IES Augusto González de Linares.

Tarea 5:

Diseño orientado a objetos.

Elaboración de diagramas estructurales.

ENTORNOS DE DESARROLLO.

(DAM_ED)

Ciclo formativo

Desarrollo De Aplicaciones Multiplataforma

(DAM)

CURSO:

2021/2022

Francisco M. Colls Gutiérrez.

0. Leyenda de colores.

Posibles Clases.

Posibles Atributos

Relaciones/Consideraciones.

1. Obtención de Clases, atributos y métodos.

Se desea modelar una aplicación que permita gestionar la administración de una comunidad de vecinos. La empresa de administración de fincas puede gestionar varias comunidades cada una de ellas con un código, dirección, vecinos, saldo, facturas, y un código para el administrador responsable.

Clase	**Comunidad**	
Atributos	-idComunidad	string
	-dirección	string
	-vecinos	vecino
	-saldo	double
	-facturas	factura
	-administrador	trabajador
Métodos	+comunidad()	void
	+getter / +setter	

- Los atributos vecinos y facturas son unas estructuras de almacenamiento, por lo tanto son declarados según el tipo de dato que van a almacenar.

- El atributo administrador es una instanciación de la clase trabajador.

- Método constructor de la clase +cominidad().

- Métodos getter y setter para el resto de atributos, los cuales no fueron incluidos en el diagrama para resaltar solamanete los métodos particulares, pero se entiende que para desarrollar realmente el programa es necesaria la implementación de dichos métodos.

Francisco M. Colls Gutiérrez.

Los usuarios del sistema pueden ser los vecinos y los trabajadores de la empresa de administración de fincas. De los vecinos nos interesa su código de inmueble. Del inmueble nos interesa saber si tiene o no cuotas pendientes (si es o no moroso) que se visualiza con el saldo total el código del inmueble y el tipo (piso, garaje o trastero). Un mismo vecino puede ser propietario de varios inmuebles, por lo que se deben poder asignar varios inmuebles a un mismo vecino. De los trabajadores nos interesa conocer su nombre, su DNI, y su cargo. Existe un vecino concreto que es el presidente y existe un trabajador concreto que es el administrador.

Los vecinos pueden registrarse en el sitio web. Cuando un usuario se hace cliente debe proporcionar los siguientes datos: nombre completo, DNI, correo electrónico y dirección además de los códigos de los inmuebles de los que es propietario, así como el número de cuenta bancaria. Los vecinos pueden visualizar sus pisos y pagar su deuda siempre. Así mismo puede modificar sus datos personales.

Clase	***Usuario***	**Abstracta**
Atributos	-nombre	string
	-apellidos	string
	-dni	string
Métodos	+usuario()	void
	+getter / +setter	

- Los usuarios de la aplicación son los trabajadores de la empresa y los vecinos de la comunidad, de ambos grupos se necesitan una serie de datos comúnes como son el nombre, apellidos y dni. Por lo que, se creo una clase llamada usuario que contiene éstos atributos y las clases trabajador y vecino que heredan de usuario.

Clase	**Trabajador**	
Atributos	-cargo	string
Métodos	+trabajador()	void
	+getter / +setter	

Francisco M. Colls Gutiérrez.

Clase	Vecino	
Atributos	-email	string
	-dirección	string
	-inmuebles	inmueble
	-iban	string
Métodos	+vecino()	void
	+getter / +setter	
	+verPisos()	
	+pagarDeudas()	
	+modificarDatos()	

- En el caso de los vecinos se especificó que a través de la web ellos pueden visualizar sus pisos y pagar su deuda siempre. Así mismo puede modificar sus datos personales. Por lo que se crearon los métodos verPisos, pagarDeudas y modificarDatos.

Clase	Inmueble	
Atributos	-cuotasPendientes	boolean
	-idInmueble	string
	-tipo	string
Métodos	+inmueble()	void
	+getter / +setter	

Los usuarios (vecinos y trabajadores) navegan por la web para ver las facturas de la comunidad, que pueden estar pagadas o pendientes de pago. De las facturas nos interesa el nombre de la empresa, el CIF, descripción, fecha e importe.

Clase	Factura	
Atributos	-nombreEmpresa	String
	-cif	string
	-descripcion	string
	-fecha	date
	-importe	double
Métodos	+factura()	void
	+getter / +setter	

Por otro lado, los usuarios deben poder visualizar las actas de las reuniones de vecinos, donde aparecerá la fecha, los códigos de los vecinos que han asistido, y los ítems del orden del día sabiendo si han sido aprobados o rechazados en junta, cada acta corresponde con una comunidad concreta.

Los trabajadores pueden solicitar servicios a empresas para que realicen un servicio en la comunidad. Dichas empresas generarán facturas que podrán pagar con el saldo de la comunidad. No se da de alta en el sistema ninguna empresa que no haya generado factura.

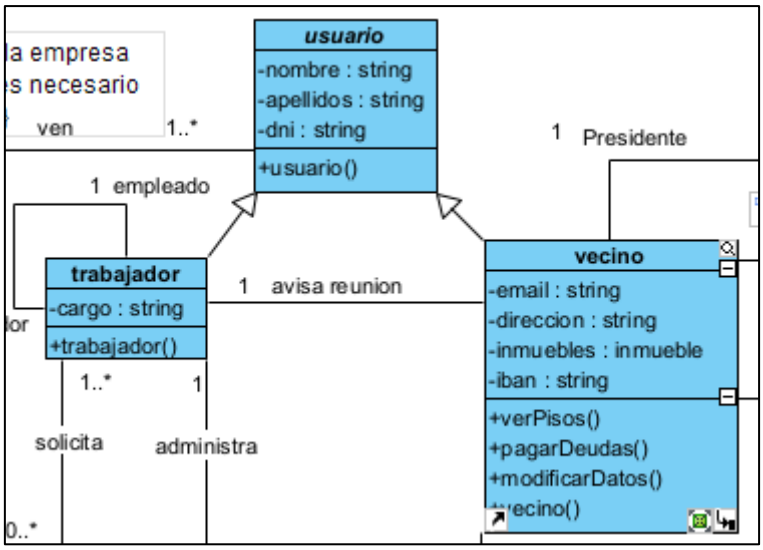
Clase	**Servicios**	
Atributos	-facturas	factura
	-saldo	double
Métodos	+servicios()	void
	+getter / +setter	

Las reuniones de vecinos las convoca un vecino que es presidente de la comunidad que avisa al administrador de fincas que es quien convoca oficialmente la reunión mandando una carta a cada vecino. sería un tipo de empleado encargado de realizar el conteo de vecinos que asisten a la comunidad y contar el número de votos a favor y en contra de cada punto del orden del día. Los pisos con cuotas pendientes (morosos) no tienen derecho a voto en la reunión.

Clase	**Reuniones**	
Atributos	-asistentes	int
	-votosAFavor	int
	-votosEnContra	int
Métodos	+reuniones()	void
	+getter / +setter	

Clase	**Actas**	
Atributos	-fecha	date
	-vecinosAsistentes	vecino
	-items	Boolean
Métodos	+actas()	void
	+getter / +setter	

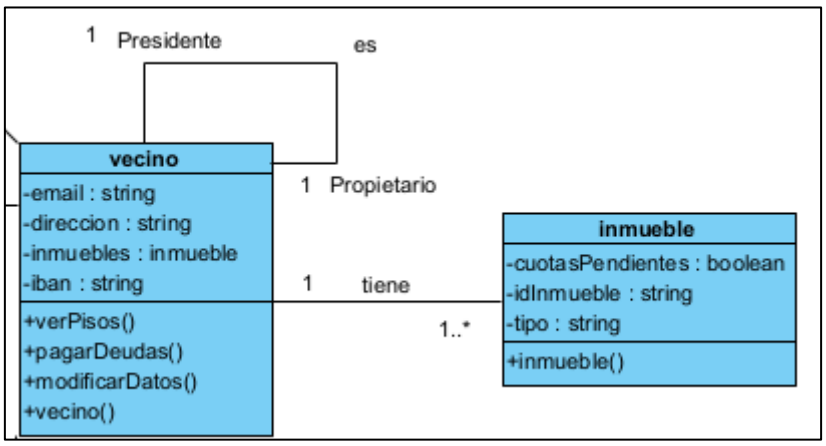
2. Obtención de relaciones.



La clase trabajador y vecino heredan de usuario.

Relación de asociación entre vecino y trabajador.

El presidente de la comunidad (1) avisa al administrador (1) para convocar la reunión.



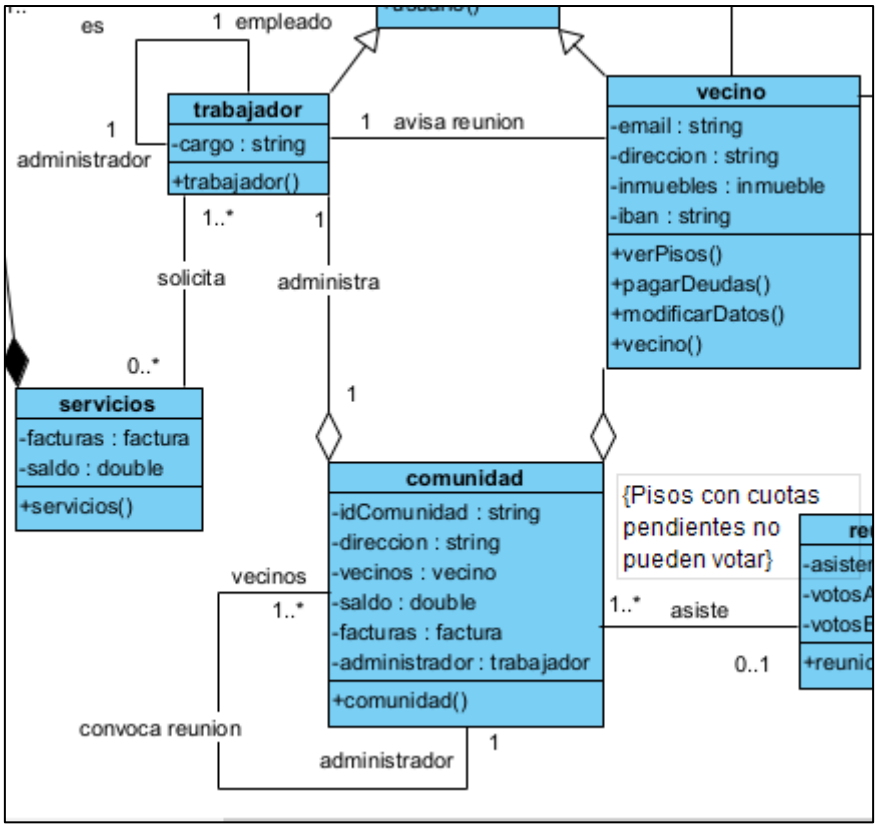
Relaciones unarias de una clase consigo misma (vecino).

1 vecino es propietario, y 1 vecino es presidente de la comunidad.

Relación de asociación entre vecino e inmueble.

1 vecino tiene 1 o varios inmuebles, y 1 inmueble pertenece a 1 vecino.

Francisco M. Colls Gutiérrez.



Comunidad tiene una **relación de agregación** con las clases vecino y trabajador.

Relaciones unarias de una clase consigo misma (trabajador).

1 trabajador es administrador, y 1 trabajador es empleado de la empresa gestora del programa.

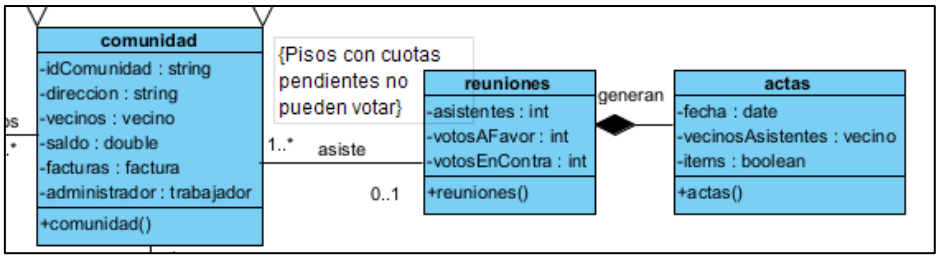
Relación de asociación entre trabajador y servicios.

Los trabajadores pueden solicitar 0 o varios servicios a una empresa, y un servicio es solicitado por 1 o varios trabajadores.

Relaciones unarias de una clase consigo misma (comunidad).

En la comunidad de vecinos la reunión es convocada por 1 administrador, y en la comidad de vecinos son convocados 1 o muchos vecinos a la reunión.

Francisco M. Colls Gutiérrez.



Relación de asociación entre comunidad y reuniones.

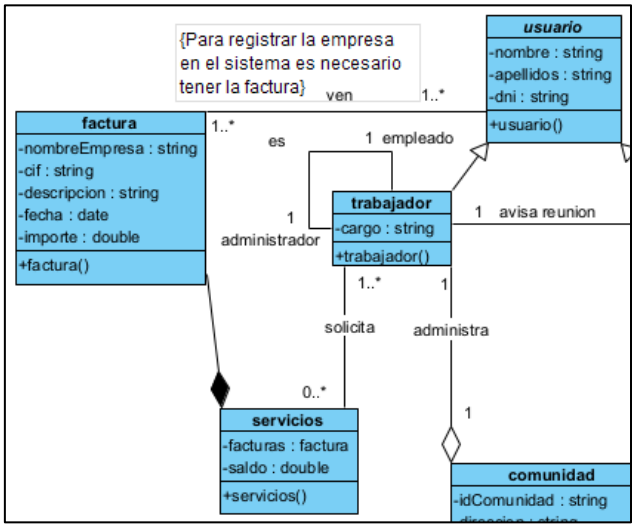
Los integreantes de la comunidad (vecinos y administrador) asisten 0 o 1 reunión.

A las reuniones asisten 1 o muchas personas (integrantes de la comunidad).

Restricción a los asistentes de las reuniones, se especifica entre llaves {}.

Los pisos con cuotas pendientes (morosos) no tienen derecho a voto en la reunión.

Relación de composición entre reuniones y actas, ya que si no existen las reuniones no se generan las actas.



Relación de composición entre servicios y factura, ya que si no existen los servicios no se generan las facturas.

Relación de asociación entre usuario y factura.

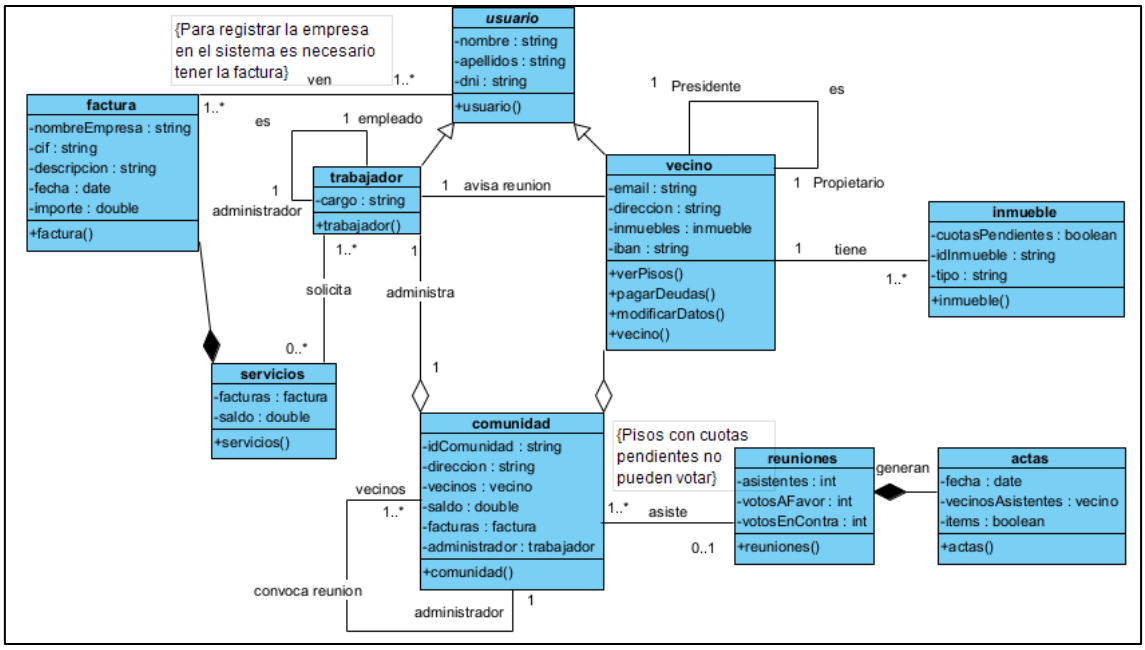
Los usuarios pueden ver 1 o varias facturas, las facturas son vistas por 1 o varios usuarios.

Restriccíon de agregacíon. Si no se ha generado factura no se puede agregar al sistema, por lo tanto esa factura no puede ser visualizada por los usuarios.

Francisco M. Colls Gutiérrez.

3. Diagrama final.

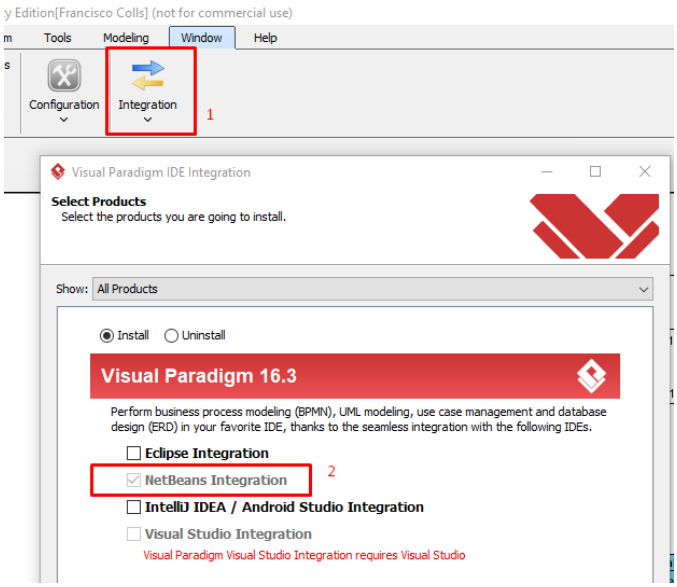
Durante el proceso de creación del diagrama en el programa Visual Paradigm algunas cosas fueron cambiando sobre todo las relaciones entre ciertas clases que no eran del todo claras, como es el caso de las 3 relaciones unarias existentes en el diagrama, las cuales fueron las últimas en agregarse.



4. Importar el proyecto creado VP-UML en un proyecto de NetBeans.

1.- Debemos integrar Visual Paradigm con NetBeans.

Hacemos clic en Integration > Seleccionamos NetBeans > Buscamos la ruta donde se encuentre NetBeans > Aceptar.

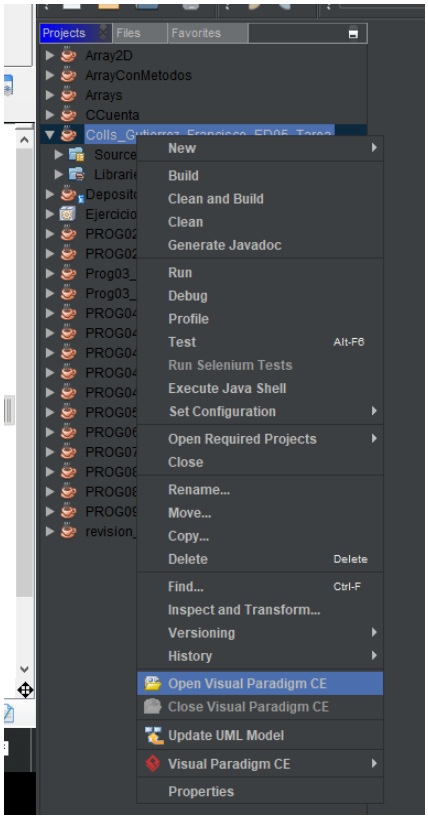


Francisco M. Colls Gutiérrez.

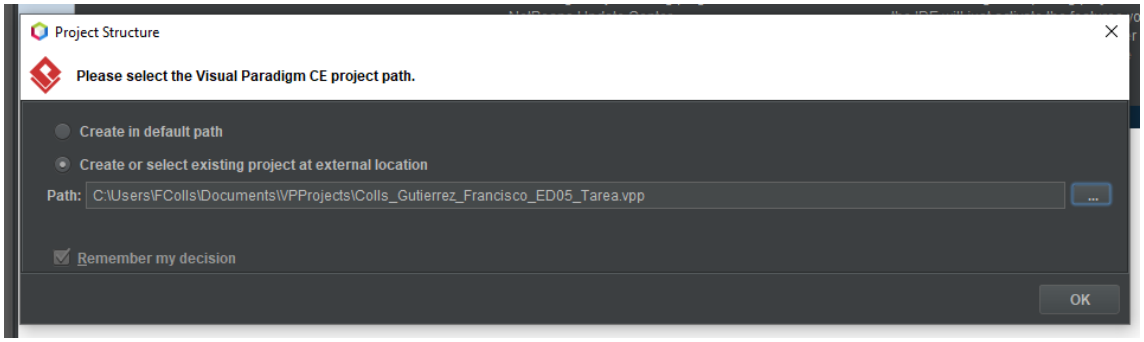
2.- Crear un proyecto en NetBeans.

3.- Desde NetBeans.

Hacemos clic derecho sobre el proyecto y seleccionamos Open Visual Paradigm CE.

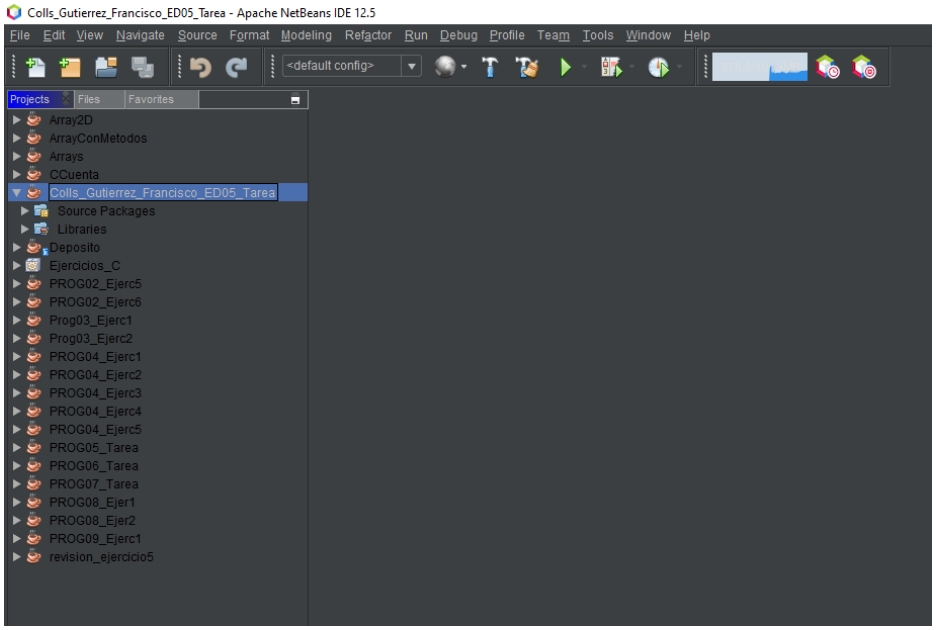


4.- Seleccionamos el proyecto que creamos anteriormente.



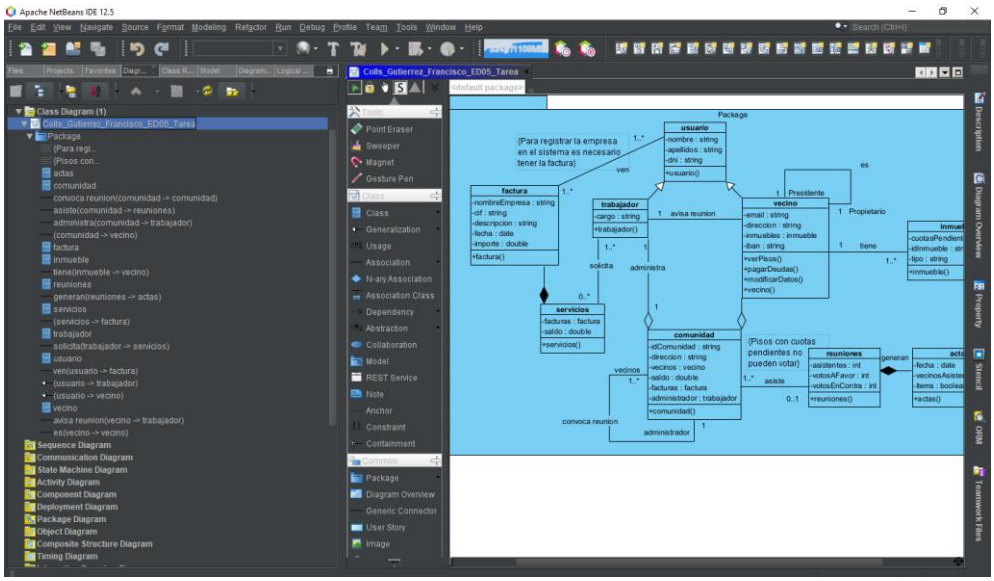
Francisco M. Colls Gutiérrez.

Antes de cargar el proyecto.

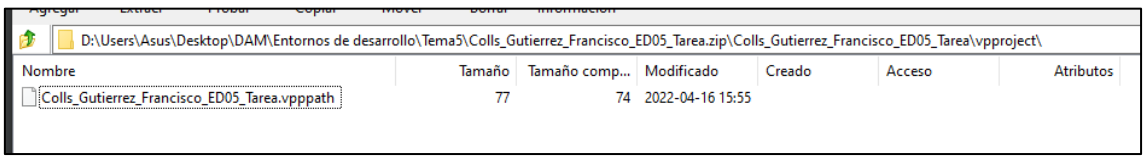


Luego de cargar el proyecto.

Aparecen nuevas opciones y podemos visualizar el diagrama y todas las clases, atributos, métodos y relaciones creadas.



Se crea un archivo vpppath en la carpeta vpproject de nuestro proyecto en NetBeans.



Francisco M. Colls Gutiérrez.