

Autores: Álvaro Quesada Pimentel y Daniel Serrano Torres.

On the use of GUI and Threads in the Road Racing/Cycling Problem.

En este documento se explica en que consiste la práctica que ha realizado, su alcance, sus objetivos, las soluciones dadas, etcétera…

Tecnología de la programación:  
Práctica final.

Contenido

[1. Requisitos 3](#_Toc350415336)

[2. Objetivos 5](#_Toc350415337)

[3. Resumen 5](#_Toc350415347)

[4. Abstract 6](#_Toc350415348)

[5. Observaciones 6](#_Toc350415349)

[6. Sugerencias 6](#_Toc350415350)

[7. Introducción 6](#_Toc350415351)

[8. Objetivos que se alcanzarán 7](#_Toc350415352)

[9. Relación con la docencia cursada 8](#_Toc350415353)

[10. Viabilidad 8](#_Toc350415354)

[11. Estado del arte y fundamentación teórica 8](#_Toc350415355)

[12. Conclusión 11](#_Toc350415358)

[13. Bibliografía 12](#_Toc350415359)

**PRACTICA FINAL**

On the use of GUI and Threads in the Road Racing/Cycling Problem

**Fecha de entrega**: 14 de Junio de 2013

1. **Requisitos**

Para realizar esta práctica es preciso conocer el temario impartido en clase, comprendido en los temas primero y quinto.

Este documento se va a centrar en explicar las diferencias entre clases y objetos, la herencia y sus relaciones y tipos.

**¿Qué es una clase?**

Es un conjunto coherente que consiste en un tipo particular de metadatos (información sobre los datos, son datos que describen otros datos y nos permite obtenerlos).

Es un modelo que describe el estado y el comportamiento que tienen todos los objetos del mundo real.

Encapsula el estado y el comportamiento del concepto que representa, encapsula los atributos y comportamientos.

Una clase es una plantilla o un prototipo para crear objetos, por eso se dice que los objetos son instancias de clases.

**¿Qué es un objeto?**

Es el encapsulamiento de un conjunto de operaciones (métodos) que pueden ser invocados externamente, y de un estado que recuerda el efecto de los servicios.

Propiedades de un objeto, el tiempo de vida, estado, y el comportamiento definido por los métodos.

**Herencia**

En orientación a objetos la herencia es, después de la agregación o composición, el mecanismo más utilizado para alcanzar objetivos como lo son la reutilización y la extensibilidad. A través de ella los diseñadores pueden crear nuevas clases partiendo de una clase o de una jerarquía de clases preexistente evitando con ello el rediseño, la modificación y verificación de la parte ya implementada. La herencia facilita la creación de objetos a partir de otros ya existentes e implica que una subclase obtiene todo el comportamiento y eventualmente los atributos de su superclase.

**Tipos de herencia**

En el caso del lenguaje de programación escogido para esta práctica y utilizado en esta cátedra existen dos tipos de herencia.

El primer tipo de herencia es la herencia de estructura de datos y comportamiento, donde una clase hija obtiene todo lo que su padre le proporciona, tanto datos como comportamiento.

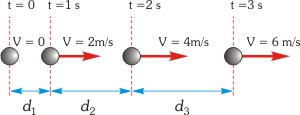
El segundo tipo de herencia, es la herencia de comportamiento o de interfaz, la cual se aplica con interfaces, esta herencia permite definir el comportamiento que se va a heredar por los elementos hijos.

En java la herencia múltiple solo se puede realizar con la herencia de interfaz o de comportamiento.

También será necesario tener un conocimiento básico de física, relacionado con el movimiento rectilíneo uniformemente variado. Se denomina así a aquel movimiento rectilíneo que se caracteriza porque su aceleración permanece constante en el tiempo (en módulo y dirección).

En este tipo de movimiento el valor de la velocidad aumenta o disminuye uniformemente al transcurrir el tiempo, esto quiere decir que los cambios de velocidad son proporcionales al tiempo transcurrido, o, lo que es equivalente, en tiempos iguales la velocidad del móvil aumenta o disminuye en una misma cantidad.

Veamos un ejemplo:



En este caso tenemos un móvil que se mueve horizontalmente describiendo un MRUV en donde en cada segundo el valor de su velocidad aumenta en 2 m/s. Debido a esto, el valor de la aceleración constante con que se mueve el móvil es 2 metros por segundo cuadrado:

**a = 2 m/s2**

1. **Objetivos**
   1. Generales

El objetivo de la práctica es trabajar con la herencia, una característica de la programación orientada a objetos. Tratando de crear un sistema que represente un ciclista. Mostrándose esta en una vista con interfaz gráfica.

Además se pide la incorporación de un parser que traduzca unos comandos introducidos por teclado o fichero a unos órdenes que pueda entender el ciclista.

También se introducirá una carretera por la que ira la bici, esta carretera estará dividida en tramos, y estos tramos tendrán una pendiente, positiva o negativa, y un viento que puede ser a favor o en contra.

Para ello se recurrirá a los conocimientos sobre polimorfismo, sobrecarga, sobrescrita, vinculación estática y dinámica. Se utilizarán en la medida que sean necesarios.

Todo ello tendrá una representación gráfica mediante swing, con una interfaz amigable para los usuarios inexpertos y un sistema de comandos que le permitan añadir más instrucciones a usuarios más avanzados.

* 1. Particulares
* Estudiar la herencia simple y múltiple en la programación orientada a objetos, y la herencia de interfaz.
* Construir un manejador del tiempo como contenedor de números. (horas, minutos y segundos).
* Construir un paseador de ordenes sencillo que permita la interacción del

Usuario con el ciclista.

* Construir un mapa de carretera en el que se especifique la pendiente y el viento de los tramos.
* Recordar conceptos de física básicos (movimiento rectilíneo uniformemente variado).
* Utilizar eclipse como entorno de desarrollo y jUnit (librería de java para realizar pruebas sobre el software).

1. **Resumen**

Se ha realizado una simulación de un ciclista y su bicicleta con el fin de incluirlo en posteriores ampliaciones. Se ha considerado que el ciclista es una persona y que existen numerosas "apariencias" o "vistas" de bicicleta. El tiempo se ha acumulado en un reloj y se ha tratado para permitir nuestro interés. La salida de datos se ha considerado parte fundamental del problema para poder apreciar el funcionamiento de la simulación. Se ha hecho una validación y verificación exhaustiva.

El ciclista va a recibir una serie de órdenes. Estas órdenes indicaran al ciclista lo que debe hacer, por ejemplo frenar, acelerar, aumentar plato, etc. Se transmitirán al ciclista por medio de teclado o de un fichero.

También se le incorporara al ciclista una carrera por la que ira circulando, esta carretera estará dividida en tramos, y estos tramos tendrán la pendiente y el viento que hay en ese punto.

1. **Abstract**

There has been a simulation of a cyclist and his bike in order to include it in subsequent extensions. It has been considered that the rider is a person and that there are many "appearances" or "views" of cycling. The time is accumulated in a clock and has tried to allow our interest. The data output is considered part of the problem in order to assess the performance of the simulation. It has made a thorough validation and verification.

The cyclist is going to receive a series of orders. These orders will tell the cyclist what it has to do, for example slowing down, accelerate, etc. the orders will be transmitted to the cyclist by keyboard or files input.

Also a track has been added in which the cyclist will be running. The track is divided in tracts, and these tracts have slope and wind information on every point of the track.

1. **Observaciones**

El proceso de diseñar el sistema de la salida de datos has sido ligeramente costoso, ya que la manera correcta de hacerlo es una representación antinatural del “mundo real” el cual tratamos de representar en nuestro modelo de software.

La creación del parser ha sido bastante dura, puesto que el paso de comandos a órdenes puede llegar a ser un proceso costoso y de difícil aprendizaje.

1. **Sugerencias**

Definir los objetivos de las prácticas más exhaustivamente y proporcionar el material utilizado en las sesiones.

1. **Introducción**

La bicicleta, a menudo llamada una bici es un [vehículo](http://es.wikipedia.org/wiki/Veh%C3%ADculo) de [transporte](http://es.wikipedia.org/wiki/Transporte) personal de [propulsión humana](http://es.wikipedia.org/wiki/Veh%C3%ADculos_de_tracci%C3%B3n_humana), es decir por el propio viajero. Sus componentes básicos son dos [ruedas](http://es.wikipedia.org/wiki/Rueda_de_bicicleta), generalmente de igual diámetro y dispuestas en línea, un [sistema de transmisión](http://es.wikipedia.org/wiki/Transmisi%C3%B3n_de_bicicleta) a [pedales](http://es.wikipedia.org/wiki/Pedal_de_bicicleta), un [cuadro](http://es.wikipedia.org/wiki/Cuadro_de_bicicleta) metálico que le da la estructura e integra los componentes, un [manillar](http://es.wikipedia.org/wiki/Manillar) para controlar la dirección y un [sillín](http://es.wikipedia.org/wiki/Sill%C3%ADn) para sentarse. El desplazamiento se obtiene al girar con las piernas la caja de los [pedales](http://es.wikipedia.org/wiki/Pedal_de_bicicleta) que a través de una [cadena](http://es.wikipedia.org/wiki/Cadena_de_transmisi%C3%B3n) hace girar un [piñón](http://es.wikipedia.org/wiki/Pi%C3%B1%C3%B3n_(mecanismo)) que a su vez hace girar la rueda trasera sobre el pavimento. El diseño y configuración básica de la bicicleta ha cambiado poco desde el primer modelo de transmisión de cadena desarrollado alrededor de [1885](http://es.wikipedia.org/wiki/1885).[2](http://es.wikipedia.org/wiki/Bicicleta#cite_note-2)

La paternidad de la bicicleta se le atribuye al barón [Karl von Drais](http://es.wikipedia.org/wiki/Karl_Drais), un inventor [alemán](http://es.wikipedia.org/wiki/Alemania) que nació en [1785](http://es.wikipedia.org/wiki/1785). Su rudimentario artefacto, creado alrededor de [1817](http://es.wikipedia.org/wiki/1817), se impulsaba apoyando los pies alternativamente sobre el suelo.[3](http://es.wikipedia.org/wiki/Bicicleta#cite_note-3) En la actualidad hay alrededor de 800 millones de bicicletas en el mundo (la mayor parte de ellas en China), bien como medio de transporte principal o bien como vehículo de ocio.



(Bicicleta diseñada para el transporte urbano)

Es un [medio de transporte](http://es.wikipedia.org/wiki/Medio_de_transporte) [sano](http://es.wikipedia.org/wiki/Salud), [ecológico](http://es.wikipedia.org/wiki/Huella_ecol%C3%B3gica), [sostenible](http://es.wikipedia.org/wiki/Desarrollo_sostenible) y muy [económico](http://es.wikipedia.org/wiki/Ahorro), tanto para trasladarse por [ciudad](http://es.wikipedia.org/wiki/Ciudad) como por zonas [rurales](http://es.wikipedia.org/wiki/Medio_rural). Su uso está generalizado en casi toda [Europa](http://es.wikipedia.org/wiki/Europa), siendo en países como [Suiza](http://es.wikipedia.org/wiki/Suiza), [Alemania](http://es.wikipedia.org/wiki/Alemania), [Países Bajos](http://es.wikipedia.org/wiki/Pa%C3%ADses_Bajos), algunas zonas de [Polonia](http://es.wikipedia.org/wiki/Polonia) y los [países escandinavos](http://es.wikipedia.org/wiki/Escandinavia) uno de los principales medios de transporte. En [Asia](http://es.wikipedia.org/wiki/Asia), especialmente en [China](http://es.wikipedia.org/wiki/Rep%C3%BAblica_Popular_China) y la [India](http://es.wikipedia.org/wiki/India), es el principal medio de transporte.

Las bicicletas fueron muy populares en la década de 1890, en los años 50, 70, y ahora su uso nuevamente ha venido a crecer considerablemente en todo el mundo.

*(Bicicleta de carretera)*

 En la actualidad hay alrededor de 800 millones de bicicletas en el mundo (la mayor parte de ellas en China), bien como medio de transporte principal o bien como vehículo de ocio.

Es un [medio de transporte](http://es.wikipedia.org/wiki/Medio_de_transporte) [sano](http://es.wikipedia.org/wiki/Salud), [ecológico](http://es.wikipedia.org/wiki/Huella_ecol%C3%B3gica), [sostenible](http://es.wikipedia.org/wiki/Desarrollo_sostenible) y muy [económico](http://es.wikipedia.org/wiki/Ahorro), tanto para trasladarse por [ciudad](http://es.wikipedia.org/wiki/Ciudad) como por zonas [rurales](http://es.wikipedia.org/wiki/Medio_rural). Su uso está generalizado en casi toda [Europa](http://es.wikipedia.org/wiki/Europa), siendo en países como [Suiza](http://es.wikipedia.org/wiki/Suiza), [Alemania](http://es.wikipedia.org/wiki/Alemania), [Países Bajos](http://es.wikipedia.org/wiki/Pa%C3%ADses_Bajos), algunas zonas de [Polonia](http://es.wikipedia.org/wiki/Polonia) y los [países escandinavos](http://es.wikipedia.org/wiki/Escandinavia) uno de los principales medios de transporte. En [Asia](http://es.wikipedia.org/wiki/Asia), especialmente en [China](http://es.wikipedia.org/wiki/Rep%C3%BAblica_Popular_China) y la [India](http://es.wikipedia.org/wiki/India), es el principal medio de transporte.

Las bicicletas fueron muy populares en la década de 1890, en los años 50, 70, y ahora su uso nuevamente ha venido a crecer considerablemente en todo el mundo.

### http://upload.wikimedia.org/wikipedia/commons/thumb/7/78/Pierre_Lallemant_1870.jpg/175px-Pierre_Lallemant_1870.jpgHistoria

[Pierre Lallement](http://es.wikipedia.org/wiki/Pierre_Lallement), Montado en su invención

En el [Antiguo Egipto](http://es.wikipedia.org/wiki/Antiguo_Egipto) se fabricaron artefactos rudimentarios compuestos por dos ruedas unidas por una barra. También en China un artilugio muy similar, pero con ruedas hechas de bambú. Las primeras noticias que se tienen sobre una bicicleta datan del año [1490](http://es.wikipedia.org/wiki/1490), aproximadamente, en la obra [Codex Atlanticus](http://es.wikipedia.org/wiki/Codex_Atlanticus), de [Leonardo da Vinci](http://es.wikipedia.org/wiki/Leonardo_da_Vinci). En ellos puede verse un boceto de una bicicleta con transmisión de cadena impulsada por unos pedales, mismo método empleado por las actuales.

Se dijo que en [1790](http://es.wikipedia.org/wiki/1790) el [conde](http://es.wikipedia.org/wiki/Conde) [francés](http://es.wikipedia.org/wiki/Francia) [Mede de Sivrac](http://es.wikipedia.org/w/index.php?title=Mede_de_Sivrac&action=edit&redlink=1) había inventado en [París](http://es.wikipedia.org/wiki/Par%C3%ADs) el «celerífero», al que también llamaban «caballo de ruedas». Este consistía en un listón de madera, terminado en una cabeza de león, de dragón o de ciervo, y montado sobre dos ruedas. No tenía articulación alguna, y para las maniobras había que echar pie a tierra; esa misma rigidez hacía que todas las variaciones del terreno repercutieran sobre el cuerpo de su montura. Sin embargo, el conde Mede Sivrac, inventor de célérifère, nunca existió. El personaje fue creado en 1891 por el periodista francés, especialista en la locomoción terrestre, Louis Baudry de Saunier ( 1865 - 1938 ). Para él, era más gratificante realizar una copia de la invención de Karl Drais para 1790 y atribuirlo a un francés, en su Historia General el [velocípedo](http://es.wikipedia.org/wiki/Veloc%C3%ADpedo), que apareció en 1891.

### La draisiana



La draisiana (ca. 1820) era el primer vehículo de dos ruedas dispuestas en línea, y el primer vehículo práctico de propulsión humana.

En [1817](http://es.wikipedia.org/wiki/1817), el [barón](http://es.wikipedia.org/wiki/Bar%C3%B3n) [alemán](http://es.wikipedia.org/wiki/Alemania) [Karl Christian Ludwig Drais von Sauerbronn](http://es.wikipedia.org/wiki/Karl_Drais) inventó el primer vehículo de dos ruedas, al que llamó máquina andante (en alemán, laufmaschine), precursora de la bicicleta y la motocicleta. Esta «máquina andante» consistía en una especie de carrito de dos ruedas, colocadas una detrás de otra, y un manillar. La persona se mantenía sentada sobre una pequeña montura, colocada en el centro de un pequeño marco de madera. Para moverse, empujaba alternativamente con el pie izquierdo y el derecho hacia adelante, en forma parecida al movimiento de un patinador. Con este impulso, el vehículo adquiría una velocidad casi idéntica a la de un coche. Sus brazos descansaban sobre un apoyabrazos de hierro, y con las manos sostenía una vara de madera, unida a la rueda delantera, que giraba en la dirección hacia la cual quería ir el conductor.

Este invento estaba basado en la idea de que una persona, al caminar, desperdicia mucha fuerza por tener que desplazar su peso en forma alternada de un pie al otro. Drais logró crear este sencillo vehículo que le permitió al hombre evitar ese trabajo. Esta máquina, denominada inicialmente draisiana en honor a su inventor y posteriormente llamada más comúnmente [velocípedo](http://es.wikipedia.org/wiki/Veloc%C3%ADpedo), evolucionó rápidamente.

El herrero e inventor francés [Pierre Michaux](http://es.wikipedia.org/wiki/Pierre_Michaux) también es considerado uno de los desarrolladores de la bicicleta moderna.

### La bicicleta de pedales

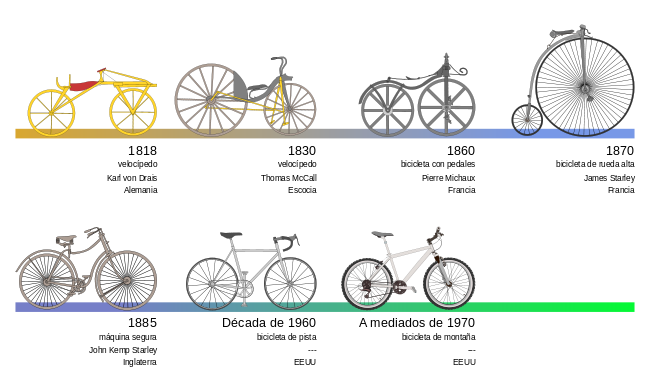


Como podemos observar en este modelo de los años 1900, el diseño del cuadro «diamante» se ha mantenido más de cien años.

La construcción de la primera bicicleta con pedales se atribuye al [escocés](http://es.wikipedia.org/wiki/Escocia) [Kirkpatrick Macmillan](http://es.wikipedia.org/w/index.php?title=Kirkpatrick_Macmillan&action=edit&redlink=1), en el año [1839](http://es.wikipedia.org/wiki/1839). Una copia de la bicicleta de Macmillan se exhibe en el Museo de Ciencias en [Londres](http://es.wikipedia.org/wiki/Londres), [Inglaterra](http://es.wikipedia.org/wiki/Inglaterra). Macmillan nunca patentó el invento, que posteriormente fue copiado en [1846](http://es.wikipedia.org/wiki/1846) por [Gavin Dalzell](http://es.wikipedia.org/w/index.php?title=Gavin_Dalzell&action=edit&redlink=1) de [Lesmahagow](http://es.wikipedia.org/w/index.php?title=Lesmahagow&action=edit&redlink=1), quien lo difundió tan ampliamente que fue considerado durante cincuenta años el inventor de la bicicleta.[4](http://es.wikipedia.org/wiki/Bicicleta#cite_note-4)

Cerca de [1890](http://es.wikipedia.org/wiki/1890), el inglés [John Boyd Dunlop](http://es.wikipedia.org/wiki/John_Boyd_Dunlop) (aficionado al ciclismo y creador de la empresa homónima) inventó una cámara de tela y caucho, que se inflaba con aire y se colocaba en la llanta. Para evitar pinchazos, Dunlop inventó además una cubierta también de caucho. Estos inventos de Dunlop casi no han sufrido variaciones significativas desde su invención.

**EVOLUCIÓN DE LA BICICLETA**

Evolución histórica de la bicicleta.

### Usos

Las bicicletas han tenido y han sido empleadas para muchos usos:

* **Utilidad**: [transporte](http://es.wikipedia.org/wiki/Transporte), [bicicleta de conmutaje](http://es.wikipedia.org/wiki/Bicicleta_de_conmutaje) y [bicicleta utilitaria](http://es.wikipedia.org/w/index.php?title=Bicleta_utilitaria&action=edit&redlink=1)
* **Trabajo**: [reparto de correo](http://es.wikipedia.org/wiki/Correo), [paramedicina](http://es.wikipedia.org/w/index.php?title=Paramedicina&action=edit&redlink=1), [policía](http://es.wikipedia.org/w/index.php?title=Bicicleta_de_polic%C3%ADa&action=edit&redlink=1), [mensajería](http://es.wikipedia.org/w/index.php?title=Mensajero_ciclista&action=edit&redlink=1), y [distribución](http://es.wikipedia.org/wiki/Distribuci%C3%B3n) general.
* **Recreativa**: [cicloturismo](http://es.wikipedia.org/wiki/Cicloturismo), [bicicleta de montaña](http://es.wikipedia.org/wiki/Bicicleta_de_monta%C3%B1a), [BMX](http://es.wikipedia.org/wiki/BMX), [fitness](http://es.wikipedia.org/wiki/Fitness), bicicleta de época y juego, aprendizaje y experimentación.
* [**Carreras**](http://es.wikipedia.org/wiki/Ciclismo): [ciclismo en pista](http://es.wikipedia.org/wiki/Ciclismo_en_pista), [criterium](http://es.wikipedia.org/w/index.php?title=Criterium&action=edit&redlink=1), [rodillos de bicicleta](http://es.wikipedia.org/w/index.php?title=Rodillos_de_bicicleta&action=edit&redlink=1) y [competición contrarreloj](http://es.wikipedia.org/wiki/Competici%C3%B3n_contrarreloj) de eventos multietapa (vueltas ciclistas) como el [Tour de California](http://es.wikipedia.org/wiki/Tour_de_California),[Giro d'Italia](http://es.wikipedia.org/wiki/Giro_d%27Italia), [Tour de France](http://es.wikipedia.org/wiki/Tour_de_France), la [Vuelta a España](http://es.wikipedia.org/wiki/Vuelta_a_Espa%C3%B1a), la [Volta a Portugal](http://es.wikipedia.org/w/index.php?title=Volta_a_Portugal&action=edit&redlink=1), y otros.
* **Militar**.
* **Espectáculo**: entretenimiento y actuación, ej. [ciclismo artístico](http://es.wikipedia.org/wiki/Ciclismo_art%C3%ADstico).

### Clasificación general de bicicletas

Bicicleta todo terreno aparcada en una calle de [Barcelona](http://es.wikipedia.org/wiki/Barcelona).

Existen diversas modalidades [deportivas](http://es.wikipedia.org/wiki/Deporte), englobadas dentro del [ciclismo](http://es.wikipedia.org/wiki/Ciclismo), que se practican con este [vehículo](http://es.wikipedia.org/wiki/Veh%C3%ADculo). La principal clasificación de las bicicletas toma en cuenta la función para la que están diseñadas, así los principales tipos de bicicletas son:

### La bicicleta doméstica

La bicicleta es el medio de [transporte](http://es.wikipedia.org/wiki/Transporte) personal preferido por muchas personas en todo el mundo. 800 millones de bicicletas son utilizadas diariamente en el mundo, la mayoría son bicicletas domésticas y [de paseo](http://es.wikipedia.org/wiki/Bicicleta_playera), también descritas como «bicicleta urbana» o City-bike (ver: [bicicleta híbrida](http://es.wikipedia.org/wiki/Bicicleta#Otras_bicicletas)), dedicadas a todo tipo de usos cotidianos donde debemos enfrentarnos a muchos trayectos pequeños que se pueden recorrer cuatro veces más rápidos que a pie. Destaca su énfasis en la comodidad a costa del peso, con [asiento](http://en.wikipedia.org/wiki/File:Gary_Fisher_Simple_City_3_set_I_.jpg) y [manubrio](http://en.wikipedia.org/wiki/File:Gary_Fisher_Simple_City_3_set_II.jpg) cómodos, sistemas de [transmisión](http://es.wikipedia.org/wiki/Cambios_internos) integrados en el propio [buje](http://es.wikipedia.org/wiki/Buje), o de [un solo cambio](http://es.wikipedia.org/wiki/Bicicleta_monomarcha), [guardabarros](http://en.wikipedia.org/wiki/File:Bicicleta_naranja_3.jpg), además de contar generalmente con una o más [canastillas](http://en.wikipedia.org/wiki/File:Pashley_wicker_basket.jpg) para el transporte de objetos. También es común que tengan accesorios urbanos como [timbre](http://en.wikipedia.org/wiki/File:Aspice_Christophorum....jpg), [candado](http://en.wikipedia.org/wiki/File:Bike_O_Lock_Japan.jpg), [luces](http://en.wikipedia.org/wiki/File:2009-11-28-fahrradmesse-by-RalfR-14.jpg) y [retrorreflectores](http://es.wikipedia.org/wiki/Retrorreflector#Retrorreflectores_como_elemento_de_seguridad).

Las bicicletas tradicionales también están en auge en Europa, siguiendo el arquetipo del diseño clásico aunque con componentes modernos y ligeros. Entre estos se encuentran los modelos de [Batavus](http://es.wikipedia.org/wiki/Batavus), [Gazelle](http://es.wikipedia.org/wiki/Bicicletas_Gazelle), [Kronan](http://es.wikipedia.org/wiki/Bicicletas_Kronan) y [Pashley](http://es.wikipedia.org/wiki/Pashley_Cycles)entre muchos otros, que abundan en ciudades como [Ámsterdam](http://es.wikipedia.org/wiki/%C3%81msterdam) y [Copenhague](http://es.wikipedia.org/wiki/Copenhague). Otro modelo que se encuentra en las grandes ciudades son las [plegables](http://es.wikipedia.org/wiki/Bicicleta#La_bicicleta_plegable) como la también clásica [Brompton](http://es.wikipedia.org/wiki/Bicicleta_Brompton), cuyos simpáticos y compactos diseños optimizan su utilización combinándola con el [transporte público](http://es.wikipedia.org/wiki/Transporte_p%C3%BAblico).

### La bicicleta de montaña

La bicicleta de montaña o bicicleta todo terreno (BTT) es una bicicleta destinada para el ámbito deportivo en terrenos agrestes, por lo que la resistencia de sus partes es un punto principal, también lo es la protección de sus partes al lodo y la tierra, también cuenta con varias relaciones de transmisión para adaptar el pedaleo a las condiciones del terreno. Deportes practicados son el [Cross-country](http://es.wikipedia.org/wiki/Cross-country), [Enduro](http://es.wikipedia.org/wiki/Enduro), [Freeride](http://es.wikipedia.org/wiki/Freeride) y [Descenso](http://es.wikipedia.org/wiki/Descenso_(DH)).

* [Bicicleta 29](http://es.wikipedia.org/wiki/Bicicleta_29): La bicicleta 29 o 29er (*Two-niners*, en inglés) son bicicletas de montaña que usan ruedas de 29 pulgadas.

### La bicicleta de carreras

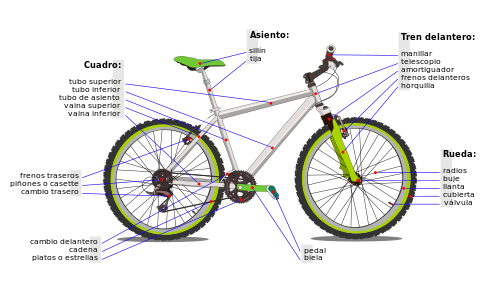
Las carreras de bicicletas profesionales son uno de los [deportes](http://es.wikipedia.org/wiki/Deporte) más duros del mundo. La bicicleta de carreras, comúnmente conocido como una «bicicleta de carretera», está diseñada para la velocidad, una batalla corta, ángulos de asiento y frontales muy verticales, un eje de pedalear alto, y muy poca curvatura en de la [horquilla](http://es.wikipedia.org/wiki/Horquilla) y donde la ligereza es importante, así mismo el manubrio tiene diseños particulares según tipo de competición para que el ciclista adopte posiciones aerodinámicas.

### Tipos de bicicletas de carrera:

* Bicicleta para [carretera](http://es.wikipedia.org/wiki/Ciclismo_en_ruta)
* Bicicleta para [ciclocrós](http://es.wikipedia.org/wiki/Ciclocr%C3%B3s)
* [Bicicleta para pista](http://es.wikipedia.org/wiki/Bicicleta_de_pista)
* Bicicleta para el [Tour de Francia](http://es.wikipedia.org/wiki/Tour_de_Francia)
* Bicicleta para [triatlón](http://es.wikipedia.org/wiki/Triatl%C3%B3n)
* Bicicleta para [Criterium](http://es.wikipedia.org/wiki/Crit%C3%A9rium_Internacional)

### Anatomía de la bicicleta

Existen diferentes tipos de bicicletas, pero básicamente todas son similares, aunque los componentes difieran en calidad, diseño y peso, así como en la agilidad y modalidad de uso. En orden de importancia, una bicicleta está formada por los siguientes componentes:



* **Cuadro:** El más común, es en forma de rombo, también llamado de diamante o de doble triángulo. Los clásicos eran de hierro o acero; hoy en día, cuando es [acero](http://es.wikipedia.org/wiki/Acero) el [cromo-molibdeno](http://es.wikipedia.org/w/index.php?title=Cromo-molibdeno&action=edit&redlink=1) se denomina «Cro-Moly» o «Cromoly». También pueden ser de [aluminio](http://es.wikipedia.org/wiki/Aluminio) o de [titanio](http://es.wikipedia.org/wiki/Titanio), o incluso de fibra de carbono entre otros materiales.

* **Horquilla:** Pieza formada por el tubo de dirección que sujeta el [buje](http://es.wikipedia.org/w/index.php?title=Buje_de_Bicicleta&action=edit&redlink=1) de la rueda delantera; puede ser fija o con [suspensión](http://es.wikipedia.org/wiki/Suspensi%C3%B3n_de_bicicleta).
* **Ruedas:** Después del cuadro, las ruedas son el elemento de mayor importancia para el rendimiento de la bicicleta.
* **Neumático:** El neumático es parte de la rueda y son la combinación de una cubierta protectora, una cámara inflable y la llanta que le da rigidez y sirve de estructura al eje de rodadura de la bicicleta.
* **Transmisión:** Incluye los cambios de marcha externos tipo [desviadores](http://es.wikipedia.org/wiki/Desviador) (dérailleur) delanteros y traseros y [cambios internos](http://es.wikipedia.org/wiki/Cambios_internos) en el buje de la rueda trasera, ambos manejados por palancas de cambio.
* **Palanca de cambio:** Cambiadores de marchas incluyen cambiadores de puño y cambiadores de pulgar entre otros.
* **Frenos**: Incluye las palancas de freno y sistemas de frenos.
* **Potencia**: La potencia (o tija del manillar), en conjunto con la horquilla delantera, son los componentes de una bicicleta que proporcionan una interfaz entre sí con el tubo frontal del cuadro.
* **Manillar**: Los manillares varían entre una anchura de 52,5 a 60 cm (21 a 24 pulgadas), los anchos permiten un control a velocidades bajas mientras los estrechos son mejores para velocidades altas, los estrechos además son convenientes en la ciudad para escurrir entre los automóviles. Un tipo de manillar se denomina «cola de ballena». Se distingue de los demás en que carece de los extremos libres que caracterizan al manillar tradicional.
* **Sillín**: De los sillines existentes en el mercado, unos son delgados y ligeros para reducir el peso mientras otros modelos anatómicos están diseñados para el confort.
* **Tija de sillín:** Se denomina tija al tubo de soporte del sillín.

1. **Introducción a la práctica**

El estudio de la herencia en POO con el uso de la herramienta Java presenta restricciones que no aparecen en otros lenguajes como C++. Es el caso de la herencia múltiple, que en java se resuelve con la herencia de interface.

El hecho de que dichas restricciones o mecanismos tengan palabras clave en Java para estos conceptos indica que estas ideas fueron consideradas los suficientemente importantes como para proporcionar un soporte directo. La palabra clave **interface** lleva el concepto de abstracción un paso más allá. La palabra clave **abstract** permite crear uno o más métodos no definidos dentro de una clase: proporcionamos parte de la interfaz, pero sin proporcionar la implementación correspondiente. La implementación se proporciona de las clases que hereden de la clase actual. La palabra clave **interface** produce una clase completamente abstracta, que no proporciona ninguna implementación en absoluto. Las interfaces permiten al creador determinar los nombres de los métodos, las listas de argumentos y los tipos de retorno, pero sin especificar ningún cuerpo de ningún método. Una interfaz proporciona simplemente un comportamiento o forma, sin ninguna implementación.

El modelo elegido ha sido una aplicación que tiene una persona, que es un ciclista, y este a su vez monta una bicicleta, y lleva un "reloj" que mide su propio tiempo de forma habitual. Este modelo muestra todos los datos del modelo por la Salida de Datos.

Atendiendo al desacoplamiento completo. Cuando un método funciona con una clase en lugar de con una interfaz, estamos limitados a utilizar dicha clase o sus subclases. Si quisiéramos aplicar ese método a una clase que no se encontrara en esa jerarquía, no podíamos. Las interfaces relajan esta restricción considerablemente. Como resultado, permiten escribir código más reutilizable.

También se hace uso y adaptación de patrones, se pueden ver claramente en el sistema interno de control de tiempo, y en el sub-sistema de órdenes del sistema.

1. Introducción

La aplicación es capaz de dejar correr de manera secuencial todos los objetos durante el transcurso de un determinado tiempo de ejecución, mostrando su estado en cada momento mediante una vista completamente independiente, que para esta última práctica está realizada bajo swing.

Esto “simula” la ejecución de hilos controlados por nuestro sistema basándonos en un ROUND ROBIN, donde hay un orden específico de ejecución de estos “hilos”.

En esta tercera práctica la vista será representada por un interfaz gráfico realizado completamente con el api de java, swing. Sobre este sistema se deberá utilizar el patrón MVC (modelo vista controlador).

Desde esta vista también se proporciona un sistema de entrada de los comandos al sistema.

El tiempo del sistema será controlado por dos relojes, el de la plataforma donde se está ejecutando el sistema, y un reloj interno del sistema que administrará un tiempo propio de este.

1. Conceptos y definiciones

**Herencia simple**: Se produce cuando una clase extiende o hereda únicamente de otra o de una sola interfaz.

**Herencia múltiple**: Una clase puede heredar las características de varias clases base, es decir, puede tener varios padres. En este aspecto hay discrepancias entre los diseñadores de lenguajes. Algunos de ellos han preferido no admitir la herencia múltiple debido a que los potenciales conflictos entre métodos y variables con igual nombre, y eventualmente con comportamientos diferentes crea un desajuste cognitivo que va en contra de los principio de la programación orientada a objetos. Por ello, la mayoría de los lenguajes orientados a objetos admite herencia simple.

**Herencia de comportamiento o interfaz**: Este tipo de herencia es característico del lenguaje Java, donde se utiliza para realizar la herencia múltiple.

**Polimorfismo**: posibilidad de enviar un mensaje a un grupo de objetos cuya naturaleza puede ser heterogénea. El único requisito que deben cumplir los objetos que se utilizan de manera polimórfica es saber responder al mensaje que se les envía. En lenguajes basados en clases y con un sistema de tipos de datos fuerte (independientemente de si la verificación se realiza en tiempo de compilación o de ejecución), el modo de poder utilizar objetos de manera polimórfica es que compartan una raíz común (jerarquía de clases).

**Thread:** Aunque en esta práctica no se gestionan hilos como tales, si es importante saber que son. Un thread es un único flujo de ejecución dentro de un proceso. Un thread no puede ejecutarse el solo; requieren la supervisión de un proceso padre para correr. Un proceso padre puede tener varios thread corriendo a la vez. Los thread pueden estar en 3 estados diferentes: running, wait y sleep.

* **Running**: significa que está corriendo, es decir en ejecución y seguirá así hasta que acabe su ejecución, se le mate o se le cambie el estado.
* **Wait**: Cuando está en espera, el thread está parado hasta que ocurra algún evento específico que hará que se reanude.
* **Sleep**: Si se duerme un thread, este se quedara en este estado hasta que pase el tiempo especificado previamente para que se despierte.

Existe un 4º estado llamado dead, este estado ocurre cuando el thread termina su ejecución de manera natural o si se le invoca el método stop(), que matara el hilo. Un thread en este estado no se puede revivir y ejecutar de nuevo.

**Swing**: Es un toolkit para Java. Es parte de Sun Microsystems ' Java Foundation Classes (JFC) - un API para proporcionar una interfaz gráfica de usuario (GUI) para programas de Java.

Swing fue desarrollado para proporcionar un sofisticado conjunto más del GUI componentes que el anterior Window Toolkit Resumen. Incluye widgets para interfaz gráfica de usuario tales como cajas de texto, botones, desplegables y tablas.

Algunas de sus ventajas son:

• El diseño en Java puro posee menos limitaciones de plataforma.

• El desarrollo de componentes Swing es más activo.

• Los componentes de Swing soportan más características.

**Modelo Vista Controlador** **(MVC):** es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario. Este patrón de diseño se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

De manera genérica, los componentes de MVC se podrían definir como sigue:

* **Modelo**: Es la representación específica de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la 'vista' aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al 'modelo' a través del ‘controlador’.
* **Controlador**: Responde a eventos (usualmente acciones del usuario) e invoca peticiones al 'modelo' cuando se hace alguna solicitud sobre la información (por ejemplo, editar un documento o un registro en una base de datos). También puede enviar comandos a su 'vista' asociada si se solicita un cambio en la forma en que se presenta de 'modelo' (por ejemplo, desplazamiento o scroll por un documento o por los diferentes registros de una base de datos), por tanto se podría decir que el 'controlador' hace de intermediario entre la 'vista' y el 'modelo' (véase Middleware).
* **Vista**: Presenta el 'modelo' (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario) por tanto requiere de dicho 'modelo' la información que debe representar como salida.

**Patrones**: Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Un patrón de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su **efectividad** resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser **reutilizable**, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

**Singleton**: El patrón de diseño **singleton** (instancia única) está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella.

Este patrón se implementa creando en nuestra clase un método que crea una instancia del objeto sólo si todavía no existe alguna. Para asegurar que la clase no puede ser instanciada nuevamente se regula el alcance del constructor (con atributos como protegido o privado).

La instrumentación del patrón puede ser delicada en programas con múltiples hilos de ejecución. Si dos hilos de ejecución intentan crear la instancia al mismo tiempo y esta no existe todavía, sólo uno de ellos debe lograr crear el objeto. La solución clásica para este problema es utilizar exclusión mutua en el método de creación de la clase que implementa el patrón.

Las situaciones más habituales de aplicación de este patrón son aquellas en las que dicha clase controla el acceso a un recurso físico único (como puede ser el ratón o un archivo abierto en modo exclusivo) o cuando cierto tipo de datos debe estar disponible para todos los demás objetos de la aplicación.

El patrón *singleton* provee una única instancia global gracias a que:

* La propia clase es responsable de crear la única instancia.
* Permite el acceso global a dicha instancia mediante un método de clase.
* Declara el constructor de clase como privado para que no sea instanciable directamente.

**Mediador**: El patrón mediador define un objeto que encapsula cómo un conjunto de objetos interactúan. Este patrón de diseño está considerado como un patrón de comportamiento debido al hecho de que puede alterar el comportamiento del programa en ejecución.

Habitualmente un programa está compuesto de un número de clases (muchas veces elevado). La lógica y computación es distribuida entre esas clases. Sin embargo, cuantas más clases son desarrolladas en un programa, especialmente durante mantenimiento y/o refactorización, el problema de comunicación entre estas clases quizás llegue a ser más complejo. Esto hace que el programa sea más difícil de leer y mantener. Además, puede llegar a ser difícil cambiar el programa, ya que cualquier cambio podría afectar código en muchas otras clases.

Con el patrón mediador, la comunicación entre objetos es encapsulada con un objeto mediador. Los objetos no se comunican de forma directa entre ellos, en lugar de ello se comunican mediante el mediador. Esto reduce las dependencias entre los objetos en comunicación, reduciendo entonces la Dependencia de código.

**Command**: El patrón **command** permite solicitar una operación a un objeto sin conocer realmente el contenido de esta operación, ni el receptor real de la misma. Para ello se encapsula la petición como un objeto, con lo que además se facilita la parametrización de los métodos.

* Encapsula un mensaje como un objeto, con lo que permite gestionar colas o registro de mensaje y deshacer operaciones.
* Soportar restaurar el estado a partir de un momento dado.
* Ofrecer una interfaz común que permita invocar las acciones de forma uniforme y extender el sistema con nuevas acciones de forma más sencilla.

El concepto de "orden" puede ser ambiguo y complejo en los sistemas actuales y al mismo tiempo muy extendido: intérpretes de órdenes del sistema operativo, lenguajes de macros de paquetes ofimáticos, gestores de bases de datos, protocolos de servidores de Internet, etc.

Este patrón presenta una forma sencilla y versátil de implementar un sistema basado en comandos facilitándose su uso y ampliación.

* Facilitar la parametrización de las acciones a realizar.
* Independizar el momento de petición del de ejecución.
* Implementar CallBacks, especificando que órdenes queremos que se ejecuten en ciertas situaciones de otras órdenes. Es decir, un parámetro de una orden puede ser otra orden a ejecutar.
* Soportar el "deshacer".

Desarrollar sistemas utilizando órdenes de alto nivel que se construyen con operaciones sencillas (primitivas)

1. **Objetivos que se alcanzarán**

* Comprender y utilizar los diferentes tipos de herencia, las vinculaciones dinámicas y estáticas.
* Aprender a documentar una práctica para futuros proyectos.
* Aprender el mecanismo por el cual una serie de comandos se convierten en unas órdenes que pueda interpretar nuestra aplicación.
* Aprender a adaptar los patrones de diseño de software.
* Crear proyectos software con gran capacidad de extensibilidad.
* Comprender la creación de swing.

1. **Relación con la docencia cursada**

* Esta práctica nos introduce a la herencia, el polimorfismo, y la programación orientada a objetos, así como en el testing de software y en patrones de desarrollo, y los factores de calidad del software.
* En ella van a estar involucrados los conocimientos adquiridos sobre la programación orientada a objetos, herencia, polimorfismo, vinculación dinámica y estática, abstracción de un problema real, etc...

El modelado de clases, la utilización de objetos como elementos concretos del problema. La interacción de estos elementos por el sistema de paso de mensajes que caracteriza a la programación orientada a objetos.

1. **Viabilidad**

Esta práctica contiene un grado de complejidad importante, dado que el sistema tiene que manejar ciclistas haciendo una carrera, el viento del sistema, condiciones del circuito (pendientes, curvas), las características del recorrido realizado por los ciclistas, y unidades propias de tiempo que afectas a los elementos del sistema. También incorpora todo un complejo sistema de representación de los datos y método de entrada para el usuario.

Dada esta complejidad, comienza a acercarse a la inviabilidad en cuanto a su desarrollo de una forma idónea con respecto a la correcta construcción de un software.

1. **Estado del arte y fundamentación teórica**

En esta práctica nos hemos encontrado con varios problemas. Estos problemas se citan a continuación con su correspondiente solución elegida.

**1º Problema**

A la hora de realizar el modelo del sistema, sobre todo en cuanto a la estructura de la herencia, es preciso conocer el porqué de la existencia de los tipos de herencia en el lenguaje Java.

* **Antecedente**

La herencia en la programación orientada a objetos es algo habitual y fundamental, ya que con ella se alcanzan algunos de los objetivos más preciados en el desarrollo del software como son la reutilización y la extensibilidad.

El problema de la herencia múltiple que ocurre en lenguajes como C++, estos problemas vienen debido a las situaciones en las que potencialmente se pueden producir conflictos entere métodos y variables con igual nombre, eventualmente con comportamientos diferentes, esto crea un desajuste cognitivo que va en contra de los principios de la programación orientada a objetos.

* **Propuestas de solución**:
  1. Primera solución:
     + Diseñar mediante el proceso de abstracción una clase que represente las personas y de ella se obtendrá una que represente al ciclista.
     + Diseñar un modelo de herencia para la bicicleta y sus diferentes tipos.
     + Cada entidad o elemento del sistema poseerá un método para proveer una salida de datos, correspondiente a su estado.
  2. Segunda solución:
     + Diseñar mediante el proceso de abstracción una clase que represente las personas y de ella se obtendrá una que represente al ciclista.
     + Diseñar un modelo de herencia para la bicicleta y sus diferentes tipos.
     + Se provee con la herencia de interfaz un comportamiento común, tanto para las entidades que se ejecutarán como para las que poseerán una salida de datos, siendo estos dos comportamientos interfaces diferentes.
     + Se diseñará un sistema de órdenes para la comunicación externa no el sistema.
     + Utilizar el patrón modelo-vista-controlador (MVC), para diferenciar el sistema central (modelo), la representación de los datos (vista), y la interacción del usuario con el sistema (controlador).
* **Desestimaciones**

Se desestima la primera solución por no cumplir con las directrices de la programación orientada a objetos, ya que no es un código con dificultad de mantenimiento, por ejemplo: si a la hora de cambiar la salida de datos o el método en el que se muestra hay que cambiar cada una de las clases que van a mostrar los datos.

* **Solución escogida**

Se ha elegido la segunda solución porque se cree que se cumple con el paradigma de la programación orientada a objetos.

Con esta solución se ha podido realizar la práctica cumpliendo con las especificaciones de esta, y superando las pruebas realizadas para garantizar que se cumple con la especificación y que su funcionamiento es correcto (dentro del ámbito de las pruebas desarrolladas para esta práctica).

**2º Problema**

Era necesario introducir un sistema de reconocimiento de comandos o instrucciones que atendiesen a través del teclado y de ficheros. Además este sistema debe evitar la espera activa del sistema

* **Propuesta de solución**

Solución

* Se ha creado una clase para cada entrada de comando, por ficheros y por teclado. Estas dos clases heredan de una interfaz que les provee de un mismo comportamiento. Y también hemos hecho uso de una clase que se sirve de esas dos para gestionar las lecturas y cargas de ficheros.
* Para construir las órdenes necesarias en el sistema a partir de los comandos introducidos se ha optado por crear un sistema de reconocimiento de comandos.
* Este sistema se vale de una clase distribuidor (“Comandero”) que se encarga de distribuir las órdenes a los elementos que les correspondan.

Esta fue la solución escogida puesto que fue la única propuesta que se nos ocurrió y encajaba perfectamente con las necesidades de la práctica.

**3º Problema**

Al introducir una carretera por la que circula el ciclista y que afecta al estado de la misma, nos hemos encontrado con una serie de consideraciones a tener en cuenta. Como él quien se va a encargar de modificar el estado de la bicicleta. Para ello se nos ocurrieron dos soluciones.

* **Propuesta de solución**

1. Primera solución
   * + Crear un mapa de carreteras que tuviera como valor una clase tramo. Esa clase tendría el km inicial, la pendiente, la dirección del viento y la velocidad del viento en ese tramo.
     + Luego ese mapa se añadiría como un atributo a la clase Bicicleta y esta se ocuparía de modificar la velocidad en proporción del tramo de la carretera en el que estuviera circulando.
2. Segunda solución
   * + Crear un mapa de carreteras que tuviera como valor una clase tramo. Esa clase tendría el km inicial, la pendiente, la dirección del viento y la velocidad del viento en ese tramo.
     + Crear una nueva clase llamada FactoreExternos que heredaría de la interface ObjetosQueSeEjecutan. Esta clase tendría como atributos el mapa de carreteras y un objeto tipo Bicicleta.
     + Esta clase se ocuparía de mirar el tramo en el que se encuentra la bicicleta y modificar la velocidad en proporción al tramo en el que se encuentra.
     + En el manager se tendría que ejecutar la clase.

* **Desestimaciones**

Se desestimó la primera solución ya que añadía nuevos atributos a la clase bicicleta lo que la hacía menos portable y reutilizable. Además nos parecía que al ser unos factores que no tenían nada que ver con la bicicleta no tenía sentido que se incorporara a esta.

* **Solución Escogida**

Se eligió la segunda solución, ya que nos pareció más adecuado que al ser unos elementos externos, estos fueran externos a la bicicleta. Además así le añadía un grado más alto de portabilidad y reusabilidad.

**4º Problema**

Este problema viene referido a la vista del sistema, lo primero que ocurre en estos casos cuando se usas sistemas con componentes pesados (swing) es el hecho de que tardan mucho más tiempo en estar listos para que los datos del modelo lleguen a las vista. Y no es una buena solución que una vista, como es la realizada con swing, pertenezca al mismo hilo que el del sistema ya que el sub-sistema de eventos puede funcionar lentamente e incluso erráticamente. Por lo tanto se ha de separar en otro hilo.

Solución

* Utilizar las herramientas que proporciona java para lanzar interfaces gráficas en un hilo diferente, y así liberar el sistema de eventos de esta.

1. **Conclusión**

Al realizar esta práctica hemos aprendido la parte de la programación orientada a objetos que compete la herencia, que es algo fundamental en este paradigma de programación. Hemos trabajado los diferentes tipos de herencia, simple, múltiple, y de comportamiento o interfaz.

Se ha profundizado en la vinculación dinámica y estática en Java, cuya repercusión en la memoria puede llegar a ser notable en problemas de software de gran envergadura.

1. **Líneas Futuras**

De cara al futuro se espera seguir mejorando los conocimientos sobre el lenguaje java y la programación orientada a objetos, con el fin de construir software que cumpla con los principios del software y de calidad.

1. **Bibliografía**

**Estructura del documento actual:**

* Transparencias sobre la estructura de las prácticas, del profesor Francisco Javier Crespo.

**Documentación sobre la bicicleta y carreras ciclistas:**

* <http://es.wikipedia.org/wiki/Bicicleta>

**Programación orientada a objetos:**

* Piensa en Java 4ª Edición, Bruce Eckel
* <http://es.wikipedia.org/wiki/Clase_%28inform%C3%A1tica%29>
* <http://es.wikipedia.org/wiki/Metadato>
* <http://es.wikipedia.org/wiki/Objeto_%28programaci%C3%B3n%29>
* <http://es.wikipedia.org/wiki/Herencia_%28programaci%C3%B3n_orientada_a_objetos%29>
* <http://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o>

**Herramientas:**

* <http://www.cjavaperu.com/pdf/uml2.pdf>

**Física:**

* <http://www.didactika.com/fisica/cinematica/movimiento_rectilineo_uniformemente_variado.html>