Curso de Java

Unidad II

“Conceptos Básicos de Java”

Rogelio Ferreira Escutia

Contenido

1) Conceptos

1) Conceptos

Java

Objetivos de Java:

- **Orientado a Objetos**
- **Distribuido**
- **Simple**
- **Multithreaded (Multihilos, Multitarea)**
- **Seguro**
- **Independiente de plataforma**

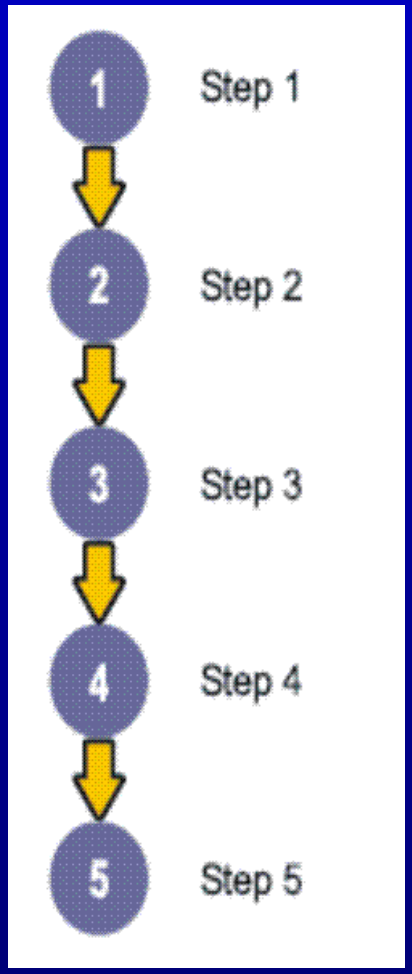
Java

Orientado a Objetos

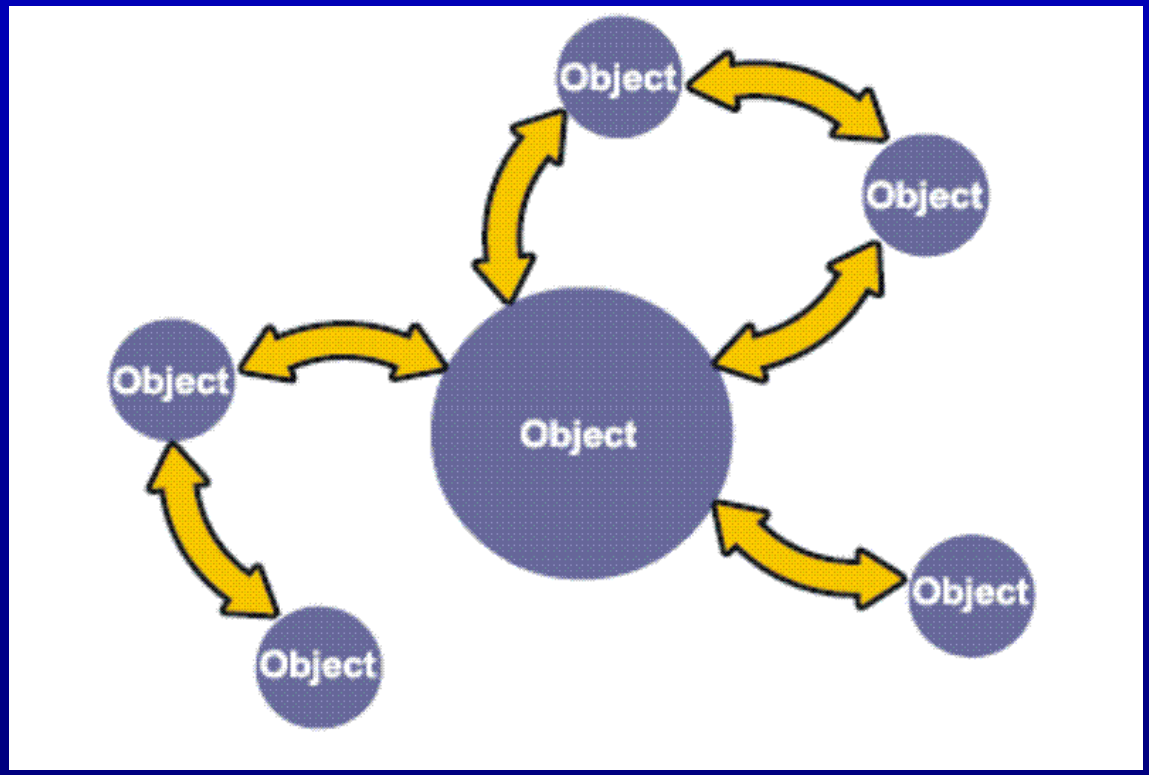
- Uno de los principales objetivos de Java es crear objetos, *piezas de código autónomo*, que puedan interactuar con otros objetos para resolver un problema.

- La POO difiere de la *programación procedural* porque ésta enfatiza la secuencia de pasos codificados para resolver un problema, mientras que la POO hace énfasis en la creación e interacción de objetos.

Java



Secuencial



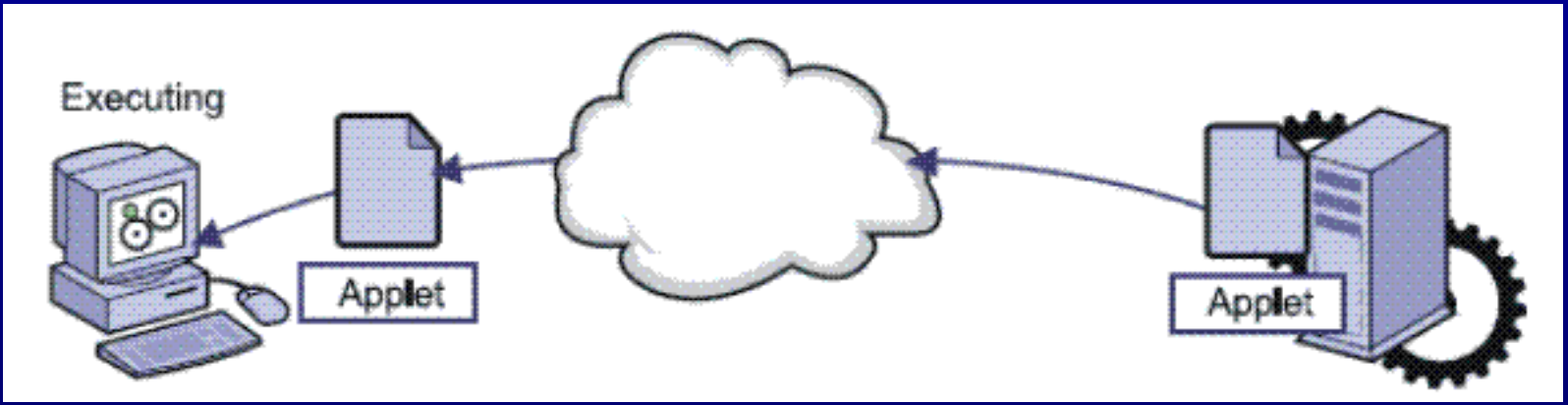
Orientado a objetos

Java

Distribuido

- Proporciona soporte para tecnologías de redes distribuidas, tal como RMI (Remote Method Invocation), CORBA (Common Object Request Broker Architecture, y URL (Universal Resource Locator).

- Adicionalmente, las capacidades de carga dinámica de clases permite que piezas de código sean descargadas de Internet y ejecutadas en una computadora personal.



Java

Diferencias con otros lenguajes:

- **Se quitó mucho de lo complejo o programación oscura que se encuentra en otros lenguajes de programación.**
- **No permite que los programadores manipulen directamente *apuntadores* a localidades de memoria, ya que Java solo permite manipular objetos usando referencias a objetos.**
- **Existe el *Garbage Collector* para monitorear y eliminar objetos que no estén referenciados.**
- **El tipo de datos *boolean* solo pueden tener valor de true o false.**

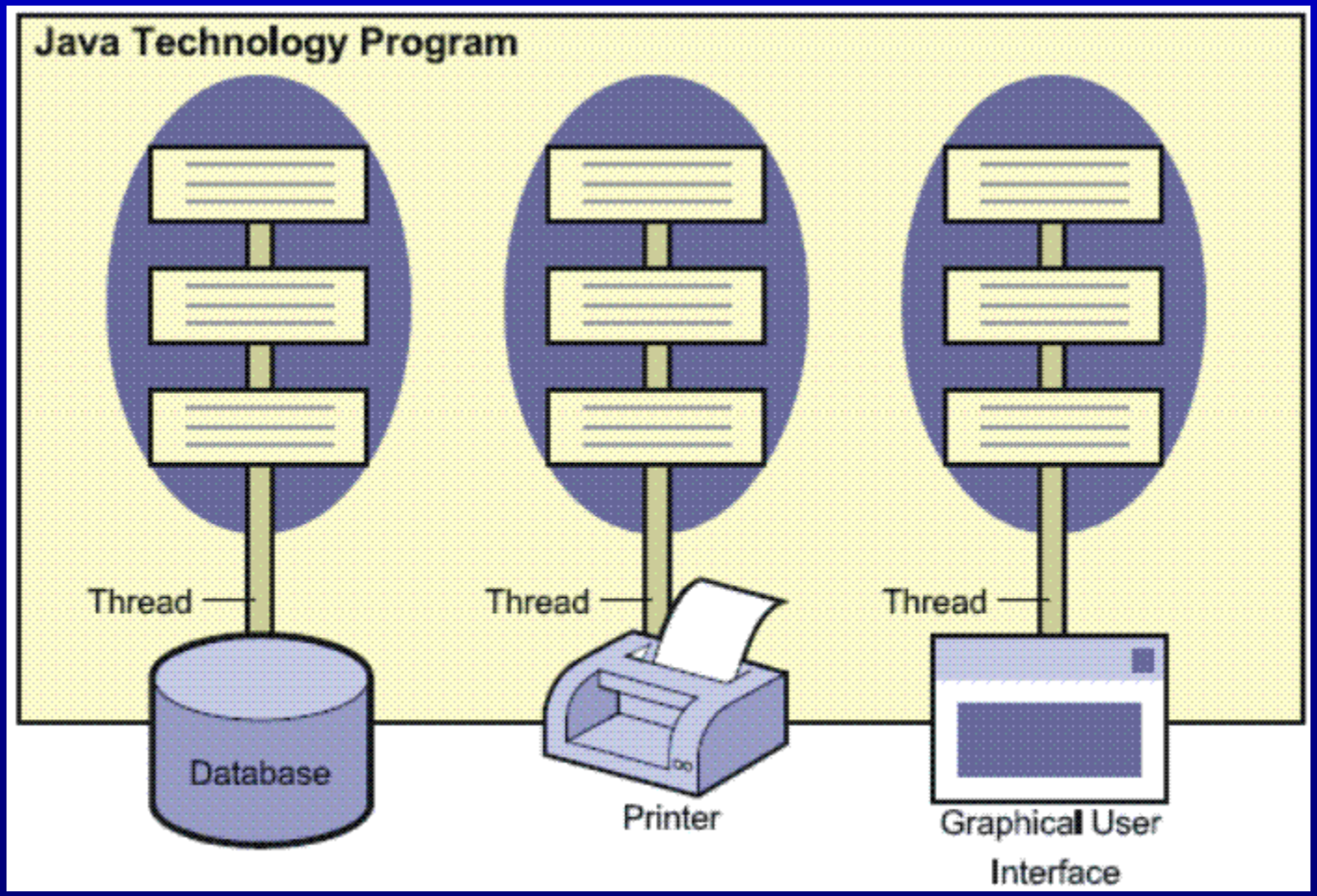
Java

Multihilos:

- **Esto significa que ejecuta varias tareas en un tiempo, tal como preguntar a una base de datos y desplegar una interfaz de usuario.**

- **Ser multihilos permite que un programa pueda ser muy eficiente en el uso de los recursos del sistema.**

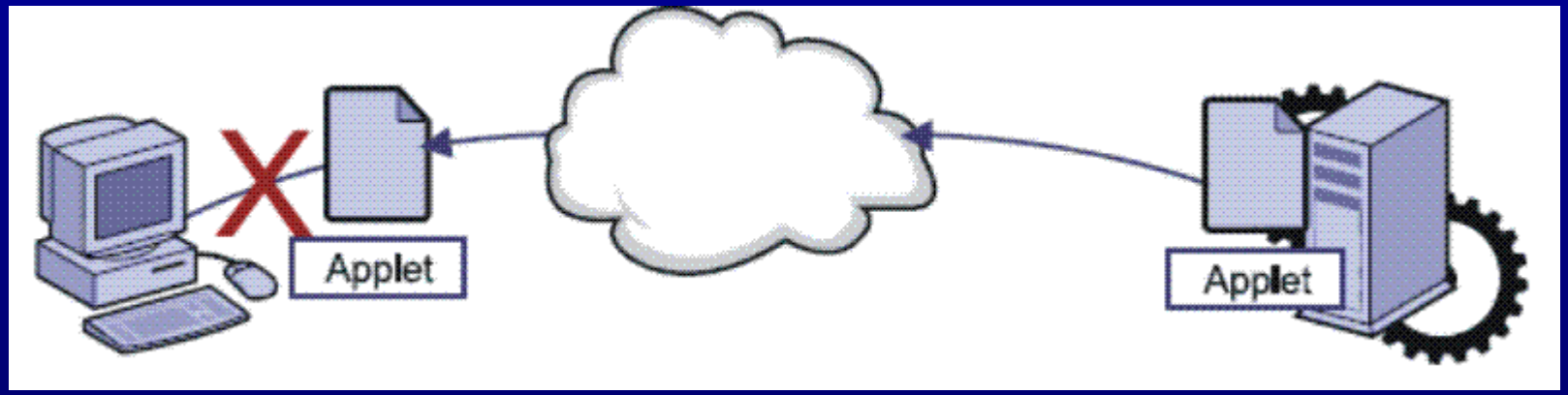
Java - Multihilo



Java

Seguridad:

- Prohíbe la manipulación de memoria usando punteros.
- Verifica que todos los programas Java contengan código válido.
- Soporta digital signatures.
- El código Java puede ser "firmado" por una compañía o persona, de manera que la persona que reciba el código puede verificar la legitimidad del código.



Java

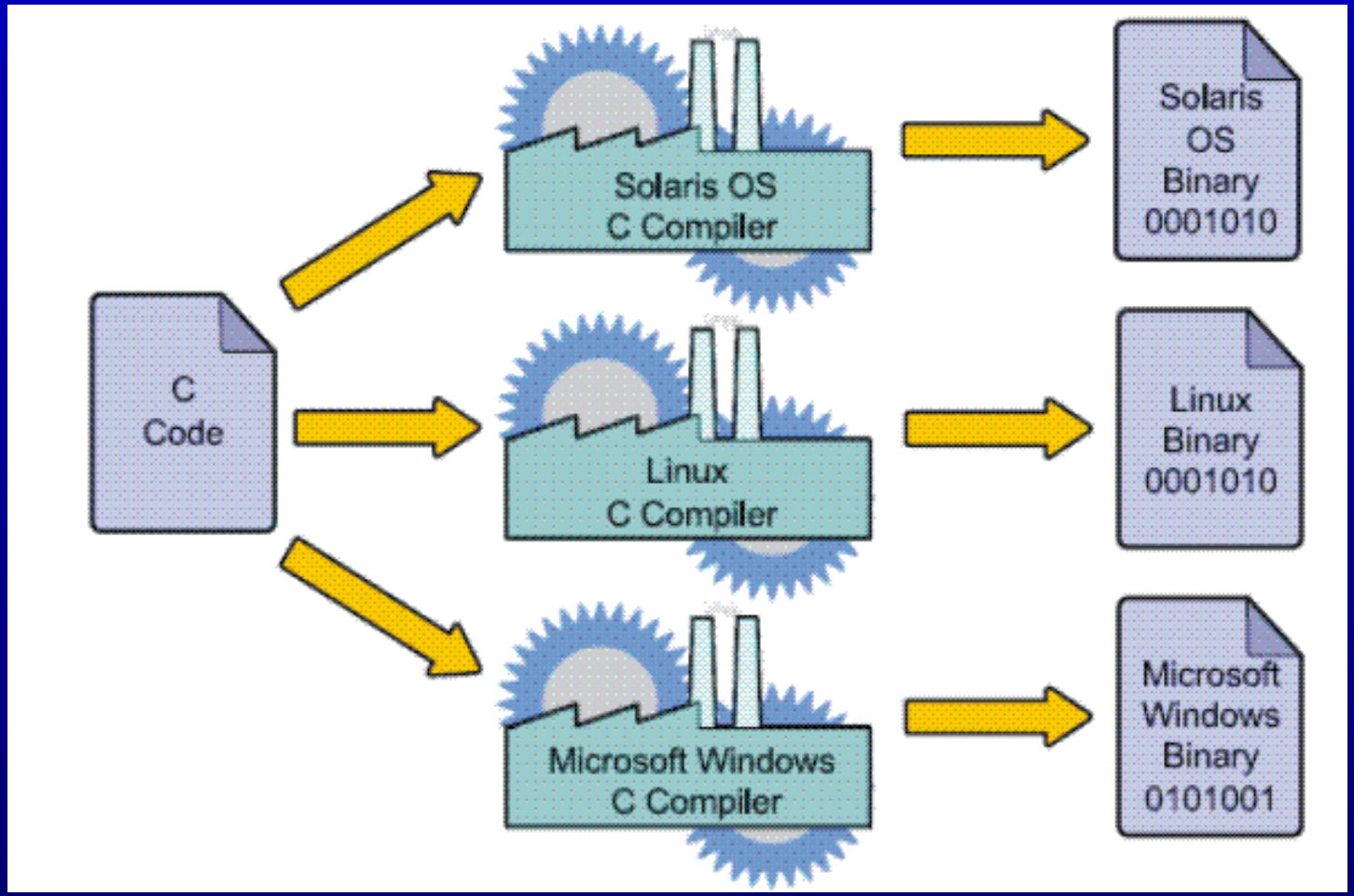
Independiente de la Plataforma:

- **La mayoría de los LP requieren compilar y ligar sus programas, para dar como resultado un programa ejecutable para una plataforma específica. A diferencia de esto, Java es de plataforma independiente.**

- **Los programas de Java pueden ejecutarse en diferentes CPU's con diferentes Sistemas Operativos, con muy pocas modificaciones o quizás con ninguna.**

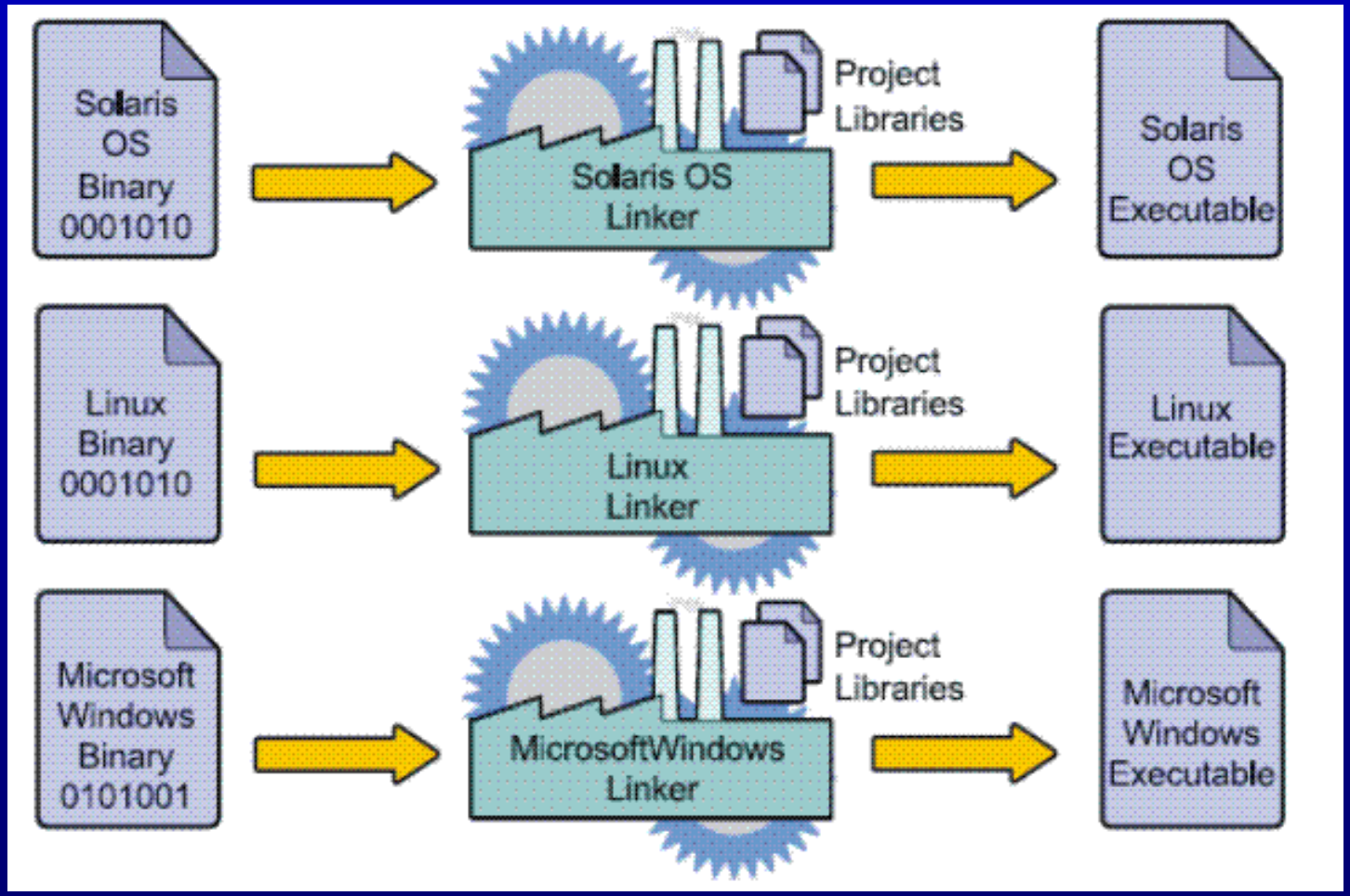
Java

- Dependiente de la Plataforma (creando un archivo binario)



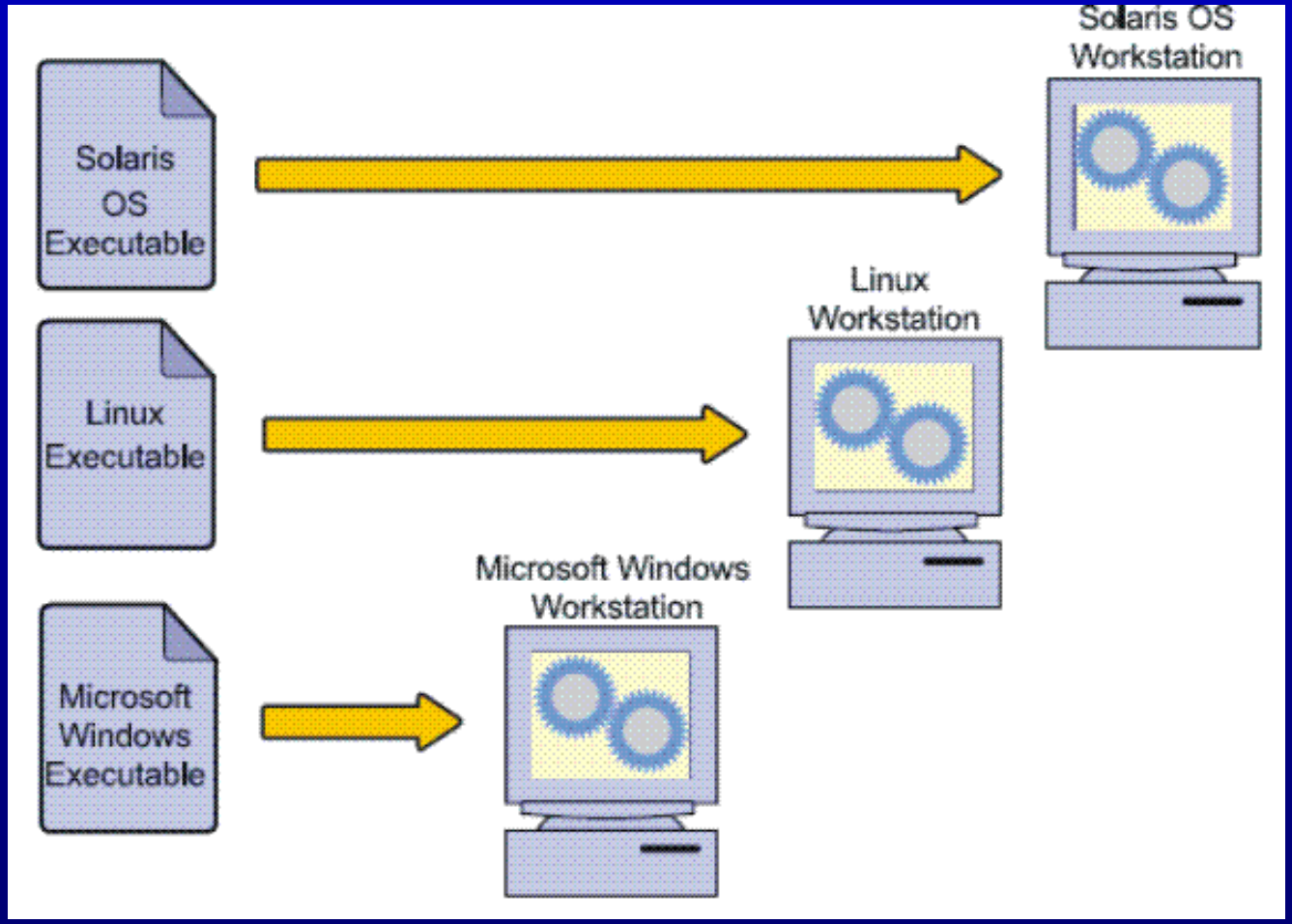
Java

- **Dependiente de la Plataforma (creando archivo ejecutable)**



Java

- **Dependiente de la Plataforma (ejecutable)**



Java

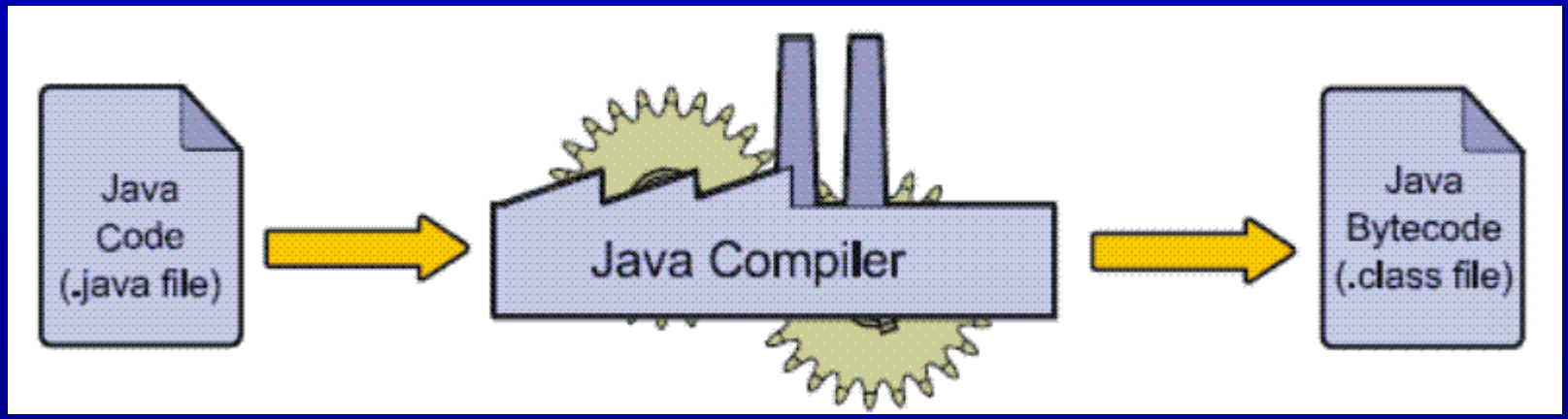
Java es Independiente de la Plataforma:

- **El resultado de compilar un programa de Java es un *bytecode* en lugar de *machine code* (código máquina) de una máquina específica. Después de que el bytecode es creado, éste es interpretado (ejecutado) por un intérprete de bytecode llamado la *virtual machine*.**

- **Una *virtual machine* es un programa que entiende (interpreta) el bytecode y lo ejecuta sobre una plataforma particular. Por esta razón, se dice que Java es un lenguaje *interpretado*, y sus programas portables o ejecutables en cualquier plataforma. Para un programa Java se necesita una JVM para cada plataforma donde el programa vaya a ejecutarse. Sin embargo, se necesita también un conjunto de *class libraries* . La combinación de JVM y las librerías de clases se les llama Java Runtime Environment (JRE), el cuál está disponible para muchas plataformas.**

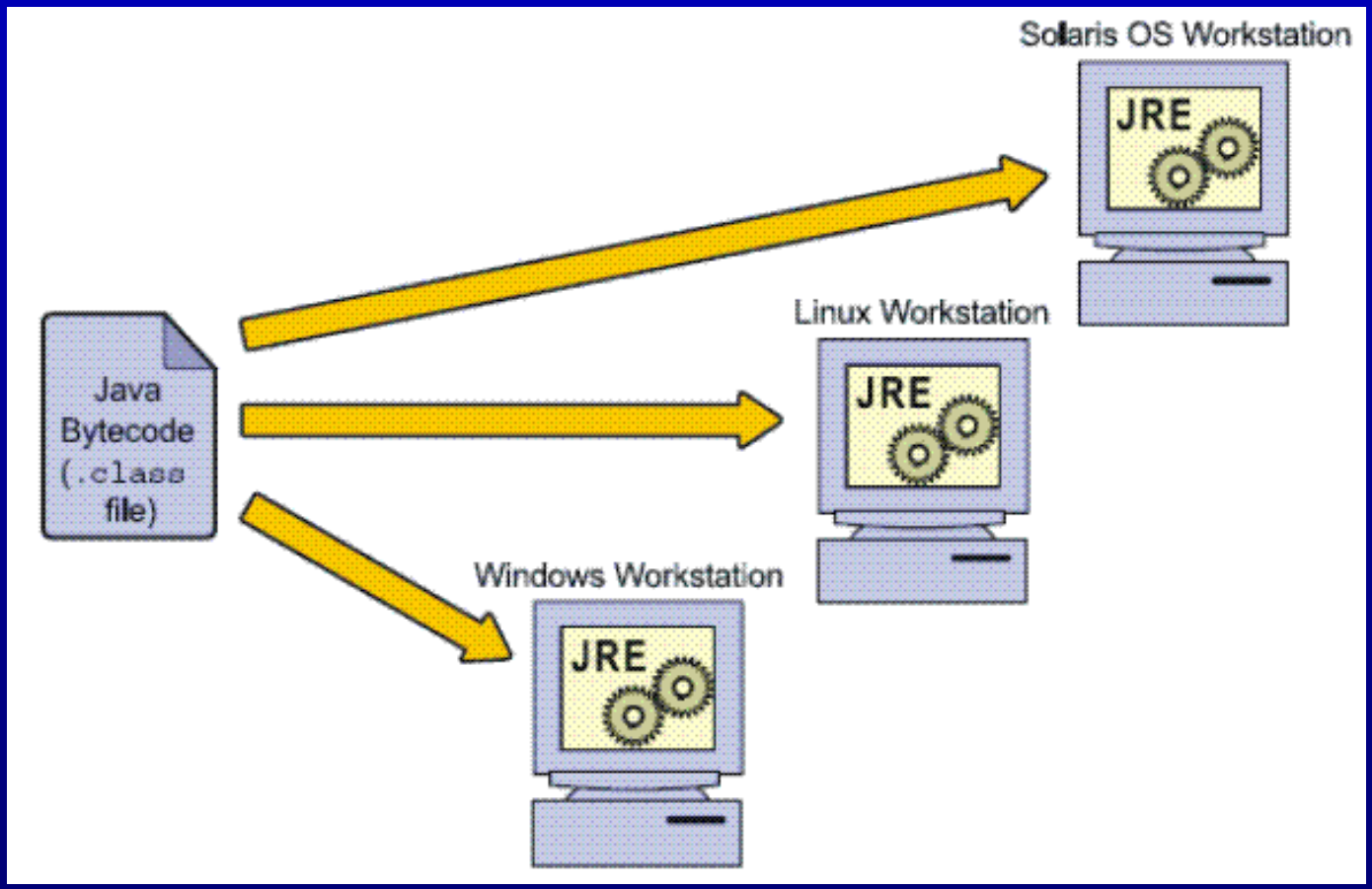
Java

Creando Bytecode



Java

Ejecutando Bytecode



Java

Java Virtual Machine (JVM):

- **Carga código - Ejecutado por el *class loader.***
- **Verifica código - Ejecutado por el verificador de bytecode.**
- **Ejecuta código - Ejecutado por el intérprete en tiempo de ejecución (*runtime interpreter*).**

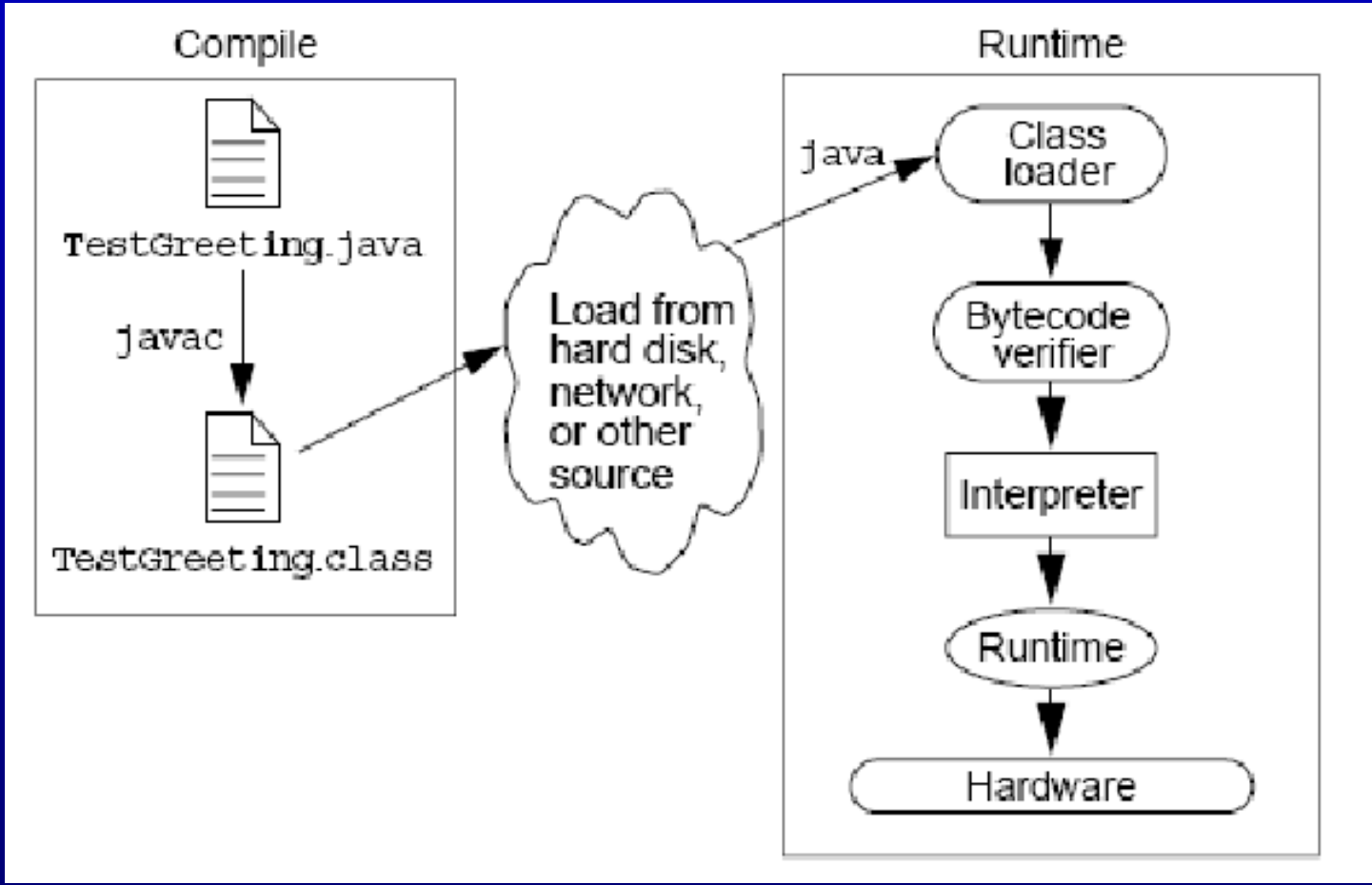
Java

Garbage Collector:

- Java elimina la responsabilidad de asignar o liberar memoria.
- Proporciona un proceso a nivel sistema que rastrea cada asignación de memoria.
- Durante el ciclo inactivo de la JVM, el garbage collector checa y libera la memoria que pueda ser liberada.
- El Garbage Collector es automático durante el tiempo de vida de un programa, eliminando la necesidad de liberar memoria y evitar desperdicio de la misma.

Java

- **Java Runtime Environment**



Java

- **Una aplicación Java**

Code 1-1 The TestGreeting.java Application

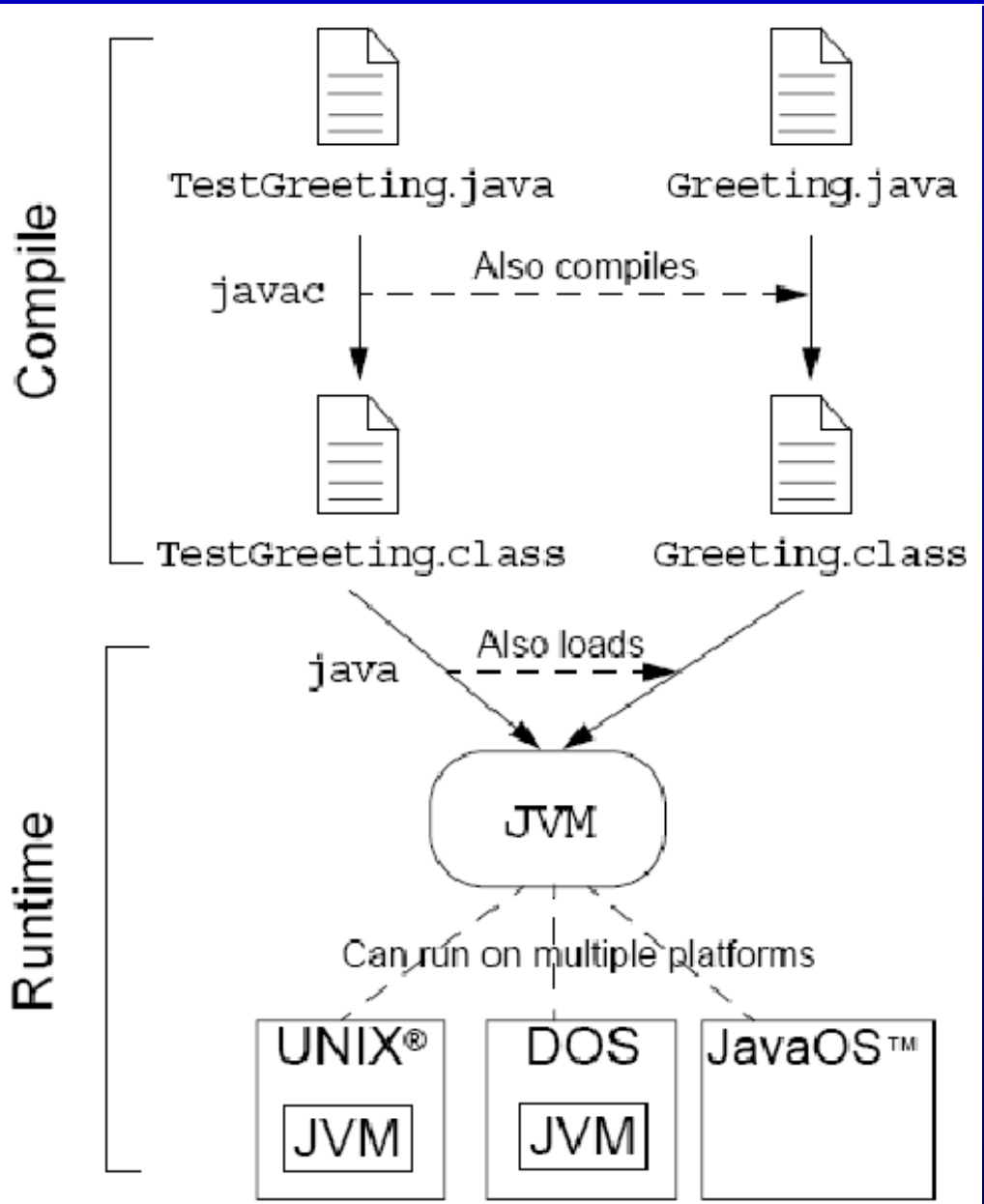
```
1   //
2   // Sample "Hello World" application
3   //
4   public class TestGreeting {
5      public static void main (String[] args) {
6         Greeting hello = new Greeting();
7         hello.greet();
8      }
9   }
```

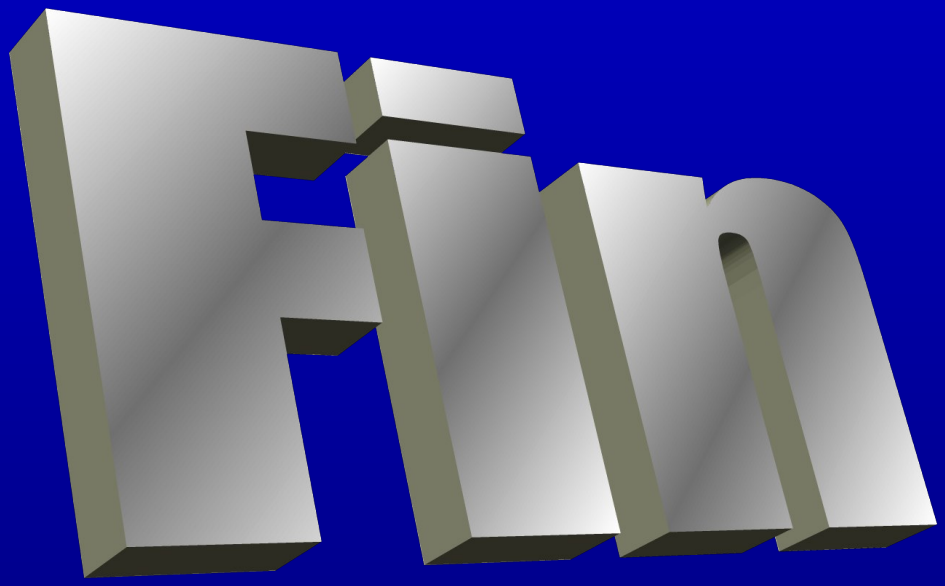
Code 1-2 The Greeting.java Class

```
1   public class Greeting {
2       public void greet() {
3          System.out.println("hi");
4       }
5   }
```

Java

- **Ejecutando Java**





Unidad II – Conceptos Básicos de Java