

UNIVERSIDAD PRIVADA DE TACNA



INGENIERIA DE SISTEMAS

TITULO:

TRABAJO FINAL DE UNIDAD I

CURSO:

BASE DE DATOS II

DOCENTE(ING):

Patrick Cuadros Quiroga

Integrantes:

Balaguer Valles Angela Lesslie	(2016054494)
Huallpa Castro Leydi Katherine	(2015053230)
Mamani Ayala Brandon Denis	(2015052715)
Pilco Quispe Mireya Flavia	(2015053234)
Quispe Mamani Angelo Brian	(2015052826)
Vizcarra Llanque Jhordy Joel	(2015052719)

Índice

1. PROBLEMA	1
2. MARCO TEORICO	2
3. DESARROLLO	4
3.1. ANALISIS (CASOS DE USO, SECUENCIAS, ACTIVIDADES)	4
3.1.1. CASOS DE USO	4
3.1.2. DIAGRAMA DE OBJETOS	6
3.1.3. DIAGRAMA DE SECUENCIA	7
3.1.4. DIAGRAMA DE ACTIVIDADES	10
3.2. DISEÑO (DIAGRAMA DE CLASES, MODELO ENTIDAD RELACION)	11
3.2.1. DIAGRAMA DE CLASES	11
3.2.2. MODELO ENTIDAD RELACION	12
3.3. METODOS DE CLASES UTILIZADOS	13

1. PROBLEMA

El siguiente proyecto se enmarca en el contexto del repositorio de la Universidad Privada de Tacna, el cual se caracteriza por cumplir con la necesidad de implementación del servicio de repositorio virtual. Para representar en este documento el diseño de distintos diagramas de UML para evaluar los distintos procesos del repositorio para la comunidad universitaria o externa, para mejorar las condiciones de búsqueda y acceso a la información por parte de estudiantes y docentes.

En la primera parte del documento se desarrollará el análisis de los diferentes diagramas de UML de los procesos como el acceso a la información, etc. Posteriormente, se presentará el digrama de clases y el modelo de entidad - relacion que permite conocer cuáles son los objetos que interactuan en el repositorio.

Finalmente se mostrarán los metodos realizados para evaluar su funcionamiento en las pruebas unitarias realizadas en Visual Studio, el modelo en el que se representan los procedimientos y actividades del repositorio para que la Institución los tome como herramientas en el momento que se realice su implementación y así potencie los servicios que esta biblioteca puede llegar a ofrecer a sus usuarios

2. MARCO TEORICO

A través de los años los sistemas de administración de Bases de Datos han evolucionado hacia Sistemas de Administración de Base de Datos Relacionales (RDBMS). Una base de datos relacional es un modelo organizado de entidades que posee características que tienen relaciones entre ellas. Una base de datos relacional bien diseñada provee información de un negocio o un proceso y su uso más común es para almacenar y recuperar información. Entre las mayores ventajas de RDBMS están la forma en la que almacena y recupera información y cómo mantiene la integridad de la misma. Las estructuras RDBMS son fáciles de comprender y construir, pues son lógicamente representadas utilizando Diagramas Entidad-Relación. Las bases de datos relacionales tienen las siguientes características principales:

- • Estructuras. Son objetos que almacenan o acceden a los datos de la base de datos (Tablas, vistas e índices).
- • Tabla. Es un objeto que almacena datos en forma de filas y columnas. Cada tabla tiene una o más columnas y filas. Las columnas guardan una parte de la información sobre cada elemento que queremos guardar en la tabla, cada fila de la tabla conforma un registro. Los datos de una tabla contienen valores atómicos, es decir que contiene elementos indivisibles.
- • Integridad. La integridad de la base de datos se refiere a la validez y la consistencia de los datos.
- • Facilidad de uso. Los usuarios tendrán fácil acceso a los datos. Las complejidades internas son ajenas al usuario, gracias al sistema de administración de la base.
- • Redundancia controlada. Los datos serán almacenados una sola vez excepto cuando existan razones técnicas o económicas que aconsejen el almacenamiento redundante.
- • Seguridad de acceso. Se evitará el acceso no autorizado de datos. Los mismos podrán estar sujetos a diferentes restricciones de acceso para distintos usuarios.
- • Operaciones. Son acciones usadas para definir las Estructuras o manipular los datos de las mismas (SELECT, CREATE)
- • Reglas de integridad. Gobiernan los tipos de acciones permitidas en los datos y la estructura de la Base de Datos (BD). Protegen los datos y estructuras de la BD. (Llaves primarias y foráneas).
- • Identificador único. No pueden existir dos tablas con el mismo nombre, así como no pueden existir dos columnas con el mismo nombre en una misma tabla y los valores almacenados en una columna deben ser del mismo tipo de dato.
- • Clave única. Cada tabla puede tener uno o más campos cuyos valores identifican de forma única cada registro de dicha tabla, es decir, no pueden existir dos o más registros diferentes cuyos valores en dichos campos sean idénticos. Este conjunto de campos se llama clave única.
- • Clave primaria. Una clave primaria es una clave única elegida entre todas las candidatas que define unívocamente a todos los demás atributos de la tabla, para especificar los datos que serán relacionados con las demás tablas. La forma de hacer esto es por medio de claves foráneas. Sólo puede existir una clave primaria por tabla y ningún campo de dicha clave puede contener valores NULL.

- • Dominios. Un dominio describe un conjunto de posibles valores para cierto atributo. Como un dominio restringe los valores del atributo, puede ser considerado como una restricción. Matemáticamente, atribuir un dominio a un atributo significa "todos los valores de este atributo deben de ser elementos del conjunto especificado".
- • Normalización. Las bases de datos relacionales pasan por un proceso al que se le conoce como normalización, el resultado de dicho proceso es un esquema que permite que la base de datos sea usada de manera óptima.
- Una base de datos es un "almacén" que nos permite guardar grandes cantidades de información de forma organizada para que luego podamos encontrar y utilizar fácilmente.

Ademas se esta utilizando Visual studio que ha sido desarrollado con el objetivo de entregar a los usuarios de programación informática un paquete de utilidades simples y accesibles. Es por esto que Visual Studio puede ser usado y fácilmente comprendido por expertos como también por usuarios principiantes. Visual studio es además un lenguaje de programación guiado por eventos que permite mayor operatibilidad y mejores resultados.

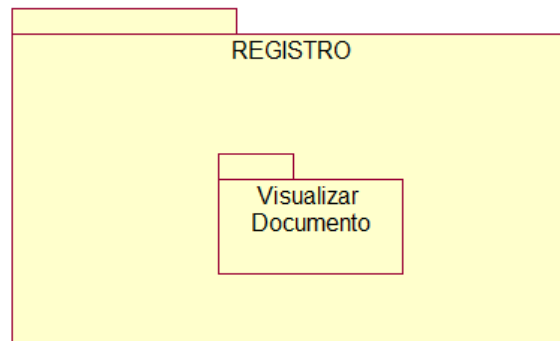
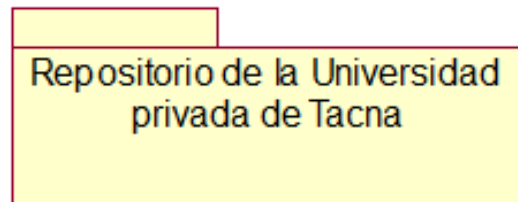
La creación de interfaces gráficas para diferentes utilidades es una de las principales funciones del Visual Studio y es por esto que es altamente usado en espacios profesionales donde se requieren soportes gráficos para mayor organización de los contenidos y materiales. La programación gráfica se puede llevar a cabo directamente ya que el Visual studio no requerirá de los usuarios la escritura de los códigos de programación. Visual studio trabaja a partir de lenguajes RAD, en inglés Rapid Application Development, o desarrollo rápido de aplicaciones específicas para cada necesidad y función. Al mismo tiempo, el Visual studio, gracias a su simple lenguaje, es perfectamente adaptable a las plataformas de los sistemas Windows y es fácilmente transformable a otros lenguajes más complejos.

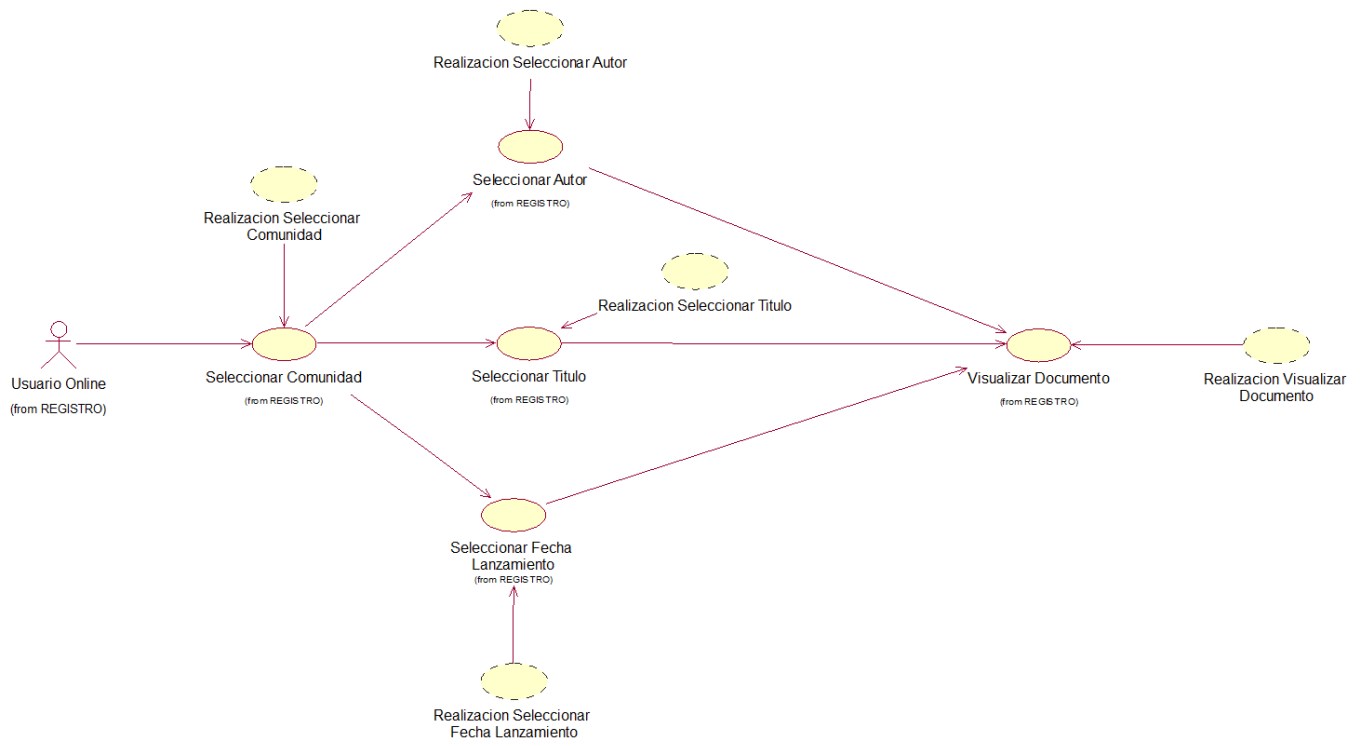
Microsoft ha desarrollado numerosas versiones para Visual studio. Una de las más antiguas data de 1992 y si bien presentaba el lenguaje en forma de texto, permitía ya disfrutar y acceder a algunos de los elementos más importantes del futuro Visual studio. Hoy en día, la versión 2015 es la más difundida a nivel mundial gracias a la combinación de elementos simples y de elementos perfeccionados.

3. DESARROLLO

3.1. ANALISIS (CASOS DE USO, SECUENCIAS, ACTIVIDADES)

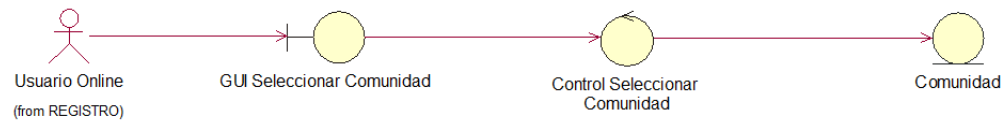
3.1.1. CASOS DE USO



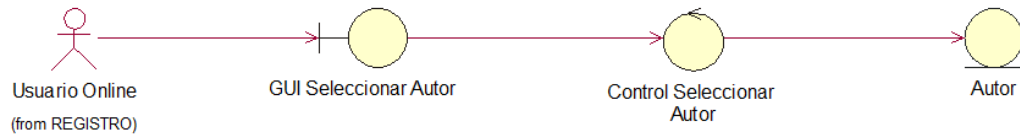


3.1.2. DIAGRAMA DE OBJETOS

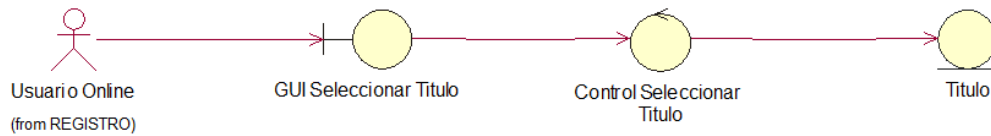
Seleccionar Comunidad



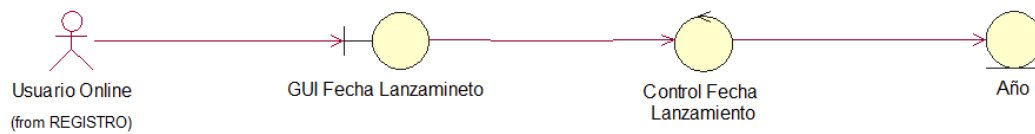
Seleccionar Autor



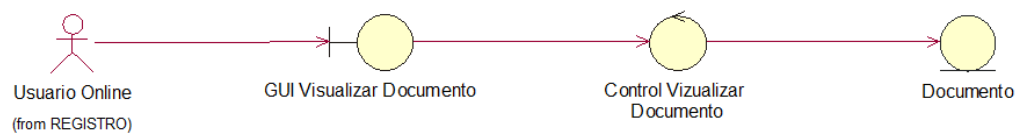
Seleccionar Titulo



Seleccionar Año

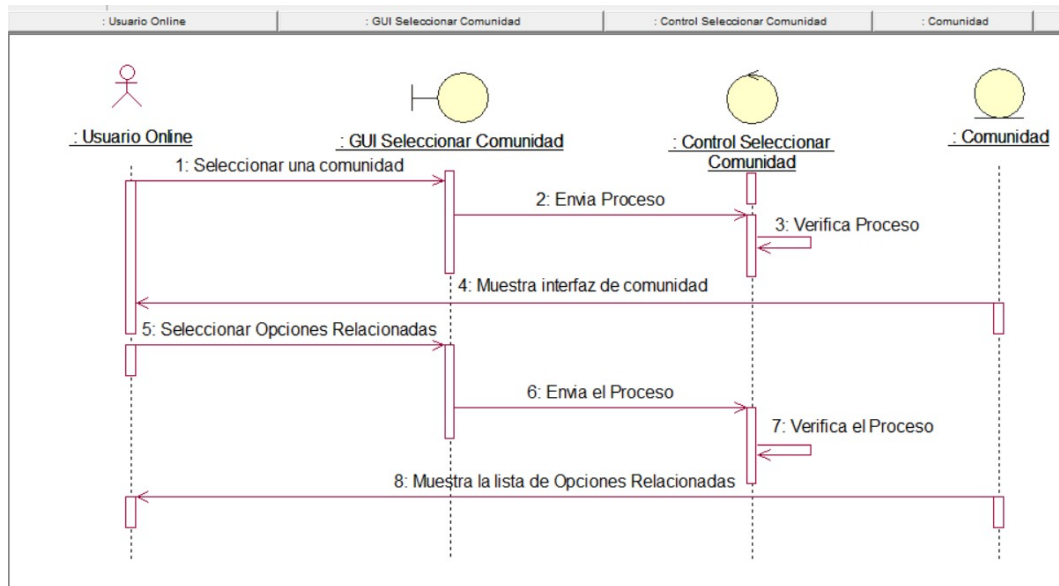


Seleccionar Documento

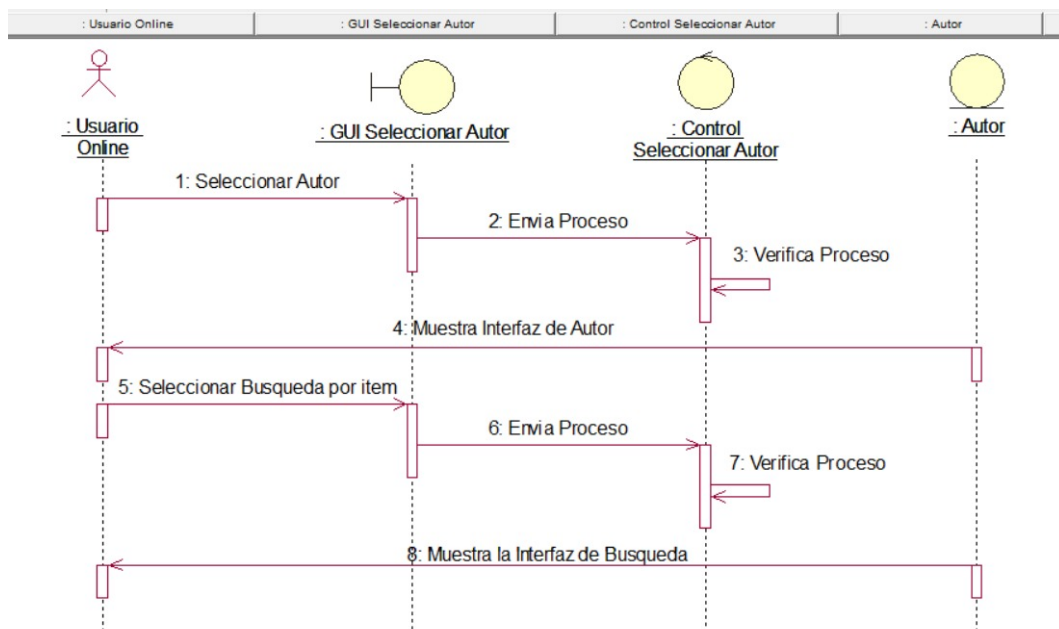


3.1.3. DIAGRAMA DE SECUENCIA

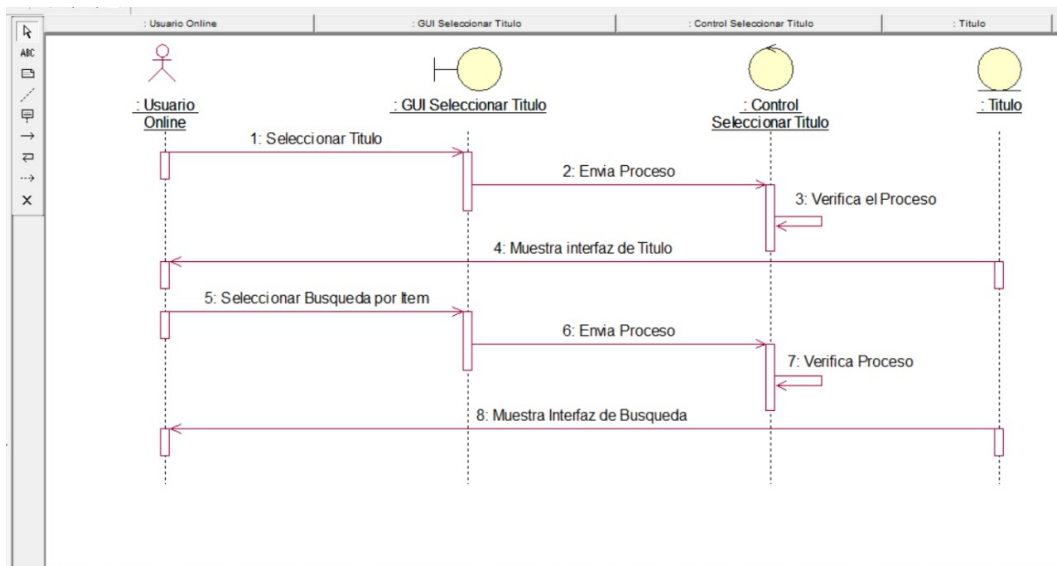
Seleccionar Comunidad



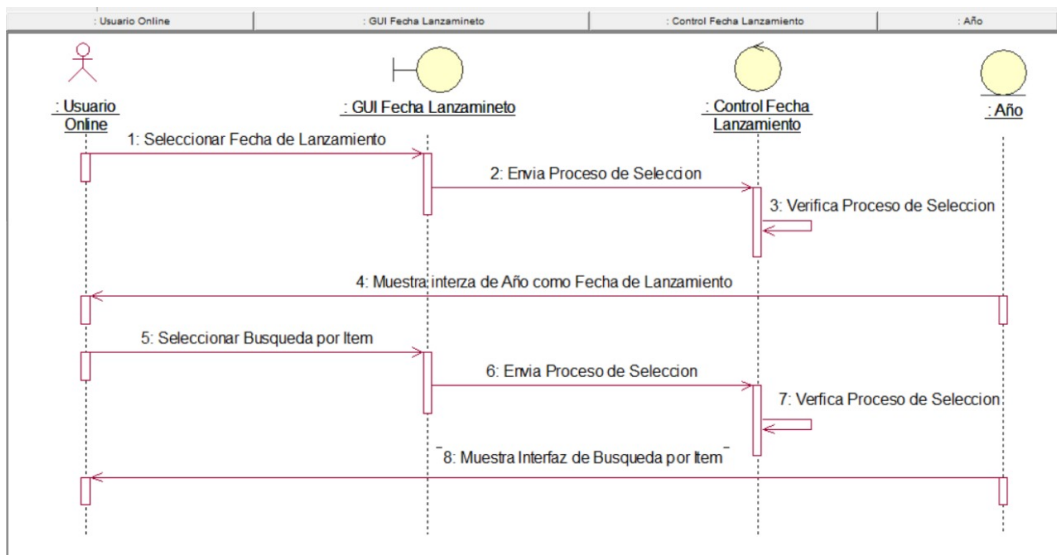
Seleccionar Autor



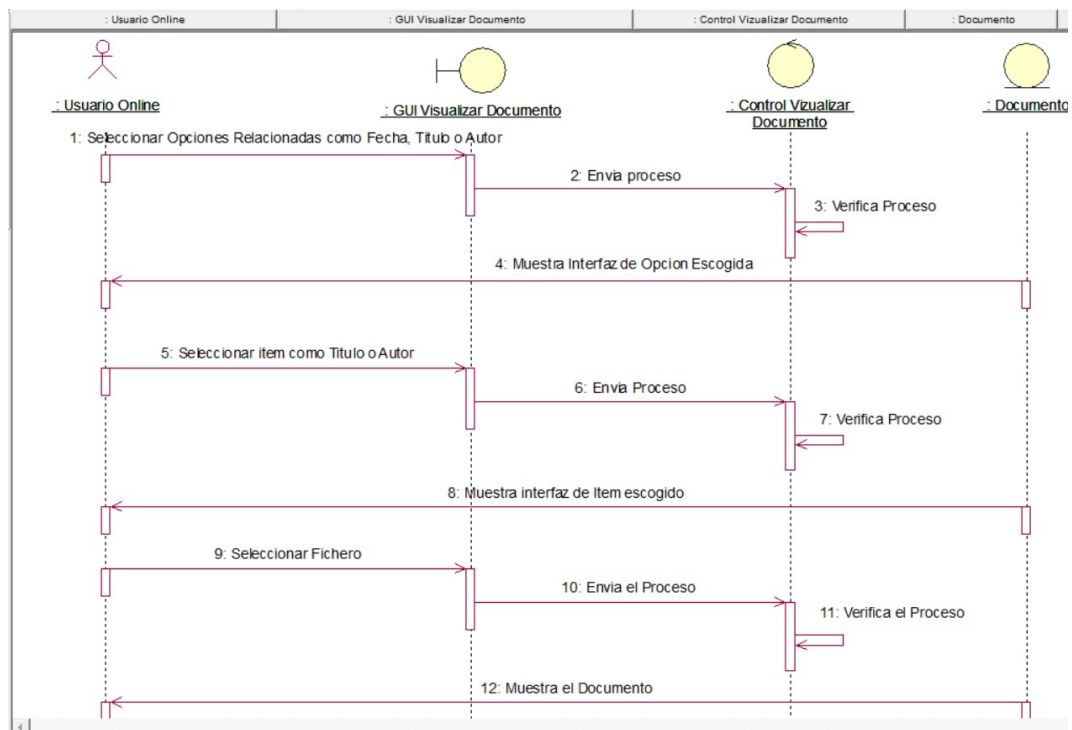
Seleccionar Titulo



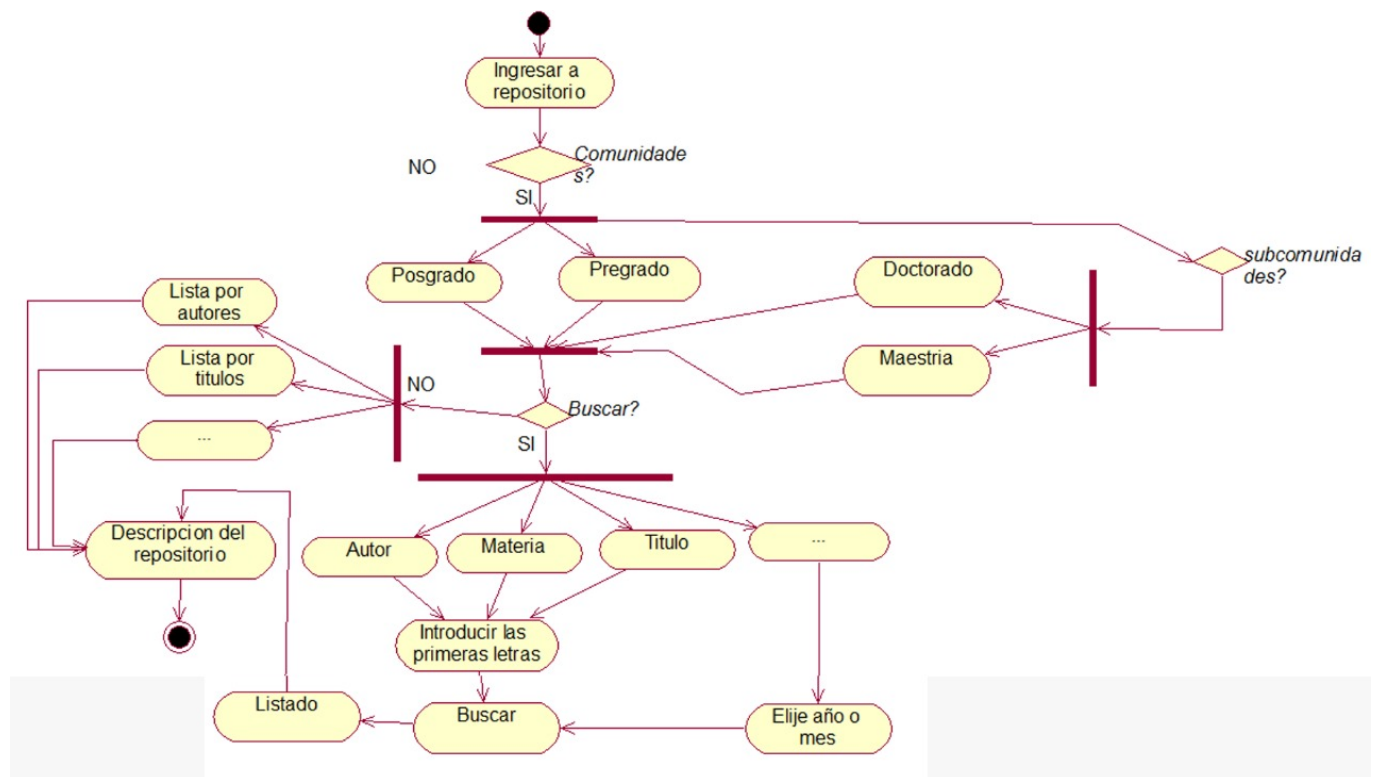
Seleccionar Año



Seleccionar Documento

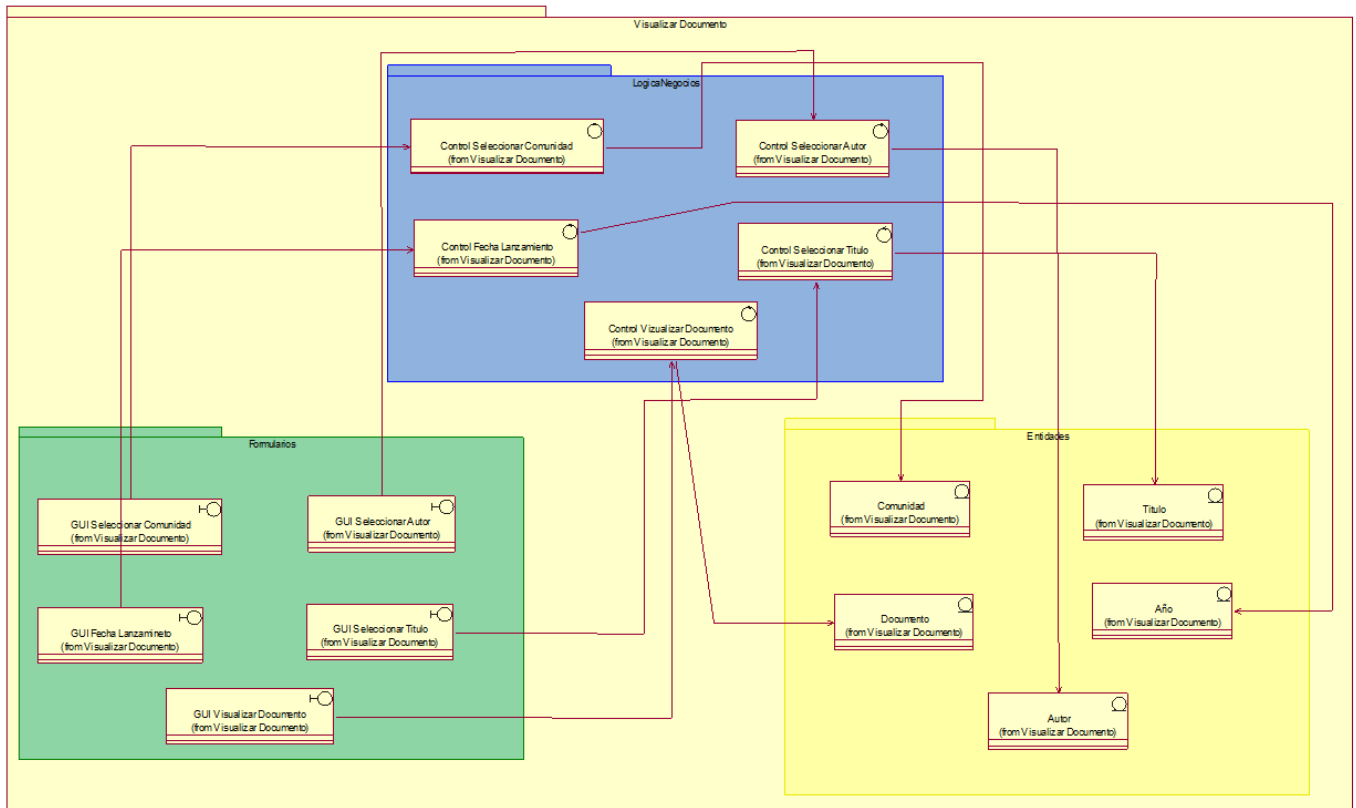


3.1.4. DIAGRAMA DE ACTIVIDADES

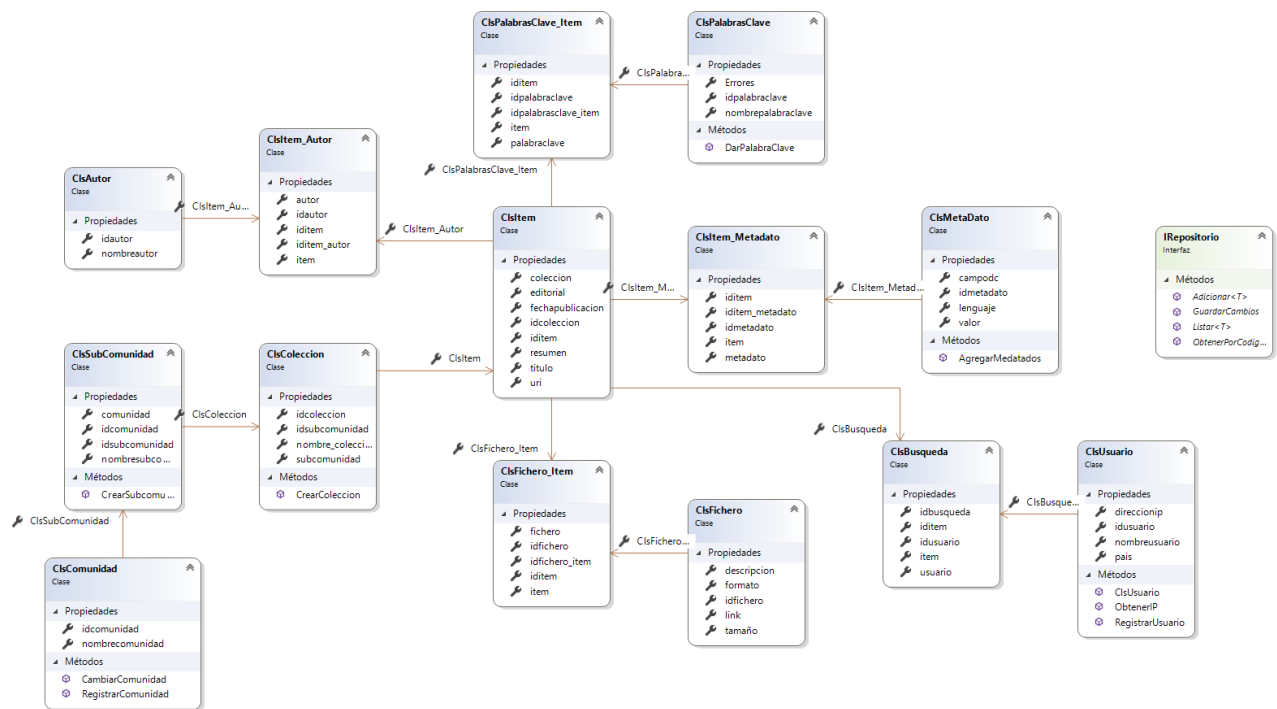


3.2. DISEÑO (DIAGRAMA DE CLASES, MODELO ENTIDAD RELACION)

3.2.1. DIAGRAMA DE CLASES



3.2.2. MODELO ENTIDAD RELACION



3.3. METODOS DE CLASES UTILIZADOS

Registrar Comunidad

```
namespace Repositorio.Pruebas.Unitarias
{
    [TestClass]
    0 referencias
    public class Comunidad_Test
    {
        ClsComunidad createcomunidad = new ClsComunidad();
        [TestMethod]
        0 referencias
        public void RegistorComunidadCompleto_Test()
        {
            //Arrange

            var _idcomunidad = 10;
            var _nombrecomunidad = "";

            //Act

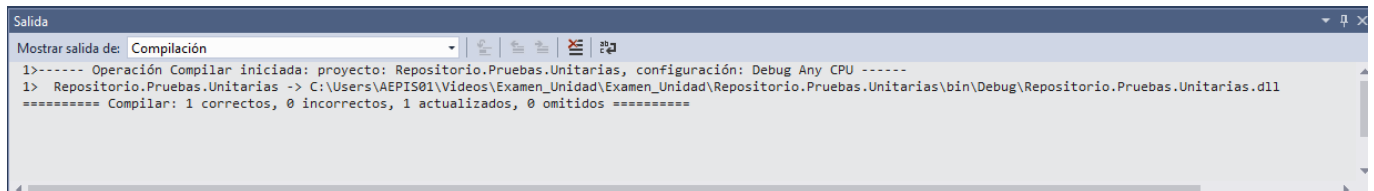
            var result = createcomunidad.RegistrarComunidad(_idcomunidad,_nombrecomunidad);

            //Assert
            Assert.IsNotNull(result);
        }

        0 referencias
        public void CambioComunidadCorrecto_Test()
        {
            //Arrange
            var _nuevonombrecomunidad = "Marco Antonio";
            var _nombrecomunidad = createcomunidad.nombrecomunidad;

            //Act
            var result = createcomunidad.CambiarComunidad(_nuevonombrecomunidad);

            //Assert
            Assert.AreEqual(_nombrecomunidad,_nuevonombrecomunidad);
        }
    }
}
```



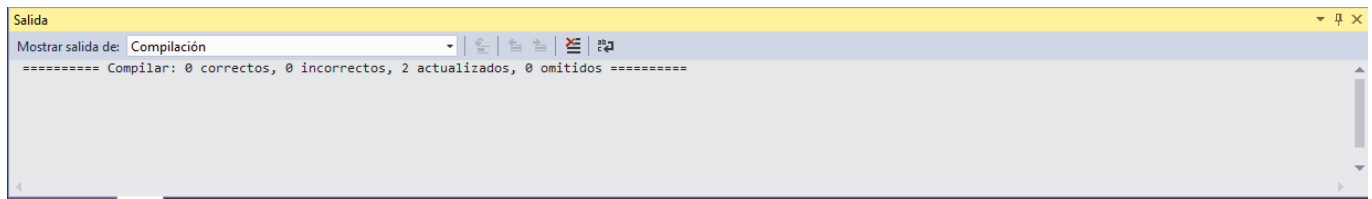
```
Salida
Mostrar salida de: Compilación
1>----- Operación Compilar iniciada: proyecto: Repositorio.Pruebas.Unitarias, configuración: Debug Any CPU -----
1> Repositorio.Pruebas.Unitarias -> C:\Users\AEPIS01\Videos\Examen_Unidad\Examen_Unidad\Repositorio.Pruebas.Unitarias\bin\Debug\Repositorio.Pruebas.Unitarias.dll
***** Compilar: 1 correctos, 0 incorrectos, 1 actualizados, 0 omitidos *****
```

Palabara Clave

```

5 namespace Repositorio.Pruebas.Unitarias._Pruebas_Unitarias
6 {
7     [TestClass]
8     0 referencias
9     public class PalabraClave_Test
10    {
11        [TestMethod]
12        0 referencias
13        public void ObtenerPalabraClave()
14        {
15            ClsPalabrasClave words = new ClsPalabrasClave();
16            var _word = "Administracion de Negocios";
17            var result = words.DarPalabraClave(_word);
18
19            Assert.IsNotNull(result);
20        }
21    }
22 }

```



Registrar Subcomunidad

```

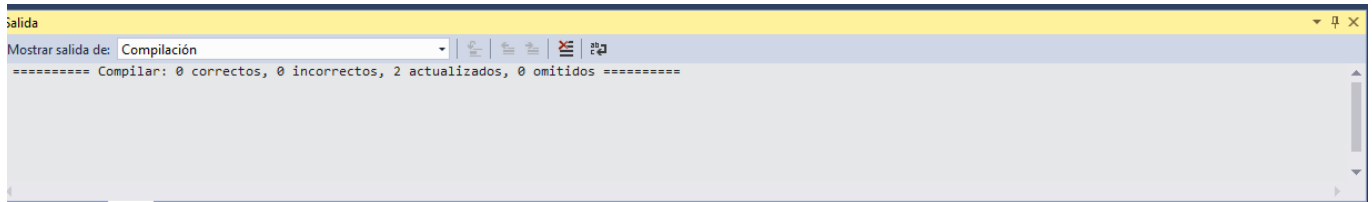
namespace Repositorio.Pruebas.Unitarias._Pruebas_Unitarias
{
    [TestClass]
    0 referencias
    public class Subcomunidad_Test
    {
        0 referencias
        public void RegistorSubcomunidadCompleto_Test()
        {
            //Arrange
            ClsSubComunidad createsubc = new ClsSubComunidad();
            var _idcomunidad = 10;
            var _nombresubcomunidad = "";

            //Act

            var result = createsubc.CrearSubcomunidad(_nombresubcomunidad, _idcomunidad);

            //Assert
            Assert.IsNotNull(result);
        }
    }
}

```

Registrar Usuario

```
namespace Repositorio.Pruebas.Unitarias
{
    [TestClass]
    O referencias
    public class Usuario_Test
    {
        [TestMethod]
        O referencias
        public void RegistroCompleto()
        {
            ClsUsuario createuser = new ClsUsuario();
            //Arrange
            var _stringip = "";

            var _idusuario = 1;
            var _nombreusuario = "Jhordy Joel";
            var _direccionip = createuser.ObtenerIP(_stringip);
            var _pais = "Ucrania";

            //Act
            var result_rc = createuser.RegistrarUsuario(_idusuario,
                                                         _nombreusuario,
                                                         _direccionip,
                                                         _pais);

            //Assert
            Assert.IsNotNull(result_rc);
        }
    }
}
```

[TestMethod]

0 referencias

```
public void RegistroIncompleto()
{
    //Arrange
    var _idusuario = 0;
    var _nombreusuario = "";
    var _direccionip = "";
    var _pais = "";
    ClsUsuario createuser = new ClsUsuario();

    //Act
    var result_ri = createuser.RegistrarUsuario(_idusuario,
                                                _nombreusuario,
                                                _direccionip,
                                                _pais);

    //Assert

    Assert.IsNull(result_ri);
}
```

Salida

Mostrar salida de: Compilación

```
1>----- Operación Compilar iniciada: proyecto: Repositorio.Pruebas.Unitarias, configuración: Debug Any CPU -----
1> Repositorio.Pruebas.Unitarias -> C:\Users\AEPIS01\Videos\Examen_Unidad\Examen_Unidad\Repositorio.Pruebas.Unitarias\bin\Debug\Repositorio.Pruebas.Unitarias.dll
***** Compilar: 1 correctos, 0 incorrectos, 1 actualizados, 0 omitidos *****
```