

## CHƯƠNG 2. CẤU TRÚC HỆ ĐIỀU HÀNH

Hệ điều hành cung cấp môi trường cho các chương trình thực thi. Các hệ điều hành rất khác biệt nhau về kiến trúc, chúng được tổ chức cùng với các dòng khác nhau. Thiết kế một hệ điều hành mới là một công việc rất khó khăn. Các mục đích của hệ thống phải được định nghĩa rõ ràng trước khi thiết kế bắt đầu. Kiểu hệ thống mong muốn là cơ sở cho việc chọn lựa giữa các giải thuật và chiến lược khác nhau.

Hệ điều hành có thể được nhìn từ nhiều lợi ích khác nhau. Người này xem xét các dịch vụ mà hệ điều hành cung cấp. Người kia quan tâm đến giao diện mà hệ điều hành mang lại cho người dùng và người lập trình. Người khác lại phân rã hệ thống thành những thành phần và các mối quan hệ bên trong của chúng. Trong chương này chúng ta tìm hiểu cả ba khía cạnh của hệ điều hành, thể hiện ba quan điểm của người dùng, người lập trình và người thiết kế hệ điều hành. Chúng ta xem xét các dịch vụ mà hệ điều hành cung cấp, cách chúng được cung cấp và các phương pháp khác nhau được dùng cho việc thiết kế hệ điều hành.

### I. Các thành phần hệ thống

Chúng ta có thể tạo ra một hệ thống lớn và phức tạp như hệ điều hành chỉ khi phân chia hệ điều hành thành những phần nhỏ hơn. Mỗi phần nên là một thành phần được mô tả rõ ràng của hệ thống, với xuất, nhập và các chức năng được định nghĩa cẩn thận. Tuy nhiên, nhiều hệ thống hiện đại chia sẻ mục tiêu hỗ trợ các thành phần hệ thống được liệt kê sau đây:

#### 1.1 Quản lý tiến trình

Một chương trình không làm gì trừ khi các chỉ thị của nó được thực thi bởi một CPU. Một tiến trình có thể được xem như một chương đang thực thi, nhưng định nghĩa của nó sẽ mở rộng khi chúng ta khám phá chi tiết hơn. Một chương trình người dùng được chia thời chẳng hạn như một trình biên dịch là một tiến trình. Một chương trình xử lý văn bản đang được thực thi bởi một người dùng trên một PC cũng là một tiến trình. Một tác vụ hệ thống, như gửi dữ liệu xuất ra máy in cũng được xem là một tiến trình. Bây giờ chúng ta có thể xem xét một tiến trình là một công việc hay chương trình chia thời, nhưng chúng ta sẽ nghiên cứu khái niệm này tổng quát hơn trong các chương sau.

Một tiến trình cần các tài nguyên xác định-gồm thời gian CPU, bộ nhớ, tập tin, các thiết bị xuất/nhập-để hoàn thành tác vụ của nó. Các tài nguyên này được cấp cho tiến trình khi nó được tạo ra, hay được cấp phát tới nó khi nó đang chạy. Ngoài ra, các tài nguyên vật lý và luận lý khác nhau mà tiến trình nhận được khi nó được tạo, dữ liệu khởi tạo khác nhau (hay nhập) có thể được truyền qua. Thí dụ, xem xét một tiến trình có chức năng hiển thị trạng thái của một tập tin trên màn hình của một thiết bị đầu cuối. Tiến trình này sẽ được cho dữ liệu vào là tên của tập tin, và sẽ thực thi các chỉ thị thích hợp và các lời gọi hệ thống đạt được và xuất trên thiết bị cuối thông tin mong muốn. Khi tiến trình này kết thúc, hệ điều hành sẽ đòi lại bất cứ tài nguyên nào có thể dùng lại.

Chúng ta nhấn mạnh một chương trình chính nó không phải là một tiến trình;

một chương trình là một thực thể thụ động, như là nội dung của tập tin được lưu trên đĩa, trái lại một tiến trình là một thực thể hoạt động, với một bộ đếm chương trình xác định chỉ thị kế tiếp để thực thi. Việc thực thi của tiến trình phải là tuần tự. CPU thực thi một chỉ thị của tiến trình sau khi đã thực thi một chỉ thực trước đó cho đến khi tiến trình hoàn thành. Ngoài ra, tại bất kỳ thời điểm nào, tối đa một chỉ thị được thực thi cho tiến trình. Do đó, mặc dù hai tiến trình có thể được liên kết với cùng một tiến trình, vì thế chúng được xem như hai chuỗi thực thi riêng. Thông thường có một chương trình sinh ra nhiều tiến trình khi nó thực thi.

Một tiến trình là một đơn vị công việc trong hệ thống. Một hệ thống chứa tập các tiến trình, một vài tiến trình này là các tiến trình hệ điều hành (thực thi mã hệ thống) và các tiến trình còn lại là các tiến trình người dùng (chúng thực thi mã người dùng). Tất cả các tiến trình này có tiềm năng thực thi đồng hành bằng cách đa hợp CPU giữa các tiến trình.

Hệ điều hành có nhiệm vụ cho các hoạt động sau khi đề cập đến chức năng quản lý tiến trình:

- Tạo và xoá các tiến trình người dùng và hệ thống
- Tạm dừng và thực thi tiếp tiến trình
- Cung cấp các cơ chế đồng bộ hoá tiến trình
- Cung cấp các cơ chế giao tiếp tiến trình
- Cung cấp cơ chế quản lý deadlock

## 1.2 Quản lý bộ nhớ chính

Bộ nhớ chính là trung tâm điều hành của một máy tính hiện đại. Bộ nhớ chính là một mảng các từ (words) hay bytes có kích thước lớn từ hàng trăm ngàn tới hàng tỉ. Mỗi từ hay byte có địa chỉ riêng. Bộ nhớ chính là một kho chứa dữ liệu có khả năng truy xuất nhanh được chia sẻ bởi CPU và các thiết bị xuất/nhập. Bộ xử lý trung tâm đọc các chỉ thị từ bộ nhớ trong chu kỳ lấy chỉ thị, nó đọc và viết dữ liệu từ bộ nhớ chính trong chu kỳ lấy dữ liệu. Bộ nhớ chính thường là thiết bị lưu trữ lớn mà CPU có thể định địa chỉ và truy xuất trực tiếp. Thí dụ, đối với CPU xử lý dữ liệu từ đĩa, dữ liệu trước tiên được chuyển tới bộ nhớ chính bởi lời gọi xuất/nhập được sinh ra bởi CPU. Tương tự, các chỉ thị phải ở trong bộ nhớ cho CPU thực thi chúng.

Đối với một chương trình được thực thi, nó phải được ánh xạ các địa chỉ và được nạp vào bộ nhớ. Khi chương trình thực thi, nó truy xuất các chỉ thị chương trình và dữ liệu từ bộ nhớ bằng cách tạo ra các địa chỉ tuyệt đối này. Cuối cùng, chương trình kết thúc, không gian bộ nhớ của nó được khai báo sẵn, và chương trình có thể được nạp và thực thi.

Để cải tiến việc sử dụng CPU và tốc độ đáp ứng của máy tính cho người dùng, chúng ta phải giữ nhiều chương trình vào bộ nhớ. Nhiều cơ chế quản lý bộ nhớ khác nhau được dùng và tính hiệu quả của các giải thuật phụ thuộc vào từng trường hợp cụ thể. Chọn một cơ chế quản lý bộ nhớ cho một hệ thống xác định phụ thuộc vào nhiều yếu tố đặc biệt trên thiết kế phần cứng của hệ thống. Mỗi giải thuật đòi hỏi sự hỗ trợ phần cứng của nó.

Hệ điều hành có nhiệm vụ cho các hoạt động sau khi đề cập tới việc quản lý bộ nhớ

- Giữ vết về phần nào của bộ nhớ hiện đang được dùng và quá trình nào đang dùng.
- Quyết định quá trình nào được nạp vào bộ nhớ khi không gian bộ nhớ trở nên sẵn dùng.
- Cấp phát và thu hồi không gian bộ nhớ khi được yêu cầu.

### 1.3 Quản lý tập tin

Quản lý tập tin là một trong những thành phần có thể nhìn thấy nhất của hệ điều hành. Máy tính có thể lưu thông tin trên nhiều loại phương tiện lưu trữ vật lý khác nhau. Băng từ, đĩa từ, đĩa quang là những phương tiện thông dụng nhất. Mỗi phương tiện này có đặc điểm và tổ chức riêng. Mỗi phương tiện được điều khiển bởi một thiết bị, như một ổ đĩa hay ổ băng từ. Các thuộc tính này bao gồm tốc độ truy xuất, dung lượng, tốc độ truyền dữ liệu và phương pháp truy xuất (tuần tự hay ngẫu nhiên).

Nhờ vào việc sử dụng thuận lợi hệ thống máy tính, hệ điều hành cung cấp tầm nhìn luận lý của việc lưu trữ thông tin đồng nhất. Hệ điều hành trừu tượng hoá các thuộc tính vật lý của các thiết bị lưu trữ để định nghĩa một đơn vị lưu trữ luận lý là tập tin. Hệ điều hành ánh xạ các tập tin trên các thiết bị lưu trữ vật lý, và truy xuất các tập tin này bằng các thiết bị lưu trữ.

Tập tin là tập hợp thông tin có quan hệ được định nghĩa bởi người tạo. Thông thường, các tập tin biểu diễn chương trình và dữ liệu. Các tập tin dữ liệu có thể là số, chữ cái, chữ số. Các tập tin có dạng bất kỳ (thí dụ, các tập tin văn bản) hay có thể được định dạng có cấu trúc (thí dụ, các trường cố định). Một tập tin chứa một chuỗi các bits, bytes, các dòng hay các mẫu tin mà ý nghĩa của nó được định nghĩa bởi người tạo. Khái niệm tập tin là một khái niệm cực kỳ thông dụng.

Hệ điều hành cài đặt một khái niệm trừu tượng của tập tin bằng cách quản lý phương tiện lưu trữ như đĩa, băng từ và các thiết bị điều khiển chúng. Các tập tin cũng thường được tổ chức trong các thư mục để dễ dàng sử dụng chúng. Cuối cùng, khi nhiều người dùng truy xuất tập tin, chúng ta muốn kiểm soát ai và trong cách gì (thí dụ: đọc, viết, chèn,...) các tập tin có thể được truy xuất.

Hệ điều hành có nhiệm vụ thực hiện các hoạt động trong việc quản lý hệ thống tập tin:

- Tạo và xoá tập tin
- Tạo và xoá thư mục
- Hỗ trợ các hàm nguyên thủy để thao tác tập tin và thư mục
- Ánh xạ các tập tin trên các thiết bị lưu trữ phụ
- Sao lưu dự phòng tập tin trên các phương tiện lưu trữ ổn định

### 1.4 Quản lý hệ thống xuất/nhập

Một trong những mục đích của hệ điều hành là che giấu sự khác biệt của các thiết bị phần cứng từ người dùng. Thí dụ, trong UNIX sự khác biệt của các thiết bị

xuất/nhập bị che giấu từ phần chính của hệ điều hành bởi các hệ thống con xuất/nhập. Hệ thống con xuất/nhập chứa:

- Thành phần quản lý bộ nhớ chứa vùng đệm (buffering), lưu trữ (caching) và spooling (vùng chứa).
- Giao diện trình điều khiển thiết bị chung.
- Trình điều khiển cho các thiết bị xác định.

Chỉ trình điều khiển thiết bị biết sự khác biệt của các thiết bị xác định mà nó được gán

### 1.5 Quản lý bộ lưu trữ phụ

Mục đích chính của một hệ thống máy tính là thực thi các chương trình. Những chương trình này với dữ liệu chúng truy xuất phải nằm trong bộ nhớ chính hay lưu trữ chính trong quá trình thực thi. Vì bộ nhớ chính quá nhỏ để lưu tất cả dữ liệu và chương trình và vì dữ liệu quản lý bị mất khi mất điện, hệ thống máy tính phải cung cấp việc lưu trữ phụ để lưu dự phòng bộ nhớ chính. Hầu hết các hệ thống máy tính hiện đại dùng đĩa như phương tiện lưu trữ trực tuyến cho cả chương trình và dữ liệu.

Hầu hết các chương trình – gồm trình biên dịch, trình dịch hợp ngữ, thủ tục sắp xếp, trình soạn thảo và trình định dạng – được lưu trên đĩa cho tới khi được nạp vào trong bộ nhớ và sau đó dùng đĩa khi cả hai nguồn và đích của việc xử lý. Do đó, quản lý hợp lý việc lưu trữ đĩa có vai trò quan trọng đối với một hệ thống máy tính.

Hệ điều hành có nhiệm vụ thực hiện các hoạt động sau trong việc quản lý đĩa:

- Quản lý không gian trống
- Cấp phát lưu trữ
- Định thời đĩa

Vì lưu trữ phụ được dùng thường xuyên nên nó phải được dùng một cách hiệu quả. Tốc độ toàn bộ của các thao tác của máy tính có thể xoay quanh tốc độ hệ thống con đĩa và các giải thuật thao tác trên hệ thống con đó.

### 1.6 Mạng

Hệ phân tán là tập hợp các bộ xử lý, chúng không chia sẻ bộ nhớ, các thiết bị ngoại vi hay đồng hồ. Thay vào đó mỗi bộ xử lý có bộ nhớ, đồng hồ và các bộ xử lý giao tiếp với nhau thông qua các đường giao tiếp như bus tốc độ cao hay mạng. Các bộ xử lý trong hệ thống phân tán khác nhau về kích thước và chức năng. Chúng có thể chứa các bộ vi xử lý, trạm làm việc, máy vi tính và các hệ thống máy tính thông thường.

Các bộ xử lý trong hệ thống được nối với nhau thông qua mạng truyền thông có thể được cấu hình trong nhiều cách khác nhau. Mạng có thể được nối kết một phần hay toàn bộ. Thiết kế mạng truyền thông phải xem xét vạch đường thông điệp và các chiến lược nối kết, và các vấn đề cạnh tranh hay bảo mật.

Hệ thống phân tán tập hợp những hệ thống vật lý riêng rẽ, có thể có kiến trúc không đồng nhất thành một hệ thống chặt chẽ, cung cấp người dùng với truy xuất tới các tài nguyên khác nhau mà hệ thống duy trì. Truy xuất tới các tài nguyên chia sẻ cho

phép tăng tốc độ tính toán, chức năng, khả năng sử dụng của dữ liệu, khả năng tin cậy. Hệ điều hành thường tổng quát hoá việc truy xuất mạng như một dạng truy xuất tập tin, với những chi tiết mạng được chứa trong trình điều khiển thiết bị của giao diện mạng.

Các giao thức tạo một hệ thống phân tán có thể có một ảnh hưởng to lớn trên tiện ích và tính phổ biến của hệ thống đó. Sự đổi mới của World Wide Web đã tạo ra một phương pháp truy xuất mới cho thông tin chia sẻ. Nó đã cải tiến giao thức truyền tập tin (File Transfer Protocol-FTP) và hệ thống tập tin mạng (Network File System-NFS) đã có bằng cách xoá yêu cầu cho một người dùng đăng nhập trước khi người dùng đó được phép dùng tài nguyên ở xa. Định nghĩa một giao thức mới, giao thức truyền siêu văn bản (hypertext transfer protocol-http), dùng trong giao tiếp giữa một trình phục vụ web và trình duyệt web. Trình duyệt web chỉ cần gửi yêu cầu thông tin tới một trình phục vụ web của máy ở xa, thông tin (văn bản, đồ hoạ, liên kết tới những thông tin khác) được trả về.

### 1.7 Hệ thống bảo vệ

Nếu một hệ thống máy tính có nhiều người dùng và cho phép thực thi đồng hành của nhiều quá trình, thì các quá trình khác nhau phải được bảo vệ từ các hoạt động của quá trình khác. Cho mục đích này, các cơ chế đảm bảo rằng các tập tin, phân đoạn bộ nhớ, CPU, và các tài nguyên khác có thể được điều hành chỉ bởi các quá trình có quyền phù hợp từ hệ điều hành.

Ví dụ, phần cứng định địa chỉ bộ nhớ đảm bảo rằng một quá trình có thể thực thi chỉ trong không gian địa chỉ của chính nó. Bộ định thời đảm bảo rằng không có quá trình nào có thể đạt được điều khiển của CPU mà cuối cùng không trả lại điều khiển. Các thanh ghi điều khiển thiết bị không thể truy xuất tới người dùng vì thế tính đúng đắn của các thiết bị ngoại vi khác nhau được bảo vệ.

Bảo vệ là một cơ chế để điều khiển truy xuất của các chương trình, quá trình hay người dùng tới tài nguyên được định nghĩa bởi một hệ thống máy tính. Cơ chế này phải cung cấp phương tiện để đặc tả các điều khiển được áp đặt và phương tiện cho việc ép buộc.

Bảo vệ có thể cải tiến khả năng tin cậy bằng cách phát hiện các lỗi tiềm tàng tại các giao diện giữa các hệ thống con thành phần. Phát hiện các lỗi giao diện sớm thường có thể ngăn chặn nguy cơ ảnh hưởng tới hệ thống con bởi một hệ thống con khác. Tài nguyên không được bảo vệ không thể ngăn chặn việc sử dụng bởi người dùng không có quyền. Hệ thống hướng bảo vệ (protection-oriented system) cung cấp một phương tiện để phân biệt giữa việc dùng có quyền và không có quyền.

### 1.8 Hệ thống thông dịch lệnh

Một trong những chương trình hệ thống quan trọng nhất đối với hệ điều hành là trình thông dịch lệnh. Nó là giao diện giữa người dùng và hệ điều hành. Một vài hệ điều hành chứa trình thông dịch lệnh trong nhân (kernel). Các hệ điều hành khác nhau như MS-DOS và UNIX xem trình thông dịch lệnh như một chương trình đặc biệt đang chạy khi một công việc được khởi tạo hay khi người dùng đăng nhập lần đầu tiên (trên các hệ thống chia thời).

Nhiều lệnh (commands) được cung cấp tới hệ điều hành bởi các lệnh điều khiển (control statements). Khi một công việc mới được bắt đầu trong hệ thống bó, hay khi

một người dùng đăng nhập tới hệ thống chia thời, một chương trình đọc và thông dịch các câu lệnh điều khiển được thực thi tự động. Chương trình này còn được gọi trình thông dịch thẻ điều khiển (control-card interpreter) hay trình thông dịch dòng lệnh và thường được biết như **shell**. Chức năng của nó đơn giản là: lấy câu lệnh tiếp theo và thực thi nó.

Các hệ điều hành thường khác nhau trong vùng shell, với một trình thông dịch lệnh thân thiện với người dùng làm cho hệ thống có thể chấp nhận nhiều hơn đối với người dùng. Một dạng giao diện thân thiện người dùng là hệ thống trình đơn-cửa sổ trên cơ sở chuột (mouse-based window-and-menu system) được dùng trong Macintosh và Microsoft Windows. Chuột được di chuyển tới vị trí con trỏ chuột trên ảnh hay biểu tượng trên màn hình biểu diễn các chương trình, tập tin, và các hàm hệ thống. Phụ thuộc vào vị trí con trỏ chuột, nhấn một nút trên chuột có thể nạp một chương trình, chọn một tập tin hay thư mục hay kéo xuống một trình đơn chứa các câu lệnh. Các shell mạnh hơn, phức tạp hơn và khó học hơn được đánh giá cao bởi một số người dùng khác. Trong những shell này, các lệnh được đánh vào từ bàn phím được hiển thị trên màn hình hay in ra thiết bị đầu cuối, với phím enter (hay return) chỉ rằng một lệnh hoàn thành và sẵn sàng được thực thi. Shell của MS-DOS và UNIX điều hành theo cách này.

Các câu lệnh giải quyết việc tạo và quản lý quá trình, quản lý xuất/nhập, quản lý việc lưu trữ phụ, quản lý bộ nhớ chính, truy xuất hệ thống tập tin, bảo vệ và mạng.

## II. Các dịch vụ hệ điều hành

Hệ điều hành cung cấp một môi trường cho việc thực thi các chương trình. Nó cung cấp các dịch vụ xác định tới chương trình và tới người dùng của các chương trình đó. Dĩ nhiên, các dịch vụ được cung cấp khác nhau từ hệ điều hành này với hệ điều hành kia nhưng chúng có thể xác định các lớp chung. Các dịch vụ hệ điều hành được cung cấp sự tiện dụng cho người lập trình để thực hiện tác vụ lập trình dễ dàng.

- Thực thi chương trình: hệ thống phải có thể nạp chương trình vào bộ nhớ và chạy chương trình đó. Chương trình phải có thể kết thúc việc thực thi của nó bình thường hay không bình thường (hiển thị lỗi).

- Thao tác xuất/nhập: một chương trình đang chạy có thể yêu cầu xuất/nhập. Xuất/nhập này có thể liên quan tới tập tin hay thiết bị xuất/nhập. Đối với các thiết bị cụ thể, các chức năng đặc biệt có thể được mong muốn (như quay lại từ đầu một ổ băng từ, hay xoá màn hình). Đối với tính hiệu quả và tính bảo vệ, người dùng thường không thể điều khiển các thiết bị xuất/nhập trực tiếp. Do đó, hệ điều hành phải cung cấp một phương tiện để thực hiện xuất/nhập..

- Thao tác hệ thống tập tin: hệ thống tập tin có sự quan tâm đặc biệt. Các chương trình cần đọc từ và viết tới các tập tin. Chương trình cũng cần tạo và xoá tập tin bằng tên.

- Giao tiếp: trong nhiều trường hợp, một quá trình cần trao đổi thông tin với các quá trình khác. Giao tiếp như thế có thể xảy ra trong hai cách chính. Cách đầu tiên xảy ra giữa các quá trình được thực thi trên cùng máy tính; Cách thứ hai xảy ra giữa hai quá trình đang được thực thi trên các máy tính khác nhau được kết nối với nhau bởi một mạng máy tính. Các giao tiếp có thể được thực

hiện bằng bộ nhớ được chia sẻ, hay bằng kỹ thuật truyền thông điệp, trong đó các gói tin được di chuyển giữa các quá trình bởi hệ điều hành.

○ Phát hiện lỗi: hệ điều hành liên tục yêu cầu nhận biết các lỗi có thể phát sinh. Các lỗi có thể xảy ra trong CPU và phần cứng bộ nhớ (như lỗi bộ nhớ hay lỗi về điện), trong các thiết bị xuất/nhập (như lỗi chặn lẻ trên băng từ, lỗi nối kết mạng, hết giấy in) và trong chương trình người dùng (như tràn số học, cố gắng truy xuất một vị trí bộ nhớ không hợp lệ, dùng quá nhiều thời gian CPU). Đối với mỗi loại lỗi, hệ điều hành nên thực hiện một hoạt động hợp lý để đảm bảo tính toán đúng và không đổi. Ngoài ra, một tập chức năng khác của hệ điều hành tồn tại không giúp người dùng, nhưng đảm bảo các điều hành hữu hiệu của chính hệ thống. Các hệ thống với nhiều người dùng có thể đạt tính hữu hiệu bằng cách chia sẻ tài nguyên máy tính giữa các người dùng.

○ Cấp phát tài nguyên: khi nhiều người dùng đăng nhập vào hệ thống hay nhiều công việc đang chạy cùng lúc, tài nguyên phải được cấp tới mỗi người dùng. Nhiều loại tài nguyên khác nhau được quản lý bởi hệ điều hành. Một số tài nguyên (như chu kỳ CPU, bộ nhớ chính, lưu trữ tập tin) có mã cấp phát đặt biệt, trái lại các tài nguyên khác (như thiết bị xuất/nhập) có mã yêu cầu và giải phóng thường hơn. Thí dụ, xác định cách tốt nhất để dùng CPU, hệ điều hành có các thủ tục định thời biểu CPU. Các thủ tục này xem xét tốc độ CPU, các công việc phải được thực thi, số thanh ghi sẵn dùng và các yếu tố khác. Cũng có các thủ tục cấp phát ổ băng từ để dùng cho một công việc.

Một thủ tục như thế định vị ổ băng từ chưa được dùng và đánh dấu một bảng bên trong để ghi người dùng mới của ổ băng từ. Một thủ tục khác được dùng để xoá bảng đó. Các thủ tục này cũng có thể cấp phát các máy vẽ, modem, các thiết bị ngoại vi khác.

○ Tính toán: chúng ta muốn giữ vết người dùng nào sử dụng bao nhiêu và loại tài nguyên máy tính nào. Giữ vết này có thể được dùng để tính toán (tính tiền người dùng) hay đơn giản thống kê sử dụng. Thống kê sử dụng có thể là công cụ có giá trị cho người nghiên cứu muốn cấu hình lại hệ thống để cải tiến các dịch vụ tính toán.

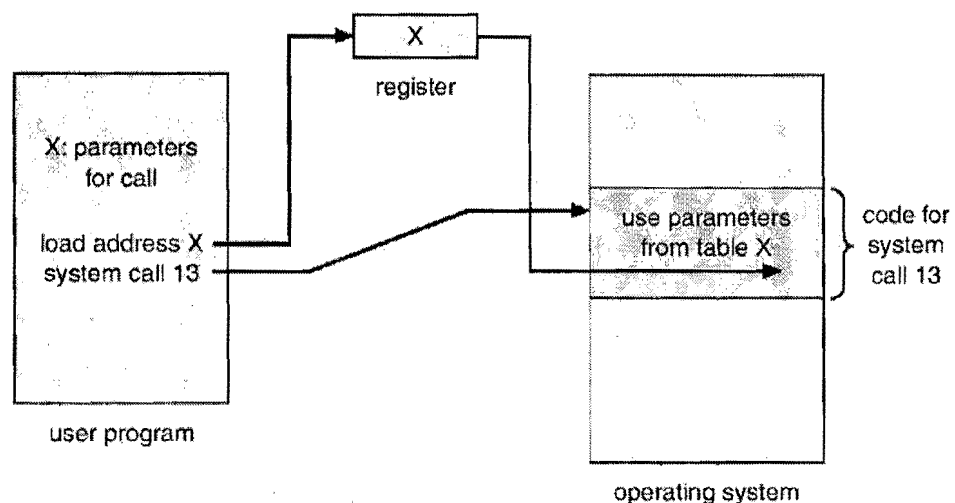
○ Bảo vệ: người sở hữu thông tin được lưu trong hệ thống máy tính đa người dùng muốn điều khiển thông tin này. Khi nhiều quá trình riêng rẽ thực thi đồng hành, không thể cho một quá trình can thiệp tới các quá trình khác hay tới chính hệ điều hành. Bảo vệ đảm bảo rằng tất cả truy xuất tài nguyên của hệ thống được kiểm soát. An toàn hệ thống từ người dùng bên ngoài cũng là vấn đề quan trọng. An toàn bắt đầu với mỗi người dùng có quyền đối với hệ thống, thường bằng mật khẩu để được phép truy xuất tài nguyên. Mở rộng việc bảo vệ đối với các thiết bị xuất/nhập bên ngoài, bao gồm modem, card mạng từ những truy xuất không hợp lệ, và ghi lại các nối kết để phát hiện đột nhập vào hệ thống. Nếu hệ thống bảo vệ và bảo mật, những cảnh báo phải được thiết lập xuyên suốt.

### 3. Lời gọi hệ thống

Lời gọi hệ thống cung cấp giao tiếp giữa một quá trình và hệ điều hành. Các lời

gọi này thường sẵn dùng như các chỉ thị hợp ngữ và chúng thường được liệt kê trong những tài liệu hướng dẫn sử dụng được dùng bởi những người lập trình hợp ngữ. Những hệ thống xác định cho phép lời gọi hệ thống được thực hiện trực tiếp từ một chương trình ngôn ngữ cấp cao, trong đó các lời gọi thường tương tự lời gọi hàm hay thủ tục được định nghĩa trước. Chúng có thể tạo ra một lời gọi tới một chương trình con tại thời điểm thực thi cụ thể.

Lời gọi hệ thống được dùng nhiều cách khác nhau, phụ thuộc vào máy tính. Thường yêu cầu nhiều thông tin bổ xung khi dùng lời gọi hệ thống. Lượng thông tin khác nhau dựa vào hệ điều hành và lời gọi cụ thể. Thí dụ, để nhập dữ liệu, chúng ta có thể cần xác định tập tin hay thiết bị dùng như nguồn nhập, địa chỉ và chiều dài vùng đệm bộ nhớ mà dữ liệu nhập sẽ được đọc vào. Dĩ nhiên, thiết bị hay tập tin và chiều dài có thể ẩn trong lời gọi.



**Hình 2.1. Truyền tham số như bảng**

Có ba phương pháp thông dụng để truyền tham số tới hệ điều hành. Phương pháp đơn giản nhất là truyền tham số trong các thanh ghi. Trong một vài trường hợp, các tham số thường lưu trữ trong một khối hay bảng trong bộ nhớ và địa chỉ của khối được truyền như một tham số trong thanh ghi (Hình 2.1). Các tham số cũng có thể được thay thế, hay được đẩy vào trong ngăn xếp bởi chương trình, và được lấy ra khỏi ngăn xếp bởi hệ điều hành. Một vài hệ điều hành dùng phương pháp khối hay ngăn xếp vì các phương pháp này không giới hạn số lượng hay chiều dài của tham số đang được truyền.

#### 4. Các chương trình hệ thống

Một khía cạnh khác của hệ thống hiện đại là tập hợp các chương trình hệ thống. Xem lại hình 1.1, nó minh họa cấu trúc phân cấp máy tính logic. Tại cấp thấp nhất là phần cứng. Kế đến là hệ điều hành, sau đó các chương trình hệ thống và cuối cùng là các chương trình ứng dụng. Các chương trình hệ thống cung cấp môi trường thuận lợi cho việc phát triển và thực thi chương trình. Vài trong chúng là các giao diện người dùng đơn giản cho các lời gọi hệ thống; các hệ thống còn lại được xem xét phức tạp hơn. Chúng có thể được chia thành các loại sau:

- **Quản lý tập tin:** các chương trình tạo, xóa, chép, đổi tên, in, kết xuất,



liệt kê, và các thao tác tập tin thư mục thông thường.

- **Thông tin trạng thái:** một vài chương trình đơn giản yêu cầu hệ thống ngày, giờ, lượng bộ nhớ hay đĩa sẵn dùng, số lượng người dùng, hay thông tin trạng thái tương tự. Sau đó, thông tin được định dạng và được in tới thiết bị đầu cuối hay thiết bị xuất khác hoặc tập tin.

- **Thay đổi tập tin:** nhiều trình soạn thảo văn bản có thể sẵn dùng để tạo và thay đổi nội dung của tập tin được lưu trên đĩa hay băng từ.

- **Hỗ trợ ngôn ngữ lập trình:** trình biên dịch, trình hợp ngữ và trình thông dịch cho các ngôn ngữ lập trình thông dụng (như C, C++, Java, Visual Basic và PERL) thường được cung cấp tới người dùng với hệ điều hành.

Hiện nay, một vài chương trình này được cung cấp riêng và có giá cả riêng.

- **Nạp và thực thi chương trình:** một khi chương trình được tập hợp hay được biên dịch, nó phải được nạp vào bộ nhớ để được thực thi. Hệ thống có thể cung cấp bộ nạp tuyệt đối, bộ nạp có thể tái định vị, bộ soạn thảo liên kết và bộ nạp phủ lấp. Các hệ thống gỡ rối cho các ngôn ngữ cấp cao hay ngôn ngữ máy cũng được yêu cầu.

- **Giao tiếp:** các chương trình này cung cấp cơ chế tạo các nối kết ảo giữa các quá trình, người dùng, các hệ thống máy tính khác. Chúng cho phép người dùng gửi các thông điệp tới màn hình của người dùng khác, hiển thị các trang web, gửi thư điện tử, đăng nhập từ xa hay để chuyển các tập tin từ máy tính này tới máy tính khác.

Nhiều hệ điều hành được cung cấp với các chương trình giải quyết các vấn đề giao tiếp thông thường hay thực hiện các thao tác phổ biến. Những chương trình như thế gồm các trình duyệt Web, bộ xử lý văn bản và bộ định dạng văn bản, hệ cơ sở dữ liệu, trình biên dịch, các gói phần mềm đồ họa và phân tích thống kê, trò chơi,... Những chương trình này được gọi là các tiện ích hệ thống hay chương trình ứng dụng.

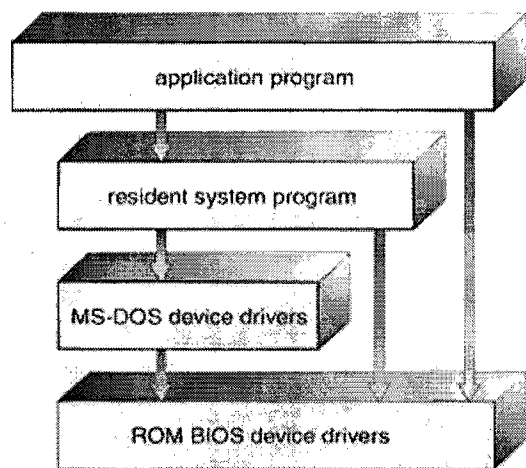
Hầu hết người dùng nhìn hệ điều hành như các chương trình hệ thống hơn các lời gọi hệ thống thực sự. Nghĩ về việc sử dụng một PC. Khi máy tính của chúng ta chạy hệ điều hành Microsoft Windows, chúng ta có thể thấy một trình thông dịch dòng lệnh MS-DOS hay giao diện cửa sổ và trình đơn đồ họa. Cả hai sử dụng cùng một tập lời gọi hệ thống như lời gọi hệ thống trông rất khác và hoạt động trong các cách khác nhau. Do đó, tầm nhìn của chúng ta về thực chất có thể bị tách rời với cấu trúc hệ thống thực sự. Vì thế, thiết kế một giao diện tiện dụng và thân thiện không là một chức năng trực tiếp của hệ điều hành. Trong giáo trình này chúng ta sẽ tập trung các vấn đề cơ bản của việc cung cấp dịch vụ đầy đủ cho các chương trình người dùng. Từ quan điểm hệ điều hành, chúng ta không phân biệt sự khác nhau giữa các chương trình người dùng và các chương trình hệ thống.

## 5. Cấu trúc hệ thống

Một hệ thống lớn và phức tạp như một hệ điều hành hiện đại phải được xây dựng cẩn thận nếu nó thực hiện chức năng hợp lý và được hiệu chỉnh dễ dàng. Một phương pháp thông thường là chia tác vụ thành các thành phần nhỏ hơn là có một hệ thống nguyên khối. Mỗi modules này nên là một thành phần hoàn toàn xác định với

nhập, xuất, chức năng được định nghĩa cẩn thận. Trong phần này chúng ta sẽ thảo luận về cách thức mà các thành phần được nối kết và trộn lẫn trong một nhân.

5.1 Cấu trúc đơn giản



Hình 2.2. Cấu trúc phân tầng của MS-DOS

(the users)		
shells and commands compilers and interpreters system libraries		
system-call interface to the kernel		
signals terminal handling character I/O system terminal drivers	file system swapping block I/O system disk and tape drivers	CPU scheduling page replacement demand paging virtual memory
kernel interface to the hardware		
terminal controllers terminals	device controllers disks and tapes	memory controllers physical memory

Hình 2.3. Cấu trúc hệ thống của UNIX

Nhiều hệ thống thương mại không có kiến trúc rõ ràng. Thường các hệ điều hành như thế được bắt đầu như các hệ thống nhỏ, đơn giản và có giới hạn. Sau đó chúng lớn lên ngoài giới hạn mã ban đầu của chúng.

MS-DOS là một thí dụ cho hệ thống dạng này. Ban đầu, nó được thiết kế và thực hiện bởi một vài người mà họ không tưởng rằng chúng sẽ trở nên quá phổ biến. Nó được viết để cung cấp các khả năng nhiều nhất trong không gian ít nhất (vì bị giới hạn bởi phần cứng mà nó đang chạy) vì nó không được phân chia thành các modules một cách cẩn thận. Hình 2.2, hiển thị cấu trúc của nó.

UNIX là một hệ điều hành khác mà ban đầu nó bị giới hạn bởi chức năng phần cứng. Nó chứa hai phần có thể tách rời nhau: nhân và các chương trình hệ thống.

Nhân lại được chia thành một loạt các giao diện và trình điều khiển thiết bị mà chúng được thêm vào và mở rộng qua nhiều năm khi UNIX được cải tiến. Chúng ta có thể hiển thị hệ điều hành UNIX truyền thống khi nó được phân tầng như hình 2.3. Mọi

thứ bên dưới giao diện lời gọi hệ thống và bên trên phần cứng vật lý là nhân. Nhân cung cấp hệ thống tập tin, bộ định thời CPU, quản lý bộ nhớ và các chức năng khác của hệ điều hành thông qua lời gọi hệ thống. Có rất nhiều chức năng được nối kết trong cấp thứ nhất. Điều này làm cho UNIX khó có thể nâng cấp khi những thay đổi trong phần một ảnh hưởng bất lợi cho những phần khác.

Lời gọi hệ thống định nghĩa giao diện lập trình ứng dụng (API-Application Programming Interface) cho UNIX; tập hợp các chương trình hệ thống thường sử dụng định nghĩa giao diện người dùng. Người lập trình và giao diện người dùng định nghĩa ngữ cảnh mà nhân phải hỗ trợ. Những ấn bản mới của UNIX được thiết kế để dùng phần cứng tiên tiến hơn.

Được cung cấp sự hỗ trợ phần cứng hợp lý, các hệ điều hành có thể được chia thành nhiều phần nhỏ hơn và phù hợp hơn là các hệ thống MS-DOS và UNIX ban đầu. Sau đó, các hệ điều hành có thể giữ lại việc điều khiển lớn hơn qua máy tính và qua các ứng dụng thực hiện việc sử dụng máy tính đó. Những người cài đặt thoải mái hơn trong việc thực hiện những thay đổi các hoạt động bên trong của hệ thống và trong việc tạo các hệ điều hành theo module. Dưới phương pháp từ trên-xuống (top-down), toàn bộ các chức năng và đặc điểm được xác định và được chia thành các thành phần. Sự phân chia này cho phép các người lập trình che giấu thông tin; do đó họ tự do cài đặt các thủ tục cấp thấp khi họ thấy thích hợp, được cung cấp các giao diện bên ngoài của các thủ tục không bị thay đổi do chính thủ tục đó thực hiện các tác vụ thông thường.

System calls					Interrupts and traps		
Terminal handing		Sockets	File naming	Map-ping	Page faults	Signal handling	Process creation and termination
Raw tty	Cooked tty	Network protocols	File systems	Virtual memory			
	Line disciplines	Routing	Buffer cache	Page cache		Process scheduling	
Character devices		Network device drivers	Disk device drivers		Process dispatching		
Hardware							

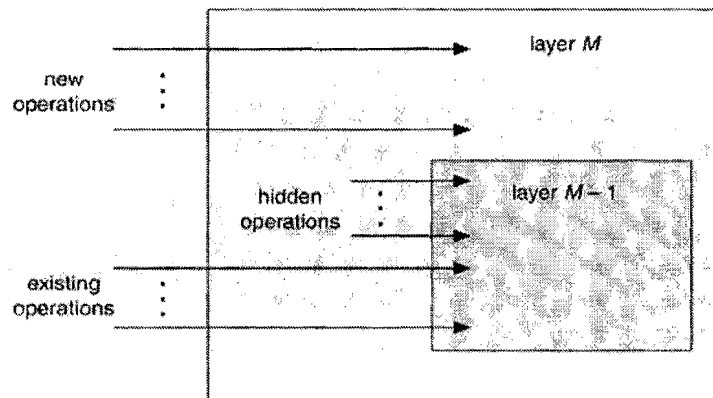
Hình 2.4. Nhân của hệ điều hành UNIX

## 5.2. Phương pháp phân tầng

Việc phân chia từng phần của một hệ thống có thể được thực hiện trong nhiều cách. Một trong những phương pháp này là thực hiện tiếp cận phân tầng. Trong tiếp cận này hệ điều hành được chia thành nhiều tầng (hay cấp), mỗi tầng được xây dựng trên đỉnh của tầng dưới nó. Tầng cuối cùng (tầng 0) là phần cứng; tầng cao nhất (tầng

N) là giao diện người dùng.

Một tầng hệ điều hành là sự cài đặt của một đối tượng trừu tượng. Đối tượng trừu tượng này là sự bao gói dữ liệu và các điều hành có thể thao tác dữ liệu đó. Một tầng hệ điều hành điển hình –tầng M- được mô tả trong hình 2.5. Nó chứa các cấu trúc dữ liệu và tập hợp các thủ tục có thể được gọi bởi các tầng cấp cao hơn. Sau đó, tầng M có thể gọi các thao tác trên tầng cấp thấp hơn.



**Hình 2.5. Một tầng hệ điều hành**

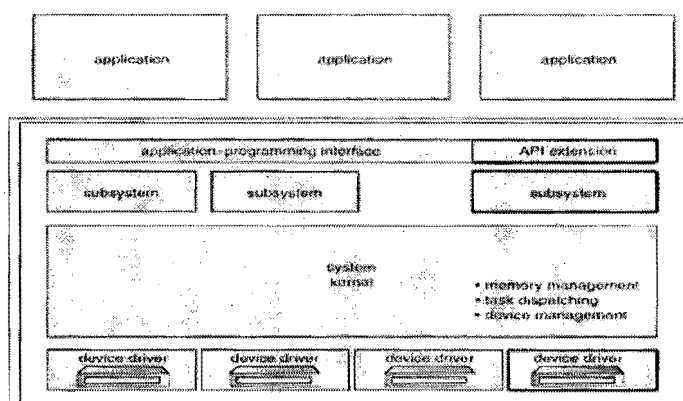
Lợi điểm chủ yếu của tiếp cận phân tầng là tính module. Các tầng được chọn dựa trên cơ sở tầng trên sử dụng chức năng (hay các điều hành) và các dịch vụ chỉ của tầng cấp dưới nó. Tiếp cận này đơn giản hóa việc gỡ rối và kiểm tra hệ thống. Tầng đầu tiên có thể được gỡ rối mà không có bất cứ sự quan tâm nào cho phần còn lại của hệ thống. Bởi vì theo định nghĩa, nó chỉ sử dụng phần cứng cơ bản để cài đặt các chức năng của nó. Một khi tầng đầu tiên được gỡ rối, chức năng sửa lỗi của nó có thể được đảm đương trong khi tầng thứ hai được gỡ rối, ... Nếu một lỗi được tìm thấy trong khi gỡ rối cho một tầng xác định, lỗi phải được nằm trên tầng đó vì các tầng bên dưới đã được gỡ rối rồi. Do đó, thiết kế và cài đặt hệ thống được đơn giản hóa khi hệ thống được phân chia thành nhiều tầng.

Mỗi tầng được cài đặt chỉ với các thao tác được cung cấp bởi các tầng bên dưới. Một tầng không cần biết các thao tác được cài đặt như thế nào; nó chỉ cần biết các thao tác đó làm gì. Do đó, mỗi tầng che giấu sự tồn tại của cấu trúc dữ liệu, thao tác và phần cứng từ các tầng cấp cao hơn.

Khó khăn chính của tiếp cận phân tầng liên quan tới việc định nghĩa cẩn thận các tầng vì một tầng chỉ có thể sử dụng các tầng bên dưới nó. Thí dụ, trình điều khiển thiết bị cho không gian đĩa được dùng bởi các giải thuật bộ nhớ ảo phải nằm ở tầng thấp hơn trình điều khiển thiết bị của các thủ tục quản lý bộ nhớ vì quản lý bộ nhớ yêu cầu khả năng sử dụng không gian đĩa.

Các yêu cầu có thể không thật sự rõ ràng. Thường thì các trình điều khiển lưu trữ dữ liệu nằm trên bộ định thời CPU vì trình điều khiển cần phải chờ nhập/xuất và CPU có thể được định thời lại trong thời gian này. Tuy nhiên, trên hệ thống lớn, bộ định thời có thể có nhiều thông tin hơn về tất cả quá trình đang hoạt động hơn là có thể đặt vừa trong bộ nhớ. Do đó, thông tin này có thể cần được hoán vị vào và ra bộ nhớ, yêu cầu thủ tục trình điều khiển lưu trữ dữ liệu nằm bên dưới bộ định thời CPU.

Vấn đề cuối cùng với các cài đặt phân tầng là chúng có khuynh hướng ít hiệu quả hơn các loại khác. Thí dụ, khi chương trình người dùng thực thi thao tác nhập/xuất, nó thực thi một lời gọi hệ thống. Lời gọi hệ thống này được bẫy (trapped) tới tầng nhập/xuất, nó yêu cầu tầng quản lý bộ nhớ, sau đó gọi tầng định thời CPU, sau đó được truyền tới phần cứng. Tại mỗi tầng, các tham số có thể được hiệu chỉnh, dữ liệu có thể được truyền,... Mỗi tầng thêm chi phí cho lời gọi hệ thống; kết quả thực sự là lời gọi hệ thống mất thời gian lâu hơn khi chúng thực hiện trên hệ thống không phân tầng.



**Hình 2.6. Cấu trúc phân tầng của OS/2**

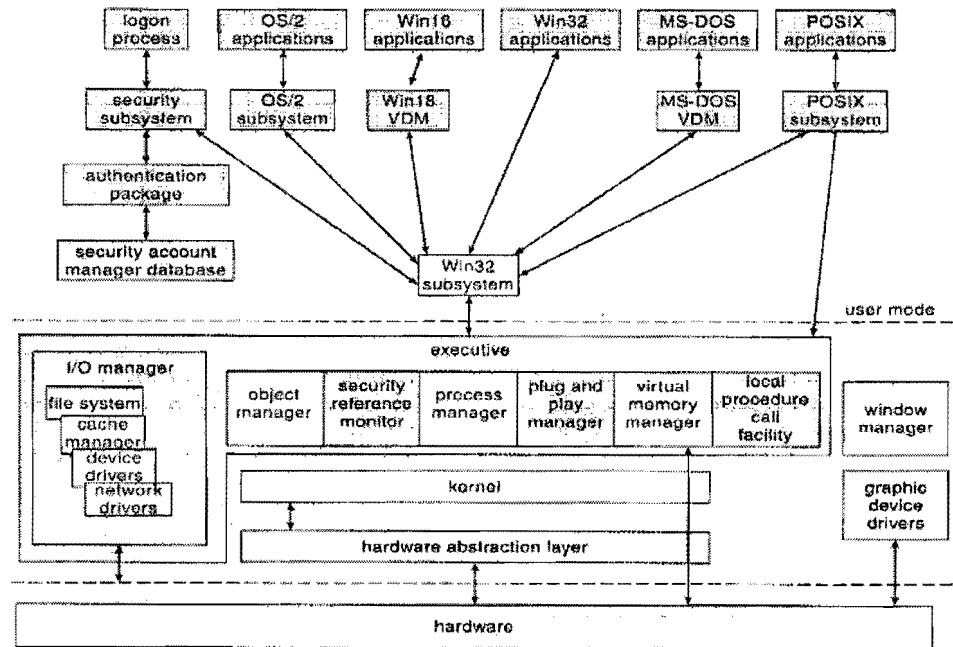
Những giới hạn này gây một phản ứng nhỏ chống lại việc phân tầng trong những năm gần đây. Rất ít các tầng với nhiều chức năng được thiết kế, cung cấp nhiều lợi điểm của mã được module trong khi tránh những vấn đề khó khăn của định nghĩa và giao tiếp tầng. Thí dụ, OS/2 bổ sung thêm tính năng đa tác vụ và điều hành hai chế độ cùng một số đặc điểm mới. Vì tính phức tạp được bổ sung và phần cứng mạnh hơn mà OS/2 được thiết kế, hệ thống được cài đặt trong dạng phân tầng.

Cấu trúc hạt nhân (Kernel Mode) Windows XP là phần ở giữa trong sơ đồ trên. Trong đó, các lớp chính là:

1. HAL (Hardware abstraction layer)
2. Kernel
3. Executive (thí hành)
4. Một số hệ thống phụ (subsystems) và các dịch vụ

Lớp 1, 2, 3 chạy trong chế độ giám sát (kernel mode) còn lớp 4 chạy trong chế độ ứng dụng (user mode). Hệ thống phụ ở chế độ người dùng (user mode) gồm 2 dạng:

- Các hệ thống phụ môi trường (cạnh tranh với các hệ điều hành khác)
- Các hệ thống phụ bảo vệ (cung cấp chức năng bảo vệ)



**Hình 2.7. Cấu trúc phân tầng của hệ điều hành Windows XP**

**Hardware-Abstraction Layer (lớp trừu tượng hóa hệ thống):**

HAL là lớp thuộc phần mềm, che giấu những khác biệt của phần cứng với những cấp trên của hệ điều hành, làm cho WindowsXP có khả năng di chuyển dễ dàng (portable).

Ưu điểm: Chỉ cần duy nhất một phiên bản đơn của mỗi device driver (điều khiển thiết bị) – nó chạy trên tất cả các phần cứng mà không cần mã công của driver.

**Kernel:**

Kernel trong WindowsXP cung cấp cơ sở cho sự thi hành và subsystems.

Kernel gồm 4 chức năng chính:

- + Thread scheduling
- + Interrupt and exception handling
- + Low-level processor synchronization
- + Recovery after a power failure

**\* Kernel Dispatcher:** Kernel dispatcher gồm các nhiệm vụ:

- Thread scheduling
- Implementation of synchronization primitives (Bổ sung các đồng bộ hóa gốc)
- Timer management
- Software interrupts
- Exception dispatching

**\* Threads và Scheduling (lịch):**

- Giống như các hệ điều hành hiện đại khác, WindowsXP cũng sử dụng các process và thread để thực thi code.
- 6 trạng thái rõ ràng của thread là: sẵn sàng, tạm dừng, chạy, chờ, chuyển tiếp, hoàn tất.
- Người gửi sử dụng sự phối hợp quyền ưu tiên 32-cấp để xác định thứ tự thực thi thread. Quyền ưu tiên được chia làm 2 lớp: lớp biến thiên và lớp thời gian thực.
- WindowsXP không phải là một hệ điều hành thời gian thực chính xác, nó không đảm bảo rằng thread thời gian thực sẽ bắt đầu để thực thi trong vòng một giới hạn thời gian riêng biệt.

**\* Implementation of synchronization primitives :**

- Mấu chốt cấu trúc dữ liệu hệ điều hành được quản lý thành các đối tượng sử dụng các điều kiện để dàng cho sự phân phối, đếm thăm quyền và bảo mật.
- Các đối tượng bao gồm:
  - Dispatcher objects (thu phát)
  - Event object (sự kiện)
  - Semaphore object (cờ hiệu)
  - Thread object (luồng)
  - Process object
  - Timer object (thiết bị bấm giờ)

**\* Ngắt phần mềm:** Thủ tục gọi không đồng thời và trì hoãn. Người gửi bỏ sung hai dạng của ngắt phần mềm:

- Asynchronous procedure calls (APCs) ngắt thành một thread và gọi một thủ tục.
- Deferred procedure calls (DPCs) dùng để trì hoãn tiến trình ngắt.

**\* Ngoại lệ và Ngắt:**

- Người gửi cũng cung cấp trap handling cho các ngoại lệ và các ngắt phát sinh bằng phần cứng hoặc phần mềm.
- WindowsXP định nghĩa vài ngoại lệ có cấu trúc không phụ thuộc, bao gồm:
  - Memory-access violation
  - Integer overflow
  - Floating-point overflow or underflow
  - Integer divide by zero
  - Floating-point divide by zero

- Illegal instruction
- Data misalignment
- Privileged instruction
- Page-read error
- Access violation
- Paging file quota exceeded
- Debugger breakpoint
- Debugger single step

Có 32 cấp độ yêu cầu ngắt (IRQLs) trong WindowsXP. 8 cấp độ được dự trữ để dùng bởi kernel; 24 cấp độ còn lại thay mặt cho các ngắt phần cứng thông qua HAL (mặc dù hầu hết các hệ thống IA32 đều dùng 16 cấp độ). Các ngắt của WindowsXP được định nghĩa ở bảng sau:

interrupt levels	types of interrupts
31	machine check or bus error
30	power fail
29	interprocessor notification (request another processor to act; e.g., dispatch a process or update the TLB)
28	clock (used to keep track of time)
27	profile
3-26	traditional PC IRQ hardware interrupts
2	dispatch and deferred procedure call (DPC) (kernel)
1	asynchronous procedure call (APC)
0	passive

Windows XP interrupt request levels.

**Executive (Thực thi):**

Tính thực thi trong WindowsXP cung cấp một bộ các dịch vụ mà tất cả các subsystems môi trường sử dụng. Các dịch vụ được nhóm lại thành:

- Quản lý đối tượng;
- Quản lý bộ nhớ ảo;
- Quản lý tiến trình;
- Quản lý I/O;
- Quản lý cache;
- Bộ kiểm tra tham chiếu bảo mật;
- Quản lý plug-and-play và bảo mật;
- Đăng kí;
- Booting.

**Một số hệ thống phụ (subsystems) và các dịch vụ:**

- OS/2 system



- Win16 application
- Win32 application: Với Win32 là môi trường ứng dụng chính của Windows XP. Các ứng dụng trên môi trường Win32 bao gồm các ứng dụng Windows 32-bit chuyên viết cho Windows 95, Windows 98, Windows NT và Windows 2000, Windows XP.
- POSIX application: POSIX là các chuẩn để viết các ứng dụng chạy trên máy UNIX khác nhau. Các ứng dụng trên môi trường POSIX bao gồm các ứng dụng được phát triển dựa trên các chuẩn này
- MS-DOS application.
- Hệ thống con Win32 hỗ trợ VDM ( Virtual DOS Machine)

Mỗi chương trình Win32, POSIX, OS/2 chạy trên các không gian nhớ phân biệt. Vì thế, mỗi chương trình bị treo, các chương trình khác không bị ảnh hưởng. Mỗi hệ thống phụ của Windows XP cung cấp môi trường có thể chạy và hỗ trợ được các ứng dụng của nhiều hệ điều hành khác nhau.

### 5.3 Vi nhân (Microkernels)

Khi hệ điều hành UNIX được mở rộng, nhân trở nên lớn và khó quản lý. Vào giữa những năm 1980, các nhà nghiên cứu tại đại học Carnegie Mellon phát triển một hệ điều hành được gọi là Mach mà module hóa nhân dùng **tiếp cận vi nhân** (micro kernel). Phương pháp này định kiến trúc của hệ điều hành bằng xóa tất cả thành phần không quan trọng từ nhân và cài chúng như các chương trình cấp người dùng và hệ thống. Kết quả này làm cho nhân nhỏ hơn. Có rất ít sự nhất trí liên quan đến việc quyết định dịch vụ nào nên để lại trong nhân và dịch vụ nào nên được cài đặt trong không gian người dùng. Tuy nhiên, thường thì các vi nhân điển hình cung cấp quá trình và quản lý bộ nhớ tối thiểu ngoài phương tiện giao tiếp.

Chức năng chính của vi nhân là cung cấp tiện nghi giao tiếp giữa chương trình khách hàng và các dịch vụ khác mà chúng đang chạy trong không gian người dùng. Giao tiếp được cung cấp bằng truyền thông điệp. Thí dụ, nếu chương trình khách hàng muốn truy xuất một tập tin, nó phải giao tiếp với trình phục vụ tập tin (file server).

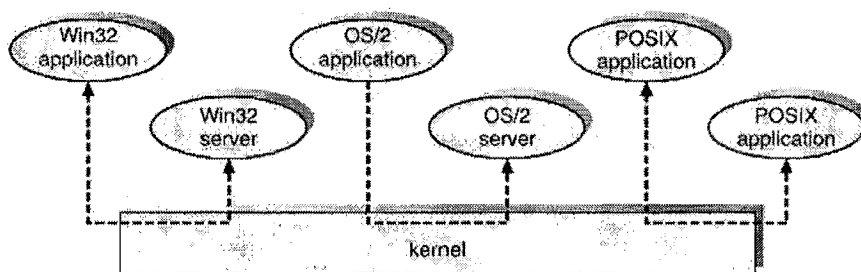
Chương trình người dùng và dịch vụ không bao giờ giao tiếp trực tiếp. Đúng hơn là chúng giao tiếp gián tiếp bằng cách truyền thông điệp với vi nhân.

Thuận lợi của tiếp cận vi nhân là dễ dàng mở rộng hệ điều hành. Tất cả dịch vụ mới được thêm tới không gian người dùng và do đó không yêu cầu phải hiệu chỉnh nhân. Kết quả là hệ điều hành dễ dàng hơn để chuyển đổi từ thiết kế phần cứng này sang thiết kế phần cứng khác. Vi nhân cũng cung cấp khả năng an toàn và tin cậy hơn vì hầu hết các dịch vụ đang chạy như người dùng – hơn là nhân – các quá trình. Nếu một dịch vụ bị lỗi, phần còn lại của hệ điều hành vẫn không bị ảnh hưởng.

Một số hệ điều hành hiện đại dùng tiếp cận vi nhân. Tru64 UNIX (Digital UNIX trước đây) cung cấp giao diện UNIX tới người dùng, nhưng nó được cài đặt với nhân Mach. Nhân Mach ánh xạ các lời gọi hệ thống vào các thông điệp tới các dịch vụ cấp người dùng tương ứng. Hệ điều hành Apple MacOS Server được dựa trên cơ sở nhân Mach. QNX là hệ điều hành thời thực cũng dựa trên cơ sở thiết kế vi nhân. Vi nhân QNX cung cấp các dịch vụ cho việc truyền thông điệp và định thời quá trình. Nó

cũng quản lý giao tiếp mạng cấp thấp và các ngắt phần cứng. Tất cả dịch vụ khác trong QNX được cung cấp bởi các quá trình chuẩn chạy bên ngoài nhân trong chế độ người dùng.

Windows NT dùng một cấu trúc tổng hợp. Windows NT được thiết kế để chạy các ứng dụng khác nhau, gồm Win32 (ứng dụng thuần Windows), OS/2, và POSIX (Portable Operating System Interface for uniX). Nó cung cấp một server chạy trong không gian người dùng cho mỗi loại ứng dụng. Các chương trình khách hàng cho mỗi loại ứng dụng chạy trong không gian người dùng. Nhân điều phối việc truyền thông điệp giữa các ứng dụng khách hàng và server ứng dụng. Cấu trúc client-server của Windows NT được mô tả trong hình 2.8



**Hình 2.8.** Cấu trúc client-server của Windows NT

## 6. Máy ảo

### 6.1. Khái niệm

Về mặt khái niệm, một hệ thống máy tính được cấu thành từ các tầng. Phần cứng là cấp thấp nhất trong tất cả hệ thống như thế. Nhân chạy tại cấp kế tiếp dùng các chỉ thị phần cứng để tạo một tập lời gọi hệ thống cho việc sử dụng các tầng bên ngoài. Do đó, các chương trình hệ thống trên nhân có thể dùng các lời gọi hệ thống hay các chỉ thị phần cứng. Trong nhiều trường hợp, các chương trình này không có sự khác biệt giữa hai cách thực hiện. Do đó, mặc dù chúng được truy xuất khác nhau, nhưng cả hai cung cấp chức năng mà chương trình có thể dùng để tạo thậm chí nhiều chức năng tiên tiến hơn. Sau đó, các chương trình hệ thống xem phần cứng và các lời gọi hệ thống như chúng đang ở cùng một cấp.

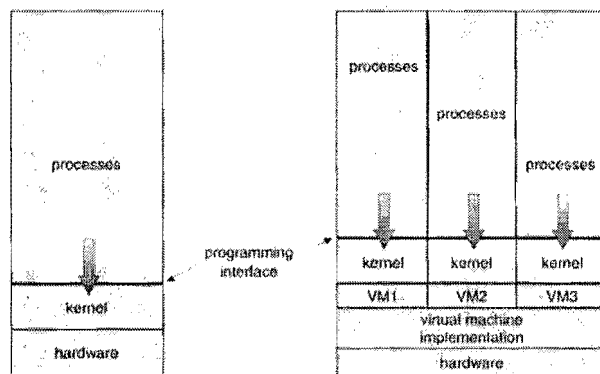
Một vài hệ thống thực hiện cơ chế này một cách chi tiết hơn bằng cách cho phép các chương trình hệ thống được gọi dễ dàng bởi các chương trình ứng dụng.

Trước đó, mặc dù các chương trình hệ thống ở tại cấp cao hơn các thủ tục khác, nhưng các chương trình ứng dụng có thể hiển thị mọi thứ dưới chúng trong cấu trúc phân cấp như là một phần của chính máy đó. Tiếp cận phân tầng này được đưa đến một kết luận luận lý trong khái niệm **máy ảo** (virtual machine). Một hệ điều hành máy ảo cho các hệ thống IBM là một thí dụ điển hình nhất về khái niệm máy ảo vì IBM tiên phong thực hiện trong lĩnh vực này.

Bằng cách sử dụng bộ định thời CPU và kỹ thuật bộ nhớ ảo, một hệ điều hành có thể tạo một hình ảnh mà một quá trình có bộ xử lý của chính nó với bộ nhớ (ảo) của chính nó. Dĩ nhiên, thường thì một quá trình có các đặc điểm khác nhau, như các lời gọi hệ thống và hệ thống tập tin, mà không được cung cấp bởi phần cứng trợ.

Thêm vào đó, tiếp cận máy ảo không cung cấp bất kỳ chức năng bổ sung nào; nhưng đúng hơn là cung cấp một giao diện giống hệt như phần cứng trợ ở bên dưới. Mỗi quá trình được cung cấp với một bản sao (ảo) của máy tính bên dưới (Hình 2.9.).

Một khó khăn chính với tiếp cận máy ảo liên quan đến hệ thống đĩa. Giả sử rằng máy vật lý có ba ổ đĩa nhưng muốn hỗ trợ bảy máy ảo. Rõ ràng, nó không thể cấp phát một ổ đĩa tới mỗi máy ảo. Nhớ rằng chính phần mềm máy ảo sẽ cần không gian đĩa liên tục để cung cấp bộ nhớ ảo. Giải pháp này cung cấp đĩa ảo, mà nó đúng trong tất cả khía cạnh ngoại trừ kích thước-được thuật ngữ hóa **đĩa nhỏ** (minidisks) trong hệ điều hành máy ảo của IBM. Hệ thống cài đặt nhiều đĩa nhỏ bằng cách cấp phát nhiều rãnh ghi trên đĩa vật lý như là các đĩa nhỏ khi cần. Hiển nhiên, tổng kích thước của tất cả đĩa nhỏ là nhỏ hơn kích thước của không gian đĩa vật lý sẵn có.



**Hình 2.9.** Các mô hình hệ thống. (a) Máy không ảo. (b) máy ảo

Do đó, người dùng được cho máy ảo của chính họ. Sau đó, họ có thể chạy bất kỳ hệ điều hành hay gói phần mềm nào sẵn dùng trên phần cứng bên dưới. Đối với hệ thống IBM VM, một người dùng thường chạy CMS-một hệ điều hành giao tiếp đơn người dùng. Phần mềm máy ảo được quan tâm với đa máy ảo đa chương trên một máy vật lý nhưng không cần xem xét bất cứ phần mềm hỗ trợ người dùng. Việc sắp xếp này có thể cung cấp một sự phân chia hữu ích thành hai phần nhỏ hơn của vấn đề thiết kế một hệ thống giao tiếp đa người dùng.

## 6.2. Cài đặt

Mặc dù khái niệm máy ảo là hữu ích nhưng rất khó cài đặt. Nhiều công việc được yêu cầu cung cấp một bản sao chính xác của máy bên dưới. Máy bên dưới có hai chế độ: chế độ người dùng và chế độ kiểm soát. Phần mềm máy ảo có thể chạy trong chế độ kiểm soát vì nó là hệ điều hành. Chính máy ảo có thể thực thi chỉ trong chế độ người dùng. Tuy nhiên, chỉ khi máy vật lý có hai chế độ thì nó mới là máy ảo. Do đó, chúng ta phải có một chế độ người dùng ảo và một chế độ kiểm soát ảo. Cả hai đều chạy trong chế độ người dùng vật lý. Các hoạt động đó gây ra sự chuyển từ chế độ người dùng tới chế độ kiểm soát trên một máy thật (như lời gọi hệ thống hay cố gắng thực thi một chỉ thị được cấp quyền) cũng phải gây ra sự chuyển đổi từ chế độ người dùng ảo tới chế độ kiểm soát ảo trên một máy ảo.

## 6.3. Thuận lợi

Có hai lợi điểm chính trong việc sử dụng máy ảo. Thứ nhất, bằng cách bảo vệ hoàn toàn các tài nguyên hệ thống, máy ảo cung cấp mức độ bảo mật cao. Thứ hai, máy ảo cho phép phát triển hệ thống được thực hiện mà không cần phá vỡ hoạt động hệ thống thông thường.

Mỗi máy ảo hoàn toàn bị cô lập từ các máy ảo khác, vì thế chúng ta không gặp phải bất kỳ vấn đề bảo mật nào như tài nguyên hệ thống khác hoàn toàn được bảo vệ. Thí dụ, các ứng dụng không được tin cậy được tải về từ Internet có thể được chạy trong một máy ảo riêng. Một bất lợi của môi trường này là không có sự chia sẻ tài nguyên trực tiếp. Hai tiếp cận cung cấp sự chia sẻ được cài đặt. Thứ nhất, có thể chia sẻ một đĩa nhỏ. Cơ chế này được làm mẫu sau một đĩa được chia sẻ vật lý. Thứ hai, có thể định nghĩa một mạng của các máy ảo, mỗi máy ảo có thể gửi thông tin qua các mạng giao tiếp này nhưng nó được cài đặt bằng phần mềm.

Những hệ thống máy ảo như thế là một phương tiện truyền thông hữu hiệu cho việc nghiên cứu và phát triển hệ điều hành. Thông thường, thay đổi một hệ điều hành là một tác vụ khó. Vì các hệ điều hành là các chương trình lớn và phức tạp, sự thay đổi trên một phần này có thể gây một lỗi khó hiểu trong những phần khác. Sức mạnh của hệ điều hành làm cho trường hợp này là cực kỳ nguy hiểm. Vì hệ điều hành thực thi trong chế độ kiểm soát, một thay đổi sai trong một con trỏ có thể gây lỗi và có thể phá hủy toàn hệ thống tập tin. Do đó, cần phải kiểm tra tất cả thay đổi của hệ điều hành một cách cẩn thận.

Tuy nhiên, hệ điều hành chạy trên máy và điều khiển hoàn toàn máy đó. Do đó, hệ thống hiện hành phải bị dừng và ngừng việc sử dụng trong khi những thay đổi được thực hiện và kiểm tra. Thời điểm này thường được gọi là thời gian phát triển hệ thống. Vì nó làm cho hệ thống không sẵn dùng đối với người sử dụng nên thời gian phát triển hệ thống thường được lập thời biểu vào buổi tối hay cuối tuần, khi tải hệ thống thấp.

Một hệ thống máy ảo có thể loại trừ nhiều vấn đề này. Người lập trình hệ thống được cung cấp chính máy ảo của họ, và phát triển hệ thống được thực hiện trên máy ảo thay vì trên máy vật lý thật sự. Một hệ điều hành thông thường ít khi bị phá vỡ vì phát triển hệ thống. Mặc dù những thuận lợi này, nhưng rất ít cải tiến trên kỹ thuật này được thực hiện gần đây.

## 7. Ghi nhớ

Hệ điều hành cung cấp một số dịch vụ. Tại cấp thấp nhất, lời gọi hệ thống cho phép một chương trình đang chạy thực hiện yêu cầu trực tiếp từ hệ thống. Tại cấp cao hơn, trình thông dịch cung cấp cơ chế cho người dùng đưa ra yêu cầu mà không viết chương trình. Các lệnh có thể xuất phát từ tập tin trong suốt thời gian thực thi theo chế độ xử lý theo lô, hay trực tiếp từ bàn phím trong chế độ tương tác hay chia sẻ thời gian. Các chương trình hệ thống được cung cấp để thoả mãn nhiều yêu cầu thông thường của người dùng.

Các loại yêu cầu khác nhau dựa theo cấp yêu cầu. Cấp gọi hệ thống phải cung cấp các chức năng cơ bản, như điều khiển quá trình, quản lý tập tin và thiết bị. Các yêu cầu cấp cao hơn được thoả mãn bởi trình thông dịch lệnh và chương trình hệ thống được dịch vào một chuỗi các lời gọi hệ thống. Các dịch vụ hệ thống có thể được phân cấp thành nhiều loại: điều khiển chương trình, yêu cầu trạng thái, yêu cầu nhập/xuất. Lỗi chương trình được xem xét các yêu cầu ẩn cho dịch vụ.

Một khi dịch vụ hệ thống được định nghĩa, cấu trúc của hệ điều hành được phát triển. Các bảng khác nhau cần được ghi thông tin định nghĩa trạng thái của hệ thống máy tính và trạng thái của công việc hệ thống.

Thiết kế một hệ điều hành mới là công việc rất quan trọng. Thiết kế hệ thống như thứ tự của các tầng hay sử dụng vi nhân được xem là một kỹ thuật tốt. Khái niệm máy ảo thực hiện tiếp cận phân tầng và xem nhân của hệ điều hành và phần cứng như là phần cứng của nó. Các hệ điều hành khác có thể được nạp trên đỉnh của máy ảo.