TEC | Tecnológico de Costa Rica

Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación Sede Central

Curso IC-6600
Principios de Sistemas Operativos

Documentación del segundo proyecto de sistemas operativos.

Realizado por:

Olman Castillo Picado 2015148651
Jimmy Fallas Delgado 2015099456
Pablo Navarro Altamirano 2015114121

25 de mayo del 2018

I-Decisiones tomadas

- Tipo de semáforos utilizados para el desarrollo del proyecto: Se decidió utilizar la librería *semaphore.h*, especializada en el manejo de semáforos debido a que permite crear un semáforo que puede ser ingresado desde diferentes procesos, por lo que facilita la administración de los sectores de memoria.

- Cómo se logra la sincronización: para lograr la sincronización se utilizó el modelo de escritores y lectores haciendo que los writer y readers selfish se comporten como escritores, y que lo readers tengan comportamiento de lectores.

- La diferencia de utilizar mmap: al usar mmap se obtienen algunos beneficios a comparación del uso de shmat.
    - Los dos proveen la capacidad de que varios procesos ingresen a un sector de memoria compartido Sin embargo, shmat provee menos cantidad de procesos.
    - La portabilidad del programa es más fácil mediante el uso de mmap.
    - La privacidad del sector puede ser establecida con mmap.

II-Análisis de resultados

Se mostrará una lista de objetos y se definirá tres categorías para determinar el estado del producto. Las categorías son:

- Completada
- Funcionamiento parcial
- Incompleto

En caso de determinar que la sección está incompleta se especificará por qué está incompleta

- General
- Programa inicializador
    - Pedir recursos: completado
    - Parámetro cantidad de líneas: completado
- Programa Writer
    - N escritores: completado
    - Hilo para cada escritor: completado
    - Escribir mensaje: completado
    - Solo uno puede ingresar: completado
    - Dormir por una cantidad de segundos: completado
    - Parámetros segundos de dormir y escribir: completado
- Mensaje
    - Estructura: PID, hora, fecha, línea del archivo: completado

- Programa Reader
    - N Lectores: completado
    - Iniciar el archivo de nuevo: completado
    - Varios pueden leer: completado
    - Parámetros segundos de dormir y leer: completado
- Programa Reader Egoísta
    - N Lectores: completado
    - Selecciona de manera aleatoria la línea: completado
    - Solo uno puede ingresar: completado
    - No más de 3 readers acceden a memoria: incompleto
    - Si no hay nadie más pueden seguir leyendo: incompleto
- Programa finalizador
    - Matar todos los procesos: completado
    - Devolver los recursos: completado
- Programa espia
    - Ver todas la líneas: completado
    - Estado de lo Writers: completado
    - Estado de lo Readers: completado
    - Estado de lo Readers Egoístas: completado
    - Estado: incompleto
        - PID procesos que están en archivo: completado
        - PID procesos que están dormidos: completado
        - PID procesos que están bloqueados: completado
- Bitácora
    - Registro de acciones: completado
    - Información a guardar: completado
        - PID: completado
        - Tipo (reader, writer o reader egoísta): completado
        - hora: completado
        - Mensaje: completado
    - Debe almacenarse en un archivo: completado

III-Aspectos relevantes

El problema fue planteado de tal manera que los proceso tuvieran acceso a dos tipos diferentes de memoria una en la cual se crea el archivo y otro para comunicar el estado actual de los proceso y la información.
Esto se puede representar en la siguiente imagen.

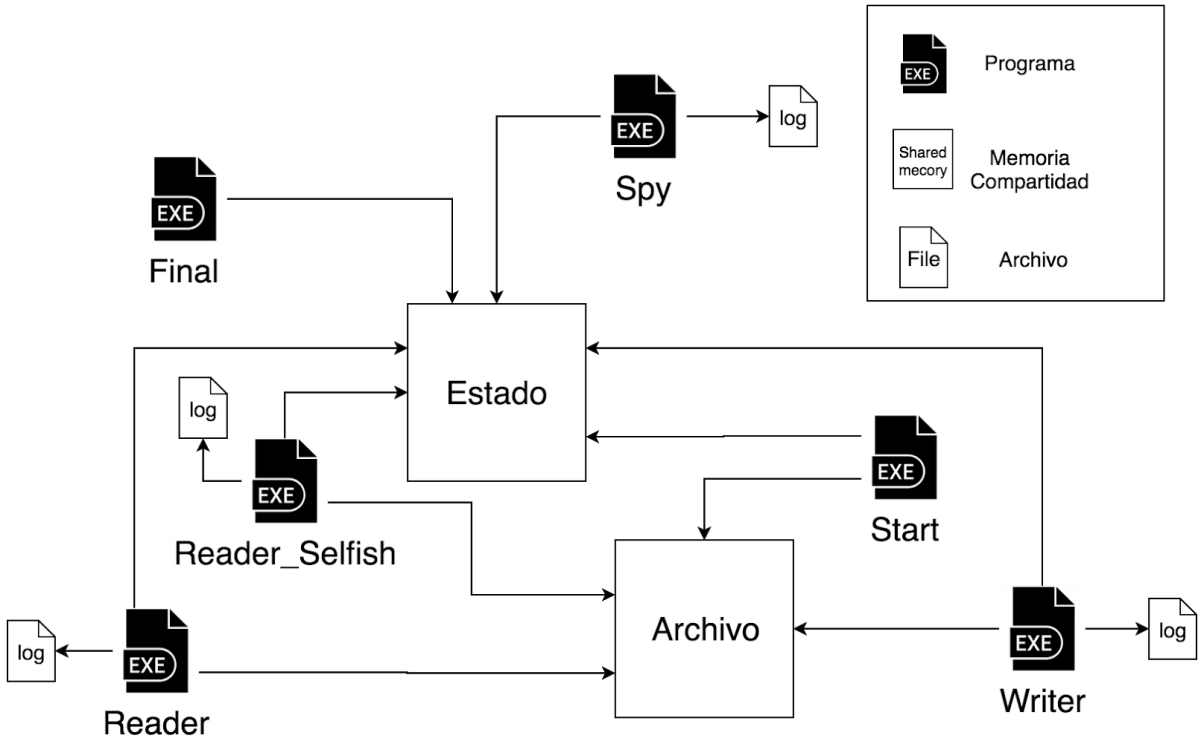


Imagen-1:Resolución del proyecto

Tal como se plantea en la *Imagen 1, todos los procesos se comunicará por medio de dos procesos*:

- **Estado**: este segmento de memoria se encargará de tener los estados de los writer, reader y el reader selfish. Además este sector de memoria tendrá información importante para los procesos.
  A Continuación se representa el cómo se distribuye la memoria de estado.

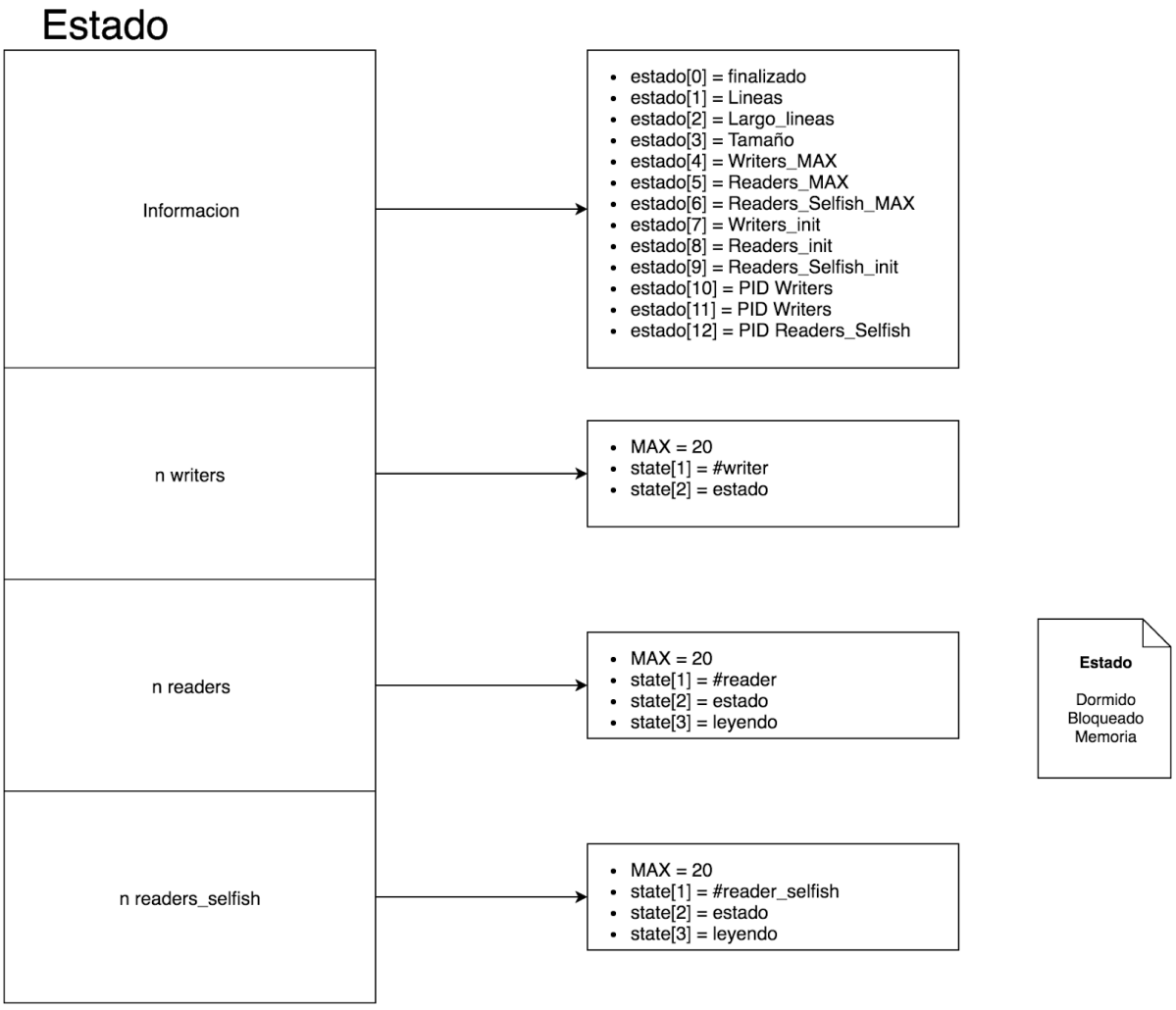


imagen-2:Representación de la lógica del estado.

- **Archivo:** la distribución que se planificó para el archivo es un segmento donde se ingresan n líneas, del mismo tamaño para almacenar la información, que generan los writer y así permitirle al los readers y los reader selfish leer de memoria.
  A Continuación se representa el cómo se distribuye la memoria del archivo.

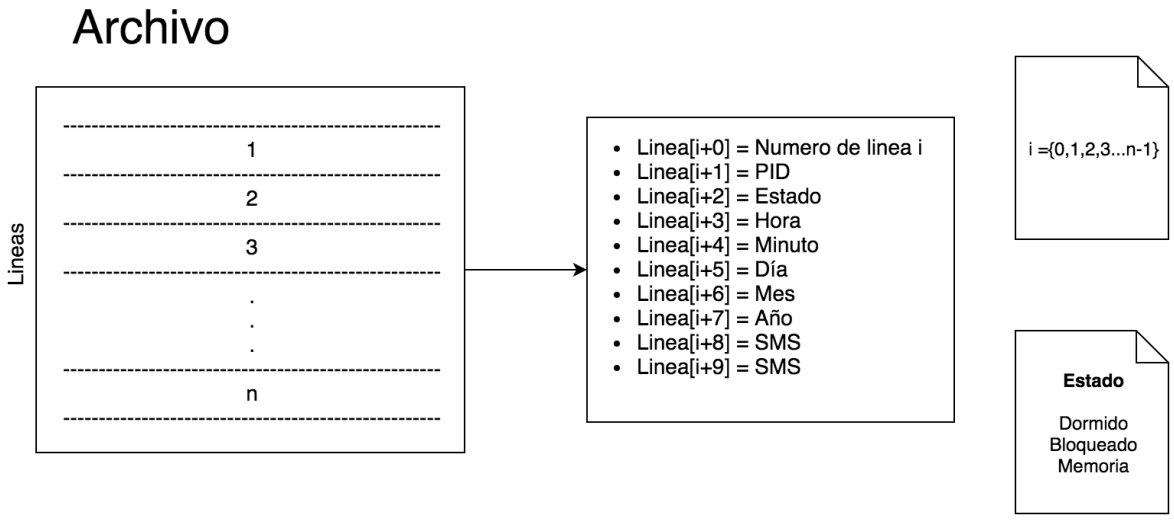


Imagen-3:Representación de la lógica del estado.

Como se muestra en la *imagen-1* los programas writer, reader y reader selfish, generan un archivo log, estos son almacenados en un archivo **.html**, el cual es incluido en el archivo **bitacora.html** que genera el programa finalizar.

IV-Compilación y ejecución de los programas

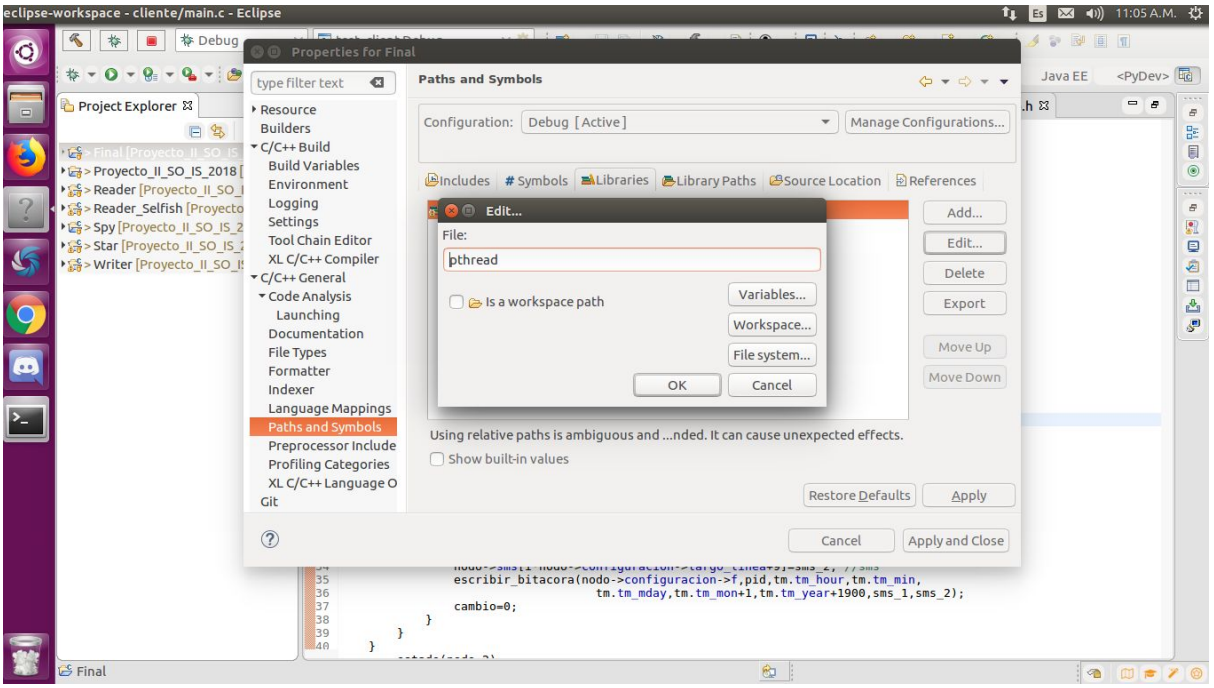
Compilación:
El proyecto fue elaborado usando el entorno de desarrollo:

Eclipse IDE for C/C++ Developers
Version: Oxygen.3 Release (4.7.3RC3)
Build id: 20180301-1623

El IDE Eclipse [3], probe la opcion de compilación automática (Build) la cual fué utilizada para la compilación del proyecto.
La estructura del proyecto generada por el IDE eclipse es la siguiente

1. Start
   ```
   Star
           ->Debug
           ->src
   ```
2. Final
   ```
   Final
           ->Debug
           ->src
   ```
3. Writer
   ```
   Writer
           ->Debug
           ->src
   ```
4. Reader
   ```
   Reader
           ->Debug
           ->src
   ```
5. Reader_Selfish
   ```
   Reader_Selfish
           ->Debug
           ->src
   ```

En la cual se crearía en la parte de Debug el archivo ejecutable luego del proceso de compilación, la carpeta src será la que contenga el código fuente a compilar.
Para la compilación es necesario vincular la librería pthread al proyecto.
Se debe dar click derecho sobre el proyecto ir a la opción
properties -> C/C++ General -> Paths and Symbols -> Libraries -> Add



Luego se presiona ctrl+b y se compila el programa.

Ejecución:

Para la ejecución se usará la terminal de ubuntu, se buscará carpeta Debug donde se encontrará el objeto ejecutable

En el caso del programa inicializador, se debe ingresar el número de líneas.

- Start

```
./Start 20
```

Para el programa finalizador se debe ingresar de la siguiente manera.

- Final

```
./Final
```

Para los programas writer, reader y reader selfish, se ingresaron dos valores el tiempo de escribir y el tiempo de dormir.

- Writer

```
./Writer 2 2
```

- Reader

```
./Reader 2 2
```

- Reader Selfish

```
./Reader_Selfish 2 2
```

Para el programa espía se debe ingresar de la siguiente manera.

- Spy

```
./Spy
```

V-Casos de prueba

1. Primer caso

Objetivo de la prueba: Probar el uso en conjunto de los programas tratando de identificar la características de ejecución deseadas.

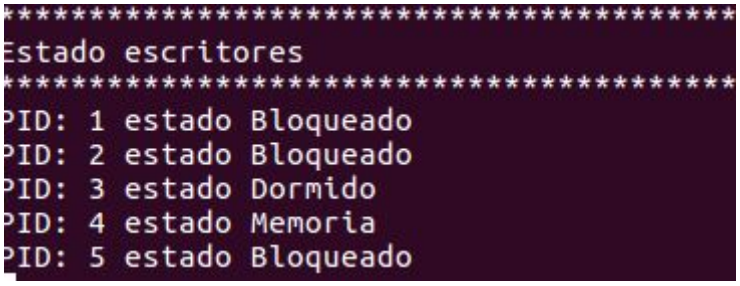
Configuración de los programas

```
./Start 20
./Writer 5 2 2
./Reader 2 2 2
./Reader_Selfish 2 2 2
./Final
```

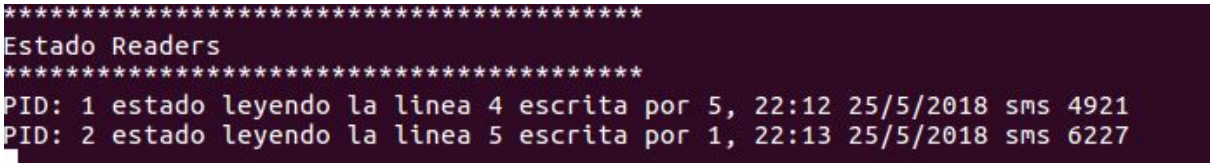
Capturas de los resultados.

Programa inicializador

Programa Writer



Programa Reader



Programa Reader Selfish

```
**************************************
Estado Readers selfish
**************************************
PID: 1 estado leyendo la linea 16 escrita por 5, 22:13 25/5/2018 sms 2258
PID: 2 estado leyendo la linea 16 escrita por 5, 22:13 25/5/2018 sms 2258
```

Programa Spy

Spy líneas

```
****************************************
lineas
****************************************
Linea 0 PID 3 Estado 1 22:0 25/5/2018 SMS 6735
Linea 1 PID 3 Estado 1 22:0 25/5/2018 SMS 7715
Linea 2 PID 1 Estado 1 22:0 25/5/2018 SMS 9335
Linea 3 PID 4 Estado 1 22:0 25/5/2018 SMS 8692
Linea 4 PID 5 Estado 1 22:0 25/5/2018 SMS 4921
Linea 5 PID 4 Estado 1 22:0 25/5/2018 SMS 6227
Linea 6 PID 1 Estado 1 22:1 25/5/2018 SMS 1570
Linea 7 PID 1 Estado 1 22:0 25/5/2018 SMS 6326
Linea 8 PID 2 Estado 1 22:1 25/5/2018 SMS 5673
Linea 9 PID 4 Estado 1 22:0 25/5/2018 SMS 1142
Linea 10 PID 4 Estado 1 22:0 25/5/2018 SMS 1168
Linea 11 PID 5 Estado 1 22:0 25/5/2018 SMS 6729
Linea 12 PID 1 Estado 1 22:0 25/5/2018 SMS 8230
Linea 13 PID 3 Estado 1 22:1 25/5/2018 SMS 1326
Linea 14 PID 1 Estado 1 22:1 25/5/2018 SMS 9180
Linea 15 PID 5 Estado 1 22:0 25/5/2018 SMS 2258
Linea 16 PID 1 Estado 1 22:0 25/5/2018 SMS 6967
Linea 17 PID 1 Estado 1 22:1 25/5/2018 SMS 8437
Linea 18 PID 3 Estado 1 22:0 25/5/2018 SMS 2973
Linea 19 PID 5 Estado 1 22:0 25/5/2018 SMS 2119
```

Spy estado de los programas

```
****************************************
writers
****************************************
PID 1 estado Memoria
PID 2 estado Memoria
PID 3 estado Dormido
PID 4 estado Dormido
PID 5 estado Dormido

****************************************
readers
****************************************
PID 1 estado Dormido
PID 3 estado Normal

****************************************
readers egoistas
****************************************
PID 1 estado Memoria
PID 2 estado Normal
```

Spy procesos bloqueados, dormidos y en memoria

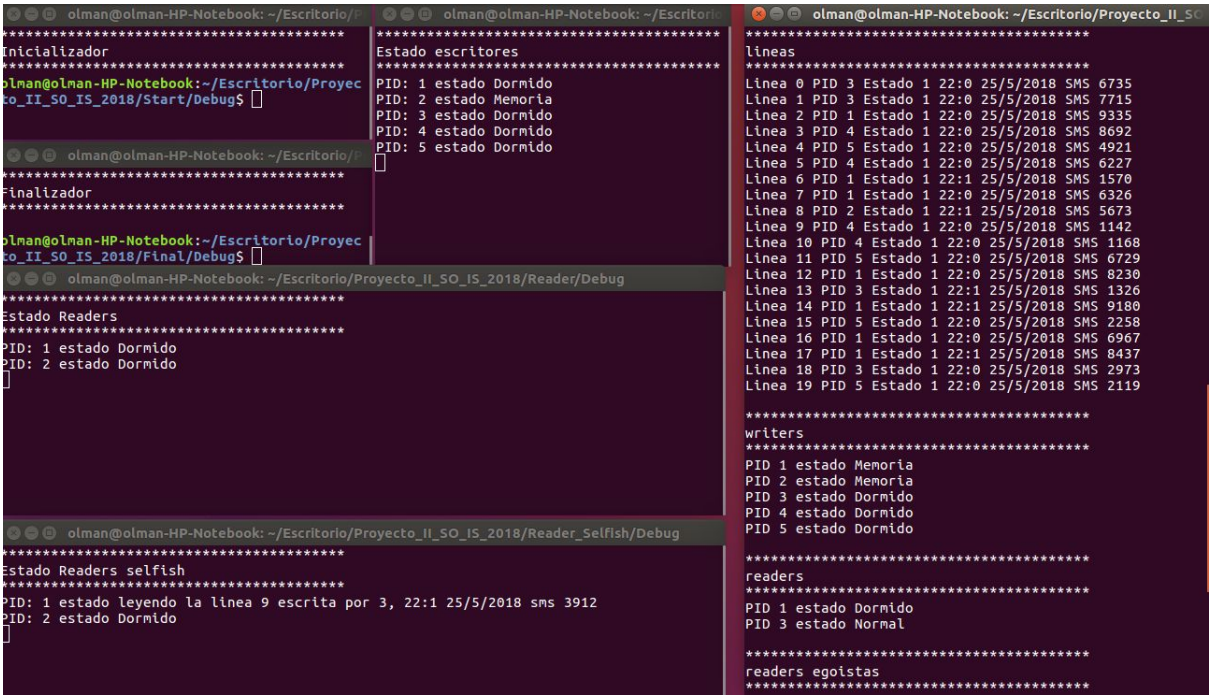
```
****************************************
Bloqueados
****************************************

****************************************
Memoria
****************************************
PID 3 writer
****************************************
Dormido
****************************************
PID 1 writer
PID 2 writer
PID 4 writer
PID 5 writer
PID 1 reader
PID 1 reader selfish
```

Programa Finalizador

```
****************************************
Finalizador
****************************************
```

Todos lo programas corriendo en conjunto



Bibliografía

[1] apfohl. (2018). Shared memory example. [online]

Disponible en:

https://gist.github.com/apfohl/c3603f79b444495e5187

[Acceso 25 mayo. 2018].

[2] reterVision. (2018). semaphor_sample.c. [online]

Disponible en:

https://gist.github.com/apfohl/c3603f79b444495e5187

[Acceso 25 mayo. 2018].

[3] Eclipse.org. (2018). *Eclipse IDE for C/C++ Developers* | *Packages*. [online]

Disponible en:

https://www.eclipse.org/downloads/packages/eclipse-ide-cc-developers/oxygen3

[Acceso 25 mayo. 2018].

[4] www.ibm.com. (2018). Understanding memory mapping. [online]

Disponible en:

https://www.ibm.com/support/knowledgecenter/en/ssw_aix_72/com.ibm.aix.genprogc/understanding_mem_mapping.htm

[Acceso 25 mayo. 2018].

[5] Silberschatz, A., Galvin, P. and Gagne, G. (n.d.). *Operating system concepts*.