MEMORIA FINAL

iNotHere

Proyecto Final de Carrera .NET

Jose Antonio Erustes Lobato

2011-2012 2C

Consultor: David Riu Herraiz

A mi mujer.
Sin ella, este trabajo y esta carrera
hubieran sido imposibles

Índice

1. Introducción.

1.1 Justificación y objetivos del proyecto.

Una de las actividades más complejas para un centro docente es la organización interna, y a su vez, el horario general del centro. Una buena organización y un horario general coherente hacen que el funcionamiento del Centro sea mucho más lógico y genere menos problemas. Respecto al funcionamiento diario existen según mi opinión, algunos aspectos muy mejorables y uno de ellos es la gestión de ausencias del profesorado. Un centro con unos 100 profesores debería tener un sistema sencillo de control de ausencias, de información y difusión de las mismas.

Partiendo de esta problemática inicial, y teniendo en cuenta que el profesorado puede faltar a sus clases por diferentes motivos como son:

- Por salidas extraescolares o complementarias.
- Por motivos previstos con anterioridad, debidamente justificados.
- Por motivos imprevistos debidamente justificados.
- Por cualquier motivo sin justificar.

En el diseño únicamente tendremos en cuenta los tres primeros casos, ya que el cuarto se refiere a faltas injustificadas y deben tratarse mediante los instrumentos sancionadores oficiales.

Por tanto es necesario poner a disposición del profesorado una herramienta para solicitar permiso, cuando tiene previsto faltar por un motivo justificado y para solicitar que se justifiquen sus ausencias, una vez ya han tenido lugar.

Al mismo tiempo, el jefe de estudios es la persona que debe llevar el control de todas las ausencias, cuantificarlas y comunicarlas al departamento de inspección educativa para su revisión y evaluación. El control de estas ausencias se hace de manera diaria ya que antes de empezar las sesiones lectivas con los alumnos, el jefe de estudios debe comunicar al profesorado las posibles ausencias (por los motivos que fueren). Actualmente este proceso se lleva a cabo de manera manual: el jefe de estudios consulta el profesorado que ha solicitado permiso, consulta el horario, escribe en cada casilla horaria si es el caso el grupo que va a faltar y un comentario. Al mismo tiempo, si algún profesor se pone enfermo y comunica telefónicamente que va a faltar se debe apuntar siguiendo el proceso anterior. Además, si hay alguna salida extraescolar el profesor que acompaña a los alumnos debe añadirse a la hoja de ausencias (aunque esta ausencia estará justificada directamente).

Como se puede observar el proceso es lento y poco eficiente y dado que la carga de trabajo a la hora de entrada es alta, provoca que el jefe de estudios deba desatender otros asuntos.

Buscando funcionalidades similares en documentación y empresas que se dedican al desarrollo de software específico para centros docentes podemos encontrar algunos ejemplos interesantes pero que no se adaptan a la situación que necesitamos resolver. Algún software es capaz de registrar las faltas del profesorado y poner un profesor

substituto (Untis 2012[1]) pero no queremos esa funcionalidad, y además no es capaz de generar un informe diario con el formato adecuado. Otros como *Kronowin o Gestión Horarios Centros Docentes* o *Horarios*, ni siquiera permiten la gestión de ausencias, ya que su principal función es la generación de horarios.

En definitiva la presencia en el mercado de software que gestiona horarios es relativamente abundante pero ninguno permite responder a nuestras necesidades.

Debido a todo ello, es el objetivo de este Trabajo Final de Carrera el poner a disposición del profesorado y del jefe de estudios, una herramienta que permita automatizar en la medida de lo posible las gestiones que hoy en día se hacen en papel y de manera manual, como se ha explicado antes.

1.2. Objetivo del proyecto.

El objetivo del proyecto es diseñar una solución que permita dos funciones principales:

- Gestionar las ausencias del profesorado.
- Gestionar las peticiones de permiso.

Si bien esta es una simplificación bastante importante, da una idea principal de que la solución está dividida en dos proyectos diferenciados. Esos dos proyectos coinciden con funcionalidades destinadas a los administradores y funcionalidades destinadas a los usuarios. Concretando un poco los objetivos del proyecto:

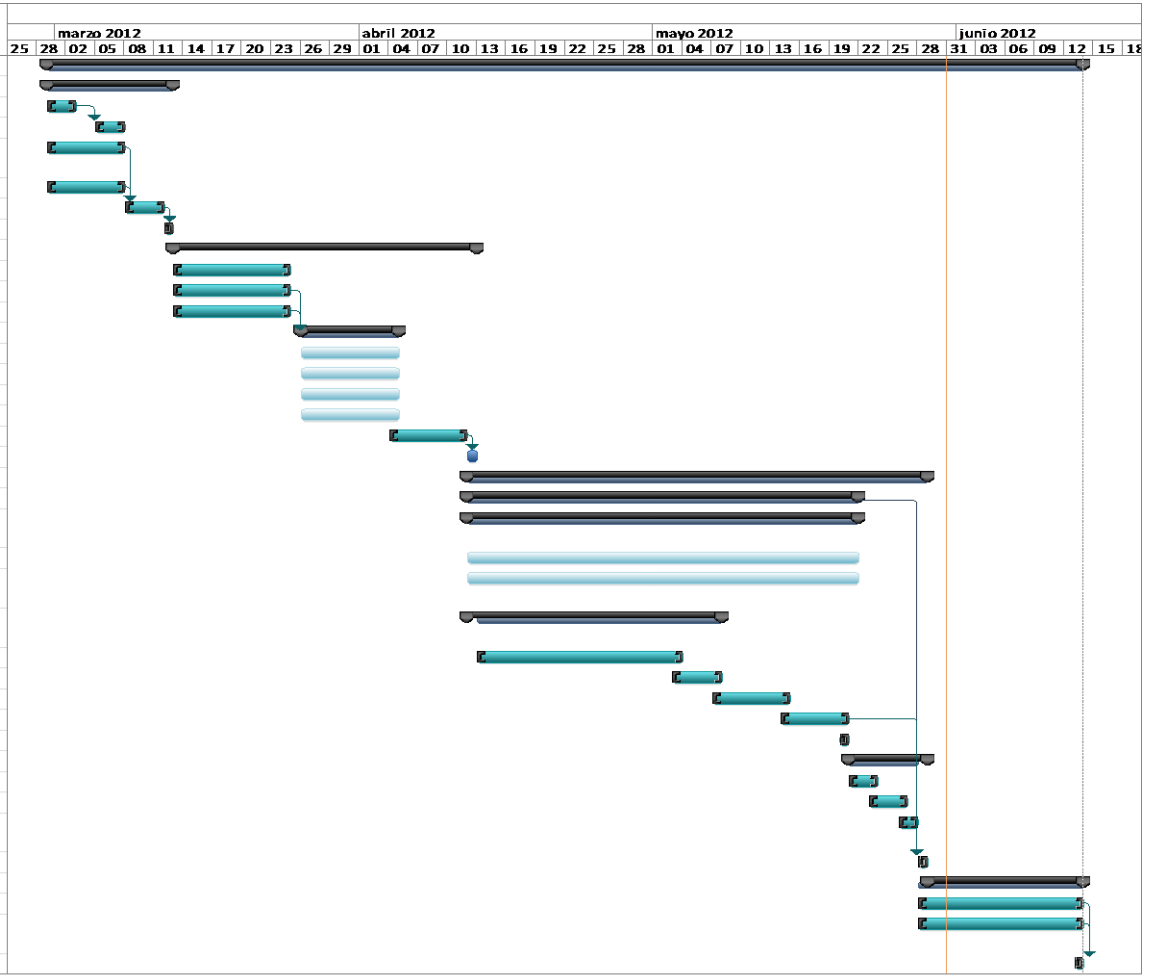
- Diseñar una aplicación para gestionar las faltas (añadir, modificar, eliminar) del profesorado. Además debe ser capaz de marcar faltas como justificadas o no justificadas.
- Diseñar un informe diario para publicitar las faltas de asistencia diarias.
- Diseñar formularios capaces de modificar los datos registrados.
- Diseñar una interfaz que permita importar datos desde archivos de texto producidos por otra aplicación.
- Diseñar una aplicación que permita solicitar permisos y justificación de faltas de asistencia. Además debe permitir al usuario poder cambiar datos imprescindibles (clave de acceso…).

Por tanto se va a diseñar una solución compuesta por dos aplicaciones de escritorio: una para el administrador y otra para los usuarios con acceso por contraseña.

[1] Untis 2012 se pude encontrar en http://www.grupet.at

2. Planificación Inicial y real.

Id	Modo de tarea	Nombre de tarea	Duración
1		**TFC .NET - iNotHere**	**76 días**
2		**Fase 1. Plan de Trabajo**	**9 días**
3		Descarga del software	3 días
4		Elección del proyecto	3 días
5		Estudio de funcionalidades requeridas	6 días
6		Estudio productos similares	6 días
7		Redacción del plan de trabajo	3 días
8		Entrega PAC1	1 día
9		**Fase 2. Especificación y diseño**	**23 días**
10		Estudio de alternativas posibles	10 días
11		Diseño de base de datos	10 días
12		Diseño de la interface carga de dat	10 días
13		**Especificación aplicación**	**8 días**
14		Diagramas de casos de uso	8 días
15		Diagrama de clases	8 días
16		Diagrama E/R	8 días
17		Diseño de interfaces	8 días
18		Escribir PAC2	6 días
19		Entrega PAC2	1 día
20		**Fase 3. Implementación de la Aplica**	**33 días**
21		**Implementación .NET**	**28 días**
22		**Implementación del servicio WCF/Acceso BBDD**	**28 días**
23		Implementación métodos	28 días
24		Implementación clase tableFaltas	28 días
25		**Implementación proyecto administración**	**18 días**
26		Implementación de formulari	15 días
27		Implementación de informes	3 días
28		Implementación proyecto usuario	6 días
29		Pruebas y despuración	**6 días**
30		Primera versión	1 día
31		**Documentación**	**6 días**
32		Manual de usuario	3 días
33		Manual de instalación	4 días
34		Aspectos Interesantes de la aplicaión	2 días
35		Entrega PAC3	1 día
36		**Fase 4. Memoria Final**	**12 días**
37		Elaboración del informe	13 días
38		Elaboración de la presentación viertual	13 días
39		Entrega PAC4	1 día



La planificación real ha sufrido algunas variaciones sobretodo en cuanto al aspecto formal de la planificación. En la fase 3 se ha concretado la fase de implementación para que sea mucho más descriptiva y coherente. Se ha subdividido en Implementación del servicio WCF, implementación del proyecto de administración e implementación del proyecto de usuario. También se ha añadido la tarea de manual de instalación y manual de usuario.

Por otra parte en la fase 2 se ha subdividido la tarea de especificación de la aplicación en diseño y realización de diagramas (casos de uso, E/R, clases…) y el diseño de ventanas.

Finalmente pues, el proyecto final de carrera se divide en cuatro fases fundamentales:

- Plan de trabajo.
    - Descarga del software e instalación del software. Concretamente MS Project 2010, MS Visual Studio 2010 y MS SQL Server 2008 R2. Descarga de Crystal Reports para VS 2010.
    - Elección del proyecto. En mi caso opté por diseñar un proyecto propio como se ha visto anteriormente.
    - Estudio de funcionalidades requeridas. Las funcionalidades eran inherentes al proyecto ya que conocía las necesidades del proyecto porque soy principalmente el que lo va a usar y conozco la gestión de faltas de asistencia puesto que soy el responsable.
    - Estudio de productos similares.
    - Redacción y entrega de la PAC1.
- Diseño y especificación.
    - Estudio de alternativas. Estudié con antelación las alternativas posibles. En una primera aproximación opté por una aplicación de escritorio y una móvil. También barajé la posibilidad de que la aplicación de escritorio fuera una aplicación web. Finalmente he optado por una solución con dos aplicaciones de escritorio con acceso por contraseña.
    - Diseño de la base de datos. El diseño de la base de datos ha venido definido en gran parte por los datos a importar.
    - Diseño de la interfaz de carga de datos. Es una de las partes más importantes de la aplicación ya que se encarga de importar los datos del programa externo. Sin esos datos iniciales la solución no tiene sentido alguno ya que no puede generar datos propios de clases/materias/aulas.
    - Especificación.
        - Diseño de diagramas. Casos de uso, clases y E/R principalmente. El diagrama E/R se ha realizado al mismo tiempo que el diseño de la BBDD.
        - Diseño de ventanas. Aunque luego he tenido que modificar algunos diseños de ventanas ya que durante la implementación he comprobado que esos cambios eran mejoras importantes en las funcionalidades y en la usabilidad de la aplicación final.
    - Redacción y entrega de la segunda PAC.
- Implementación.

- Implementación .NET. Es una de las fases clave en todo el desarrollo aunque viene bastante definida por la fase de especificación. Sin embargo, ha habido algunos cambios de importancia que no estaban previstos a nivel de especificación. He intentado seguir el modelo RUP[2] y por tanto los cambios en el desarrollo han producido algunos cambios en el diseño.
    - Implementación servicio WCF y acceso a datos. El acceso a datos ha sido posible mediante un proyecto propio separado LINQtoSQL. El servicio (*Windows Comunication Foundation*) se encarga de enviar las consultas al proyecto de acceso a datos. El servicio es clave en todo el proyecto ya que la aplicación solicita los datos al servicio y el servicio los solicita a la capa de datos. Esto hace que se puedan ejecutar varios usuarios y administradores al mismo servicio, que debe estar instalado en un servidor.
    - Implementación proyecto Administrador. El proyecto administrador contiene la mayoría de funcionalidades requeridas. Es el que contiene más formularios.
    - Implementación proyecto Usuario. Este segundo proyecto es más simple en cuanto a funcionalidades ya que no se necesitan.
  - Pruebas y depuración. He llevado a cabo diferentes pruebas y he añadido en este punto algunas líneas de código para poder evitar errores graves y excepciones no capturadas.
  - Documentación. Una vez desarrolladas todas las funcionalidades he realizado un manual sencillo de instalación y configuración y un manual básico de uso.
- Memoria final. En esta fase he recopilado toda la información de las otras tres fases y las he contextualizado.

En el desarrollo no ha habido contingencias destacables aunque si he de comentar aquí que el diseño de las ventanas no fue todo lo bueno que debió ser y por tanto se tuvieron que realizar algunos cambios importantes. Por otra parte, también se han realizado cambios en la base de datos inicial, añadiendo algunos campos y una tabla siguiendo las recomendaciones de mi consultor.

Además he descartado la implementación de la aplicación móvil ya que no aportaba ninguna funcionalidad especial al proyecto. A cambio he realizado una aplicación de escritorio para los usuarios.

2.1. Calendario final programado.

Nombre de tarea	Duración	Comienzo	Fin
TFC .NET - iNotHere	**76 días**	**mié 29/02/12**	**mié 13/06/12**
Fase 1. Plan de Trabajo	**9 días**	**mié 29/02/12**	**lun 12/03/12**
Descarga del software	3 días	mié 29/02/12	vie 02/03/12
Elección del proyecto	3 días	lun 05/03/12	mié 07/03/12
Estudio de funcionalidades requeridas	6 días	mié 29/02/12	mié 07/03/12
Estudio productos similares	6 días	mié 29/02/12	mié 07/03/12
Redacción del plan de trabajo	3 días	jue 08/03/12	dom 11/03/12
Entrega PAC1	1 día	lun 12/03/12	lun 12/03/12

[2] Por ejemplo http://rupmodelo.blogspot.com.es/2010/11/todo-aserca-del-modelo-de-desarrollo.html

Fase 2. Especificación y diseño	**23 días**	**mar 13/03/12**	**jue 12/04/12**
Estudio de alternativas posibles	10 días	mar 13/03/12	sáb 24/03/12
Diseño de base de datos	10 días	mar 13/03/12	sáb 24/03/12
Diseño de la interface carga de datos	10 días	mar 13/03/12	sáb 24/03/12
Especificación aplicación	**8 días**	**lun 26/03/12**	**mié 04/04/12**
Diagramas de casos de uso	8 días		
Diagrama de clases	8 días		
Diagrama E/R	8 días		
Diseño de interfaces	8 días		
Escribir PAC2	6 días	mié 04/04/12	mié 11/04/12
Entrega PAC2	1 día	jue 12/04/12	jue 12/04/12
Fase 3. Implementación de la Aplicación	**33 días**	**jue 12/04/12**	**lun 28/05/12**
Implementación .NET	**28 días**	**jue 12/04/12**	**lun 21/05/12**
Implementación del servicio WCF/Acceso BBDD	**28 días**	**jue 12/04/12**	**lun 21/05/12**
Implementación métodos	28 días		
Implementación clase tableFaltas	28 días		
Implementación proyecto administración	**18 días**	**jue 12/04/12**	**lun 07/05/12**
Implementación de formularios	15 días	vie 13/04/12	jue 03/05/12
Implementación de informes CR	3 días	jue 03/05/12	lun 07/05/12
Implementación proyecto usuario	6 días	lun 07/05/12	lun 14/05/12
Pruebas y depuración	**6 días**	lun 14/05/12	dom 20/05/12
Primera versión	1 día	dom 20/05/12	dom 20/05/12
Documentación	**6 días**	**lun 21/05/12**	**lun 28/05/12**
Manual de usuario	3 días	lun 21/05/12	mié 23/05/12
Manual de instalación	4 días	mié 23/05/12	sáb 26/05/12
Aspectos Interesantes de la aplicación	2 días	sáb 26/05/12	dom 27/05/12
Entrega PAC3	1 día	lun 28/05/12	lun 28/05/12
Fase 4. Memoria Final	**12 días**	**mar 29/05/12**	**mié 13/06/12**
Elaboración del informe	13 días	lun 28/05/12	mié 13/06/12
Elaboración de la presentación virtual	13 días	lun 28/05/12	mié 13/06/12
Entrega PAC4	1 día	mié 13/06/12	mié 13/06/12

3. Productos obtenidos.

Hemos obtenido productos documentales y aplicaciones.

3.1. Productos documentales.

En cuanto a los productos documentales he realizado durante el desarrollo de este proyecto:

- Documento PAC1. Con el plan de trabajo previsto, ya comentado.
- Documento PAC2. Con el diseño e implementación. Ha sufrido algunas variaciones que ya he esbozado en puntos anteriores y que iré describiendo en apartados posteriores.
- Manual de instalación. Breve documento donde se detalla el proceso de instalación del software obtenido.
- Manual de usuario. En este manual se describen las funcionalidades del producto informático obtenido.
- Memoria final. Este mismo documento.
- Presentación virtual en formato de video.

3.2. Aplicaciones.

En cuanto al producto en si, hemos obtenido los siguientes productos:

- Aplicación UsuarioAdministrador. La aplicación que se encarga de administrar las ausencias del profesorado. De tipo Windows Forms.
- Aplicación Usuario. Se encarga de gestionar las solicitudes del profesorado. De tipo Windows Forms.
- Aplicación servicio. El servicio se encarga de la comunicación entre la aplicación de escritorio y la base de datos. De tipo WCF.
- Aplicación acceso a datos. Se encarga de la gestión de los datos (base de datos). Genera en tiempo de ejecución LINQtoSQL.

4. Requerimientos.

4.1. Requerimientos de hardware.

Para el desarrollo de este proyecto no es necesario un equipo demasiado sofisticado. Únicamente necesitamos un ordenador P4 o superior capaz de ejecutar un sistema operativo Windows XP o superior holgadamente. De todas formas el proyecto actual se ha llevado a cabo en un equipo:

- Intel Core i5 2500k 3,0 GHz
- 8 GB RAM
- 1 TB de disco duro.
- Red 10/100/1000 MB integrada.

No se requieren otro hardware específico. Como referencia, con el equipo anterior el desarrollo es rápido y cómodo, los tiempos de compilado son muy cortos y el servicio se crea con mucha rapidez. La aplicación final es de respuesta rápida y fluida.

Para el uso a nivel de producción es necesario alojar el servicio en un servidor que tampoco requiere características especiales.

4.2. Requerimientos de software.

En cuanto al software es necesario que el proceso de desarrollo se lleve a cabo en un entorno Windows (como mínimo Windows XP). Además en el desarrollo es necesario contar con licencia de

- MS SQL Server
- MS Visual Studio 2010 SP1
- MS Project 2010
- SQL Server 2008 R2

4.3. Requerimientos funcionales.

Los requerimientos que se plantean en este proyecto son:

- ✓ Crear una interface capaz de importar desde archivos .txt los datos de los horarios procedentes del programa generador de horarios (UNTIS).
- ✓ Mostrar por pantalla el horario de un profesor seleccionado anteriormente y poder seleccionar las horas en las que no va a asistir el profesor así como la fecha exacta. También debe permitir la eliminación de dichas faltas si finalmente la ausencia no se produce.
- ✓ La aplicación estará dotada de un panel informativo donde se detallará la información relevante sobre los procesos de inserción de faltas, eliminación, importación e impresión, así como cualquiera otro que sea necesario.
- ✓ Generar informes estadísticos, con el número de ausencias justificadas o sin justificar.
- ✓ Generar listado de ausencias en formato pdf para una fecha dada, para cada una de las horas de clase del día.

- ✓ Restringir el acceso con claves para diferentes usuarios administradores o usuarios restringidos. La contraseña quedará guardada en la base de datos encriptada con SHA1.
- ✓ Posibilitar que el administrador pueda justificar ausencias (previstas o no).
- ✓ Permitir que al iniciar por primera vez (con la base de datos vacía) se pueda acceder al sistema en modo administrador.
- ✓ Evitar que durante el primer inicio el usuario pueda salir del sistema sin crear un usuario administrador.
- ✓ Permitir que el administrador pueda modificar datos de los usuarios y aulas.
- ✓ Permitir a los usuarios pedir permiso para ausentarse o solicitar que se les justifique una ausencia determinada mediante una interfaz diferenciada.
- ✓ Entorno basado en formularios.
- ✓ Permitir a los usuarios conocer el estado de sus peticiones.
- ✓ Posibilitar cambios de contraseña a los usuarios.
- ✓ Uso fácil y rápido con el mínimo número de clics y cuadros de diálogo.
- ✓ Respuesta rápida del sistema (máximo 5 segundos).

4.4. Análisis de riesgos

Al tratarse de un software de gestión educativa que no tiene acceso a datos personales sensibles, no se ha establecido un mecanismo especial para controlar ataques externos. De todas formas será necesario tener en cuenta posibles ataques a las contraseñas del administrador y de los usuarios privilegiados en general y por tanto en la base de datos se almacenarán usando el algoritmo SHA1, más que suficiente para el propósito buscado.

La aplicación también deberá estar debidamente protegida para evitar ataques de la que se escalen privilegios, aunque, por otra parte, el software siempre será ejecutado por un usuario limitado dentro de un dominio Windows (Active Directory).

Otros peligros no son esperables dado que el software únicamente será manipulado por un número reducido de personas.

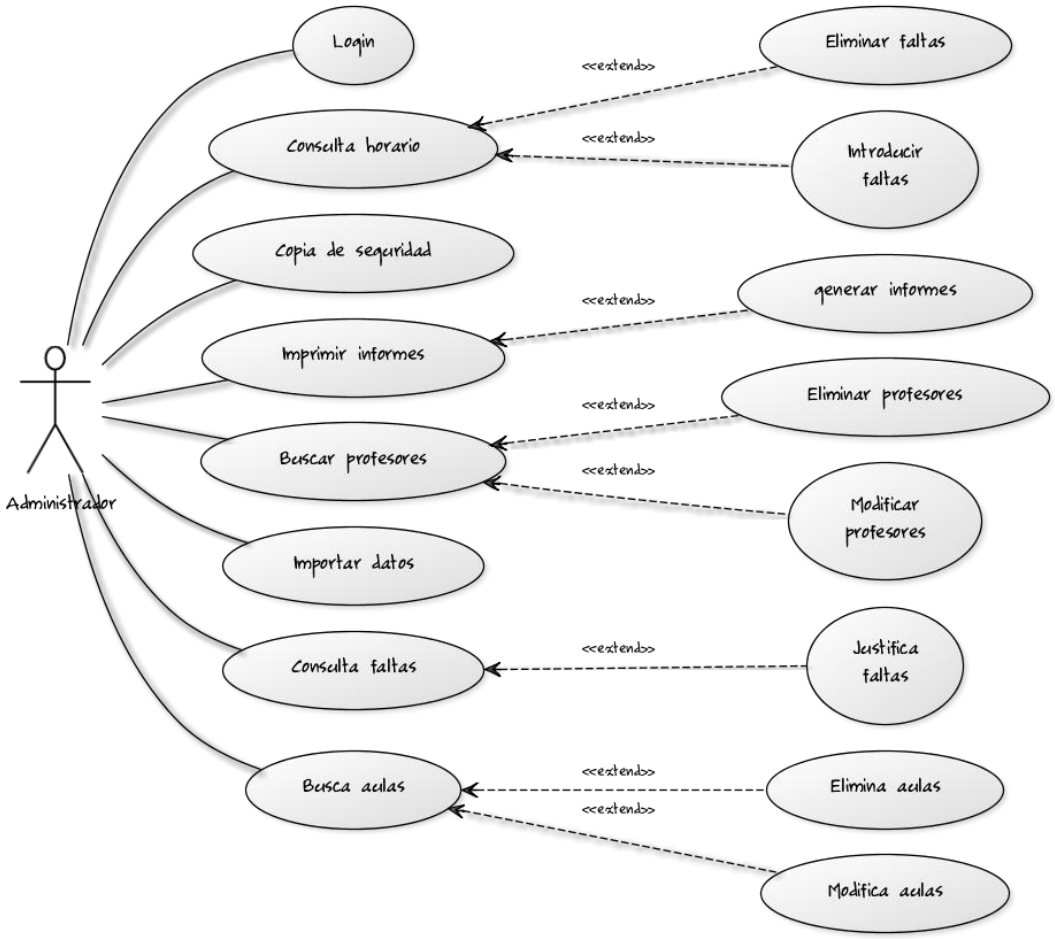
5. Documentación.

5.2. Diagramas de casos de uso.

El diagrama de casos de uso para cada proyecto se puede ver a continuación.

5.2.1. Administrador

Durante la etapa de análisis y diseño se desarrolló un diagrama de casos de uso con dos actores diferentes para este proyecto: director y subdirector. Una vez en la fase de desarrollo he pensado que las funcionalidades de cada uno de los actores eran prácticamente las mismas y por tanto no es en absoluto necesario diferenciar actores en este proyecto. Por tanto finalmente queda el actor Administrador como único en este proyecto. Para simplificar el diagrama se han obviado los <<extends>> desde Login de manera deliberada.



Nombre	Identificarse
Descripción	El usuario entra en el sistema
Actores	Administrador.
C.U. relacionados	
Precondición	El usuario tiene credenciales de administrador. El usuario no ha accedido al sistema. El usuario conoce la contraseña. El usuario está en la base de datos.

Postcondición	El usuario accede al sistema.
Flujo normal	El usuario selecciona de una lista su nombre. El usuario introduce la contraseña.
Flujo alternativo	El usuario no introduce correctamente la contraseña. El usuario no selecciona ningún nombre de usuario. No hay ningún usuario dentro de la base de datos. No hay ningún usuario administrador que pueda autenticarse en el sistema.

Nombre	Consulta horario
Descripción	El usuario consulta el horario del profesor
Actores	Administrador.
C.U. relacionados	Introducir faltas, eliminar faltas, identificarse
Precondición	El usuario tiene credenciales de administrador. El usuario no ha accedido al sistema.
Postcondición	El usuario accede al sistema.
Flujo normal	El usuario selecciona de una lista su nombre. El usuario introduce la contraseña.
Flujo alternativo	El usuario no introduce correctamente la contraseña. El usuario no selecciona ningún nombre de usuario.

Nombre	Eliminar faltas
Descripción	El usuario elimina una falta de un profesor
Actores	Administrador.
C.U. relacionados	Identificarse, consultar horario.
Precondición	Hay faltas previamente introducidas en el sistema para el profesor seleccionado
Postcondición	La falta queda eliminada de la base de datos.
Flujo normal	El usuario selecciona un profesor. El usuario selecciona una fecha El usuario selecciona la falta a eliminar y pulsa sobre eliminar falta
Flujo alternativo	No hay ninguna falta por eliminar. El usuario no introduce ningún usuario.

Nombre	Añadir faltas
Descripción	El usuario añade una falta a un profesor.
Actores	Administrador.
C.U. relacionados	Consultar horario, identificarse
Precondición	El profesor tiene alguna clase en la fecha seleccionada
Postcondición	Se añade una falta a la lista de faltas del profesor.
Flujo normal	El usuario selecciona de una al profesor El usuario selecciona una fecha. El usuario pulsa añadir falta o añadir salida extraescolar.
Flujo alternativo	El profesor no tiene ninguna clase ese día. No se selecciona ningún profesor de la lista.

Nombre	Copia de seguridad
Descripción	El usuario accede a la información.
Actores	Administrador.
C.U. relacionados	Identificarse
Precondición	
Postcondición	
Flujo normal	
Flujo alternativo	

Nombre	Imprimir Informe
Descripción	El usuario imprime un informe generado.
Actores	Administrador
C.U. relacionados	Generar informe, identificarse
Precondición	El usuario ha generado el informe
Postcondición	El usuario obtiene el informe impreso (o en pdf)
Flujo normal	1 El usuario elige un tipo de informe. 2. El usuario elige una fecha. 3. El usuario genera el informe. 2a. El usuario genera el informe.
Flujo alternativo	

Nombre	Buscar profesores
Descripción	El usuario busca a un profesor y obtiene los datos almacenados.
Actores	Administrador.
C.U. relacionados	Identificarse
Precondición	El profesor buscado existe en la base de datos.
Postcondición	Se muestra la información del profesor solicitado.
Flujo normal	El usuario selecciona un profesor.
Flujo alternativo	El usuario no selecciona ningún profesor pero pulsa el botón buscar. El usuario pulsa el botón actualizar, eliminar.

Nombre	Eliminar profesor
Descripción	El usuario eliminar de la base de datos a un profesor.
Actores	Administrador.
C.U. relacionados	Busca profesor, identificarse
Precondición	El profesor buscado existe en la base de datos.
Postcondición	El profesor es eliminado de la base de datos.
Flujo normal	El usuario selecciona un profesor y pulsa el botón eliminar.
Flujo alternativo	El usuario no ha seleccionado ningún profesor. El usuario no pulsa el botón eliminar.

Nombre	Modificar profesor
Descripción	El usuario modifica algunos campos de la tabla profesores
Actores	Administrador.
C.U. relacionados	Busca profesor, identificarse
Precondición	El profesor buscado existe en la base de datos. El usuario introduce una nueva contraseña para el profesor
Postcondición	El profesor es actualizado y la contraseña cambiada.
Flujo normal	El usuario selecciona un profesor. El usuario cambia los parámetros necesarios y pulsa el botón actualizar.
Flujo alternativo	El usuario no introduce ninguna contraseña El usuario no introduce alguno de los campos necesarios. El usuario no pulsa el botón actualizar.

Nombre	Importar datos.
Descripción	El usuario puede importar datos de horarios (clases, profesores, aulas, materias) a la base de datos local.
Actores	Administrador.
C.U. relacionados	Identificarse
Precondición	
Postcondición	Se insertan todos los datos en la base de datos.
Flujo normal	El usuario selecciona la carpeta donde están los archivos de importación.
Flujo alternativo	En la carpeta no se encuentran los archivos de importación

Nombre	Consulta faltas
Descripción	El usuario consulta las faltas de un profesor en una fecha.
Actores	Administrador
C.U. relacionados	Identificarse
Precondición	El profesor tiene faltas introducidas en su ficha.
Postcondición	
Flujo normal	El usuario selecciona una fecha y un profesor.
Flujo alternativo	El usuario no selecciona ningún profesor.

Nombre	Justifica faltas
Descripción	El usuario justifica las faltas seleccionadas.
Actores	Administrador
C.U. relacionados	Consultar faltas, identificarse
Precondición	El profesor tiene faltas en su ficha.
Postcondición	Se marcan como justificadas las faltas seleccionadas.
Flujo normal	1. El usuario elige como desea ver la información. Por profesor, por fecha, por fecha y por profesor y todas. 2. El usuario selecciona la falta que quiere justificar.

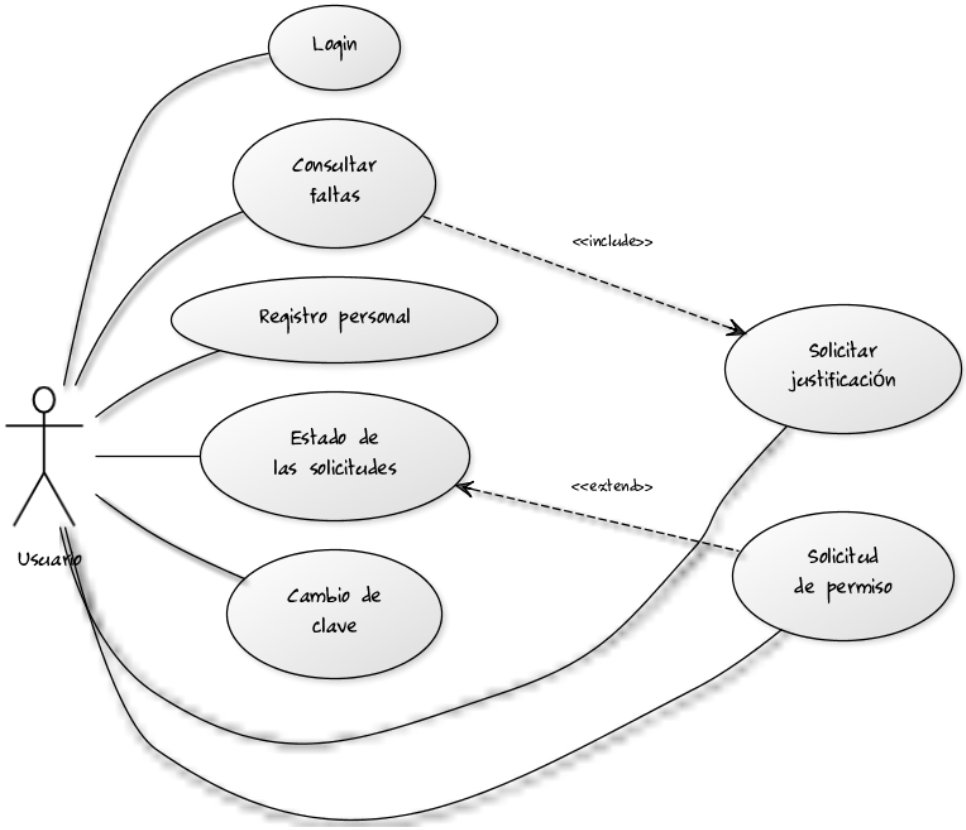
Flujo alternativo	El usuario no elige ningún profesor ni fecha.

Nombre	Elimina aulas
Descripción	El usuario elimina una aula de la lista
Actores	Administrador
C.U. relacionados	Modifica aula, identificarse
Precondición	El aula seleccionada existe.
Postcondición	El aula se borra de la base de datos.
Flujo normal	El usuario busca una aula y pulsa sobre el botón eliminar
Flujo alternativo	El usuario no selecciona ninguna aula. El usuario no presiona el botón eliminar.

Nombre	Modificar aulas
Descripción	El usuario modifica el nombre o la descripción del aula.
Actores	Administrador
C.U. relacionados	Identificarse, buscar aula.
Precondición	El aula existe
Postcondición	El aula queda modificada.
Flujo normal	El usuario busca una aula. El usuario modifica los detalles del aula y pulsa modificar.
Flujo alternativo	El aula no existe. El usuario no pulsa el botón modificar.

5.2.2. Usuario.

En este proyecto desde un principio estaba claro que con la participación de un actor era suficiente y por tanto no hubo dudas en este aspecto. Cabe tener en cuenta que un actor administrador en el proyecto Administración es usuario en este proyecto. Al igual que antes se han omitido deliberadamente los <<extends>> desde Login.



Nombre	Identificarse
Descripción	El usuario entra en el sistema
Actores	Usuario
C.U. relacionados	
Precondición	El usuario no ha accedido al sistema. El usuario está en la base de datos. El usuario conoce la contraseña.
Postcondición	El usuario accede al sistema.
Flujo normal	El usuario selecciona de una lista su nombre. El usuario introduce la contraseña.
Flujo alternativo	El usuario no introduce correctamente la contraseña. El usuario no selecciona ningún nombre de usuario. No hay ningún usuario dentro de la base de datos.

Nombre	Consultar faltas
Descripción	Consultar faltas
Actores	Usuario
C.U. relacionados	Identificarse, solicitar justificación
Precondición	EL usuario tiene faltas.
Postcondición	El usuario ve las faltas que tiene
Flujo normal	El usuario consulta las faltas.
Flujo alternativo	

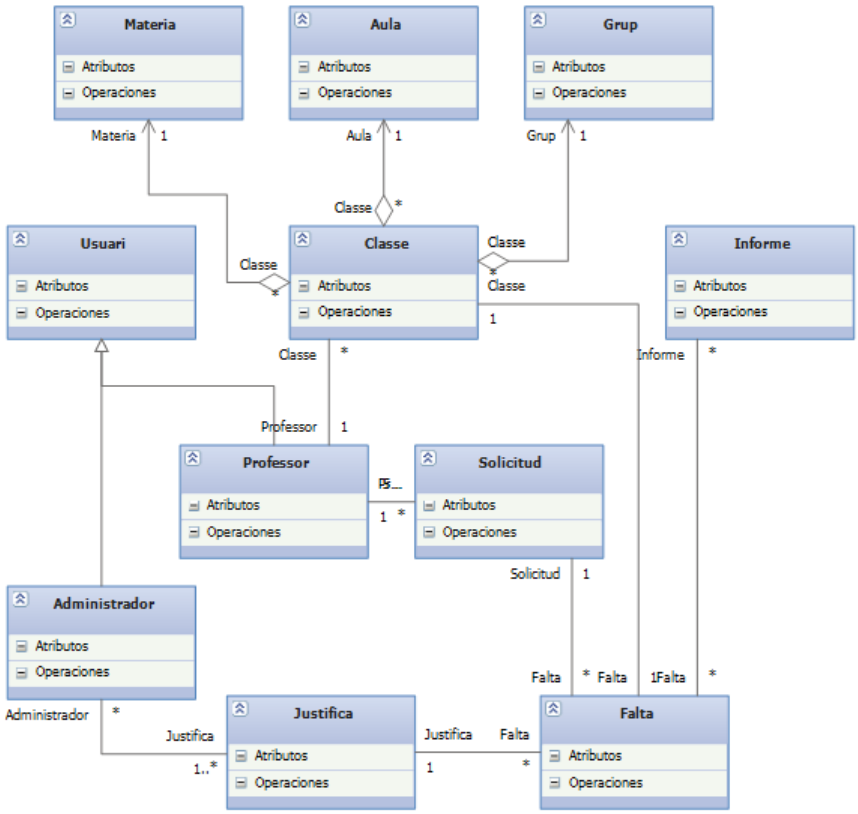
Nombre	Solicitar justificación
Descripción	El usuario pide que se le justifiquen las faltas de asistencia seleccionadas.
Actores	Usuario
C.U. relacionados	Identificación, consulta faltas
Precondición	El usuario tiene faltas SIN justificar
Postcondición	EL usuario tiene la falta seleccionada justificada.
Flujo normal	El usuario selecciona una falta y pide justificación.
Flujo alternativo	El usuario no tiene ninguna falta.

Nombre	Estado de la solicitud
Descripción	El usuario comprueba en qué estado está su solicitud.
Actores	Usuario
C.U. relacionados	Identificación
Precondición	El usuario tiene solicitudes en curso
Postcondición	
Flujo normal	
Flujo alternativo	

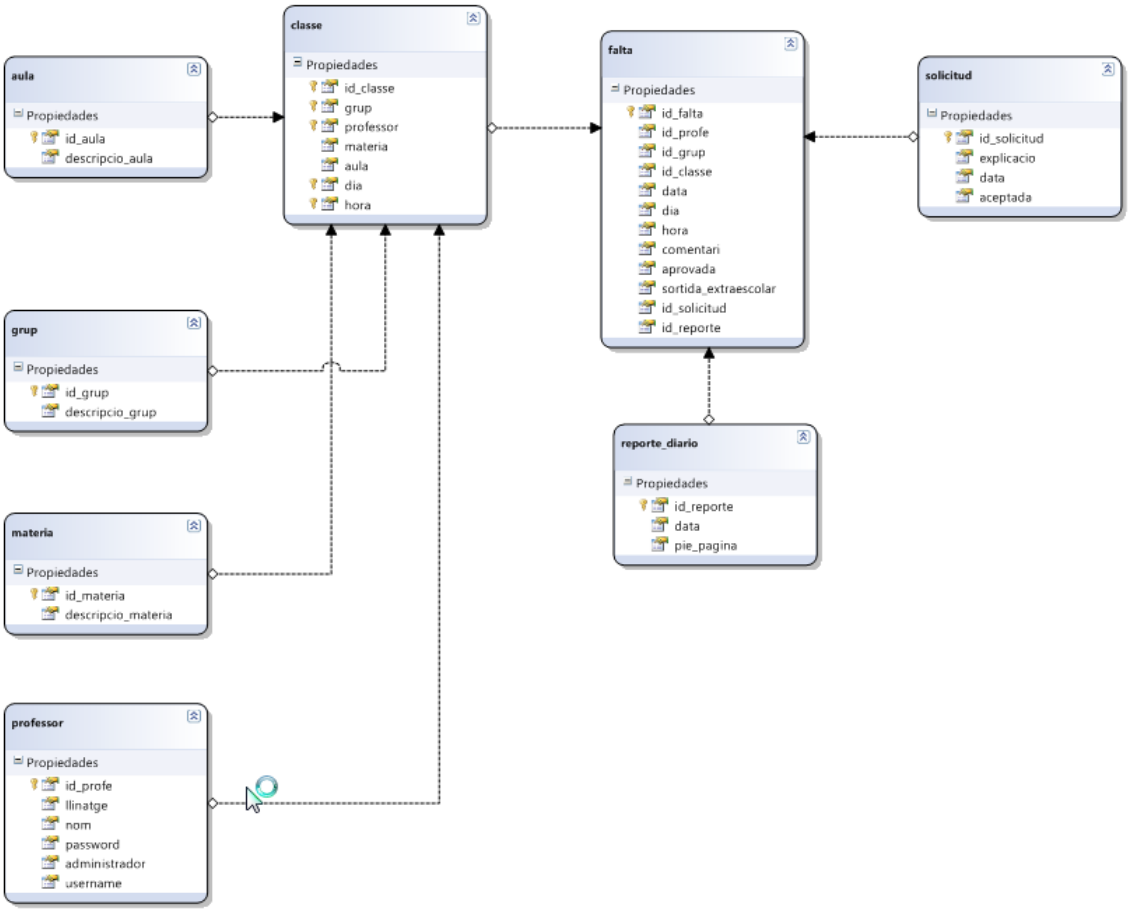
Nombre	Solicitud de permiso
Descripción	El usuario solicita permiso para no asistir a las horas seleccionadas.
Actores	Usuario
C.U. relacionados	Identificación, estado de la solicitud
Precondición	El usuario introduce un motivo justificado.
Postcondición	Se guarda la solicitud del usuario
Flujo normal	El usuario selecciona una fecha. El usuario selecciona las horas a solicitar y pulsa guardar.
Flujo alternativo	El usuario no tiene horas en un día solicitado. El usuario no introduce ningún motivo justificado.

5.3. Diagrama de clases

En el diagrama de clases he introducido clases ausentes en anteriores entregas y he completado el diagrama para que represente correctamente el proyecto presentado. Se pude ver en la figura siguiente.



5.4. Diagrama ER de la base de datos

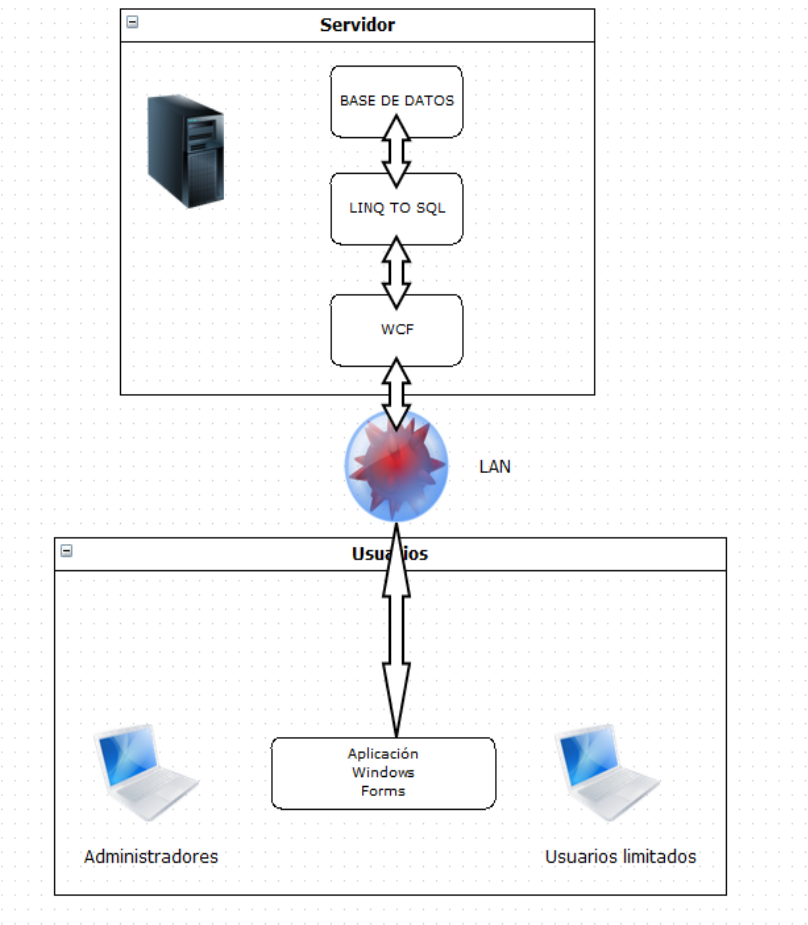


5.5. Implementación.

Durante este proyecto he utilizado la metodología de desarrollo en tres capas en cada proyecto incluido en la solución.

- ✓ La capa de presentación, donde he desarrollado la interfaz gráfica y es la que se muestra al usuario final. Los formularios son la representación física de esta capa.
- ✓ La capa de negocio es la que realiza la mayoría de funcionalidades internas. Recogen peticiones de la capa de presentación y de la propia capa de negocio y devuelve la respuesta después de procesar la información, ya sea de la capa de datos (con la que está comunicada) o no. La comunicación con la base de datos únicamente se puede hacer a través de esta capa y por tanto no hay acceso directo entre la capa de presentación y la capa de datos.
- ✓ La capa de datos es la que se encarga de acceder y administrar los datos de la base de datos SQL. Esta gestión se hace a través de peticiones desde la capa de negocio.

El diagrama de arquitectura para la solución final se puede observar en la figura siguiente.



Como ya he comentado en algún momento, el proceso de implementación ha ido en paralelo a un aprendizaje de nuevas tecnologías, tanto usadas aquí, como otras que no he usado finalmente por no adaptarse al proyecto o por requerir un aprendizaje más pausado y una dedicación más exhaustiva.

En primer lugar empecé a implementar la aplicación que contendría la lógica de la base de datos basada en tecnología LINQtoSQL. En segundo lugar fue necesaria la implementación del servicio WCF y generar los OperationContracts necesarios para que la aplicación fuera creciendo.

La fase más compleja ha sido la creación de la lógica necesaria para la conexión de la interfaz gráfica con la base de datos. Aquí he tenido algunos problemas derivados de respuestas lentas del servicio ante algunas solicitudes de datos que no estaban fuertemente tipados y por tanto hubo que hacer algunos cambios en este sentido.

Finalmente tuve que rehacer algunos diseños de la interfaz y adaptar otros a las necesidades que iba generando la fase de implemntación.

5.5.1. Tecnologías.

El desarrollo de todo el proyecto se ha basado en Windows Forms. En un primer análisis opté por tecnologías mixtas ASP .NET / Windows Forms e incluso barajé la posibilidad del uso de tecnología ASP .NET en su totalidad, pero por requerimientos funcionales y porque disponemos de todo el hardware necesario para la implementación del proyecto finalmente decidí usar únicamente Windows Forms. Más adelante en el apartado de conclusiones y mejoras hago algunas reflexiones sobre este aspecto.

El lenguaje que he usado es C# ya que parece más extendido y con más recursos a disposición. En un principio mi dominio de C# y de VB era mínimo.

El uso de **Windows Forms** permite desarrollar aplicaciones basadas en ventanas de manera fácil y rápida. También ofrece una gran flexibilidad y el uso de esta tecnología tiene una curva de aprendizaje relativamente corta.

Para la comunicación he utilizado **Windows Comunication Foundation** para crear un servicio que efectuará las peticiones a la capa de datos. Esta orientación a servicios hace que WCF sea muy interesante para poder crear todo tipo de aplicaciones que puedan conectarse a esos servicios.

ADO.NET también presentes en .NET es la tecnología que permite el acceso a datos, bien sea una base de datos SQL o bien un origen de datos ODBC, como por ejemplo durante la importación de datos de horarios en el proyecto de administración.

La principal ventaja de ADO.NET es que gracias al nivel de abstracción que se consigue se pueden acometer cambios estructurales en las bases de datos con impactos relativamente pequeños en el código.

LINQ es un sistema que permite realizar consultas a la base de datos (Language INtegrated Query). Esta tecnología permite realizar consultas a diferentes tipos de datos de manera unificada. Por ejemplo permite consultar los datos de una tabla de una base de datos o bien de un datagridview o incluso en un dataset con la misma sintaxis. Evidentemente esto tiene ventajas para el programador.

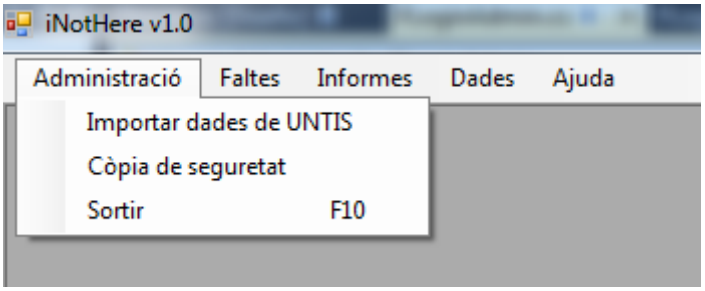
5.6. Manual del usuario: Administrador.

El usuario con perfil de administrador podrá realizar las siguientes funciones.

5.6.1. Descripción de funcionalidades.

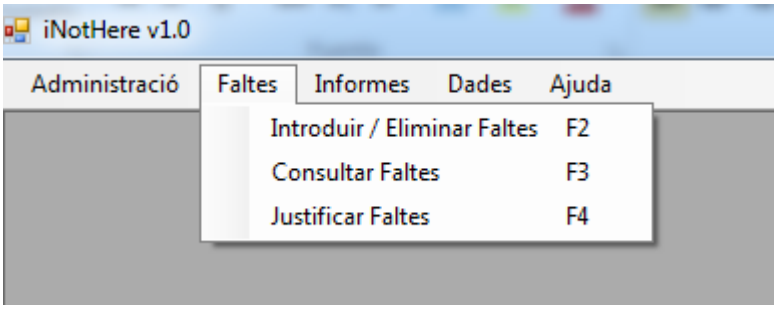
5.6.1a. Menú Administració

En este menú se puede ejecutar las siguientes funcionalidades: Importar datos del programa de horarios, hacer copia de seguridad y salir. En el caso de la importación de datos desde el programa UNTIS (externo a esta aplicación) se deberá tener exportados los datos con anterioridad (ver 1.3.2).



Menú Faltes.

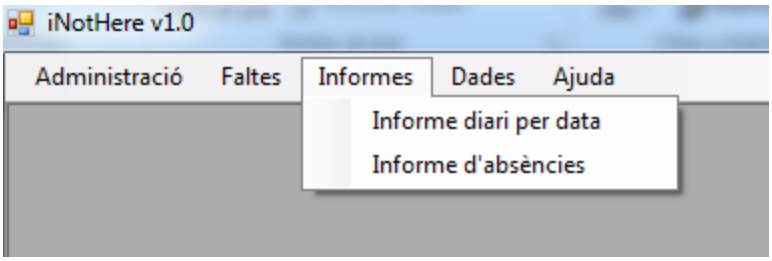
Aquí se desarrollan las funcionalidades siguientes:



- **Introduir/eliminar faltes**. El administrador podrá introducir o eliminar faltas una vez tenga constancia de que un profesor no va a acudir a clase. En este caso, el administrador introducirá las sesiones en las que se vaya a ausentar el profesor. Si se produce algún error de entrada de datos, el admin podrá eliminar una falta errónea.
- **Consultar Faltas**. El admin podrá consultar las faltas para una fecha concreta y generar un informe por sesiones. En secundaria actualmente hay hasta 7 sesiones diarias descontando los recesos para descansar.
- **Justificar Faltes**. El admin podrá justificar faltas de asistencia del profesorado. También podrá rectificar si ha habido algún error.

Menú Informes.

Se desarrollan las funcionalidades:

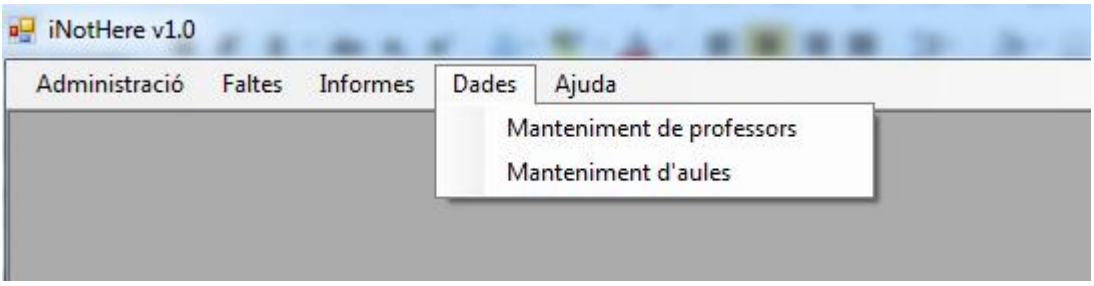


- **Informe diari per data**. Con esta entrada, se genera un informe diario de faltas de asistencia que se expondrá en la sala de profesores para conocimiento del profesorado que debe sustituir ausencias.
- **Informe d'absències**. En este informe se listan todas las ausencias de todos los profesores.

Menú dades.

En este menú encontramos las siguientes funcionalidades:

- **Manteniment de professors**. Podemos cambiar datos de profesores y contraseñas para poder subsanar errores o cambios.
- **Mantenimient d'aules**. Podemos cambiar el nombre de una aula o de un espacio dentro de la base de datos.
- **Manteniment d'assignatures**. Podemos cambiar el nombre de una asignatura si es necesario.



5.6.2. Estructura del menú.

Como se ha visto en el apartado anterior la estructura del menú principal es la siguiente:

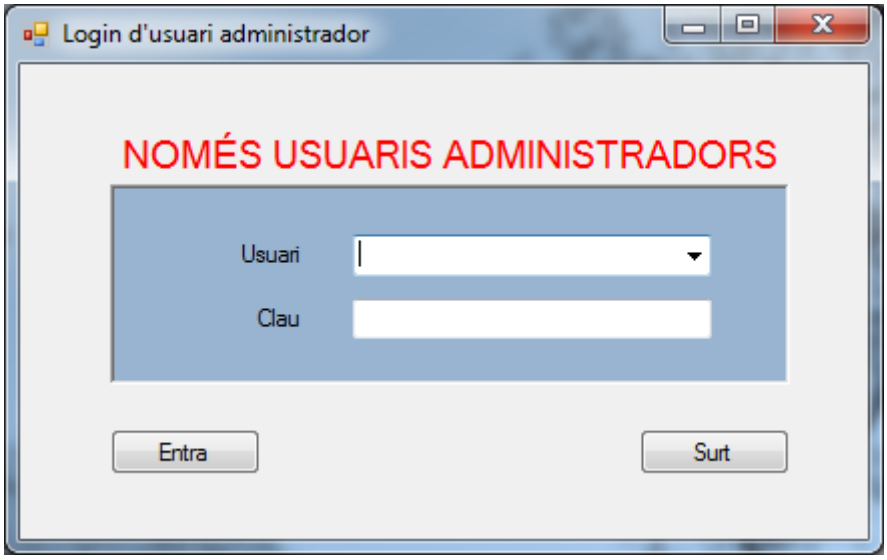
Administració	Faltes	Informes	Dades	Ajuda
Importar dades Còpia de seguretat	Introduir/Eliminar faltes Consultar faltes	Informe diari per data Informe d'absències	Mant. De professors Mant. D'aules	
Sortir	Justificar faltes			

5.6.2a. Menu Administració.

En este menú podemos ejecutar operaciones de mantenimiento de la base de datos en general.

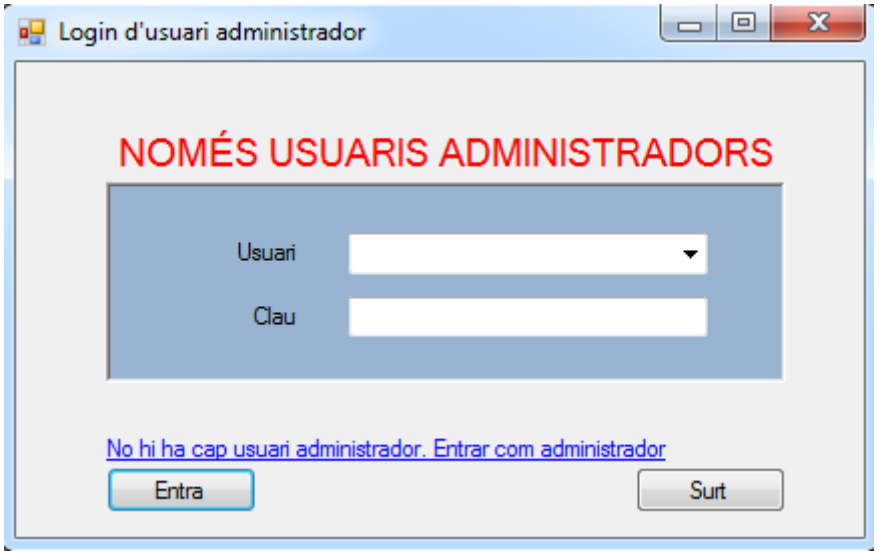
Autenticación de usuario.

Antes de acceder a la aplicación es necesario autenticarse. La figura siguiente muestra el cuadro de diálogo que permite identificar al usuario como administrador.



Al introducir el nombre y la contraseña del administrador se inicia el programa de administración.

Atención: En el caso de que no haya ningún usuario en la base de datos o que no haya ningún administrador en la base de datos, el sistema permitirá la entrada para subsanar el problema.



Haciendo clic en el mensaje tendremos acceso al programa. Recuerde que el programa no permitirá salir del programa si no se ha seleccionado al menos un administrador.

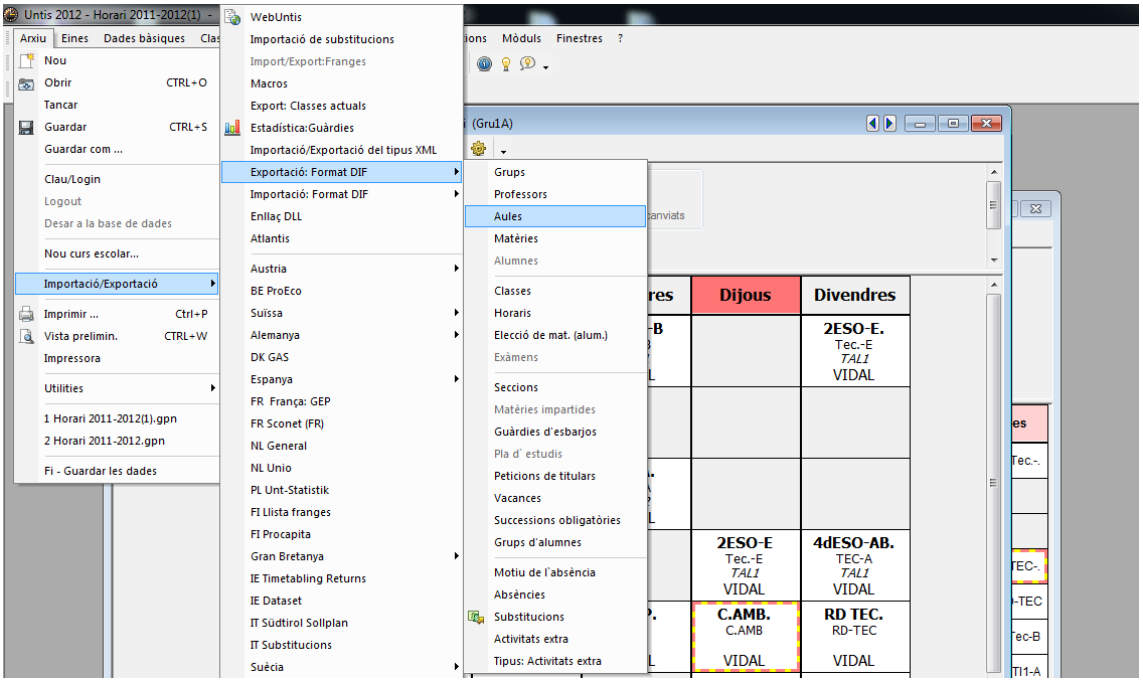
Administració → Importar dades.

Una vez identificado como administrador en la opción seleccionada podemos importar los datos procedentes de UNTIS (programa de horarios). Untis permite gestionar y

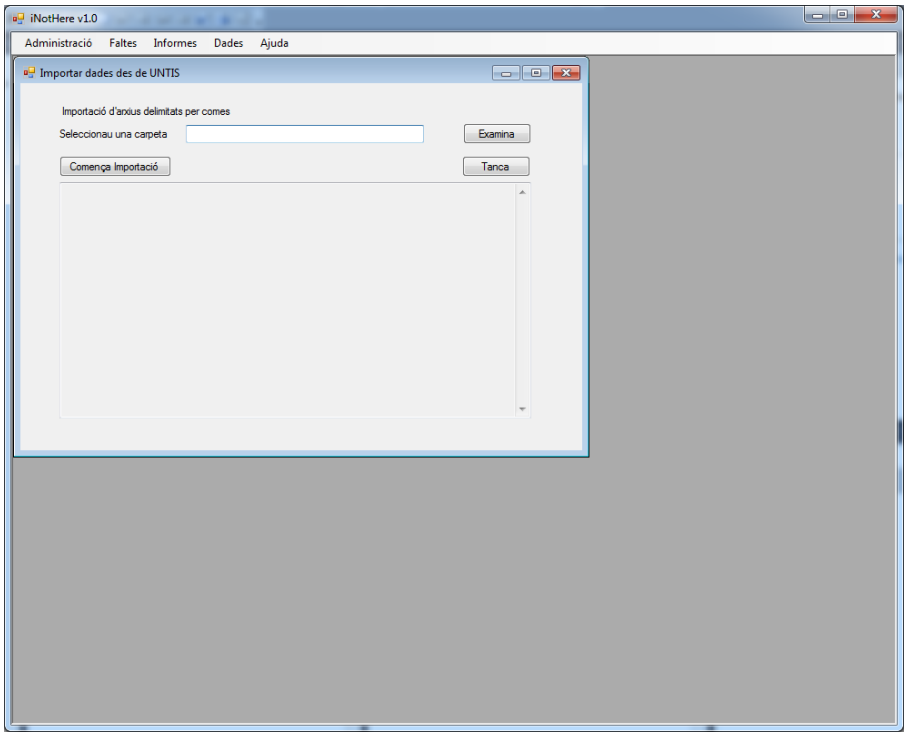
organizar horarios escolares y exportar los datos necesarios. Por tanto antes de nada debemos exportar todos los datos necesarios de UNTIS.

Para ello abrimos UNTIS con el icono del escritorio o menú de inicio. Una vez iniciado el programa y en el menú archivo→importación /exportación→ Exportación en formato DIF debemos exportar clases, grupos, profesores aulas y materias.

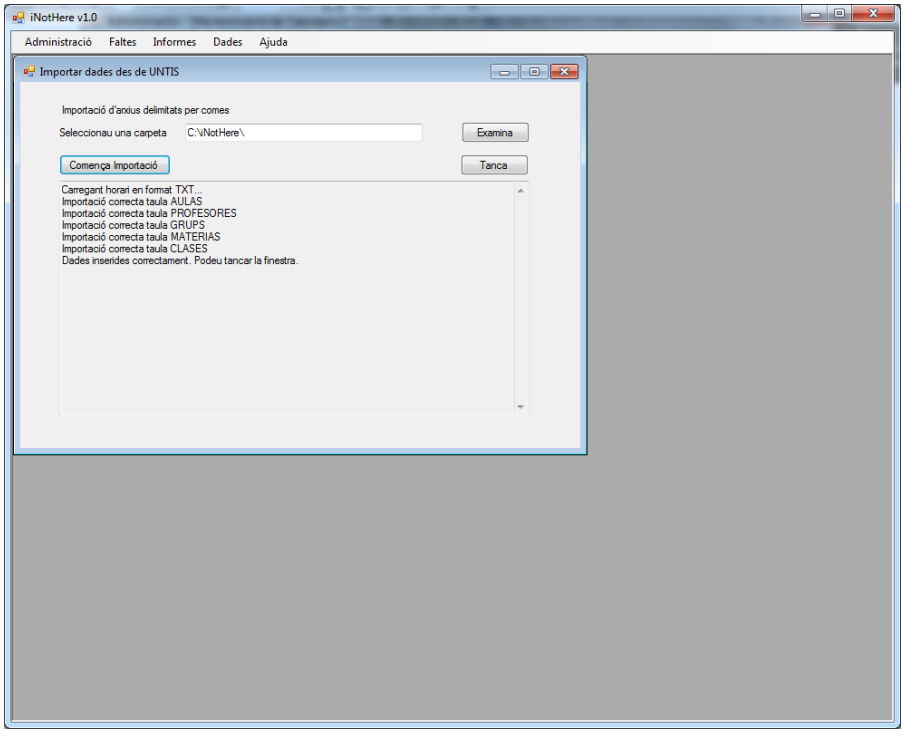
En la imagen se puede ver el proceso.



Una vez exportados todos los datos de Untis volvemos a iNotHere y seleccionamos la carpeta donde hemos ubicado los archivos a importar. Finalmente pulsamos el botón *comença importació*.



Obtendremos un mensaje como el que aparece en la figura siguiente

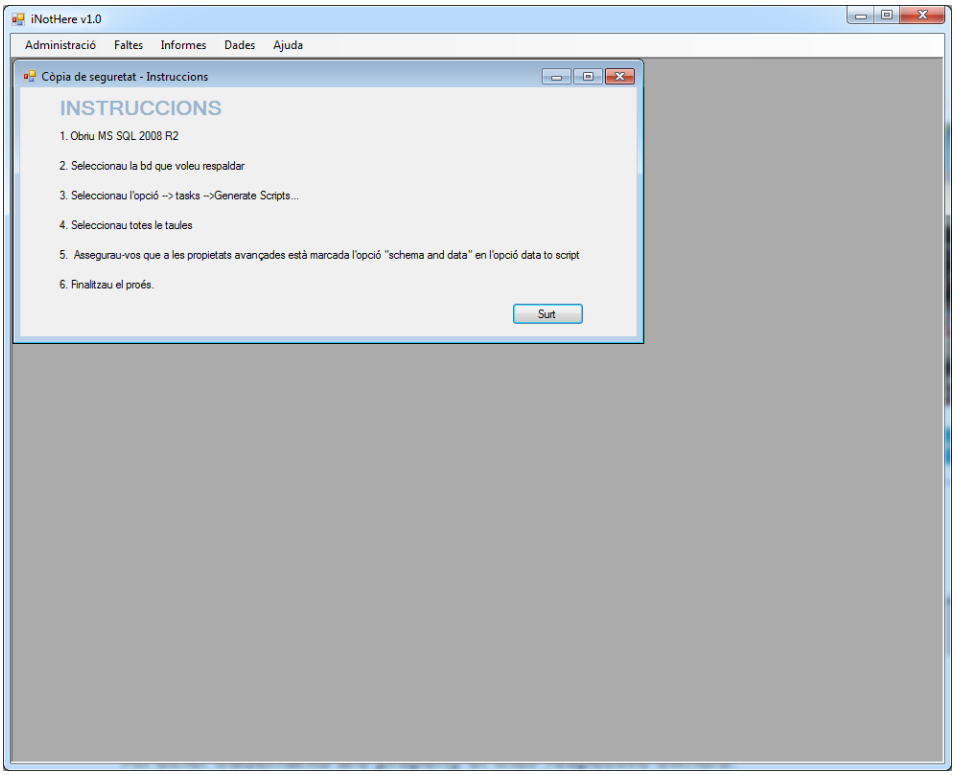


Con esto, se habrán cargado los nuevos datos en la base de datos del sistema.

⚠ Todos los datos del sistema se borrarán sin posibilidad de volver a recuperarse.

Administració→Còpia de seguretat.

Simplemente se deben seguir las instrucciones que aparecen.

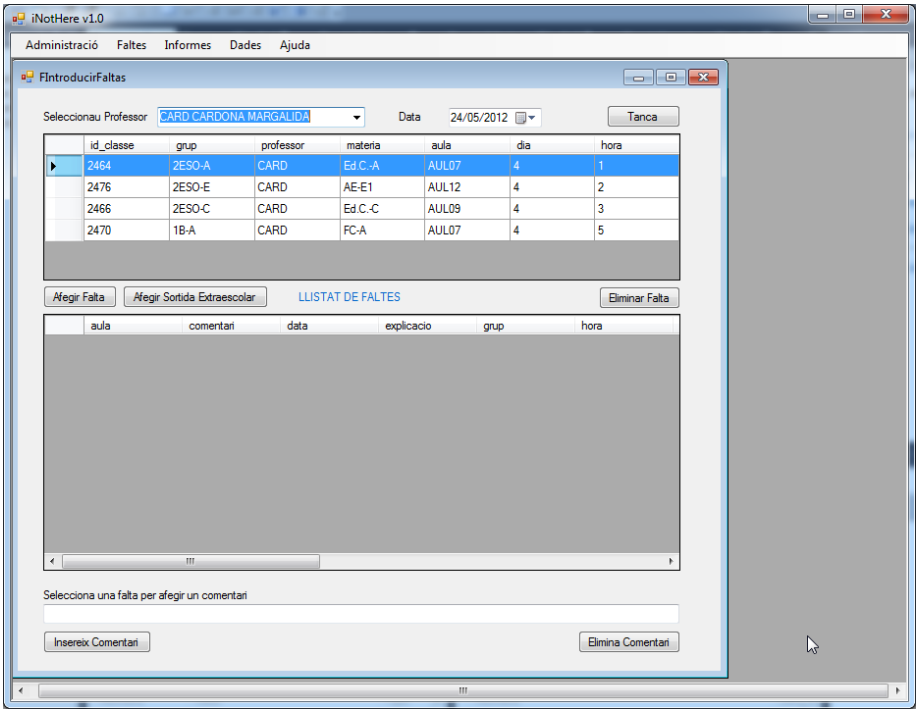


5.6.2b. Menú Faltes.

En este menú se encuentran las opciones de añadir/eliminar/justificar faltas.

Introduir/Eliminar faltes

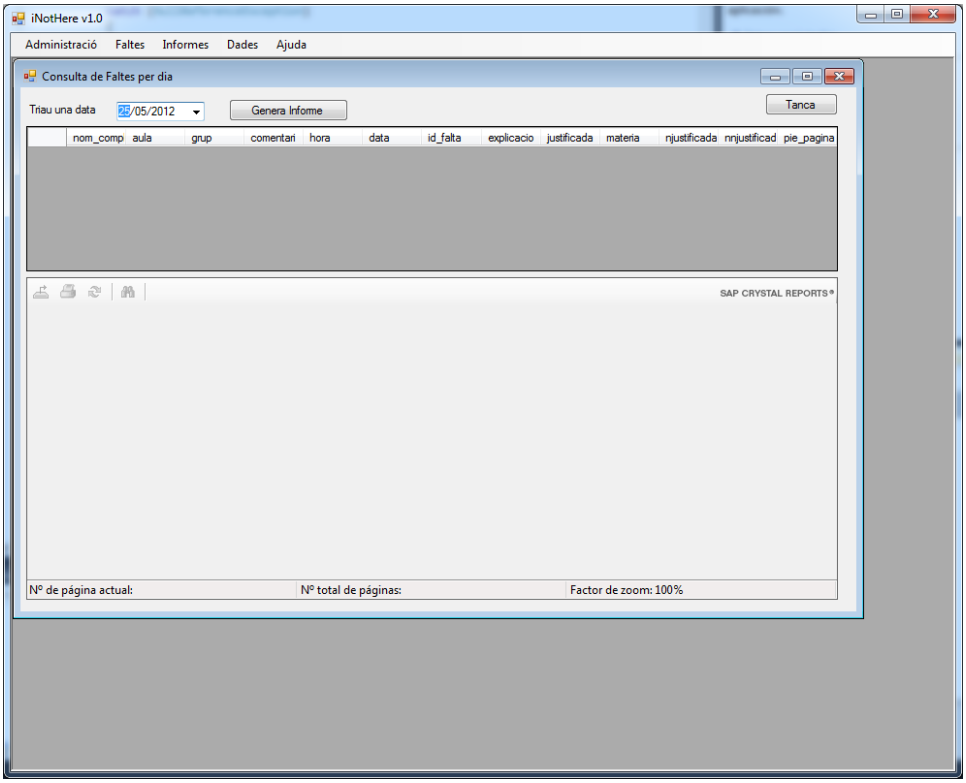
En este menú podremos introducir o eliminar faltas. Para introducir debemos seleccionar a un profesor de la lista desplegable y una fecha. Aparecerá automáticamente el horario del profesor seleccionado.



Con los botones Afegir Falta y Eliminar Falta podemos pasar las horas que va a faltar el profesor al cuadro inferior. Se puede añadir un comentario a la falta que se va a introducir. Las faltas quedarán guardadas inmediatamente. Para insertar el comentario será necesario pulsar el botón correspondiente. Si la falta que queremos introducir corresponde a una falta por una salida extraescolar debemos pulsar el botón *Sortida Extraescolar*.

Consultar Faltes.

Esta opción permite al administrador consultar todas las faltas para un día determinado y generar un informe con las ausencias de ese día. El usuario puede seleccionar el día que desee.



Justificar faltes.

Desde aquí podemos justificar faltas del profesorado. Hay tres tipos de faltas:

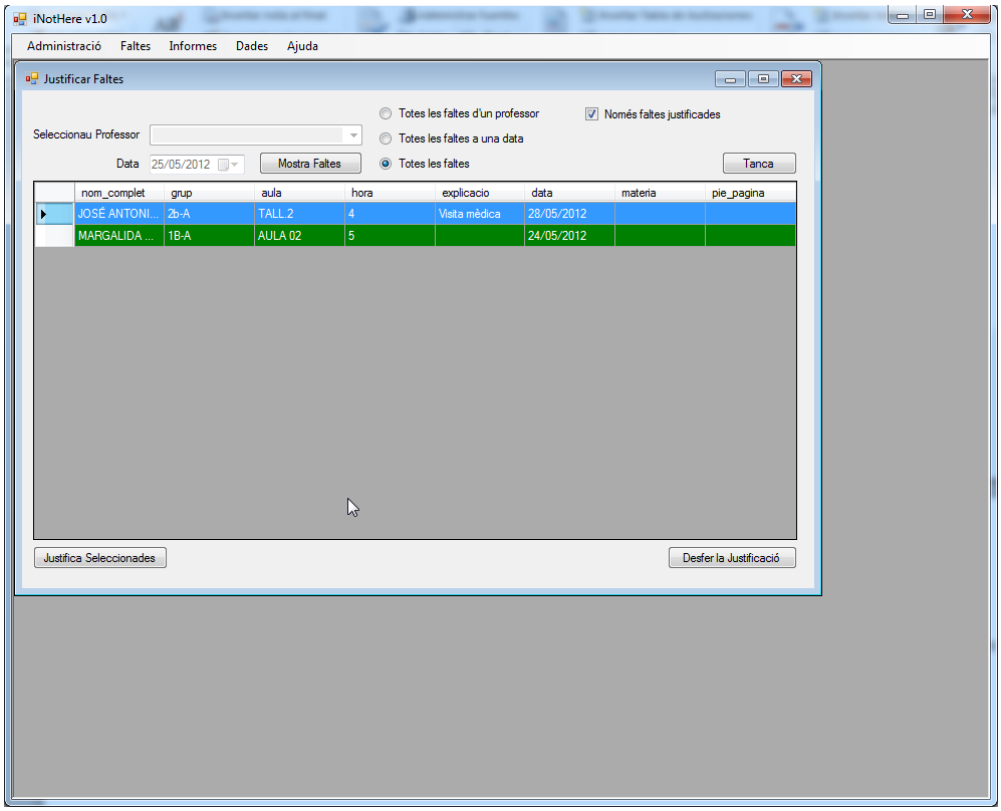
- Faltas previstas con anterioridad (solicitud de permiso).
- Faltas imprevistas justificables con posteiroridad (solicitud de justificación).
- Faltas por salidas a actividades complementarias (justificadas por defecto).

Las faltas que corresponden a salidas de los profesores con alumnos se consideran automáticamente por el programa como faltas justificadas. Por tanto, a no ser que tengamos el checkbox "*només faltes sense justificar*" desactivado, ya no aparecerán en la lista de esta ventana. En la lista podrán aparecer faltas de los tipos 1 y 2 anteriores.

En esta ventana podemos elegir la lista de faltas de diferentes maneras:

- Ver todas las faltas de un profesor.
- Ver todas las faltas en una fecha.
- Ver todas las faltas.
- Ver todas las faltas de un profesor en una fecha.
- Todas las combinaciones anteriores de faltas justificadas y sin justificar.

Una vez elegida la opción deseada debemos pulsar sobre el botón "*justifica faltes*" para que las faltas queden justificadas. Las faltas justificadas aparecerán de color verde.



Si ha habido algún error podemos deshacer la justificación de faltas en cualquier momento. Para ello seleccionamos las faltas que deseamos (selección múltiple con CTRL) y pulsamos sobre el botón "*desfer la justificació*".

5.4.2c. Menú Informes.

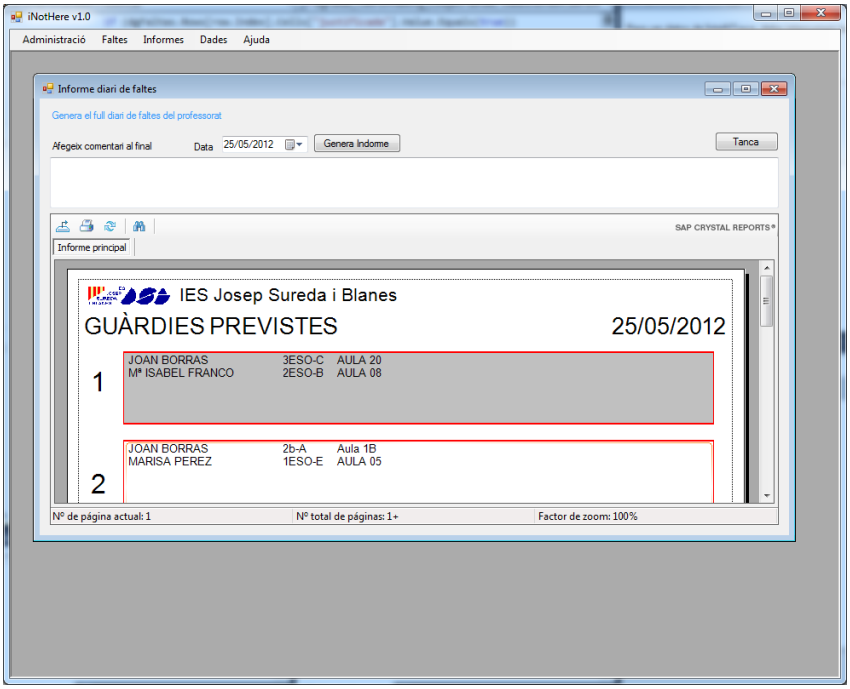
Informe diari de faltes.

Aquí tenemos una de las opciones más importantes del programa, el informe diario de faltas. Este informe se genera automáticamente con las faltas previstas para la fecha dada. Además, el programa permite añadir un comentario al final de la página para poder realizar algún aviso o informar de algún aspecto relevante a criterio del administrador.

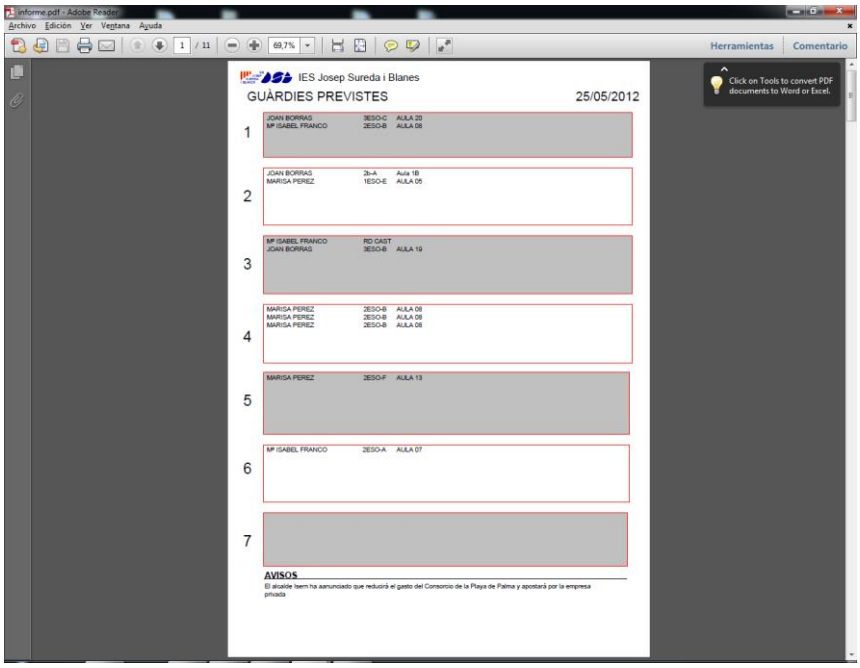
El informe está generado con Crystal Reports y por tanto es necesario tener instalado este software. Para más información ver manual de instalación.

En primer lugar seleccionamos una fecha. Por defecto el programa abre el informe para el día de hoy con lo que se acorta el número de clics. Al cambiar la fecha debemos pulsar el botón “genera informe” para que se genere el informe correspondiente a la fecha introducida.

Si añadimos un comentario al final, el informe debe volverse a generar y por tanto se debe volver a pulsar el botón “generar informe”.



El informe tiene una estructura que se puede ver en la figura:

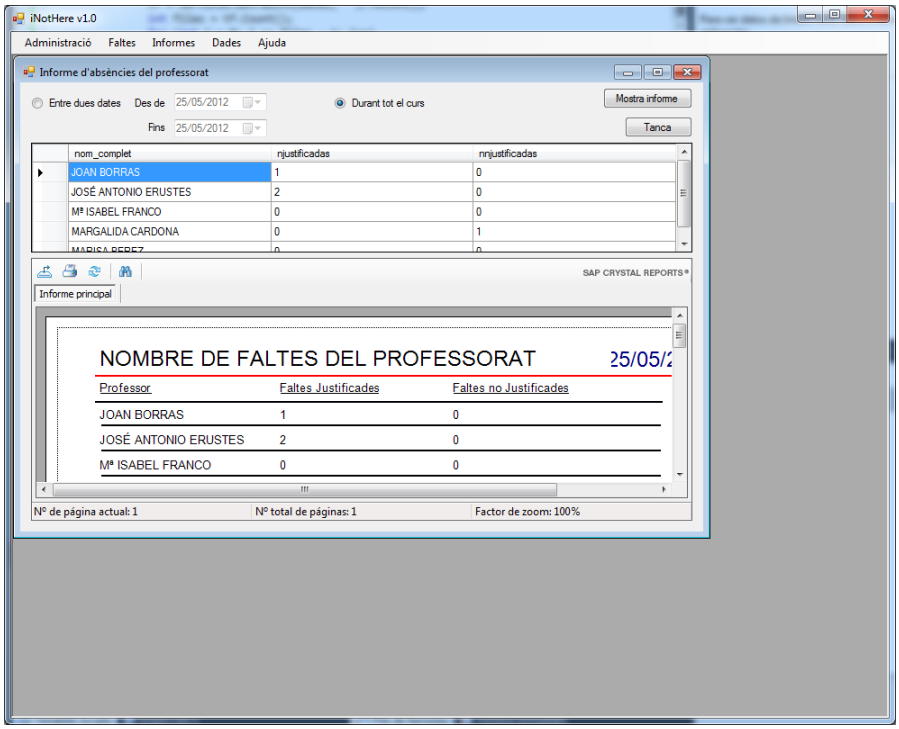


Donde cada número representa la franja horaria correspondiente. Desde la pantalla anterior, también es posible imprimir directamente el informe o exportar a pdf o muchos otros formatos habituales.

Informe d'absències.

En este informe se presentan el número total de ausencias justificadas y no justificadas de todo el profesorado que tenga faltas.

Este informe se presenta en pantalla y también puede ser exportado a varios formatos, entre ellos .pdf, e imprimido.



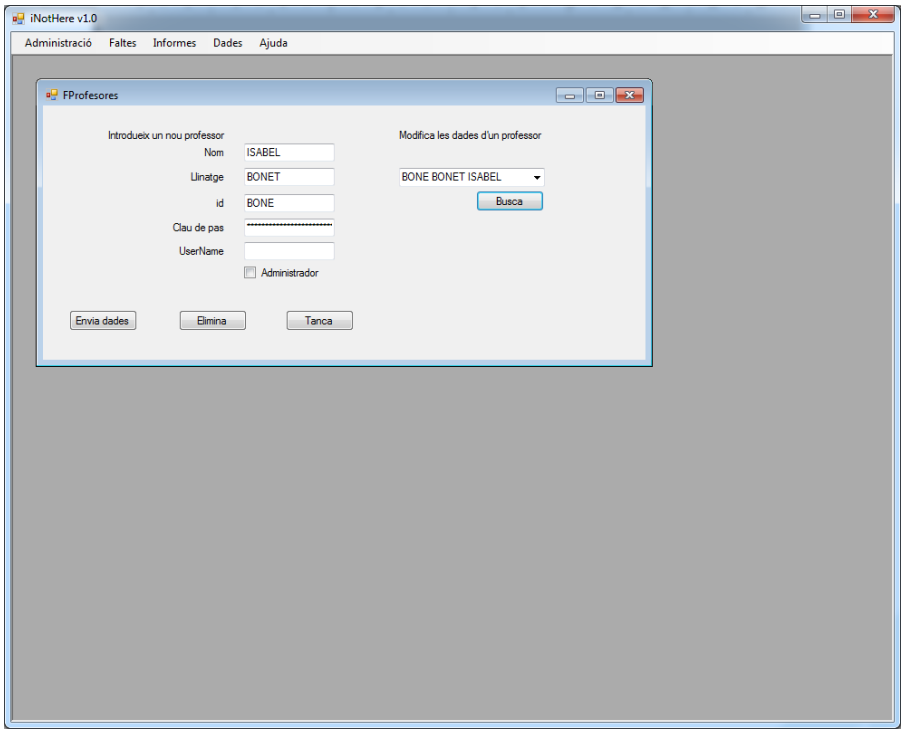
Este informe también permite seleccionar las fechas entre las que se quiere hacer el recuento de faltas, o bien seleccionar todas las faltas en cualquier fecha.

5.6.2d. Menú dades.

En este menú encontramos las opciones de modificar datos de aulas, profesores y asignaturas para una mayor flexibilidad.

Manteniment de professors.

Con esta opción podemos añadir profesores, eliminar profesores o modificar los datos de un profesor previamente seleccionado.



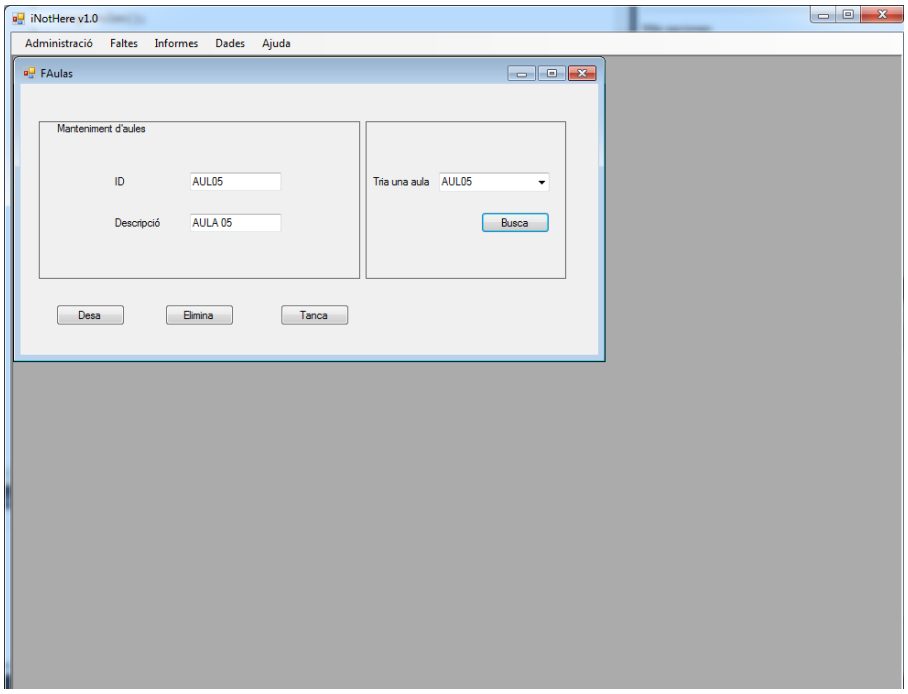
Para añadir un nuevo profesor a la base de datos, simplemente debemos introducir los datos del profesor y pulsar el botón "*Entrar dades".*

En el caso de que queramos modificar los datos de un profesor, primero debemos seleccionar uno de la lista, modificar los campos requeridos y pulsar el botón "*entrar dades"*. AL pulsar el botón el sistema nos preguntará pedirá una confirmación.

Para eliminar un profesor de la lista de profesores disponibles debemos seleccionar uno de la lista y pulsar el botón eliminar.

Manteniment d'aules.

Opción muy similar a la anterior pero en esta ocasión para el mantenimiento de las aulas. Podemos añadir, cambiar o eliminar aulas.



5.6.3. Zona de usuarios.

La estructura del menú es la siguiente.

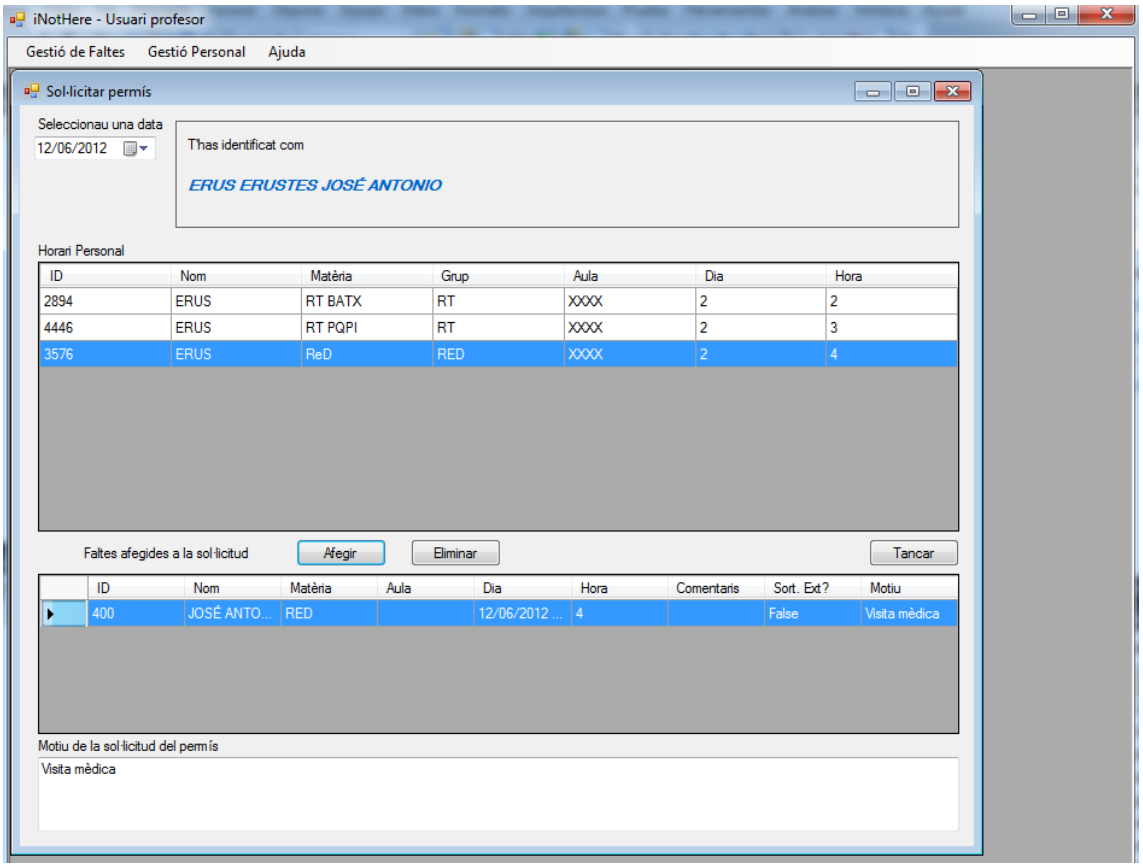
Gestió de faltes	Gestió personal
Sol·licitud de permís	Canvi de clau
Sol·licitud de justificació	Estat de les sol·licituds
Registre personal d'absències	
Sortir	

5.6.3a. Gestió de faltes.

En este menú se pueden realizar operaciones de gestión / solicitud de justificación y permisos a la dirección del centro.

Sol·licitud de permís.

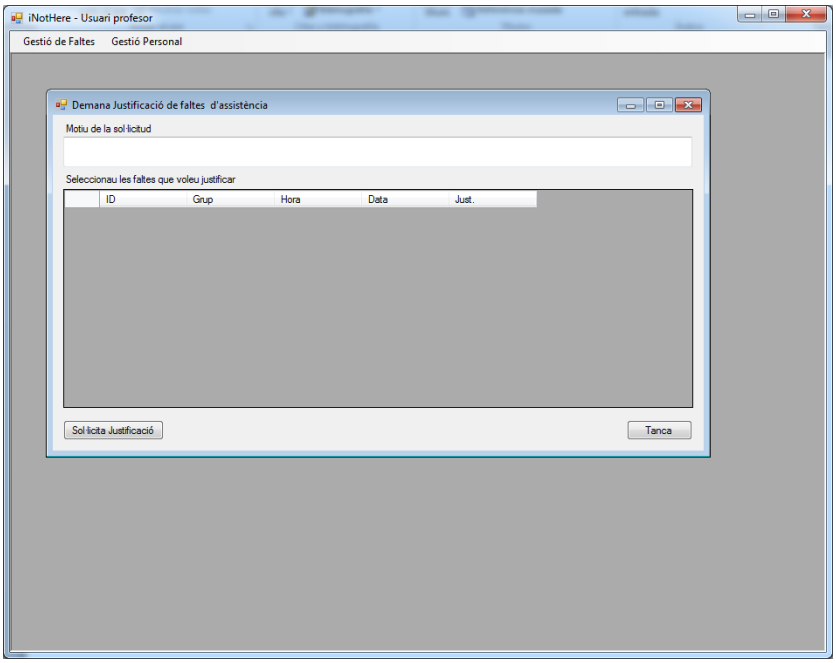
Mediante esta opción el usuario (profesor) puede solicitar permiso para no asistir a alguna o varias sesiones previstas. El usuario en primer lugar debe selccionar una fecha y automáticamente se cargará el horario lectivo para esa fecha. Después con los botones usuario puede añadir o eliminar sesiones.



Es imprescindible indicar un motivo justificado para la no asistencia a una sesión prevista. Una vez indicado el motivo, debemos pulsar el botón "*guardar dades*". En la propia ventana se detallan las instrucciones a seguir. Una vez enviada la solicitud no se pude eliminar.

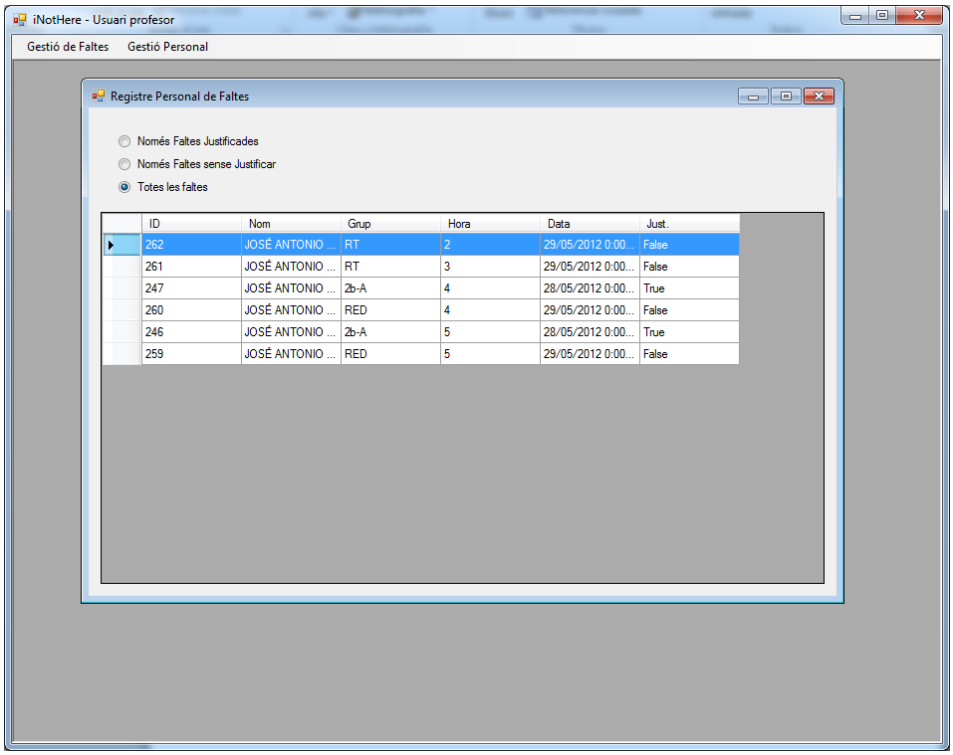
Sol·licitud de justificació de faltes.

Con esta opción el usuario puede solicitar que se le justifiquen las faltas por motivos establecidos en la normativa vigente. El usuario debe introducir un motivo y establecer las faltas que desea que se justifiquen. La solicitud llegará al administrador que decidirá si son justificables.



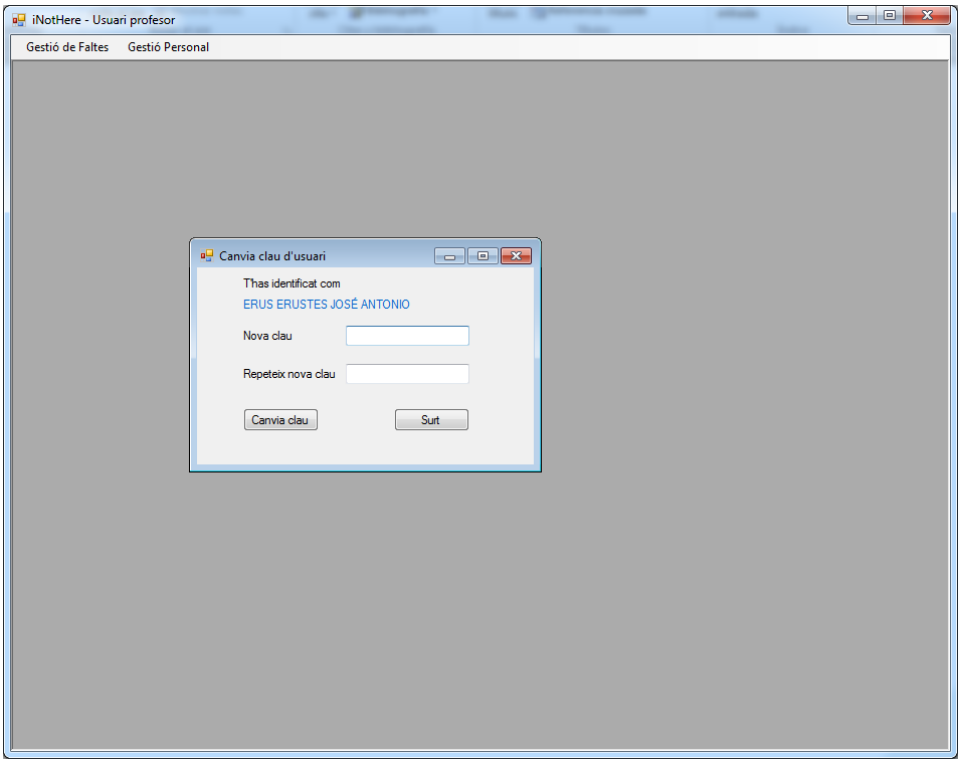
Registro personal.

En el registro personal se pueden ver las faltas justificadas, no justificadas y las faltas totales para que el usuario pueda tener un registro personal. Este registro no es imprimible por el usuario, solamente el administrador puede imprimir informes.



5.6.3b. Gestió personal

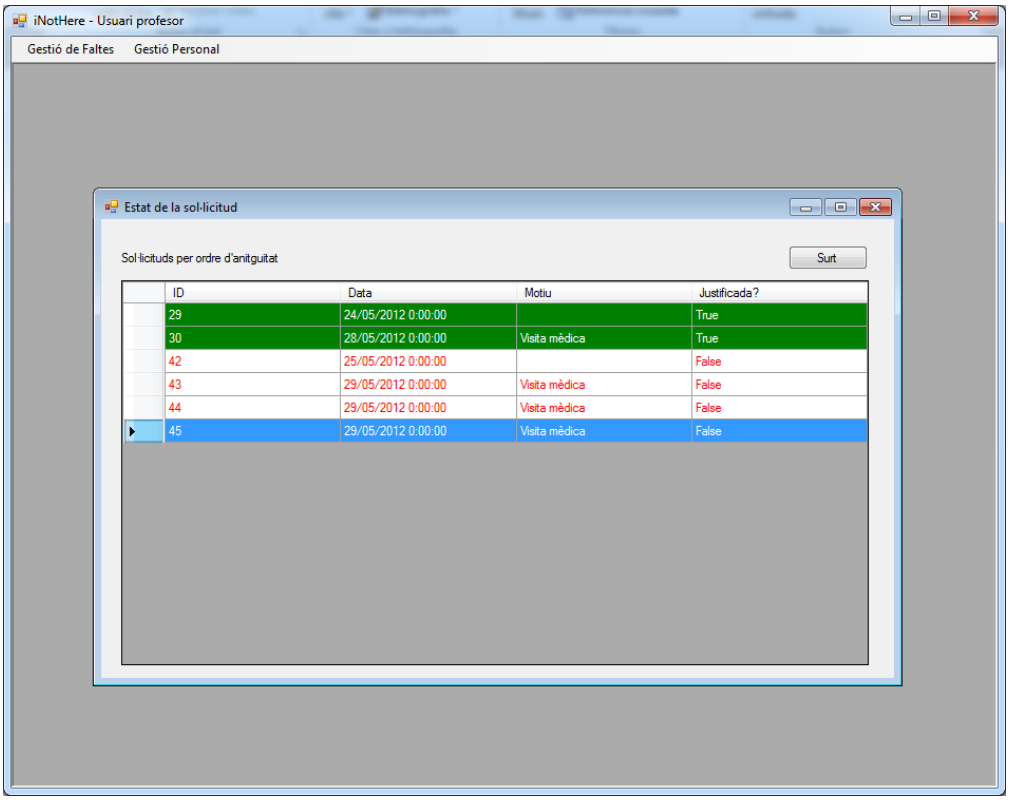
En este menú el usuario podrá gestionar aspectos básicos como el cambio de contraseña y el estado de las solicitudes.



Canvi de clau.

Con esta opción se puede modificar la clave del propio usuario. Esta clave no tiene ninguna restricción y se debe introducir dos veces. Irá encriptada en la base de datos con el algoritmo SHA2.

Estat de les sol·licituds.



Con esta opción el usuario podrá visualizar el estado de las solicitudes de justificación o de permiso. En verde aparecen las aceptadas. En el resto el texto es de color rojo.

5.7. Manual de instalación.

5.7.1. Recursos utilizados y requerimientos mínimos.

Ordenador con procesador P4 o superior con instalación de Windows XP/Vista/7.

Para el funcionamiento de la aplicación es necesario tener instalado:

- **Visual Studio 2010** o superior (2011 beta).
- **Microsoft SQL Server 2008 R2**.
- **Crystal Reports 14** para Visual Studio 2010. Se puede descargar de http://www.businessobjects.com/jump/xi/crvs2010/us2_default.asp
- **NET Framework 4.0** o superior.

5.7.2. Contenido del archivo comprimido.

El archivo iNotHere1.2.rar contiene la siguiente estructura:

- ✓ **…\iNotHere\DBAccess**
  - En esta carpeta se encuentra la lógica de conexión a la base de datos LINQtoSQL accesible desde el servicio WCF.
- ✓ **..\iNotHere\UsuarioAdministrador**
  - Aquí está la lógica de negocio con el proyecto de WindowsForms que se encarga de administrar la aplicación.
- ✓ **..\iNotHere\UsuarioNormal**
  - En esta ubicación encontramos el proyecto accesible a todos los usuarios.

- ✓ **..\iNotHere\ServicioWCF**
  - La carpeta contiene el servicio de windows comunication foundation (dll) que se encarga de la conexión entre la aplicación principal y la capa de datos.
- ✓ **..\iNotHere\SQL scripts**
  - Contiene los scripts de generación de la base de datos con la que trabaja el proyecto con datos de ejemplo para poder probar el programa.
- ✓ **..\iNotHere\Datos Importacion**
  - Contiene los archivos de texto para la importación de los datos a la base de datos que proceden de UNTIS (generador de horarios).

Por tanto la capa de presentación está formada por los proyectos UsuarioAdministrador y UsuarioNormal, desarrollado en C# con WindowsForms. La capa de comunicación está diseñada con WCF en C# y finalmente la capa de datos está formada por el proyecto DBAccess que administra el acceso a los datos mediante LINQtoSQL.

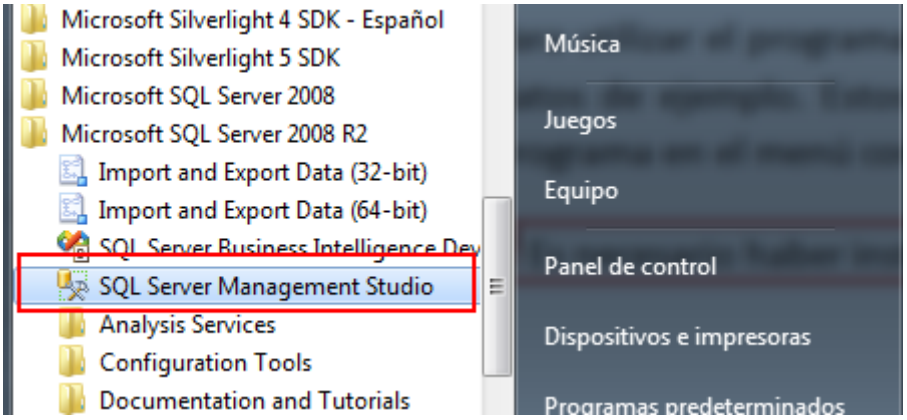
5.7.3. Instalación de la aplicación.

5.7.3a. Creación de la base de datos.

Para utilizar el programa primero es necesario crear la base de datos e introducir una serie de datos de ejemplo. Estos datos de ejemplo pueden ser sustituidos durante la ejecución del programa en el menú correspondiente.

❗ Es necesario haber instalado Microsoft SQL Server 2008 con anterioridad.

1. Ejecutamos MS SQL Server Management Studio



2. Introducimos los datos necesarios para la conexión y pulsamos **conectar**.

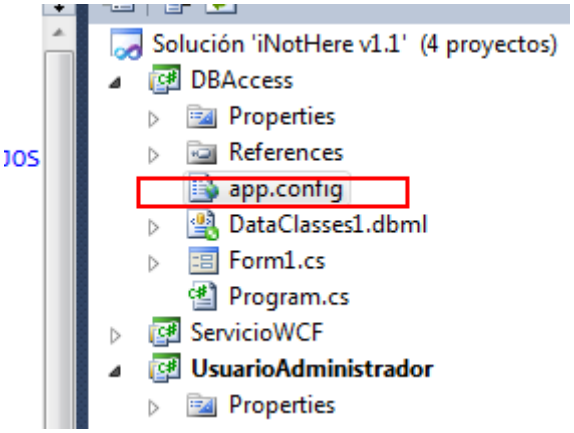
3. Seleccionamos Archivo → Abrir y seleccionamos el script SQL que se encuentra en la carpeta correspondiente. Esto generará las tablas y los datos de ejemplo de la aplicación.

5.7.3b. Instalación de Crystal Reports 14 para Visual Studio 2010.

Descargamos desde aquí el instalador de CR 14 y lo instalamos de la manera habitual.

5.7.3c. Configuración del proyecto de acceso a datos DBAccess.

Abrimos Visual Studio 2010 (en adelante VS2010) y buscamos el archivo app.config.



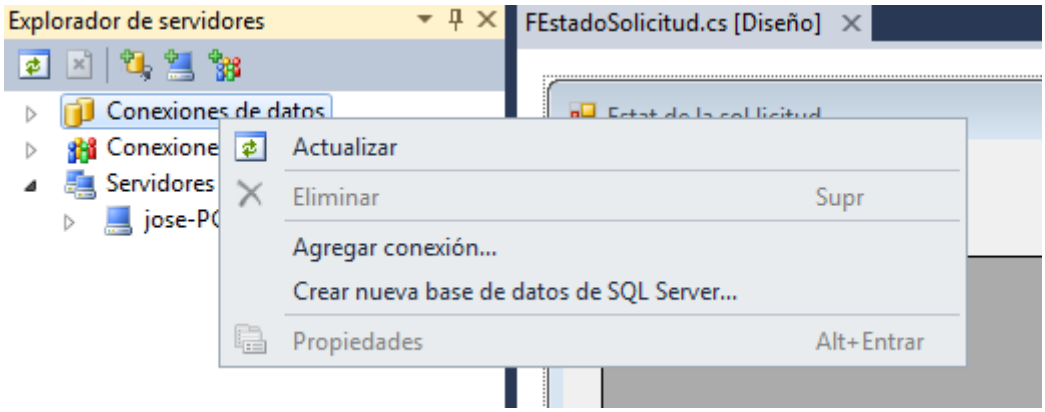
Seguidamente modificamos el parámetro *ConnectionString* como se observa en la imagen, para que se adapte a nuestras necesidades:

```
<?xml version="1.0" ?>
<configuration>
    <configSections>
    </configSections>
    <connectionStrings>
        <add name="DBAcess.Properties.Settings.iNotHereConnectionString"
            connectionString="Data Source=JOSE-PC;Initial Catalog=iNotHere;Integrated Security=True"
            providerName="System.Data.SqlClient"/>
    </connectionStrings>
<startup><supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.0"/></startup></configuration>
```

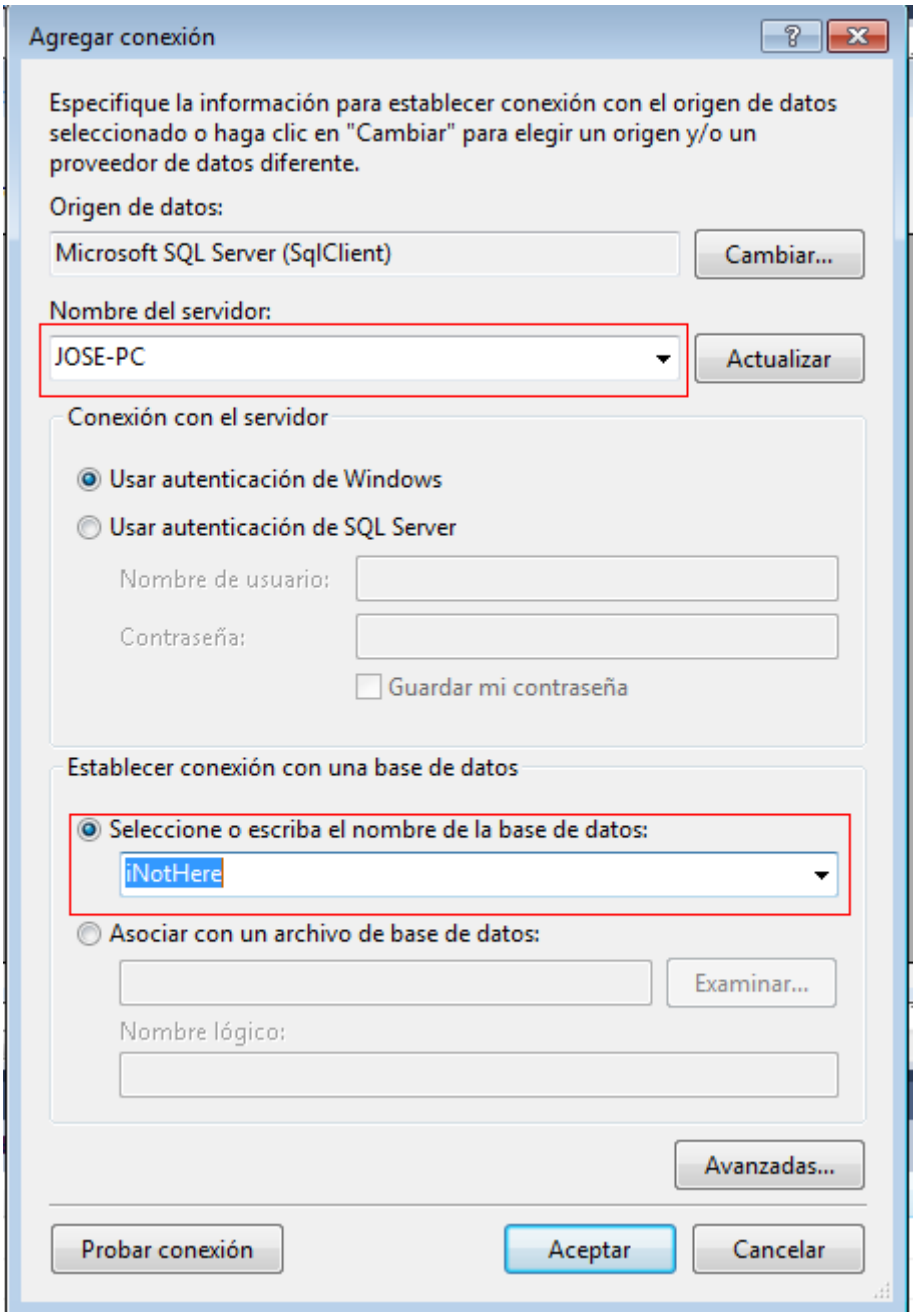
En este caso, debemos sustituir la cadena que hay después de Data Source=". Para localizar exactamente que el nombre debemos abrir MS SQL Server 2008 y comprobar el nombre del servidor:

Una vez comprobado el nombre se cambia en el archivo app.config y se guarda.

Para concluir es necesario añadir una nueva conexión:

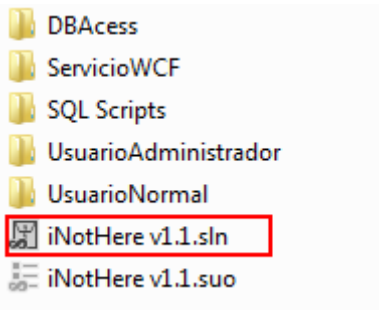


Posteriormente elegimos el nombre del servidor y el nombre de la base de datos, que en nuestro caso será iNotHere:



5.7.3d. Acceso al archivo de la solución.

Finalmente, debemos abrir el archivo de la solución, en este caso iNotHere v1.1.sln com ose puede ver en la figura.

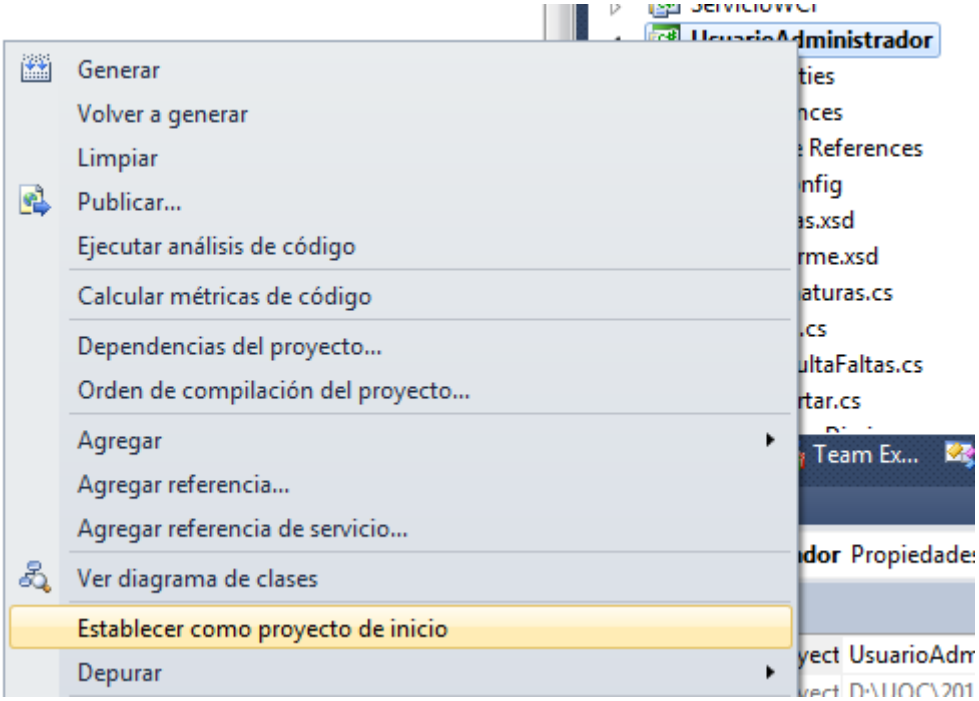


En este caso, hacemos doble clic en el archivo que se muestra en la figura y a continuación, se abrirá la versión de VS que tengamos instalada. En este caso la versión VS2010.

Para ejecutar el programa, simplemente deberemos pulsar F5 con lo que se lanzará el servicio WCF y se ejecutará el software.

Todos los usuarios de los datos de ejemplo tienen la contraseña “123456” y el usuario administrador corresponde al usuario “ERUS ERUSTES JOSE ANTONIO”. El usuario administrador podrá entrar en las dos interfaces mientras que el usuario restringido solo podrá acceder al programa de usuario.

Para arrancar uno u otro programa debemos colocar el puntero del ratón encima del proyecto que queremos iniciar. En caso de que no podamos acceder al proyecto deseado haremos clic izquierdo sobre el proyecto que deseamos arrancar tal como se ve en la figura, y seleccionaremos “Establecer como proyecto de inicio”.



6. Objetivos conseguidos.

Finalmente después del proceso de formación continua, planificación, diseño, desarrollo del proyecto y depuración he obtenido un producto que ya ha sido probado durante estos días en un entorno de producción y que supone una mejora importante en la gestión de ausencias del profesorado. Si bien el ámbito de aplicación de este proyecto no es muy ámplio, si que puede ser útil a otros centros como el nuestro en el que se hace imprescindible un control estricto y una buena transmisión de información al profesorado.

Dentro de este punto, objetivos conseguidos, creo que es justo añadir aspectos mejorables del proyecto, ya que además este proyecto ha sido probado a lo largo de los últimos días en un entorno real.

6.1. Objetivos conseguidos.

Uno de los objetivos más importantes planteados al principio de este proyecto era diseñar un producto que pudiera satisfacer las necesidades y que tuviera las características explicadas con anterioridad. Si bien el proyecto tiene posibles mejoras (que ya explicaré posteriormente), creo que los objetivos se han cumplido y además el producto será utilizado (ya ha sido probado) en un entrono de producción, lo cual a mi modo de ver, es un aspecto que realza el producto.

6.2. Aspectos mejorables.

Después de algunas pruebas reales, he detectado que evidentemente el proyecto tiene un amplio margen de mejora.

En primer lugar me planteo la posibilidad de que si no hubiera sido más útil y cómodo para el usuario final, tener una aplicación web (desarrollo con ASP.NET) en lugar de una aplicación de escritorio. Tanto para el administrador como para el usuario. Es un aspecto a tener en cuenta ya que el proyecto podría estar integrado en el marco de una intranet, proyecto mucho más ambicioso y que queda fuera de los objetivos de este trabajo. En el punto *trabajos futuros* volveré sobre este punto.

La solución desarrollada está destinada a ser implantada en una intranet corporativa donde los usuarios abren una sesión de windows contra un servidor de *Active Directory* Windows Server 2003. Por tanto creo que hubiera sido mejor que los usuarios se pudieran conectar automáticamente a la aplicación con su nombre de usuario de AD. Esto hubiera hecho mucho más cómodo tanto para el usuario como para el administrador la autenticación.

Por otra parte el programa tiene algunos aspectos a nivel de usabilidad que también podrían mejorar para disminuir el número de clics de ratón e incrementar la velocidad de manejo, si bien he hecho un esfuerzo para que la usabilidad sea buena.

7. Evaluación de costes.

El coste de este proyecto se puede entender de dos formas diferentes. La primera forma de calcular los costes es suponiendo que ya se dispone del hardware y la infraestructura necesaria. Esta suposición está justificada en el hecho de que este software se dirige a un segmento muy determinado como es el de los centros educativos en donde la dotación mínima siempre incluye un servidor Windows 2003 Server con características suficientes para una red con 200 IP privadas. Además incluye las licencias de Windows XP/7.

Todo esto hace que en este únicamente se debe tener en cuenta el trabajo y las licencias del software de apoyo. Por tanto:

Producto	Cantidad	Precio
SQL Server Express 2008	1	Gratuito
Visual Studio 2010 Express	1	Gratuito
Crystal Reports 14	1	Gratuito
Total		**0 €**

En el caso de un entorno de desarrollo mucho más profesional se requiere una inversión en licencias del software con versiones mucho más funcionales:

Producto	Cantidad	Precio
SQL Server Standard Edition 2012	1	2500 €
Visual Studio 2010 Professional	1	550 €
Crystal Reports 14	1	Gratuito
Total		**3050 €**

Por otra parte en cuanto a los servicios tenemos la planificación que se puede ver en la tabla. En ella se detalla que se ha trabajado durante 76 días que corresponden a los meses de marzo, abril y mayo en gran parte y si descontamos los fines de semana (13 en total) obtenemos que los días trabajados son 76-(13x2) = 50 días.

Si trabajamos una media de 5h en el proyecto cada día y el precio por hora son 50 € tenemos que el coste de los servicios de desarrollo es de 12.500 €.

	TFC .NET - iNotHere	76 días
	Fase 1. Plan de Trabajo	9 días
	Fase 2. Especificación y diseño	23 días
	Fase 3. Implementación de la Aplicación	33 días
	Fase 4. Memoria Final	12 días

A esto debemos sumarle los servicios extraordinarios como traducción que a un precio estimado de 0,07 € por palabra con unas 100 palabras tenemos 7 €.

8. Trabajos futuros.

Es evidente que el margen de mejora de este proyecto es amplio y que puede ir en varias direcciones, aunque por otra parte creo que el proyecto ha cumplido los objetivos planteados en un principio.

Los trabajos futuros pueden ir en dos direcciones básicamente. En primer lugar aprovechar todas las funcionalidades obtenidas y crear un proyecto más amplio con más funcionalidades ya sea utilizando tecnología ASP.NET o Windows Forms, como por ejemplo añadir una sección para actividades extraescolares, un calendario de reuniones etc.

En este sentido puede que al añadir funcionalidades fuera mejor usar tecnología ASP.NET con el fin de permitir un acceso mucho más sencillo a los usuarios que ya están acostumbrados a utilizar un navegador para algunas aplicaciones corporativas. Por otra el alojamiento web actual debería revisarse ya que funciona bajo sistema operativo Linux. En la intranet corporativa no habría ningún problema pero hay que tener en cuenta que la intranet corporativa no es accesible desde el exterior, ni está previsto que lo sea a corto o medio plazo.

Y en segundo lugar, es necesario reconocer que el cada vez más presente sistema operativo Ubuntu en los centros docentes hace que este software deba ser revisado (si bien existen plataformas como MOMO que hacen compatible el desarrollo con .NET en Linux).

Añadir nuevas funcionalidades como mensajería interna, eliminación de la pantalla de autenticación gracias a la integración con los servicios Active Directory de Windows u otras funcionalidades, también son buenas opciones para desarrollos futuros.

9. Conclusiones.

He de reconocer que mis conocimientos iniciales sobre la tecnología .NET eran muy escasos y mi experiencia previa, nula. Es por esa razón que durante todo el proceso de elaboración de las diversas partes del proyecto, me he visto obligado a una tarea paralela de aprendizaje y familiarización con la tecnología de este proyecto.

Por otra parte, es la primera vez que me enfrento a un proyecto de similar envergadura. Hasta la fecha únicamente había desarrollado aplicaciones basadas en línea de comandos. La organización y el diseño del proyecto son fundamentales y lo he podido comprobar de manera fehaciente durante estos meses.

Sobre la tecnología .NET puedo decir que estoy gratamente sorprendido. El entorno que ofrece Visual Studio 2010 es fantástico y el desarrollo se hace mucho más sencillo que con otros IDE que he podido experimentar durante mi fase de formación en la ingeniería. La cantidad de clases y librerías disponibles y más aún, la enorme disponibilidad de documentación en formatos múltiples también me han dejado gratamente sorprendido.

En cuanto al proyecto en si, he experimentado varias dificultades. La primera es que he tenido que enfrentarme por primera vez a la programación de un servicio. La comunicación de la capa de presentación con la capa de datos ha sido una de las dificultades más importantes.

La segunda dificultad para mi ha sido enfrentarme a LINQ, cuya potencia ahora reconozco pero que ha requerido un esfuerzo suplementario por mi parte, puesto que como he dicho antes, desconocía su manejo y su existencia.

Finalmente la creación de reportes con Crystal Reports me ha parecido muy interesante pese a que creo que la herramienta no es todo lo flexible que podría, sobretodo a nivel visual.

10. Bibliografía.

Manual del usuario de Untis 2012 (programa de horarios).
http://www.grupet.at/espanol/support/webhelp/untis/index.html

Curso de iniciación al C#
http://www.elguille.info/NET/cursoCSharpErik/Entrega1/Entrega1.htm

Área C#
http://www.ehu.es/mrodriguez/Area-CSharp.html

Tutoriales ADO.NET
http://www.ehu.es/mrodriguez/videosBBDD.html

Tutoriales y herramientas de aprendizaje Microsoft TechNet
http://technet.microsoft.com/es-es/bb291022

ENCICLOPEDIA DE MICROSOFT VISUAL C# (3ª ED.) (EN PAPEL)**,** FRANCISCO JAVIER CEBALLOS, RA-MA, 2010