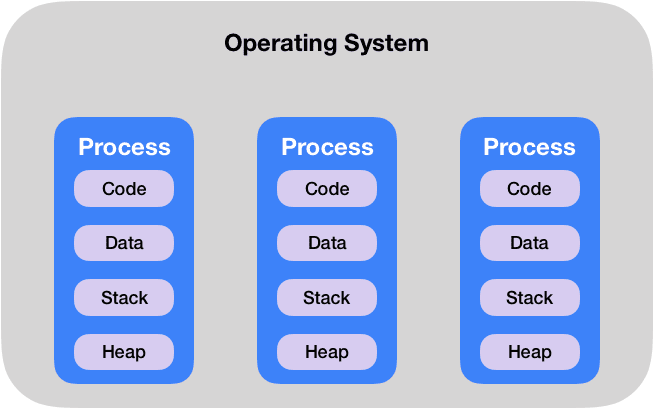
**[프로세스란?]**

1. **프로세스와 스레드**
2. **프로세스(Process)**

: 실행 중인 프로그램으로 디스크로부터 메모리에 적재되어 CPU의 할당을 받을 수 있는 것을 말한다. 운영체제로부터 주소 공간, 파일, 메모리 등을 할당받으며 이것들을 총칭하여 프로세스라고 한다.

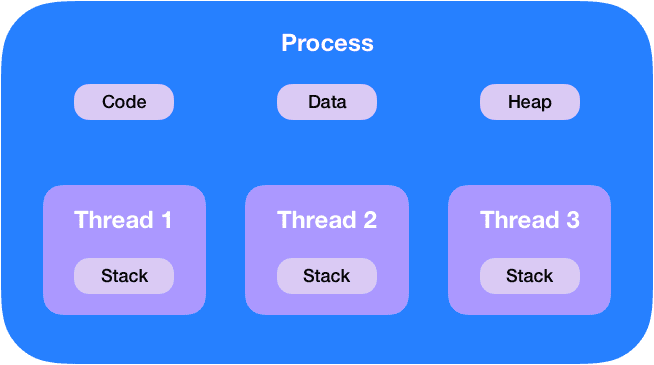


1. **프로세스 제어 블록(Process Control Block, PCB)**

: 특정 프로세스에 대한 중요한 정보를 저장 하고 있는 운영체제의 자료구조이다. 운영체제는 프로세스를 관리하기 위해 프로세스의 생성과 동시에 고유한 PCB를 생성한다. 만약 프로세스가 CPU를 할당받아 작업을 처리하는 도중 프로세스 전환이 발생하면 진행하던 작업의 진행 상황을 모두 PCB에 저장하고 CPU를 반환한다. 그 후 다시 CPU를 할당 받으면 PCB에 저장되어 있던 내용을 불러와 이전에 종료했던 시점부터 다시 작업을 수행한다.

1. **스레드(Thread)**

: 프로세스의 실행 단위이다. 하나의 프로세스 내에서 동작되는 여러 실행 흐름으로 프로세스 내의 주소 공간이나 자원을 공유할 수 있다. 스레드는 스레드 ID, 프로그램 카운터, 레지스터 집합, 스택으로 구성된다.



1. **멀티 스레딩**

: 하나의 프로세스를 다수의 실행 단위로 구분하여 자원을 공유하고 자원의 생성과 관리의 중복성을 최소화하여 수행 능력을 향상시키는 것이다. 이 경우 각각의 스레드는 독립적인 작업을 수행해야하기 때문에 각자의 스택과 PC 레지스터 값을 갖고 있다.

**4-1) 스택을 스레드마다 독립적으로 할당하는 이유**

: 스택은 함수 호출 시 전달되는 인자, 되돌아갈 주소값 및 함수 내에서 선언하는 변수 등을 저장하기 위해 사용되는 메모리 공간이다. 스택 메모리 공간이 독립적이라는 것은 독립적인 함수 호출이 가능하다는 것이고, 이는 독립적인 실행 흐름이 추가되는 것이다.

**4-2) PC 레지스터를 스레드마다 독립적으로 할당하는 이유**

: PC 값은 스레드가 명령어의 어느 단계까지 수행했는지를 나타낸다. 스레드는 CPU를 할당받았다가 스케줄러에 의해 다시 선점당하기 때문에 명령어가 연속적으로 수행되지 못한다. 따라서 어느 단계까지 수행했는지를 기억할 필요가 있다.

1. **멀티 스레딩의 장점**

: 프로세스를 이용하여 동시에 처리하던 일을 스레드로 구현할 경우, 메모리 공간과 시스템 자원 소모가 줄어들게 된다. 스레드 간의 통신은 별도의 자원을 이용하는 것이 아니라 전역 변수의 공간 또는 동적으로 할당된 공간인 Heap 영역을 이용하여 데이터를 주고받기 때문에 프로세스 간 통신 방법에 비해 훨씬 간단하다. 또한, 스레드의 Context Switch 는 프로세스의 Context Switch와 달리 캐시 메모리를 비울 필요가 없기 때문에 속도가 더 빠르다. 따라서 시스템의 Throughput이 향상되고 자원 소모가 줄어들어 프로그램의 응답 시간이 단축되기 때문에 여러 프로세스로 할 수 있는 작업들을 하나의 프로세스에서 스레드로 나누어 수행하는 것이다.

1. **멀티 스레딩의 단점**

: 멀티 프로세스 기반으로 프로그래밍할 때는 프로세스 간 공유하는 자원이 없기 때문에 동일한 자원에 동시에 접근하는 일이 없었지만, 멀티 스레딩을 기반으로 프로그래밍할 때는 서로 다른 스레드가 데이터와 Heap 영역을 공유하기 때문에 다른 스레드에서 사용중인 변수나 자료구조에 접근하여 엉뚱한 값을 읽어오거나 수정할 가능성이 있다. 따라서 멀티 스레딩 환경에서는 동기화 작업이 필요하다. 동기화를 통해 작업 처리 순서를 컨트롤하고 공유 자원에 대한 접근을 컨트롤 하는 것이다. 하지만 이로 인한 병목현상 때문에 성능이 저하될 가능성이 높다. 그러므로 과도한 락으로 인한 병목현상을 줄여야 한다.

1. **멀티 스레드 vs 멀티 프로세스**

: 멀티 스레드는 멀티 프로세스보다 적은 메모리 공간을 차지하고 문맥 전환이 빠르다는 장점이 있지만, 오류로 인해 하나의 스레드가 종료되면 전체 스레드가 종료될 수 있다는 단점과 동기화를 통한 컨트롤 문제를 가지고 있다. 반면 멀티 프로세스 방식은 하나의 프로세스가 죽더라도 다른 프로세스에는 영향을 끼치지 않고 정상적으로 수행된다는 장점이 있지만, 멀티 스레드보다 많은 메모리 공간과 CPU 시간을 차지한다는 단점이 존재한다.

동시에 여러 작업을 수행한다는 점에서 비슷하지만, 분명한 장/단점이 존재하기 때문에 적용해야하는 대상 시스템의 특징에 따라 적합한 방식을 선택하고 적용해야 할 필요가 있다.