IN77J – Orientación al Objeto para el e-business

Unidad 1

Introducción

Temario

- 1. Introducción
  - Crisis del Software
  - Programación Procedural
  - Orientación a Objetos
  - Principales Lenguajes OO

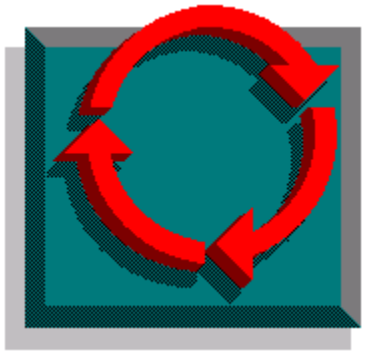
Crisis del Software

- A fines de los años ‘60 se comenzó a hablar de “Crisis del Software”:
  - Bajo nivel de reutilización de código
  - Alto costo de mantenimiento
  - El tamaño de los sistemas impacta directamente en su complejidad (“Programming in the Large”)

Acerca de la Reutilización

- Dificultad para construir software reutilizable

- Dificultad para socializar el software construido



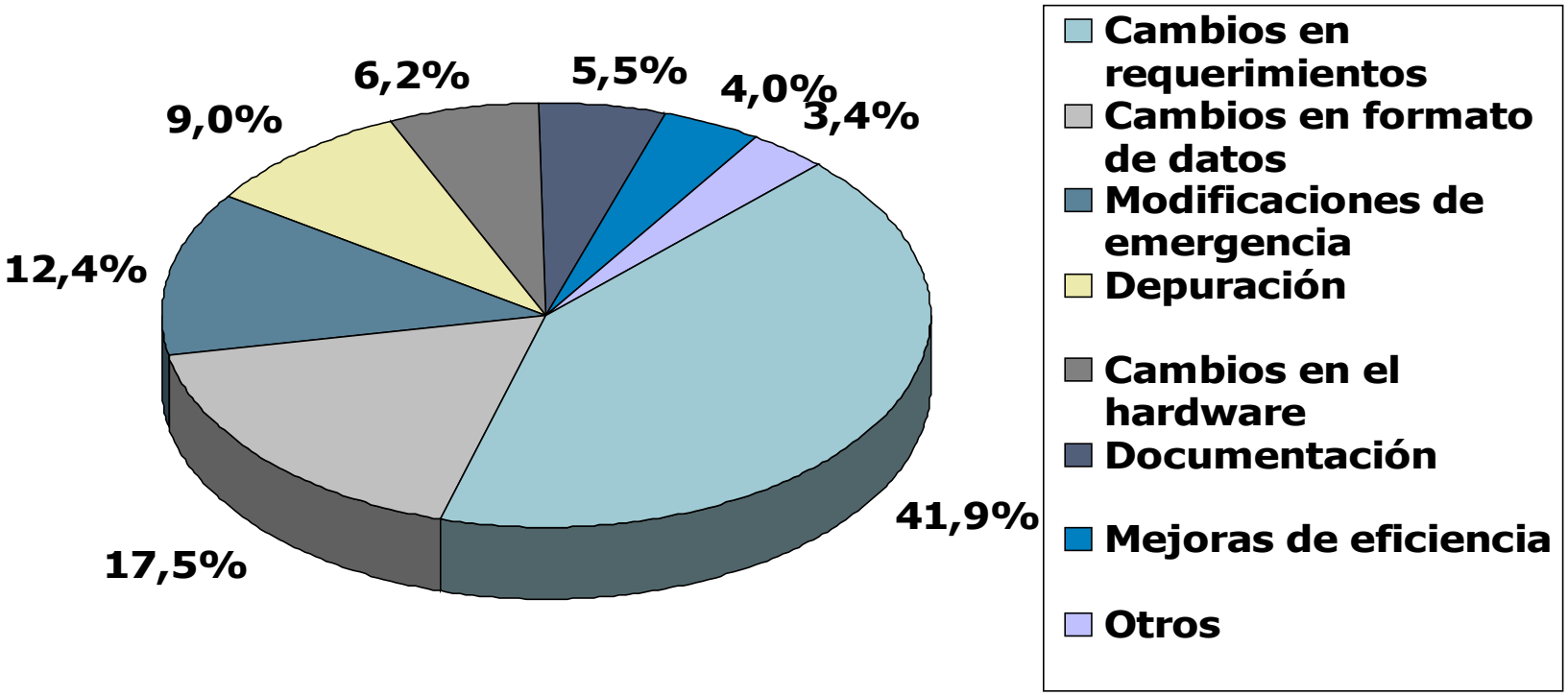
Acerca de la Reutilización

```
// ¿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

Acerca del Mantenimiento

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

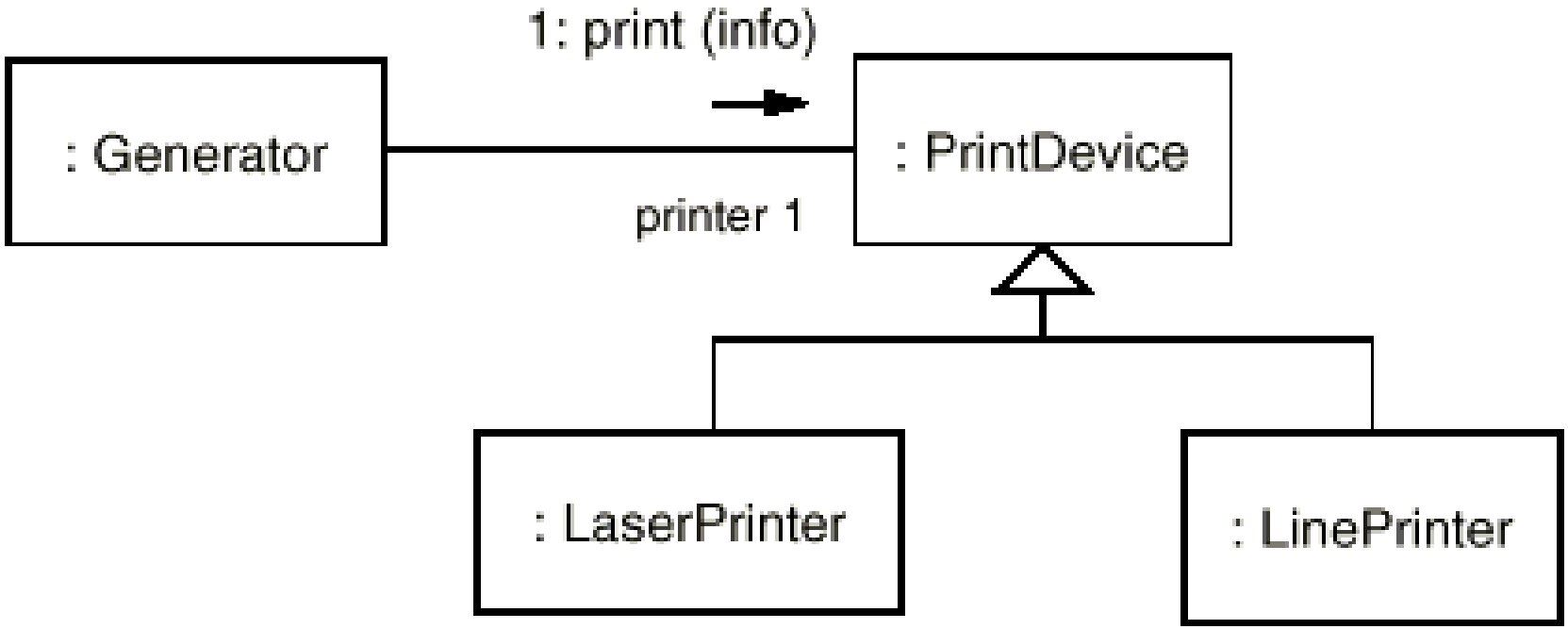


Acerca de la Complejidad

- Grady Booch propone 3 técnicas para enfrentar la complejidad inherente al software (ref: Grady Booch, Object-Oriented Analysis and Design, 1994):
  - Descomposición
  - Abstracción
  - Generalización

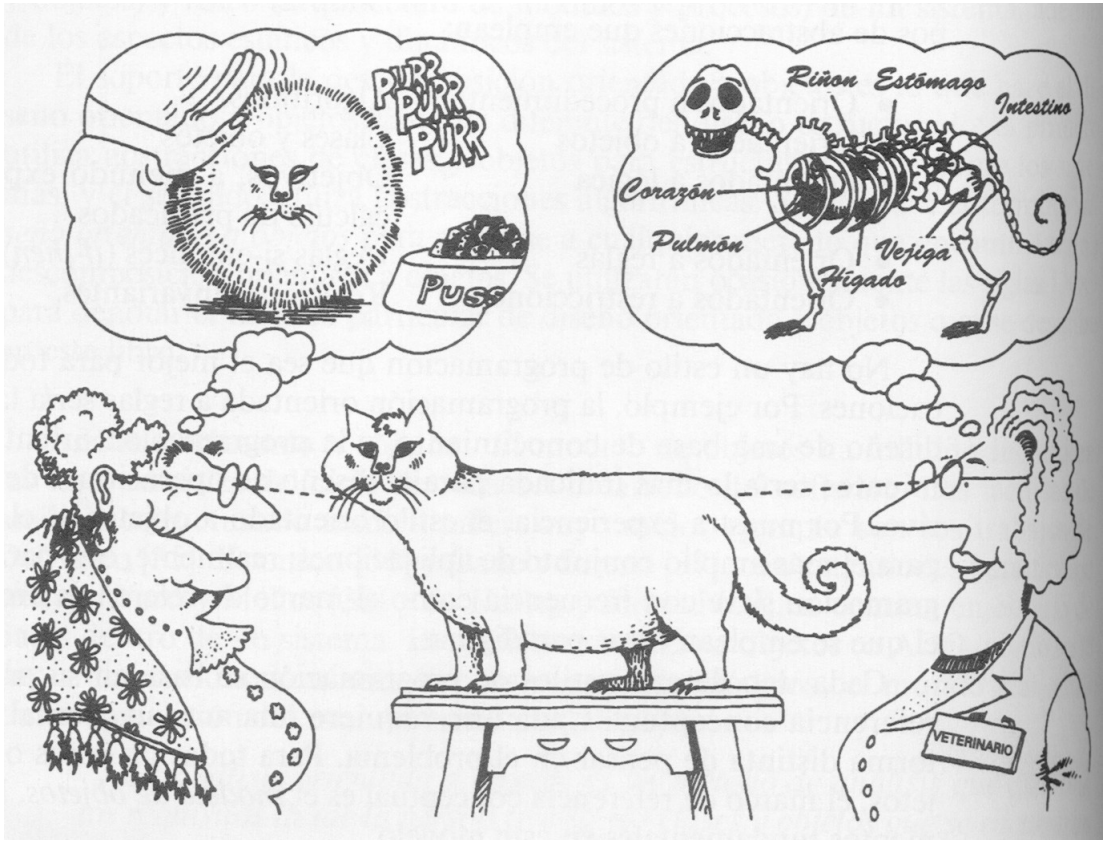
Acerca de la Complejidad

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



Acerca de la Complejidad

- Abstracción: ignorar los detalles no esenciales de un objeto, manteniendo un modelo simplificado



Acerca de la Complejidad

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

Modularidad

- La clave para incrementar la reusabilidad y la mantenibilidad del software, y reducir su complejidad, es la modularidad

- Modularidad de acuerdo a Edward Yourdon:
  - alta cohesión
  - bajo acoplamiento

- OOP / OOD:
  - incorpora el concepto de clase, incrementando la cohesión
  - reduce el acoplamiento:
    - ocultamiento de información
    - dependencia de interfaces

Programación Procedural

- El foco de la programación procedural es dividir una tarea de programación en un conjunto de estructuras de datos y rutinas
- Lenguajes de programación procedural:
  - Algol
  - Basic
  - C
  - Fortran
  - Pascal

Programación Orientada a Objetos

- Grady Booch:
  - *"La Programación Orientada a Objetos es un método de implementación en el que los programas se organizan como colecciones de objetos, cada uno de los cuales representa una instancia de alguna clase, y cuyas clases pertenecen a una jerarquía"*
  - *"Un objeto tiene estado, exhibe algún comportamiento bien definido, y tiene una identidad única"*

Programación Orientada a Objetos

- 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

Programación Procedural v/s OO

- En la programación procedural, los datos se encuentran separados de las rutinas
- En OOP, los datos y las rutinas se integran en las clases
- OOP entrega herramientas para generar código reutilizable y más fácilmente mantenible

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)

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

Java

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

- Influenciado por C++
- Java es un lenguaje orientado a objetos más puro que C++ y menos puro que Smalltalk
- Soporta clases, encapsulación, herencia simple, polimorfismo, interfaces, garbage collection

Java

- Objetivo inicial: un lenguaje de programación para dispositivos de consumo
- Requerimientos: pequeño, rápido, confiable y portable
- En 1994 se produce la explosión del Web, y Sun advierte que Java es ideal para aplicaciones Internet:
  - Independiente de la plataforma
  - Pequeño
  - Seguro

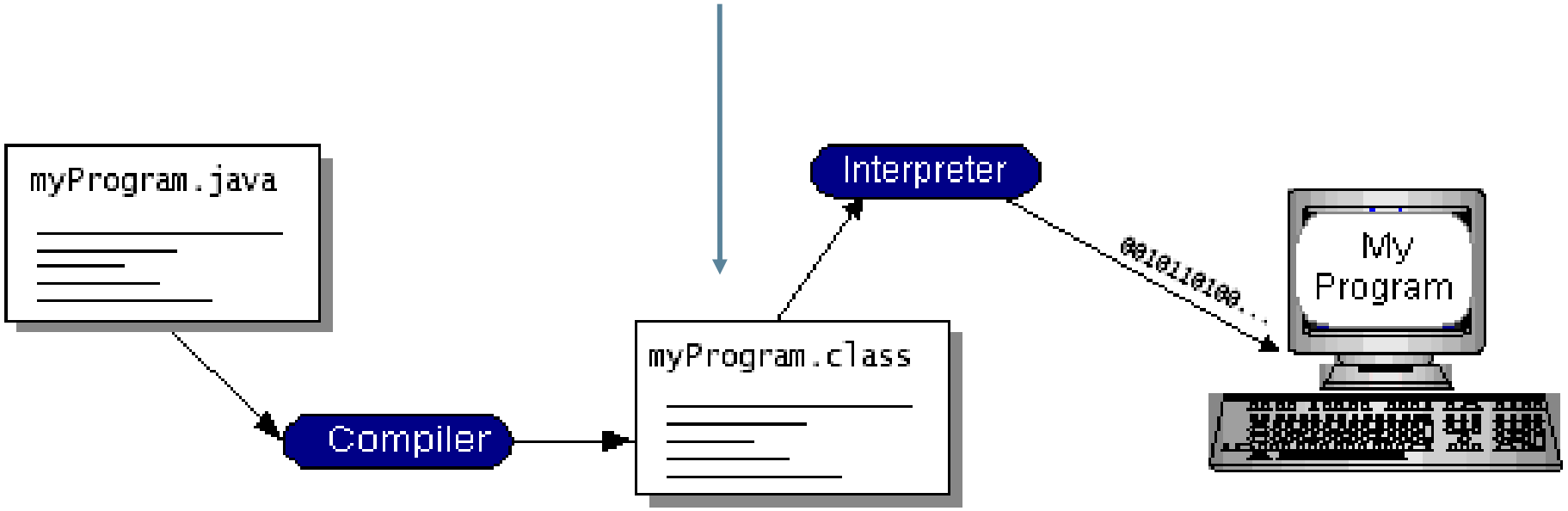
El Lenguaje Java

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

  - Ver http://java.sun.com/docs/white/langenv/

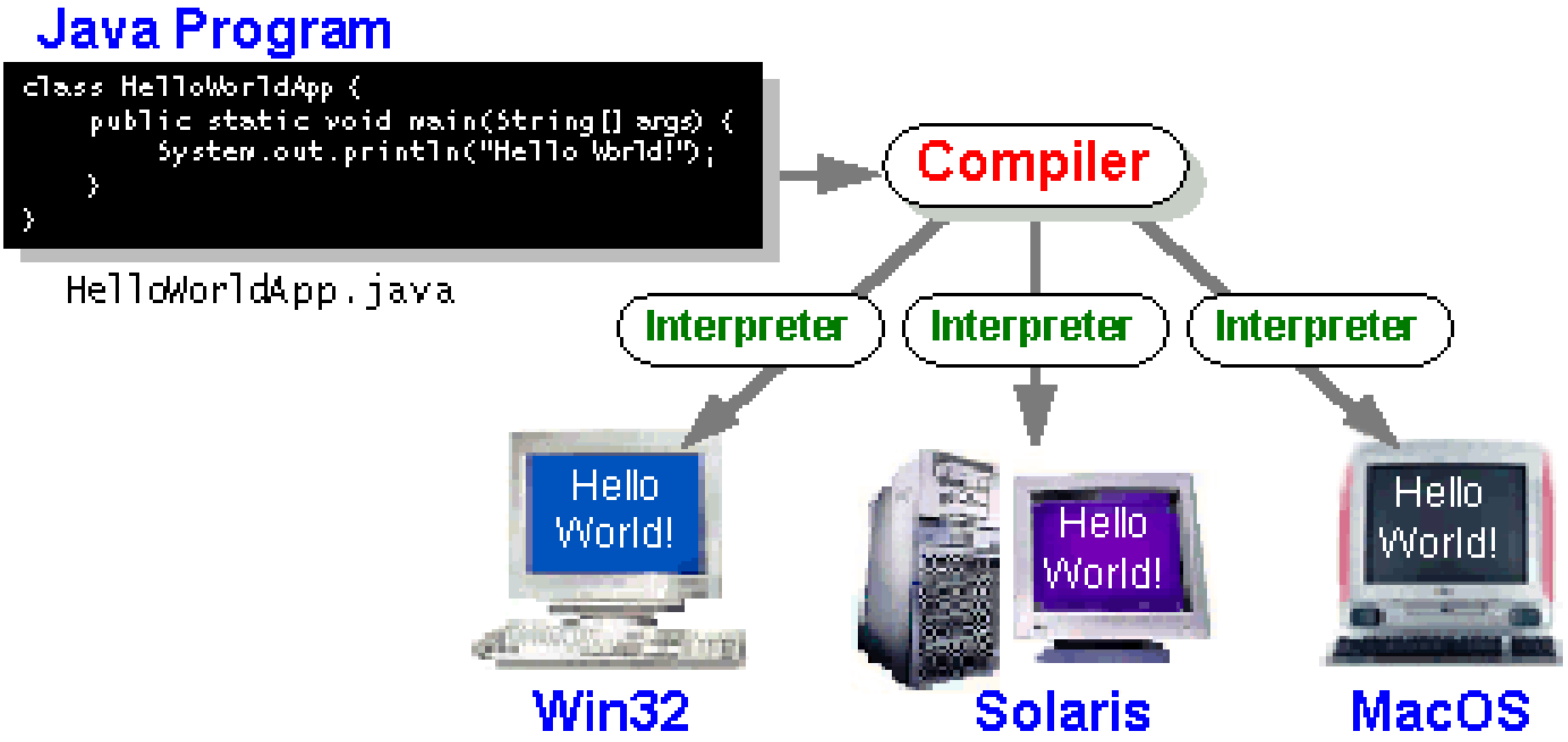
El Modelo Java

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



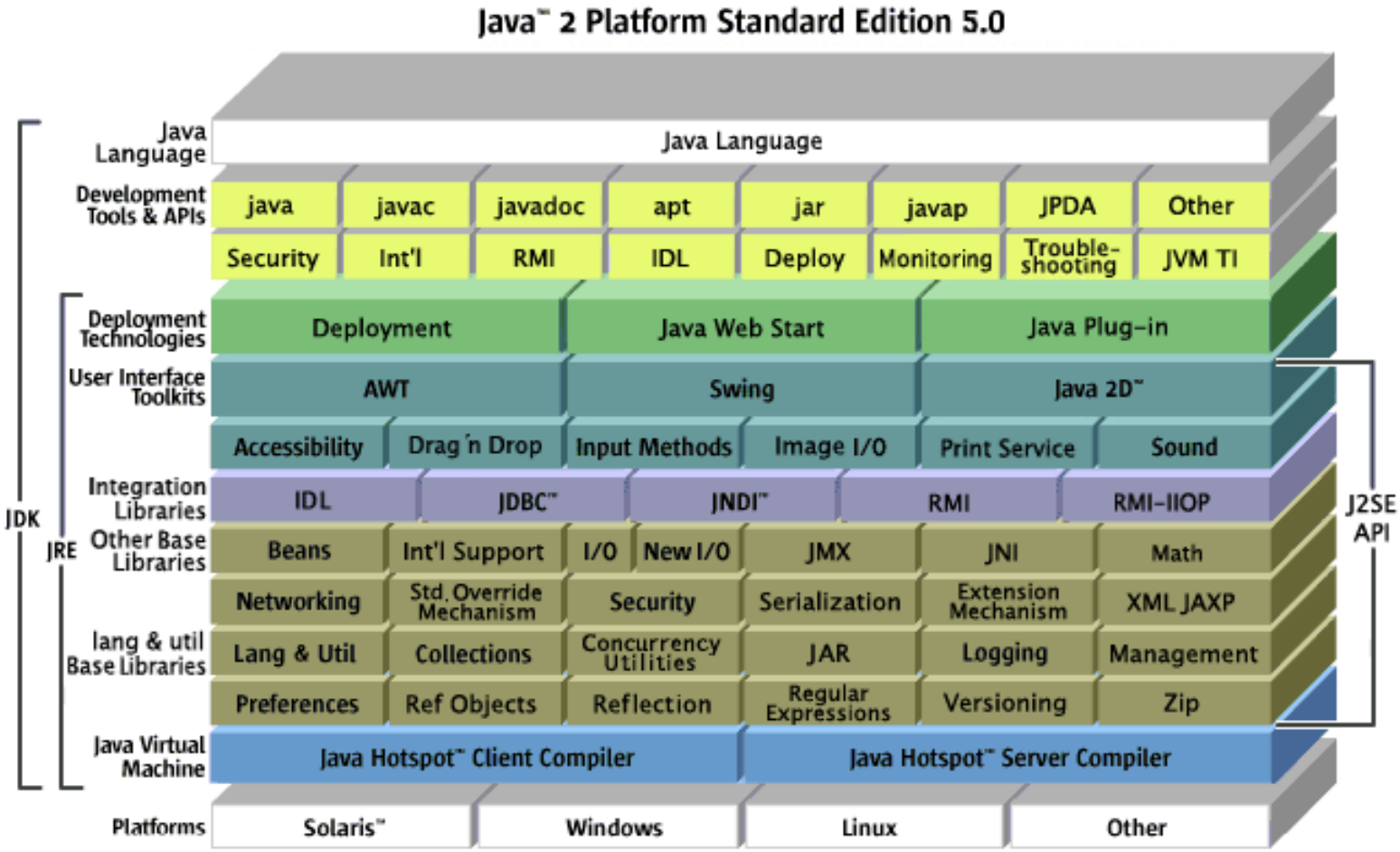
El Modelo Java

- El código bytecode es portable entre diferentes plataformas



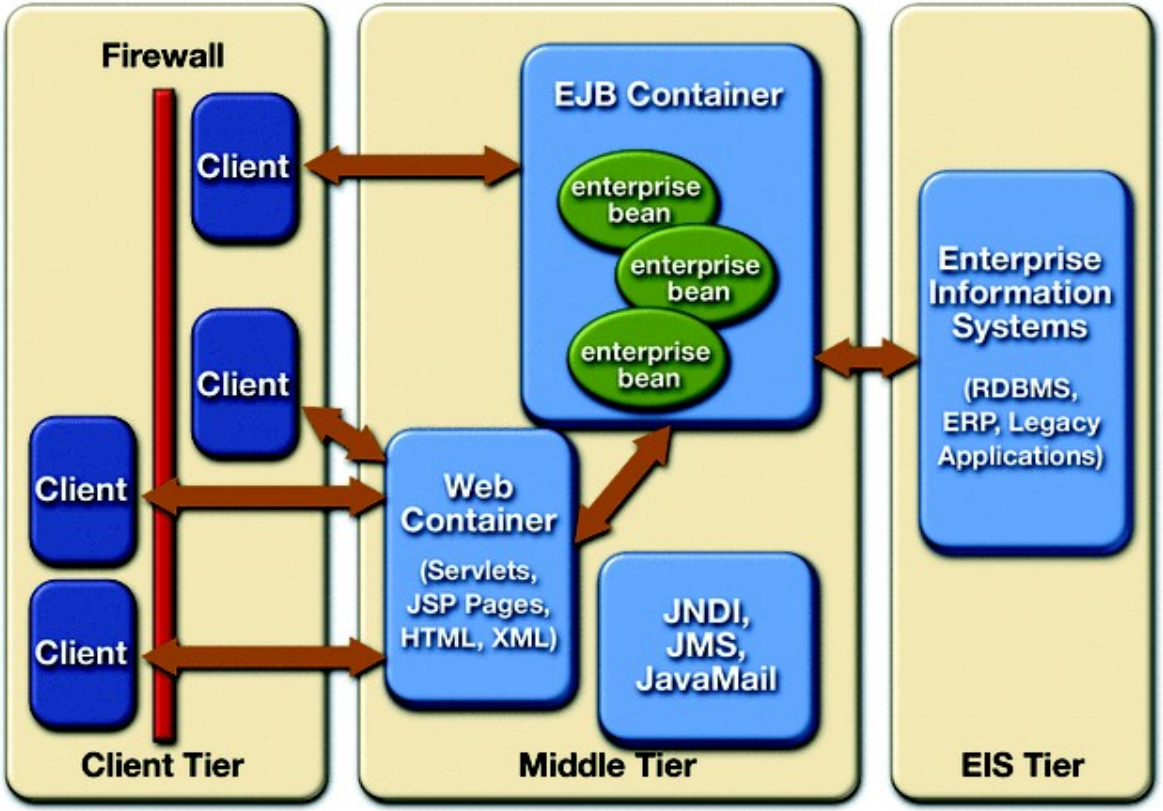
Java Standard Edition

- JSE es la edición estándar de Java, sobre la cual están construidas JME (Mobile Edition) y JEE (Enterprise Edition)



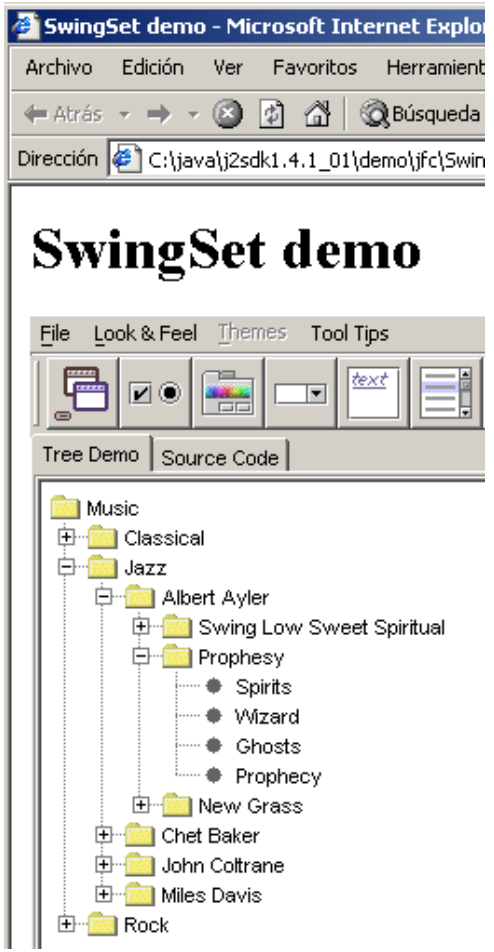
Java Enterprise Edition

- JEE: la edición de Java para la construcción de aplicaciones empresariales



Tipos de Aplicaciones

- Usando Java es posible escribir:
  - Aplicaciones stand-alone
  - Aplicaciones Web (applets, servlets, JSP)
  - Componentes (JavaBeans, Enterprise JavaBeans)
  - Web Services
  - ...



Resumen

- El objetivo de la OOP es aumentar la modularidad del código, mejorando su reusabilidad, y disminuyendo el costo de mantenimiento
- Principales técnicas de OOP: descomposición, abstracción y generalización
- El compilador Java genera bytecode que corre en cualquier sistema que implemente la Java VM
- Java es un lenguaje orientado a objetos, simple, robusto, y seguro