Lenguaje Unificado de Modelado

UML

1

Introducción
Modelado de SW

2

Para qué modelar

- ¿Cómo se define una empresa de SW exitosa?
- El modelado es una parte central de todas las actividades que conducen a la producción de BUEN software.

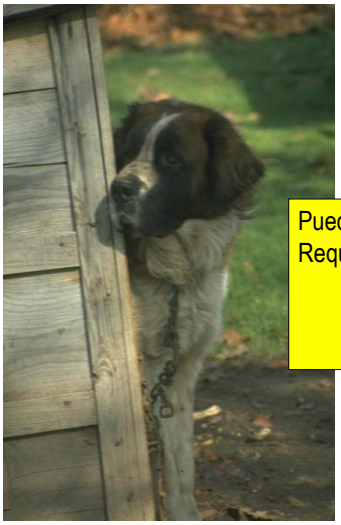
3

Para qué modelar

- Se construyen modelos para:
  - Comunicar la estructura deseada y el comportamiento del sistema
  - Para visualizar y controlar la arquitectura
  - Para comprender mejor el sistema que se está construyendo
  - Para controlar riesgos

4

Construcción de una casa para "fido"



Puede hacerlo una sola persona
Requiere:
- Modelado mínimo
- Proceso simple
- Herramientas simples

5

Construcción de una casa

Construida eficientemente y en un tiempo razonable por un equipo
Requiere:
- Modelado
- Proceso bien definido
- Herramientas más sofisticadas

6

Construcción de un rascacielos

¿Cómo se construye?

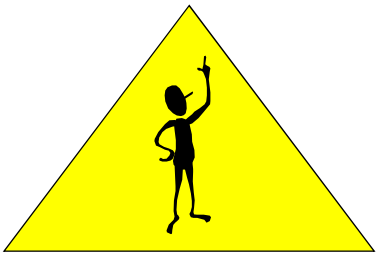
- La idea es construir un rascacielos, pero se enfrenta el problema como si se enfocara en la construcción de la caseta de un perro.
- Los equipos de desarrollo recurren a lo que se supone se debe hacer: generar toneladas de líneas de código
- El modelado es una técnica de ingeniería probada y bien aceptada

¿Qué es un modelo?

- Es una simplificación de la realidad
- Se construyen modelos de sistemas complejos por que no se puede comprender el sistema en su totalidad
- En general un modelado formal no se realiza en las empresas
- Se provee entonces un lenguaje común para modelar

9

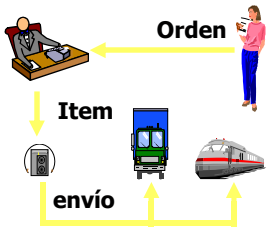
Claves en Desarrollo de SI



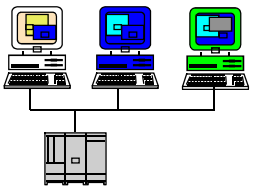
10

Abstracción - Modelado Visual (MV)

"El modelado captura las partes esenciales del sistema"



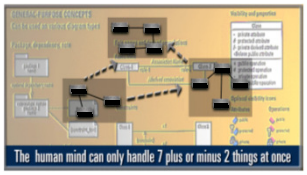
Proceso de Negocios

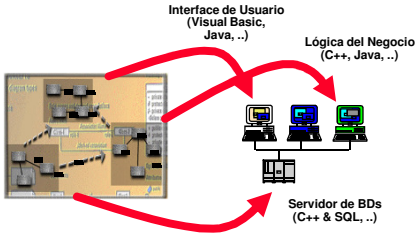


Sistema Computacional

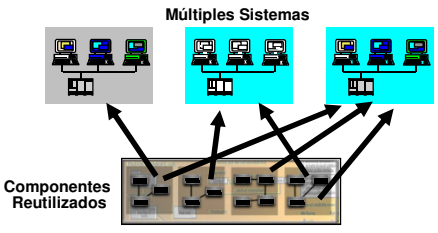
II. Notación (Visual) - Beneficios

Manejar la complejidad





"Modelar el sistema independientemente del lenguaje de implementación"



Promover la Reutilización

Principios del modelado

1. La elección de qué modelos crear tiene una profunda influencia sobre cómo se acomete un problema y cómo se da forma a una solución
2. Todo modelo puede ser expresado a diferentes niveles de precisión
3. Los mejores modelos están ligados a la realidad
4. Un único modelo no es suficiente. Cualquier sistema no trivial se aborda mejor a través de un pequeño conjunto de modelos casi independientes

13

Breve Revisión: UML

14

¿Qué es UML?

- UML = Unified Modeling Language
- Un lenguaje de propósito general (visualizar, especificar, construir y documentar sistemas) para el modelado orientado a objetos
-
- UML combina notaciones provenientes desde:
  - Modelado Orientado a Objetos
  - Modelado de Datos
  - Modelado de Componentes
  - Modelado de Flujos de Trabajo (Workflows)

15

¿Dónde puede utilizarse?

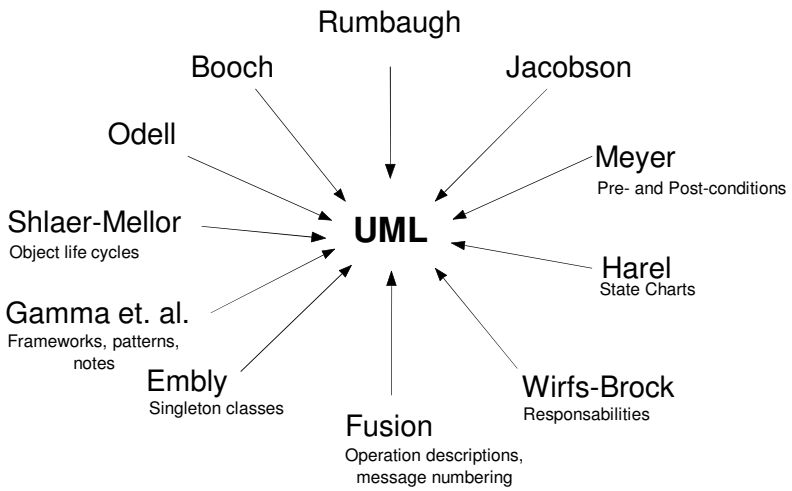
- Sistemas de información de empresas
- Bancos y servicios financieros
- Telecomunicaciones
- Transporte
- Defensa/ industria aeroespacial
- Comercio
- Electrónica médica
- Ámbito científico
- Servicios distribuidos basados en la Web

16

Participantes en UML 1.0

- Rational Software (Grady Booch, Jim Rumbaugh y Ivar Jacobson)
- Digital Equipment
- Hewlett-Packard
- i-Logix (David Harel)
- IBM
- ICON Computing (Desmond D'Souza)
- Intellicorp and James Martin & co. (James Odell)
- MCI Systemhouse
- Microsoft
- ObjecTime
- Oracle Corp.
- Platinium Technology
- Sterling Software
- Taskon
- Texas Instruments
- Unisys

UML "integra" enfoques OO



Inconvenientes en UML

- Definición del proceso de desarrollo usando UML. UML no es una metodología
- Falta integración con respecto de otras técnicas tales como patrones de diseño, interfaces de usuario, documentación, etc.
- Ejemplos aislados
- "Monopolio de conceptos, técnicas y métodos en torno a UML"

Perspectivas de UML

- UML será el lenguaje de modelado orientado a objetos estándar predominante los próximos años
- Razones:
  - Participación de metodólogos influyentes
  - Participación de importantes empresas
  - Aceptación del OMG como notación estándar
- Evidencias:
  - Herramientas que proveen la notación UML
  - "Edición" de libros
  - Congresos, cursos etc.

Breve Tour por UML

21

Diagramas de UML

- Diagrama de Casos de Uso
- Diagrama de Clases

Diagramas de Comportamiento

- Diagrama de Estados
- Diagrama de Actividad

Diagramas de Interacción

- Diagrama de Secuencia
- Diagrama de Colaboración

Diagramas de implementación

- Diagrama de Componentes
- Diagrama de Despliegue

22

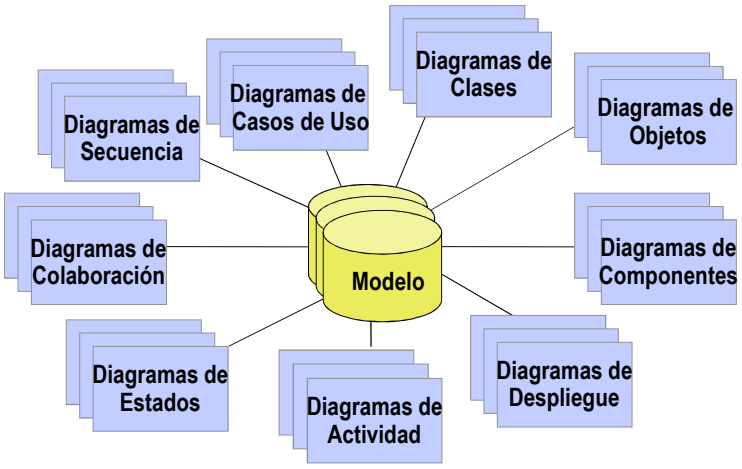
Diagramas de UML

Diagramas de administración de Modelo

- Diagramas de Paquetes
- Diagramas de Subsistemas
- Diagramas de Modelos

... Diagramas de UML

Los diagramas expresan gráficamente partes de un modelo



Diagramas

- Clases: Muestra un conjunto de clases, interfaces y colaboraciones. Vista de Diseño estática
- Objetos: Conjunto de objetos y sus relaciones. Diseño estática
- Casos de uso: casos de uso, con actores y sus relaciones. Comportamiento del sistema

25

Diagramas

- Secuencia: resalta la ordenación temporal de los mensajes. Vista dinámica
- Colaboración: resalta la organización estructural de los objetos que envían y reciben mensajes.
- Estados: Consta de estados, transiciones, eventos y actividades. Vista dinámica

26

Diagramas

- Actividades: Tipo especial de diagramas de estado que muestra el flujo de actividades dentro de un sistema. Vista dinámica
- Componentes: Muestra la organización y las dependencias entre un cjto. De componentes. Vista de implementación estática.
- Despliegue: Configuración de nodos de procesamiento en tiempo de ejecución y los componentes que residen en ellos.

27

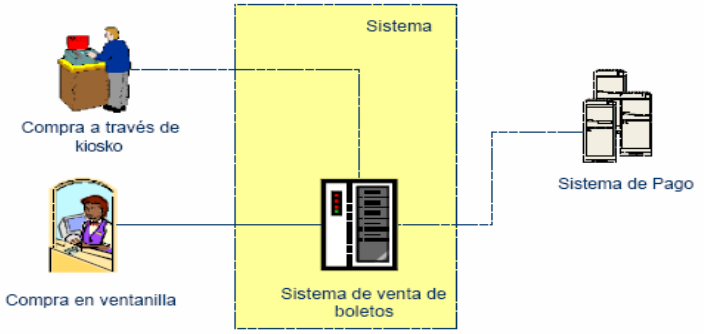
Vistas

Modelo	Vista	Diagramas asociados
Estructural	Estática	Diagrama de clase
	Caso de uso	Diagrama de casos de uso
	Implementación	Diagrama de componente
	Estructura	Diagrama de estructura (deployment view)
Dinámico	Maquina de estado	Diagrama de estado
	Actividad	Diagrama de actividad
	Interacción	Diagrama de secuencia
		Diagrama de colaboración

28

Dominio de ejemplo
Ventas de entradas en un teatro

Un teatro tiene un sistema computacional de venta de entradas. Las entradas pueden ser vendidas de dos maneras, a través de un kiosko o directamente en la ventanilla del teatro. El cliente puede comprar la entrada para una función en particular o para una temporada completa.



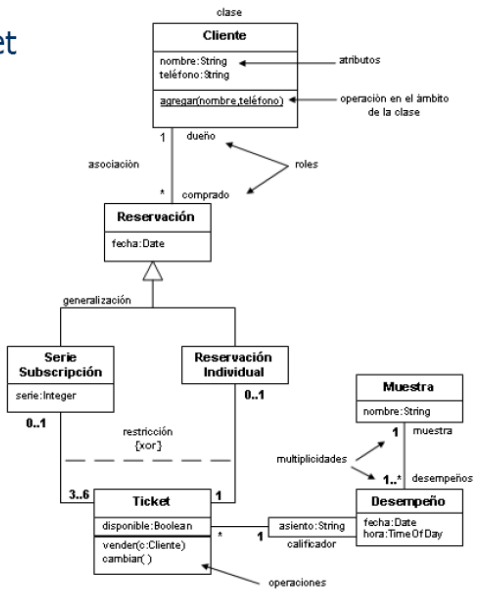
29

Vistas en UML
Vista estática

- Modela los conceptos en el dominio de la aplicación.
- Diagrama asociado: Diagrama de clases
- Clases: Descripción de un concepto del dominio de la aplicación.

30

Venta de ticket



31

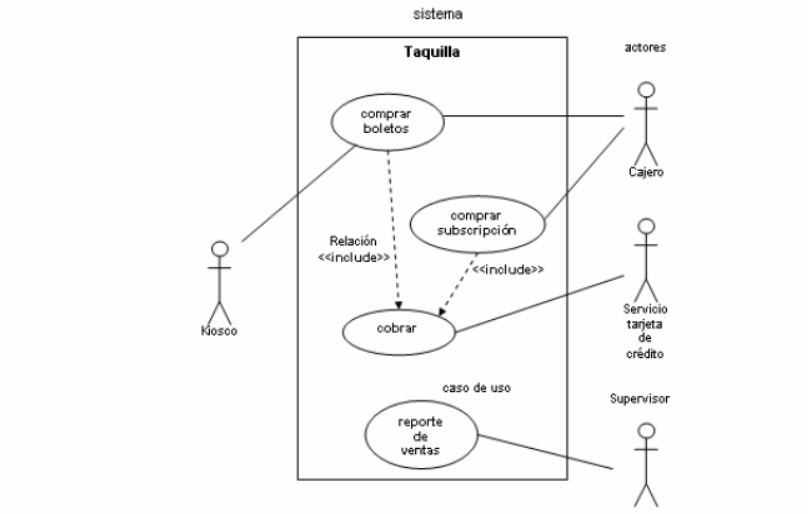
Vistas en UML

Vista de casos de uso

- Modela la funcionalidad del sistema percibida por usuarios externos.
- Diagrama asociado: Diagrama de casos de uso
- Actor: Abstracción de entidades que interactúan directamente con el sistema. Puede ser un sistema externo o personas.

32

- Casos de uso para la oficina de ventas



33

Vistas en UML
Vista de interacción

- Describe la secuencia de intercambio de mensajes entre entidades.
- Muestra el flujo de control entre diferentes objetos.
- Dos diagramas asociados:
  - Diagrama de secuencia
  - Diagrama de colaboración

34

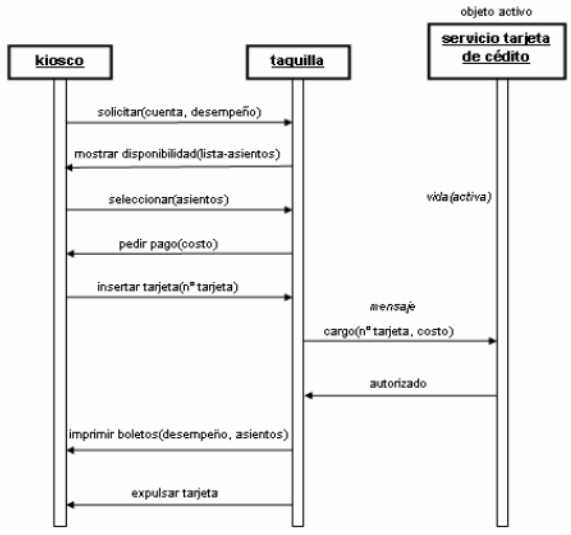
Vistas en UML

Vista de interacción: Diagrama de secuencia

- Muestra mensajes entre objetos activos a través de líneas de tiempo.
- Un uso de el caso de uso es mostrar la secuencia de eventos de un caso de uso específico.

35

- Diagrama de secuencia para la compra de entradas



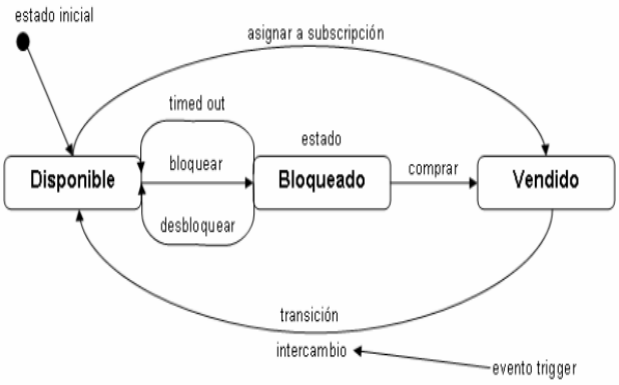
36

Vistas en UML
Vista de maquina de estado

- Sirve para modelar los posibles estados de un objeto o clase.
- Diagrama asociado: Diagrama de estados
- Estados conectados por transiciones
- Estados modela un período de tiempo durante la vida del objeto durante el cual se satisfacen ciertas restricciones.
- Un evento gatilla una transición y genera un nuevo estado.

37

- Diagrama de estado para una entrada.



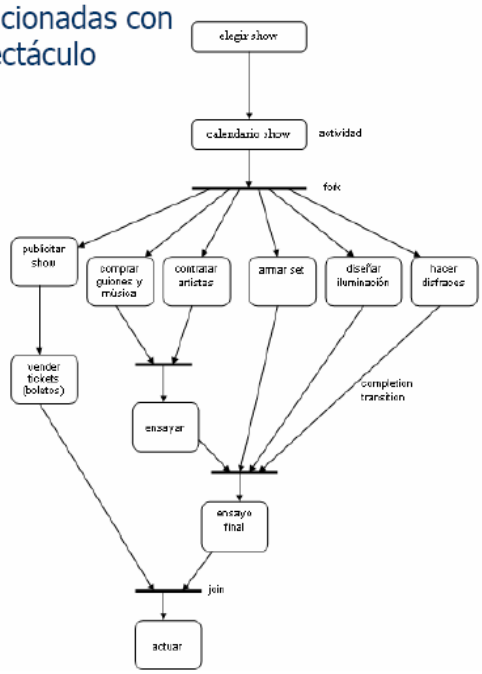
38

Vistas en UML
Vista de actividad

- Muestra las actividades computacionales involucradas en la realización de cierta operación.
- Diagrama asociado: Diagrama de actividad.
- Pueden ser utilizados para modelar workflows del negocio o del SW.

39

- Actividades relacionadas con montar un espectáculo



Vistas en UML
Vista física

- Modela la estructura de la aplicación en términos de componentes y nodos.
  - Componente: Agrupación de alto nivel de clases, documentos e ítems relacionados según cierto dominio.
  - Nodo: representa alguna pieza de unidad de cómputo, por lo general es HW.

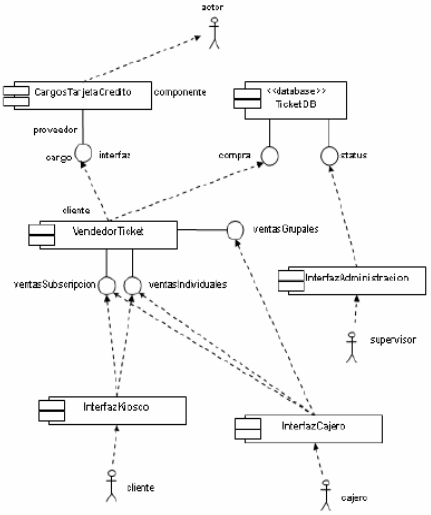
41

Vistas en UML
Vista física

- Existen dos tipos de vistas físicas:
- Vista de implementación: modela los componentes del sistema.
  - Diagrama asociado: Diagrama de componentes
- Vista de emplazamiento: muestra las relaciones físicas entre los componentes de software y de hardware de un sistema.
  - Diagrama asociado: Diagrama de emplazamiento

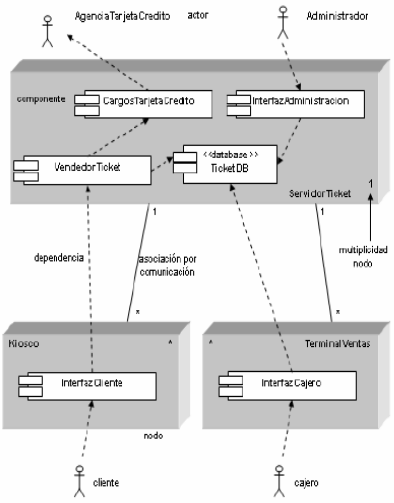
42

Diagrama de componentes para la oficina de ventas



43

Diagrama de emplazamiento para la oficina de ventas.



44

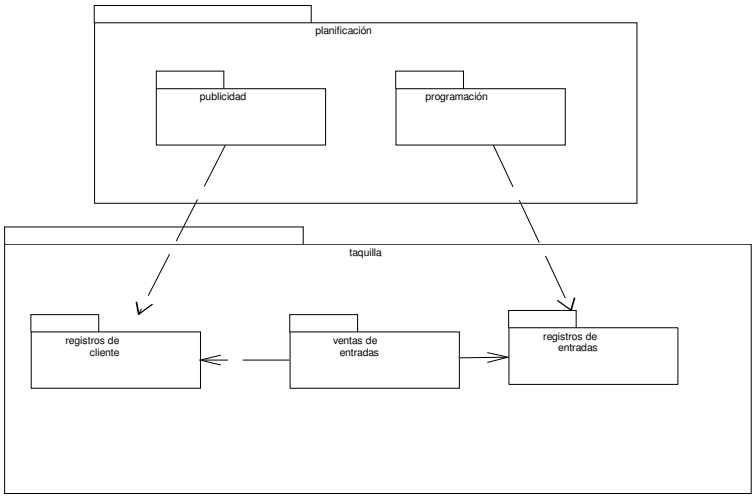
Vistas en UML

Vista de administración de modelo

- Modela la organización del modelo, en termino de paquetes.
- Ejemplo: descomposición de la puesta en marcha de una obra.

45

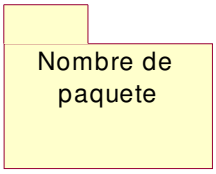
Paquetes



46

Paquetes en UML

- Los paquetes ofrecen un mecanismo general para la organización de los modelos/subsistemas agrupando elementos de modelado
- Se representan gráficamente como:



47

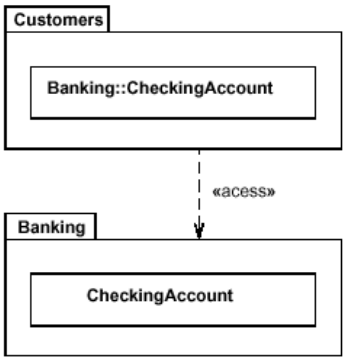
… Paquetes en UML

- Cada paquete corresponde a un submodelo (subsistema) del modelo (sistema)
- Un paquete puede contener otros paquetes, sin límite de anidamiento pero cada elemento pertenece a (está definido en) sólo un paquete
- Una clase de un paquete puede aparecer en otro paquete por la importación a través de una relación de dependencia entre paquetes

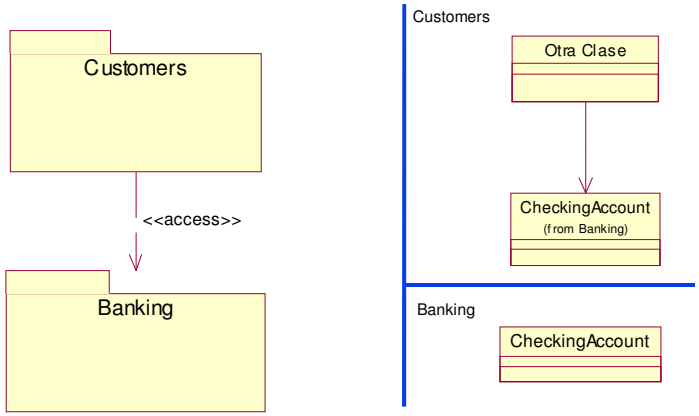
48

… Paquetes en UML

- Todas las clases no son necesariamente visibles desde el exterior del paquete, es decir, un paquete encapsula a la vez que agrupa
- El operador "::" permite designar una clase definida en un contexto distinto del actual



…Paquetes en Rational Rose



… Paquetes en UML

Para importar clases de un paquete se usa el comando **import**.
Se puede importar una clase individual

import java.awt.Font;

o bien, se puede importar las clases declaradas públicas de un paquete completo, utilizando un arterisco (*) para reemplazar los nombres de clase individuales.

import java.awt.*;

… Paquetes en UML

Paquete estándar (java)	**Descripción**
java.applet	Contiene las clases necesarias para crear applets que se ejecutan en la ventana del navegador
java.awt	Contiene clases para crear una aplicación GUI independiente de la plataforma
java.io	Entrada/Salida. Clases que definen distintos flujos de datos
java.lang	Contiene clases esenciales, se importa impícitamente sin necesidad de una sentencia **import**.
java.net	Se usa en combinación con las clases del paquete java.io para leer y escribir datos en la red.
java.util	Contiene otras clases útiles que ayudan al programador