Universidad de Costa Rica

Ciencias de la Computación e Informática

Curso:

Sistemas operativos

Tarea programada 1 - Diseño

Profesora:

Tracy Hernández

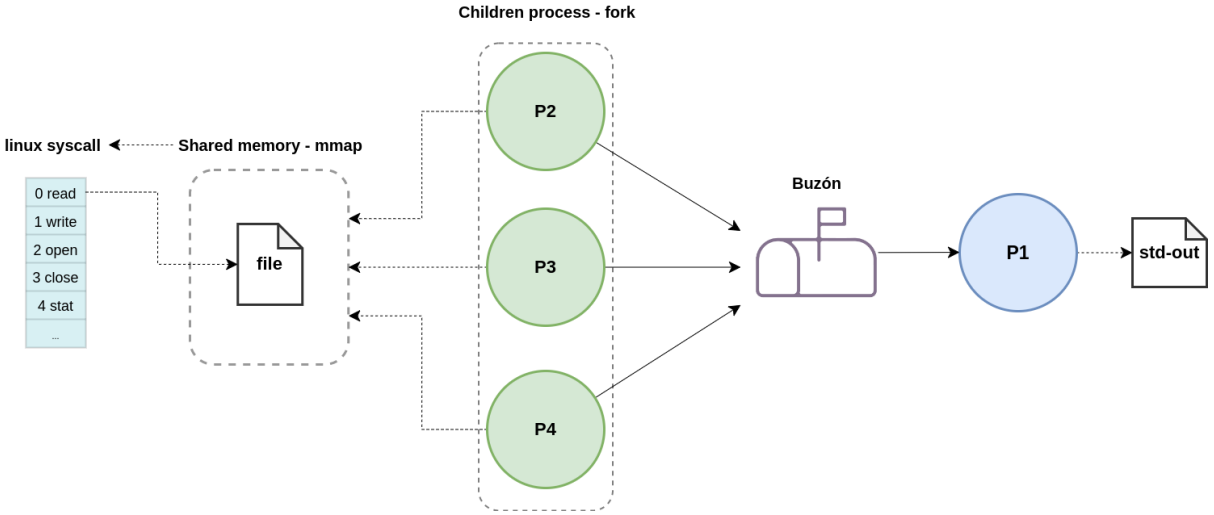
Estudiantes:

Daniel Escobar Giraldo - C02748

Fabián Orozco Chaves - B95690

Diagrama de flujo:

Estructura inicial



Nuestra idea es crear tres procesos hijos del proceso P1. Estos tres procesos se encargan de leer un archivo de texto que fue cargado previamente en una región de memoria compartida. El trabajo de estos tres procesos es de contar las líneas y la cantidad de caracteres (sin espacios) que contiene el archivo. Para esto pensamos dividir en tres secciones el archivo y cada proceso se encarga de procesar una de las tres secciones.

La división de trabajo para cada proceso se hace a partir del tamaño total del archivo. La función mmap() permite saber el tamaño del archivo, a partir de este valor dividiremos el trabajo en 3. Como el archivo es cargado en un arreglo de caracteres gracias a mmap(), pensamos indicarle a cada hilo de qué posición de inicio a que posición final debe procesar el arreglo (con un offset).

Finalmente, los procesos hijos usarán un buzón para comunicar los resultados al archivo padre que se encarga de mostrar el resultado al usuario. Según hemos investigado el buzón está hecho de forma thread safe, por lo tanto no sería necesario la implementación de un semáforo. No obstante, en caso de que surjan problemas con la sincronización de procesos, implementaremos el uso de un semáforo como podemos ver en el siguiente diseño:

Estructura inicial

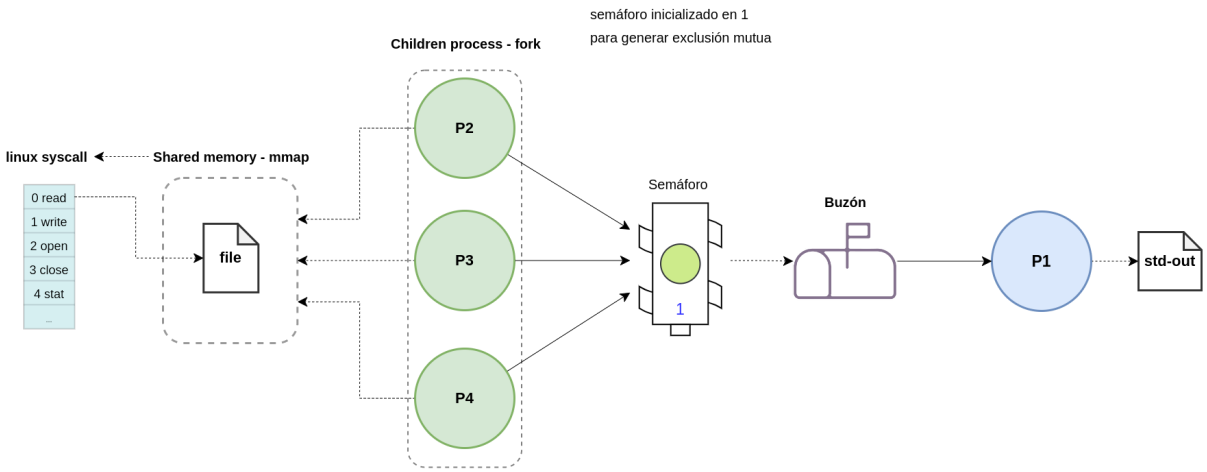
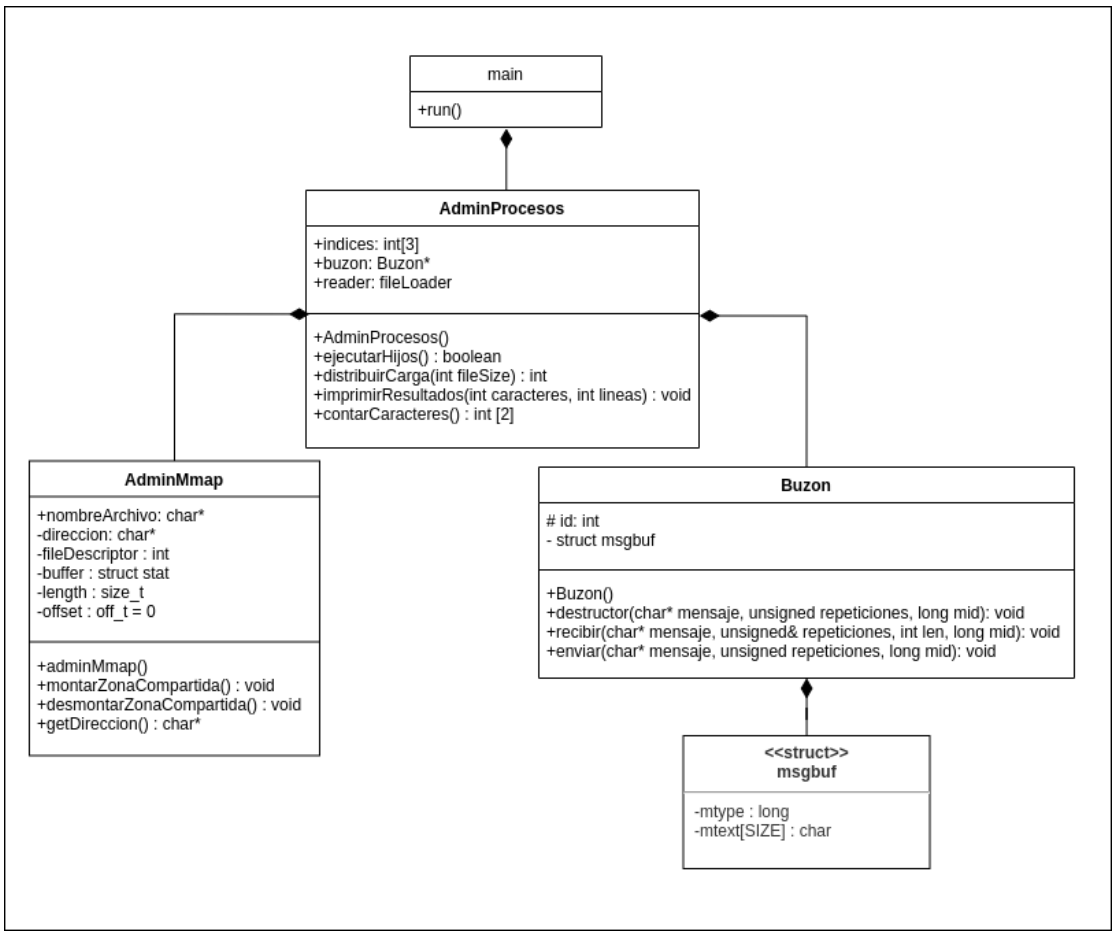


Diagrama UML:



A partir de este diagrama de clases, podemos ver que planeamos tener una clase que se encargue exclusivamente del manejo de la sección de memoria

compartida. Esto toma en cuenta cargar el archivo en la región compartida y destruir la misma al final del programa.

Otra clase que planeamos implementar es la que se encarga del manejo de las comunicaciones a través del buzón. Esta ofrece las primitivas para que los procesos hijos puedan enviar mensajes al padre.

Por último, tenemos la clase AdminProcesos que se encarga de administrar y de asignar el trabajo al proceso padre y a los procesos hijos. Esto toma en cuenta el reparto de trabajo entres los procesos y el flujo general de los datos.

Como mencionamos en el apartado de diagrama de flujo, existe la posibilidad de que sea necesario implementar un semáforo para poder usar correctamente el buzón. Aquí tenemos un diseño de clases que toma en cuenta el uso el semáforo: