encontrar el contorno humano.
- 4- Hace una clasificación de las partes humanas.
- 5- Identifica las articulaciones.
- 6- Recrea el esqueleto.



5.3.-Arquitectura de software

Para el desarrollo de LupaApp se ha optado por el patrón **MVC**. De esta forma se organiza la aplicación en tres modelos separados, el primero es *un modelo* que representa los datos de la aplicación y sus reglas de negocio, el segundo es un conjunto de *vistas* que representa los formularios de entrada y salida de información, el tercero es un *controlador* que procesa la peticiones de los usuarios y controla el flujo de ejecución del sistema.

Tal y como se puede ver en la figura de abajo, la estructura de la aplicación esta basada en este patrón. Una serie de vistas están subdivididas en cada una de las carpetas que puede contener a su vez; formularios de mantenimiento, formularios de trabajo, reportes, etc.



En el controlador recae prácticamente la totalidad de la lógica de la aplicación, esto nos facilita los futuros cambios en desarrollos, resulta un código de fácil y flexible estructuración, separando de manera eficiente las distintas funcionalidades.

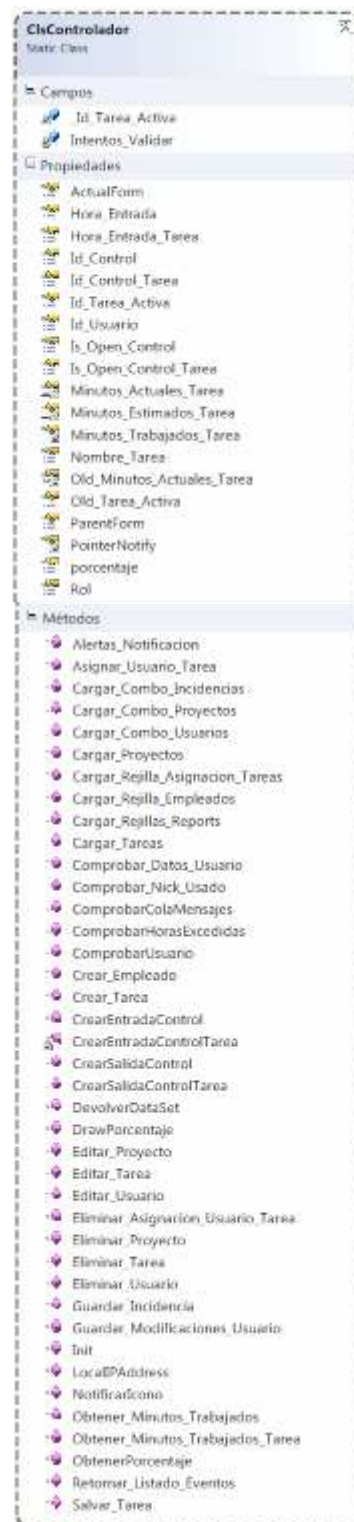
Al iniciar, es lanzado en primer lugar el controlador, este lleva a cabo una interacción continuada con cada uno de los formularios(vistas), los cuales son responsables de recoger los eventos del usuario, a la vez que representan los resultados.

Todos los datos son consultados al modelo, en este caso serán los propios servicios web, mediante los cuales, el controlador decidirá de forma arbitraria la secuencia lógica de cada proceso.

El controlador posee clases estáticas que ofrecen acceso a otras capas, con lo cual se consigue delimitar ciertos accesos que ofrecerán información pormenorizada de los sucesos producidos en la sesión del usuario (tiempos, paradas, etc). Esto resulta especialmente útil de cara a aumentar la conectividad con otros proyectos y desarrollos futuros.

Cabe resaltar a su vez, que el controlador tiene acceso a ciertos objetos de diferentes vistas. Estos objetos, tales como iconos de la barra de estado, son gestionados por esta capa, lo cual permite un control prácticamente total de los sucesos y eventos del sistema al controlador.

En la figura de abajo podemos apreciar el diagrama de clase del controlador:



Dentro de este patrón, se puede optar por un único controlador o varios. En nuestro caso hemos elegido la primera opción, debido principalmente a la capacidad de gestionar de forma fácil todas las vistas con una lógica que en muchos casos resultaba compartida para cualquiera de estas.

Mostramos un ejemplo con parte de la lógica, en esta secuencia podemos ver como mediante una propiedad no solo se toma la tarea activa, además, se actualiza la información de la barra de estado, devolviendo en cada caso y en función de la lógica implementada el resultado de que la tarea ha sido seleccionada e iniciada.

```
public static int Id_Tarea_Activa {
    set
    {
        // si coincide que es la misma que la anterior
        if (_Id_Tarea_Activa == value)
        {
            MessageBox.Show("Estás trabajando ya en esta tarea",
"CAMBIO DE TAREA");
            return;
        }
        if(CrearEntradaControlTarea(value))
        {
            _Id_Tarea_Activa = value;
            //envio abajo el nombre de tarea asi como los minutos tarea
            ParentForm.Tarea_Seleccionada.Text = Nombre_Tarea;
            Minutos_Actuales_Tarea = 0;
            Old_Minutos_Actuales_Tarea = 0;

            NotificarIcono(string.Format("Trabajando en Tarea {0}", Nombre_Tarea),
            "Iniciando Tarea");

            DrawPorcentaje();
            ParentForm.porciento realizado.Text =
string.Format("Completado Tarea: {0}%", OtenerPorcentaje());
            ParentForm.Minutos_Tarea.Text = "Minutos Tarea: 0";

            ParentForm.BotonEstado.Enabled = true;
            ParentForm.BotonEstado.Text = "Iniciada";

            ActualForm.Close();
        }
    }
    get
    {
        return _Id_Tarea_Activa;
    }
}
public static void DrawPorcentaje()
{
    ParentForm.Porcentaje.Minimum = 0;
    ParentForm.Porcentaje.Maximum = 100;
    ParentForm.Porcentaje.Value = OtenerPorcentaje() >
ParentForm.Porcentaje.Maximum ? 100 : OtenerPorcentaje();
}
public static int OtenerPorcentaje()
{
    //solamente para evitar dividir entre cer
```

```
        if (Minutos_Actuales_Tarea == 0 && Minutos_Trabajados_Tarea==0)
return 0;
        return ((Minutos_Actuales_Tarea + Minutos_Trabajados_Tarea) * 100)
/ Minutos_Estimados_Tarea;
    }
```

5.4.- Desarrollo Kinect

Kinect se basa principalmente en el trazo elemental de los nodos del esqueleto humano, retornando información completa y concisa de cada punto dentro de un espacio 3D. Gracias a sensores de profundidad se puede obtener la ubicación de la persona dentro de cualquiera de los tres ejes sin importar los aspectos relativos a la luminosidad.

Ha resultado relativamente sencilla la implementación de Kinect en LupApp gracias a la facilidad y abstracción que nos ofrece la clase Skeleton, sin embargo, una de las opciones más costosas en cuanto a requerimientos de sistema ha pasado por el uso del framework necesario para poder desarrollar con el dispositivo de Microsoft. Concretar además, que lo hicimos en WPF, lo cual ha repercutido en la necesidad de unir mediante una interfaz con la aplicación principal, la cual fue ha sido creado en su totalidad en Winform.

Comentando la clase principal que gestiona kinect (fig. 4.1), tenemos un método llamado “*InitializeNui()*” el cual es invocado de manera externa con el fin de inicializar el dispositivo (resolución y profundidad de color). Casi de manera inherente se establece un arranque de la cámara con la idea de capturar video y presentarlo en la ventana cliente. El último punto dentro de este método es la asignación del delegado o creación del evento que será invocado dentro de cada muestreo.

“*SkeletonFrameReady*” método invocado después de cada remuestreo, dentro del objeto enviado se envía la posición relativa al esqueleto del usuario. Un foreach se encarga de recorrer y localizar los nodos, una vez alcanzado el nodo principal de cabeza es localizada y en función de la altura que ha sido establecida se lanza un cambio de estado “*changestate*”. “*Changestate*” es responsable de alertar a la aplicación principal de los distintos cambios de estado del trabajador.

```

namespace KinectGettingStarted
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public event EventHandler ChangeState;

        Runtime _kinectNui;
        private Camera _cam;
        int totalFrames = 0;
        int lastFrames = 0;
        DateTime lastTime = DateTime.MaxValue;

        public MainWindow()
        {
            InitializeComponent();
        }

        private void InitializeNui()
        {
            try
            {
                _kinectNui = new Runtime();

                _kinectNui.Initialize(RuntimeOptions.UseColor |
RuntimeOptions.UseSkeletalTracking | RuntimeOptions.UseColor);
                _kinectNui.VideoStream.Open(ImageStreamType.Video, 2,
ImageResolution.Resolution640x480, ImageType.ColorYuv);

                lastTime = DateTime.Now;

                _cam = _kinectNui.NuiCamera;
                // Cada vez que haya un cambio de evento se llama a eskeletoFrameReady
                _kinectNui.SkeletonFrameReady += new
EventHandlerv<SkeletonFrameReadyEventArgs>(SkeletonFrameReady);
            }
            catch (InvalidOperationException ex)
            {
                MessageBox.Show(ex.Message);
            }
        }

        void SkeletonFrameReady(object sender, SkeletonFrameReadyEventArgs e)
        {
            foreach (SkeletonData data in e.SkeletonFrame.Skeletons)
            {
                //Tracked that defines whether a skeleton is 'tracked' or not. The untracked
skeletons only give their position.
                if (SkeletonTrackingState.Tracked != data.TrackingState) continue;

                foreach (Joint joint in data.Joints)
                {
                    if (joint.Position.W < 0.6f) return;// Quality check
                    switch (joint.ID)
                    {
                        case JointID.Head:
                            var heap = getDisplayPosition(joint);
                            if (heap.Y > 120 && heap.Y < 170)
                            {
                                //label1.Content = "Sentado";
                                ChangeState("S", null);
                            }
                            else if (heap.Y < 90)
                            {
                                //label1.Content = "En Pie";
                                ChangeState("P", null);
                            }
                            break;
                    }
                }
            }
        }
    }
}

```

```

    }
}

private Point getDisplayPosition(Joint joint)
{
    float depthX, depthY;
    _kinectNui.SkeletonEngine.SkeletonToDepthImage(joint.Position, out depthX, out
depthY);
    depthX = Math.Max(0, Math.Min(depthX * 320, 320));
    depthY = Math.Max(0, Math.Min(depthY * 240, 240));
    int colorX, colorY;
    ImageViewArea iv = new ImageViewArea();
    _kinectNui.NuiCamera.GetColorPixelCoordinatesFromDepthPixel(ImageResolution.Resolution640x480,
iv, (int)depthX, (int)depthY, (short)0, out colorX, out colorY);
    return new Point((int)(imageContainer.Width * colorX / 640.0) - 30,
(int)(imageContainer.Height * colorY / 480) - 30);
}

private void WindowLoaded(object sender, RoutedEventArgs e)
{
    InitializeNui();
}

void CalculateFps()
{
    ++totalFrames;

    var cur = DateTime.Now;
    if (cur.Subtract(lastTime) > TimeSpan.FromSeconds(1))
    {
        int frameDiff = totalFrames - lastFrames;
        lastFrames = totalFrames;
        lastTime = cur;
    }
}

void NuiVideoFrameReady(object sender, ImageFrameReadyEventArgs e)
{
    PlanarImage Image = e.ImageFrame.Image;

    image.Source = BitmapSource.Create(
        Image.Width, Image.Height, 96, 96, PixelFormats.Bgr32, null,
        Image.Bits, Image.Width * Image.BytesPerPixel);

    CalculateFps();
}

private void WindowClosed(object sender, EventArgs e)
{
    _kinectNui.Uninitialize();
}

private void BtnStopClick(object sender, RoutedEventArgs e)
{
    _kinectNui.Uninitialize();
}

private void BtnStartClick(object sender, RoutedEventArgs e)
{
    InitializeNui();
}

private void BtnCameraUpClick(object sender, RoutedEventArgs e)
{
    try
    {
        _cam.ElevationAngle = _cam.ElevationAngle + 5;
    }
}

```

```
    }
    catch (InvalidOperationException ex)
    {
        MessageBox.Show(ex.Message);
    }
    catch (ArgumentOutOfRangeException outOfRangeException)
    {
        //Elevation angle must be between Elevation Minimum/Maximum"
        MessageBox.Show(outOfRangeException.Message);
    }
}

private void BtnCameraDownClick(object sender, RoutedEventArgs e)
{
    try
    {
        _cam.ElevationAngle = _cam.ElevationAngle - 5;
    }
    catch (InvalidOperationException ex)
    {
        MessageBox.Show(ex.Message);
    }
    catch (ArgumentOutOfRangeException outOfRangeException)
    {
        MessageBox.Show(outOfRangeException.Message);
    }
}

private void Cambio_Estado()
{
}

}
```


6.-Aplicación Móvil

6.1.-Introducción

Al finalizar la aplicación de escritorio, se me ocurrió la idea de trasladar parte de la aplicación a dispositivos móviles. En la era de los smartphones que estamos viviendo veía muy interesante darle a mi aplicación escritorio un accesorio tan útil y versátil como añadir una aplicación móvil, lo cual sería de gran utilidad para el administrador del sistema.

El objetivo principal de LupaApp móvil es presentar la información más importante en el terminal, con lo cual el gerente de la empresa podrá consultar en cualquier momento los usuarios conectados, trabajando, las horas trabajadas para cada sesión, proyecto y tarea.



6.2.-Titanium Appcelerator: JavaScript

Para programar Titanium Studio proporciona, un IDE basado en Eclipse con el que crear los proyectos y editar los ficheros Javascript. Las aplicaciones se programan íntegramente con Javascript, creando y colocando “a mano” todos los controles, usando para ello una librería que hace de puente entre la aplicación Javascript y los controles del sistema.

Esto significa que las ventanas y demás controles visuales (botones, listas, menús, etc) son nativos, cuando se añade un botón, se crea un botón del sistema y se añade a la vista, lo que lo hace más rápido la creación de la pantalla y la respuesta del usuario es también rápida.

Una de las características más interesantes de Appcelerator es que al empaquetar la aplicación, el Javascript es transformado y compilado.

Después, cuando se arranca la aplicación en el móvil, el código se ejecuta dentro de un engine Javascript, que será JavaScriptCore en IOS (el intérprete del Webkit, el motor de Safari y Chrome) y MozillaRhino en Android/Blackberry.

El hecho de que el Javascript esté compilado y que los controles creados sean nativos, le hace tener el mejor rendimiento posible.

Con Appcelerator es complicado maquetar, pues no existe un HTML inicial donde añadir los controles, sino que hay que crear las ventanas y componentes “a mano” con Javascript. Aún así, se pueden añadir componentes WebKit donde introducir HTML, pero solo se recomienda hacer cuando es absolutamente necesario para mostrar texto con formato.

Los desarrollos de las librerías Javascript para cada sistema operativo evolucionan por separado por lo que es posible que no funcionen de la misma manera. Appcelerator necesita además librerías para manejar los controles nativos y su disposición en la pantalla, por lo que el desarrollo en general es más costoso.

Lo que hace Titanium Studio es generar un proyecto Xcode con el Javascript transformado junto con todas las librerías necesarias. Después es posible lanzar el simulador con la aplicación en Xcode sin salir de Titanium Studio. Una vez generado el proyecto, éste se puede abrir con Xcode y continuar empaquetándolo y configurándolo para su distribución (certificados, provisioning, logos, splash screen, etc). Desde Xcode no se puede editar el Javascript, se debe volver a editar en el Titanium Studio y regenerar el proyecto Xcode otra vez.

Sobre el soporte Android, tanto para probar en el simulador como para empaquetar la aplicación, solo hay que tener el SDK de Android instalado.

Ventajas:

- Multiplataforma móvil y también de escritorio.
- Aspecto y controles nativos proporciona un mejor rendimiento.
- Buenos ejemplos con el KitchenSink.
- Gratis, soporte de pago. Licencia Apache.

Desventajas:

- Requiere Mac y Xcode para empaquetar aplicaciones IOS.
- Definición de componentes visuales y controles “a mano”.
- Mucha documentación pero poco actualizada.



6.3. -Servicios Web de la aplicación móvil

Era consciente de que los servicios Web que usaba para la aplicación me permitirían aprovechar gran parte del trabajo que habría por desarrollar para Android e IOS (uso futuro). Únicamente, tuve que desarrollar algunos métodos que comentaré a continuación:

WServices

Servicios Web para el proyecto

Las operaciones siguientes son compatibles. Para una definición formal, revise la [descripción de servicios](#).

- [GetConectados](#)
Devuelve el recordset con los Usuarios Conectados
- [GetProyectos](#)
Devuelve el recordset con los proyectos.
- [GetUsuarios](#)
Devuelve el recordset con los usuarios.
- [Gettareas](#)
Devuelve el recordset con las tareas.

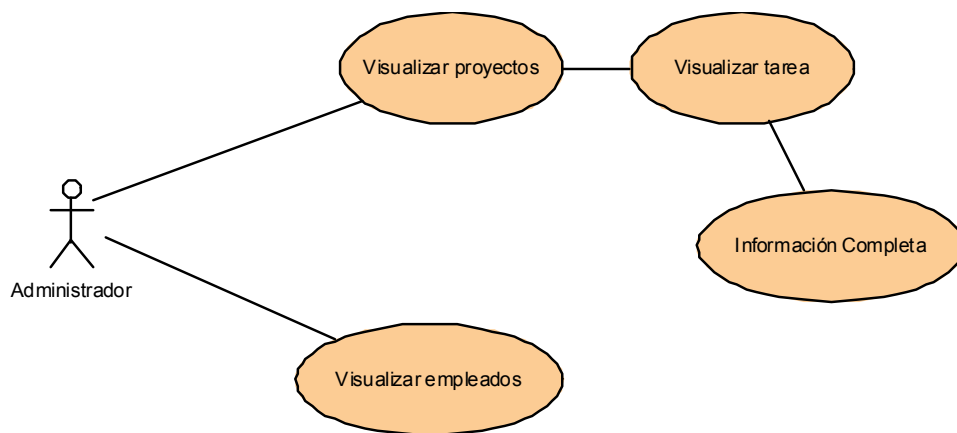
GetUsuarios nos permite ver todos los usuarios que tengo en mi BBDD.

GetConectados nos permite visualizar los usuarios conectados en este momento.

GetProyectos nos permite obtener la descripción de todos los proyectos.
Gettareas nos permite sacar de la BBDD toda la información referente a cualquier tarea de cualquier proyecto.

6.4.-Modelo Caso de uso

El administrador tiene instalada la aplicación móvil en su teléfono, puede acceder en cualquier momentos y en cualquier sitio a un resumen de las diferentes partes que tiene la aplicación.



6.5.-Diseño e implementación

En la cabecera se optará por crear lo que se denomina un TabGroup, la forma de hacerlo mediante JavaScript es la siguiente:

```
var tabGroup = Titanium.UI.createTabGroup();
```

En ella añadiremos las siguientes pestañas, Usuarios: que contendrá información referente a cada miembro y los que están conectados, Proyectos: contiene información de los proyectos, horas y tareas.

La manera mediante la cual se han creado es la siguiente:

Primero creamos la ventana (*Windows*) sobre el cual se soportará el tab.

```
var win1 = Titanium.UI.createWindow({
  title:'Usuarios',
  backgroundColor:'#fff'
});
```

A continuación creamos el *Tab* al cual se enlazará el Windows.

```
var tab1 = Titanium.UI.createTab({
  icon:'KS_nav_views.png',
  title:'Usuarios',
  window:win1
});
```

El resultado final es el mostrado en la imagen inferior.



Dentro del tab Usuarios, existirán varios objetos, el principal será una rejilla con la información de usuarios, los otros dos serán botones que permitirán conmutar resultados en la información contenida.

Para crear la rejilla usaremos la siguiente expresión:

```
var ndata=[];
var Usuarios = Titanium.UI.createTableView({
  data:ndata,
  hasDetail: true,
  top:10,
  left:0,
  right:0,
  height:550,
  borderWidth:0,
  borderRadius:5,
  backgroundColor:'#eee',
  color:'#eee',
  borderColor:'#222'
});
```

```
win1.add(Usuarios);
```

Como se puede ver, en principio tenemos un array que contiene información de los usuarios, esta rejilla es añadida a la ventana correspondiente con el método *add*.

Una vez añadida, haciendo uso de los servicios webs rellenamos la información de los distintos usuarios. Invocaremos nada más arrancar al método *GetUsuarios()*, el cual iniciará la carga de la rejilla de usuarios.

```
soap.invoke('GetUsuarios', args, function(xmlDoc) {
    var ndata=[];
    var nombres =
xmlDoc.documentElement.getElementsByTagName('Nick');

    if (nombres && nombres.length>0) {
        for(var x=0;x<nombres.length;x++)
        {
            var bgcolor = (x % 2) === 0 ? 'transparent':
'#fff';

            var row= Ti.UI.createTableViewRow();
            row.width=300;
            row.borderWidth=1;
            row.borderColor='#222';
            //row.width=380;
            var rowLabel = Ti.UI.createLabel({
                font:{fontSize:17,fontWeight:'bold'},
                width:100,
                left:30,
                color:'#968331',
                textAlign:'left',
                top:13,
                height:22,
                text:nombres.item(x).text
            });

            row.add(rowLabel);

            row.backgroundColor=bgcolor;
            //row.font={fontSize:17, fontWeight:'bold'};
            //row.shadowColor='#FAFAFA';
            //row.shadowOffset={x:0, y:1};
            ndata[x] = row;

        }

        Usuarios.data=ndata;

    } else {
```

```

        alert('Oops, could not determine result of SOAP call.');
```

Como se puede ver, después de invocar al servicio web, creamos la *TableViewCell* correspondiente y añadimos los *labels* y otros campos, después enlazamos la rejilla al valor cargado. El resultado puede apreciarse en la siguiente imagen:

wi
alex
domingo
maría
ramon

Ahora, vamos a pasar a la segunda pestaña. En este caso hablaremos de la pestaña *Proyectos*, en este *Tab* se encuentra contenida una rejilla que mantiene la información tanto de este como de las tareas. Hemos fijado un evento de *click* para que en el caso de pulsar sobre la pestaña se active la carga de proyectos.

```
tab2.addEventListener('click', Cargar_Proyectos)
```

El método invocado cargará la información de los proyectos, la carga se efectuará en cada fila con un la opción prefijada de *“hasChild en true”*. Con ello conseguimos el efecto de poder entrar en distintos niveles, nos servirá para adentrarnos en las tareas de cada proyecto y en los detalles de cada una de estas.

Desarrollo Software Android Empresa: Msms	▼
Implantación ERP Empresa: Farmicon	▼
Desarrollo aplicacion Iphone: Ership	▼
Creación Web corporativa: Matutano,S.L	▼
Implantación Intranet: Overtel	▼

Se establece el evento clic en la rejilla tareas, al pulsar haremos la carga de las tareas para ese proyecto. Estos eventos consecutivos predicen el tipo de carga, ya sea de proyecto, tareas o detalles de cada una.

```
Tareas.addEventListener('click', function(e)
{.....});
```

La función aparece tomando el argumento "e", en este nos llegará el objeto fila pulsada, ahora simplemente apuntamos a él y obtenemos el valor de la fila. Lo haremos de la siguiente forma:

```
var index = e.index;
var section = e.section;
var row = e.row;
var rowdata = e.rowData;
```

Los detalles obtenidos pueden servirnos para crear una vista en la cual iremos presentando la información de este registros, sirve como ejemplo el siguiente label que contiene la información del estado del proyecto, nótese que el estado se contiene en el objeto row, una simple if de doble condición mostrará el resultado:

```
var Estados=Titanium.UI.createLabel({
width:450,
text: row.estado=='A'?'Estado: Abierto':'Estado:
Cerrado',
top: 50,
left: 120,
color: 'blue',
font:{fontSize:20, fontWeight:'bold'}
});
```


A resaltar, *el progressbar*, una barra de progreso que nos indicará el balance actual en horas de la tarea seleccionada. Este objeto nos indica la cantidad de horas avanzadas en el desarrollo junto a las que están por finalizar.



8. Conclusiones

8.1. - Conclusiones

Los objetivos del proyecto han sido completados de forma satisfactoria. Como resultado de todo esto tenemos un conjunto de aplicaciones que permiten de manera eficaz monitorizar los desarrollos dentro de cualquier empresa.

El objetivo básico de este proyecto como se ha mencionado anteriormente era el desarrollo de un sistema que permitiera registrar los tiempos de implicación en tareas comprendidas en un proyecto así como el control de asistencia, incidencias resultantes y por último la posterior visualización de los resultados de esta monitorización.

Además de todos los requisitos funcionales, nuestra aplicación ha alcanzado también los no funcionales, esto la hace intuitiva, fácil, segura y rápida.

La escalabilidad ha sido un punto muy a tener en cuenta en nuestro desarrollo, gracias a su estructura en capas es relativamente sencillo hacer modificaciones y ampliaciones.

Como complementos de LupApp hemos integrado tanto la aplicación móvil como el uso de la clase Skeleton para controlar de manera automática la postura de los empleados:

Aplicación móvil

El control remoto de la aplicación a nivel de visualización de resultados se ha conseguido gracias al desarrollo del software para Android. Este último ha permitido que la movilidad no sea un impedimento a la hora de aunar la información de los proyectos y usuarios.

Interface con Kinect

El resultado con Kinect además de resolutivo ha sido sorprendente en términos visuales. Permite reflejar con gran resolución tanto las posturas de usuario como las pausas que se han llevado a cabo dentro de su horario laboral.

Desde mi punto de vista personal, se han cumplido todas las expectativas puestas, ya que he podido profundizar en el desarrollo con la plataforma .net, aprender a programar en C#, SqlServer, así como manejar el desarrollo en plataformas móviles con Titanium Studio.

8.2.-Líneas futuras

Dejamos abierto para ampliaciones futuras algunas opciones que pueden resultar interesantes, varias de las cuales citaremos a continuación:

Aplicación escritorio LupaApp

Hemos dejado como una opción futura la emigración de la aplicación de winform a WPF. Somos conscientes de que requiere de muchos cambios, aunque gran parte de la lógica que se encuentra en la capa de negocio permitiría hacerlo de manera relativamente rápida. Tenemos en cuenta este cambio como un planteamiento para conseguir implementar de manera más real el patrón MVC. Esto haría mucho menos costoso la creación de vistas que incluso tendrían un aspecto más vistoso y atractivo, incluyendo animaciones que resalten determinados resultados.

Aplicación móvil:

Se nos ocurren varias ampliaciones futuras, desde agregar más funcionalidad y autonomía a este hasta desarrollar tanto para IOS como para otras plataformas futuras. Android en su versión 4.0 mejora ostensiblemente la apariencia y aspecto del front, lo cual redundará en conseguir mejoras en términos de usabilidad. También tenemos otros aspectos funcionales para la aplicación móvil, tales como la comunicación directa con los distintos empleados mediante Chat o mensajes que serán enviados a los puestos de

trabajo. Destacar igualmente, que la venidera entrada de tablets y móviles con Windows 8 hará sencilla su instalación del aplicativo de escritorio en estos dispositivos con las ventajas que esto implica.

Kinect

Hasta el momento, hemos conseguido hacer una pequeña implementación de lo que sería el seguimiento postural con el dispositivo de Microsoft. Gracias al uso de la clase Skeleton, se ha conseguido de forma relativamente sencilla monitorizar la posición de usuario sentado y en pie. Sin embargo, parte del objetivo futuro de nuestro desarrollo está centrado en mejorar el algoritmo a fin de poder no solamente delimitar el estado binario (en pie y sentado), buscamos además realizar un seguimiento más completo de la trayectoria del trabajador, indicando además del estado binario otras como la higiene postural dentro del horario laboral.

Bibliografía

- **Visual C#:**

Microsoft C#. Curso de programación/ Fco Javier Ceballos.
Manual de referencia C# / Herbert Schildt.

- **Base de datos:**

SqlServer 2008 paso a paso, Mike Hotek.

- **Ingeniería del Software:**

Análisis y diseño detallado de aplicaciones informáticas de gestión: desarrollo de aplicaciones informáticas, Sara M^a García Blanco, Enrique Morales Ramos.

- **Links interesantes:**

<http://es.wikipedia.org>

<http://programacion.com>

<http://www.csharp-help.com>

<http://www.csharpfr.com/Default.aspx>

<http://docs.appcelerator.com/titanium/2.0/index.html>

<http://www.codeproject.com>

<http://msdn.microsoft.com>

<http://www.elguille.info>

Anexo 1: Manual usuario aplicación escritorio

1.- Autenticación

Al iniciar la ejecución del sistema, este siempre requerirá el nombre del usuario y la contraseña de la persona que desea utilizar la aplicación, de este modo podrá cargar o no algunas funciones del programa dependiendo de los permisos del usuario.



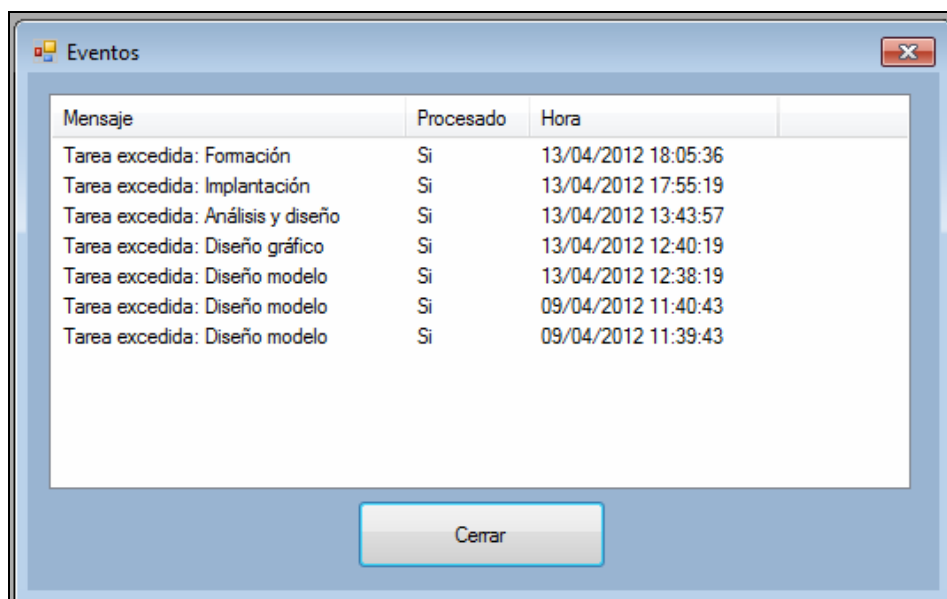
Tras introducir el nombre de usuario y la contraseña debemos pulsar el botón aceptar, y si los datos son correctos entraremos en el programa, en caso contrario, el sistema mostrara un mensaje de error.

2.- Menú principal

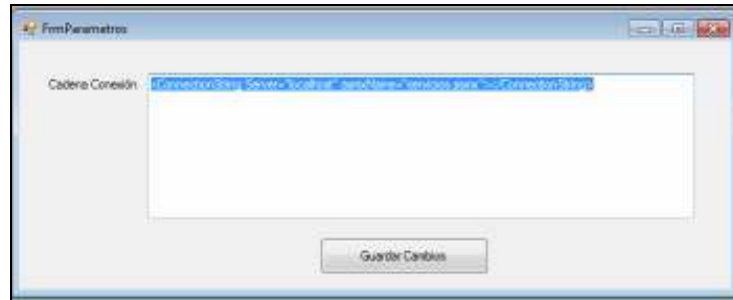
Siguiendo con la ejecución de la aplicación el usuario se encontrará con el menú principal de la aplicación:



Nada más entrar en el menú resalta un aviso de las tareas excedidas en el cuadro, para ponernos al día de las tareas que se han excedido.

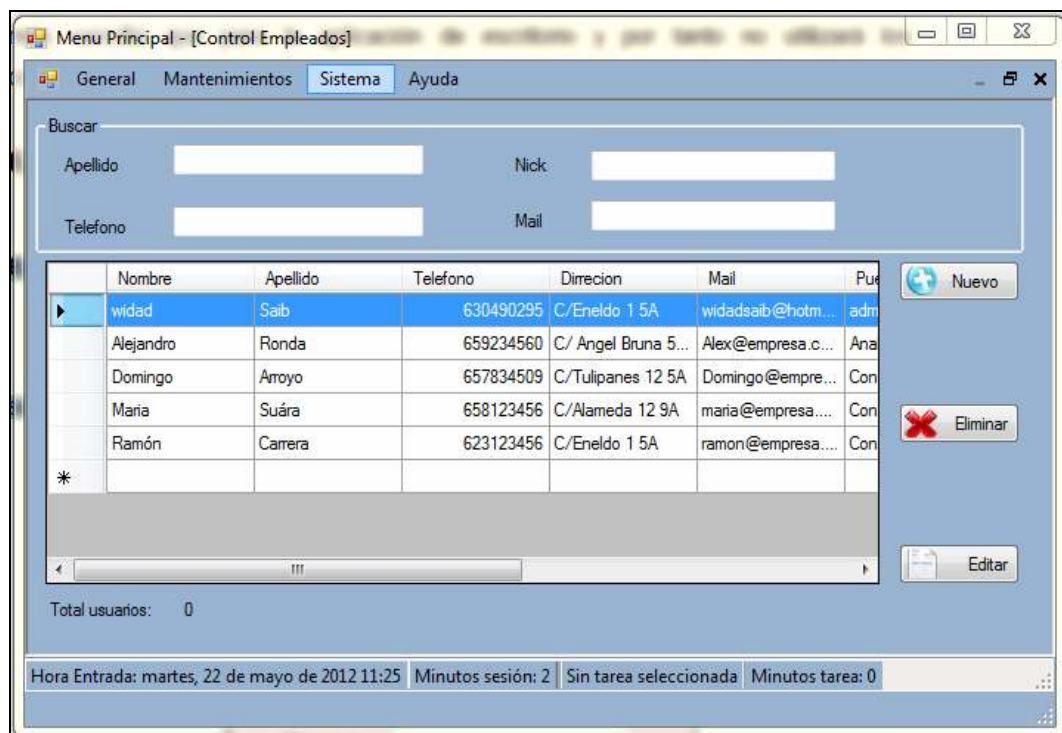


Si pulsamos la pestaña de parámetro en el menú general me sale este formulario que tiene la finalidad de poder cambiar la cadena de conexión manualmente. Solamente debemos insertar el nombre o dirección del servidor. Cuando le damos a guardar cambios se cambia automáticamente la cadena de conexión en el archivo XML.



Si pulsamos la pestaña mantenimientos nos encontramos con los diferentes mantenimientos del sistema como puede ser (alta, baja, editar) tanto empleados, proyectos como tareas.

Todos estos formularios constan de varios filtros además de un gridview donde se cargan los datos que necesitamos así como botones.



Alta empleado

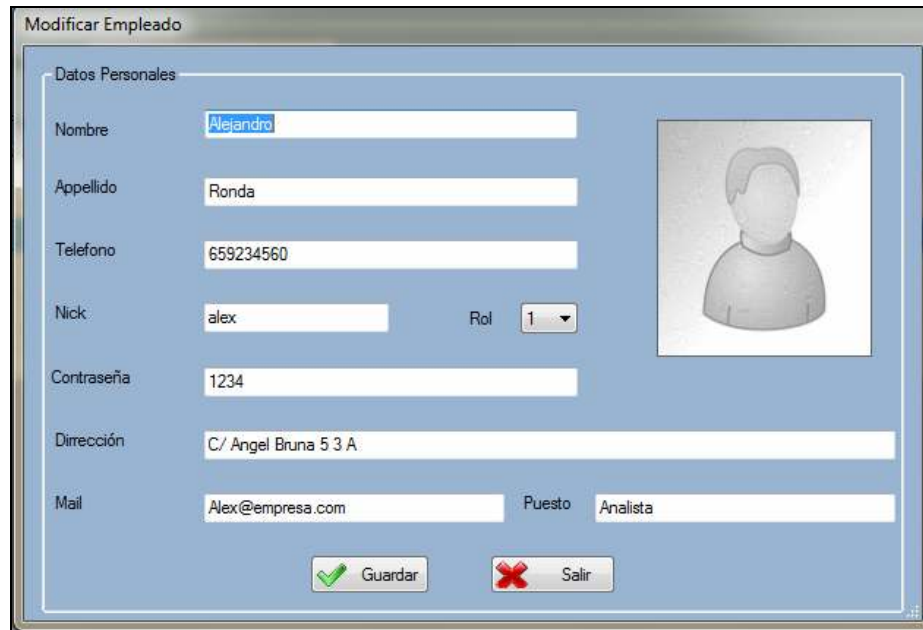
Si desea dar de alta a un nuevo empleado, se debería pulsar en el botón nuevo, situado a la derecha de la pantalla. Luego, se rellena el formulario de nuevo empleado con los datos correspondientes, y se pulsa el botón guardar. Tras pulsar este botón saldrá un mensaje de confirmación del alta o bien un mensaje de error, si algún dato introducido no es correcto.



The image shows a web form titled "FrmAltaCliente" with a light blue background. The form is organized into a "Datos Personales" section. It contains several input fields: "Nombre", "Apellido", "Telefono", "Nick", "Contraseña", "Dirección", "Mail", and "Puesto". The "Nick" field has a red error message "Nick ya usado" below it. The "Rol" field is a dropdown menu. To the right of the form is a placeholder for a profile picture, showing a generic person icon. At the bottom of the form, there are two buttons: "Guardar" with a green checkmark icon and "Salir" with a red X icon.

Editar empleado

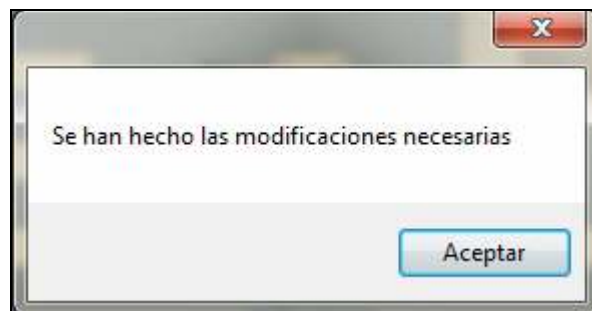
Si queremos modificar algún empleado ya registrado en el sistema, en primer lugar, seleccionamos el empleado de la lista, le damos al botón editar, con esto se nos cargara el formulario correspondiente a este evento, con los datos de dicho empleado.



The screenshot shows a web form titled "Modificar Empleado" with the following fields and controls:

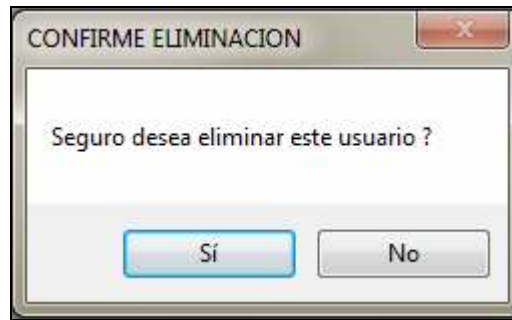
- Datos Personales** (Section Header)
- Nombre:** Text input containing "Alejandro".
- Apellido:** Text input containing "Ronda".
- Telefono:** Text input containing "659234560".
- Nick:** Text input containing "alex".
- Rol:** Dropdown menu showing "1".
- Contraseña:** Text input containing "1234".
- Dirección:** Text input containing "C/ Angel Bruna 5 3 A".
- Mail:** Text input containing "Alex@empresa.com".
- Puesto:** Text input containing "Analista".
- Buttons:** "Guardar" (with a green checkmark icon) and "Salir" (with a red X icon).
- Image:** A placeholder image of a person's head and shoulders.

Se hace los cambios necesarios y se guarda. Si todo ha ido correctamente, el sistema nos mostrará un mensaje de confirmación.



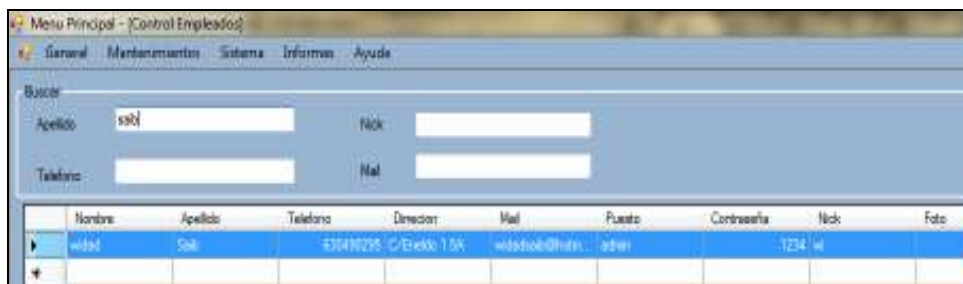
Baja empleado

Con el botón eliminar, damos de baja a los usuarios, simplemente seleccionando un empleado de la tabla. Tras pulsar dicho botón, el sistema nos pedirá confirmación de la baja.



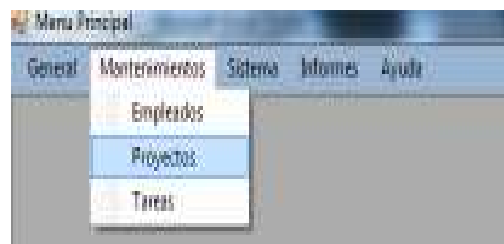
Búsqueda empleado

En caso que necesite buscar un empleado concreto ya registrado en el sistema, se puede utilizar los filtros que aparecen en la parte superior del formulario. Se introducen los datos del empleado que conozcamos. A medida que vamos introduciendo datos, en la lista de empleado abajo, se van mostrando los empleados cuyos datos coinciden con lo que hemos introducido.

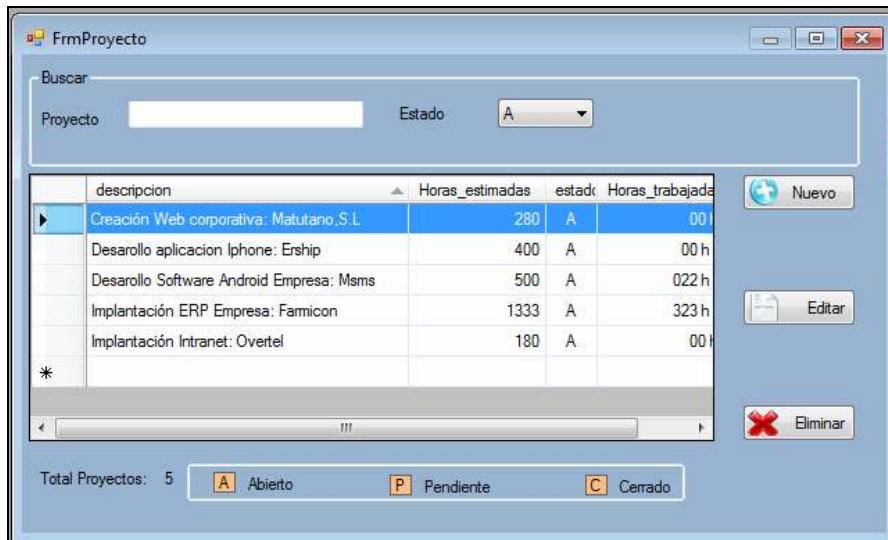


2.2.- Proyectos

Seleccionamos del menú principal la pestaña proyectos.

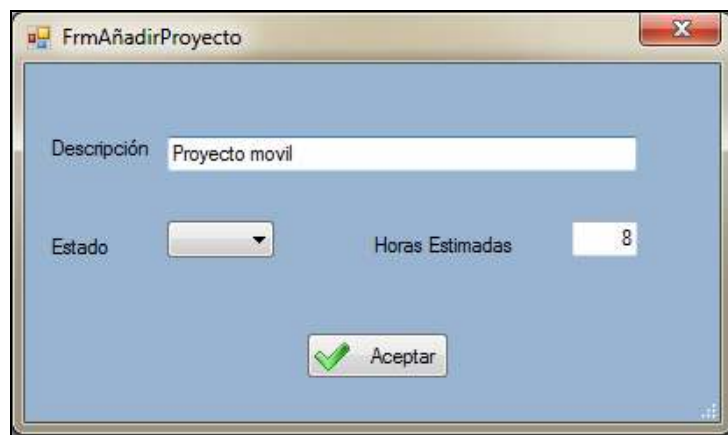


El formulario de proyecto es la siguiente.

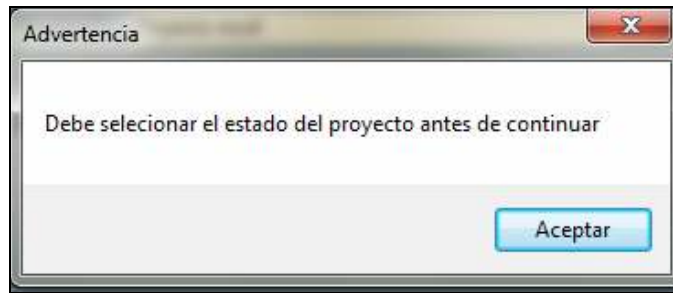


Alta Proyecto

Si desea dar de alta a un nuevo proyecto, se debería pulsar en el botón nuevo, situado a la derecha de la pantalla. Luego, se rellena el formulario de nuevo proyecto con los datos correspondientes, y se pulsa el botón aceptar.

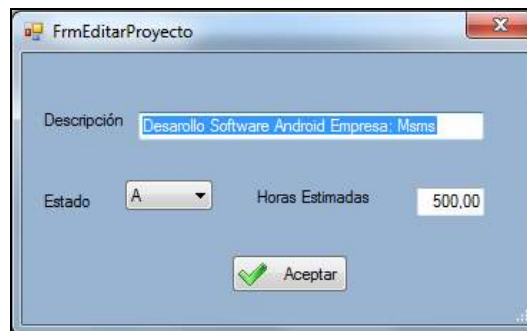


Tras pulsar este botón saldrá un mensaje de confirmación del alta o bien un mensaje de error, si algún dato introducido no es correcto.

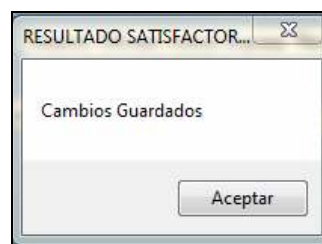


Editar proyecto

Si queremos modificar algún proyecto ya registrado en el sistema, en primer lugar, seleccionamos el proyecto de la lista, le damos al botón editar, con esto se nos cargara el formulario correspondiente a este evento, con los datos de dicho proyecto. Se hace los cambios necesarios y se guarda.

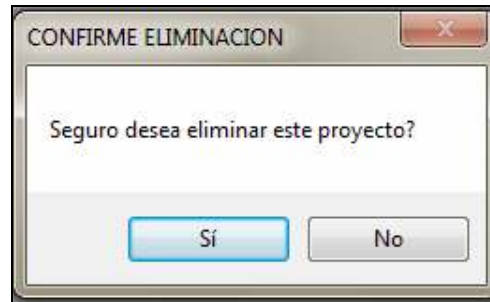


Si todo ha ido correctamente, el sistema nos mostrará un mensaje de confirmación.



Eliminar proyecto

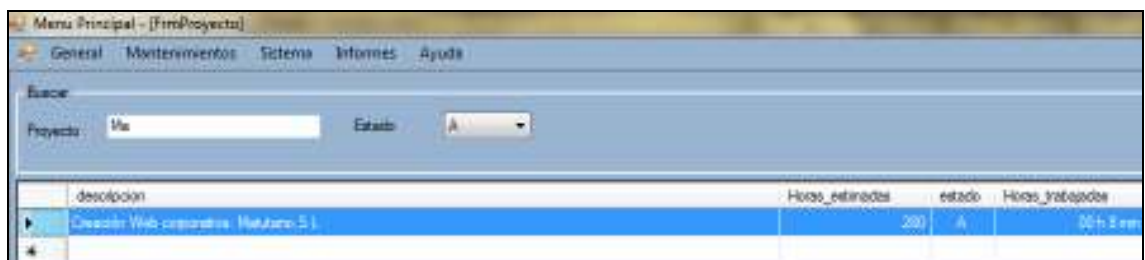
Con el botón eliminar, eliminar el proyecto, simplemente debemos seleccionar un proyecto de la tabla. Tras pulsar dicho botón, el sistema nos pedirá confirmación de la baja.



Búsqueda proyecto

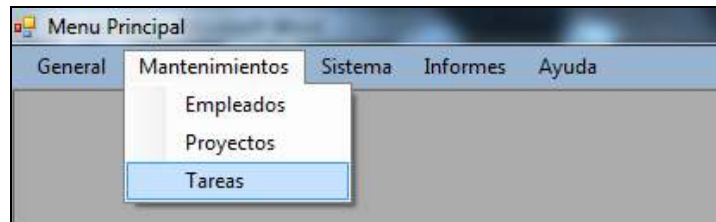
En caso que necesite buscar un proyecto concreto ya registrado en el sistema, se pueden utilizar los filtros que aparecen en la parte superior del formulario. Se introducen los datos del proyecto que conozcamos. A medida que vamos introduciendo datos, en la lista de proyectos abajo, se van mostrando los proyectos cuyos datos coinciden con lo que hemos introducido.

Hemos utilizado, la descripción del proyecto y su estado para este filtro.



2.3.- Tareas

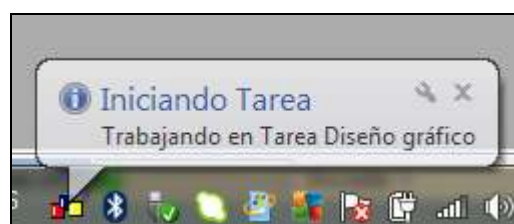
Este formulario se considera de los más importantes de la aplicación, al seleccionar tarea en la pestaña del menú.



Para empezar a trabajar en cualquier proyecto necesitamos seleccionar una tarea de la tabla de tareas.

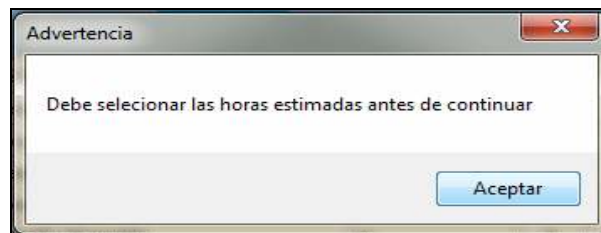
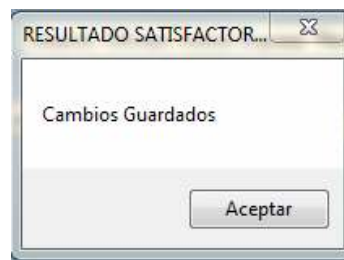
Proyecto	Descripcion	Pendiente	Asignado	Hora_inicio	Hora_termina
Desarrollo Software Web para Empresa: Bazar	Diseño gráfico	si	A	00:00	00:00
Desarrollo Software Web para Empresa: Bazar	Diseño web/interfaz	si	A	00:00	00:00
Desarrollo Software Web para Empresa: Bazar	Diseño web/interfaz	si	A	00:00	00:00
Implementación ERP Empresa: Comercio	Instalar y diseñar	si	A	00:00	00:00
Implementación ERP Empresa: Comercio	Diseño web/interfaz	si	A	00:00	00:00
Implementación ERP Empresa: Comercio	Implementación	si	A	00:00	00:00
Implementación ERP Empresa: Comercio	Implementación	si	A	00:00	00:00
Desarrollo Software Web para Empresa: Bazar	Prueba: Analizar requisitos	si	A	00:00	00:00
Desarrollo Software Web para Empresa: Bazar	Prueba: Diseño gráfico y mantenimiento	si	A	00:00	00:00
Desarrollo Software Web para Empresa: Bazar	Prueba: Diseño del modelo	si	A	00:00	00:00
Desarrollo Software Web para Empresa: Bazar	Prueba: Programación de la aplicación	si	A	00:00	00:00
Desarrollo Software Web para Empresa: Bazar	Prueba: Testeo de integración	si	A	00:00	00:00
Implementación Internet: Comercio	Diseño: Analizar y requerimientos	si	A	00:00	00:00
Implementación Internet: Comercio	Diseño: Diseño del modelo	si	A	00:00	00:00
Implementación Internet: Comercio	Diseño: Programación de la aplicación	si	A	00:00	00:00
Desarrollo Web corporativo: Naturano S.L.	Naturano: Desarrollo	si	A	00:00	00:00
Desarrollo Web corporativo: Naturano S.L.	Naturano: Desarrollo	si	A	00:00	00:00
Desarrollo Web corporativo: Naturano S.L.	Naturano: Diseño gráfico	si	A	00:00	00:00
Desarrollo Web corporativo: Naturano S.L.	Naturano: Diseño gráfico	si	A	00:00	00:00
Desarrollo Web corporativo: Naturano S.L.	Naturano: Pruebas de integración	si	A	00:00	00:00

Tras esta acción, saltará un globo emergente para informarnos de la tarea escogida.





Tras pulsar este botón saldrá un mensaje de confirmación del alta o bien un mensaje de error, si algún dato introducido no es correcto.



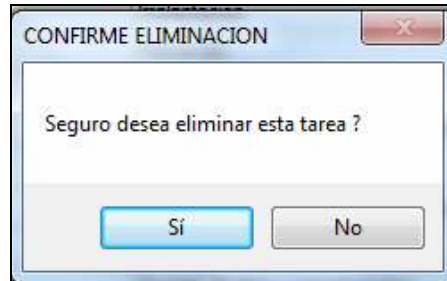
Editar tarea

Si queremos modificar alguna tarea ya registrado en el sistema, en primer lugar, seleccionamos la tarea de la tabla, le damos al botón editar, con esto se me cargara el formulario correspondiente a este evento, con los datos de dicha tarea. Se hace los cambios necesarios y se guarda. Si todo ha ido correctamente, el sistema nos mostrará un mensaje de confirmación.

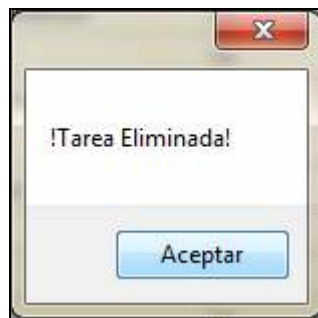


Eliminar tarea

Con el botón eliminar tarea, simplemente debemos seleccionar una tarea de la tabla. Tras pulsar dicho botón, el sistema nos pedirá confirmación de la baja.



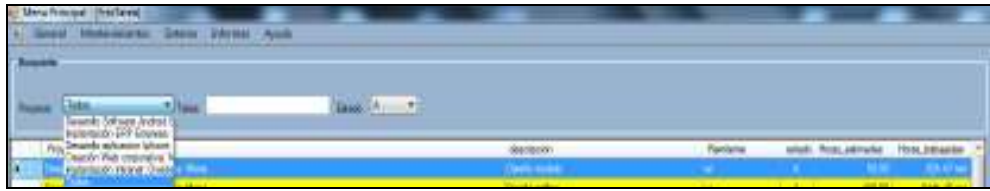
Cuando confirmo la acción, vuelve a salir un cuadro de mensaje, confirmando que se ha eliminado la tarea.



Búsqueda tarea

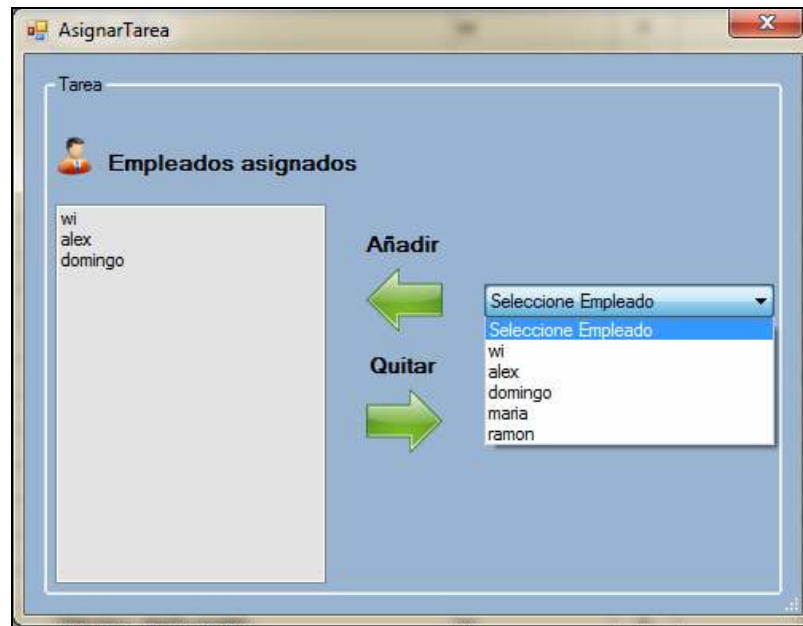
En caso que necesite buscar una tarea concreta ya registrada en el sistema, se puede utilizar los filtros que aparecen en la parte superior del formulario. Se introducen los datos de la tarea que conozcamos. A medida que vamos introduciendo datos, en la lista de tareas abajo, se van mostrando las tareas cuyos datos coinciden con lo que hemos introducido.

Hemos utilizado, la descripción de los proyectos, su estado así como la descripción de la tarea para este filtro.

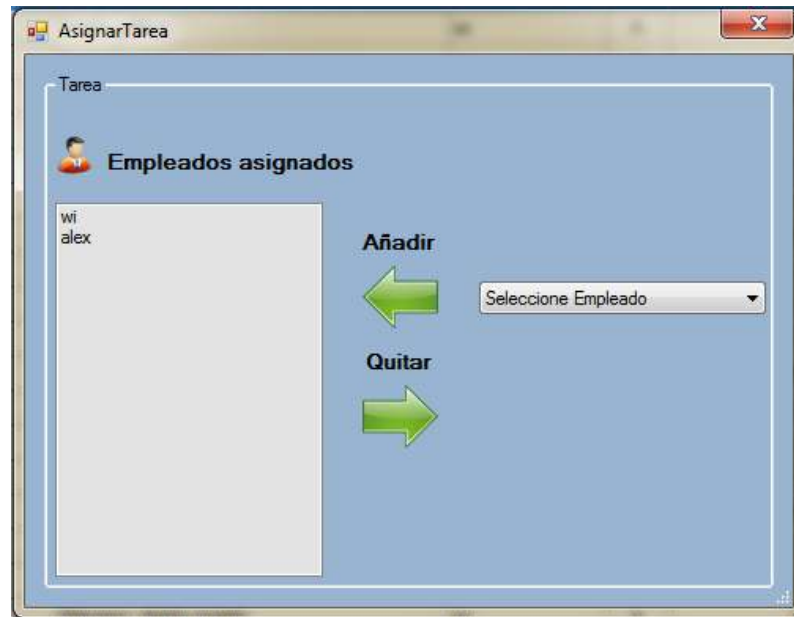


Asignar tareas

Lo que se pretende con este formulario es ir asignando tareas a usuarios. En el textbox de la derecha seleccionamos el empleado que necesitamos para ir dando el derecho en participar en las tareas de cierto proyecto. De esta manera, cuando el empleado abre su sección se le indica las tareas añadidas.

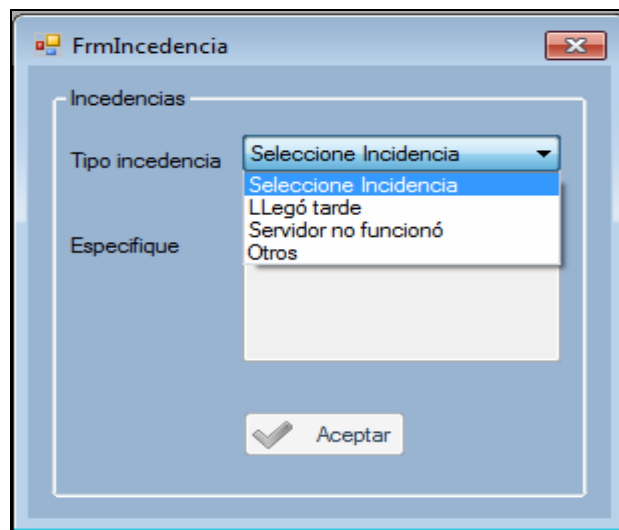


De la misma manera se puede ir quitando los empleados de cualquiera tarea con la flecha de quitar.



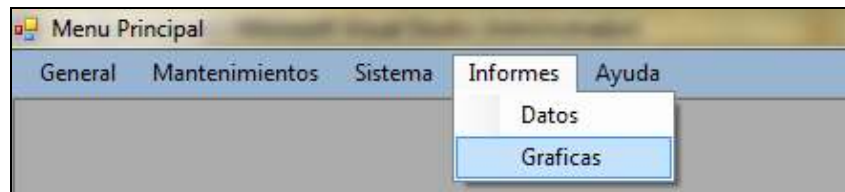
2.4.-Incidencia

El menú de incidencia, le da al empleado la posibilidad de ir anotando las incidencias que le van surgiendo a lo largo de su jornada. Hay algunas predeterminadas pero también dejamos la posibilidad al empleado de escribir su propia incidencia.



2.5.- Informes

El menú informes se ha hecho de manera que nos pueda presentar tanto dato como gráficas.



Si seleccionamos el menú de datos, obtendremos todos los datos referentes a cualquier proyecto.

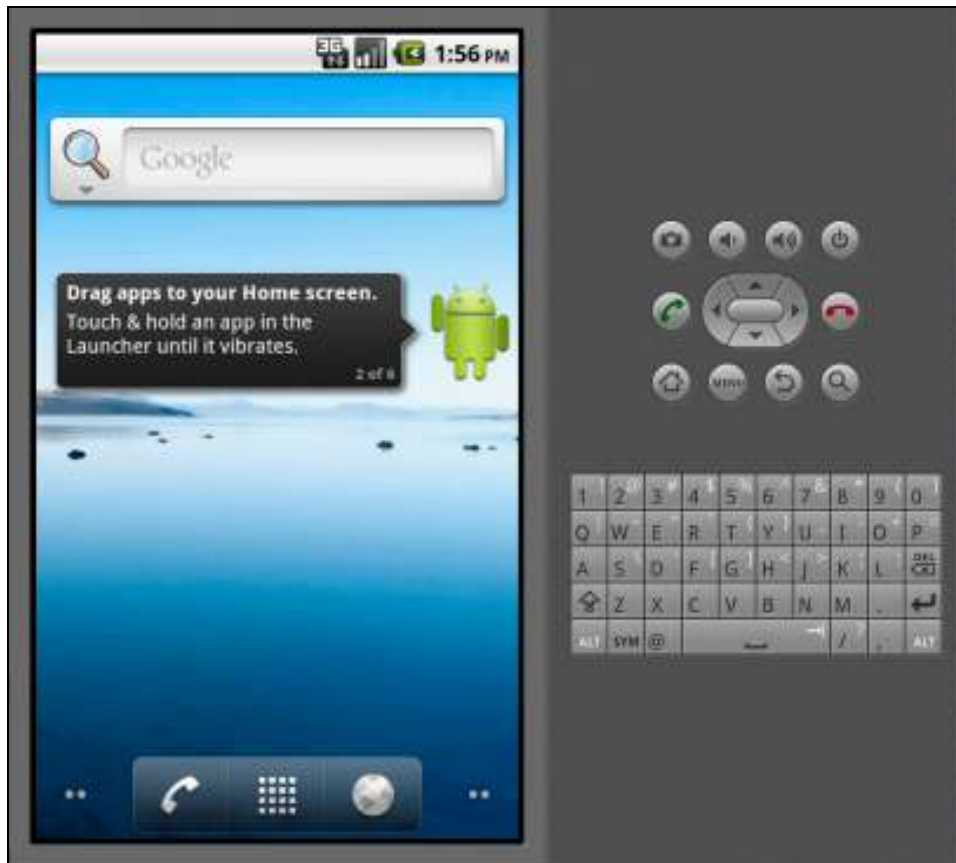
Proyecto	Tarea	Usuario	Hora Entrada	Hora Salida
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 21:54:34	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 21:55:30	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 21:56:18	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 21:56:39	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 21:57:22	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 21:57:36	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 21:57:49	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 21:58:34	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:08:11	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:09:08	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:09:15	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:09:25	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:09:46	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:09:53	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:13:54	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:14:12	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:14:18	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:14:22	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	18/05/2012 22:14:28	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	24/05/2012 18:41:45	04/06/2012 16:47:40
Desarrollo aplicacion Iphone: Enshp	Enshp: Análisis aplicacón	WI	24/05/2012 19:10:57	04/06/2012 16:47:40

Y si seleccionamos el menú referente a grafica, obtenemos una manera visual la mayoría de las consultas interesante que se pueden necesitar.

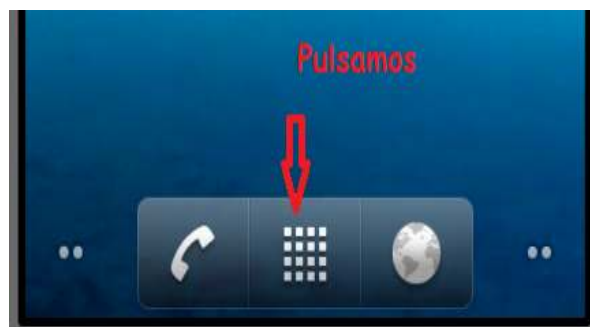


Anexo II: Aplicación móvil

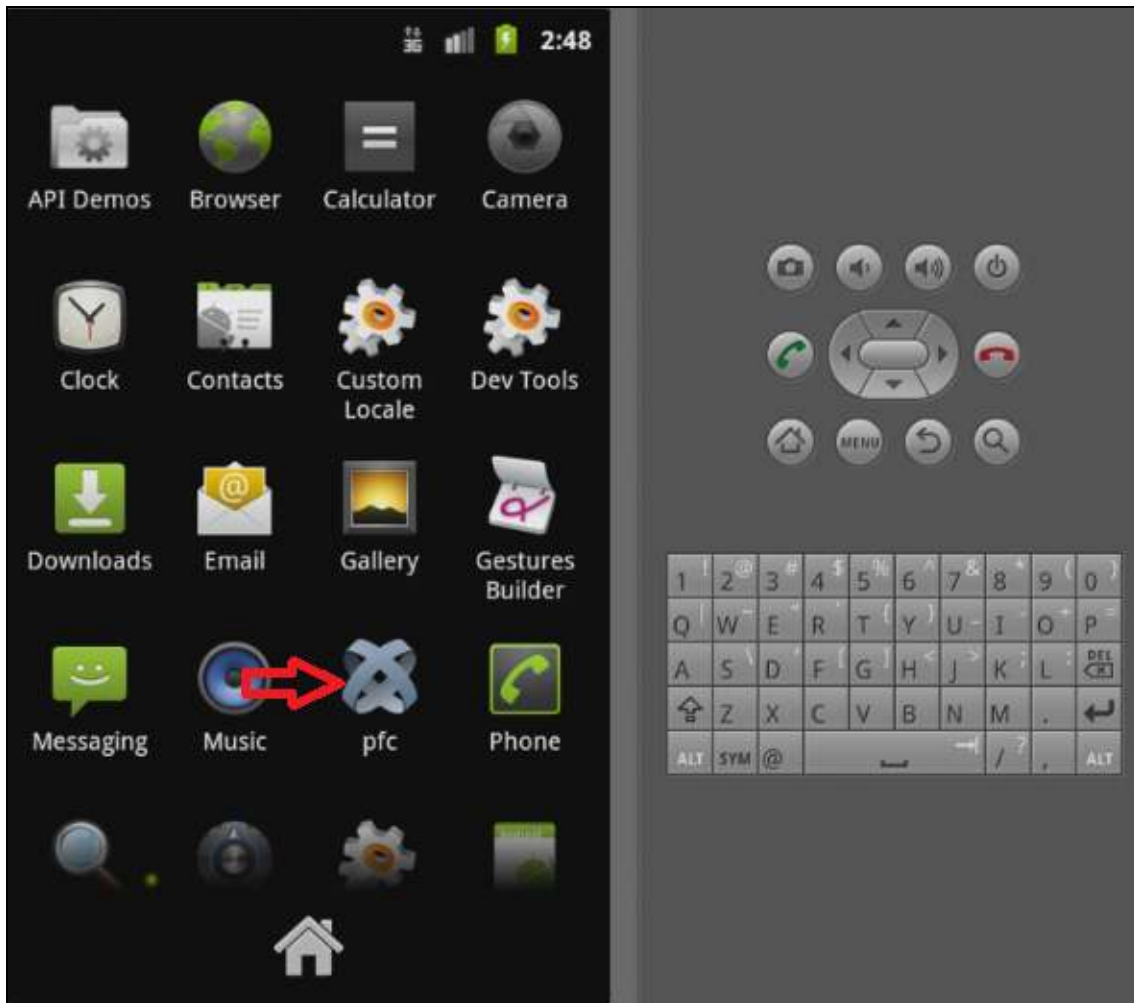
En este apartado se va a explicar el manejo y uso de la aplicación Móvil, en general el uso de esta, es muy sencillo y intuitivo. Solo los administradores tienen este sistema en su tablet.



Pasamos el dedo por donde esta señalado para acceder al menú principal:



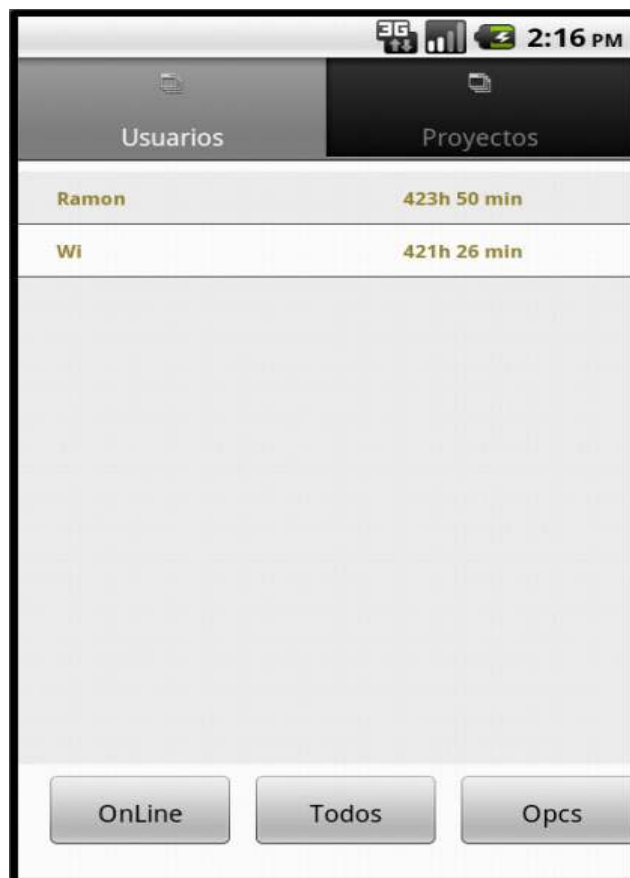
Una vez aquí seleccionamos el icono de LupApp.



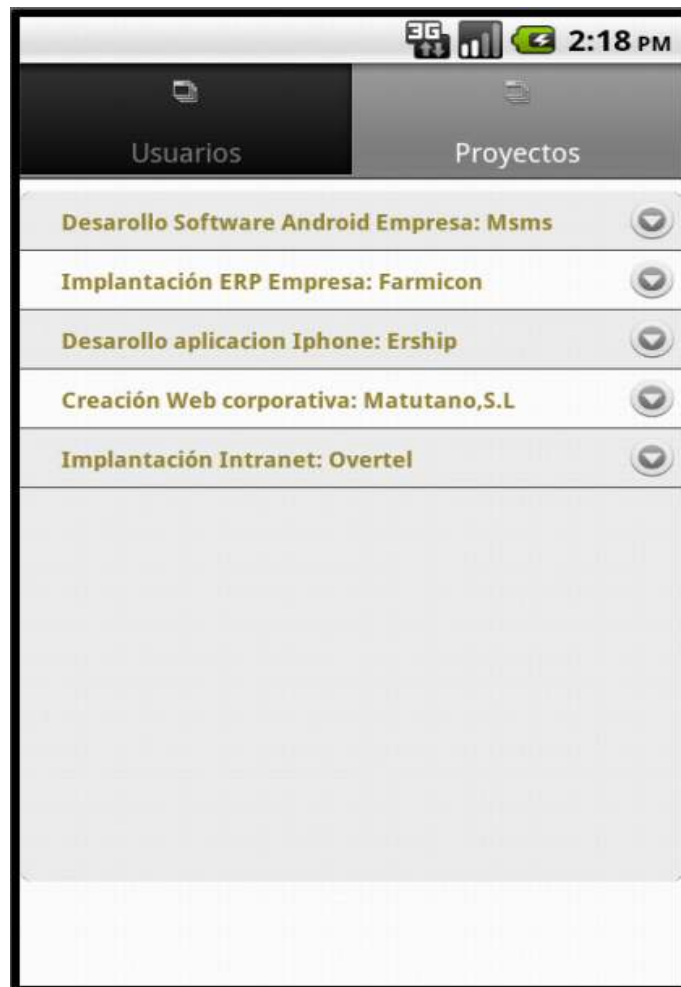
Una vez dentro de la aplicación, observaremos que la ventana esta formada por dos campos:

- *Usuarios.*
- *Proyectos.*

Si seleccionamos Usuarios tenemos la opción de ver todos los usuarios, o solo los conectados.



Si seleccionamos la pestaña de proyectos, me lista todos los proyectos:



Podemos ir navegando en cada proyectos, seleccionando la flecha que tenemos en el lateral derecha, con esto entramos en los detalles de cada tareas de cada proyecto.



La información más relevante que obtendremos es:

- Estado del proyecto.
- Horas estimadas.
- Horas trabajadas.
- Remitente.
- Barra de progreso de la tarea.



