HỆ ĐIỀU HÀNH NÂNG CAO

Trường đại học Khoa học tự nhiên

Khoa Công nghệ Thông tin

Trần Hạnh Nhi

Tổ chức

- Phụ trách Lý thuyết :
  - **Trần Hạnh Nhi**
- Phụ trách thực hành:
  - **Phạm Nguyễn Anh Huy**
  - **Trần Anh Tuấn**
  - **Lê Thụy Anh**
  - **Đinh Bá Tiến**
- Trang web của môn học :

Mục tiêu

- Kết quả mong đợi về lý thuyết :
  - Hiểu được cách thức Hệ điều hành làm việc
  - Nắm được các nguyên lý thiết kế Hệ điều hành
  - Biết được một số cơ chế, chiến lược cơ bản để giải quyết các nhiệm vụ của Hệ điều hành
- Kết quả cần đạt được về thực hành
  - Vận dụng được các kiến thức lý thuyết để cài đặt giả lập một số module của Hệ điều hành
  - Sử dụng được các cơ chế hỗ trợ của một Hệ điều hành cụ thể (Windows NT) để giải quyết các bài toán cơ bản.

Kiến thức yêu cầu

- Kiến trúc Máy tính
- Hệ điều hành cơ bản
- Lập trình C/C++

Tính điểm

- 70% Lý thuyết + 30% Thực hành
- Lý thuyết :
  - 1 bài thi cuối khoá (không tham khảo tài liệu)
  - Mỗi sinh viên làm bài độc lập
- Thực hành: 2 bài tập lớn
  - Thời hạn và cách thức nộp bài sẽ do giáo viên phụ trách thực hành qui định
  - Mỗi nhóm thực hành gồm 2 sinh viên
- Bắt buộc có nộp bài thực hành mới được thi lý thuyết

Tài liệu tham khảo

- *Trần Hạnh Nhi* : Giáo trình Hệ điều hành Nâng cao
- *A.Silberschatz & P/Galvin* : OS concepts (5e)
  - Slides :
- *W. Stallings* : Operating Systems
- *A.Tanenbaum et al* : OS Design and Implementation
  - Minix :
- *R.Finkel*:: An OS vade mecum
  - Book online :
- *Jeffrey Richter* : Advanced Windows
- *Tiến Huy- Đan Thư- Hạnh Nhi* : Kỹ thuật lập trình trên Windows NT

Nội dung

- Chương 1 : Tổ chức Hệ điều hành
- Chương 2 : Quản lý tiến trình
- Chương 3 : Liên lạc giữa các tiến trình
- Chương 4 : Quản lý bộ nhớ chính
- Chương 5 : An toàn hệ thống

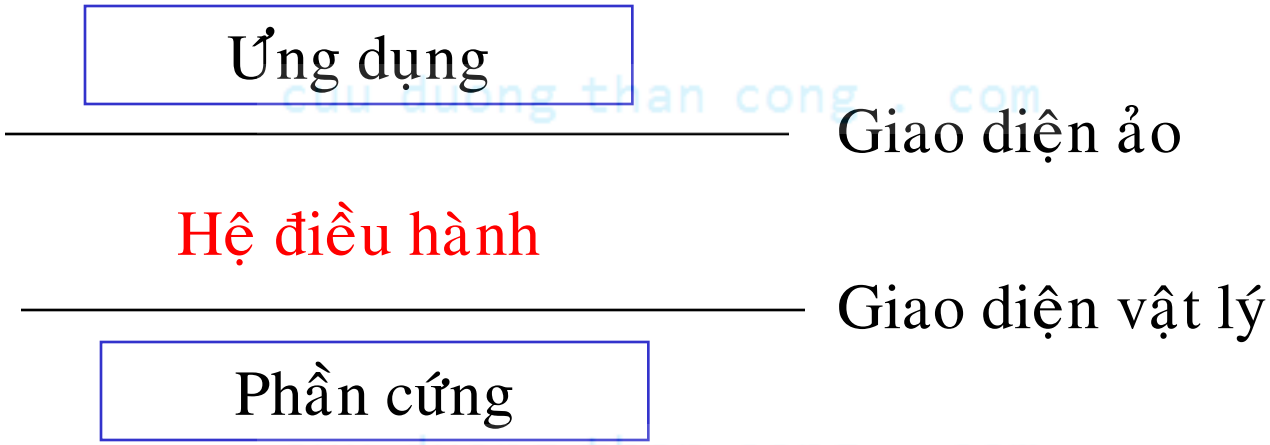
Bài giảng 1 : Giới thiệu

- Tại sao phải tìm hiểu về Hệ điều hành ?
- Hệ điều hành là gì ?
  - Vai trò trong hệ thống ?
  - Chức năng ?
  - Kiến trúc ?
- Các nguyên lý thiết kế Hệ điều hành

Tại sao cần tìm hiểu Hệ điều hành ?

- Để phá vỡ sự "bí ẩn" của hệ thống :
  - Tại sao máy tính có thể "biết" được nội dung đĩa ?
  - Tại sao có thể vừa soạn thảo, vừa nghe nhạc trên cùng 1 máy tính (có 1 CPU ?)
  - Tại sao 1 ứng dụng kích thước 1 M có thể hoạt động trên Windows mà bị báo "Not enough memory" trên DOS ?
- Để khai thác tốt hơn môi trường làm việc :
  - Lập trình trên môi trường đa nhiệm (multitask), đa xử lý(multiprocessing) với các mô hình multiprocess, multithreads..
  - Sử dụng bộ nhớ hiệu quả
  - sử dụng các cơ chế Thông tin liên lạc, an toàn & bảo mật...
- Vì là môn học bắt buộc ☺

Hệ điều hành, anh là ai ?



Chức năng của Hệ điều hành

- Quản trị tài nguyên (resource principle) :
  - Tài nguyên : CPU, Mem, IO; Files, ports, mailboxes...
  - Đối tượng sử dụng tài nguyên : Process, Thread
  - Nhiệm vụ : Cung cấp các giải thuật cấp phát, quản lý tài nguyên.cho các đối tượng hoạt động trong hệ thống
  - Mục tiêu : Cấp phát đầy đủ, công bằng R cho Ps; Sử dụng hiệu quả Rs, Nâng cao thông lượng Ps...
- Trừu tượng hoá hệ thống (beautification principle)
  - Nhiệm vụ : Cung cấp các giải thuật để che dấu chi tiết phần cứng, tạo 1 môi trường dễ làm việc hơn (hope) cho user
  - Mục tiêu : tạo môi trường an toàn, tạo sự trừu tượng hoá, độc lập thiết bị
  - Ví dụ : device driver

Các thành phần

Quản lý bộ nhớ phụ

Quản lý tiến trình

Quản lý nhập xuất

Hệ thống tập tin

Quản lý bộ nhớ chính

Hệ thống bảo vệ

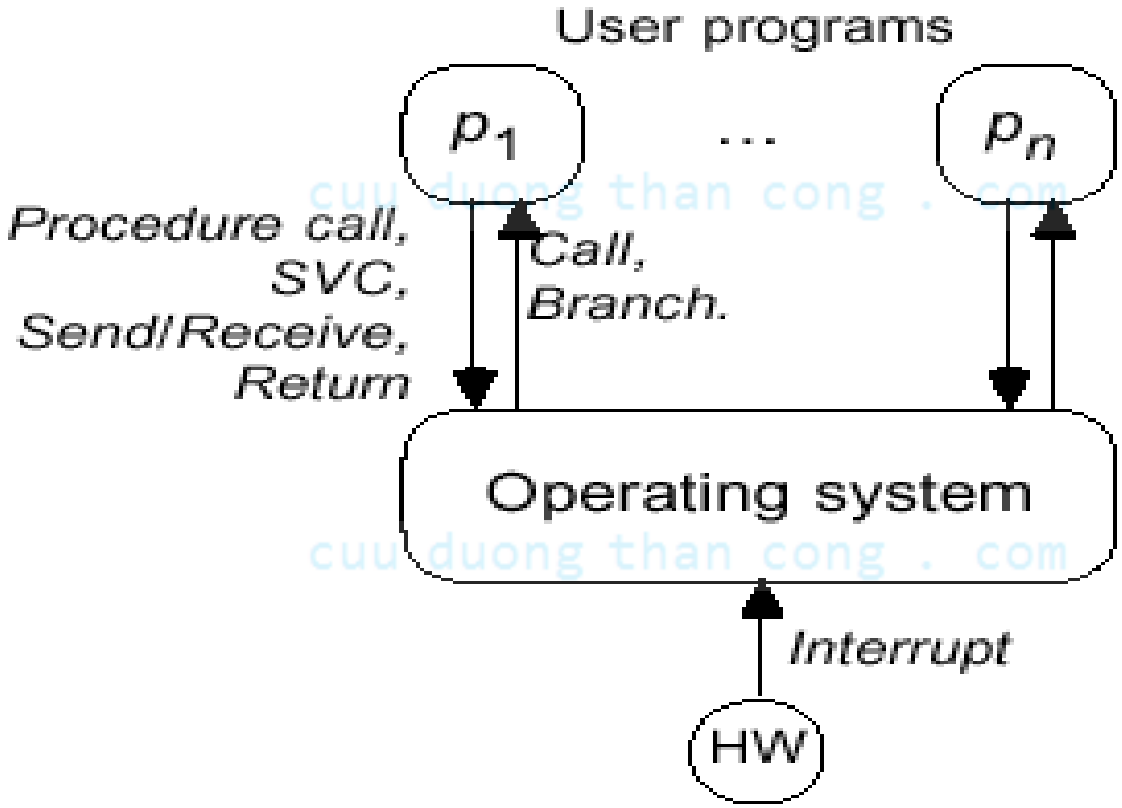
Bộ thông dịch lệnh

Giao tiếp mạng

Kiến trúc Hệ điều hành

- Đơn giản (Monolithic)
- Hạt nhân (Kernel)
- Phân lớp (Layered)
- Máy ảo (Virtual Machine)
- Hướng đối tượng (OOOS)
- Exokernel

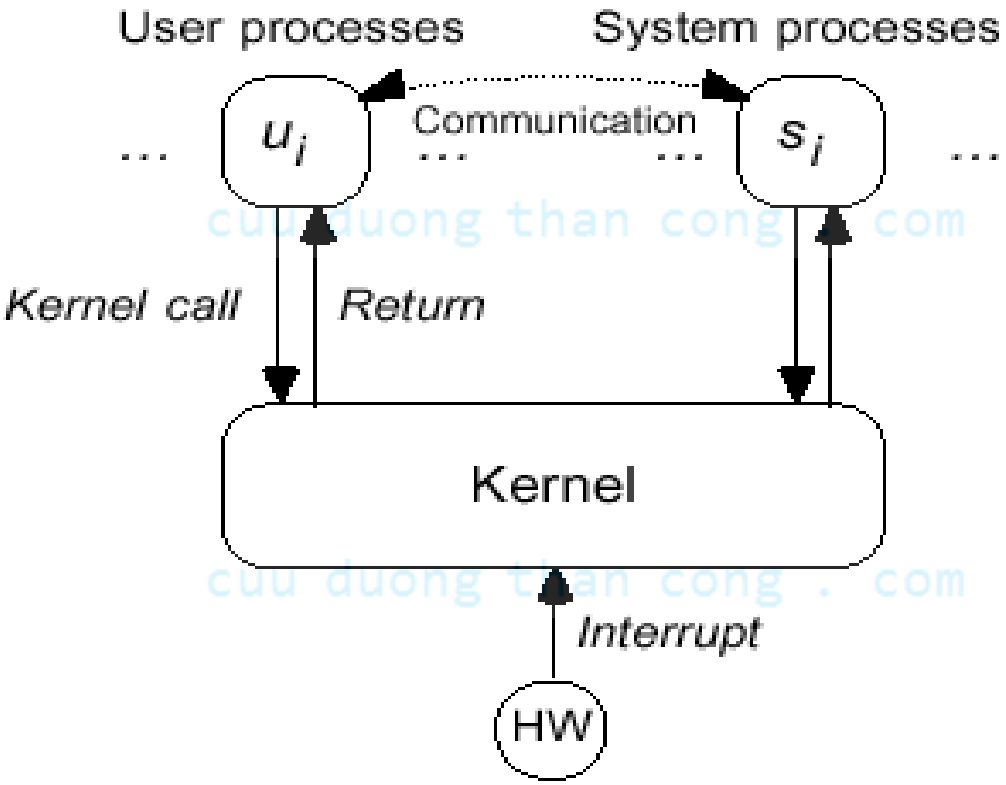
Monolithic



Monolithic

- OS = Thư viện tiện ích
- Có thể tổ chức thành nhiều module : CPU scheduling, Mem Management, Device management...nhưng chỉ có 1 trong những module này hoạt động tại một thời điểm
- Đơn nhiệm
- Quyền điều khiển được chuyển đổi thông qua lời gọi hàm
- Khi tầm vóc phát triển hệ thống trở nên thiếu tin cậy.
- Ví dụ : MS-DOS, Ultrix (mature Unix)

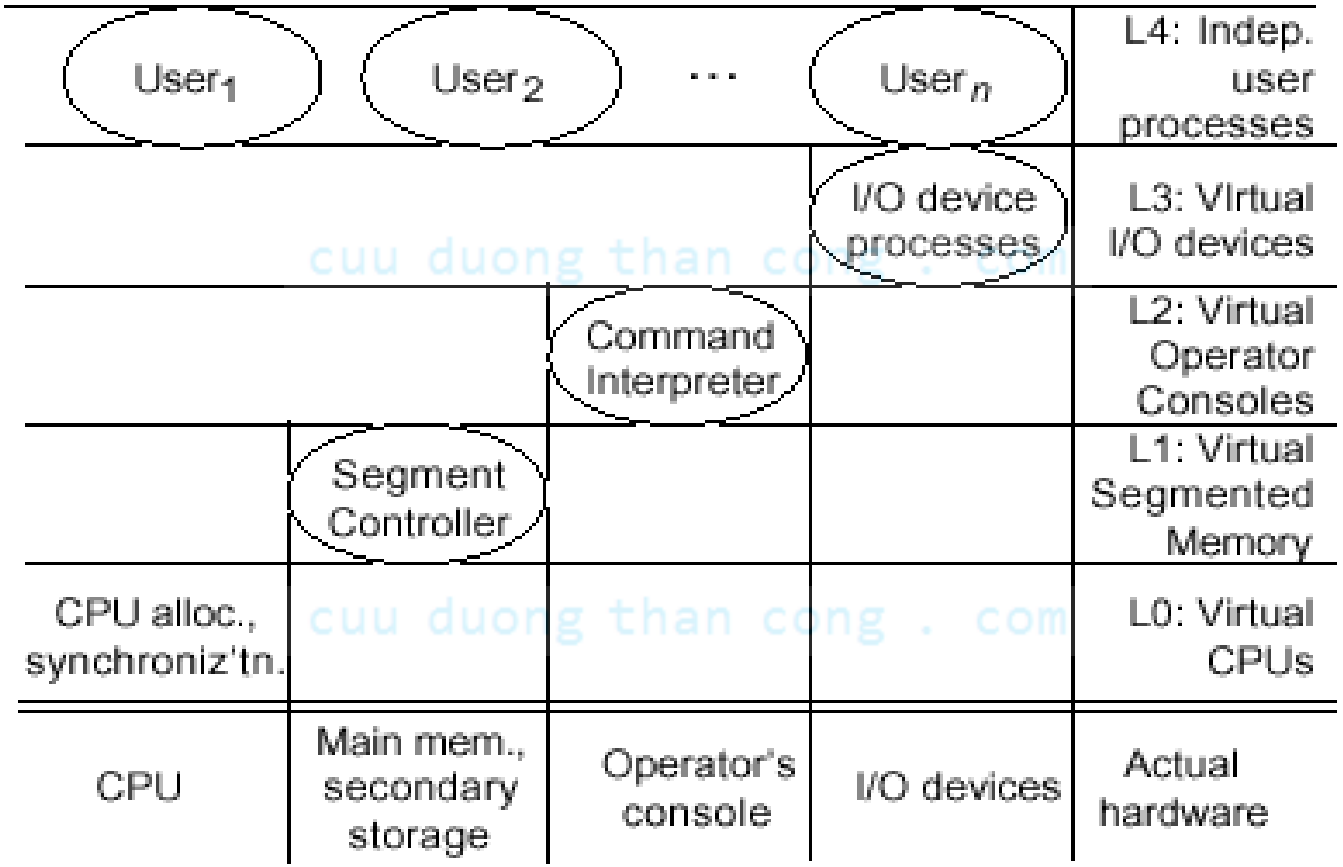
Kernel



Kernel

- OS = Kernel + System processes
- Kernel được bảo vệ
- Đa nhiệm
- Kernel chịu trách nhiệm phân chia thời gian sử dụng CPU, Giao tiếp giữa các tiến trình
- Chỉ có 2 mức kernel/non-kernel =>kernel lớn, thiếu tin cậy như trước
- Định nghĩa cứng các giao tiếp với ứng dụng trong kernel
- Ví dụ : Windows NT

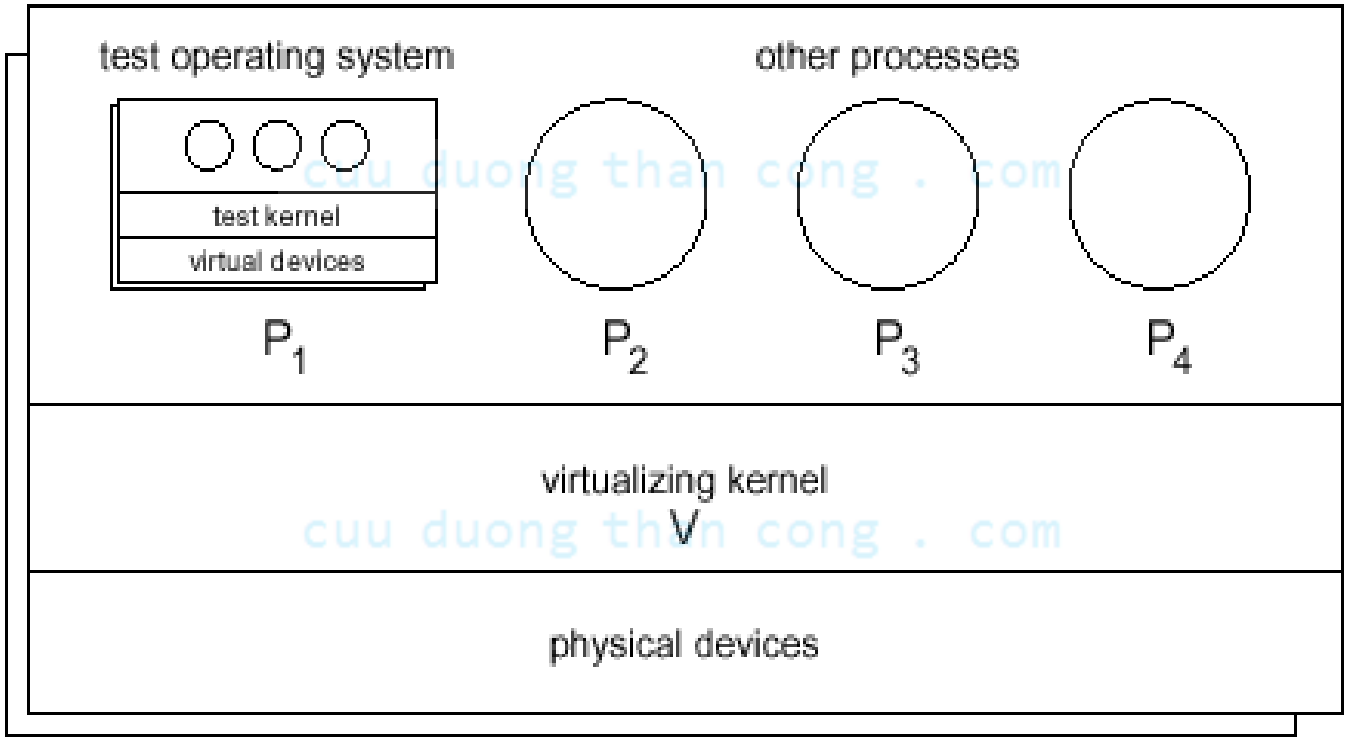
Layered



Layered

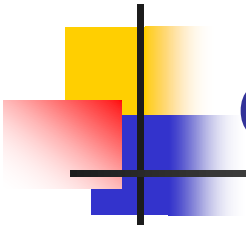
- OS = các lớp trừu tượng hoá một tác vụ quản lý
- Lớp trên được sử dụng các hàm xử lý tài nguyên thuộc tác vụ do lớp dưới cung cấp
- Khó xác định được các lớp xử lý rạch ròi, thứ tự lớp ?
  - Tạo tiến trình -> PM gọi MM
  - Bộ nhớ đầy -> MM gọi PM
- Xếp lớp theo hàm xử lý , thay vì tác vụ
  - Seg management- P scheduling- Seg creation- P creation
- Ví dụ : THE , MULTICS

Virtual Machine



Virtual Machine

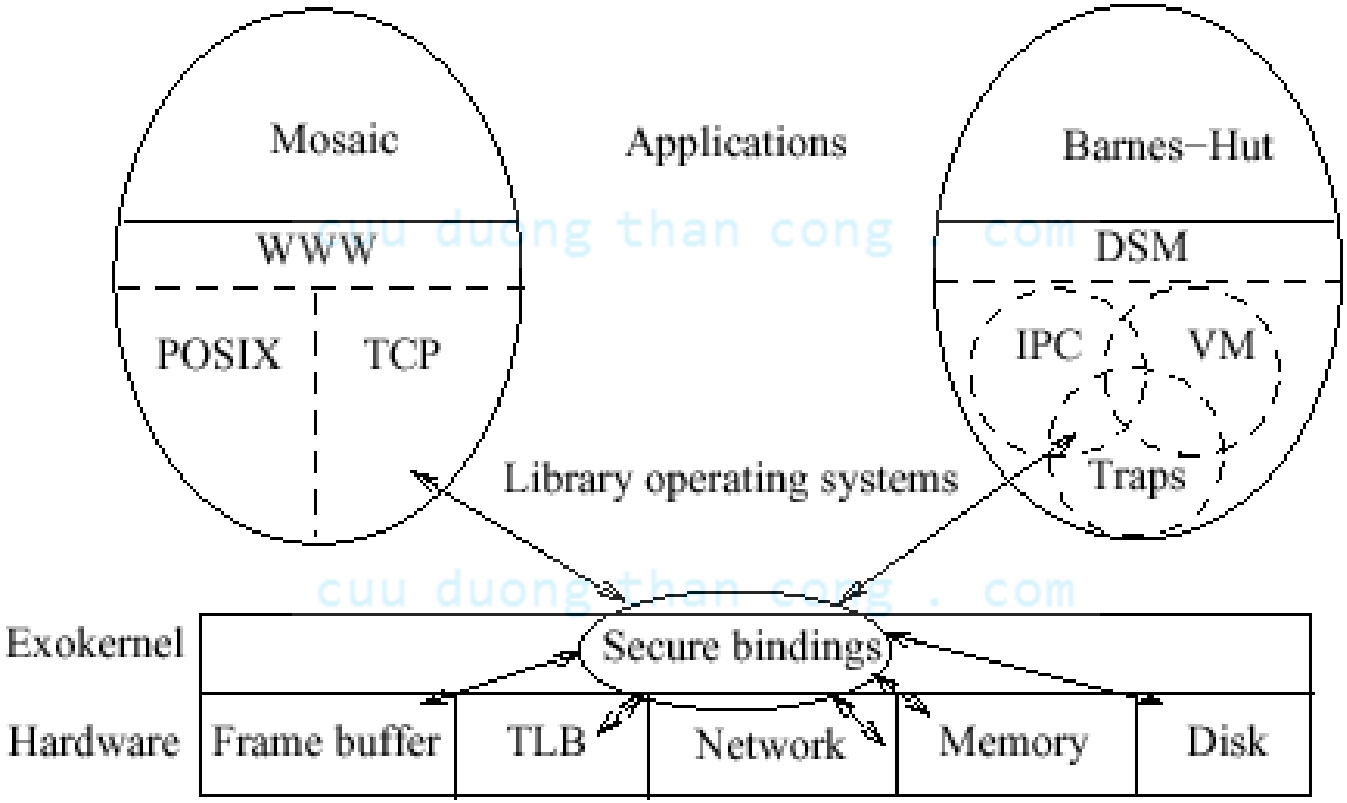
- OS = Virtualizing kernel + virtual machines
- Virtual machine = physical hardware
- Virtualizing kernel tạo ra nhiều VM trên 1 máy tính.
- Process interface = hardware interface
- Ưu điểm :
  - Môi trường thuận lợi cho sự tương thích (compatibility)
  - Tăng tính an toàn hệ thống do cung cấp các VM độc lập.
  - Dể phát triển các HDH đơn nhiệm cho mỗi VM
- Khuyết điểm:
  - Phức tạp cho việc giả lặp (transput, add translation...)
- Ví dụ : CMS(conversational Monitor System) trên VM/370 (hỗ trợ hardware)



OOOS

- OS = tập các đối tượng
- Tiến trình, tập tin, hàm, khối nhớ...
- Một hàm xử lý (kernel/non-kernel mode) thao tác trên một tập các đối tượng.
- Che dấu thông tin
- Ví dụ :CAP, StarOS, iMAX432

Exokernel



Exokernel

- Hướng đến một HDH linh đông trong giao tiếp với ứng dụng, cho phép ứng dụng chuyên biệt hoá hệ điều hành theo nhu cầu đặc thù một cách dễ dàng
- OS = Exokernel + Library OS
- Ứng dụng có thể phát triển các mô hình tổ chức VM, IPC theo nhu cầu riêng
- Ví dụ : ý tưởng của project do Dawson R Engler et al phát triển tại MIT