TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TLAXIACO

FUNDAMENTOS EN INGENIERIA DE SOFTWARE

Practica3: Reporte de diagramas de clases

Presenta:

Brenda Primavera Santos Muñoz 16161436

Carrera:

Ingeniera en Sistemas Computacionales

Profesor:

ING. JOSE ALFREDO ROMAN CRUZ

Tlaxiaco, Oax., 12 de septiembre del 2021.

"Educación, Ciencia y Tecnología, Progresos día con día" ®

ÍNDICE

Contenido

LISTA DE FIGURAS

INTRODUCCION

Los diagramas de clases UML (Lenguaje Unificado de Modelado) son un leguaje grafico el cual se utiliza para construir y especificar una clase. De la misma manera esta puede ayudar a los programadores a modelar distintas clases basados en un sistema los cuales pueden tener distintos tipos de conectores (De Herencia, Polimorfismo, Agregación, composición etc.), de esta manera el programador tiene una idea más clara y concisa sobre cómo va a funcionar sus diagramas de clases y objetos una vez ya codificados. Los diagramas de clases UML pueden ser plasmados en diversos softwares que tengan como función modelar diagramas de clases, además de ser sencillas de hacer estas pueden traducirse a código fácilmente. Architect Enterprise es un software especializado en crear diagramas UML, es fácil de usar además de contar con distintas herramientas las cuales le ayudaran al programador a diseñar los diagramas con mayor facilidad. Los diagramas de clases están compuestos por tres secciones nombre de la clase, atributos y métodos, estas son fáciles de comprender gracias al orden que lleva, dentro de los atributos encontraremos los tipos de datos y el estado que el atributo aparece en la sección de los métodos encontraremos los métodos parámetros y variables a utilizar para realizar alguna acción que sea necesaria, todo con el objetivo de que el diagrama este esquematizado como lo sería una línea de código.

OBJETIVO:

El objetivo de la práctica es aprender a realizar UML (Lenguaje Unificado de Modelado) a través del software Architect, llevando a la práctica al mismo tiempo los conceptos de agregación, composición, especialización y generalización haciendo énfasis a lo ya visto en clase.

DESCRICION DE LA PRACTICA.

La práctica está compuesta de diagramas de clases UML, las cuales fueron modeladas a computadora en el programa architect tomando como referencia a distintas clases basadas en objetos de la vida real. La práctica consta con conceptos fundamentales en los diagramas UML, como lo son la generalización, especialización, composición y agregación, esto con la intención de poner en práctica los conceptos ya dichos.

MATERIAL

computadora

Software: Architect Enterprise.

Diagramas de Clases hechas en la libreta.

PROCEDIMIENTO EN ARCHITECT

Abrimos el software: Architect Enterprise, después de que el software haya abierto nos vamos en la pestaña con el logo del software. Seleccionamos la primera opción para crear un nuevo proyecto.

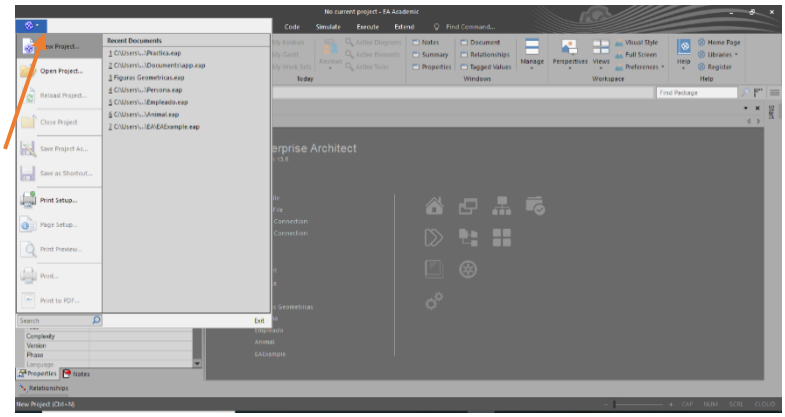
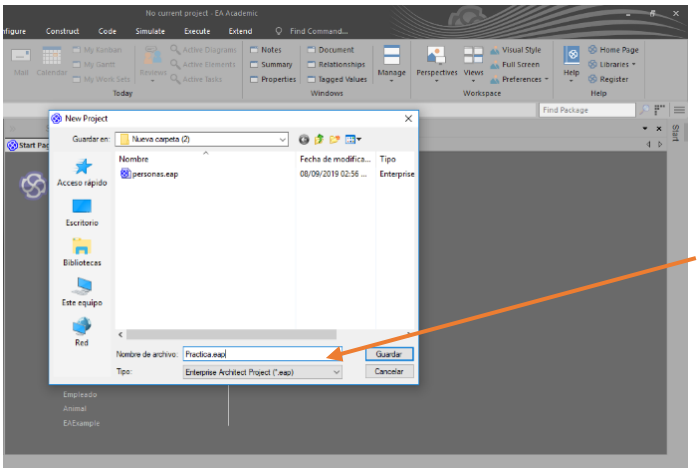


FIGURA 1 Software architect



Le otorgamos un nombre al archivo con el que se trabajara después lo guardamos como se muestra en la figura 2

FIGURA 2 nombre proyecto

Seleccionamos el modelo con el que se trabajara (en este caso es una clase por ende seleccionamos la casilla con la opción “class”).

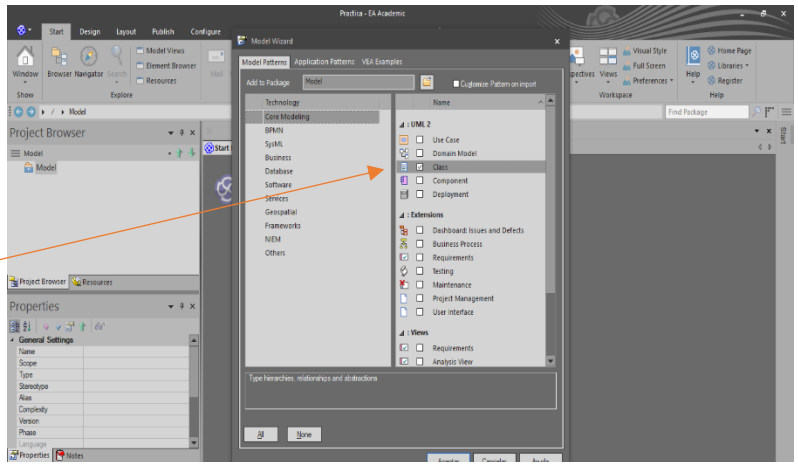


FIGURA 3S eleccion del modelo "class"

Nos dirigimos a la barra plegable del lado izquierdo con la leyenda Project browser, seleccionamos el icono de la capeta “Model” y despegamos su contenido. Seleccionamos la carpeta System y arrastramos el diagrama llamado de la misma manera “System”.

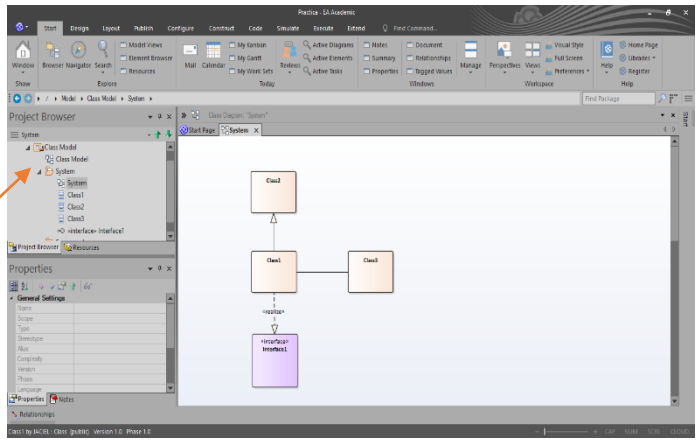
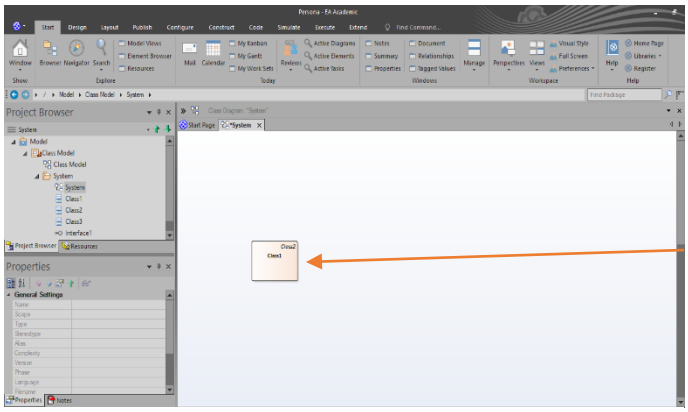


FIGURA 4 ejemplo la acción a realizar



Suprimimos las clases que no se utilizaran dentro de la hoja, después damos doble clic en la clase para modificarlo.

FIGURA 5 selección de clase

Modificamos el nombre de la clase (en este caso se llamará transporte).

Después nos deslizamos en la parte inferior hacia la opción detalle(details).

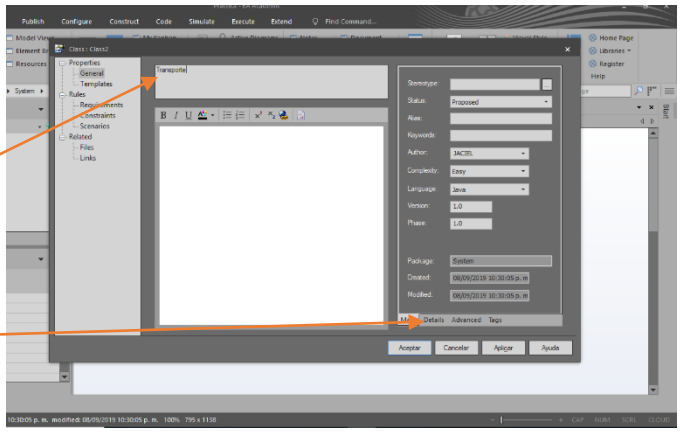


FIGURA 6 nombre de la clase

damos clic en el botón atributos (Attributes).

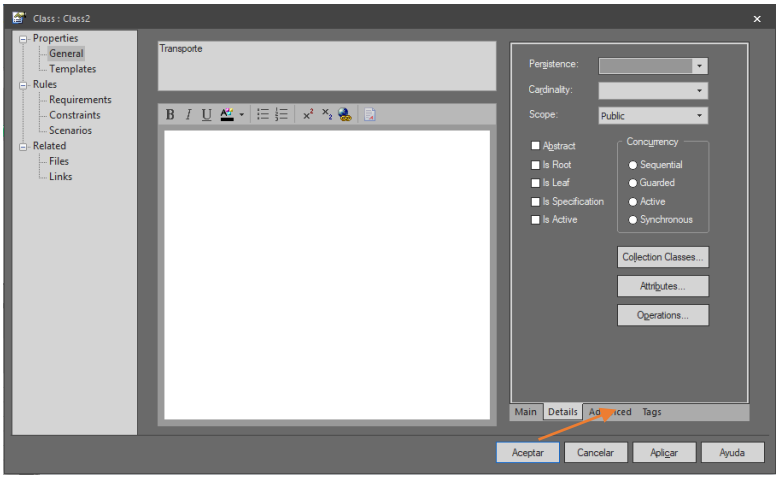


FIGURA 7 selección del botón para modificar los atributos

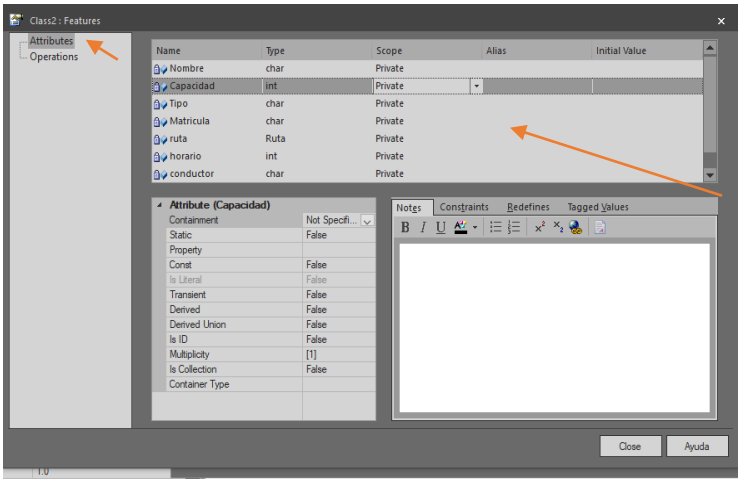


FIGURA 8 ingreso de atributos

Nos desplazamos en la pestaña “attributes” e ingresamos los atributos de la clase, después modificamos el tipo de dato de cada atributo, así como el tipo de visibilidad de cada uno.

Nos deslizamos en la pestaña de operaciones “Operations”, en ella ingresamos los métodos a utilizar.

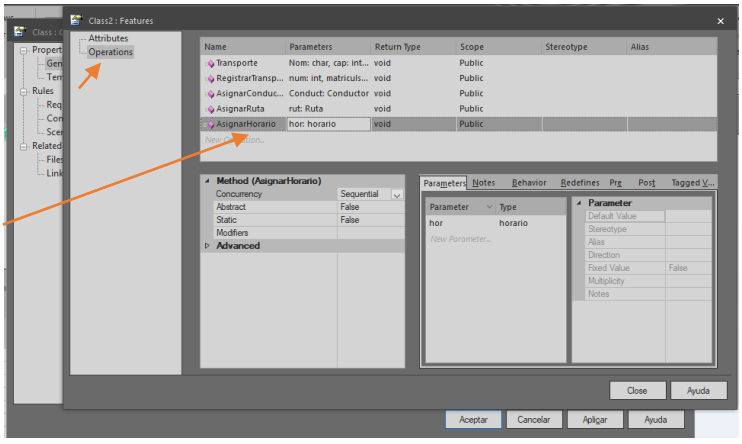


FIGURA 9 Diagrama por default

Modificamos los parámetros de cada método, así como el tipo de dato a ocupar, ya concluido el paso cerramos la ventana.

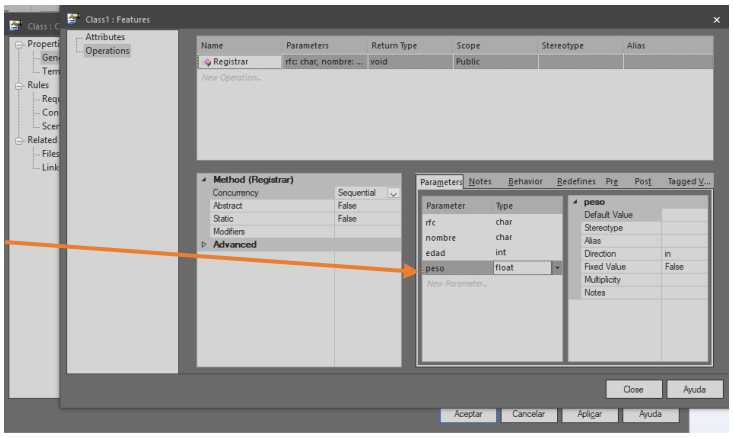


FIGURA 10 modificación de parámetros

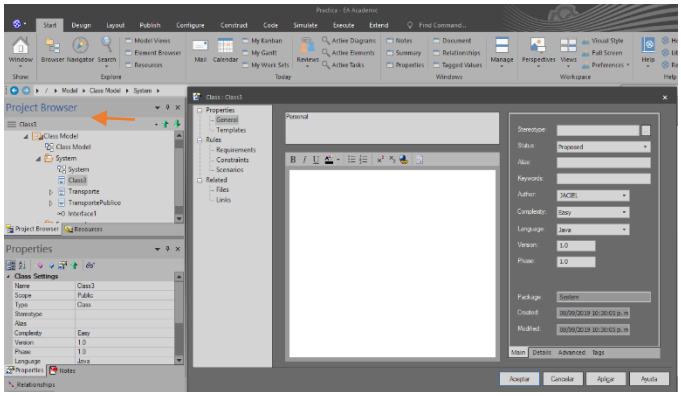


FIGURA 11 modificación de la segunda clase

Nos desplazamos hacia la barra de herramientas de la parte derecha y seleccionamos la segunda clase “class3” y modificamos el nombre.

Seguimos el mismo procedimiento para crear las otras clases agregando los atributos y operaciones que estas realizaran.

Nota: recordemos que en las operaciones se tiene que utilizar el pase por parámetro por valor o por referencia.

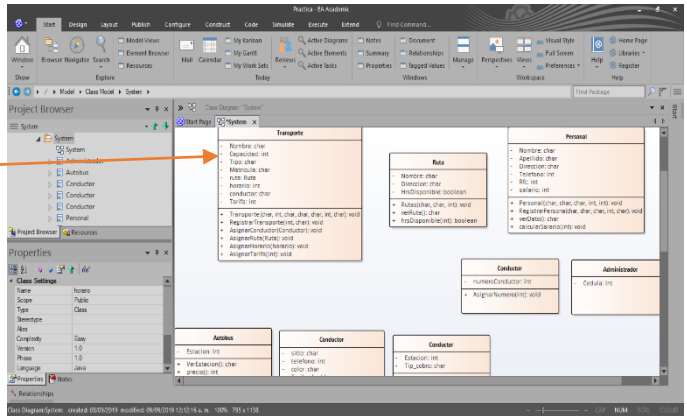
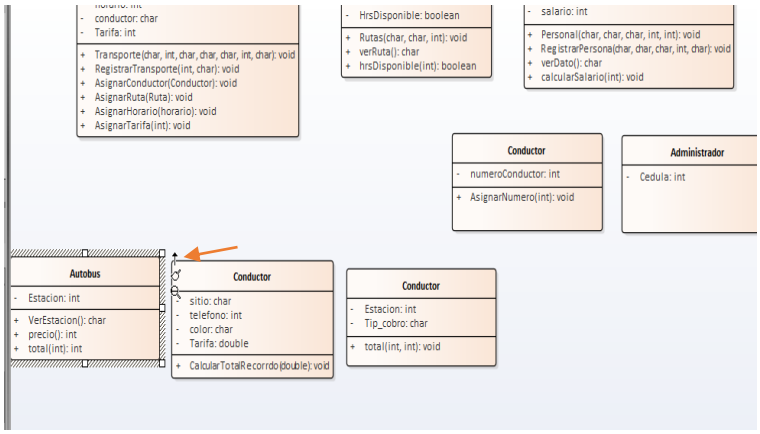


FIGURA 12 creación de las clases a utilizar



Nos deslizamos en la parte superior derecha del diagrama de clase y arrastramos la flecha de color negra hacia la clase a relacionar.

FIGURA 13 utilización de conector

Nos saldrá una ventana pagable en la cual seleccionaremos la opción (especialización, generalización, agregación, composición etc) según le corresponda.

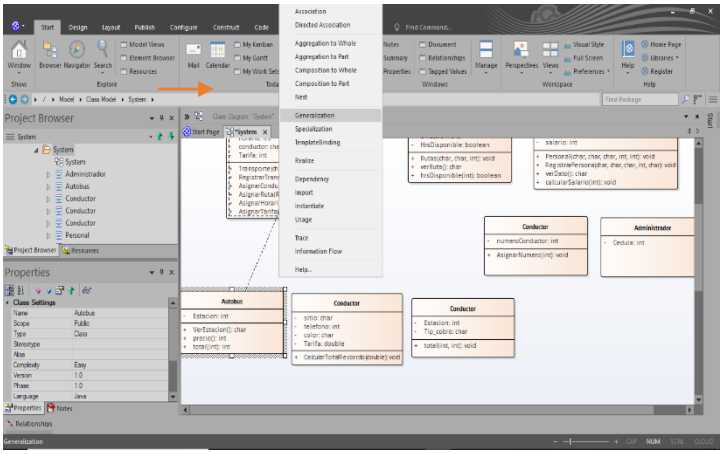
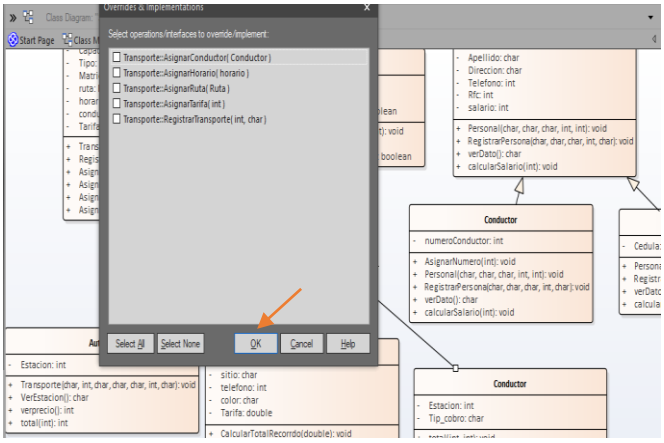
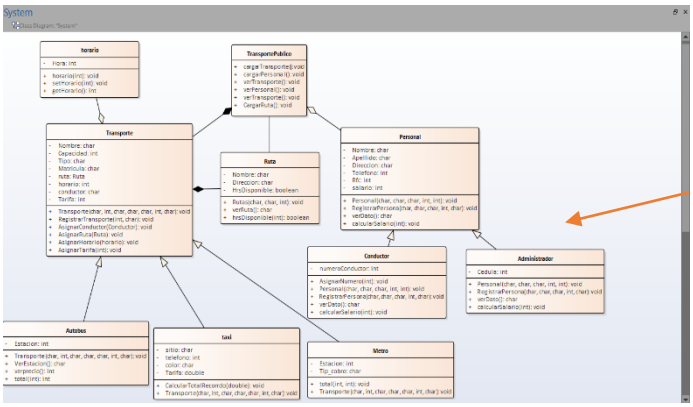


FIGURA 14 selección del tipo de conector



En seguida nos saldrá una ventana en la cual nos pedirá que marquemos los métodos que se deseen implementar, seleccionamos los métodos a implementar en nuestra clase.

FIGURA 16 selección de los métodos a implementar



El diagrama de clases nos quedara como se muestra en la figura. Esta debe utilizar relaciones (especialización, generalización, agregación, composición etc).

FIGURA 17 muestra del diagrama de clases

Ahora nos dirigiremos a la pestaña código (code) y seleccionaremos la opción generar todo (generate all).

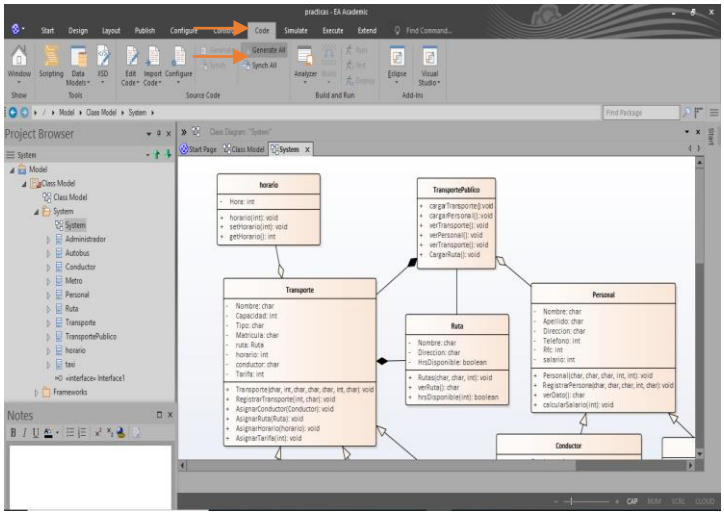
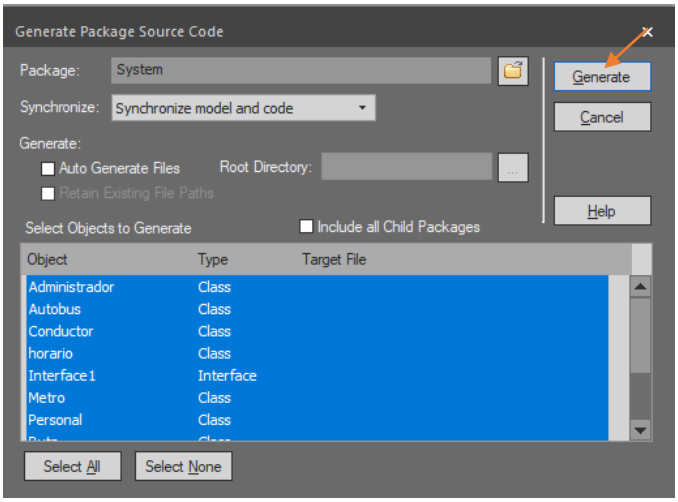
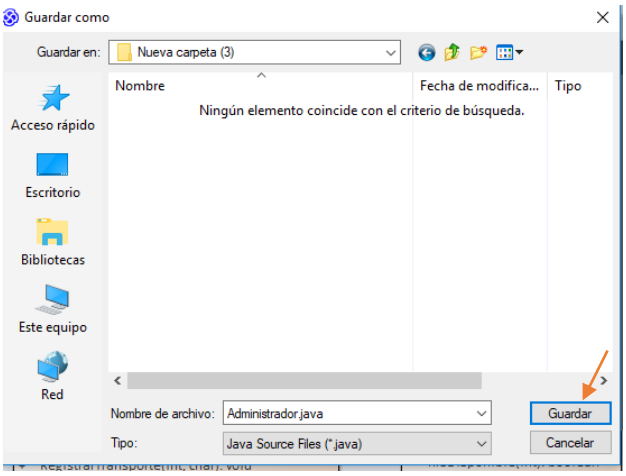


FIGURA 18 pestaña generar código



En seguida nos saldrá una ventana en el cual seleccionamos las clases a generar, así como el tipo de sincronización y el directorio. Una vez seleccionados las clases a generar damos clic en generar (generete)

FIGURA 19 selección de las clases a generar



Nos preguntara en donde deseamos que se guarden los archivos a generar. Seleccionamos la carpeta donde se almacenará nuestro código y damos clic en guardar.

FIGURA 20 ventana donde seleccionaremos la ruta a guardar

Una vez generado todas las clases nos saldrá la siguiente ventana. Damos clic en salir (close).

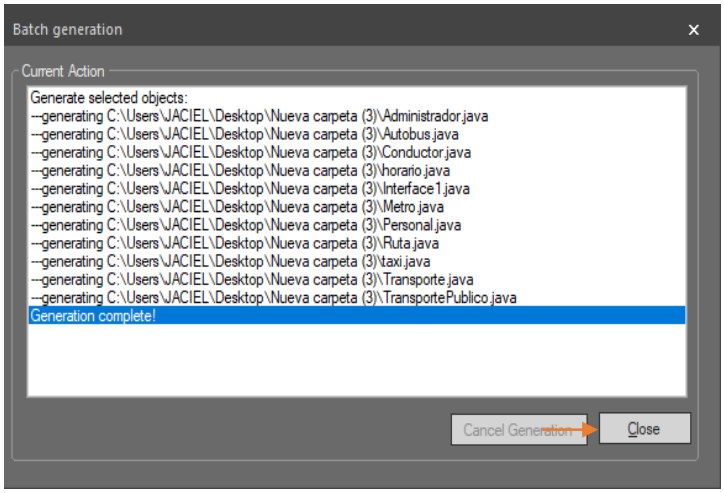
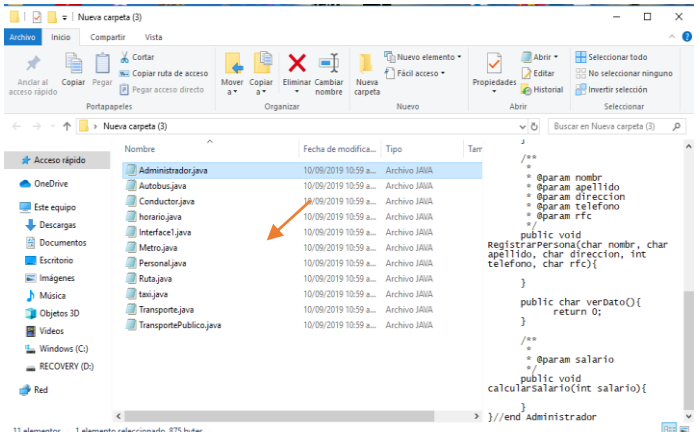
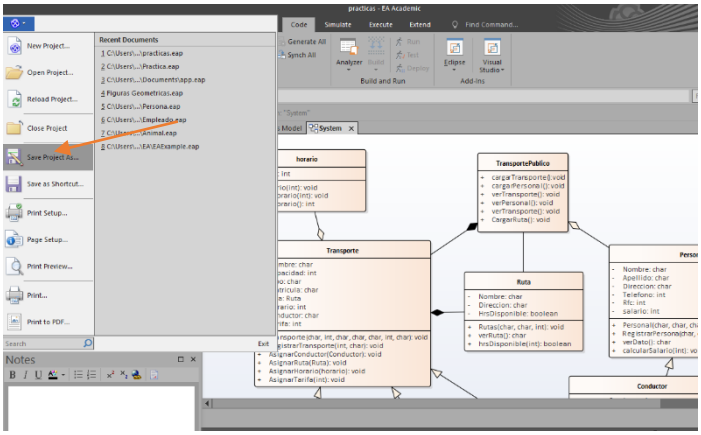


FIGURA 21 finalización de la generación de archivos



No dirigimos a la ruta para colaborar que se hayan generado las clases. Nos podemos dar cuenta que los archivos .java si fueron generados.

FIGURA 22 muestra de archivos creados



Regresamos al software y por guardamos nuestro proyecto para no perderlo.

FIGURA 23 guardando el archivo

RESULTADOS

Como resultado final nos quedan diagramas de clase UML las cuales están unidas mediante conectores, los cuales demuestran que al referirse a generalización las subclases autobús, taxis y metro heredan de la clase transporte al mismo tiempo estos utilizan el polimorfismo al utilizar el mismo método con diversas acciones en las tres clases.

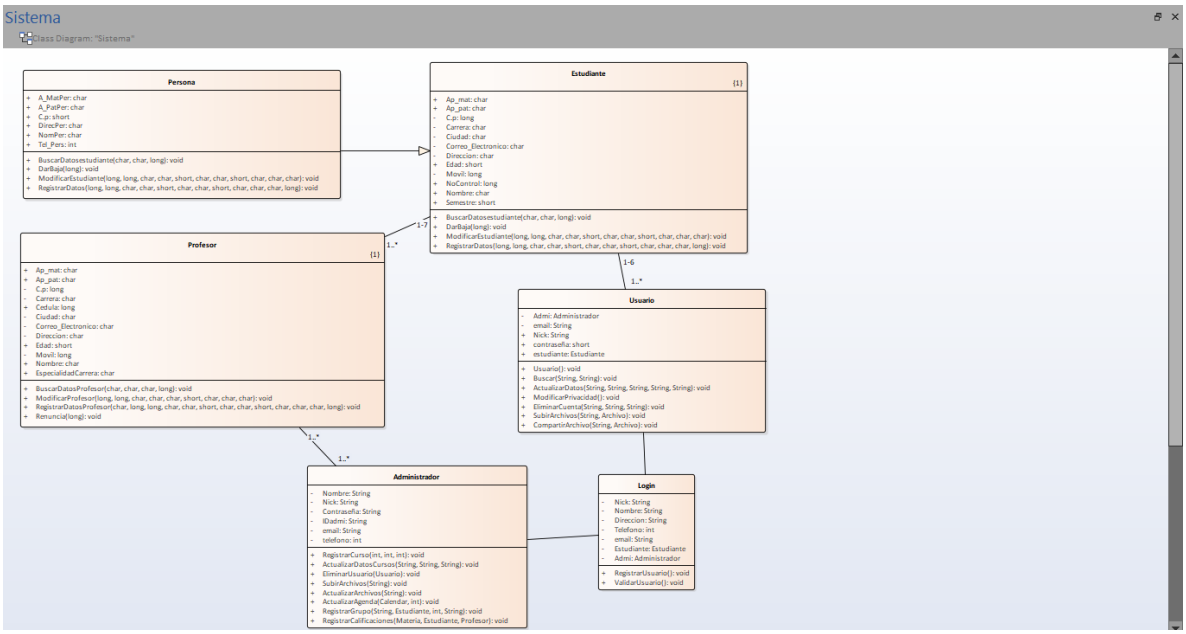


FIGURA 24 resultado

CONCLUSION

Los diagramas de clases UML nos ayudan a modelar clases antes de codificarlas, por medio de este software llamado Architect Enterprise podemos modelar diagramas de clases con una mayor facilidad y menos tiempo además de permitirnos observar cómo se comportaría nuestra clase al utilizar la Herencia y el Polimorfismo. Como se pudo observar en la práctica del ejercicio los diagramas de clases UML consta de un cuerpo de tres partes distintas, comenzando por un encabezado el cual lleva el nombre correspondiente a la clase, después en la segunda sección del cuerpo se le agregaran los atributos correspondientes a la clase y en la última sección se le agregaran los métodos que la clase utilizará al momento de ser codificada, cada parte del cuerpo es vital para la organización de nuestro diagrama de clase, además de esta manera se le da un mejor entendimiento a lo que se desea hacer en cada clase. Cada clase se agrupan en módulos los cuales nos ayudan a comprender mejor el funcionamiento de nuestras clases en nuestro diagrama. Como se vio en la práctica, los diagramas de clases UML son de gran utilidad para los programadores ya que permiten organizar mejor las ideas y también se le da un mejor entendimiento al cliente de lo que se desea hacer.

BIBLOGRAFIAS

Hanoi Torres (2005). Conceptos sobre funciones con parámetros por referencia. Fundamentos de programación. Universidad de valencia, Valencia, España. Recuperado de: http://informatica.uv.es/iiguia/FP/pract_fp_n6_04.pdf

UNID (2019). Funciones con pase por parámetro. Programación Estructurada. Universidad Interamericana para el desarrollo. Ciudad de México, México. Recuperado de:

https://moodle2.unid.edu.mx/dts_cursos_mdl/lic/TI/PE/S06/PE06_Lectura.pdf

UNIROJA (2010). Relación entre clases. Herencia entre clases. Universidad de la rioja, España. Recuperado de:

ttps://www.unirioja.es/cu/jearansa/0910/archivos/EIPR_Tema02.pdf

Elena Mediavilla (2010). II.3 UML: Modelado estructural. Programación orientada a objetos. Recuperado de:

https://www.ctr.unican.es/asignaturas/MC_OO/Doc/M_Estructural.pdf