Extended OS
OS is an extended virtual machine

- Multiplexes the “machine” between applications
  - Time sharing, multitasking, batching
- Provided a higher-level machine for
  - Ease of use
  - Portability
  - Efficiency
  - Security
  - Etc....
JAVA – Higher-level Virtual Machine

• write a program once, and run it anywhere
  – Architecture independent
  – Operating System independent

• Language itself was clean, robust, garbage collection

• Program compiled into bytecode
  – Interpreted or just-in-time compiled.
  – Lower than native performance
Interpretation

J.T
Dust-in-time

enter J.T

enter JBC

compile f1

f1

< 8C

η Μ
Issues

• Legacy applications
• No isolation nor resource management between applets
• Security
  – Trust JVM implementation? Trust underlying OS?
• Performance compared to native
Is the OS the “right” level of extended machine?

- Security
  - Trust the underlying OS?
- Legacy application and OSs
- Resource management of existing systems suitable for all applications?
- What about activities requiring “root” privileges
Virtual Machine Monitors

- Provide scheduling and resource management
- Extended “machine” is the actual machine interface.
IBM VM/370

Diagram:
- Virtual 370s
- I/O instructions here
- Trap here
- System calls here
- Trap here
- VM/370
- CMS
- CMS
- CMS
- 370 Bare hardware

Notes:
- Small trusted
Advantages

• Legacy OSes (and applications)
• Concurrent OSes
  – Linux – Windows
  – Primary – Backup
• Security
  – VMM (hopefully) small and correct
• Performance near bare hardware
  – For some applications
Figure 1-29. (a) A type 1 hypervisor. (b) A type 2 hypervisor.
Virtual R3000???

- Interpret
  - System/161
    - slow
  - JIT dynamic compilation

- Run on the real hardware??
Issues

- Privileged registers (CP0)
- Privileged instructions
- Address Spaces
- Exceptions (including syscalls, interrupts)
- Devices