##### Cơ bản về tiến trình:

a, Định nghĩa tiến trình:

Tiến trình là một dãy thay đổi trạng thái của hệ thống:

+ Xuất phát từ một trạng thái ban đầu

+ Chuyển từ trạng thái này sang trạng thái khác được thực hiện theo yêu cầu nằm trong chương trình của người sử dụng

Tài nguyên tối thiểu cho một tiến trình:

+ Bộ nhớ cho mã chương trình và dữ liệu

+ Các thanh ghi của processor phục vụ cho quá trình thực hiện chương trình

b, Quan hệ giữa tiến trình và chương trình:

+ Chương trình: là một thực thể thụ động (nội dung file trên đĩa)

+ Tiến trình: là thực thể chủ động (bộ đếm lệnh, tập tài nguyên)

Một chương trình có thể được thực hiện bởi các tiến trình khác nhau,với các bộ dữ liệu khác nhau (một chương trình, nhiều tiến trình. Do đó một chương trình có thể gọi tới nhiều tiến trình.

* Tiến trình là sự thực hiện một chương trình

c, Các trạng thái tiến trình:

Trạng thái của tiến trình là một phần trong hoạt động hiện tại của tiến trình, có các trạng thái cơ bản như:

* Khởi tạo (New): Tiến trình đang được khởi tạo
* Sẵn sàng (Ready): Tiến trình đang đợi sử dụng processor vật lý
* Thực hiện (Running): Các câu lệnh của tiến trình đang được thực hiện
* Chờ đợi (Waiting): Tiến trình đang chờ đợi một sự kiện nào đó xuất hiện (sự hoàn thành thao tác vào/ra)
* Kết thúc (Terminated): Tiến trình thực hiện xong

Nếu hệ thống có một Processor, thì luôn luôn chỉ có duy nhất một tiến trình đang ở trạng thái thực hiện, nhưng có thể có nhiều tiến trình ở trạng thái chờ đợi hoặc sẵn sàng.

##### Luồng:

a, Khái niệm về Luồng:

Khái niệm: Là đơn vị sử dụng CPU cơ bản, gồm:

* Định danh luồng (ID Thread)
* Bộ đếm chương trình (Program Counter)
* Tập các thanh ghi (Registers)
* Không gian stack

Chia sẻ cùng các luồng khác trong cùng một tiến trình:

* Đoạn mã lệnh
* Đoạn dữ liệu (đối tượng toàn cục)
* Các tài nguyên hệ điều hành khác (file đang mở)

Các luồng có thể thực hiện cùng đoạn mã với ngữ cảnh (Tập thanh ghi, Bộ đếm chương trình, stack) khác nhau.

Luồng còn được gọi là tiến trình nhẹ (LWP: Lightweight Process). Một tiến trình có ít nhất là một luồng.

Quan hệ giữa tiến trình và luồng:

|  |  |
| --- | --- |
| **Tiến trình** | **Luồng** |
| - Tiến trình có đoạn mã/dữ liệu/heap & các đoạn khác  - Phải có ít nhất một luồng trong mỗi tiến trình  - Các luồng trong phạm vi một tiến trình chia sẻ mã/dữ liệu/heap, vào/ra nhưng có stack và tập thanh ghi riêng  - Thao tác khởi tạo, luân chuyển tiến trình tốn kém  - Bảo vệ tốt do có không gian điạ chỉ riêng  - Khi tiến trình kết thúc, các tài nguyên được đòi lại và các luồng phải kết thúc theo | - Luồng không có đoạn dữ liệu hay heap riêng  - Luồng không đứng riêng mà nằm trong một tiến trình  - Có thể tồn tại nhiều luồng trong mỗt tiến trình. Luồng đầu là luồng chính và sở hữu không gian stack của tiến trình  - Thao tác khởi tạo và luân chuyển luồng không tốn kém  - Không gian điạ chỉ chung, cần phải bảo vệ  - Luồng kết thúc, stack của nó được  thu hồi |

b, Lợi ích của lập trình đa luồng:

* Tăng tính đáp ứng với người dùng: Cho phép chương trình vẫn thực hiện ngay khi một phần đang chờ đợi (block) hoặc đang thực hiện tính toán tăng cường (lengthy operation).
* Chia sẻ tài nguyên:

+ Các luồng chia sẻ bộ nhớ và tài nguyên của tiến trình chứa nó

* Tốt cho các thuật toán song song (sử dụng chung các CTDL)
* Trao đổi giữa các luồng thông qua bộ nhớ chia sẻ

+ Cho phép một ứng dụng chứa nhiều luồng hoạt động trong cùng không gian địa chỉ

* Tính kinh tế:

+ Các thao tác khởi tạo, hủy bỏ và luân chuyển luồng ít tốn kém: Minh họa được tính song song trên bộ đơn VXL do thời gian luân chuyển CPU nhanh (Thực tế chỉ một luồng thực hiện).

* Sử dụng kiến trúc nhiều vi xử lý: Các luồng chạy song song thực sự trên các bộ VXL khác nhau