

## Caso de Estudio

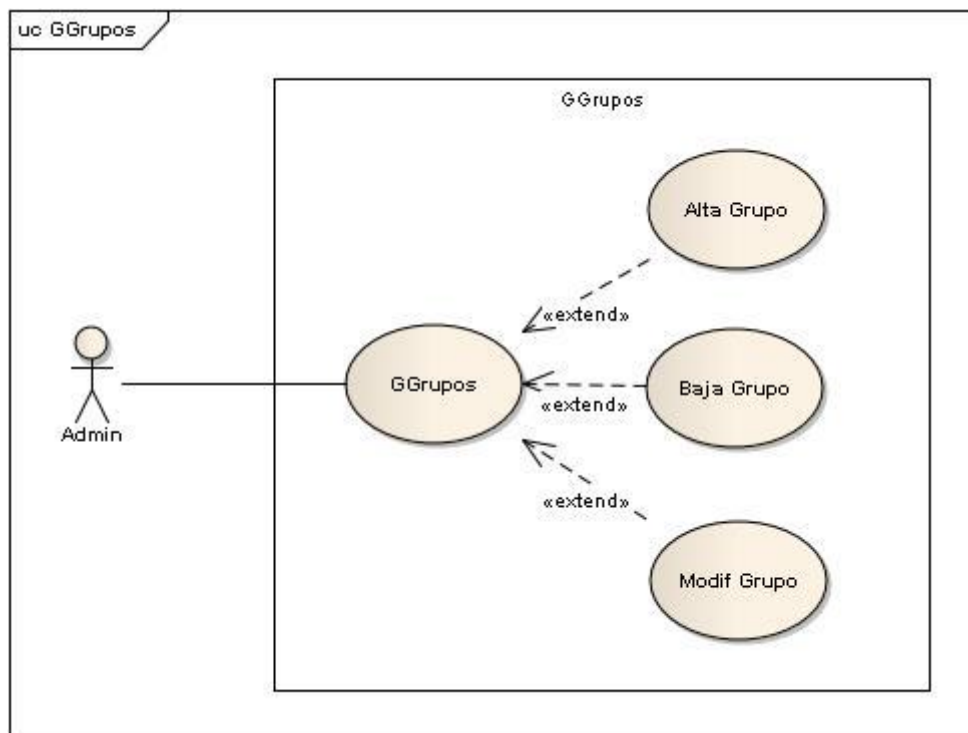
A continuación, presentamos una serie de elementos de estudio que servirán para analizar el caso, adoptar estrategias de prueba y utilizar diferentes técnicas para diseñar los casos de prueba correspondientes.

Realizaremos pruebas sobre los grupos de un modelo de seguridad. El código fuente y base de datos para el diseño y ejecución de las pruebas será proporcionado por la cátedra.

## Requerimientos

Un grupo tendrá: código, descripción, y estado. El código de grupo lo identifica unívocamente, podrá ser alfanumérico, no podrá estar en blanco, ni contener símbolos. La descripción no podrá repetirse, ni estar en blanco.

## Diagrama de Caso de Uso



## Interfaz

**::. GRUPO**

Código:

Descripción:

Estado: ☒ Activo

 Guardar  Cancelar

### Arquitectura: N capas

- VISTA.frmgrupo.cs,
- CONTROLADORA.grupos.cs,
- MODELO.grupos.cs,
- ENTIDADES.grupo.cs

### Clases Relacionadas

## 1. Prueba de Unidad – Prueba de Caja Blanca: Ruta Básica

### Ejemplo

Clase: MODELO.grupos

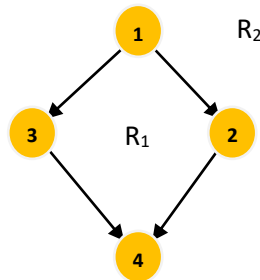
Operación: obtenerGrupo(gCodigo)

```
public ENTIDADES.grupo obtenerGrupo(string gCodigo)
{
    ENTIDADES.grupo eGrupo;
    eGrupo = _grupos.Find(delegate(ENTIDADES.grupo dGrupo) { return
    dGrupo.codigo == gCodigo; });
    ENTIDADES.grupo rGrupo = new ENTIDADES.grupo();
    if (eGrupo != null)
    {
        rGrupo.codigo = eGrupo.codigo;
        rGrupo.descripcion = eGrupo.descripcion;
        rGrupo.estado = eGrupo.estado;
        return rGrupo;
    }
    else
    {
        return null;
    }
}
```

Diagrama de flujo asociado al código:

- 1: Inicio del método `obtenerGrupo`.
- 2: Bloque de código que asigna valores a `rGrupo` y lo devuelve.
- 3: Bloque de código que devuelve `null` si no se encuentra el grupo.
- 4: Fin del método.

### GRAFICA DE FLUJO



### COMPLEJIDAD CICLOMATICA

$V(G) = 2$  regiones

$V(G) = 1$  nodo predicado + 1 = 2

$V(G) = 4$  aristas – 4 nodos + 2 = 2

### RUTAS INDEPENDIENTES

Ruta 1: 1-2-4

Ruta 2: 1-3-4

### CASOS DE PRUEBA

Ruta 1: 1-2-4

Valores de entrada: gCodigo = "LOG" (código de grupo no existente)

Valores de esperados: null (objeto nulo)

Ruta 2: 1-3-4

Valores de entrada: gCodigo = "ADMIN" (código de grupo existente)

Valores de esperados: grupo ADMIN

## RESULTADOS DE LA PRUEBA

Los resultados obtenidos en la ejecución de los casos de prueba fueron los esperados, por ende, la prueba no ha sido exitosa. No hay que realizar correcciones.

## 2. Prueba de Integración – Prueba de Caja Blanca: Ruta Básica

### Consignas

- Utilizando como base el código que se muestra a continuación, dibujar la gráfica de flujo correspondiente.
- Determinar la complejidad ciclomática de la gráfica de flujo resultante.
- Determinar un conjunto básico de rutas linealmente independientes.
- Diseñar los casos de prueba que permitan la ejecución de cada ruta en el conjunto básico.
- Ejecutar los casos de prueba y comparar los resultados obtenidos con los resultados esperados.

Clase: VISTA.frmgrupo.cs

Operación: btnGUARDARClick()

```
private void btnGUARDAR_Click(object sender, EventArgs e)
{
    if (txtDESCRIPCION.Text == "")
    {
        MessageBox.Show("Debe ingresar la descripción", "ATENCION");
        txtDESCRIPCION.Focus();
        return;
    }

    oGrupo.codigo = txtCODIGO.Text;
    oGrupo.descripcion = txtDESCRIPCION.Text;
    oGrupo.estado = chkESTADO.Checked;

    if (MODO_FORM == "A")
    {
        oGRUPOS.ALTA(oGrupo);
    }
    else
    {
        oGRUPOS.MODIFICACION(oGrupo);
    }
    this.DialogResult = DialogResult.OK;
}
```

## GRAFICA DE FLUJO

### **COMPLEJIDAD CICLOMATICA**

$V(G) = \text{regiones} =$

$V(G) = \text{nodos predicado} + 1 =$

$V(G) = \text{aristas} - \text{nodos} + 2 =$

### **RUTAS INDEPENDIENTES**

Ruta 1:

Ruta 2:

Ruta 3:

...

Ruta n:

### **CASOS DE PRUEBA**

Ruta 1:

Valores de entrada:

Valores esperados:

Ruta 2:

Valores de entrada:

Valores esperados:

...

Ruta n:

Valores de entrada:

Valores esperados:

### **RESULTADOS DE LA PRUEBA**

## **3. Prueba de Validación – Prueba de Caja Negra: Partición Equivalente**

### **Consignas**

En base al caso de uso “Alta Grupo”, derivar casos de prueba por el método de Partición Equivalente.

Según la condición de entrada, definir las clases de equivalencia válidas e inválidas para cada entrada del dominio.

Dominio de Entrada

- Código
- Descripción

Ejecutar los casos de prueba para cada entrada del dominio, y validar los resultados de acuerdo a los requerimientos especificados.

### **CASOS DE PRUEBA**

Atributo: Código

Tipo condición:

Entrada Válida:

Entradas Inválidas:

Resultados:

Atributo: Descripción

Tipo condición:

Entrada Válida:

Entradas Inválidas:

Resultados:

### **RESULTADOS DE LA PRUEBA**