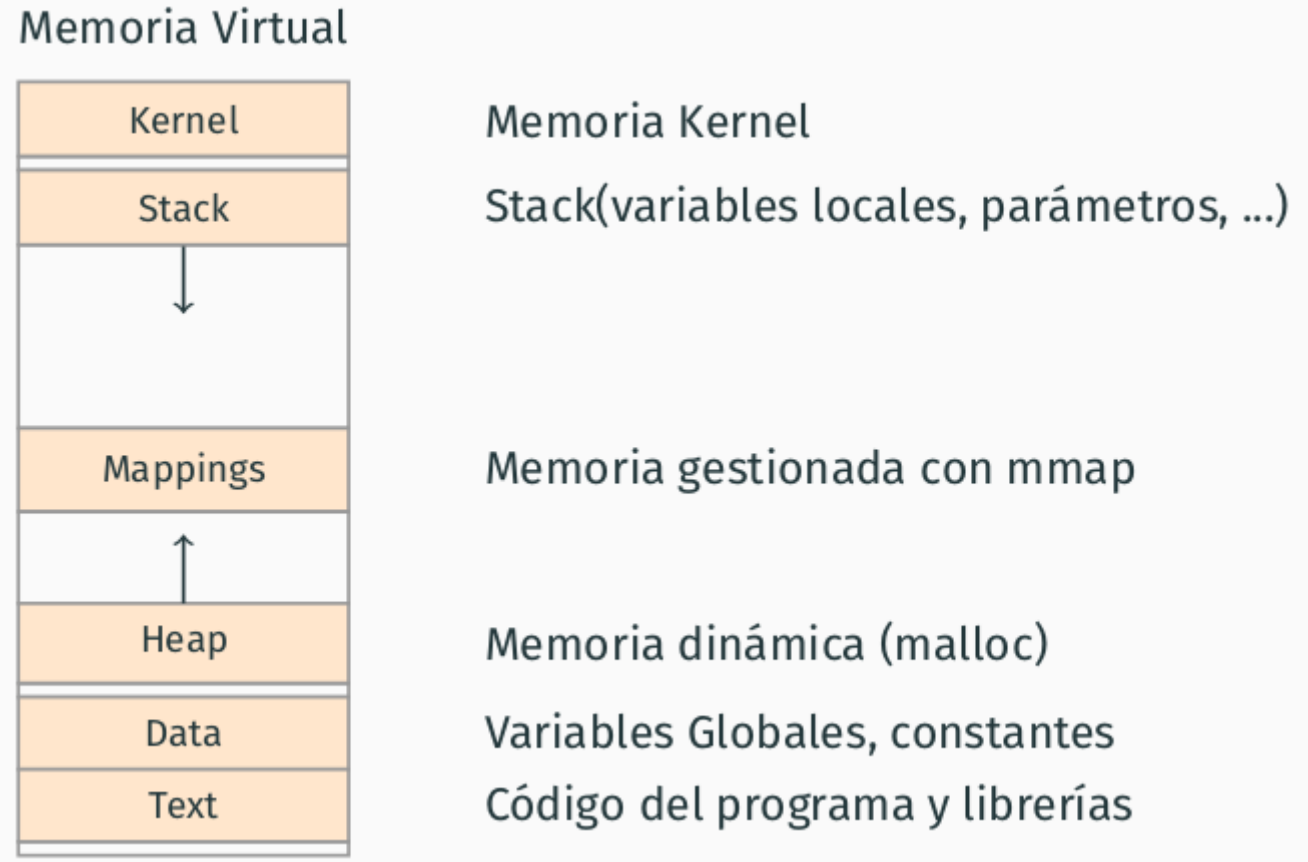
Tema 1-Introducción a concurrencia

Un proceso es una instancia en la ejecución de un programa. Puede haber más de un proceso ejecutándose a la vez para el mismo programa. Cada proceso tiene sus propios recursos (memoria, file descriptors, sockets). Al crear un proceso (fork) se copian los recursos del padre.

La estructura del espacio de direcciones virtual es así:



El código para crear un proceso sería así:

```
#include <stdio.h>
#include <unistd.h>

int pid;
if((pid=fork())==0){
        //proceso hijo
} else{
        //proceso padre
        waitpid(pid,NULL,0);
}
```

Un thread es un hilo de ejecución dentro de un proceso. Los thread del mismo proceso comparten tabla de páginas, file descriptors, sockets y señales. Eso sí, cada thread tiene su propio stack.

La estructura del espacio de direcciones virtual es así:

El código para crear un thread sería así:

```
#include <threads.h>

struct args{
        int i;
        char c;
};

int function(void *p){
         struct args *args=p;
         ...
         return 0;
}

int main(){
        thrd_t thr;
        struct args *args=malloc(sizeof(struct args));
        int result;
        ...
```

```
        thrd_create(&thr, function, args);
        thrd_join(thr, &result);
        free(args);
}
```

Existen distintas técnicas para comunicar threads y procesos:

- Ficheros
- Memoria compartida
- Paso de mensajes
- Pipes
- Colas

Todos los threads comparten las variables globales y estáticas porque solo hay un heap y un área de datos. Se puede acceder a las variables locales de otro si se tiene un puntero a ellas. En cambio los procesos tienen su propia tabla de páginas. Para utilizar zonas de memoria compartida hay que crearlas de forma explícita con mmap. Por ejemplo:

```
#include <sys/mman.h>

int main(){
        char *shared;
        int size=100;
        if((shared=mmap(NULL, size, PROT_READ | PROT_WRITE,
MAP_SHARED))==NULL){
                exit(0);
        }
        if(fork()==0){
                //Proceso hijo
        } else{
                //Proceso padre
        }
}
```