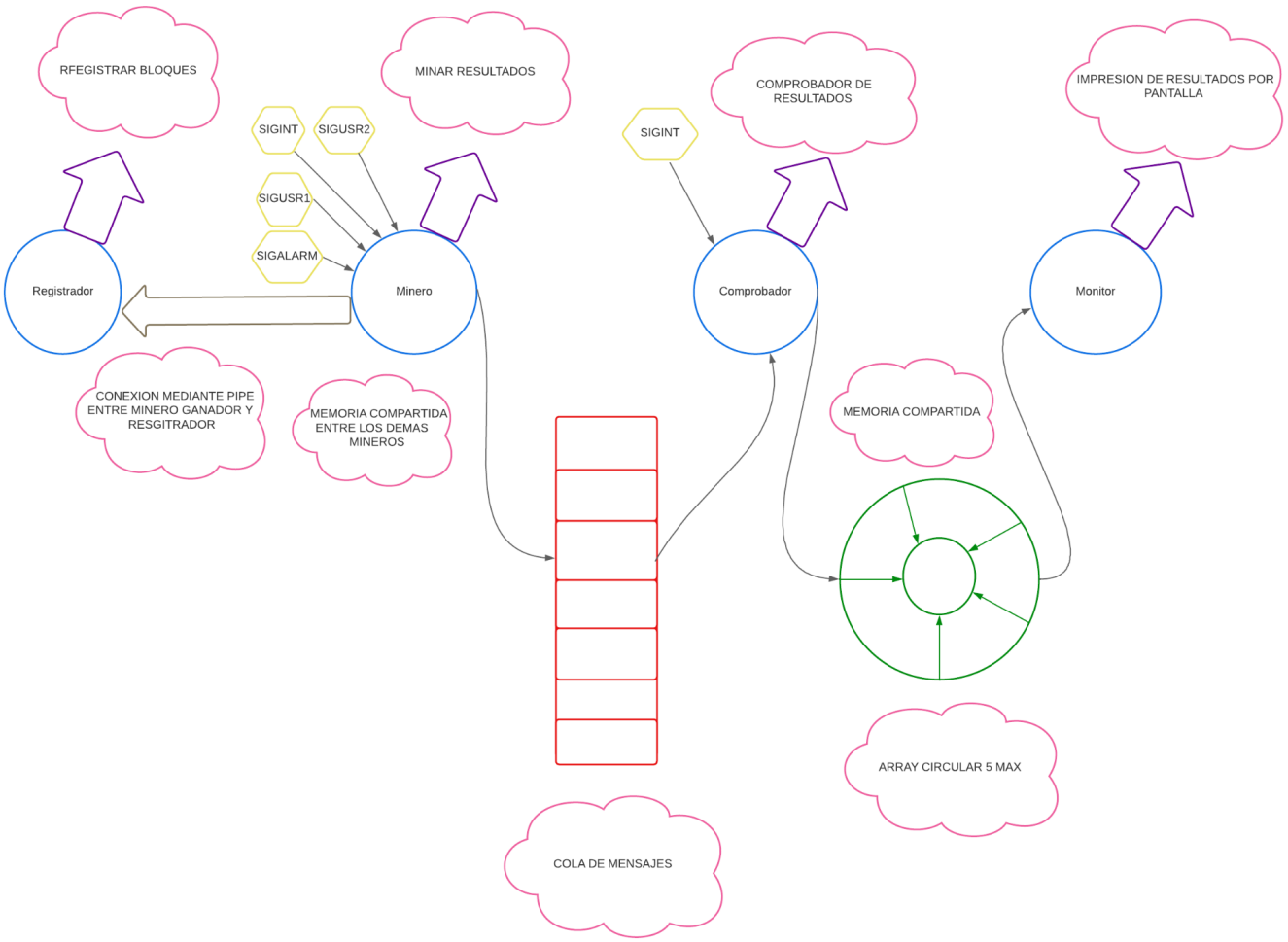
MEMORIA PROYECTO SOPER

Daniel Aquino Santiago y Enrique Gómez Fernández



DISEÑO DEL SISTEMA:

Como vemos en el dibujo expuesto, se pueden diferenciar 4 procesos principales dentro del sistema. El que se encarga de realizar el minado en busca de la solución mediante el número de hilos que se introduzca al iniciar el proceso es el proceso minero. Este, si es el primer minero del sistema, se encargará de abrir la memoria compartida por todos los mineros, a través de la cual estos se comunicarán y harán un sistema de votos. Dado que en la ejecución de estos no se sabe cuántos de ellos están en el sistema, su sincronización ha sido compleja y nos hemos tenido que ayudar de varios semáforos sin nombre como el mutex, utilizado para proteger las regiones críticas del sistema. Todos estos están incluidos en la memoria compartida para su control. A su vez, estos mineros se controlan mediante señales, en las cuales hemos encontrado bastante complejidad a la hora de cortar un minero

cuando llegaba una de estas señales y que el programa siguiera adelante con los demás mineros, pero a base de pruebas, conseguimos lograr que recogiera las señales correctamente y que se saliera correctamente del sistema. La comunicación del minero con el registrador (hijo del minero) se lleva a cabo mediante tuberías. Cada minero, al finalizar la ronda, envía a su proceso hijo(registrador) el bloque hallado a través de esta tubería. El registrador sólo acaba cuando detecta que la tubería ha sido cerrada. Como podemos ver, el minero a su vez se comunica con el proceso comprobador, Esto se lleva a cabo mediante una cola de mensajes, sin embargo, en este caso solo será llevado a cabo por el proceso que haya ganado la ronda. Este proceso comprobador, abre la cola de mensajes, pero como queremos que solo se envíen mensajes a través de la cola si el proceso comprobador está activo, antes de enviar el mensaje en el proceso minero, se comprueba que un semáforo con nombre ha sido abierto, en ese caso, existe un proceso comprobador y monitor por lo que se puede enviar el mensaje. Este proceso ha sido complejo de realizar ya que ha sido difícil coordinar ambos procesos y comunicarlos a la hora de cerrar el sistema. Finalmente optamos por abrir este semáforo con las macros O_CREAT y O_EXCL para que devolviera error si este estaba creado y en ese caso enviar el mensaje. En caso contrario se cerraría de nuevo el semáforo. Como vemos, el proceso monitor y comprobador se comunican mediante una memoria compartida creada y mapeada antes de hacer el fork, ya que sabemos que esta se hereda de proceso padre a proceso hijo. El proceso comprobador enviará el bloque recibido por el minero ganador, lo comprobará y lo escribirá en memoria mediante una cola con distintos semáforos que controlan la lectura y escritura. Además, la parte más compleja de este, es detectar que el comprobador ha de finalizar por una finalización del sistema. Esto se hace mediante la escritura de un mensaje específico en memoria compartida, el cual detecta el y monitor y posteriormente cierran todo lo usado tanto para su ejecución.

LIMITACIONES Y PRUEBAS REALIZADAS:

Para la comprobación del sistema, hemos probado a poner distintas combinaciones de mineros con distintos segundos de finalización y distinto número de hilos. También hemos probado todas las distintas ejecuciones propuestas al final del enunciado de la práctica comprobando en cada una de ellas que tanto monitor como los archivos de texto creados por los registradores se actualizaban correctamente, incluso si en medio de la ejecución algún proceso cortaba mediante un SIGINT. Hemos detectado, que el sistema no puede sobrepasar un determinado número de hilos por minero (o al menos nos ocurre en nuestros ordenadores). Es por eso por lo que hemos limitado la entrada de número de hilos la cual no podrá ser mayor que 3. Otra limitación del sistema es que si ya existe una memoria con el mismo nombre que la que crea el primer minero que llega, todos los mineros quedarán como segundos en llegar por lo que ninguno de ellos enviará SIGUSR1 a los demás. Esto

hace que queden bloqueados en un sigsuspend hasta que se envíe SIGINT o SIGALARM a dicho proceso.