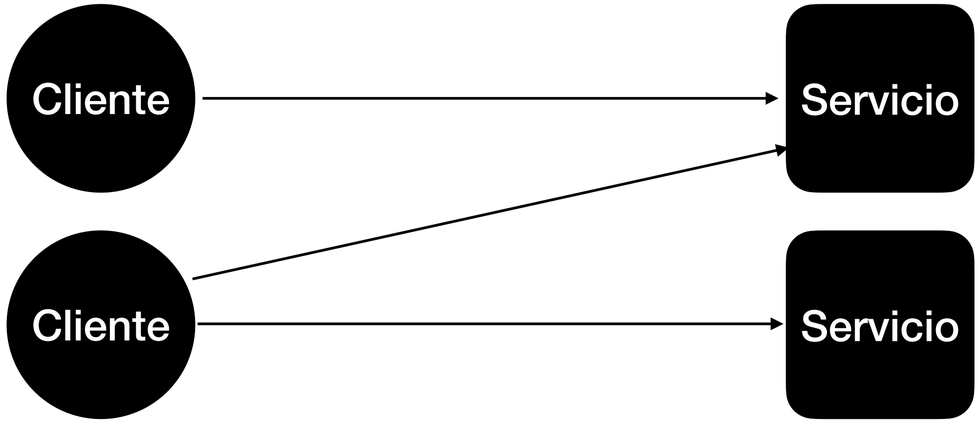
Modularidad por medio de clientes y servicios

Problema

- Aprendimos que dividir un sistema en módulos es una buena estrategia de diseño
- Si los programadores/Diseñadores no cometieran errores esto es todo lo que necesitaríamos.
- La modularización no impide que se propaguen errores en el sistema. Hay interacciones implícitas inesperadas.
- Ante el error necesitamos formas más fuertes de modularización que protejan el sistema de errores.
- Organizar el sistema con clientes y servicios protege el sistema de interacciones inesperadas.

Clientes y Servicios



Al menos tres beneficios:

- Los mensajes son la única forma de interacción (limita interacción y protege contra violación de modularidad)
- Los mensajes son la única forma en que los errores se propagan
- Los mensajes son la única forma en que un atacante penetra un servicio

Esta simple estratégia permite crear sistemas modulares, tolerantes a fallas y seguros

Modularización con virtualización

Problema

- Un computador para cada cliente y servicios es muy costoso
- Modularización en un solo computador permite interacciones inesperada que violan el contrato de modularización
- Necesitamos una forma de correr múltiples programas en un computador sin debilitar la modularidad

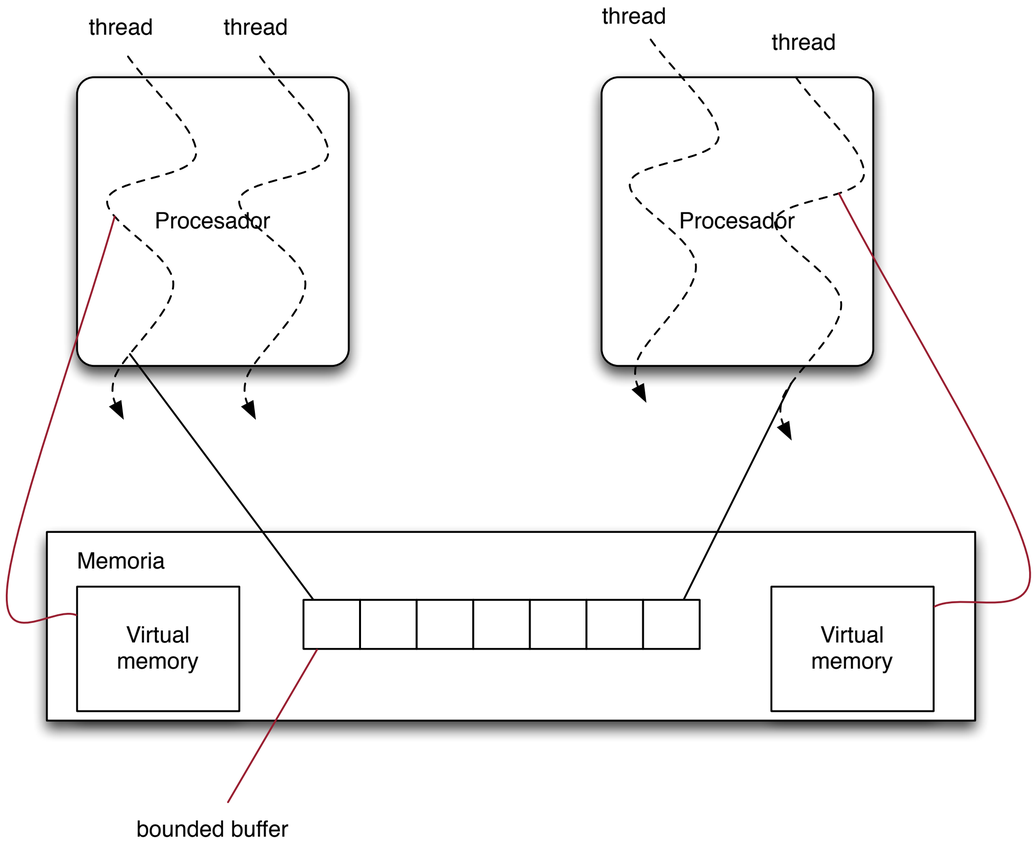
Organizando clientes y servicios usando virtualización

- Creamos múltiples computadores virtuales usando el mismo computador físico.
- Tres abstracciones virtuales de las tres abstracciones principales:
  - Comunicación con buffers con límite (enlace virtual)
  - Memoria virtual
  - Hilos (virtualizan procesador)

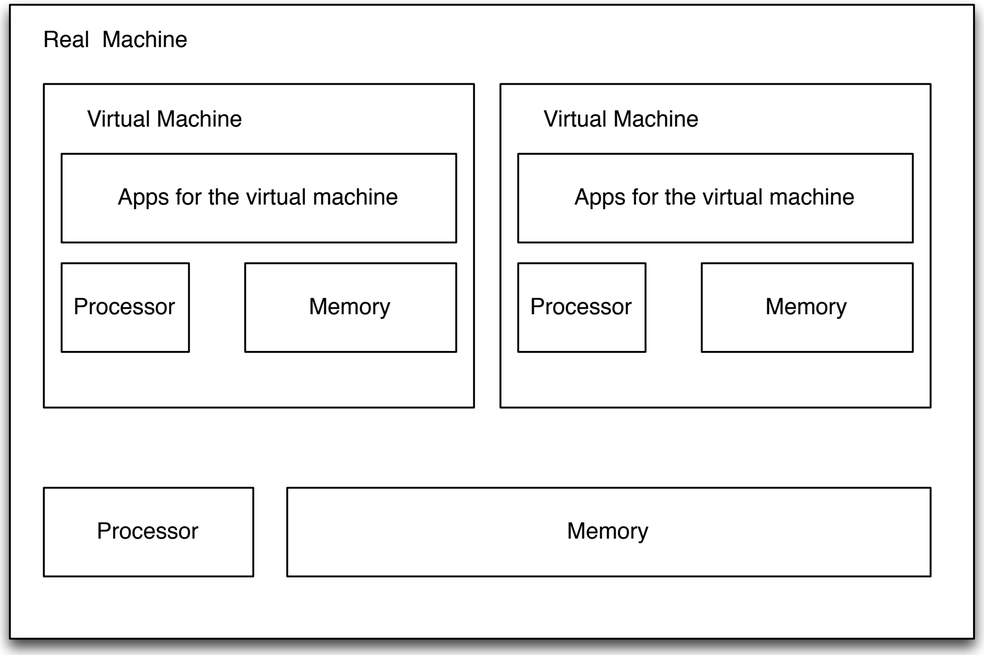
Consideraciones de diseño

- En principio se puede pensar un procesador para cada hilo
- Pero en realidad varios hilos comparten procesador
- La simulación de del buffer con límite se hace en memoria
- Consideraciones de problemas de concurrencia: condiciones de carrera, deadlocks

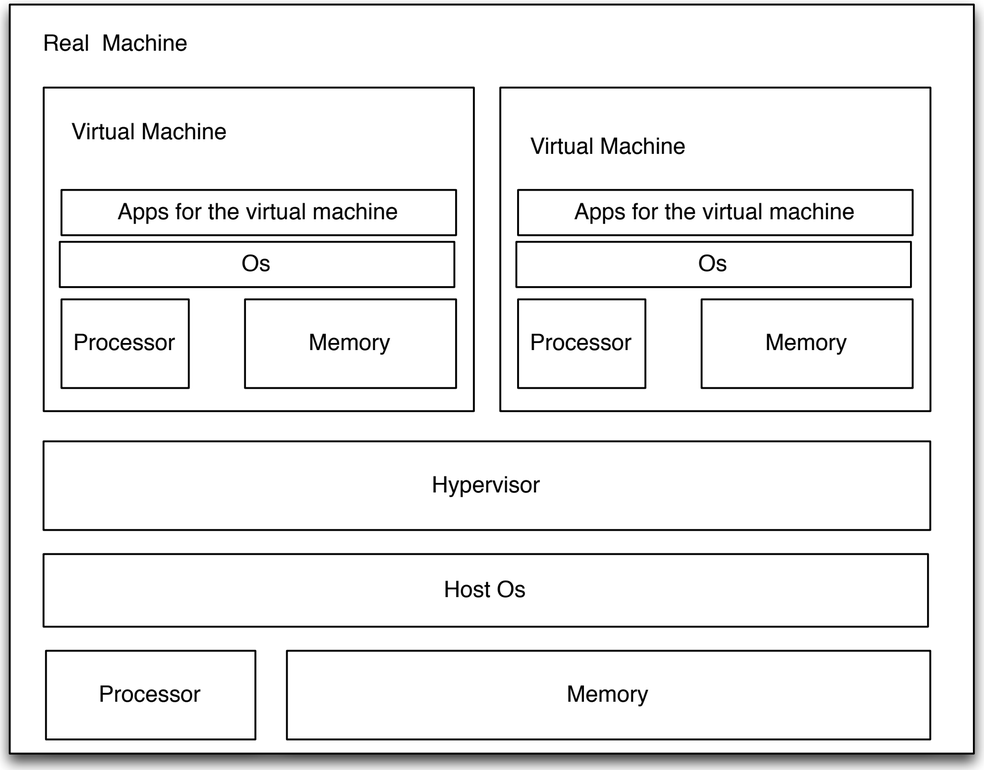
Virtualización en el OS



Máquina virtual para un LP

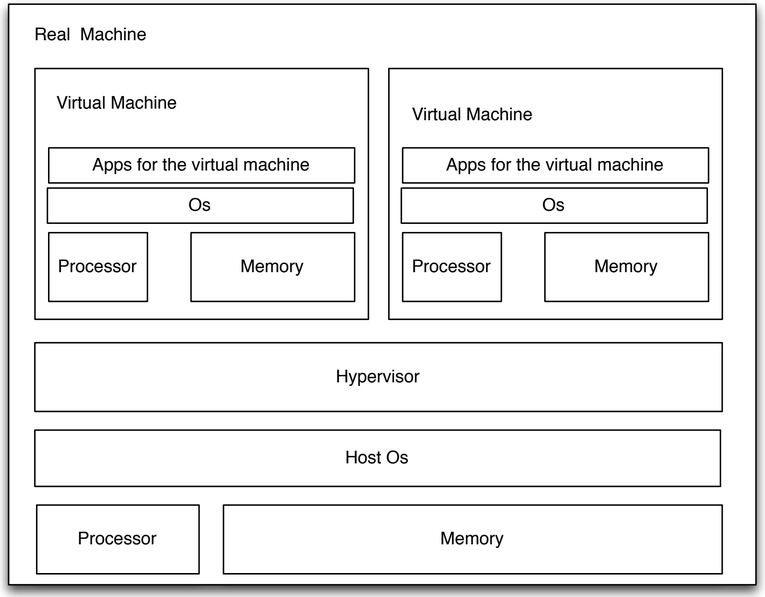


Máquina virtual

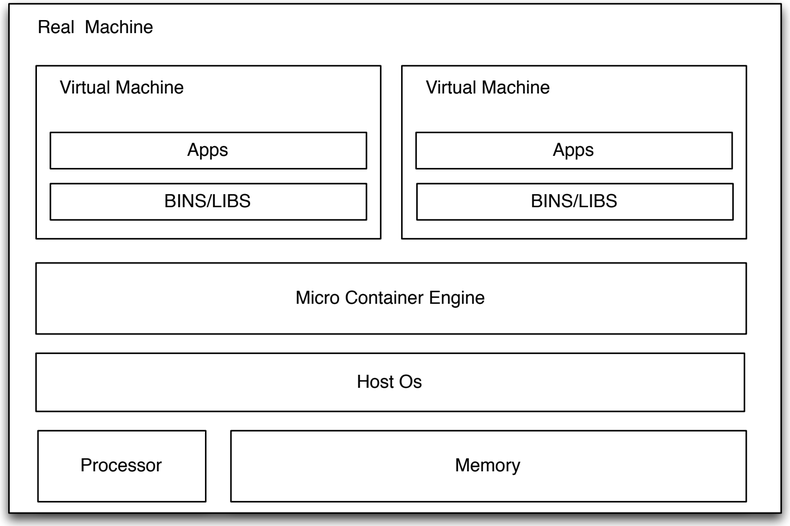


Micro containers

Virtual Machines



Micro Containers



Ejemplos

- Kernel del sistema operativo
  - Monolítico
  - Microkernel
- Hilos en Java o .Net. Cada capa puede crear su propio esquema de virtualización
- Máquinas virtuales, e.g., AWS, Docker, Heroku

Fin