

Agenda deportiva

Autor: Julián Ruiz Burgos

Tutor: Xavier Roca Marva

Índice

1. Introducción
2. Análisis de requerimientos
3. Diseño
4. Base de datos
5. Interfaz
6. Pruebas
7. Instalador y Ejecutable
8. Conclusiones y Vías de continuación

Introducción:

Objetivo y Motivaciones

- Objetivo:
 - Crear una aplicación relacionada con el mundo deportivo, concretamente una agenda deportiva.
- Motivaciones:
 - Unir formación e interés en un mismo proyecto.
 - Resultado del proyecto útil.

Introducción:

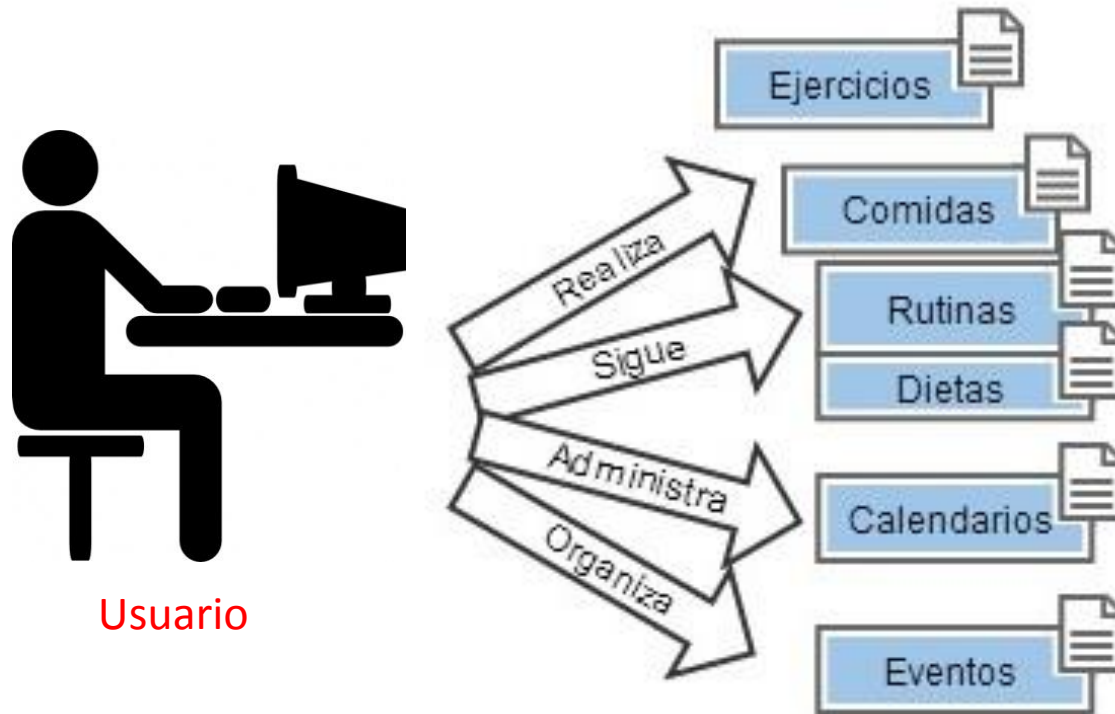
Estado del arte

- Qué existe hoy en día?
 - Aplicaciones deportivas orientadas a dispositivos móviles. Permiten registrar la actividad realizada.
 - Aplicaciones web.
- Qué no ofrecen?
 - Unir en una aplicación de escritorio la planificación personalizada del entrenamiento del usuario.

Introducción:

Descripción de la Aplicación

- *Agenda deportiva* que permite al usuario administrar sus actividades y elementos relacionados con su práctica deportiva.

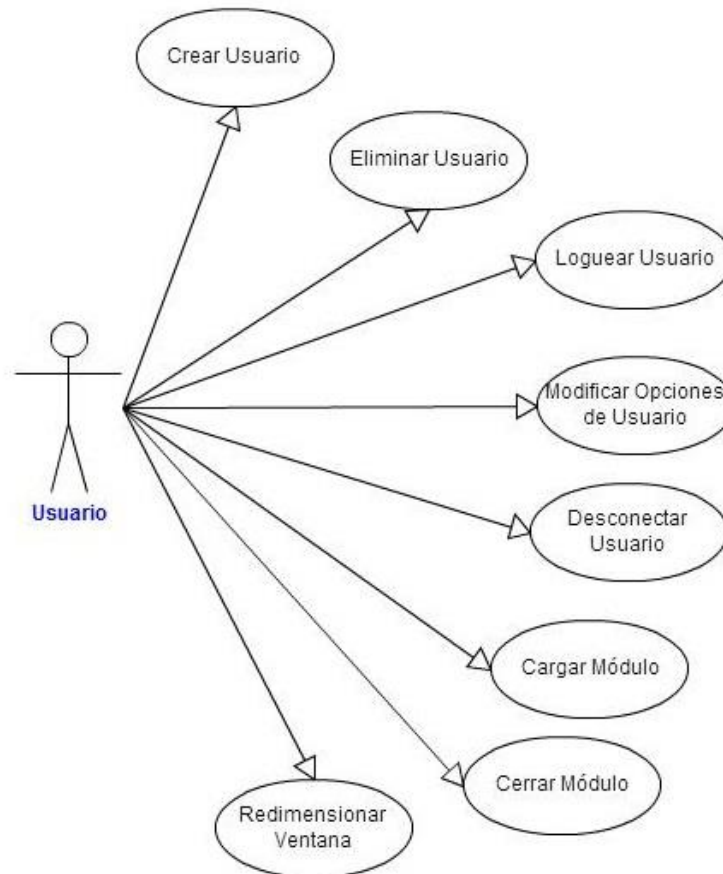


Análisis de Requerimientos: Stakeholders

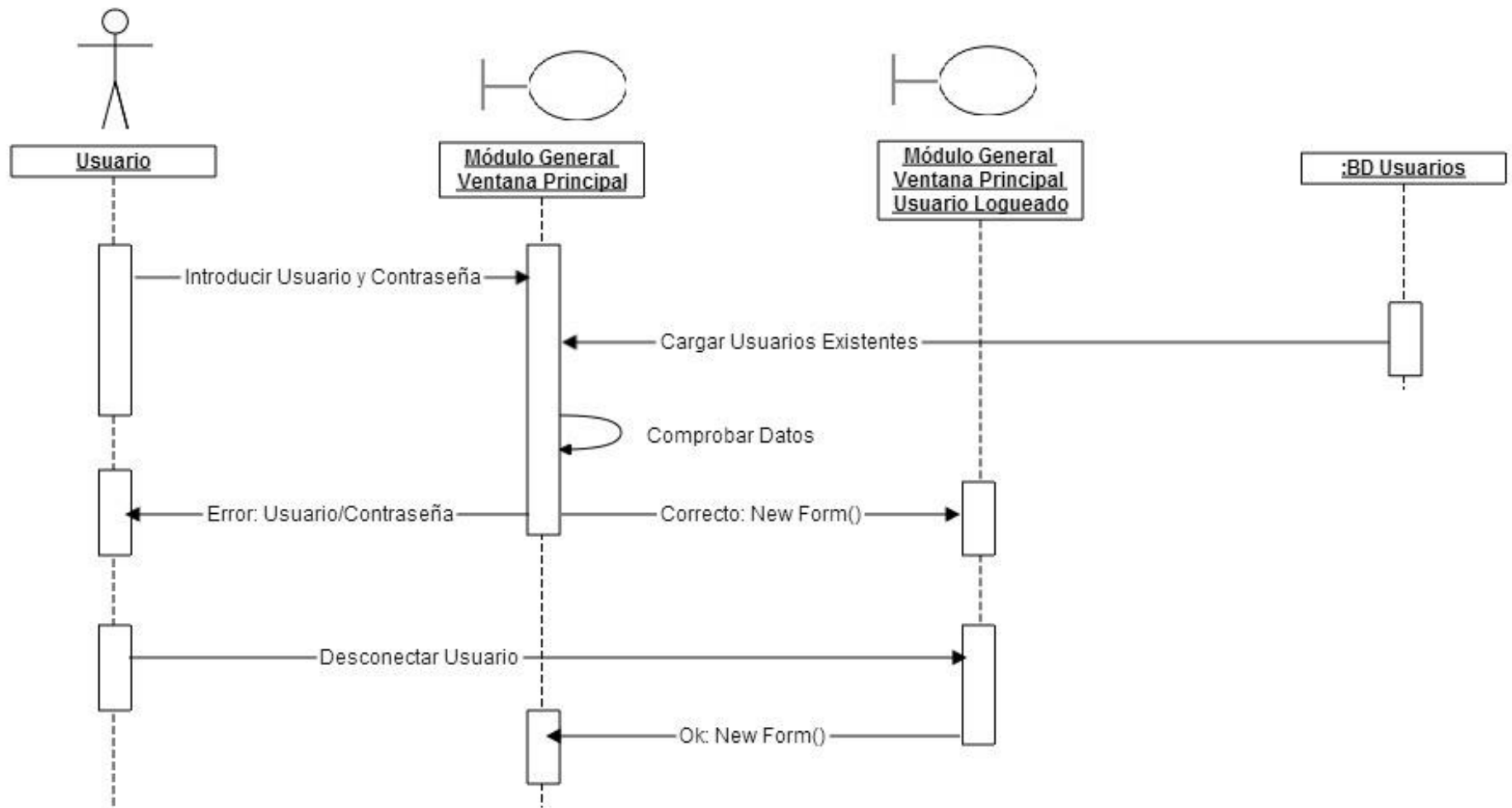
- Identificación de los interesados
 - Deportistas, Entrenadores, Monitores, etc.
 - Desarrollador del proyecto.
- Beneficios de ser a la vez interesado y desarrollador
 - Enfoque como **desarrollador**: Qué quiero que **ofrezca** el programa a los usuarios?
 - Enfoque como **usuario**: Qué quiero que me **permita** hacer la aplicación?

Análisis de Requerimientos: Requerimientos funcionales

- Modelo de casos de uso

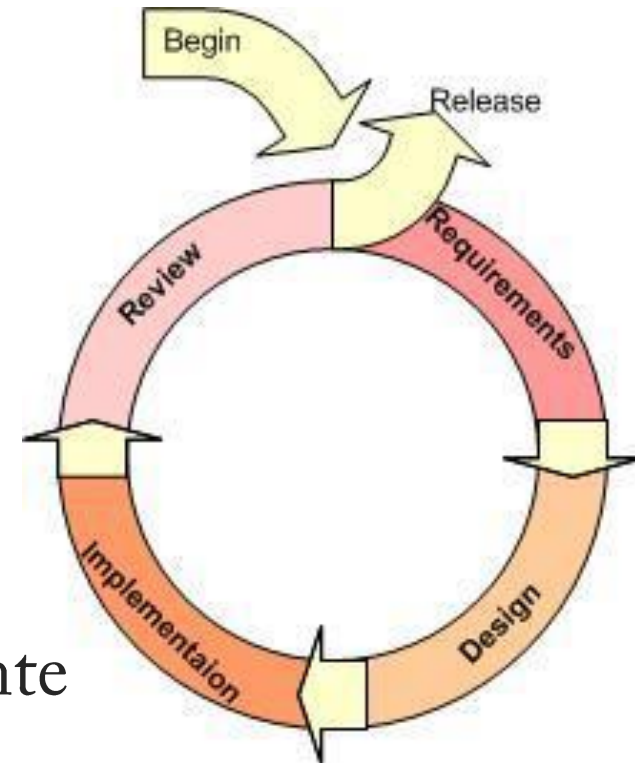


Análisis de Requerimientos: Diagramas de secuencia



Diseño: Método de desarrollo

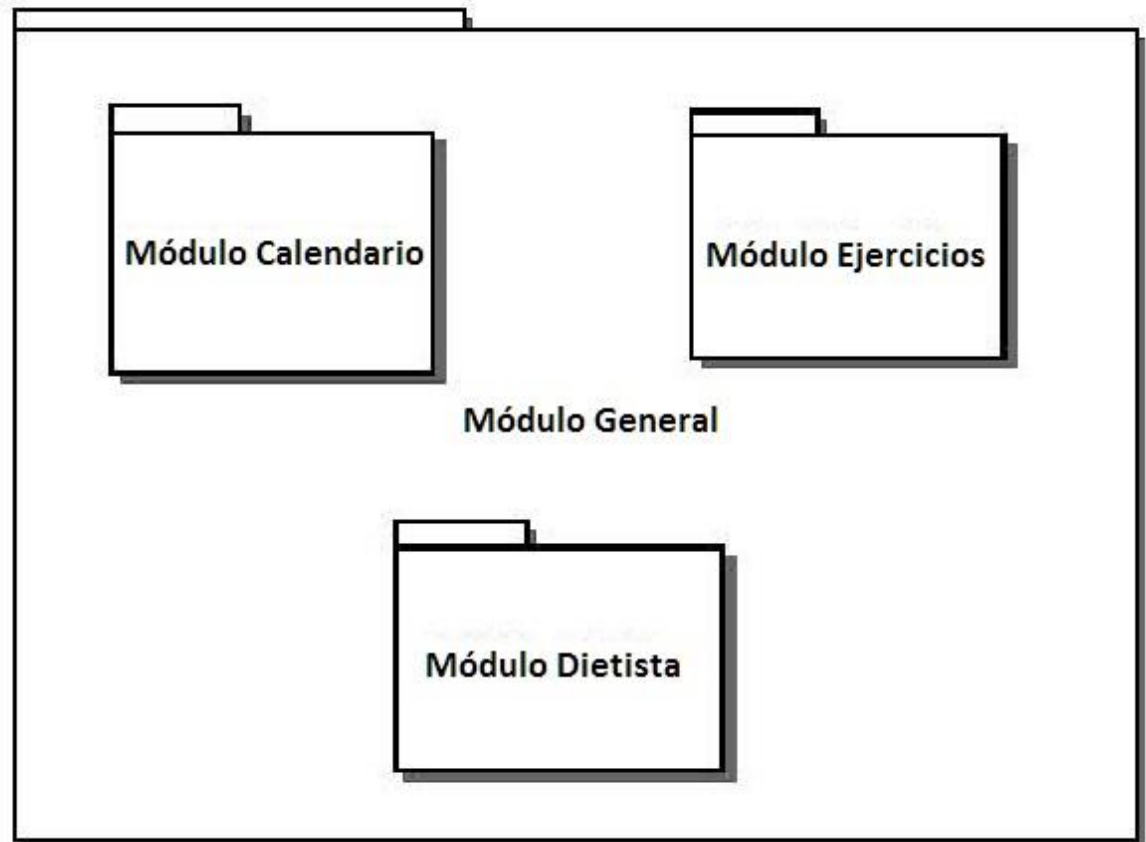
- Iterativo e Incremental
- Tiempo mejor empleado conforme avanza el proyecto.
- Módulos desarrollados posteriormente mejor diseñados e implementados.



Diseño:

Arquitectura de la aplicación

- Cuatro Módulos :
 - General
 - Calendario
 - Ejercicios
 - Dietista



Base de Datos: Tipo

SQL

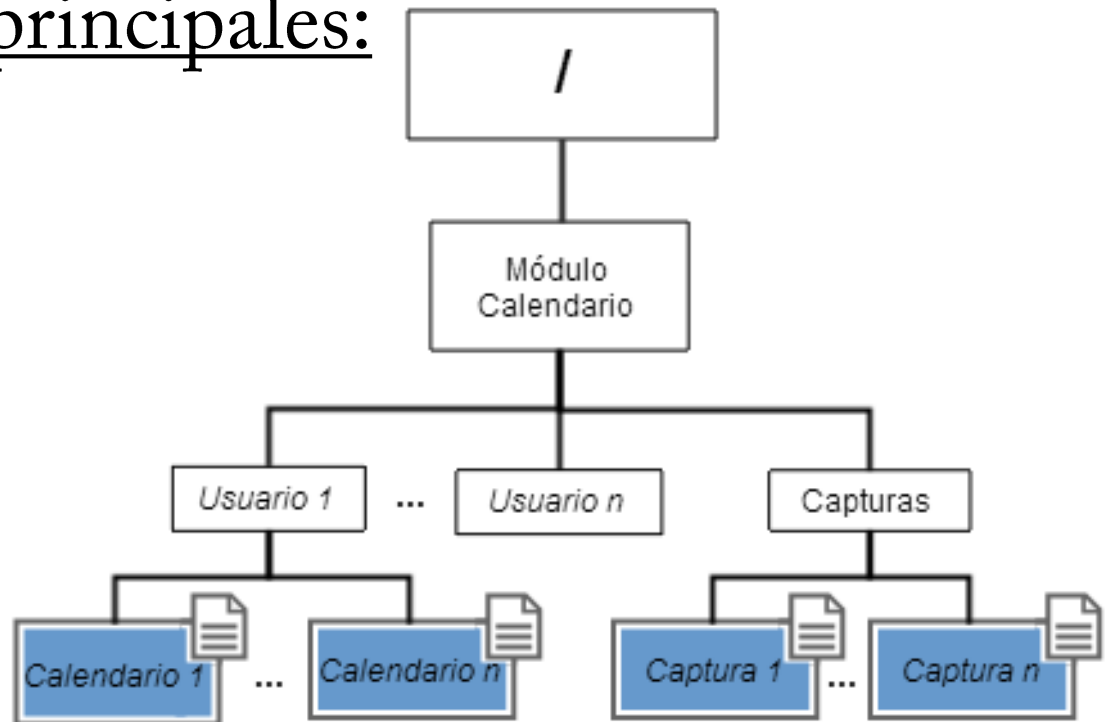


NoSQL

Base de Datos: Diseño e Implementación

- Dos elementos principales:

- Directorios
- Ficheros



Base de Datos:

Estructura de los ficheros

- Cada línea separa sus campos mediante separadores.

1r Nivel	;
2º Nivel	
3r Nivel	#
4º Nivel	} <i>Para evitar conflictos con el formato de fecha dd/mm/aa</i>
Módulo Calendario	
Resto de Módulos	
5º Nivel	{
6º Nivel	[

```

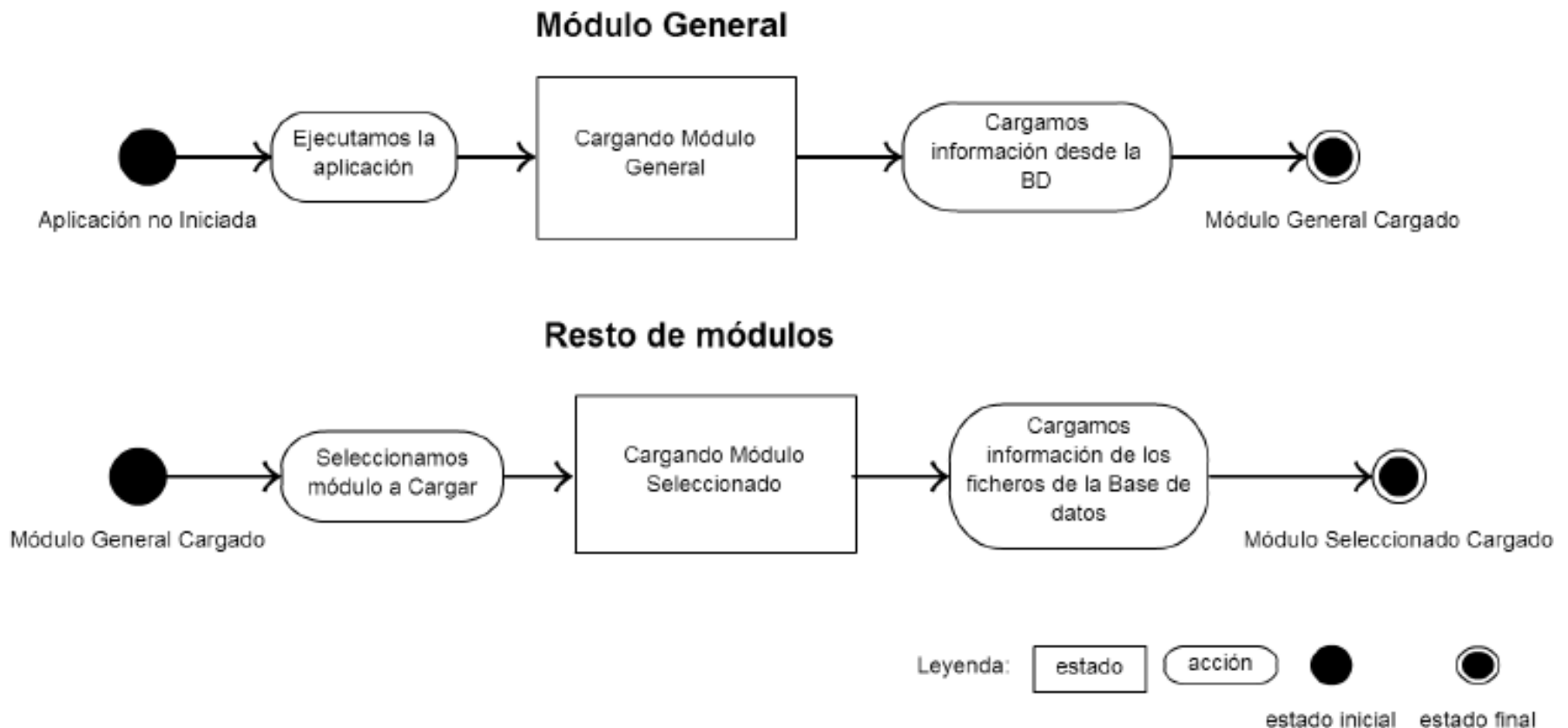
1 FechaCalendario
2 06/07/2013 12:00:00
3 Tipo;Nombre;Fecha (MM/dd/aaaa HH/mm);Duración(horas);Frecuencia;IgnorarCc
4 4;Dieta Julian;1#2}Comida Base 1}06/24/2013 08:00:00}1}5}False}False}True
5 5;Cachimba;1#3}Flexiones Simples[2[2[0[2}06/24/2013 08:00:00}0.25}5}Fal
6 2;Comida Base 1;07/22/2013 09:00:00;1;5;False;False;True;True;255|255|255
7 1;Proyecto sesión prueba creación evento;06/07/2013 15:00:00;4;1;False;Fa
8 3;Flexiones Simples[3|30|0|0|30;06/03/2013 17:30:00;0.5;3;False;False;Tru

```

Base de Datos:

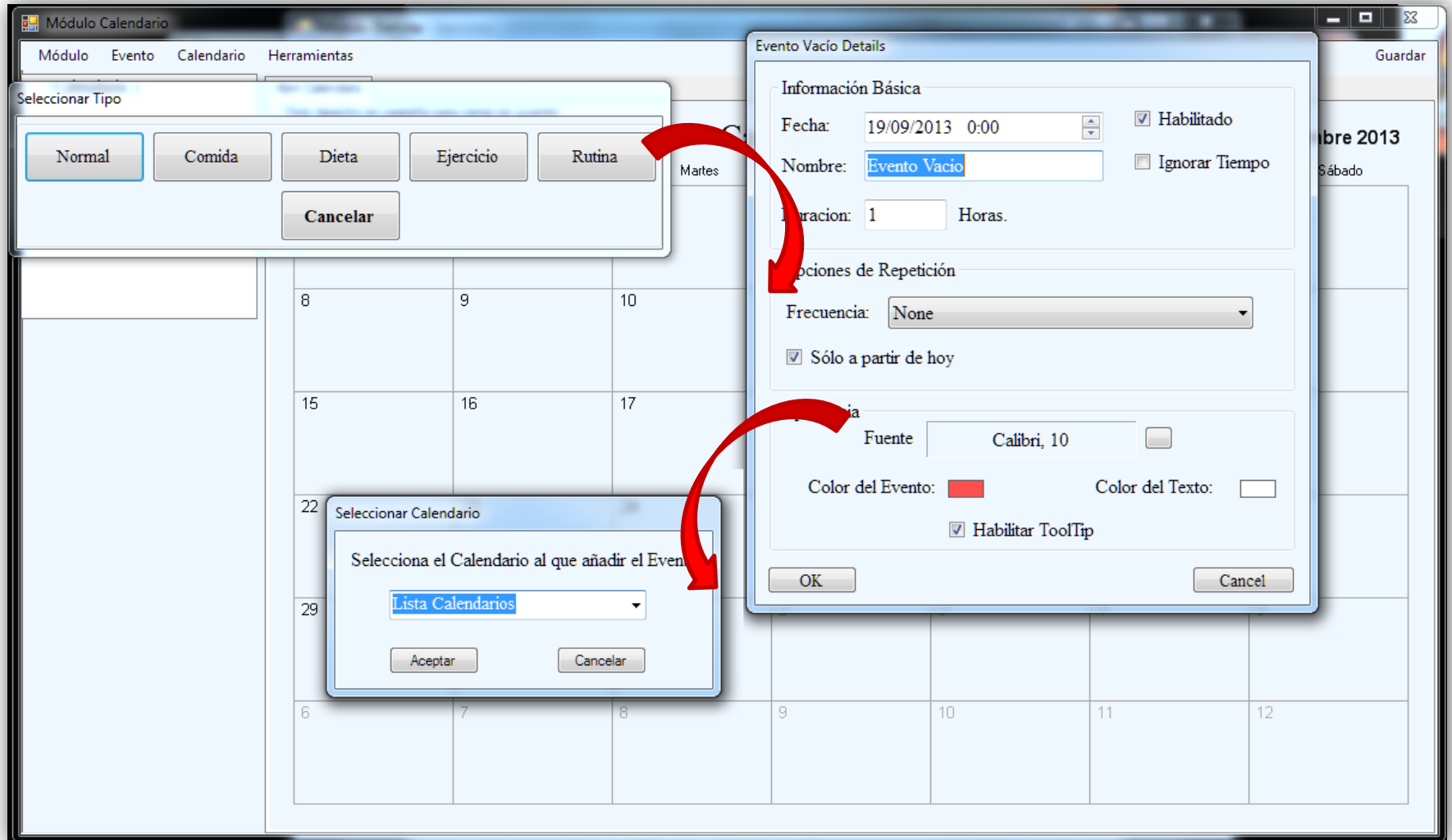
Interacción con la aplicación

- Al cargar un módulo:



Interfaz

- Tres Pasos



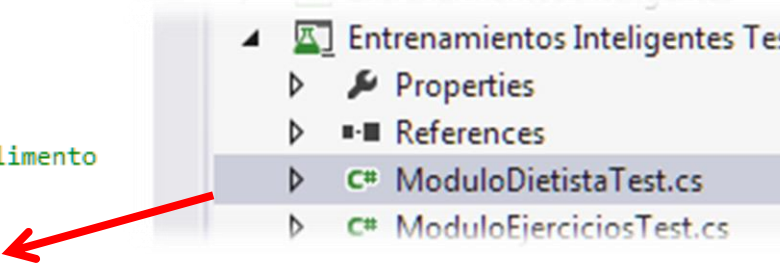
Pruebas

- Pruebas Unitarias (Unit Tests)

```
[TestClass]
public class ModuloDietistaTest
{
    // Tests correspondientes a la clase Alimento
    [TestMethod]
    public void CreacionDeAlimento()
    {
        string Nombre = "Patata";
        string Descripcion = "El tubérculo más conocido del mundo. Proveniente de América";
        string tipo = "Tubérculo";
        int cantidad = 150;
        int aporteEnergetico = 30;

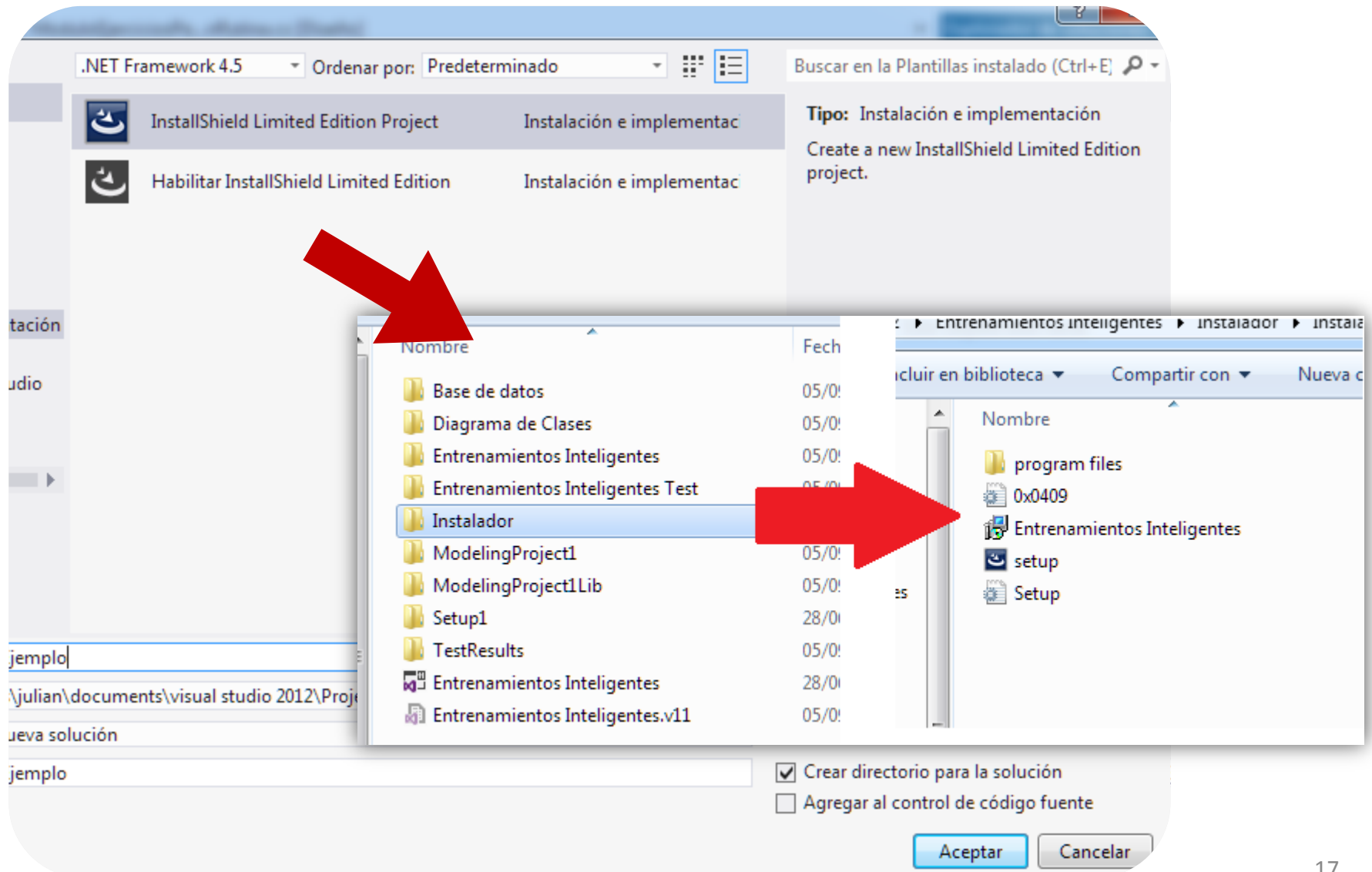
        Alimento Patata = new Alimento(Nombre, tipo, Descripcion, cantidad, aporteEnergetico);

        Assert.AreEqual(Patata.Nombre, Nombre);
    }
}
```



- Comprobación del funcionamiento esperado por la aplicación

Instalador y ejecutable



Conclusiones y Vías de continuación

- ✓ Objetivo cumplido
- ✓ Nuevos conocimientos adquiridos
- Vías de continuación?
 - Base de datos **SQL**.
 - Versión **WEB/Móvil** de la aplicación.
 - Nuevos módulos.

ANEXO

Herramientas utilizadas

- Microsoft Visual Studio 2010
 - Licencia gratuita por la universidad.
 - Herramientas para crear la Interfaz.
 - Uso intuitivo.
- Internet
 - Fuente inagotable de información.

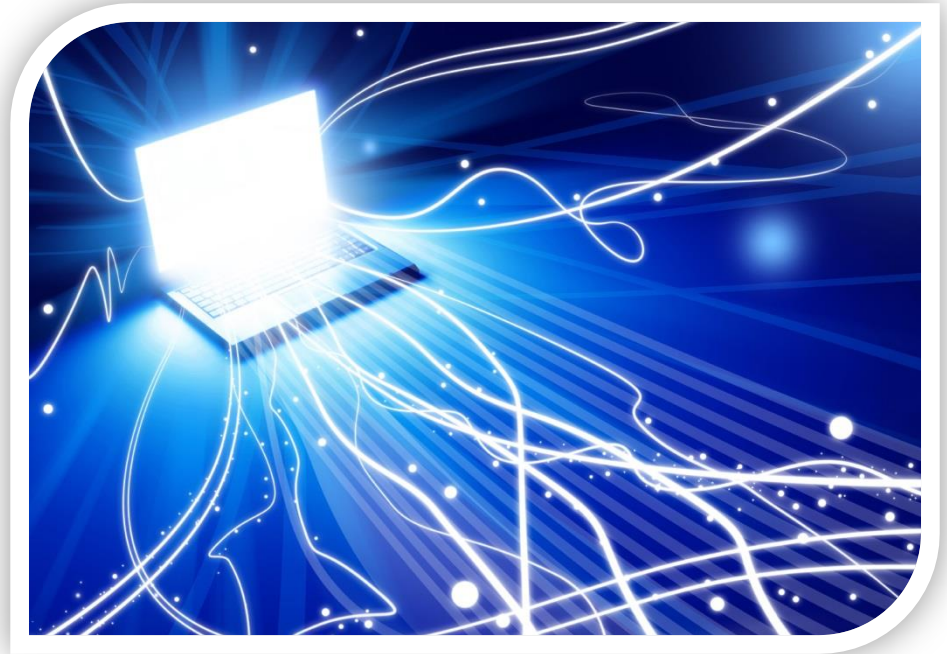
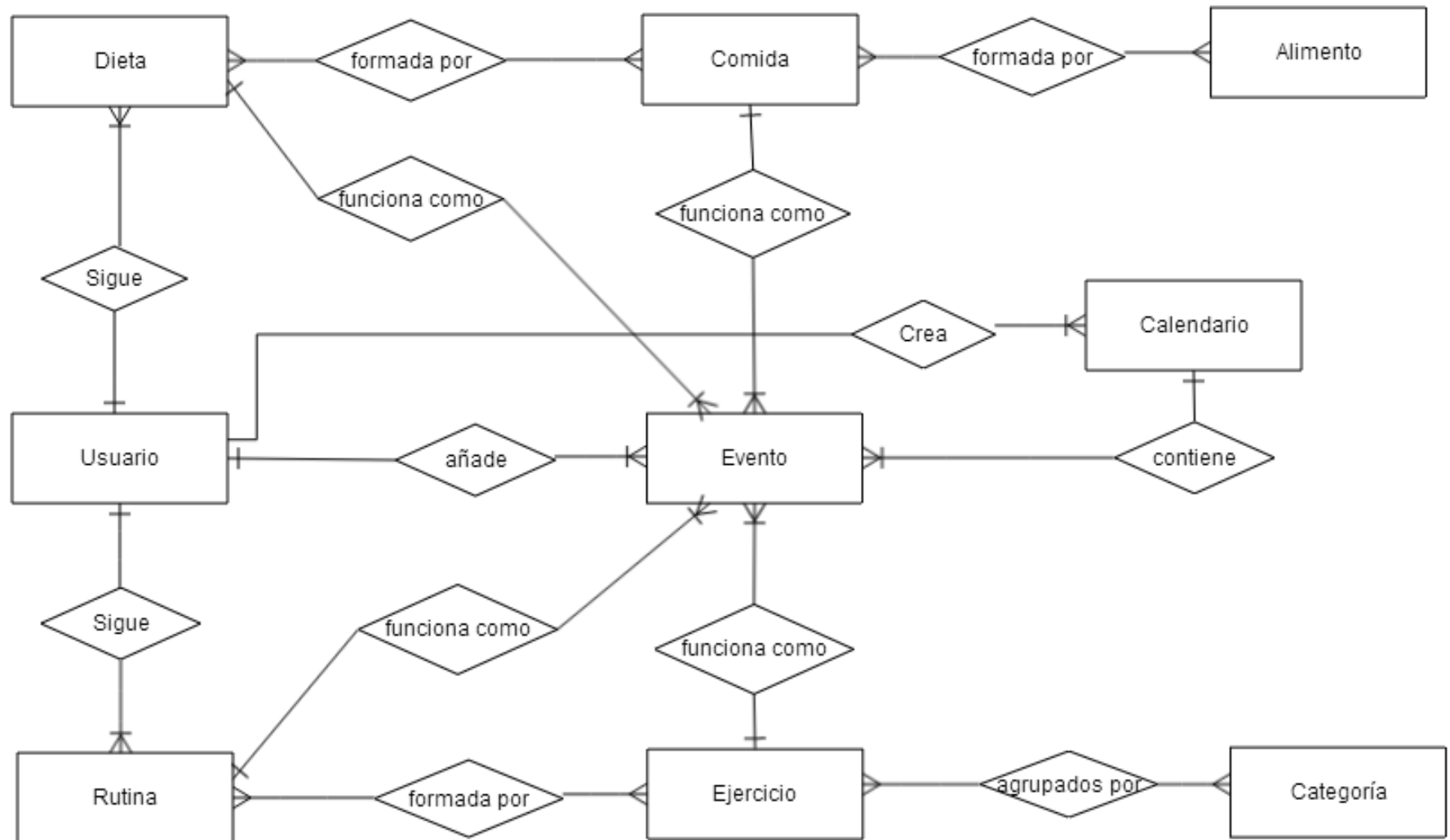
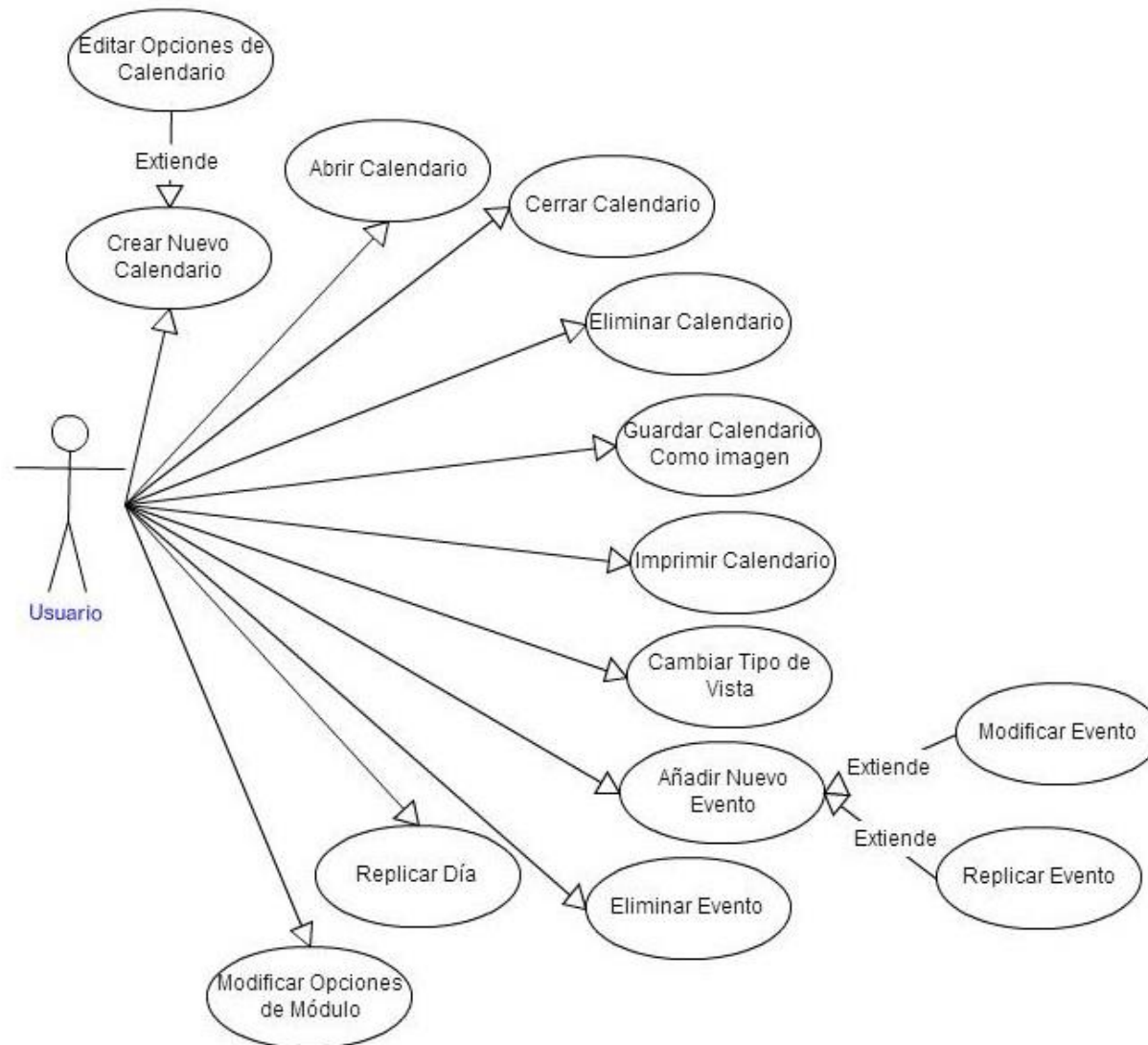


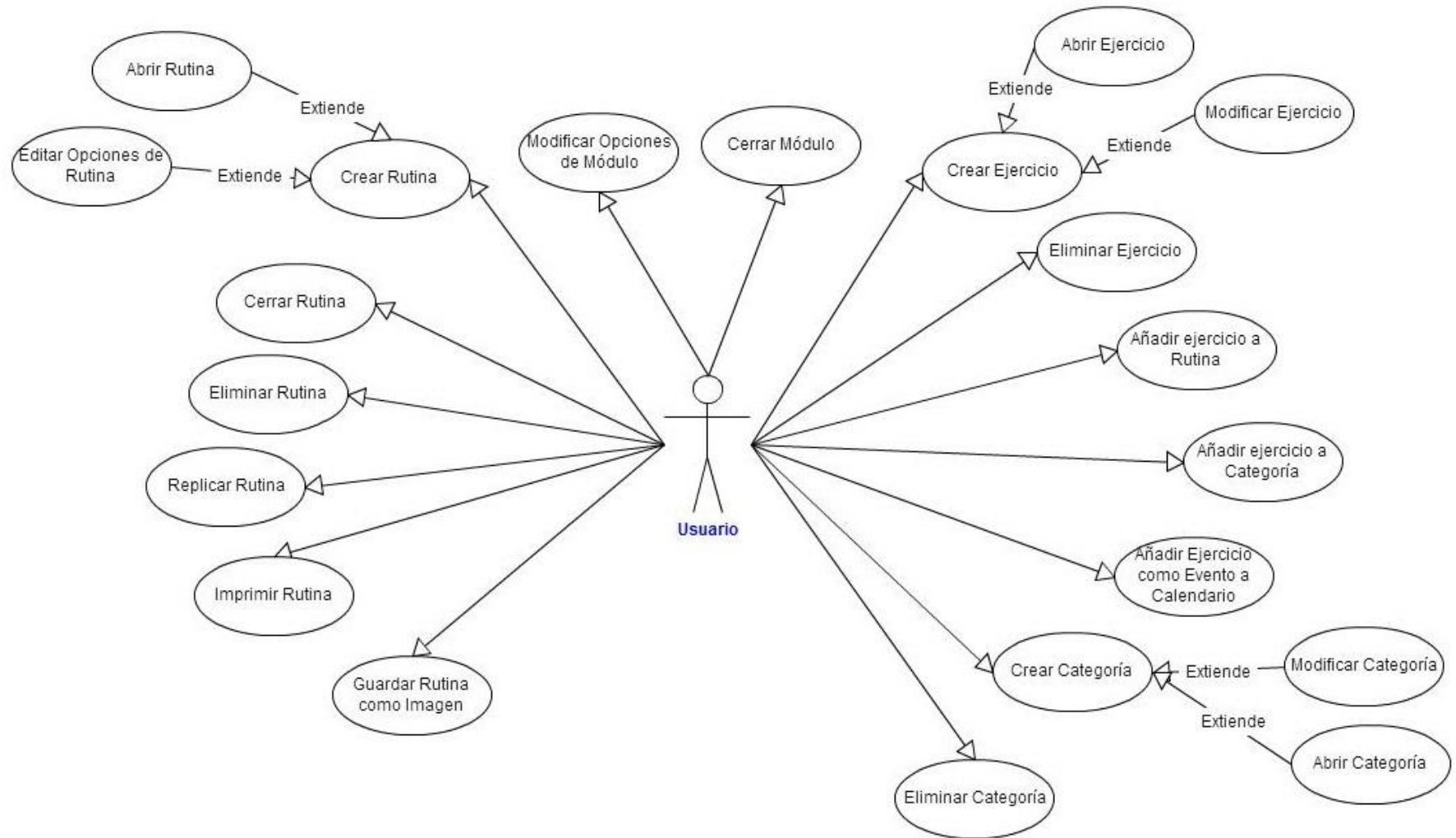
Diagrama Entidad Relación



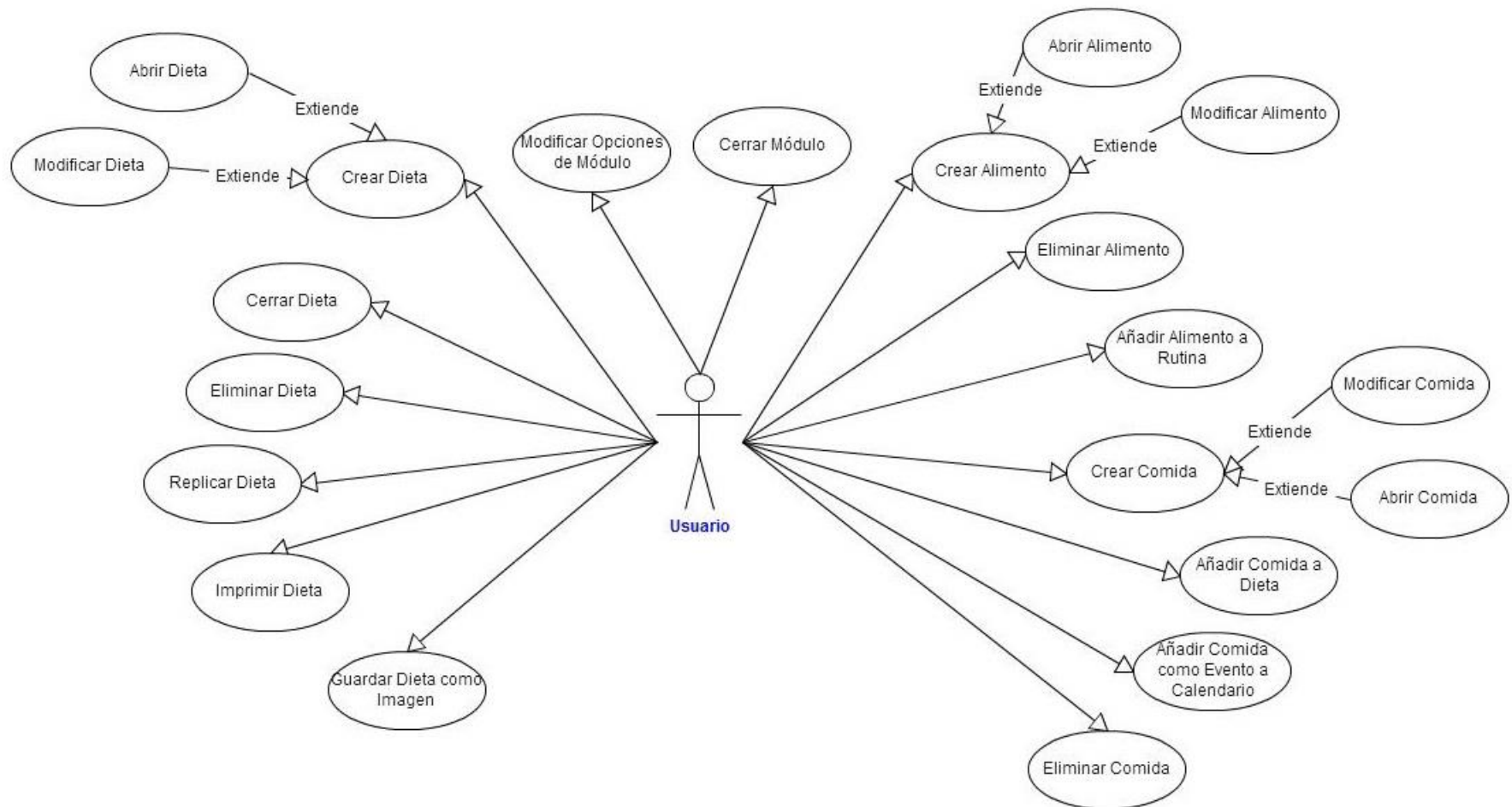
Vistas Casos de uso: Módulo Calendario



Vistas Casos de uso: Módulo Ejercicios



Vistas Casos de uso: Módulo Dietista



Base de Datos

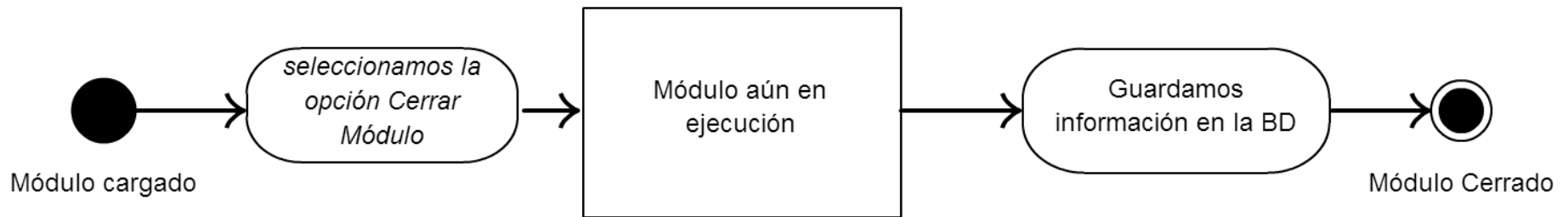
- Implementación:
 - Cada **fichero** representa:
 - Las Rutinas, Ejercicios, etc. de un usuario.
o bien
 - Un Calendario.
 - Cada **línea** de un fichero representa:
 - Una rutina, ejercicio... **concreta**.
o bien
 - Los **eventos** existentes en un calendario.

Base de Datos:

Interacción con la aplicación

- Al salir de un módulo:

Módulo Calendario/Ejercicios/Dietista



Leyenda:

estado

acción



estado inicial

estado final

[online diagramming & design] createely.com

Interfaz: Diseño

Aplicación

Módulo	Evento	Calen
Calendario 4	Añadir	
Calendario 8	Abrir	
Calendario 12	Modificar	
Calendario 15	Replicar	
	Eliminar	

Seleccionar Tipo de Evento

Ejercicio Normal Rutina Comida

Cancel

Evento 1 Details

Información Básica

Fecha: 1/1/2009

Nombre: Evento 1

Duración: 15 Horas

Opciones de Repetición

Frecuencia: Cada mes

Apariencia

Fuente: Times New Roman

Color Evento

Color Texto

OK Cancel

Seleccionar Calendario

Cancel OK

Interfaz:

Configuración del comportamiento

Información Rutina

Click derecho en nombre de la pestaña para cerrar

Nombre:

Descripción:

Duración: Semanas

Dia1	Dia2	Dia3
Flexiones Simples	Flexiones con Muñeca	Sentadilla con Barra
Abdominales Simples	Knee-To-Knee Stretch	Respiracion de fuego
	Bandal Chagui	

Consumo Calórico Total: Kcal.

```
private void TablaEjercicios_CellDoubleClick(object sender, DataGridViewCellEventArgs e)
{
    if (TablaEjercicios.Rows[e.RowIndex].Cells[e.ColumnIndex].Value != null)
    {
        Modulo_Ejercicios.Ejercicio_Tipo EjercicioAAbrir = this.parentForm.GetListaEjerciciosBase().Find(
            namespace Modulo_Ejercicios InformacionEjercicios EjercicioAbierto;
            all)
        {
            EjercicioAAbrir = this.parentForm.GetListaEjerciciosPersonalizada().Find(item => item.Nombre.E
            EjercicioAbierto = new ModuloEjerciciosPestañaInformacionEjercicios(EjercicioAAbrir, this.pare
        }
        else EjercicioAbierto = new ModuloEjerciciosPestañaInformacionEjercicios(EjercicioAAbrir, this.par
        EjercicioAbierto.FormClosed += ((s, args) => this.parentForm.CerrarPestaña());
        EjercicioAbierto.WindowState = FormWindowState.Maximized;
        EjercicioAbierto.Dock = DockStyle.Fill;
        TabPage PestañaAbrirEjercicio = new TabPage();
        PestañaAbrirEjercicio.Text = EjercicioAAbrir.Nombre;
        EjercicioAbierto.TopLevel = false;
        EjercicioAbierto.Parent = PestañaAbrirEjercicio;
        this.parentForm.añadirPestañaBarraHerramientas(PestañaAbrirEjercicio);
        this.parentForm.GetBarraPestañas().SelectedTab = PestañaAbrirEjercicio;
        EjercicioAbierto.ControlBox = false;
        EjercicioAbierto.Show();
    }
}

private void TablaEjercicios_CellMouseEnter(object sender, DataGridViewCellEventArgs e)
{
    if ((e.ColumnIndex >= 0) && (e.RowIndex >= 0))
    {
        if (TablaEjercicios[e.ColumnIndex, e.RowIndex].Value != null)
        {
            Modulo_Ejercicios.Ejercicio_Implementacion ejercicio = this.rutina.ListaEjercicios[e.Co
            string info = "Ejercicio:\t\t" + ejercicio.Nombre + "\nSeries:\t\t" + ejercicio.Series.
            TablaEjercicios.Rows[e.RowIndex].Cells[e.ColumnIndex].ToolTipText = info;
        }
    }
}
```