Tema 1: Introducción a la OOP y el lenguaje Java

Programación orientada a objetos

- Aparece a finales de los 60, pero es a principios de los 80 cuando con el lenguaje *Smalltalk* comienza un interés claro hacia este paradigma
- La programación orientada a objetos es el paradigma de programación y análisis y diseño de aplicaciones claramente dominante en la actualidad
- Hoy prácticamente no se concibe un lenguaje de programación sin características de orientación a objetos: *Eiffel*, *C++*, *Java*, C#, etc.

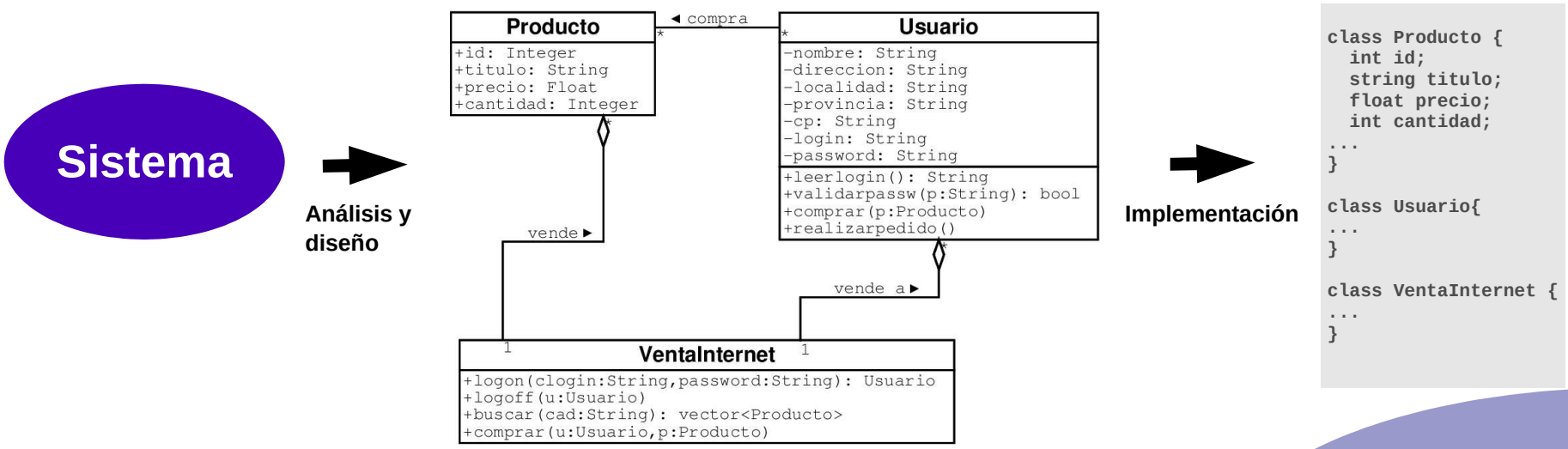
Alan Kay, creador de Smalltalk

Bjarne Stroustrup, creador de C++

- Las ventajas del paradigma OO son múltiples:
  - Es intuitiva, describe un problema en términos similares a los que utiliza la mente humana
  - Permite construir soluciones más seguras y con un mantenimiento más sencillo
  - Fomenta la reutilización y el trabajo en equipo. Escalabilidad de las aplicaciones

- Las características principales del paradigma OO:
  - Incorpora los conceptos de abstracción, ocultación de información y encapsulación heredados de los tipos de datos abstractos
  - Incorpora mecanismos específicos y extremadamente poderosos como son la herencia, el polimorfismo y la ligadura dinámica
  - Admite de manera muy natural en algunos lenguajes el soporte de genericidad (patrones) y la definición de operadores. No soportados por Java

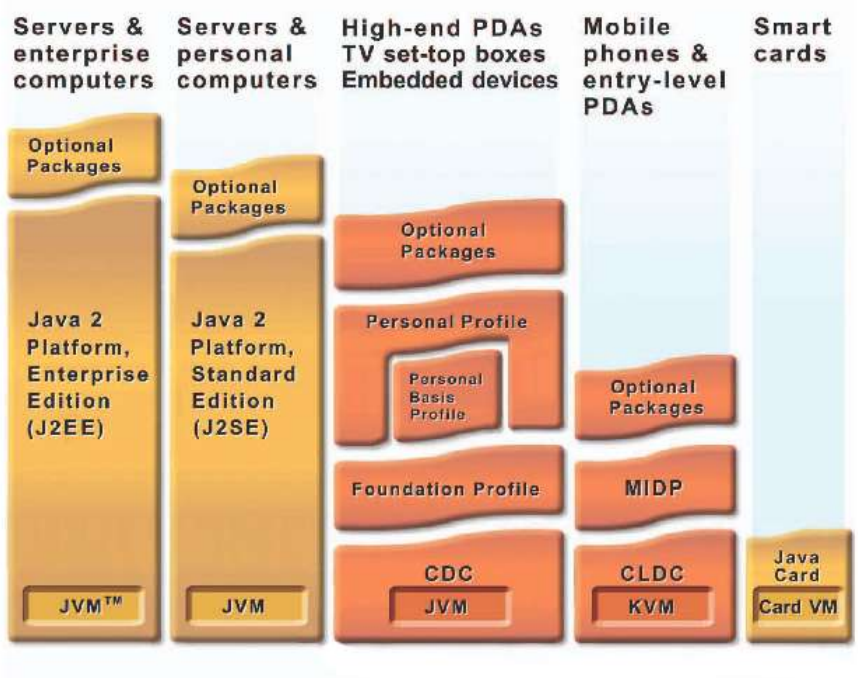
- El desarrollo siguiendo el enfoque orientado a objetos es un proceso integral que incluye métodos específicos de análisis y diseño, notación gráfica (UML) y lenguajes de programación orientados a objetos
- Las soluciones obtenidas durante la fase de análisis y diseño no son específicas para ningún lenguaje de programación orientado a objetos



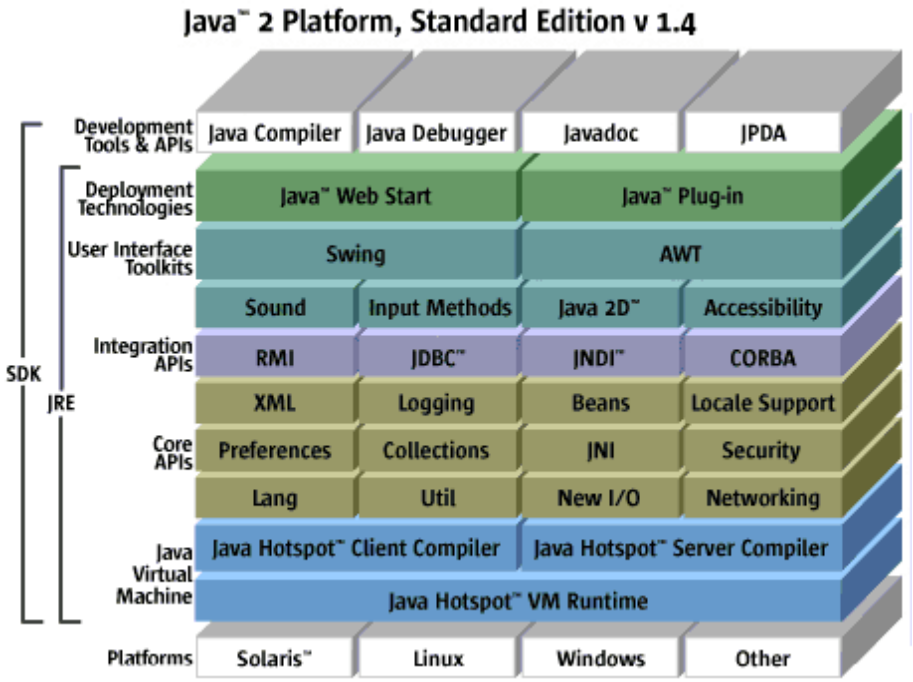
El lenguaje Java

- Desarrollado en los laboratorios de Sun, es uno de los lenguajes de programación orientado a objetos que mayor repercusión ha tenido en los últimos años
  - Basado en C++ pero simplificado, mucho más fácil de usar, de más alto nivel y menos propenso a errores
  - Amplísima biblioteca estándar de clases predefinidas
  - Las aplicaciones Java pueden ser ejecutadas indistintamente en cualquier plataforma sin necesidad de recompilación
  - Gestión avanzada de memoria mediante un recolector de basura
  - Gestión avanzada de errores, tanto en tiempo de compilación como de ejecución
  - Distribuido y multihebra
  - Lenguaje abierto. Kits de desarrollo y documentación gratuitos en la red

- Existen distintas “ediciones” de Java para el desarrollo de aplicaciones en distintos ámbitos:
  - Aplicaciones de propósito general (J2SE)
  - Aplicaciones de gestión en entornos empresariales (J2EE)
  - Aplicaciones para teléfonos móviles, PDAs y otros dispositivos electrónicos que permitan aplicaciones empotradas (J2ME)



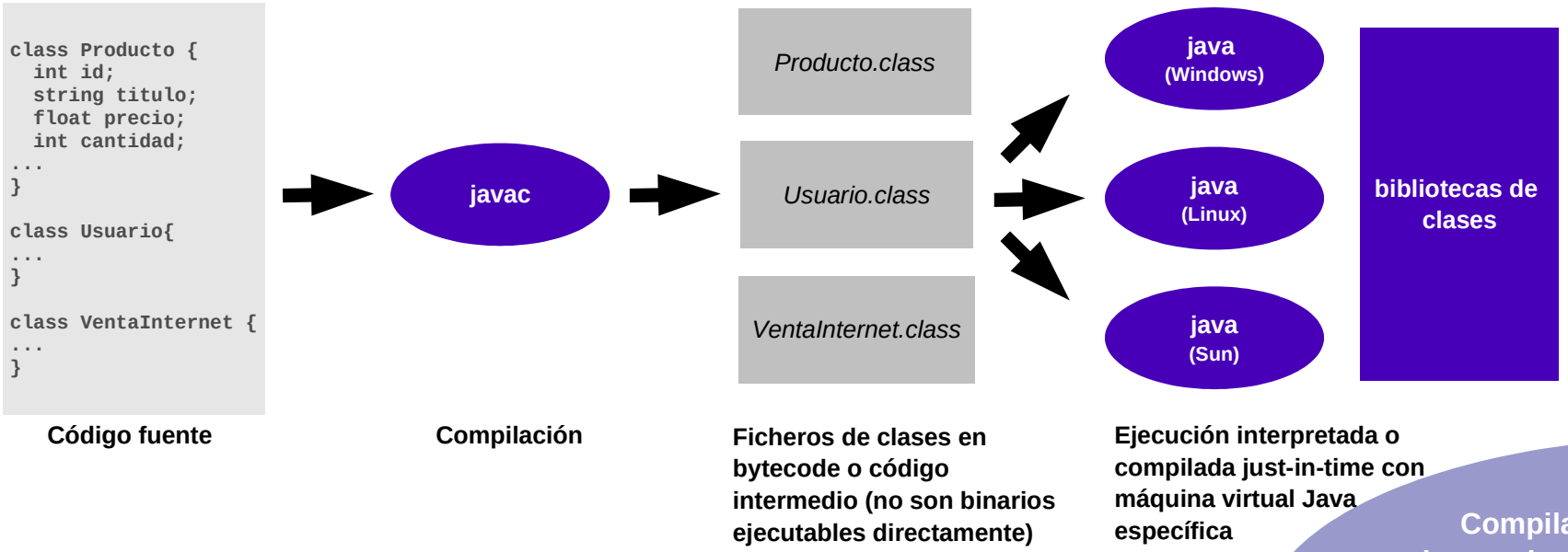
- La más utilizada es sin duda la edición estándar (J2SE). Los ejemplos de código Java que veremos a lo largo de los siguientes capítulos pertenecen a esta edición, con la excepción del capítulo de programación en Internet (J2EE)



- Existen dos kits diferentes que pueden ser descargados de la página oficial de Sun:
  - El Java Development Kit (JDK) permite desarrollar y ejecutar aplicaciones Java
  - El Java Runtime Environment (JRE) permite únicamente la ejecución
- J2SE incluye bibliotecas muy extensas y completas, que permiten la implementación de casi cualquier tipo de aplicación
  - Seguridad
  - EEDDs
  - Componentes (JavaBeans)
  - Internacionalización
  - E/S
  - XML
  - Redes y acceso a Internet
  - Programación distribuida (RMI, CORBA)
  - Matemática de precisión arbitraria
  - Sonido
  - Interfaz de usuario (AWT, SWING)
  - Gráficos 2D
  - Manipulación, carga y descarga de imágenes
  - Impresión
  - Acceso a bases de datos (JDBC)
  - Gestión de preferencias y configuraciones

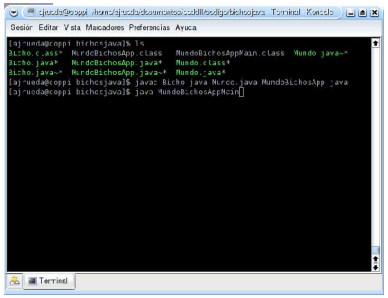
Compilación, bytecode y JVM

- Java sigue un esquema no tradicional de compilación / ejecución:
  - La compilación genera un ejecutable en bytecode o código intermedio independiente
  - Para su ejecución se requiere un JRE específico de la plataforma. El JRE está formado por una máquina virtual java (JVM) y las librerías de clases.
  - La JVM interpreta el bytecode o realiza su compilación just-in-time para que su ejecución sea más eficiente

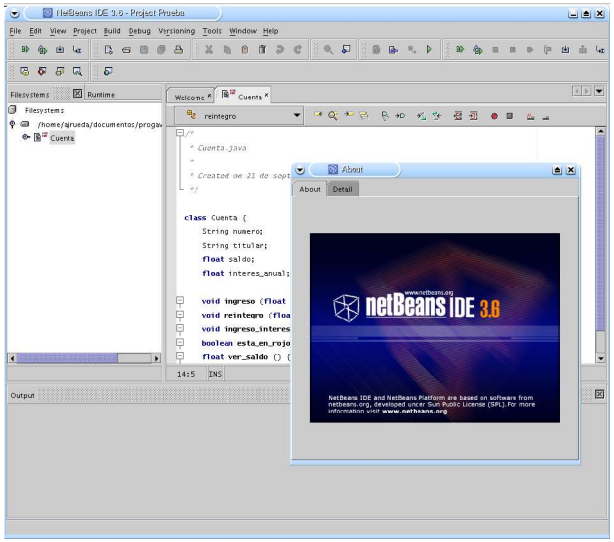


- Ventajas de este sistema:
  - Se compila la aplicación una única vez y los ejecutables en bytecode obtenidos son válidos para cualquier plataforma. El código fuente queda a salvo
  - Es muy robusto. La máquina virtual Java es capaz de detectar y notificar gran cantidad de errores durante la ejecución de la aplicación (como accesos a elementos fuera de un vector)
  - El recolector de basura no ocupa espacio en el ejecutable, ya que viene integrado en la JVM
  - Los ejecutables son pequeños porque las librerías de clases vienen proporcionadas junto a la JVM en el JRE de la plataforma concreta
- Inconvenientes:
  - Velocidad. Evidentemente la interpretación o incluso compilación just-in-time del bytecode produce aplicaciones más lentas que en el caso de la ejecución directa de un binario. El recolector de basura puede suponer una sobrecarga adicional al procesador
  - La generalidad tiene como inconveniente que no se aprovecha totalmente la potencia de la máquina y del sistema operativo. Por ejemplo, el aspecto de una aplicación Java puede resultar simple y poco atractivo en comparación con las aplicaciones nativas

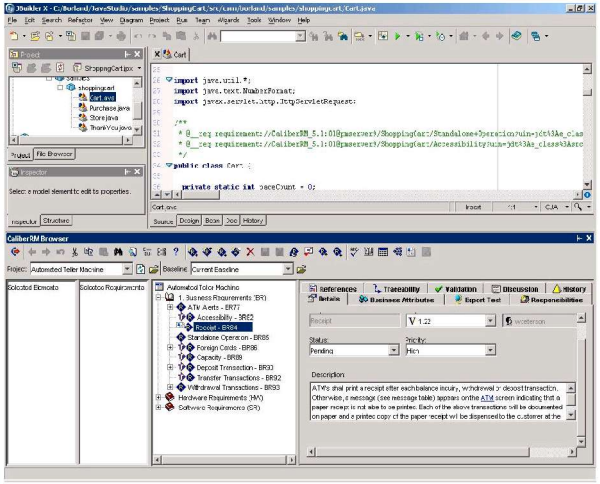
Entornos de desarrollo Java



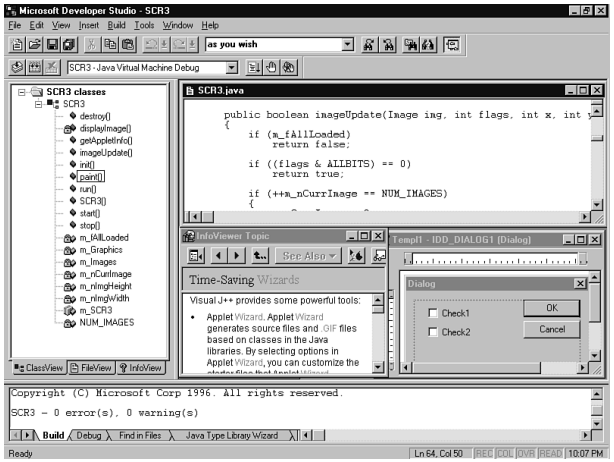
- JDK. El kit de desarrollo básico proporcionado por Sun. Es lo mínimo que se necesita para desarrollar. Útil si se necesita compilar aplicaciones Java de manera esporádica o en general para programadores con espiritu “espartano”. Puede bajarse gratuitamente para cualquier plataforma de java.sun.com



- netBeans. Entorno integrado de desarrollo Java de Sun, realizado íntegramente en Java (y por tanto multiplataforma). Consume bastantes recursos. Permite diseñar ventanas, escribir código, compilar, ejecutar etc. Requiere un JDK instalado. Puede obtenerse gratuitamente de www.netbeans.org



- Borland JBuilder. Excelente entorno integrado de desarrollo Java de Borland. Al igual que Netbeans, también está realizado íntegramente en Java. Existen versiones limitadas que pueden bajarse de www.borland.com



- Microsoft Visual J++. Uno de los más populares, aunque las aplicaciones obtenidas pueden presentar problemas de compatibilidad con el SDK oficial de Java, por el uso de librerías especificas de Microsoft. Permite construir aplicaciones Java dentro de la plataforma .NET.

- Otros: Eclipse, IBM WebSphere, Oracle JDeveloper, etc.

Java vs otros lenguajes OO

	Java	C#	C++	Eiffel	Smalltalk
Año Aparición	1995	2000	1985	1985	1970
Sintaxis	ins. C++	ins. Java	ins. C	ins. Pascal	original
Difusión	Amplia	Amplia	Amplia	Limitada	Limitada
Librería de clases	Muy Amplia	Muy Amplia	Escasa	Amplia	Amplia
Recolector basura	Si	Si	No	Si	Si
Manejo objetos	Dinámico	Dinámico	Est./Din.	Est./Din.	Dinámico
Tipo ejecutable	Bytecode	IL Code	binario	binario	Byte codes
Ejecución	mediante JVM	mediante CLR	directa	directa	mediante SVM
Velocidad ejecutable	media	media	muy alta	alta	baja
Soporte excepciones	Si	Si	Si	Si	Si
Herencia múltiple	No	No	Si	Si	No
Soporte operadores	Muy Limitado	Limitado	Si	Si	No
Soporte plantillas	No	No	Si	Si	No