Programación en JAVA

Unidad 1
Introducción

Universidad de Chile
Departamento de Ciencias de la Computación
Profesor: Felipe Aguilera V.
faguiler@dcc.uchile.cl, felipe@aguilera.cl

Temario

- Crisis del software
- Descomposición, abstracción, generalización
- Reusabilidad
- Mantenibilidad
- Programación Orientada a Objetos

Crisis del Software

- A fines de los años ‘60 se comenzó a hablar de “Crisis del Software”:
  - Dificultad para construir sistemas grandes confiables, de forma controlada -> “Programming in the Large”
  - Bajo nivel de reutilización de código
  - Alto costo de mantenimiento

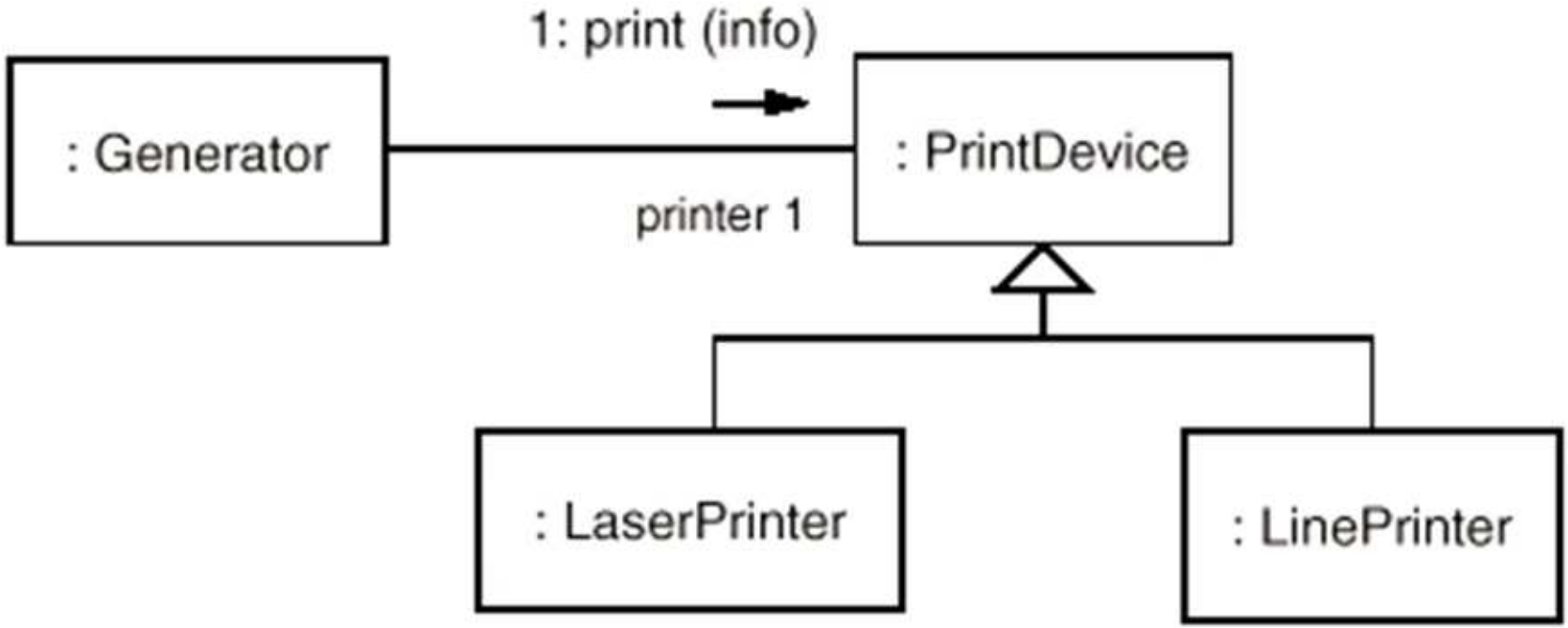
Enfrentando la Complejidad

- Grady Booch propone 3 técnicas para enfrentar la complejidad inherente al software (Grady Booch, Object-Oriented Analysis and Design with Applications, 1994):

  - Descomposición
  - Abstracción
  - Generalización

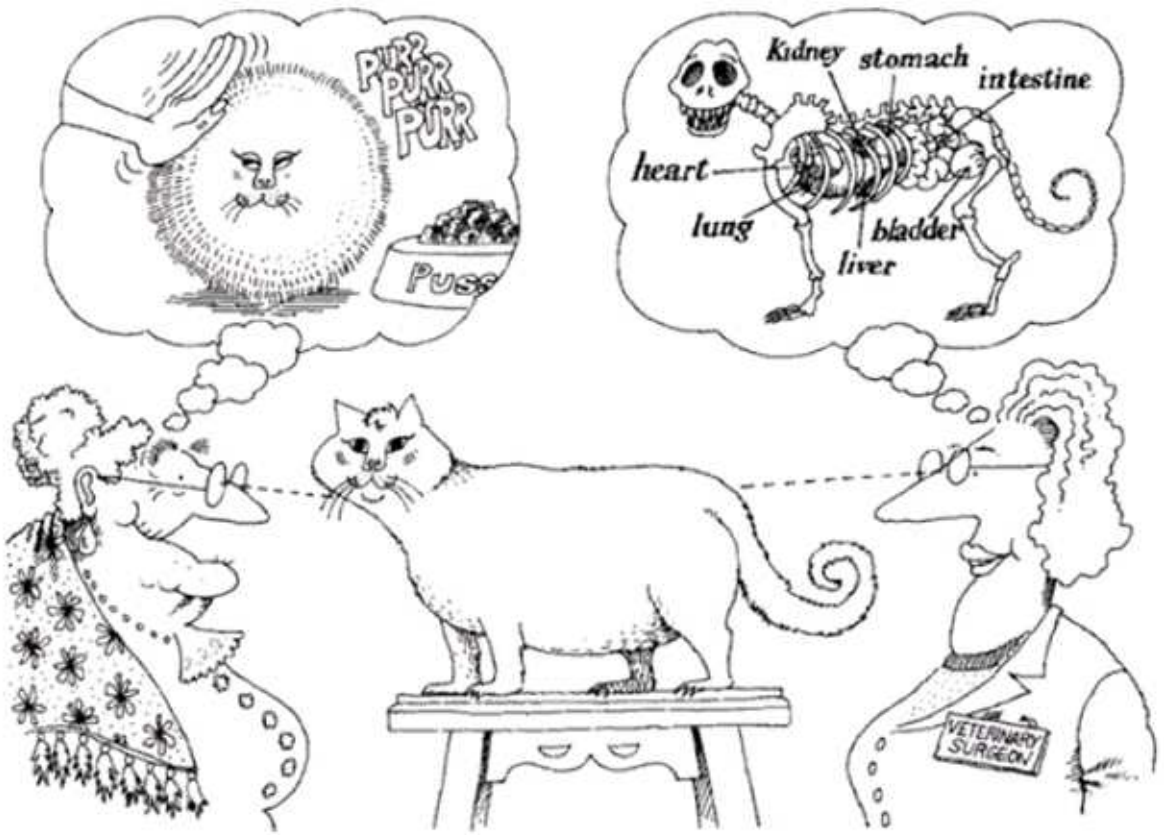
1. Descomposición

- En Diseño Orientado a Objetos (OOD) se utiliza descomposición en clases y objetos



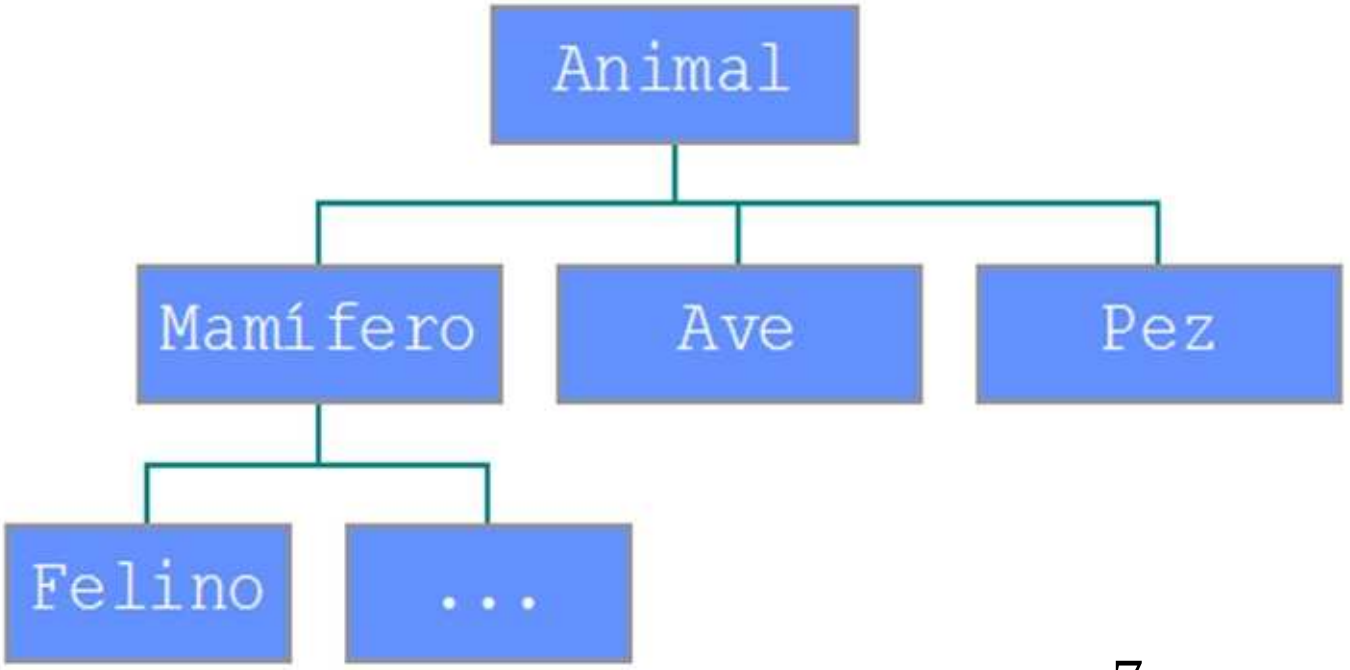
2. Abstracción

- Ignorar los detalles no esenciales de un objeto, manteniendo un modelo simplificado



3. Generalización

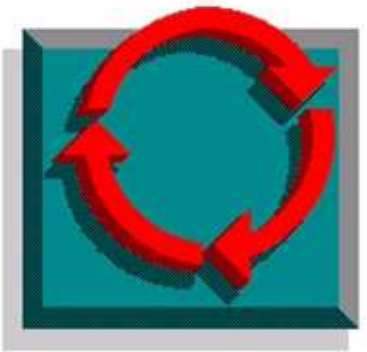
- Clasificación de los objetos en grupos de abstracciones relacionadas, explicitando el hecho de que comparten ciertas propiedades



Dificultad en la Reutilización

- Dificultad para construir software reutilizable

- Dificultad para socializar el software construido



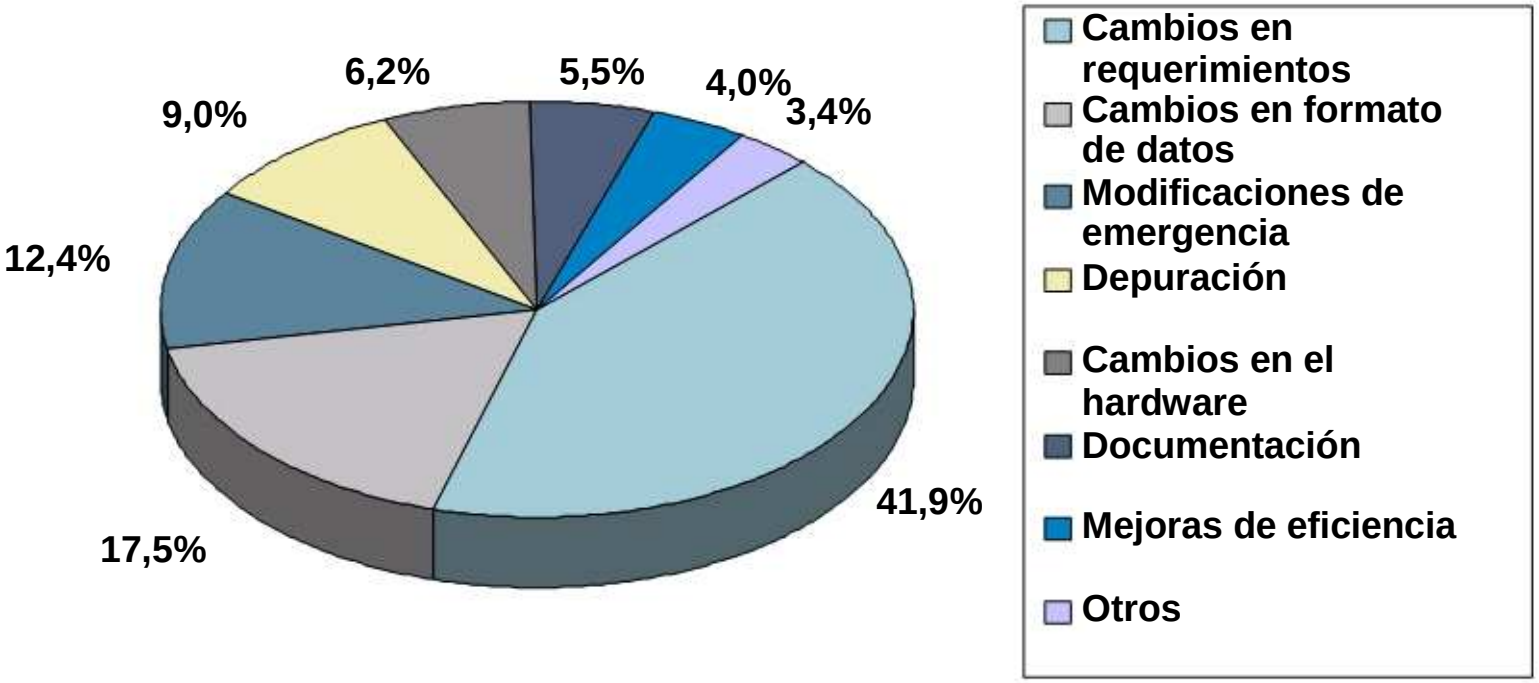
Dificultad en la Reutilización

```
// ¿Pertenece el elemento e al contenedor c?
boolean busca(CONTENEDOR c, ELEMENTO e) {
    POSICION  pos =  POSICION_INICIAL  (c, e);
    while (  POSICION_VALIDA  (c, pos)) {
        if (ENCONTRADO(c, pos, e))
            return true;
        else
            pos = POSICION_SIGUIENTE(c, pos, e);
    }
    return false;
}
```

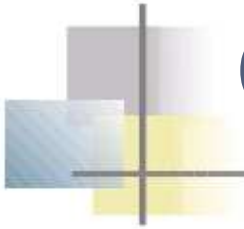
- En el ejemplo anterior, el código en mayúsculas es difícil de generalizar en un lenguaje como C

Dificultad en el Mantenimiento

- Costos de modificación de software (ref: Bertrand Meyer, Object-Oriented Software Construction, 1988)



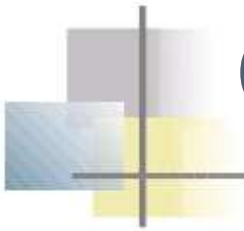
- La modularidad es el elemento clave para aumentar la reusabilidad y la mantenibilidad



OOP

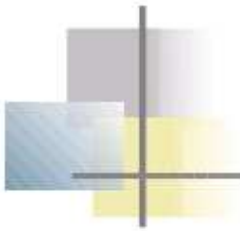
- Object-oriented programming has roots that can be traced to the 1960s. The technology focuses on data rather than processes, with programs composed of self-sufficient modules ("classes"), each instance of which ("objects") contains all the information needed to manipulate its own data structure ("members"). This is in contrast to the existing modular programming which had been dominant for many years that focused on the function of a module, rather than specifically the data.
- Object-oriented programming (OOP) is a programming paradigm that uses "objects" - data structures consisting of data fields and methods together with their interactions - to design applications and computer programs.
- An object is a discrete bundle of functions and procedures, often relating to a particular real-world concept such as a bank account holder or hockey player. Other pieces of software can access the object only by calling its functions and procedures that have been allowed to be called by outsiders.
- An object-oriented program may thus be viewed as a collection of interacting objects, as opposed to the conventional model, in which a program is seen as a list of tasks to perform.
- In OOP, each object is capable of receiving messages, processing data, and sending messages to other objects. Each object can be viewed as an independent 'machine' with a distinct role or responsibility. The actions (or "methods") on these objects are closely associated with the object.

Wikipedia



OOP

- Herramientas de la Programación Orientada a Objetos
  - Clases: información y comportamiento
  - Encapsulación: ocultamiento de la implementación
  - Herencia: creación de clases a partir de clases existentes
  - Polimorfismo: diferentes clases pueden verse a través de interfaces comunes
- Principales lenguajes OO:
  - 1967: Simula
  - 1969: Smalltalk
  - 1983: C++
  - 1995: Java
  - 2000: C#, VB.NET



Simula

- Simula I (1964) y Simula 67 (1967), desarrollados en el Centro de Computación de Noruega por Ole-Johan Dahl y Kristen Nygaard, son los dos primeros lenguajes orientados a objetos

- Influenciado por Algol 60

- Simula 67 introdujo los principales conceptos de OOP: clases, objetos, encapsulación, herencia simple, overriding de métodos, y garbage collection

Smalltalk

- Smalltalk (1969) fue desarrollado en Xerox PARC por Alan Kay, Dan Ingalls y Adele Goldberg

- Influenciado por Simula y Lisp

- Lenguaje orientado a objetos puro
- Soporta clases, metaclases, encapsulación, herencia simple, polimorfismo, garbage collection
- En Smalltalk, todo es un objeto, y cada objeto pertenece a una clase

C++

- C++ (1983) fue desarrollado en Bell Laboratories por Bjarne Stroustrup

- Influenciado por C y Simula

- Lenguaje orientado a objetos híbrido, compatible con C

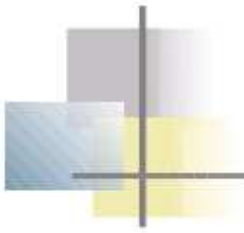
- Soporta clases, encapsulación, sobrecarga de funciones y operadores, herencia múltiple, polimorfismo, clases/funciones parametrizadas (genéricas)

Java

- Java (1995) fue desarrollado en Sun Microsystems por
  James Gosling, Bill Joy y
  Guy Steele

- Influenciado por C++

- Soporta clases, encapsulación, herencia simple, polimorfismo, interfaces, garbage collection



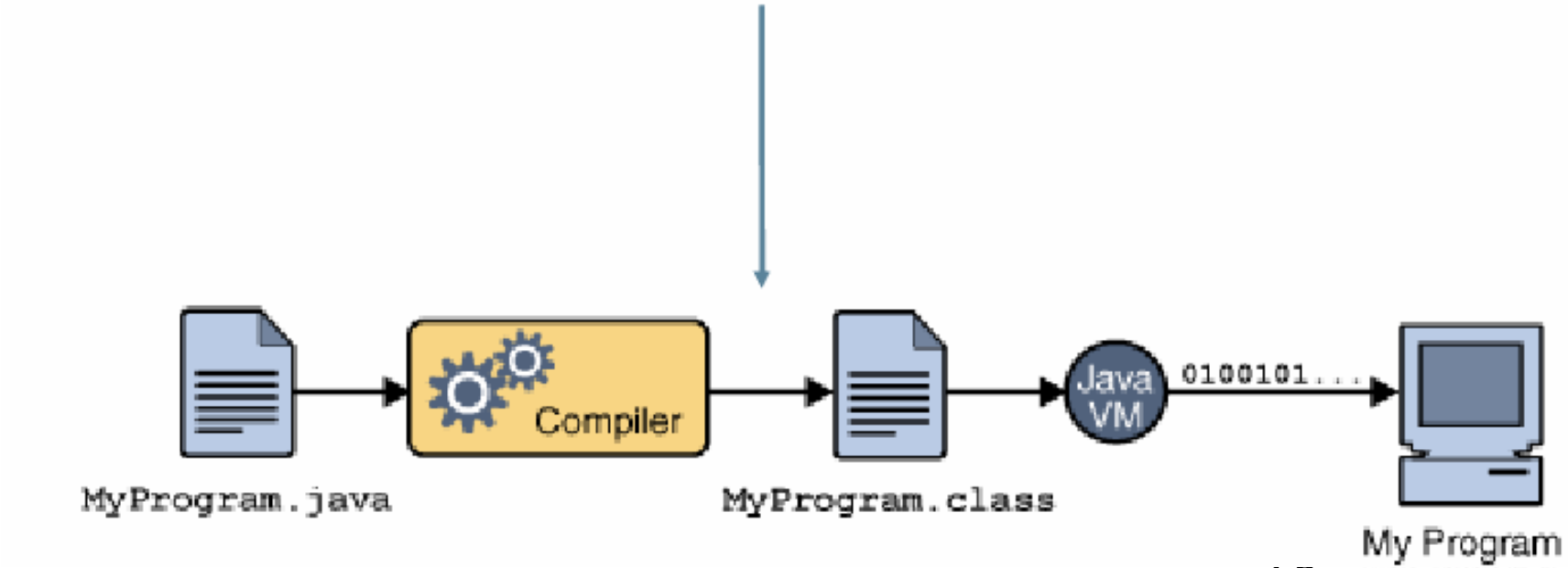
El Lenguaje Java

- Independiente de la plataforma
- Seguro
- Simple
- Robusto
- Orientado a Objetos
- Distribuido
- Multi-threaded

- Ref: http://java.sun.com/docs/overviews/java/java-overview-1.html

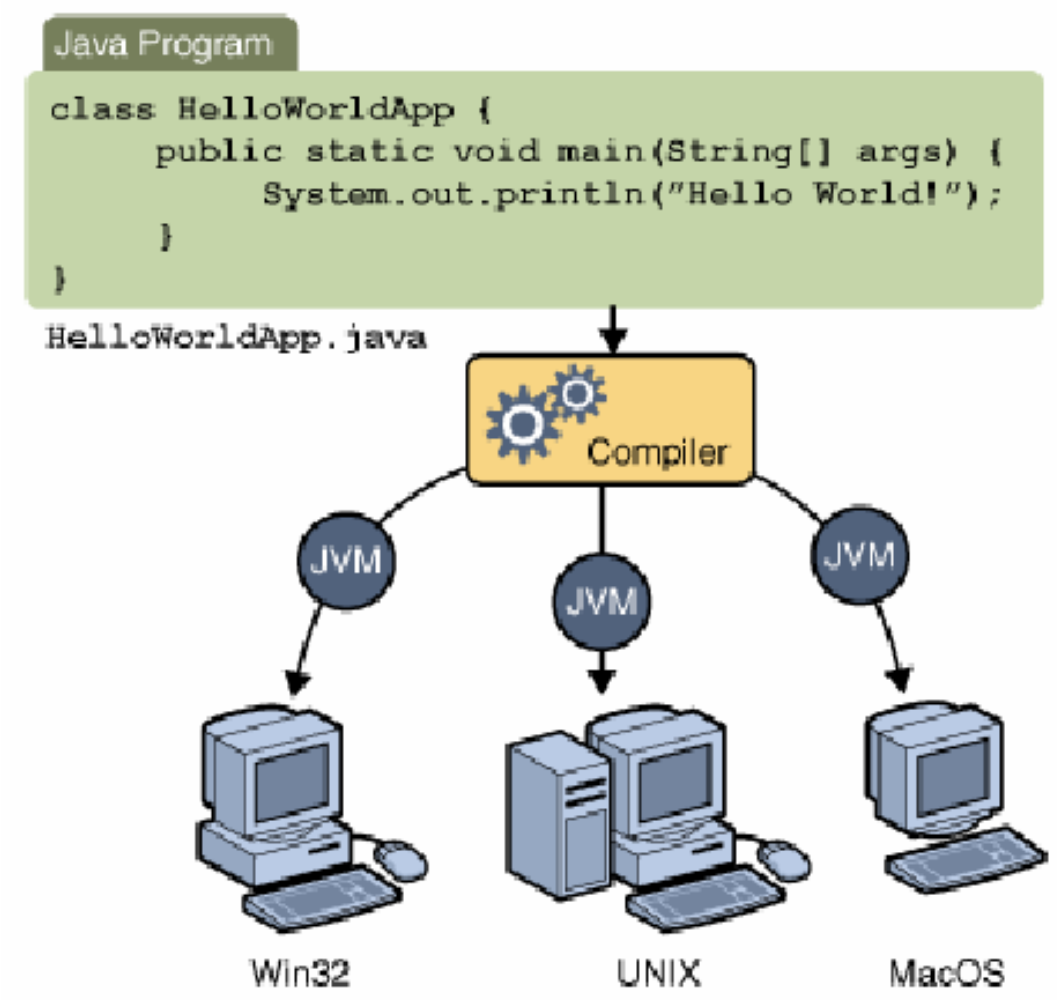
El Modelo Java

- Al compilar un programa Java, se genera un código de máquina intermedio definido por Sun, que recibe el nombre de bytecode



El Modelo Java

- El código bytecode es portable entre diferentes plataformas



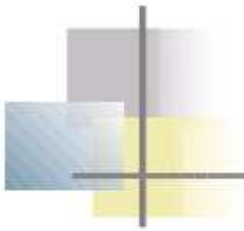
C#

- C# (2000) fue creado en Microsoft por Anders Hejlsberg

- Influenciado por Java (objetos manejados mediante referencias, herencia simple e interfaces, garbage collection) y C++ (sobrecarga de operadores)

- Novedades: propiedades, delegados, eventos

- Permite crear código intermedio que corre en el Common Language Runtime (CLR), en .NET



Resumen

- El objetivo de la OOP es aumentar la modularidad del código, mejorando su reusabilidad y mantenibilidad
- Principales técnicas de OOP: descomposición, abstracción y generalización
- Lenguages orientados a objetos:
  - Simula 67
  - Smalltalk
  - C++
  - Java
  - C#