Sistemas Operativos Orientados a Objetos

Conceptos generales de orientación a objetos

El paradigma de la orientación a objetos está basado en cuatro conceptos fundamentales:

- Objetos
- Tipos de datos abstractos (Encapsulamiento y ocultamiento de información)
- Herencia (De atributos y comportamiento)
- Polimorfismo

La programación orientada a objetos utiliza como bloque principal de construcción al objeto, a diferencia de la programación imperativa, cuyo principal bloque de construcción es la función o subrutina.

Un objeto tiene tres características principales:

1. Una identidad única (nombre o id)
2. Un estado (Representado por los valores de sus atributos internos)
3. Un comportamiento (Conjunto de atributos y métodos para accesarlos)

Un objeto se comunica con otro típicamente por paso de mensajes, y pertenece a una clase, que es el conjunto de atributos y comportamiento que define un objeto.

La abstracción de datos es la encapsulación del estado y el comportamiento de un objeto.

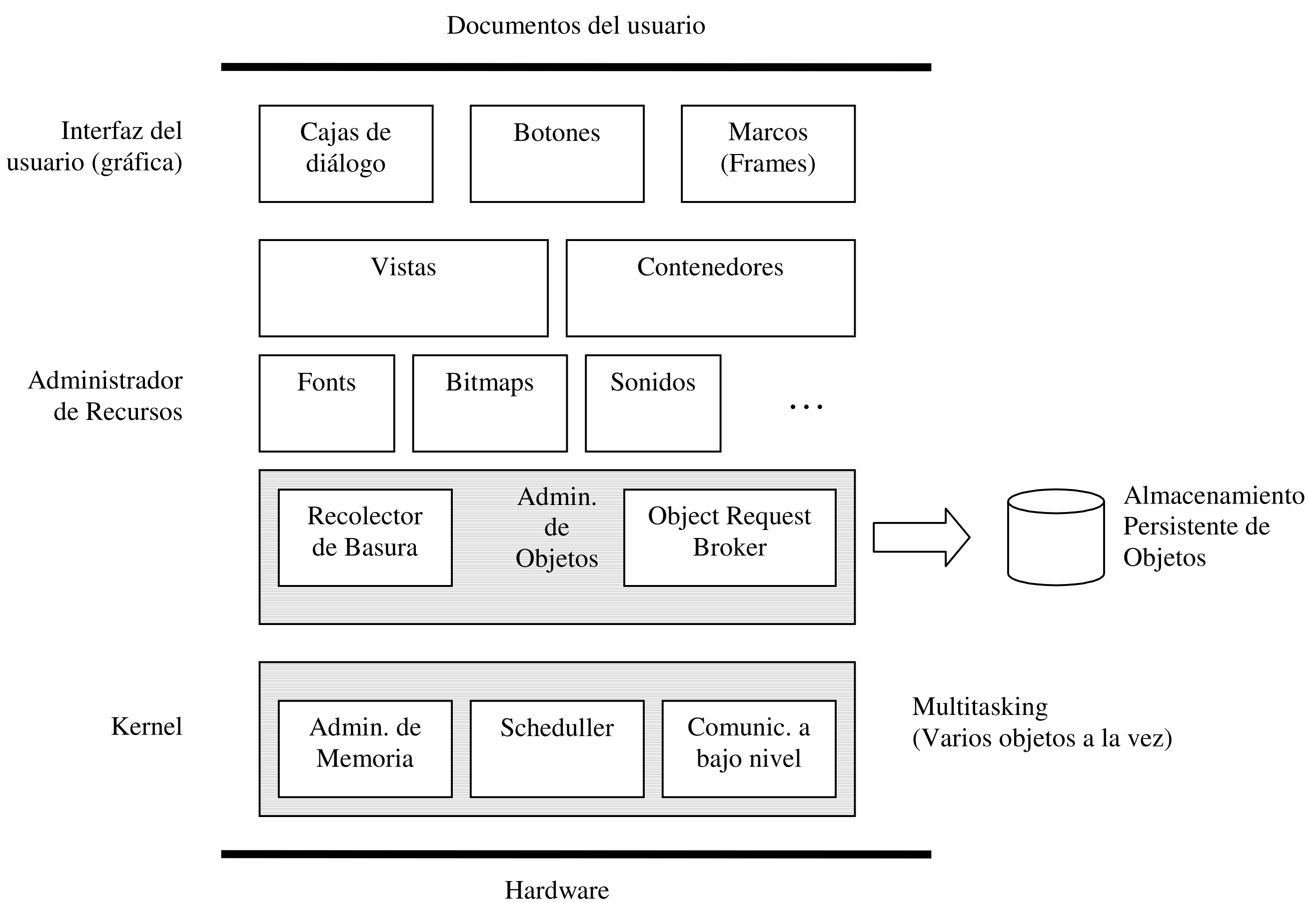
El ocultamiento se refiere a poner los detalles de la construcción del objeto fuera del alcance del usuario para proteger sus valores y comportamiento.

En lugar de la llamada a procedimientos y/o funciones, se utiliza la invocación de métodos unidos a los objetos que son los únicos que pueden modificar las características de los objetos.

El polimorfismo usa una liga dinámica y virtual por medio de la cual distintos objetos hijos de una misma clase se comportan de una manera única y distinta al mismo método

Diseño de un Sistema Operativo Orientado a Objetos (SOOO).

En general, el SOOO se compone principalmente de un microkernel cuya principal función es la administración del hardware y el resto es una serie de capas de objetos reemplazables cuando es necesario.



Objetos: Código más datos. El código se ejecuta sólo cuando los objetos intercambian mensajes. La administraciónde los objetos se realiza de una manera similar a una base de datos. El SOOO debe contar con funciones básicas de Commit y Rollback, Tolerancia a fallas a nivel software y Store and Forward.

Las aplicaciones se desarrollan por medio de la acumulación de objetos tanto del S.O. como del usuario

Ventajas y Desventajas de Sistemas Operativos Orientados a Objetos

Ventajas

+ Son SOs mejor estructurados
+ Existe una más clara separación entre especificación e implantación de objetos
+ La flexibilidad y reusabilidad de los objetos es alta
+ Es posible reutilizar componentes

+ Existe una mejor comunicación tanto interna como externa, ya que todos los objetos trabajan con paso de mensajes
+ La diferenciación de actividades es más clara.

Desventajas

- Existe una fuerte falta de compatibilidad por falta de estándares
- No existen estándares para incrustar un objeto dentro de otro
- Deficiente comunicación entre SOs heterogéneos
- El diseño es altamente complejo, lo que incrementa la probabilidad de errores
- La ejecución de programas es lenta.

Ejemplo: Amoeba

Creado en Vrije Universiteit (Amsterdam, Holanda) por Andrew Tanenbaum en 1981. Liberando la versión 1.0 en 1983.

Amoeba es un SO microkernel orientado a objetos que permite que un conjunto de procesadores y equipo de E/S se comporten como una sola computadora.

Como es un sistema microkernel, se compone de un conjunto de servidores que interactúan para administrar los recursos de la computadora.

Los servidores controlan "objetos". Estos objetos son piezas encapsuladas de datos en las que se pueden llevar a cabo operaciones definidas.

Los objetos son pasivos, por que no tienen procesos o métodos.

Para llevar a cabo una operación con un objeto, un cliente realiza un RPC con el servidor, donde se especifica el objeto, la operación a desarrollar y los parámetros necesarios. El servidor lleva a cabo el trabajo y regresa la respuesta al cliente.

Las operaciones se llevan a cabo en forma sincronizada: después de iniciar un RPC con un servidor, el hilo del cliente se bloquea hasta que el servidor responde. Sin embargo, se pueden ejecutar otros hilos en el mismo proceso.

Los objetos reciben su nombre y protección de manera uniforma mediante boletos especiales llamados "posibilidades".

Para crear un objeto, un cliente realiza una RPC con el servidor apropiado. El servidor crea el objeto y regresa al cliente una posibilidad. En cada operación subsecuente, el cliente debe presentar la posibilidad para identificar al objeto.

Puerto del Servidor	Objeto	Derechos	Verificación
48 bits	24 bits	8 bits	48 bits

Estructura de una posibilidad