* **A medida que se crea funcionalidad o crea conveniente, compruebe que todo funciona. Esta es una buena práctica para avanzar de forma adecuada.**
* **El Main deberá invocar únicamente a las funcionalidades que estén completamente desarrolladas.***(Buena Suerte, a codificar!)*

* El cliente nos envió el Diagrama UML que confeccionó junto con su personal de sistemas, el cual tomaremos como base para desarrollar nuestro pequeño sistema de gestión**.**



1) Generar la estructura y las clases indicadas en el diagrama con sus respectivas relaciones (Utilizar propiedades de acceso en cada clase).

2) Sobrecargar constructores, redefinir los métodos equals() y toString() en las clases.

3) Generar la estructura de la interface IAcciones

* **agregar(Object)** agrega un empleado o un tren, en caso del empleado validar que no exista previamente, de lo contrario, manejar el mensaje de error correspondiente.
* **buscar(int)** retorna el objeto correspondiente en base al parámetro recibido, en caso de no existir, debe retornar null.
* **quitar(int)** quita el objeto validando que exista, de lo contrario informar de error.

4) El Tren debe implementar los siguientes métodos:

* **calcularCargaTotal()** retorna la carga actual del tren.
* **validarRelacionCargaPotencia(float)** validar que la carga total actual no supere a la fuerza de carga de la Máquina.
* Capacidad máxima en kilos.
* Fuerza de Carga en toneladas.
* **agregarVagon(Vagon vagon)** agrega un vagón validando previamente Máquina tractora, permita soportar dicha carga, de lo contrario informar del error correspondiente.
* **calcularCargaDisponible()** retorna la capacidad restante disponible

5) Demostrar la funcionalidad en el **main** instanciando como mínimo:

* La clase compañíaFerroviaria
* 1 Jefe
* 1 Operario.
* 4 Vagones
* 2 MaquinaTractora.