PRUEBAS UNITARIAS ORIENTADAS A SQL SERVER

Zeus Pruebas Unitarias®

JHON RAMIREZ GOMEZ

FELIX RODRIGUEZ BORJA

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR

FACULTAD DE INGENIERÍAS

PROGRAMA DE INGENIERÍA DE SISTEMAS

CARTAGENA DE INDIAS D.T. Y C.

2013

Nota de Aceptación:

_____
_____
_____
_____
_____
_____

_____
Firma del presidente del jurado

_____
Firma del jurado

_____
Firma del jurado

Cartagena de Indias D.T. y C., 27 de Mayo de 2013.

AGRADECIMIENTOS

A Dios, porque nos dio los medios necesarios avanzar en el cumplimiento de esta meta, porque coloco personas muy especiales en nuestro camino que aportaron a lo largo de proceso y nos dio la fortaleza necesaria para superar todas las adversidades que se nos presentaron.

Expresamos nuestro más sincero agradecimiento a Isaac Zúñiga, Moisés Quintana Álvarez y Gonzalo Garzón por el todo apoyo y los conocimientos brindados durante todo el transcurso de nuestra carrera. Extendemos nuestro agradecimiento a todo el grupo de profesores del programa de ingeniería de sistemas, ya que nos dieron las bases necesarias para enfrentar los retos profesionales que nos hemos trazado desde que optamos por esta carrera.

A la Universidad Tecnológica de Bolívar por brindar el ambiente necesario para nuestro desarrollo personal y profesional.

A nuestros padres, porque infundieron la ética y valores morales en nosotros y nos ayudan cada día a ser mejores personas. También por su gran apoyo y confianza durante este proceso de formación profesional.

Por último, pero no menos importante, expresamos un profundo agradecimiento a nuestros compañeros, y colegas:

- Moisés Castellar
- Leonardo Beleño
- Jesús Restrepo
- José Villeros
- Víctor Cortes
- Diana Benedetti
- Sandra Cortes

DEDICATORIA

A mis queridos padres Javier Ramírez y Rosa Gómez, por depositar toda su confianza en mí, por apoyarme siempre y en todo momento de mi carrera y por los valores inculcados que han hecho de mí una mejor persona. Por eso hoy digo que este logro es de todos y solo me resta decir disfrútenlo que sin ustedes nada de esto hubiese sido posible.

A mi hermano Javier Ramírez, por su apoyo y por sus reiteradas subidas de ánimo, pues fue este quien más me alentó para alcanzar este logro. Gracias por ayudarme a hacer esto posible

Jhon Ramírez Gómez

DEDICATORIA

Quiero dedicar este proyecto y mi carrera en general antes que a nadie a Dios ya que él fue quien puso en mí el deseo y las ilusiones de ser un profesional de bien, ético y luchador. Después de él esta dedicatoria va para:

Mispadres: Felix Rodríguez Vargas y Myriam Borja, por el esfuerzo realizado y apoyo entregado hacia mí y por ser ellos una de las motivaciones para superarme día tras día y tener siempre como meta ser su orgullo en todos los aspectos de mi vida.

Mi Hermana: Karolina Rodríguez Borja, por todo el apoyo, confianza y amor depositado en mí.

Mi novia: Madelein Ramirez Mendoza, porser mi apoyo en todo momento y fuente de ganas de salir adelante con mis proyectos que también son los suyos.

Felix Rodríguez Borja.

CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

GLOSARIO

Testing: son las investigaciones empíricas y técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada.

Tester: Actor principal de la aplicación, es quien se encarga de realizar las pruebas necesarias a los módulos de una aplicación determinada.

Prueba Unitaria: es una forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado.

Escenario: Agrupación de pruebas unitarias para la ejecución.

Plan de Ejecución: Es procesamiento de manera más general o universal de escenarios en los que están involucradas las pruebas unitarias.

Archivo .ini: Objeto en el cual se encuentran alojados datos necesarios para el buen funcionamiento de las aplicaciones. Es común en proyectos de Zeus Pruebas Unitarias®**.**

Logueo: Hace referencia a cuando un usuario inicia sesión en un determinado sistema o aplicación.

PU: Siglas de Pruebas Unitarias

PE: Siglas de Plan de Ejecución.

INTRODUCCIÓN

El testing[1]es una técnica cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada. Las pruebas unitarias[2] hacen parte de dicha técnica, y lo que hacen es una división de procesos (pruebas) de los diferentes módulos de un software determinado.

El inconveniente que se presenta en el proceso de testing de un software es el acaparar todos los módulos de este, nuestro sistema de pruebas unitarias facilitará y automatizará dicho proceso en lo que a software orientado a base de datos se refiere.

El esquema de pruebas unitarias propuesto se basa específicamente en aplicaciones con modelos de negocios en bases de datos (SQL SERVER), de los cuales se validará el impacto de las diversas modificaciones a los objetos de dichas bases de datos.

Por lo anterior, no está de más decir que las aplicaciones en las que se realicen las pruebas deben contar con bases de datos en SQL SERVER manejando a su vez objetos de éste como son funciones, procedimientos almacenados, vistas, triggers, etc.

[1] Testing: son las investigaciones empíricas y técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada.
[2] Prueba unitaria: forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado.

1. MODELO DE NEGOCIO

Zeus Pruebas Unitarias® surge como solución a la necesidad de realizar pruebas a lo aplicativos dentro del proceso de gestión de la calidad de los proyectos en Zeus Tecnología S.A. Es un sistema simple que le permite bien sea al desarrollador o a un tester[3] realizar las respectivas pruebas y configurar planes depruebas a los módulos que se encuentran en producción antes de enviarlo a ventas, garantizando así que los productos tienen más fuerza y robustez.

Zeus Pruebas Unitarias® es un software orientado a SQLel cualserá capaz automatizar, gestionar y mostrar el impacto de modificaciones en objetos de bases de datos en los sistemas con este modelo de negocio. En detalle se podrán configurar, ejecutar y consultar las pruebas unitarias realizadas a un software determinado. Por otro lado también se podrán configurar y ejecutar escenarios[4] que prácticamente son agrupaciones de pruebas unitarias y/o ejecución de varias pruebas unitarias a la vez. También existe una manera más generalizada de ejecutar las pruebas unitarias y los escenarios que es mediante los planes de ejecución, cuyo propósito incorpora escenarios ya configurados; el sistema contará con la posibilidad de configurar, ejecutar y consultar dichos planes de ejecución.

Los principales usuarios finales de la aplicación serán los testers e incluso desarrolladores, ya que para el manejo de la herramienta se necesita de conocimientos principales de SQL y obviamente un manejo respetable de la aplicación a testear.

[3] Tester: actor principal de la aplicación, es quien se encarga de realizar las pruebas necesarias a los módulos de una aplicación determinada.
[4] Escenario: Agrupación de pruebas unitarias para la ejecución.

2. ESPEFIFICACIÓN DE REQUERIMIENTOS

2.1. USUARIOS DEL SISTEMA

Tester[5] (Único usuario del sistema): Este tipo de usuario es el encargado de gestionar todo a lo que pruebas unitarias, escenarios y planes de ejecución se refiere: configurar, ejecutar y consultar.

2.2. REQUERIMIENTOS FUNCIONALES

2.2.1. Requerimientos de usuario

- El sistema debe permitir realizar consultas SQL a la base a probar.
- El sistema debe permitir configurar pruebas unitarias.
- El sistema debe permitir ejecutar pruebas unitarias.
- El sistema debe permitir consultar pruebas unitarias.
- El sistema debe permitir configurar escenarios.
- El sistema debe permitir ejecutar escenarios.
- El sistema debe permitir configurar planes de ejecución.
- El sistema debe permitir ejecutar planes de ejecución.
- El sistema debe permitir consultar planes de ejecución.
- El sistema debe permitir consultar las dependencias.

[5]Tester: actor principal de la aplicación, es quien se encarga de realizar las pruebas necesarias a los módulos de una aplicación determinada.

2.2.2. Requerimientos del sistema

1) Ingresar al sistema

2) Gestionar consultas SQL
    a) Crear consulta
    b) Modificar consulta
    c) Eliminar consulta

3) AdministrarPruebas Unitarias
    a) configurarprueba unitaria
    b) Modificar prueba unitaria
    c) Eliminar prueba unitaria
    d) Ejecutar prueba unitaria
    e) Consultar prueba unitaria

4) AdministrarEscenarios
    a) Configurar Escenario
    b) Modificar Escenario
    c) Eliminar Escenario
    d) Ejecutar Escenario

5) Administrar Planes de Ejecución
    a) Configurar plan
    b) Modificar plan
    c) Eliminar plan
    d) Ejecutar plan

6) Consultar dependencias

2.2.3. Descripción requerimientos del sistema.

Gestionar Consulta SQL

El sistema permitirá la creación, modificación, eliminación y ejecuciónde consultas SQL que posteriormente servirán como patrones de búsqueda dentro de laspruebas.

Entradas

- Información consultas SQL:
    1. Código
    2. Descripción
    3. Base de Datos
    4. Query de consulta SQL

- Ejecución de consultas SQL.

Salidas

- Consulta SQL Creada
- Consulta SQL Modificada
- Consulta SQL Eliminada
- Consulta SQL Ejecutada

Administrar Escenarios

El sistema permitirá la configuración, modificación, eliminación y ejecución de escenarios (conjunto de una o más pruebas unitarias).

Entradas:

- Información de Escenarios:
    1. Identificador
    2. Nombre
    3. Módulo (Base de Datos)
    4. Descripción
    5. Pruebas asignadas

- Ejecución de escenarios

Salidas

- Escenario Configurado
- Escenario Modificado
- Escenario Eliminado
- Escenario Ejecutado

Administrar Pruebas Unitarias

El sistema permitirá la configuración, modificación, eliminación, ejecución y consulta de Pruebas Unitarias.

Entradas

- Información de Pruebas unitarias:
    1. Iden
    2. Nombre
    3. Modulo (Base de Datos)
    4. Procedimiento Almacenado
    5. Entrada (traza a evaluar o query)
    6. Escenario

7. Tablas afectadas
8. Campos afectados (de búsqueda y verificación)
9. Valores esperados en cada campo seleccionado.
10. Condición adicional

- Ejecución de la Prueba Unitaria
- Consulta de la Prueba Unitaria

Salidas

- Prueba Unitaria Configurada
- Prueba Unitaria Modificada
- Prueba Unitaria Eliminada
- Prueba Unitaria Ejecutada

Administrar Planes de Ejecución

El sistema permitirá la configuración, modificación, eliminación, ejecución y consulta de Planes de Ejecución.

Entradas

- Información del Plan de Ejecución.
  1. Iden
  2. Nombre
  3. Descripción
  4. Escenarios

- Ejecución del Plan de Ejecución
- Consulta del Plan de Ejecución

Salidas

- Plan de Ejecución Configurado
- Plan de Ejecución Modificado
- Plan de Ejecución Eliminado
- Plan de Ejecución Ejecutado

Consulta de Dependencias

El sistema permitirá mostrar una estructura que refleja cómo se ven afectadas directamente las tablas por cada prueba de un procedimiento almacenado. Jerarquía de mayor a menor es: Escenario, Procedimiento, Prueba, Tabla, Campo.

Entradas

- Datos del procedimiento almacenado
    1. Nombre

- Ejecución de consulta de dependencias

Salidas

- Resultado de la consulta de dependencias.

2.3. CASOS DE USO

2.3.1. Diagrama casos de uso

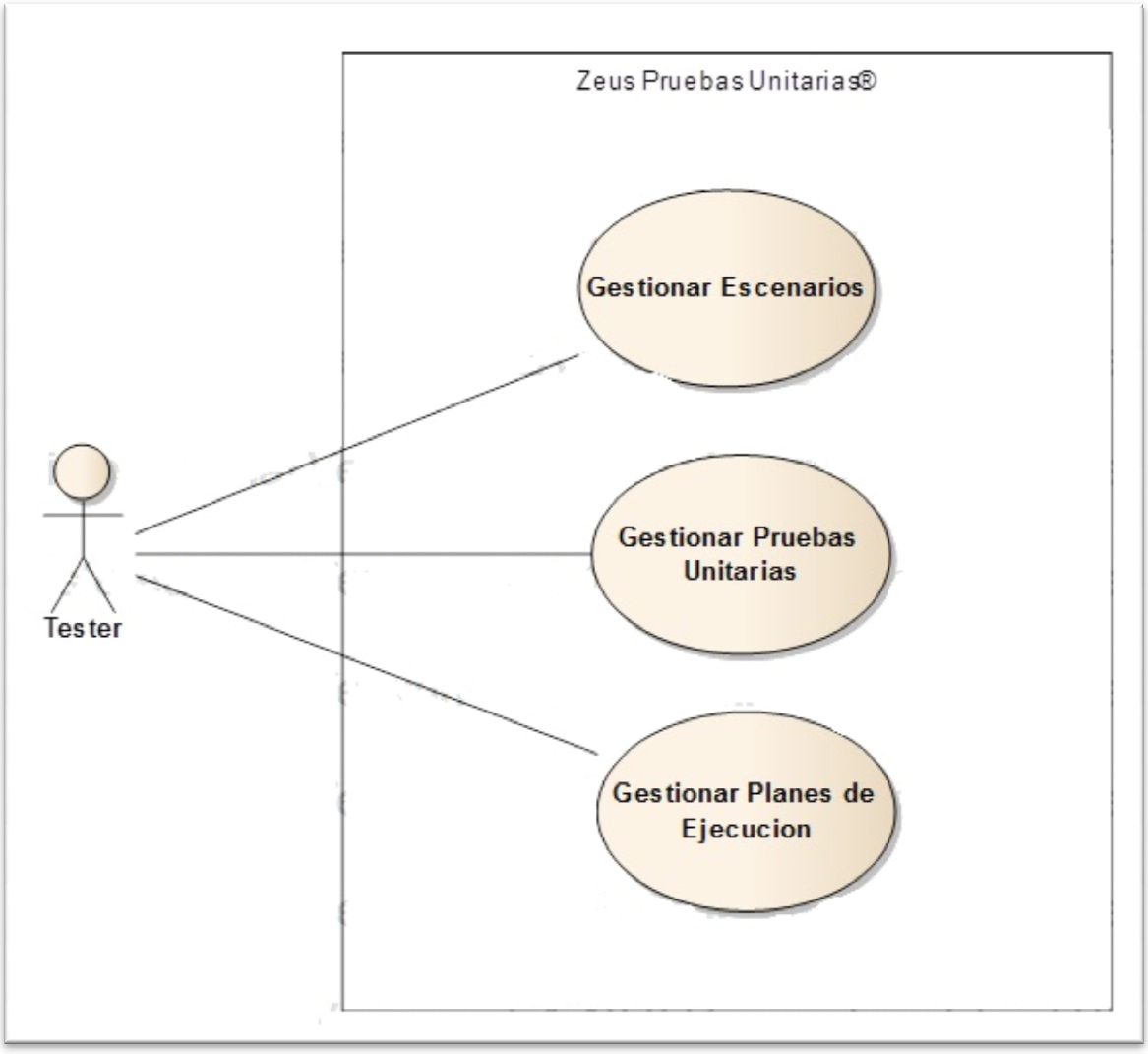


Figura Casos de Uso

2.3.2. Descripción casos de uso y Diagramas de Actividad

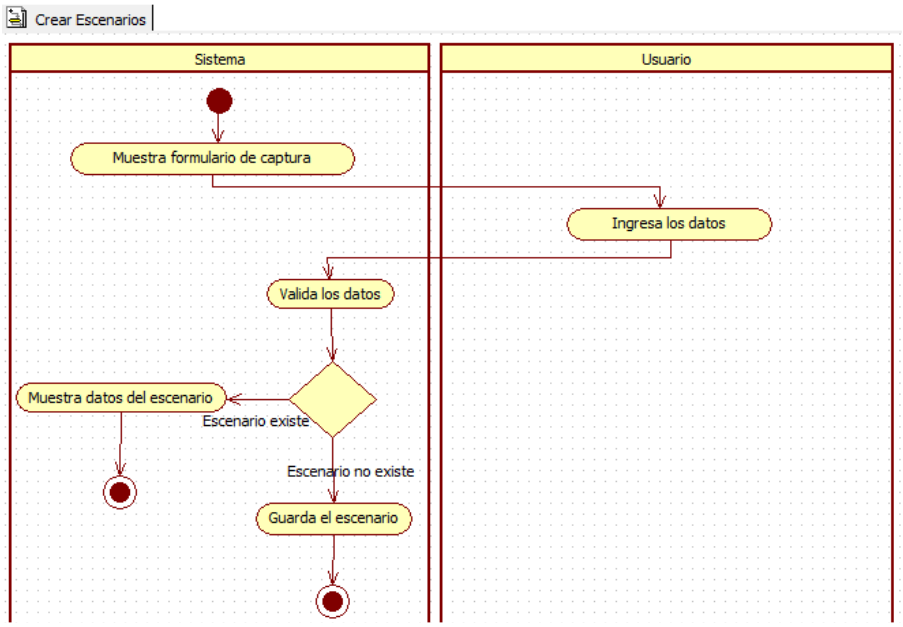
Tabla 1. C01

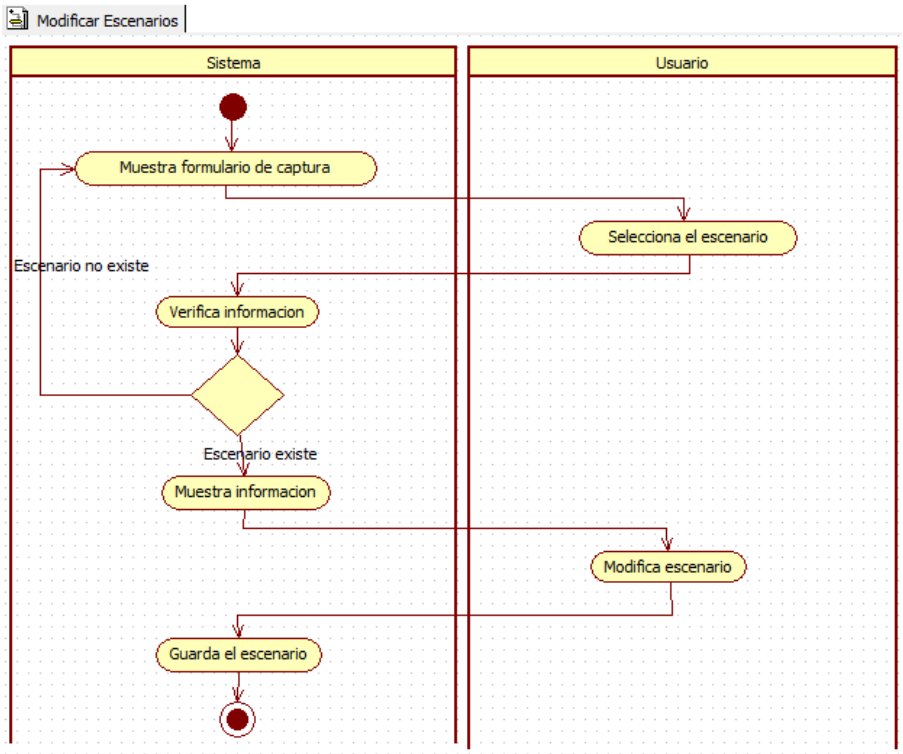
<table>
<tr><td>Identificación</td><td colspan="2">C01</td><td>Prioridad</td><td>Alta</td></tr>
<tr><td>Nombre</td><td colspan="4">GestionarEscenarios</td></tr>
<tr><td>Descripción</td><td colspan="4">El sistema permitirá la configuración, modificación, eliminación y ejecución de escenarios (conjunto de una o más pruebas unitarias).</td></tr>
<tr><td>Actores Principales</td><td colspan="4">Tester</td></tr>
<tr><td>Precondición</td><td colspan="4">o Ingreso al sistema
o Tester se encuentre logueado[6].</td></tr>
<tr><td>Poscondición</td><td colspan="4">• Gestión de un escenario en el sistema.</td></tr>
<tr><td rowspan="14">Flujo Normal</td><td>Paso</td><td colspan="3">Acción</td></tr>
<tr><td colspan="4">Creación</td></tr>
<tr><td>1</td><td colspan="3">Mostrar el formulario de captura</td></tr>
<tr><td>2</td><td colspan="3">El tester ingresa los datos</td></tr>
<tr><td>3</td><td colspan="3">El sistema verifica y valida la información ingresada</td></tr>
<tr><td>4</td><td colspan="3">El sistema guarda los datos del escenario</td></tr>
<tr><td rowspan="2">Flujo Alternativo</td><td>3.1</td><td colspan="2">El escenario ya fue creado</td></tr>
<tr><td>3.2</td><td colspan="2">El sistema muestra los datos del escenario creado con anterioridad</td></tr>
<tr><td colspan="4">Modificación</td></tr>
<tr><td>1</td><td colspan="3">En el formulario se selecciona el escenarios ya creado</td></tr>
<tr><td>2</td><td colspan="3">Modificar la información del escenario</td></tr>
<tr><td>3</td><td colspan="3">El sistema verifica y valida la información ingresada</td></tr>
<tr><td>4</td><td colspan="3">El sistema guarda los datos del escenario</td></tr>
</table>

[6]Logueo:Hace referencia a cuando un usuario inicia sesión en un determinado sistema o aplicación.

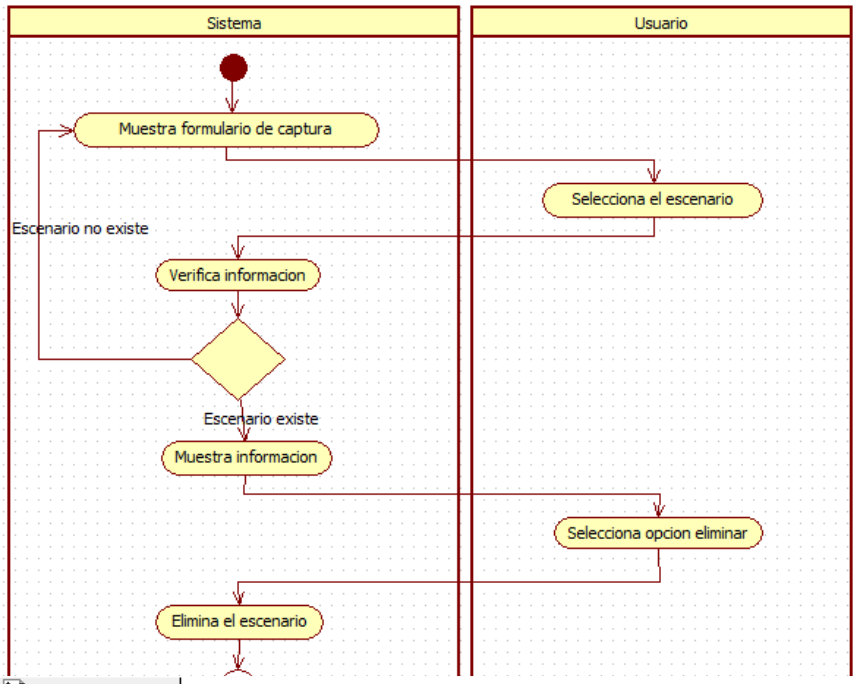
<table>
<tr><td rowspan="8"></td><td>Flujo Alternativo</td><td>3.1</td><td colspan="2">Si hay una inconsistencia en alguno de los datos, el sistema informará al usuario.</td></tr>
<tr><td colspan="4">Eliminación</td></tr>
<tr><td colspan="2">1</td><td colspan="2">En el formulario se selecciona el escenarios ya creado</td></tr>
<tr><td colspan="2">2</td><td colspan="2">El sistema permitirá eliminar el escenario</td></tr>
<tr><td colspan="4">Ejecución</td></tr>
<tr><td colspan="2">1</td><td colspan="2">Mostrar el formulario de ejecución</td></tr>
<tr><td colspan="2">2</td><td colspan="2">Se selecciona el escenario</td></tr>
<tr><td colspan="2">3</td><td colspan="2">Se ejecuta el escenario</td></tr>
<tr><td>Objetos involucrados</td><td>Frontera</td><td>Control</td><td colspan="2">Entidad</td></tr>
<tr><td></td><td>FrmConfigurarEscenario.cs
FrmEjecutarEscenario.cs</td><td>Crear
Modificar
Eliminar
Ejecutar</td><td colspan="2">• Escenario</td></tr>
</table>

Diagramas de actividad Caso de Uso 01





Eliminar Escenarios



Ejecutar Escenarios

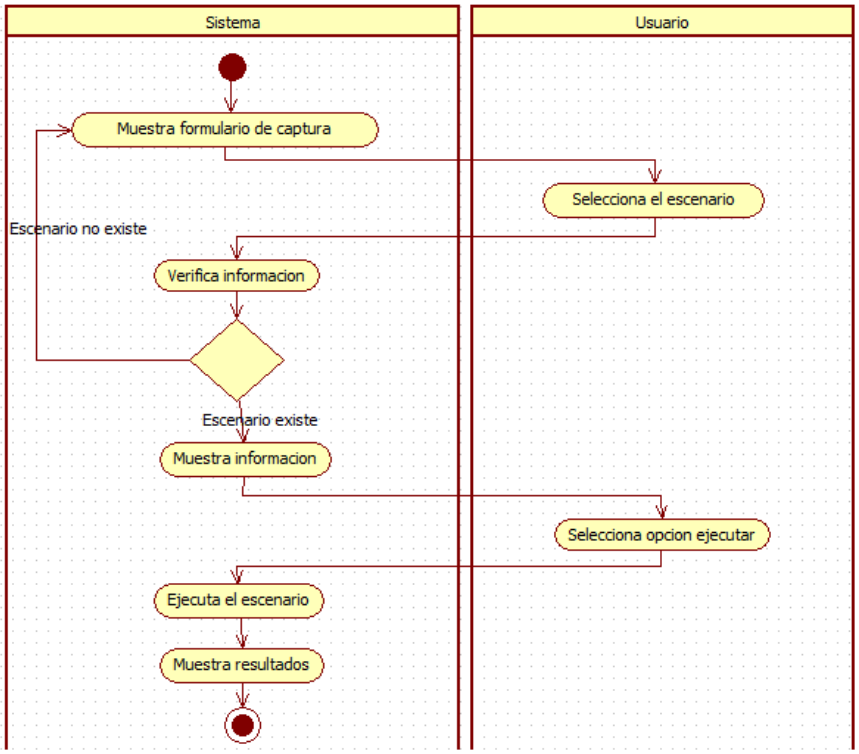
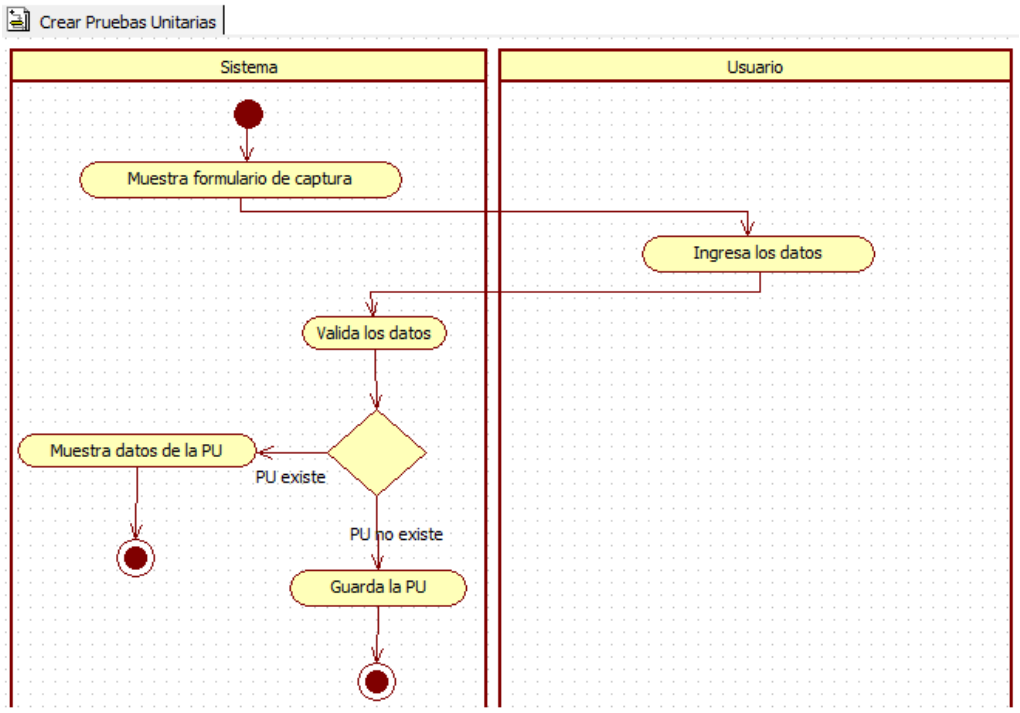


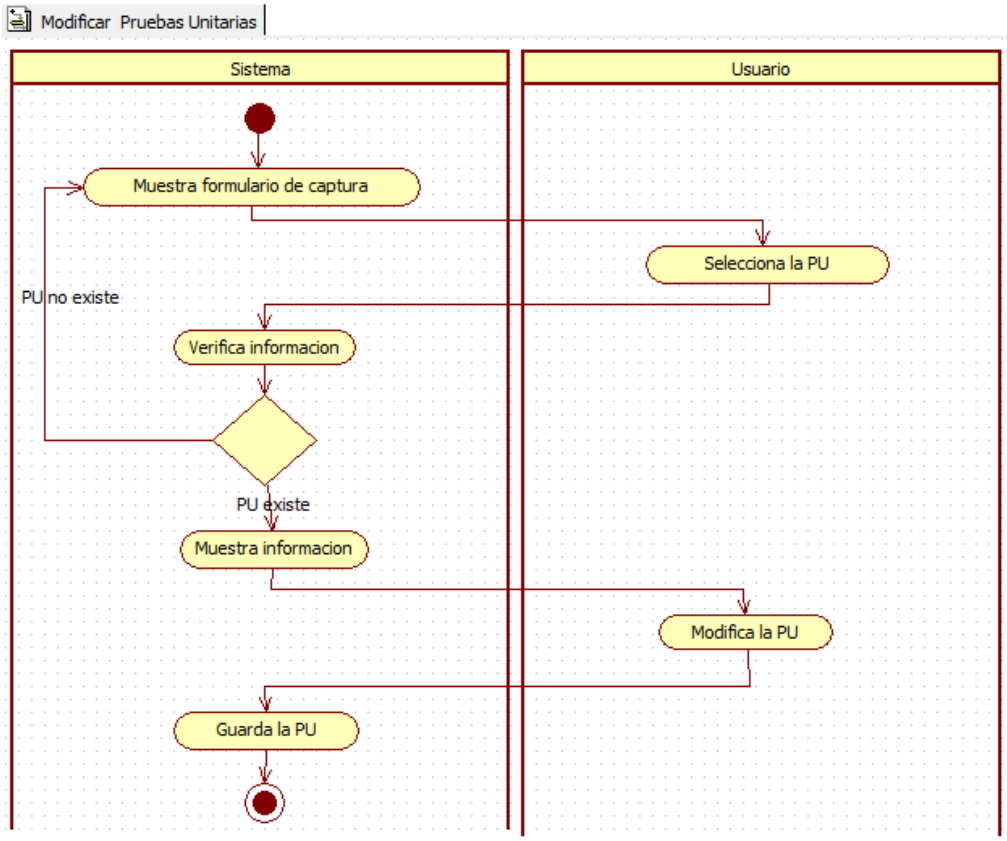
Tabla 2. C02

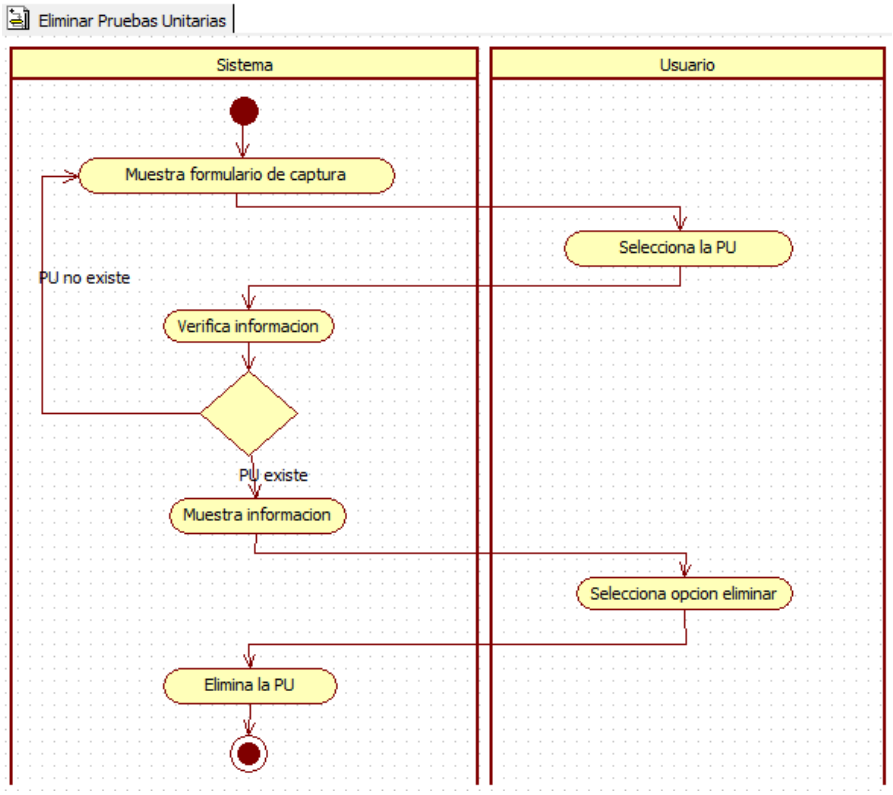
Identificación	C02	**Prioridad**	Alta
Nombre	GestionarPruebas Unitarias		
Descripción	El sistema permitirá la configuración, modificación, eliminación y ejecución de Pruebas Unitarias.		
Actores Principales	Tester		
Precondición	o Ingreso al sistema o Tester se encuentre logueado. o Exista el o los escenarios del cual hará parte la prueba unitaria		
Poscondición	• Gestión de las Pruebas Unitarias en el sistema.		
Flujo Normal	**Paso**	**Acción**	
	Creación		
	1	Mostrar el formulario de captura	
	2	El tester ingresa los datos	
	3	El sistema guarda los datos de las pruebas unitarias.	
	Flujo Alternativo	2.1	Si hay una inconsistencia en alguno de los datos, el sistema informará al usuario de la situación.
	Modificación		
	1	En el formulario se selecciona la prueba unitaria ya creada	
	2	Modificar la información de la prueba unitaria	
	3	El sistema verifica y valida la información ingresada	
	4	El sistema guarda los datos de las pruebas unitarias.	
	Flujo Alternativo	3.1	Si hay una inconsistencia en alguno de los datos, el sistema informará al usuario.

<table>
<tr><td rowspan="7"></td><td colspan="3">Eliminación</td></tr>
<tr><td>1</td><td colspan="2">En el formulario se selecciona prueba unitaria ya creada</td></tr>
<tr><td>2</td><td colspan="2">El sistema permitirá eliminar la prueba unitaria</td></tr>
<tr><td colspan="3">Ejecución</td></tr>
<tr><td>1</td><td colspan="2">Mostrar el formulario de ejecución</td></tr>
<tr><td>2</td><td colspan="2">Se selecciona la prueba unitaria</td></tr>
<tr><td>3</td><td colspan="2">Se ejecuta la prueba unitaria</td></tr>
<tr><td rowspan="2">Objetos involucrados</td><td>Frontera</td><td>Control</td><td>Entidad</td></tr>
<tr><td>FrmConfigurarPU0.cs
FrmConfigurarPUI.cs
FrmConfigurarPUII.cs
FrmConfigurarPUII.cs
FrmConfigurarPUIV.cs
FrmEjecutarPU.cs</td><td>Crear
Modificar
Eliminar
Ejecutar</td><td>• Prueba Unitaria</td></tr>
</table>

Diagramas de actividad Caso de Uso 02






Eliminar Pruebas Unitarias
Sistema
Usuario
Muestra formulario de captura
Selecciona la PU
PU no existe
Verifica informacion
PU existe
Muestra informacion
Selecciona opcion eliminar
Elimina la PU

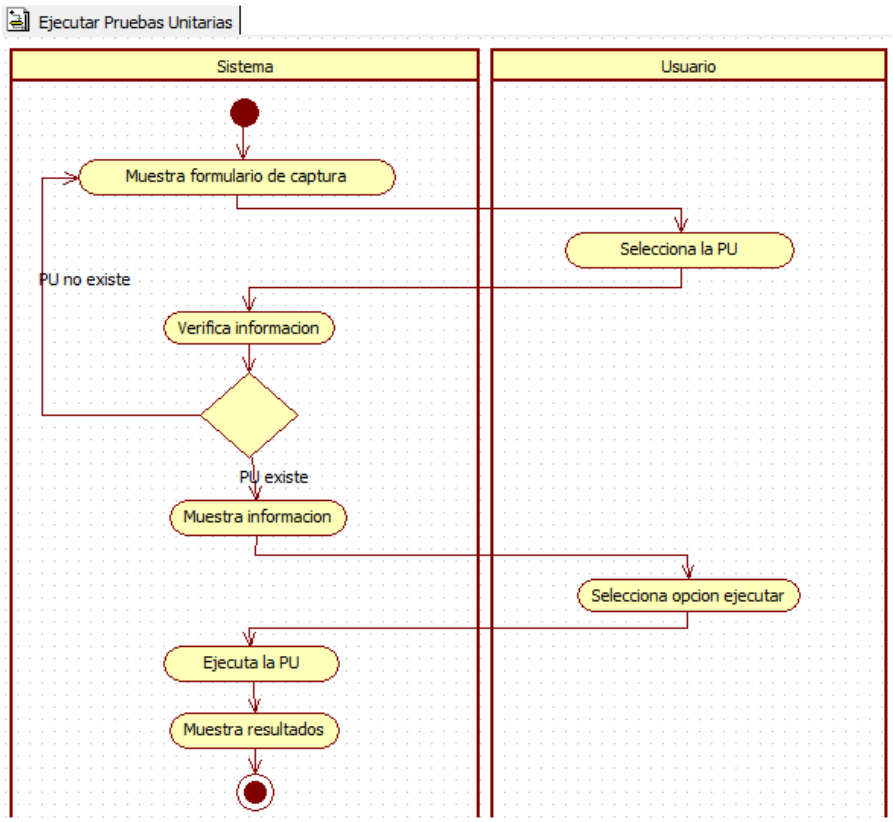
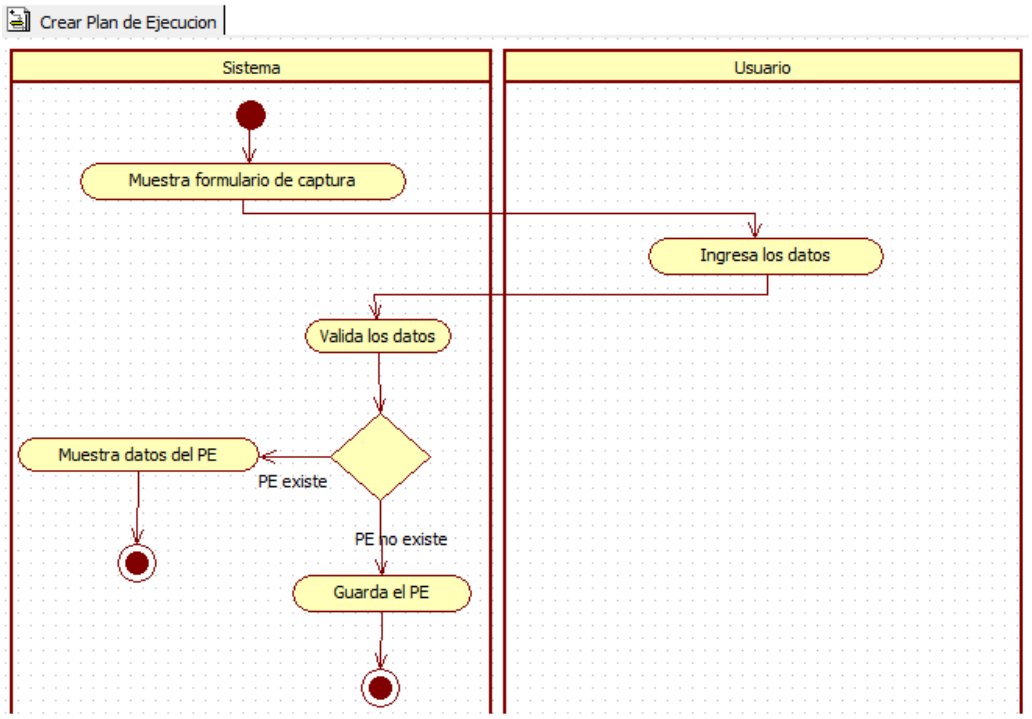

Ejecutar Pruebas Unitarias
Sistema
Usuario
Muestra formulario de captura
Selecciona la PU
PU no existe
Verifica informacion
PU existe
Muestra informacion
Selecciona opcion ejecutar
Ejecuta la PU
Muestra resultados

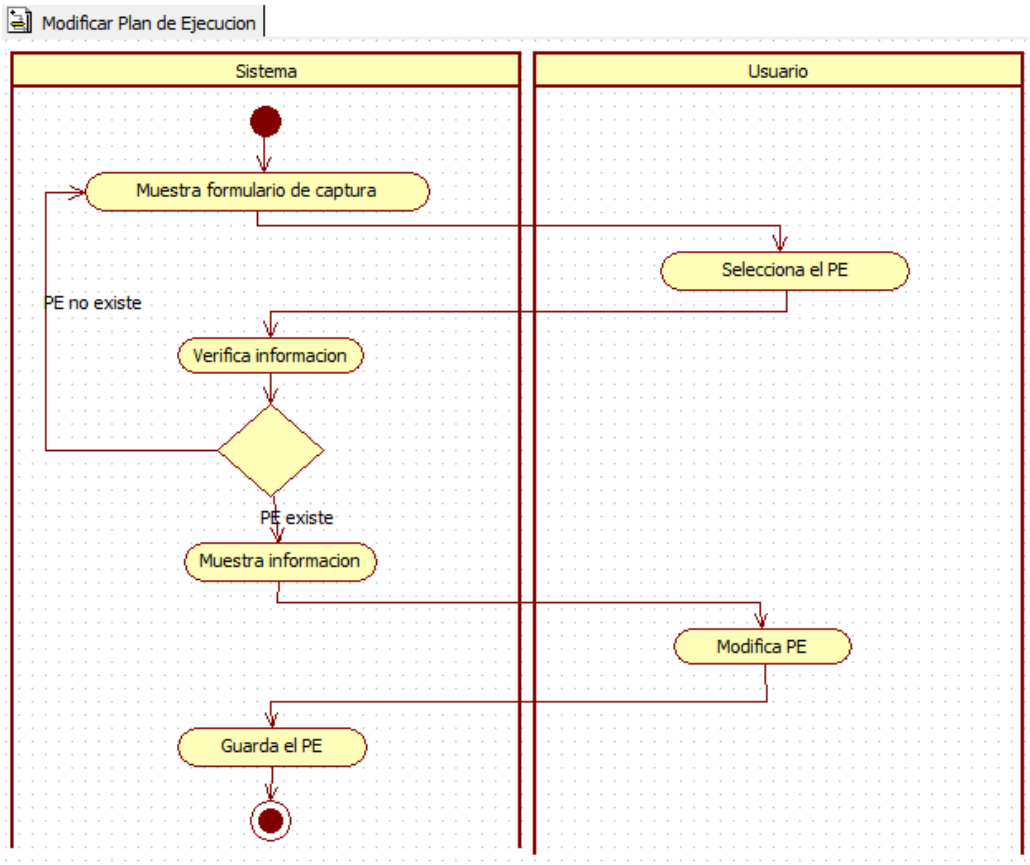
Tabla 3. C03

Identificación	C03		
Prioridad	Alta		
Nombre	Gestionar Planes de Ejecución		
Descripción	El sistema permitirá la configuración, modificación, eliminación y ejecución de Planes de Ejecución		
Actores Principales	Tester		
Precondición	o Ingreso al sistema o Tester se encuentre logeado. o Exista el o los escenarios con sus respectivas pruebas unitarias asignadas		
Poscondición	• Gestión de los Planes de Ejecución en el sistema.		
Flujo Normal	**Paso**	**Acción**	
	Creación		
	1	Mostrar el formulario de captura	
	2	El tester ingresa los datos	
	3	Seleccionar los escenarios.	
	4	El sistema guarda los datos del plan de ejecución.	
	Flujo Alternativo	3.1	Si hay una inconsistencia en alguno de los datos, el sistema informará al usuario de la situación.
	Modificación		
	1	En el formulario se selecciona el plan de ejecución ya creado	
	2	Modificar la información del plan de ejecución	
	3	El sistema verifica y valida la información ingresada	
	4	El sistema guarda los datos del plan de ejecución	
	Eliminación		
	1	En el formulario se selecciona el plan de ejecución ya creado	
	2	El sistema permitirá eliminar el plan de	

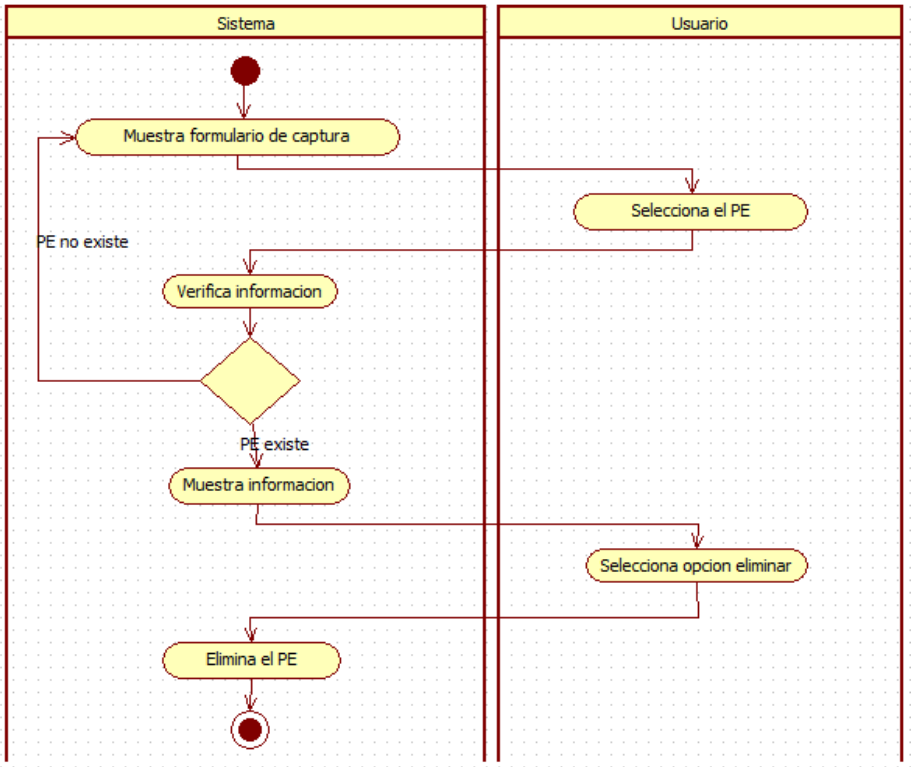
		ejecución	
	Ejecución		
	1	Mostrar el formulario de ejecución	
	2	Se selecciona el plan de ejecución	
	3	Se ejecuta el plan de ejecución	
Objetos involucrados	**Frontera**	**Control**	**Entidad**
	FrmConfigurarPE.cs FrmEjecutarPE.cs	Crear Modificar Eliminar Ejecutar	• Plan de ejecución

Diagramas de actividad Caso de Uso 03

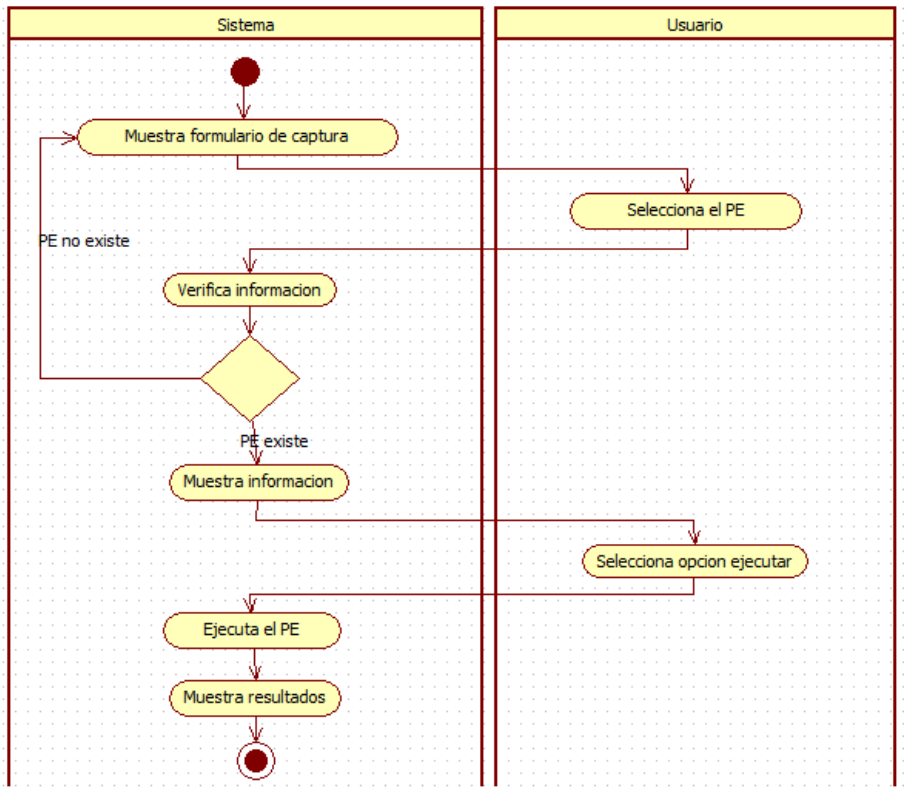




Eliminar Plan de Ejecucion



Ejecutar Plan de Ejecucion



2.4. REQUERIMIENTOS NO FUNCIONALES

- Disponibilidad del servidor
- Facilidad de uso
- Fácil accesibilidad
- Seguridad
- Escalabilidad
- Desempeño

3. MODELO DE ANÁLISIS

3.1. Diagrama de clases

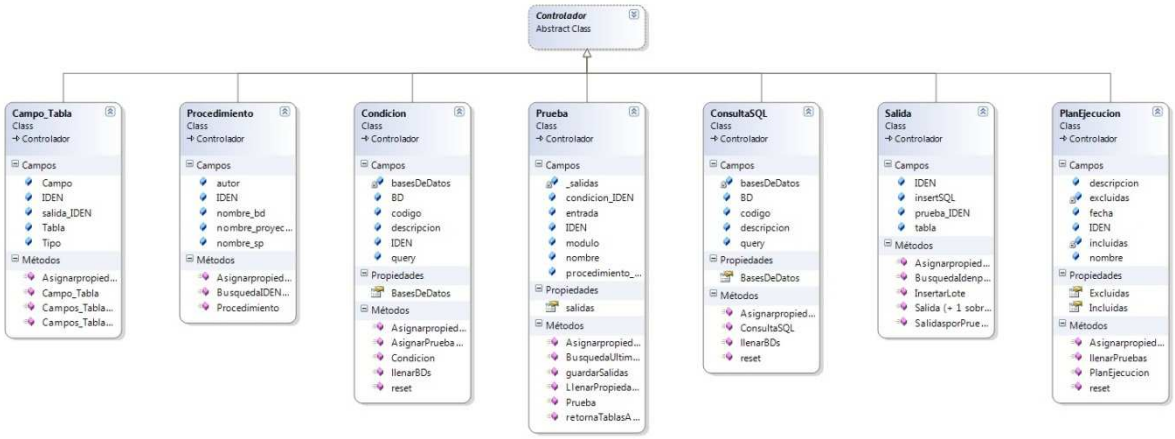


Figura Diagrama de Clases 1

Figura Diagrama de Clases 2

Figura Diagrama de Clases 3

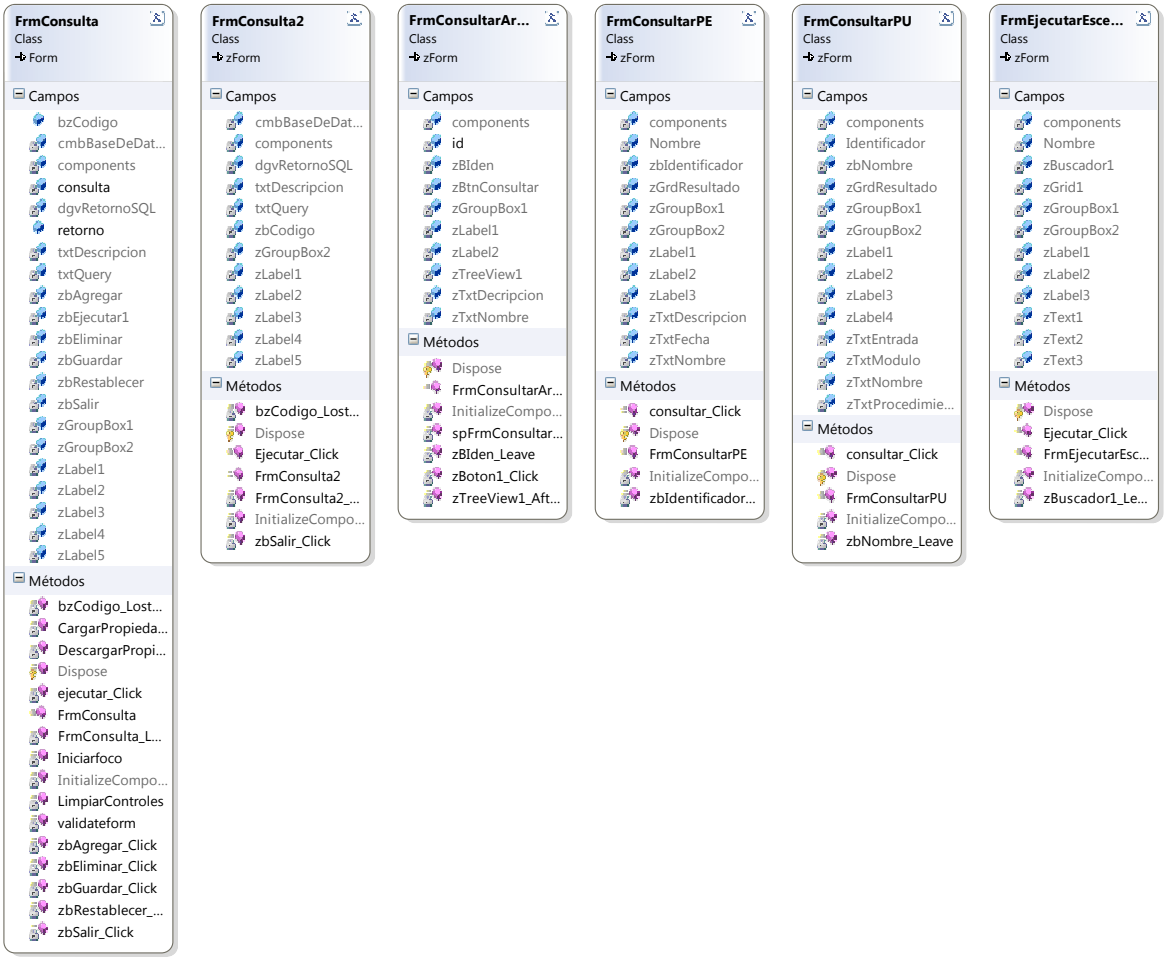


Figura Diagrama de Clases 4

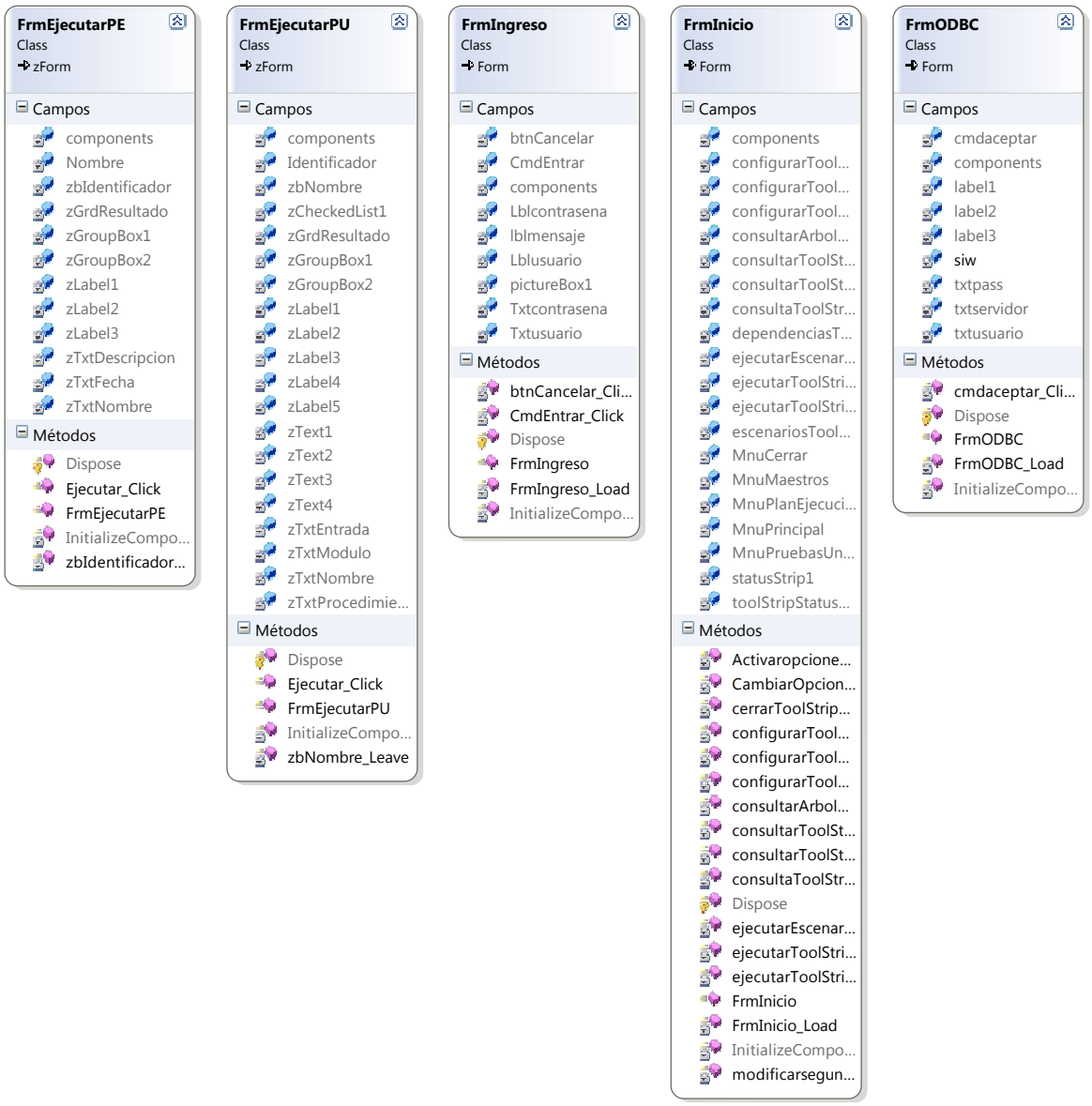


Figura Diagrama de Clases 5

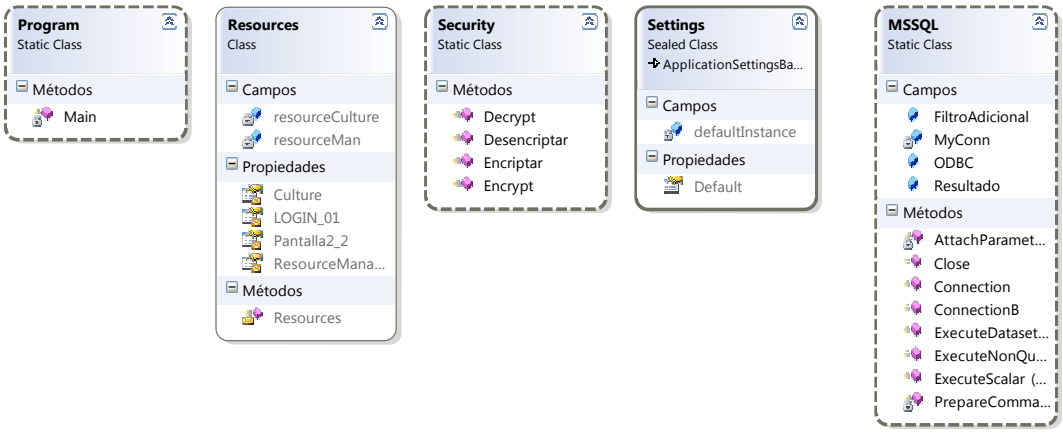


Figura Diagrama de Clases 6

4. MANUAL DEL SISTEMA

4.1. CONTENIDO DEL SISTEMA

El sistema sobre el que se soporta está desarrollado en el lenguaje de programación C# a través de la herramienta Visual Studio 2008. El Sistema contiene una serie formularios desarrollados en el mismo lenguaje de programación y se distribuyen como muestra la siguiente imagen:

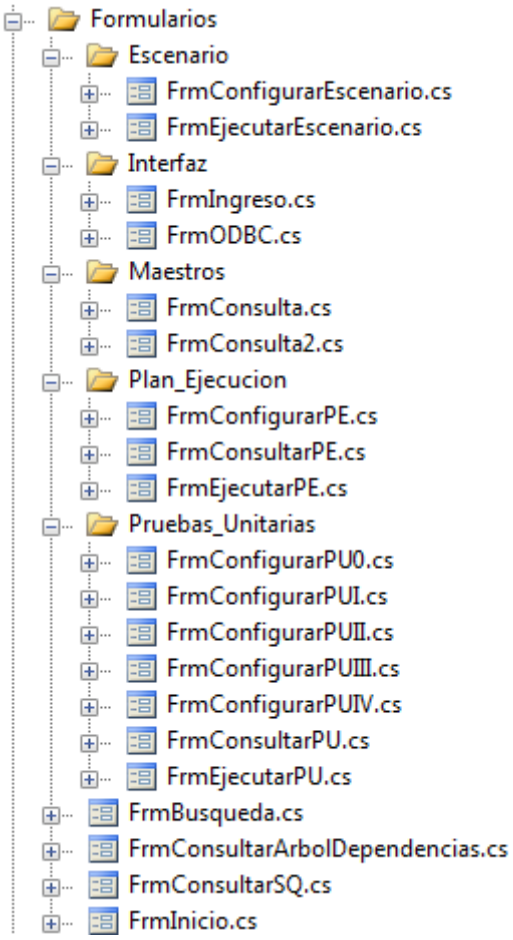


Figura Formularios del Proyecto

El sistema también contiene una serie de clases que le permiten al sistema agregar registros a la base de datos y la interacción con esta. Las clases son descritas en la siguiente imagen.

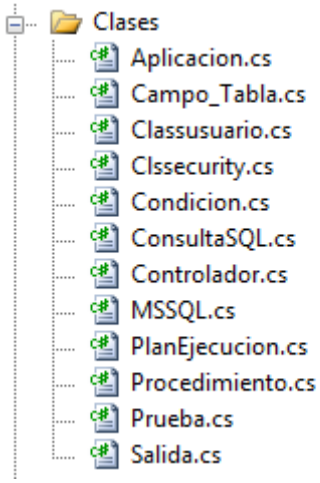


Figura Clases del Proyecto

El sistema contiene una base de datos SQL server creada a través de la herramienta SQL SERVER 2008 R2, Esta es usada para el almacenamientos de los datos que el sistema necesita para un funcionamiento óptimo, ya que la herramienta Zeus Pruebas Unitarias® necesita el ingreso de los datos de Escenarios, Pruebas Unitarias y Planes de Ejecución. Estos datos son necesarios para que el usuario pueda interactuar con la herramienta, hecho importante ya que Zeus Pruebas Unitarias® tiene la lógica en la base de datos. El sistema en el cual se alojaran los datos necesarios para la herramienta se describe a continuación.

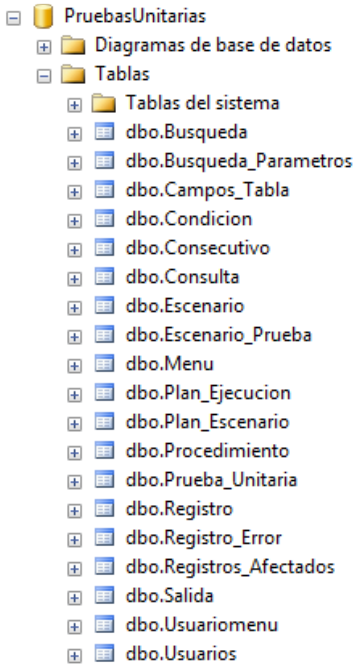


Figura Tablas SQL del Proyecto

Diagrama de la base de datos y descripción de objetos:

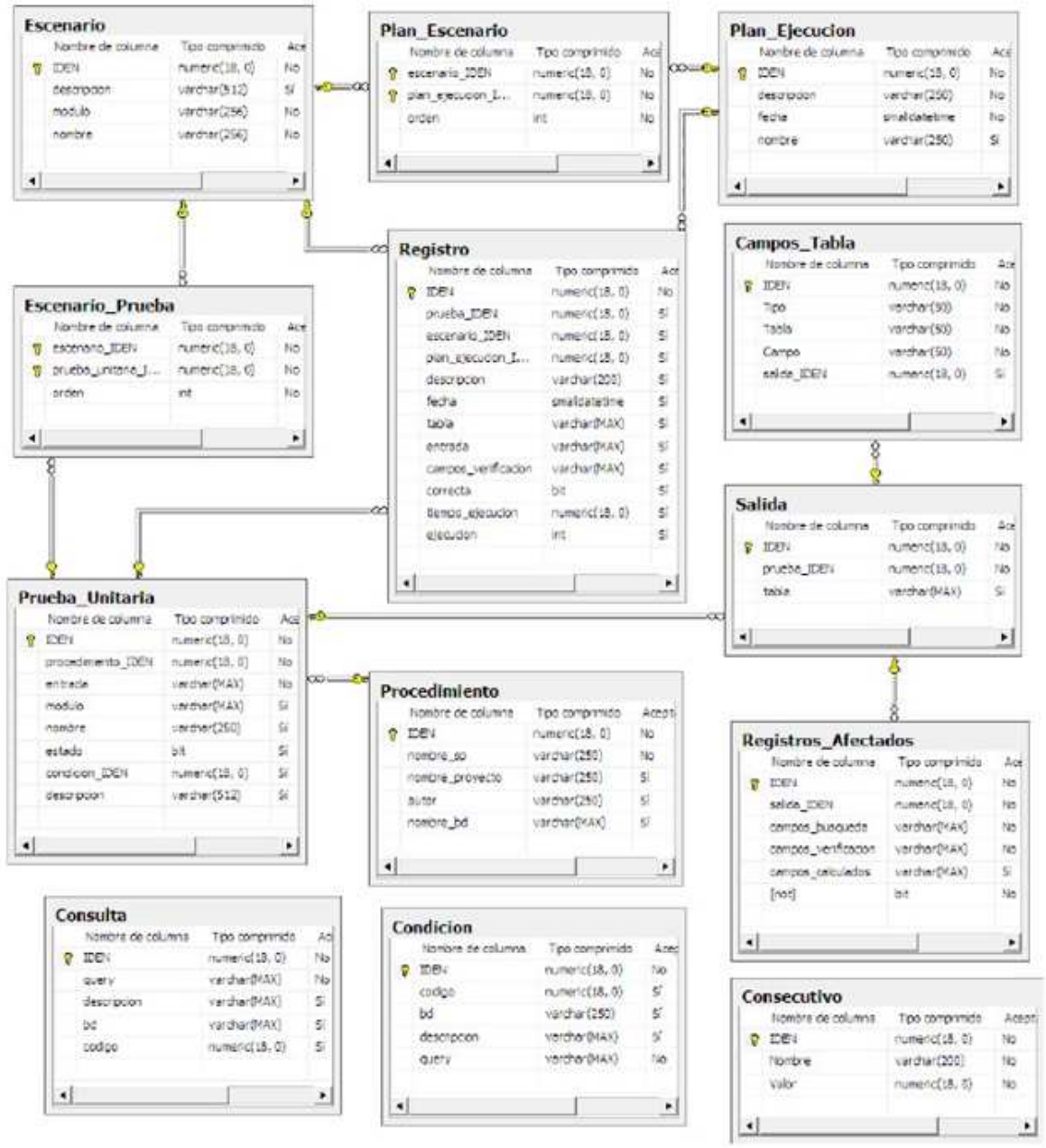


Figura Diagrama de Clases del Proyecto

Escenario: En esta tabla se almacenan diferentes situaciones en las cuales deberá ser probado un determinado software. Estas situaciones hacen alusión a las configuraciones de los parámetros un sistema, que llevarán a que este se comporte de una forma determinada. En ésta tabla encontramos los siguientes campos:

1. Nombre: Un nombre representativo para el escenario.

2. Descripción: La descripción de la situación del escenario, en donde se mencionen los valores de las principales variables que deberían cambiar el comportamiento del sistema.

3. Modulo: El nombre del aplicativo que se va probar.

Prueba_Unitaria: En esta tabla se configuran cada una de las pruebas que serán realizadas acada aspecto de un procedimiento almacenado.En ésta tabla encontramos los siguientes campos:

1. Nombre: Un nombre representativo para la prueba.

2. Descripción: Una descripción que le permita al desarrollador identificar que aspecto delprocedimiento almacenado serán probados.

3. Entrada: La instrucción SQL para ejecutar el procedimiento a probar. En este se debenespecificar los parámetros necesarios para probar un aspecto del SP. Ej.: “sp_ejemplo@parametro1=’valor1’, @parametro2=’valor2’”.

4. Procedimiento_Iden: El id del procedimiento almacenado a probar.

5. Condicion_Iden: El id de la condición necesaria para que se ejecute el SP.

6. Estado: Este campo indica si está activo.

Escenario_Prueba: Esta tabla relaciona los escenarios con las pruebas unitarias. El campoorden nos indica el “orden” en que serán ejecutadas las pruebas.

Plan_Ejecucion: En un plan de ejecución se configura un conjunto de pruebas unitarias paraque sean ejecutadas en bloque. Los planes de ejecución se utilizan para probar un aspecto delsoftware.

En ésta tabla encontramos los siguientes campos:
1. Nombre: Un nombre representativo para el plan de ejecución.
2. Descripción: Una descripción que le permita al desarrollador identificar que aspecto delsoftware serán probados.
3. Fecha: La fecha de creación del plan.

Plan_Escenario: Esta tabla relaciona los planes de ejecución con los escenarios. El campoorden nos indica el “orden” en que serán ejecutados los planes si se desea ejecutar todos losplanes de un escenario.

Procedimiento: Aquí se persisten todos los procedimientos almacenados para luego serprobados.
En ésta tabla encontramos los siguientes campos:
1. Nombre_sp: El nombre del procedimiento almacenado.
2. Nombre_bd: El nombre de la base de datos en donde está almacenado el SP.
3. Nombre_Proyecto: El nombre del software al cual pertenece el SP.
4. Autor: El autor del SP.

Condición: En esta tabla se almacenan expresiones SQL que serán evaluadas por el sistema, yque indicarán si la prueba unitaria relacionada a esta deberá ejecutarseEn ésta tabla encontramos los siguientes campos:
1. Código: Un número único que permita al desarrollador identificar la condición.
2. Descripción: Breve texto que le permite al desarrollador comprender qué verifica lacondición.
3. Bd: Nombre de la base de datos en donde se verificará la condición.
4. Query: Una expresión SQL que se ejecutará y permitirá detectar si la prueba unitariarelacionada a esta se debe ejecutar. Si la expresión SQL devuelve por lo menos una filala condición se considera exitosa por lo contrario se considera fallida.

Salida: Cada prueba unitaria configurada deberá tener una serie de resultados, representadosen registros esperados en diferentes tablas. Es decir, al configurar una prueba unitariase deberá establecer que cambios se espera que realice en la base de datos (inserción,modificación o eliminación de registros), para que de esta manera el sistema verifique si alejecutar la prueba unitaria se obtienen los mismos resultados. En la tabla salida se almacenancada tabla que afecte la prueba unitaria relacionada a esta.En ésta tabla encontramos los siguientes campos:

1. Tabla: El nombre de la tabla que será afectada.

2. Prueba_Iden: El id de la prueba unitaria relaciona a esta.

Registros_Afectados: En esta tabla se almacenan los registros de las diferentes tablas que sedeberán afectar al ejecutarse la prueba unitaria. Aquí se debe indicar cuales son los valores delos diferentes registros que deberán estar en la base de datos una vez que se ejecute la pruebaunitaria.

El sistema permite establecer valores fijos cuando se conoce cuál debe ser el valor finaldel campo (“campo=1”) , variaciones cuando se conoce, en cuanto debe variar el valor delcampo, se antepone el signo “+” al valor del incremento del campo (“campo=+1”) y “&” para undecremento (“campo=&1”), además se puede establecer que el valor del campo debe ser igualal valor arrojado de una consulta, es decir cuando el valor de este depende de otros camposen la base de datos (“campo=#id_consulta”), en donde id_consulta es el identificador de unregistro de la tabla consulta.En ésta tabla encontramos los siguientes campos:

1. Campos_Busqueda: Es una expresión SQL que el sistema utilizará para buscarel registro esperado. La expresión debe estar compuesta por el campo o camposde la tabla relacionada seguidos del valor esperado y unidos por el signo igual.

Ej.: “campo1=’valor1’”. Si se desea buscar por varios campos a la vez se deberán unirmediante operadores lógicos SQL. Ej.: “campo1=’valor1’ AND campo2=’valor2’”.

Nota: La expresión SQL contenida en este campo, el sistema lo ubica en el where de lasentencia que utiliza para verificar si existe el registro esperado.
2. Campos_Verificacion: Expresión SQL con la cual se verificará que la prueba unitariaarrojó los resultados esperados. Esta expresión tiene la misma estructura quecampos_busqueda.
3. [not]: Este campo se utiliza para especificar que la prueba unitaria deberá eliminareste registro. Si este está en true el sistema verificará que el registro no exista, y si loencuentra la prueba se considera fallida.
4. Salida_Iden: El id de la salida al que está relacionado.

Campos_Tabla: Esta es una tabla interna que utiliza el sistema en el proceso de configuraciónde pruebas unitarias.En ésta tabla encontramos los siguientes campos:
1. Tabla: El nombre de la tabla
2. Campo: Nombre del campo
3. Tipo: Si es de verificación o de búsqueda.
4. Salida_Iden: El id de la salida al que está relacionado.

Consulta: En esta tabla se almacenan las diferentes consultas que se utilizarán para verificarlos valores de los registros afectados.En ésta tabla encontramos los siguientes campos:
1. Código: Un número único que permita al desarrollador identificar la consulta.
2. Descripción: Breve texto que le permite al desarrollador comprender qué calor devuelvela consulta.
3. Bd: Nombre de la base de datos en donde se ejecutará la consulta.
4. Query: Una expresión SQL que se ejecutará y que devolverá una columna con un solovalor para compararlo con el valor esperado en registros_afectados.

4.2. PUESTA EN MARCHA

Para la puesta en marcha del sistema se creará un ambiente de prueba en una maquina con SQL SERVER 2008 R2 y .Net FrameWork 3.5 mínimo

4.3. FORMA DE ACCESO AL SISTEMA

Para acceder al sistema por parte de los usuarios se necesita acceso al servidor donde está alojada la base de datos.

5. MANUAL DE USUARIO

5.1 Crear Conexión

La primera vez que se corre el aplicativo luego de ser instalado, mostrara una ventana como lade la Figura 1.1 Crear Conexión donde se le debe especificar los siguientes campos:

1. La instancia de SQL Server a la cual se va a conectar

2. El nombre de usuario con el que se conecta al servidor de base de datos

3. La contraseña

Si la conexión es errada, el sistema le muestra un mensaje de error y le permite volver aintentar crear la conexión, si la conexión no se crea correctamente el sistema no funcionará. Sila conexión es correcta, el sistema le permite el paso al formulario de ingreso y habrá creadoun archivo encriptado .ini[7] que contiene datos de conexión para el sistema.

Figura 1.1 Crear Conexión

[7]Archivo .ini**:** Objeto en el cual se encuentran alojados datos necesarios para el buen funcionamiento de las aplicaciones. Es común en proyectos de Zeus Pruebas Unitarias®.

5.2 Formulario de Ingreso

Este formulario se despliega cada vez que el aplicativo inicia, a no ser que no encuentre elarchivo de conexión, en este caso se despliega primero el formulario de conexión. En la Figura1.2 Formulario de Inicio se deben colocar nombre de usuario y contraseña del sistema paraentrar y poder hacer uso de él. En caso de eso debe mostrar la ventana principal o de locontrario mostrará un mensaje de error (Figura 1.3 Error de Inicio de Sesión) y permanecerá enla ventana actual.

Figura 1.2 Formulario de Inicio

Figura 1.3 Error de Inicio de Sesión

5.3 Ventana Principal

En la Figura 1.4 Ventana Principal, se observa el entorno del programa en la parte superior se encuentra el Menú Principal desde el cual se puede acceder a cada funcionalidad del aplicativo, esta ventana solo se muestra luego de haberse logueado correctamente.

Figura 1.4 Ventana Principal

5.4 Barra Zeus

La Barra de Zeus es un componente que se ubica en la parte de arriba en los formulariosfuncionales de los aplicativos, para este programa en especial la barra puede tener un máximo de 6 botones, pues estos aparecen dependiendo a la funcionalidad.

1. Salir_ Salida segura de un formulario.

2. Guardar: Graba los datos de los controles.

3. Eliminar: Elimina el registro activo en los controles.

4. Nuevo: Limpia los controles.

5. Refrescar: Es una forma de actualizar los datos en pantalla.

6. Ejecutar: Su funcionalidad puede variar pero predefinidamente se utiliza para ejecutar comandos en la base de datos.

Figura 2.1 Barra de Zeus

5.5 Buscador Zeus

El buscador de Zeus está compuesto por una caja de texto y un botón (Figura 2.2) quedespliega la búsqueda (Figura 2.3)

Figura 2.2 Componente de Buscador

En el formulario de búsqueda se pueden configurar las búsquedas por diferentes tipos depatrones

1. Seleccione Opción: Es un desplegable por los campos de búsqueda de la tabla

2. Texto A buscar: Es el texto que se va a buscar en el campo seleccionado anteriormente

3. Anidamiento: Consta de un OR o AND para unir patrones de búsqueda

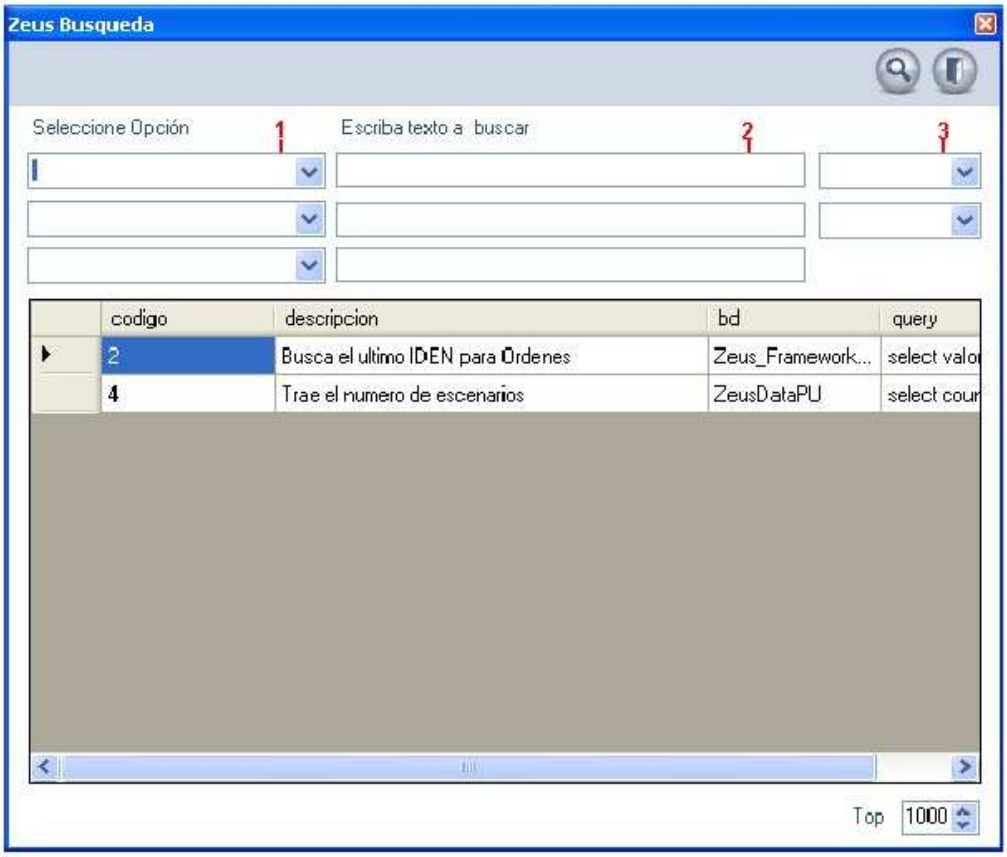


Figura 2.3 Formulario de Búsqueda

5.6 Consultas SQL

En esta opción se configuran consultas SQL (como se observa en la Figura 3.1 Configuración de Consulta Exitosa) que posteriormente servirán como patrones de búsqueda dentro de las pruebas y se debe diligenciar de la siguiente forma:

1. Código: se puede utilizar para buscar una consulta que ya haya sido registrada en el sistema, pero es un consecutivo autogenerado

2. Descripción: se debe dar un nombre a la consulta

3. Base de datos: se debe elegir la base de datos de la instancia en la cual se va a realizar la consulta,

4. SQL: caja de texto para escribir la consulta. Esta consulta debe retornar un solo valor,de lo contrario aparecerá un mensaje de error.

5. Retorno SQL: Es una grilla que retornara el resultado de la ejecución de la consulta

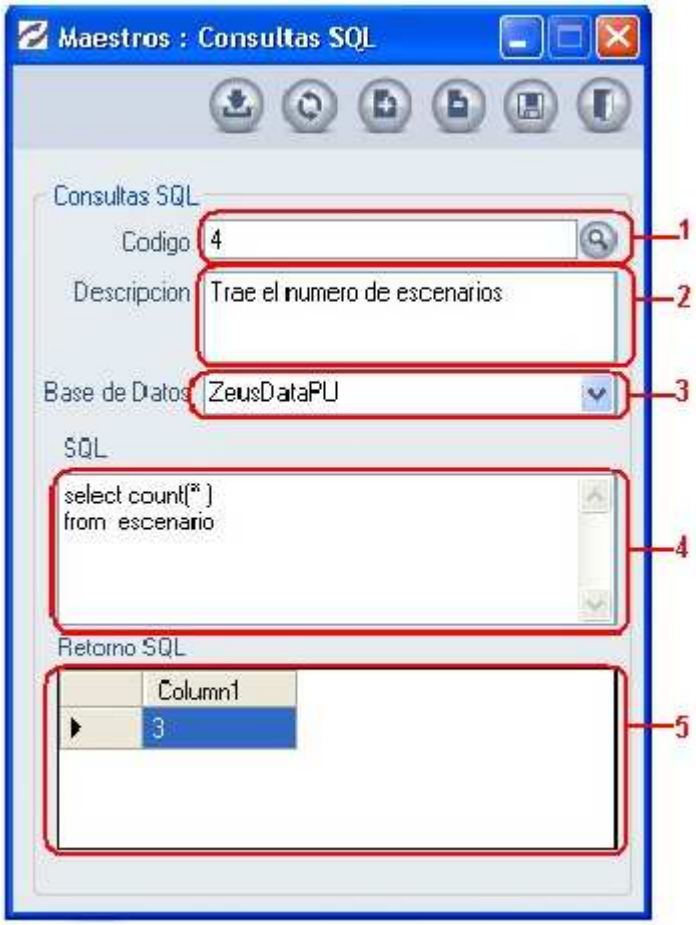


Figura 3.1 Configuración de Consulta Exitosa

5.7 Configurar Escenario

Para configurar correctamente un formulario, como se muestra en la figura 4.1, es necesario ingresar los siguientes datos:

1. Identificador: Es un campo autogenerado, sirve también para búsquedas.

2. Nombre: Nombre del escenario.

3. Modulo: Es la base de datos dentro de la cual se ejecutaran las pruebas

4. Descripción: Especificar las condiciones el escenario de pruebas

5. Pruebas: Contiene una lista de origen donde se encuentran las pruebas que ya han sido configuradas y una lista de destino donde se deben colocar las pruebas asignadas al escenario.

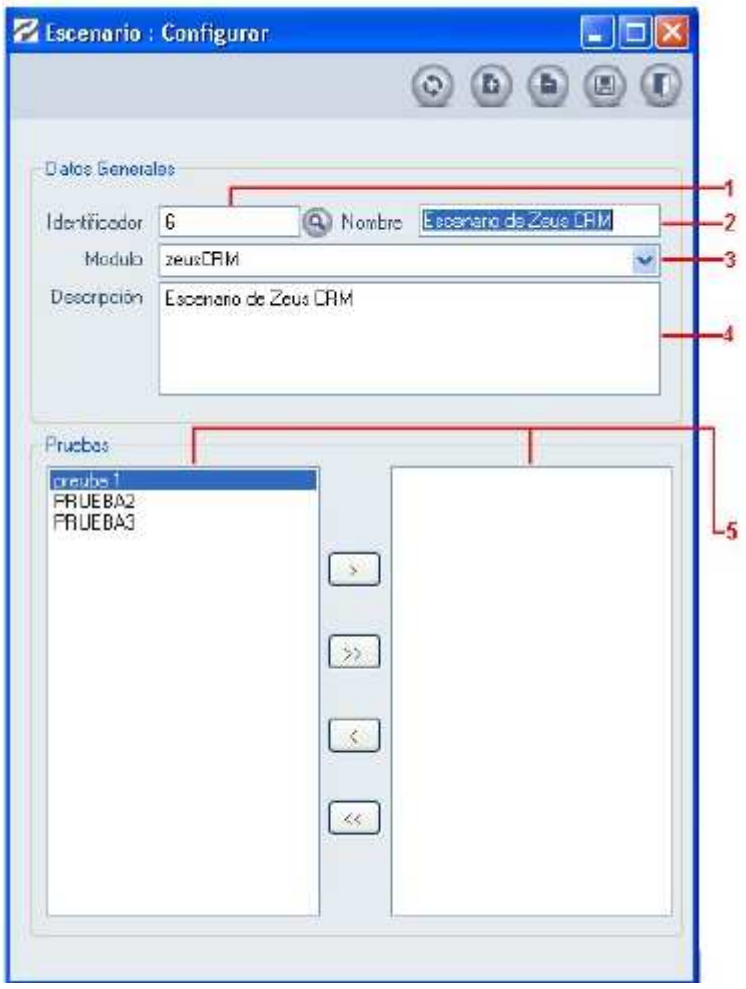


Figura 4.1 Configurar Escenario.

Cabe señalar que no es necesario asignar pruebas al escenario al momento de crearlo, pues estas pueden ser añadidas a la hora de configurar la prueba (Capitulo 5).

5.8 Ejecutar Escenario

Esta funcionalidad permite ejecutar todas las pruebas pertenecientes a un escenario mostrandoal final una grilla con los resultados como se observa en la Figura 4.2.Para ejecutar un escenario se necesita realizar los siguientes pasos:

1. Buscar el escenario haciendo uso del Buscador de Zeus, lo cual hará que se llenen loscampos de nombre, modulo y descripción.

2. Ejecutar el escenario presionando el botón Ejecutar de la barra de Zeus

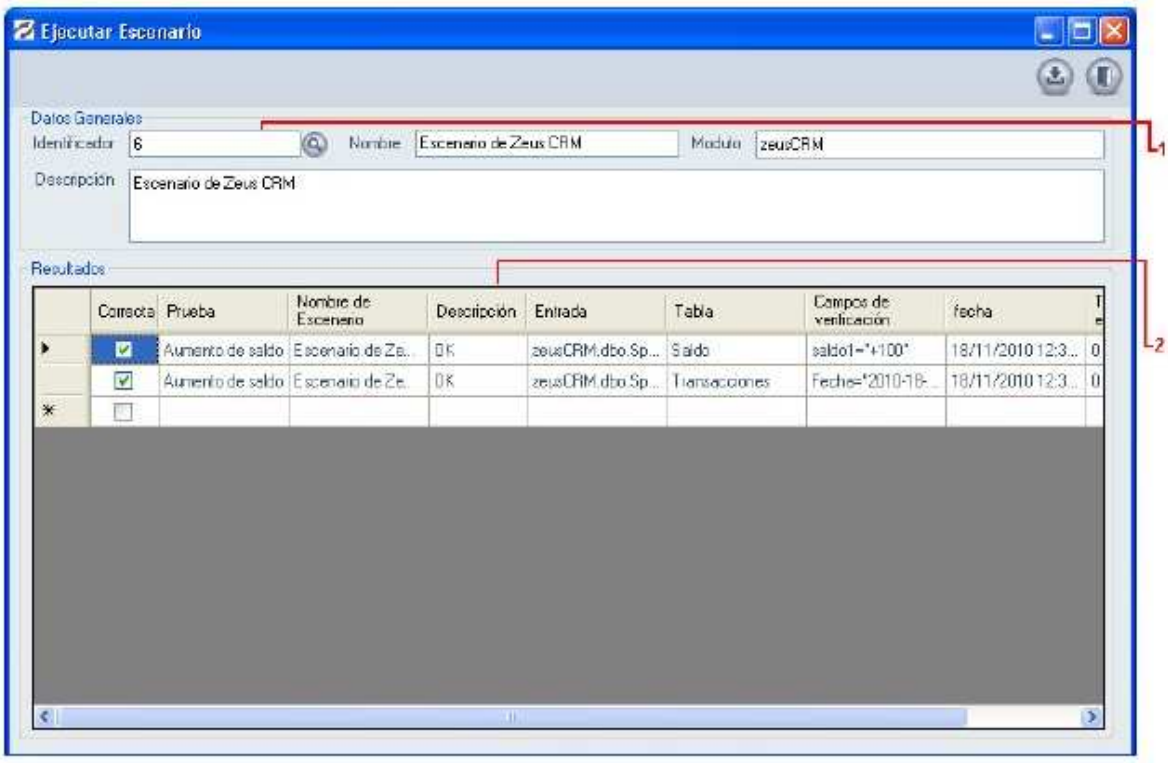


Figura 4.2 Ejecutar escenario

5.9 Configurar Prueba

Para configurar correctamente una prueba se debe seguir una serie de pasos descritos acontinuación.

Paso 1: Configuración General

Para la configuración general de la prueba es necesario seguir las siguientes indicaciones

1. Iden: El campo Iden es autogenerado, pero también sirve para la búsqueda.

2. Nombre: Nombre de la prueba.

3. Modulo: Se elige entre las bases de datos de la instancia.

4. Procedimiento: Luego de haber seleccionado el modulo, se despliegan losprocedimientos de este, elegir uno.

5. Entrada: En este campo debe insertar la traza que será evaluada.

6. Descripción: Coloque aquí observaciones o una descripción de la prueba.

7. Escenarios: Elija el o los escenarios dentro de los cuales se puede efectuar la prueba

8. Click en el botón de guardado de la Barra de Zeus

9. Click en el botón siguiente.

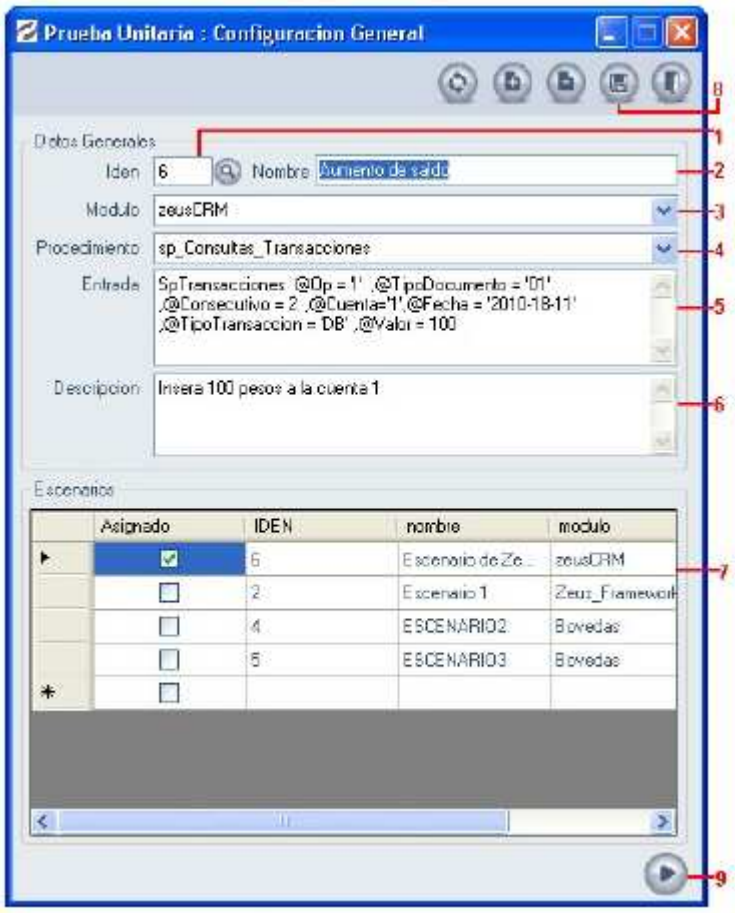


Figura 5.1 Configuración General

Paso 2: Tablas Afectadas

En este paso lo que se debe hacer es seleccionar las tablas que serán afectadas con la corridadel procedimiento, seleccionándolas de la tabla de origen a la tabla de destino, para hacer estoes necesario seguir los siguientes pasos:

1. Seleccionar la(s) tabla(s) de la lista de la izquierda

2. Presionar el botón de asignación simple

3. Click en siguiente

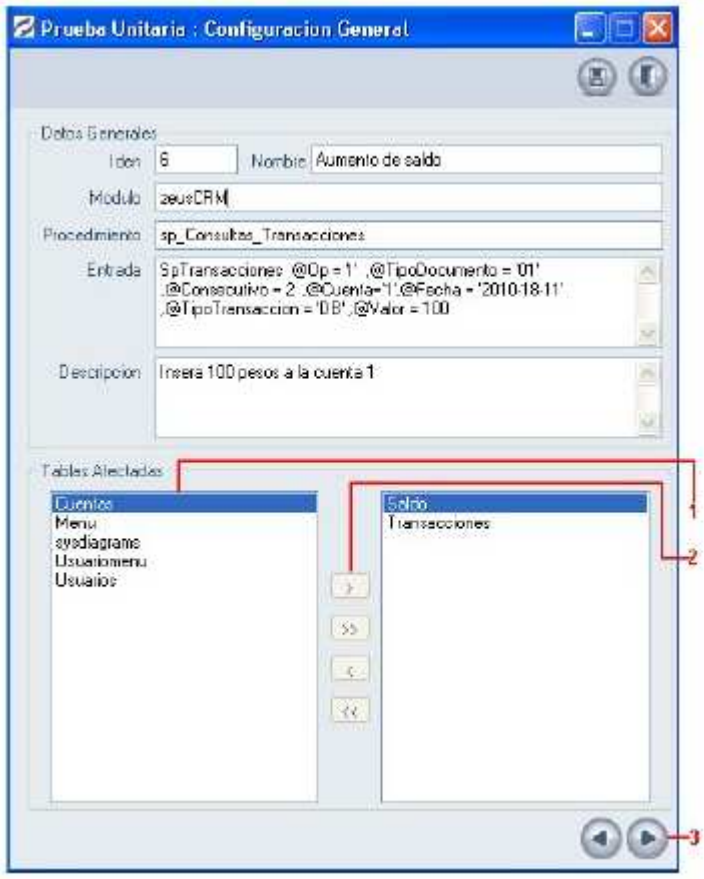


Figura 5.2 Tablas Afectadas

También se puede asignar un registro de la lista de origen a destino pulsando doble click.Cabe destacar que los campos de datos generales en este formulario **no son editables**.

Paso 3: Configurar Salidas

Por cada tabla asignada en el paso anterior se generará una pestaña dentro de este formulario.En la parte superior aparecerán los datos generales a modo de solo lectura para que sirvande guía a la hora de configurar las salidas. Para cada una de las pestañas se debe hacer losiguiente:

1. Note que a la izquierda aparecen todos los campos de la tabla

2. Con los botones de asignaciónenvíe los campos a las listas de la derecha.

3. Campos de búsqueda: Son los campos por los cuales el sistema realizará la búsqueda del registro en la tabla.

4. Campos de verificación: Son los campos que el sistema consultará a ver si losresultados esperados son iguales a los obtenidos.

5. Click en siguiente.

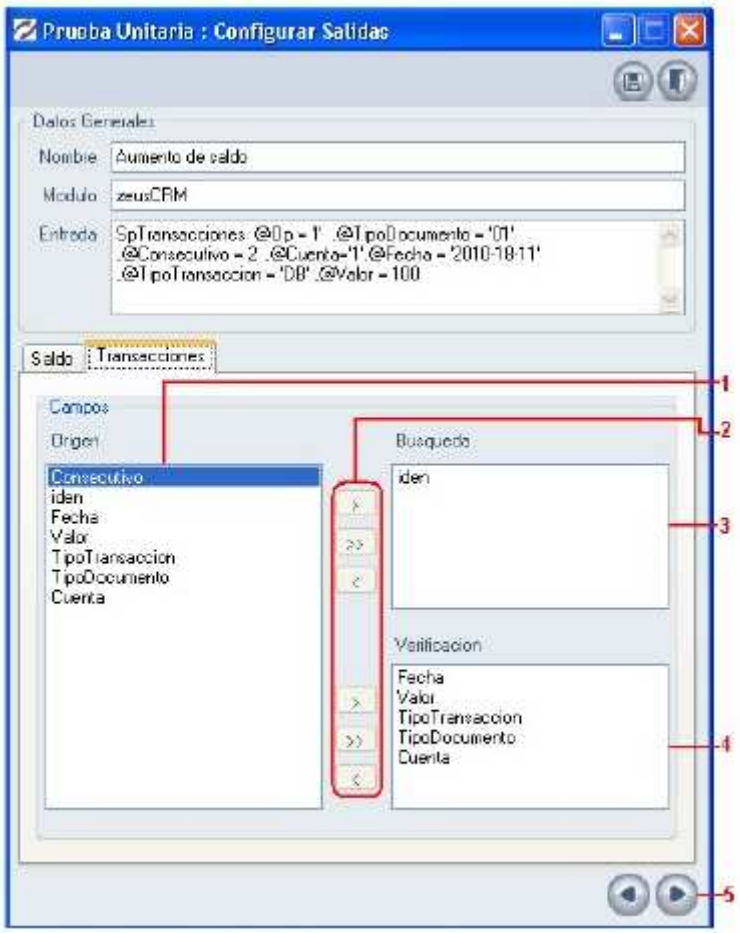


Figura 5.3 Configurar Salidas

Paso 4: Configurar Valores

En este formulario se encuentra en la parte de arriba los datos generales de la prueba, en la parte central del formulario, se genera una pestaña por cada tabla del elegida previamente y una grilla en cada pestaña, cuyas filas corresponden a los campos de búsqueda y verificación seleccionados anteriormente. Los campos

de búsqueda tienen la particularidad de estar sombreados en gris para diferenciarse de los campos de verificación. En la parte inferior además de los botones de desplazamiento se encuentra una leyenda para gestionar estos campos. Las consideraciones que se deben tener en cuenta para este formulario se describen a continuación:

1. Si en un campo de verificación existe un gradiente positivo debe colocarse el signo '+'antes del valor estipulado. (Figura 5.4)

2. Si en un campo de verificación existe un gradiente negativo debe colocarse el signo '&'antes del valor estipulado.

3. Al dar doble click en una celda se despliega un formulario como el de la figura 5.5(Formulario de Subconsulta) donde se debe bien sea buscar o ingresar una nuevaconsulta que será el patrón de búsqueda o verificación para el campo, para insertar labúsqueda dentro de la grilla del formulario padre debe dar click en guardar y dentro dela celda aparecerá un numero precedido de un '#' para que el sistema comprenda quees una subconsulta*.

4. Campo Not (Figura 5.4) este campo es una negación y sirve, por ejemplo en el caso deque se vaya a borrar un registro, entonces la verificación sería que no se encuentre enla base de datos.

5. Pulsar en el botón siguiente para continuar

***Para entender mejor se propone un ejemplo: Supóngase que se va a insertar un registronuevo en la base de datos, entonces el campo de búsqueda debe ser una llave primaria,debido a que se desconoce cuál es la llave primaria al momento de insertar, se puede hacer una subconsulta que busque la última llave generada y a partir de ahí realizar la comparación de los valores que entran en la traza y lo que encuentra en la base de datos.

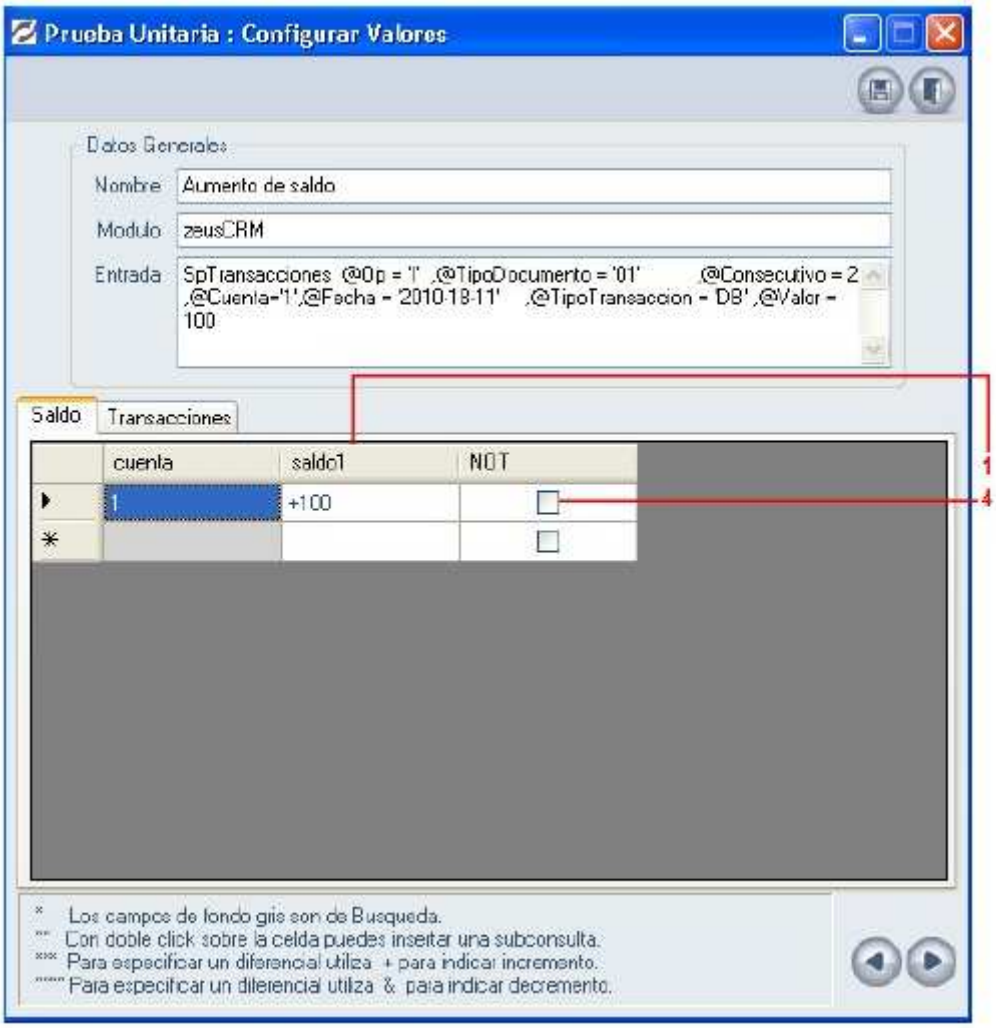


Figura 5.5 Configurar Valores

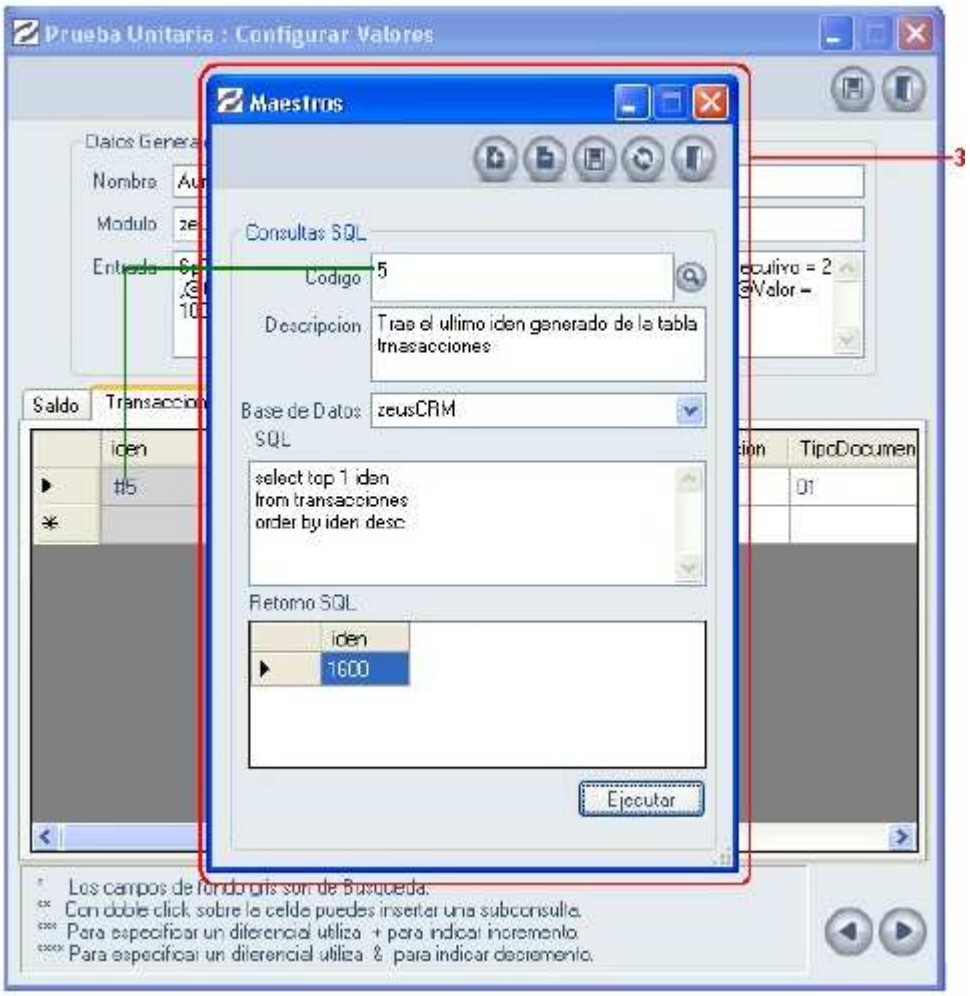


Figura 5.6 Insertar una subconsulta en la Grilla

Paso 6: Configurar Condición

El paso final para la configuración de una prueba unitaria, es establecer una condición paraque esta sea ejecutada. Esta condición es opcional, solo que en ocasiones hay pruebas quese deben ejecutar si existe un parámetro en especia o si el estado de la base de datos ofreceun entorno adecuado, por eso antes de ejecutar la prueba, el software valída la condición. Paraesto se deben seguir los siguientes pasos:

1. Seleccionar la casilla de verificación**¿Desea agregarle una condición?**

2. El Iden se genera automáticamente, pero también funciona como buscador

3. Descripción: Permite especificar una explicación de la condición

4. Base de Datos: Se debe elegir el modulo en el cual se va a evaluar la condición

5. SQL: caja de texto para escribir la consulta. Esta consulta debe retornar un solo valor,de lo contrario aparecerá un mensaje de error.

6. Retorno SQL: Es una grilla que retornara el resultado de la ejecución de la consulta.

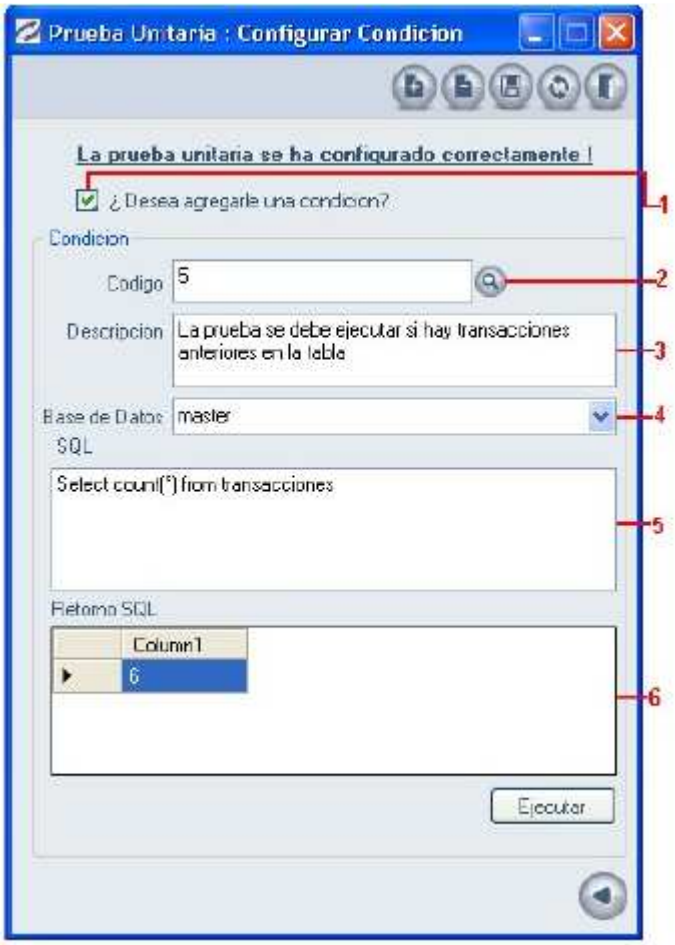


Figura 5.6 Configurar Condición

5.10 Ejecutar prueba

Esta funcionalidad permite la ejecución de una prueba unitaria en el (los) escenario(s) en loscuales la prueba tiene validez. La ventana está conformada por varias regiones

1. Buscador de Zeus: Permite buscar la prueba y esto desencadena que los demás camposde la región Prueba se rellenen con los valores que pertenecen a la prueba.

2. Escenarios: Esta lista permite elegir en que escenarios se va a ejecutar la prueba. Es elúnico campo editable de la región, aparte del buscador.

3. Resultado: Al ejecutar el escenario con el botón de la Barra de Zeus, se muestran losresultados en la grilla de la región Resultado

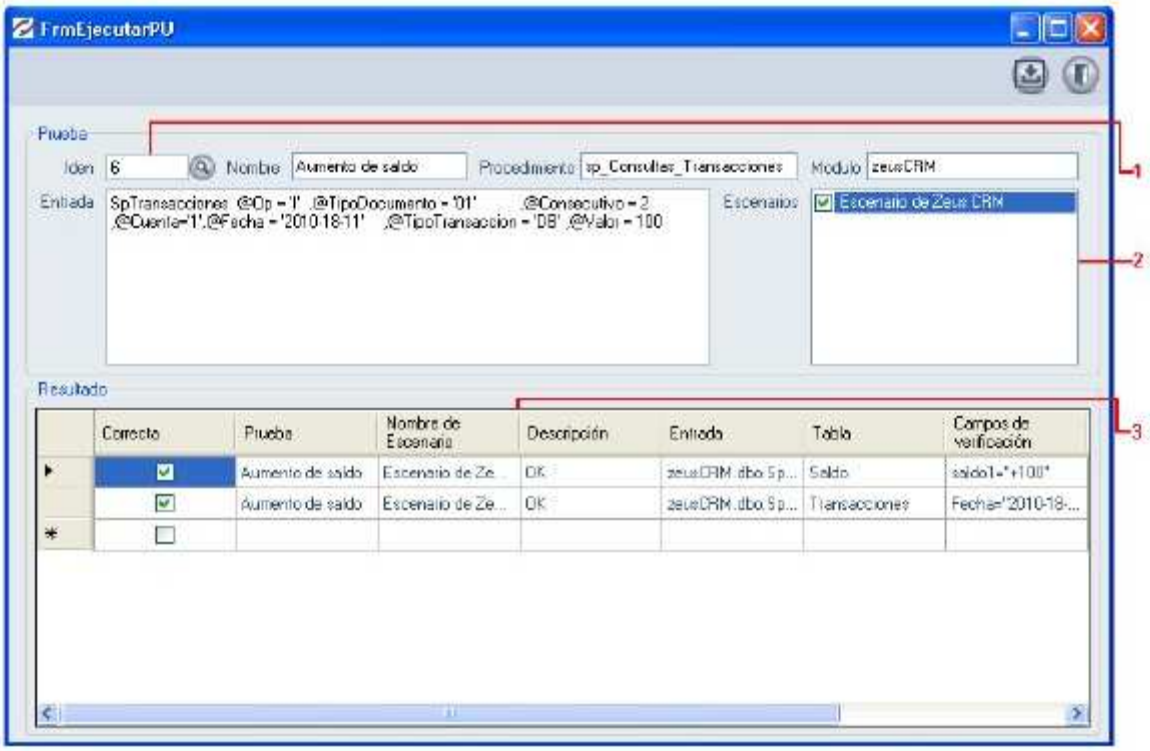


Figura 5.7 Ejecutar Prueba

5.11 Consultar prueba

Esta funcionalidad le permite al usuario consultar cuantas veces ha sido ejecutada una pruebaobteniendo los resultados con fecha de ejecución. Para hacer una consulta exitosa se debenseguir los siguientes pasos:

1. Realizar la búsqueda del Iden de la prueba.

2. Presionar el Botón de Consulta de la Barra de Zeus.

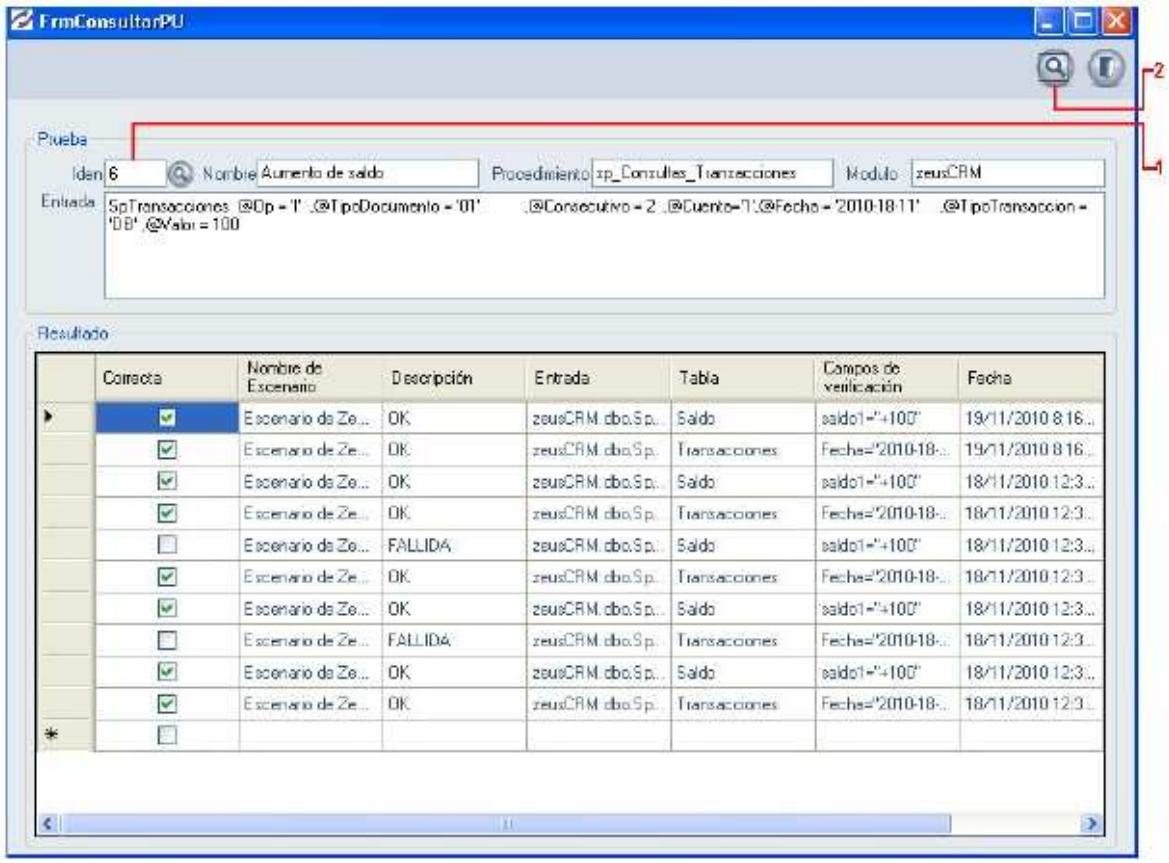


Figura 5.8 Consultar prueba

5.12 Configurar Plan

Esta funcionalidad permite configurar planes de ejecución, teniendo en cuenta los escenariospreviamente configurados. Para su correcta configuración un plan de ejecución se debegestionar los siguientes campos:

1. Iden: Es un campo autogenerado, que también sirve para la búsqueda de otros planes.

2. Nombre: Es el nombre del plan de ejecución

3. Descripción: Explicar bajo qué condiciones se aplica dicho plan de ejecución.

4. Escenarios: Son dos listas de asignación, donde la de la izquierda contiene losescenarios del sistema y la de la derecha los asignados al plan de ejecución.

5. Presionar el botón Guardar de la Barra de Zeus.

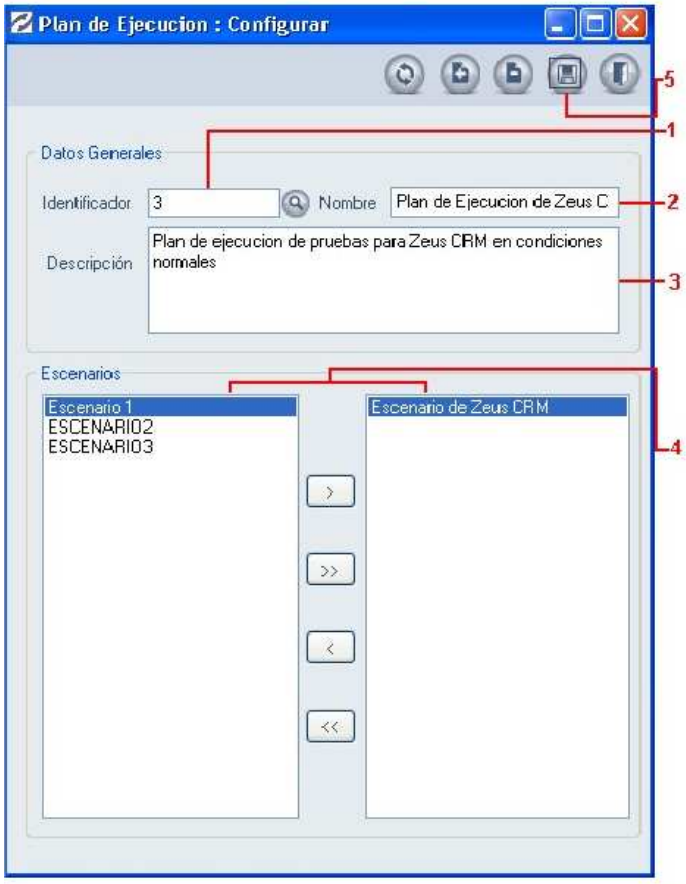


Figura 6.1 Configurar Plan de Ejecución

5.13 Ejecutar Plan

Luego de configurado, para ejecutar un Plan de Ejecución el proceso es similar a como ejecutarun escenario, es decir, esta funcionalidad permite ejecutar todos los escenarios pertenecientesa un escenario mostrando al final una grilla con los resultado.

Para ejecutar un Plan de Ejecución se necesita realizar los siguientes pasos:

1. Buscar el Plan de Ejecución haciendo uso del Buscador de Zeus, lo cual hará que sellenen los campos de nombre, fecha y descripción.

2. Ejecutar el Plan de Ejecución presionando el botón Ejecutar de la barra de Zeus

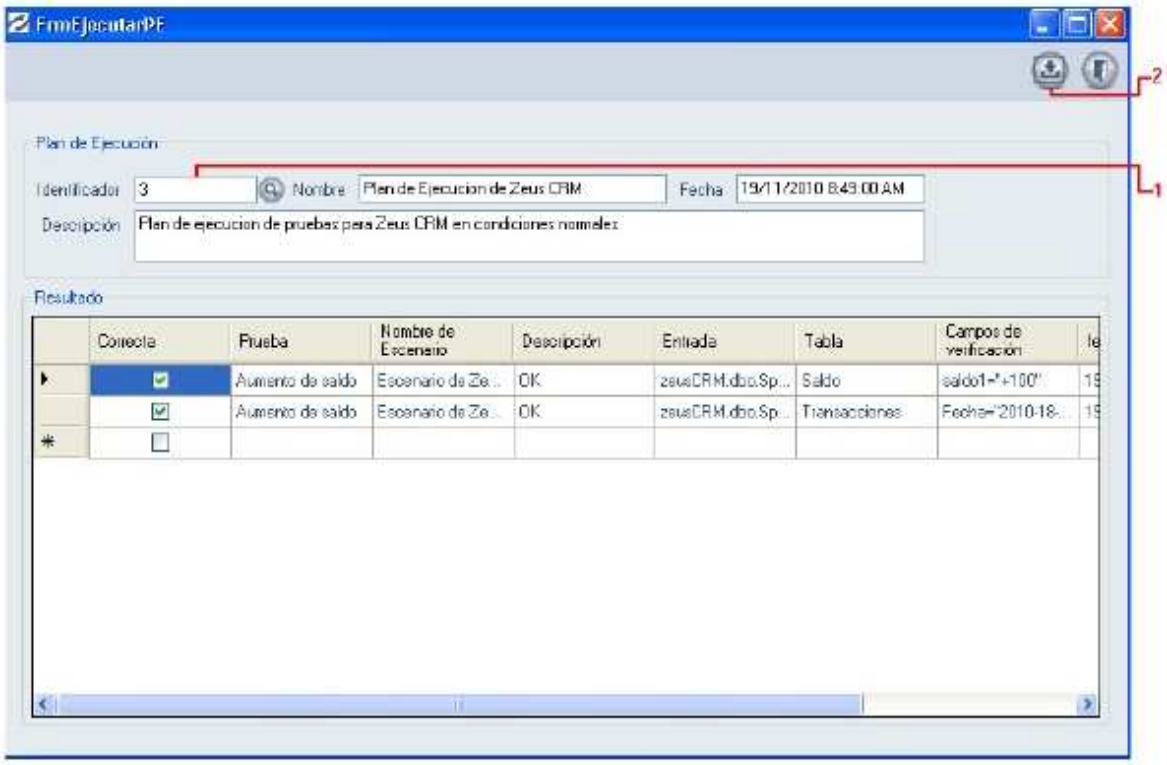


Figura 6.2 Ejecutar Plan

5.14 Consultar Plan

Esta funcionalidad le permite al usuario consultar cuantas veces ha sido ejecutado un Plande Ejecución obteniendo los resultados con fecha. Para hacer una consulta exitosa se debenseguir los siguientes pasos:

1. Realizar la búsqueda del Iden de la prueba.

2. Presionar el Botón de Consulta de la Barra de Zeus.

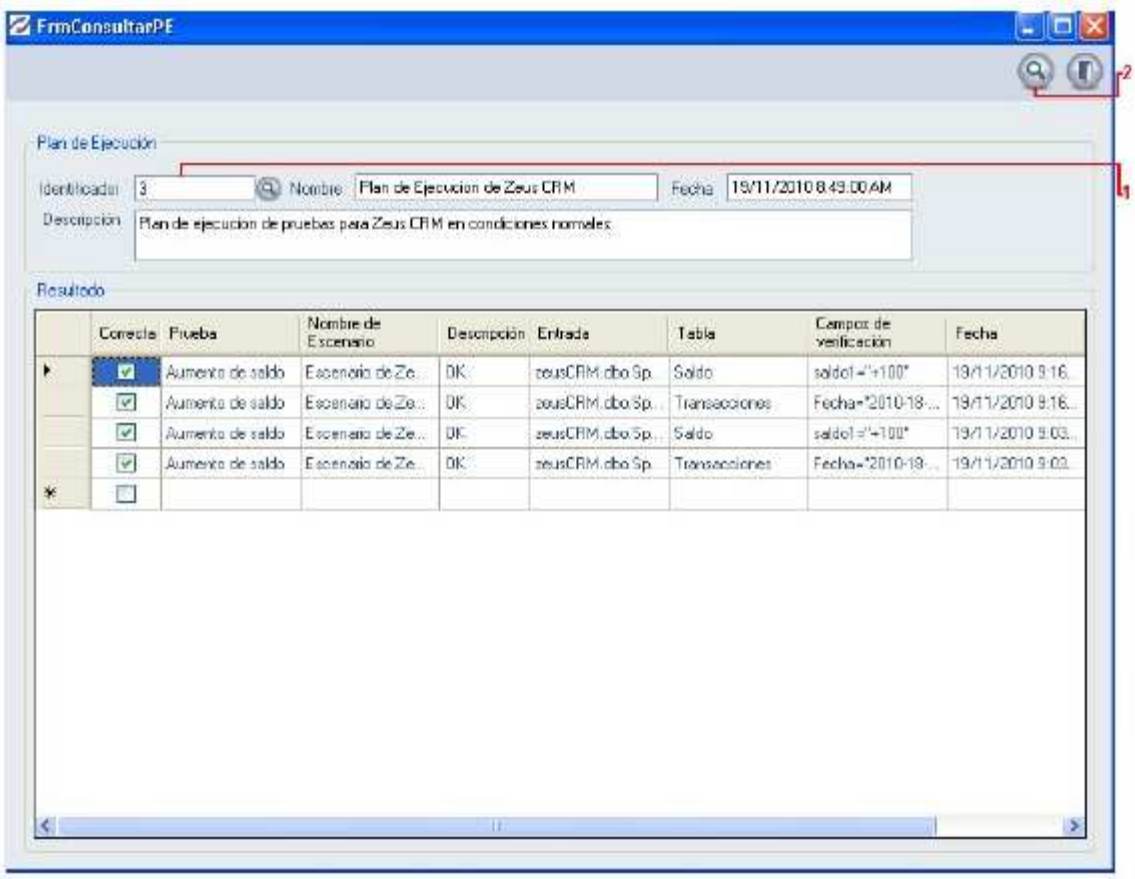


Figura 6.3 Consultar Plan

5.15 Consultar Árbol

A medida que se van configurando pruebas y escenarios, se va generando un Árbol deDependencia. En este árbol se encuentra jerarquizada una estructura que muestra cómo se ven afectadas directamente las tablas por cada prueba de un procedimiento. Jerarquía de mayor a menor es:Escenario, Procedimiento, Prueba, Tabla, Campo; como se muestra en la siguiente Figura

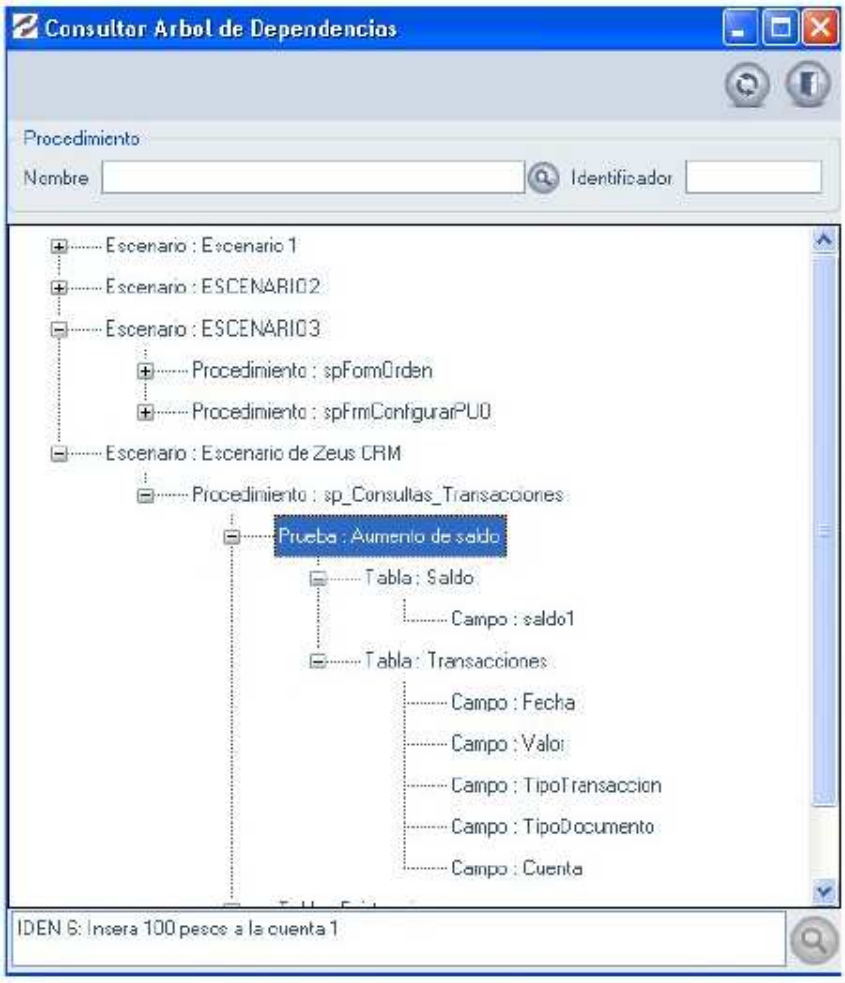


Figura 6.3 Consultar Plan

5.16 Salir

Salida segura del sistema

6. CONCLUSIONES

- Se concluye que el fundamento principalde las pruebas unitarias es asegurar el correcto funcionamiento de las interfaces, o flujo de datos entre componentes de manera tal que a la hora de realizar una unificación de los diferentes componentes que conforman el sistema en general, exista una congruencia que favorezca el desarrollo de la aplicación que se quiere realizar.

- Por otro lado que para poder tener un buen desarrollo de nuestro sistema debemos realizar un buen modelamiento del mismo, porque si no tenemos en cuenta esto incurriremos en la reingeniería.

- También podemos concluir que para que los usuarios puedan interactuar y poner en marcha un sistema debemos detallarlo, ya que ellos no conocen el funcionamiento del sistema y puede generarle dudas a la hora de interactuar con el mismo.

- Finalmente se concluyó que la metodología de desarrollo en cascada, es una metodología que permite avanzar rápido a la hora de desarrollar un proyecto, pero también encontramos que es una metodología que con la cual se incurre en la reingeniería dado que al final del proceso se vuelve a comenzar y aunque por lo general se debe hacer esto dado que es posible que no siempre se encuentren todos los requerimientos desde un comienzo es metodología hace que ese proceso sea más rápido.

7. RECOMENDACIONES

Este proyecto se desarrolló para los estudiantes de la universidad tecnológica de bolívar para que estos puedan tener una idea más clara de lo que debe tener un proyecto y las partes que este lo componen, ya que lo que se busca es mejorar el nivel de conocimiento de los estudiantes enfocado al desarrollo de software utilizando las herramientas que nos brinda Microsoft .NET a través de su herramienta Visual Studio 2008.

Las recomendaciones para proyectos similares a Zeus Pruebas Unitarias®, es tener una buena abstracción del problema, entenderlo, saber cuáles son los requerimientos más importantes, también tener un plan de desarrollo organizado y empezar a desarrollarlos desde el principio del proyecto teniendo en cuenta las buenas practicas del desarrollo, además de tener en cuenta los patrones de diseño para tener un proyecto más escalable y más organizado, ya que con esto nos ayuda a tener claro los demás requerimientos y nos va a resultar más fácil continuar con el proyecto.

8. BIBLIOGRAFIA

- Troelsen Andrew, Pro C# 2008 and the .NET 3.5 Platform, cuartaedición.
- PangoFernández, Yuri J. SQL Server 2008, primera edición.
- Sack Joseph, SQL Server 2008 Transact-SQL Recipes, primera edición
- Troelsen Andrew, Pro C# 2008 and the .NET 3.5 Platform, cuartaedición.
- Rumbaugh James, El Lenguaje Unificado de Modelado.
- Calidad y Software: http://www.calidadysoftware.com
- MSDN Magazine: http://msdn.microsoft.com/es-es/magazine/cc164243.aspx