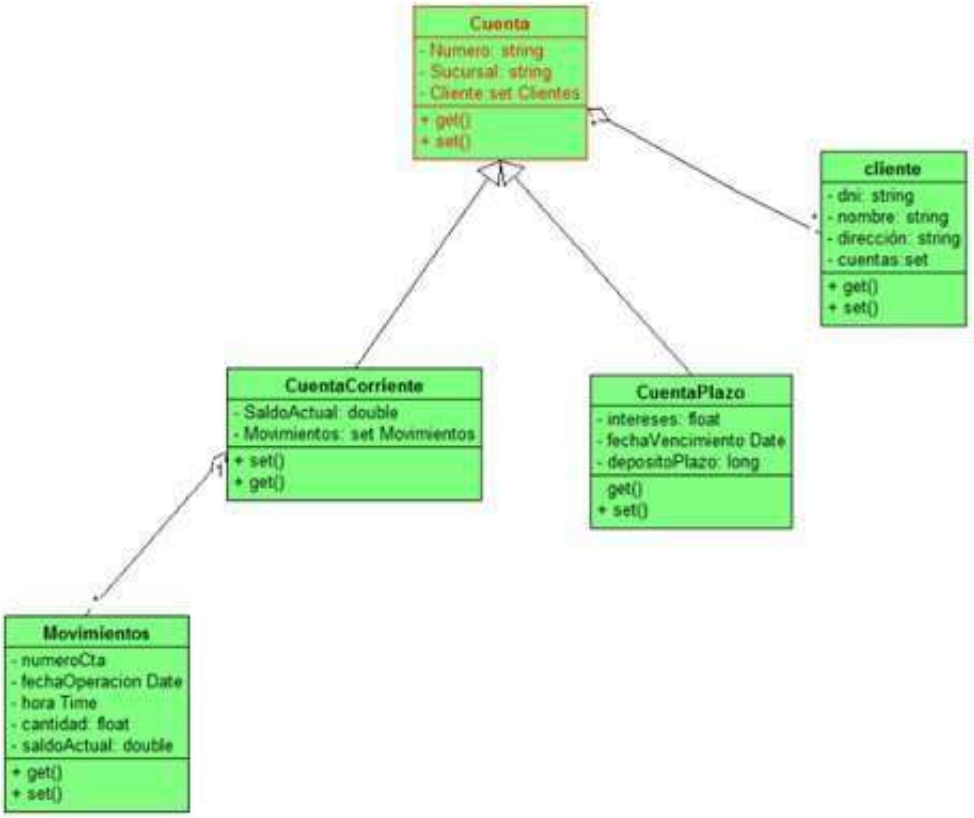
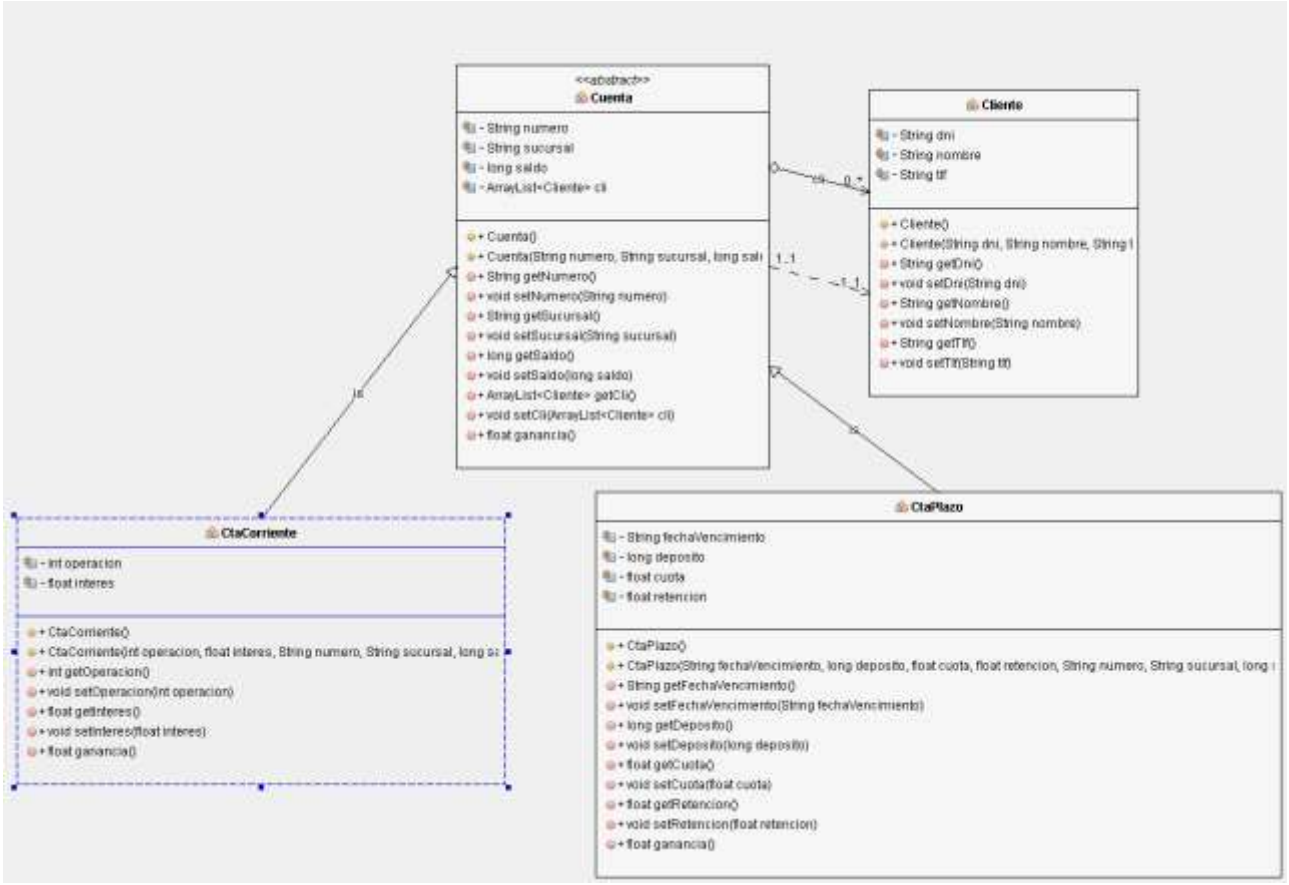
Partiendo del siguiente **UML que refleja herencia y agregación**. Realizar un programa en java, que:

- utilice **ficheros secuenciales,** usando la técnica de **seriación de objetos**,
- partimos de **cinco clases,** con relaciones entre ellas como **se indica en el UML,**



Como se ha indicado en las sesiones de clase, se puede prescindir de la Clase Movimientos y solamente hacer el proyecto con las otras cuatro. Reduciendo así las operaciones a gestionar.



La clase Cuenta, es la superclase y será el contenedor que usaremos para la aplicación, entonces crearemos un fichero secuencial de **tipo Cuenta**, con una relación de **herencia** con las clases cuentacorriente , cuentaPlazo y **agregación** con la clase Cliente.

La subclase CuentaCorriente, tiene a su vez una relación de agregación con la clase Movimientos, esta recoge tod**os los movimientos de ingreso ó reintegro que se han realizado en un plazo determinado .** (no se contempla si se ha prescindido de dicha clase)

MÉTODOS:

1-**Altas en fichero secuencial.** (mínimo exigible)

1.1- cuentasCorriente.
1.2- cuentasPlazo.
1.3-Clientes.

Para poder dar de alta una cuenta ,tendremos que:

- comprobar que el número de cuenta tenga una longitud de 6 caracteres
  - de los cuales los 5 primeros seán números del 0 al 9
  - y el último una letra de la A a la Z,
- una vez validado comprobaremos que no existe una cuenta con ese mismo número
- y a continuación añadimos el cliente ó clientes que solicitan dar de alta esta cuenta,
  - comprobaremos si este cliente ya existe en otras cuentas,

- si existiera cogeríamos la dirección del objeto y la colocamos en la nueva cuenta, para cuando tengamos que hacer las modificaciones del cliente ,al cambiar en una cuenta se propaga el cambio en las demás
- si por el contrario cada vez que creamos la cuenta creamos una nueva dirección del cliente aunque este ya exista en otras cuentas, tendríamos que cambiar los datos en todas las direcciones de ese cliente, solo así podremos crear el objeto cuenta.

2-**AltasMovimientos.** (apartado de ampliación)

Para poder dar de alta un movimiento de una cuenta corriente,

- tendremos que solicitar el número de cuenta, validarlo como en el caso anterior, y comprobar que existe ese número,
- a continuación comprobar que si es un reintegro,
  - la cantidad a retirar, no puede ser superior al saldo existente, en cuyo caso se emite el menasje:"SALDO INSUFICIENTE" y no se realizará la operación,
  - en el caso contrario, se actualizará el valor del saldo.

3-**Bajas físicas:** (interesante a incluir para tener funcionalidad mayor)

3.1-Cuenta corriente.
3.2-Cliente en una cuenta.

- Para poder dar de baja una cuenta ó un cliente, tenemos que:
  - comprobar que exista y a continuación darlo de baja(borrarlo) ,
  - en caso que no exista, emitir mensaje "No Existe".

4-**Modificaciones:**

4.1-Cliente.
4.1.1-Dirección.

Para poder modificar los datos de un cliente tendremos que:

- pedir el DNI del cliente para buscarlo y al encontrarlo cambiar el atributo dirección con el método set().
  - Antes de modificar tendremos que visualizar los atributos del cliente
  - en el caso de no existir el cliente emitir mensaje "No existe este CLIENTE".

5- **Consultas:**

5.1-Visualizar todos los clientes de una cuenta determinada, introduciendo el n.º de cuenta. (mínimo exigible)
5.2-Visualizar todas las cuentas de un cliente determinado, introduciendo el dni del cliente. (mínimo exigible)
5.3-Visualizar todos los movimientos de una determinada cuenta corriente entre dos fechas dadas.

NOTA: Para coger la fecha y la hora del sistema:

```
Date fecha=new Date(new java.util.Date().getTime());
    Time hora=new Time(fecha.getTime());
```

NOTA. Siempre se va a tener en memoria cargada la estructura con las cuentas y sólo se hará lectura o escritura al principio y final de la ejecución respectivamente.