Cátedra de Sistemas operativos UTN - FRBA	
Título de documento	EMULASO – Especificación de Trabajo práctico.
Contenido	El presente documento contiene la especificación formal del trabajo práctico cuatrimestral requerido para la aprobación del área práctica de la materia Sistemas operativos (UTN-FRBA).

Trabajo práctico 2do Cuatrimestre 2007

Cátedra de sistemas operativos

EMULASO

Versión 2.0

Índice

Introducción

El trabajo práctico EMULASO consiste en el desarrollo de un sistema distribuido cuya principal finalidad es la de emular el funcionamiento de un sistema operativo con balanceo de carga.
Este sistema operativo permitirá que múltiples usuarios inicien sesión en él, de manera remota y concurrente, y emitan comandos. El sistema ejecutará dichos comandos de manera balanceada sobre aquellos nodos de la red que se encuentren configurados como nodos de red de carga.

Una característica importante del sistema es que los usuarios lo percibirán como un sistema operativo convencional, pero éste, en realidad, esconderá una compleja estructura de nodos de red de carga sobre los cuales balanceará las peticiones del usuario.

A continuación se enumerarán los conceptos teóricos más significativos que aplicará EMULASO:

- Balanceo de carga.
- Migración de procesos.
- Planificación de procesos.
- Seguridad:
  - Aplicación del algoritmo de encripción simétrico.
  - Aplicación de Listas de capacidad (CL).
- Administración de memoria.
- Semáforos.

Objetivos del trabajo práctico

- Que los alumnos afirmen los conceptos teóricos enseñados en la materia mediante la implementación práctica de algunos de ellos.
- Que los alumnos adquieran los conocimientos prácticos del uso de un conjunto de herramientas que ofrecen los sistemas operativos modernos (IPC – Inter Process Communication, creación de procesos, sincronización, entre otros).
- Que los alumnos comprendan la necesidad de un adecuado trabajo en equipo para llevar adelante el desarrollo de un sistema de gran envergadura.
- Que se familiaricen con técnicas de programación de sistemas, como por ejemplo el empleo de Makefiles, archivos de configuración y archivos de log.
- Que los alumnos conozcan con grado de detalle la operatoria de Linux mediante la utilización de un lenguaje de programación de relativo bajo nivel como el C.
- Que comprendan la importancia de una norma o protocolo estándar para llevar a cabo la comunicación entre procesos.

Nomenclatura del documento

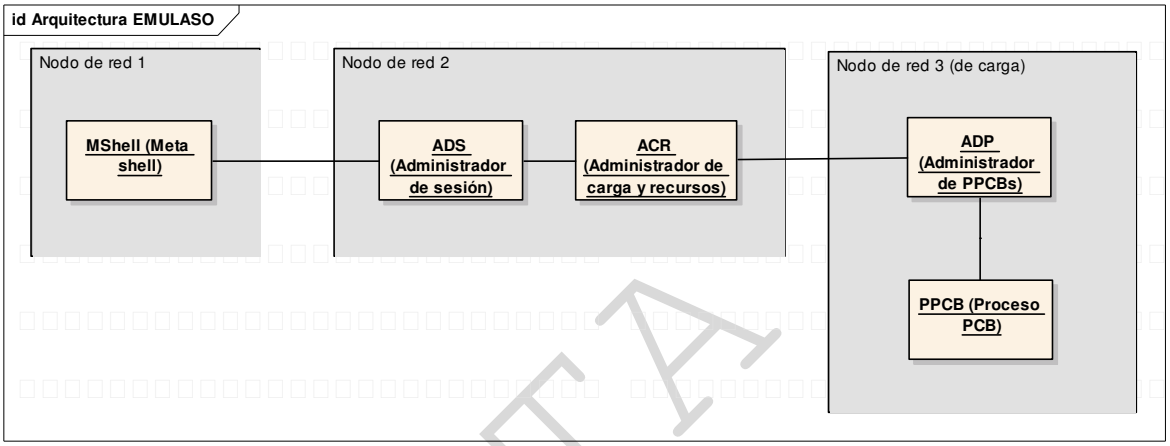
- Los términos que se encuentren en estilo subrayado figurarán en el glosario.
- Los párrafos que se encuentren en estilo *itálica* corresponderán a aclaraciones y/o ejemplos que faciliten la comprensión del texto previo.
- Dentro de la especificación funcional, los términos en estilo **negrita** referenciarán a funciones del mismo ú otro proceso.

Arquitectura del trabajo práctico

EMULASO estará formado por lo siguientes componentes, los cuales deberán ser implementados como **procesos**:

Proceso	Instancias
MShell (Meta Shell)	Pueden existir N por nodo de red
ADS (Administrador de sesión)	1 en todo el sistema
ACR (Administrador de carga y recursos)	1 en todo el sistema
ADP (Administrador de PPCBs)	1 por nodo de red de carga.
PPCB (Proceso PCB)	Pueden existir N por nodo de red de carga

A continuación se presenta un diagrama que intenta mostrar de manera simplificada los componentes y la comunicación permitida[1] entre ellos:



Los objetivos de cada uno de los procesos serán:

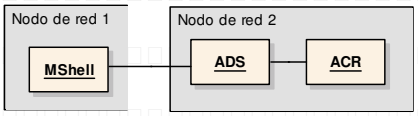
- MShell (Meta Shell):
  Leer por consola comandos del usuario, Iniciar sesión sobre el ADS, Presentar al usuario la información provista por el ADS, Encriptar la información y Desencriptar la información.
- ADS (Administrador de sesión):
  Negociar el logueo de un usuario, Finalizar una sesión, Entregar al Mshell correspondiente la información provista por el ACR, Comunicar al ACR la ejecución de un programa, Encriptar la información, Desencriptar la información y Volcar información sobre el estado del ADS.
- ACR (Administrador de carga y recursos):
  Iniciar la ejecución de un programa, Migrar PPCB a un nodo de red de carga, Determinar nodo de red de carga sobre el cuál migrar un PPCB, Planificar la LPP, Administrar los recursos compartidos y gestionar su otorgamiento y Volcar información sobre el estado del sistema.
- ADP (Administrador de PPCBs):
  Verificar ante su creación la no existencia de otro ADP en el nodo, Gestionar la recepción de un PPCB, Planificar la ejecución de los PPCBs que residan en el nodo de carga, Proporcionar información en tiempo real del nodo de red de carga, Administrar en tiempo real el límite 2 de load average, Retransmitir al ACR la solicitud de recursos compartidos que le realicen sus PPCBs en ejecución, Retransmitir al ACR la información generada por los PPCBs y Volcar información sobre el estado del nodo de red de carga.
- PPCB (Proceso PCB):
  Migrar hacia un nodo de carga, Ejecutar sentencias siempre que se posea el quantum, Conectarse al ACR para indicar el deseo de migrar y Volcar información de estado del PPCB.

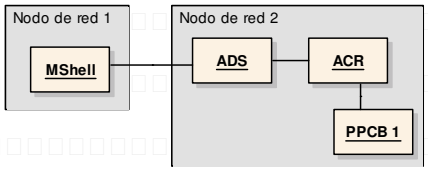
1 No se permitirá la existencia de otros canales de comunicación diferentes a los especificados en el documento, por ejemplo, no será válido crear una conexión entre el Mshell y el PPCB.

Flujo de ejecución ejemplificador

A continuación se presentará de manera simplificada un ejemplo de uso básico del sistema:

Dadas 4 computadoras conectadas formando una red se produce la siguiente situación: Un usuario ejecuta sobre el nodo 2 el programa correspondiente al ACR, luego se crea el proceso ADS. En segundo lugar, un usuario en el nodo 1 con el deseo de ejecutar un programa dentro de EMULASO, ejecuta el programa Mshell y mediante éste inicia una sesión sobre el ADS.

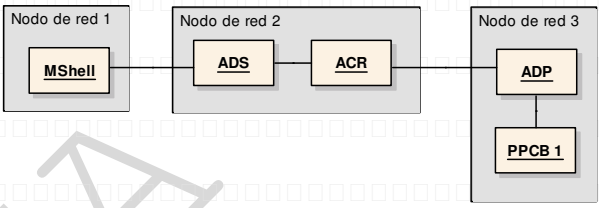




Luego de iniciada, el usuario invoca al comando "exec programa1", lo cual provoca que el Mshell lo retransmita, de manera encriptada, al ADS para que éste finalmente lo envíe al ACR.

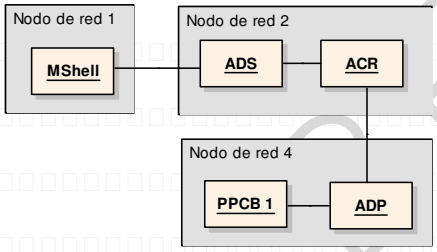
Cuando el ACR recibe ese mensaje crea un nuevo proceso PPCB, al cual le asigna el PPCB ID: 1. Luego de esto el ACR detecta que no existe ningún nodo de red configurado como nodo de red de carga (esto es porque no posee ningún ADP conectado), entonces decide conservar al PPCB en la LPP (Lista de PPCBs pendientes).

Transcurrido un minuto otro usuario decide ejecutar sobre el nodo 3 el programa ADP, de esta manera el nodo se convierte en nodo de red de carga.



Al detectar este evento, el ACR decide revisar toda su LPP en busca de PPCBs pendientes de migración, al hacer esto detecta que existe el PPCB de ID 1, con lo cual intenta migrarlo. La migración es válida, concluye el ADP, por esta razón le indica al PPCB de ID 1 que migre hacia el ADP que se encuentra en el nodo 3.

Al finalizar la migración, el PPCB comienza a ser planificado por el ADP.



Otro usuario ejecuta un ADP sobre el nodo 4, de esta manera el ACR tiene conocimiento de la existencia de dos nodos de red de carga.

Transcurridos unos segundos, un problema en el código del ADP del nodo 3 genera un error fatal, haciendo que finalice inmediatamente. El PPCB de ID 1 detecta este evento, entonces decide comunicarse con el ACR para que lo migre hacia otro nodo de red de carga.

El ACR lo trae (migra) hacia su nodo y luego lleva a cabo el proceso de búsqueda de nodo al cual migrar, esto produce como resultado el nodo de red de carga 4, provocando así que el PPCB de ID 1 migre hacia el nodo 4 y continúe su ejecución hasta finalizar.

Durante todo este tiempo, el usuario que inició sesión sólo supo que ejecutó un programa, pero nunca supo lo que verdaderamente se escondía detrás: EMULASO.

Especificación detallada del trabajo práctico

En la siguiente sección del documento se realizará una definición detallada de cada uno de los procesos que componen EMULASO.

MShell (Meta Shell)

Proceso que actúa como interfaz entre el usuario y el ADS. Mediante el MShell el usuario podrá iniciar sesión en el sistema y emitir comandos que luego serán ejecutados por el ACR.

Especificación funcional

El Mshell tendrá bajo su responsabilidad las siguientes funciones:

Leer por consola comandos del usuario

El Mshell, de manera análoga al shell que brinda unix, permitirá al usuario ingresar un conjunto de comandos que luego serán replicados hacia el ADS a través de la conexión establecida entre ellos. Los comandos que el Mshell deberá reconocer son los siguientes[2]:

Comando	Descripción general
login <nombreUsuario> *Ej: login pepe*	Utilizado para **iniciar sesión sobre el ADS** con el usuario <nombreDeUsuario>.
exec <nombrePrograma> *Ej: exec programa1*	Invocará la ejecución del programa denominado <nombrePrograma>. El usuario podrá ejecutar tantos comandos *exec* como desee sin la necesidad de esperar la finalización del anterior. Esto permite que existan múltiples PPCBs simultáneos asociados a una misma sesión.
logout	Utilizado para finalizar la sesión actual. El cierre de sesión no implica la finalización del Mshell, esto hace posible que desde un mismo Mshell puedan efectuarse, de manera secuencial[3], diferentes sesiones.
exit	Finaliza la ejecución del proceso Mshell.

Iniciar sesión sobre el ADS

Cuando el usuario ejecute sobre la consola del Mshell el comando *login* seguido del nombre de usuario con el cual desea loguearse, el Mshell deberá leer del archivo de configuración la información sobre el ADS y conectarse a él. Acto seguido deberá **negociar el logueo del usuario** con éste último, para ello enviará en primer lugar el nombre del usuario y en caso de ser aprobado por el ADS, se le solicitará al usuario que ingrese el password, transmitiéndolo luego también. Finalizada esta negociación, el Mshell deberá informar por pantalla el resultado de la misma, esto es, sesión iniciada exitosamente ó en caso contrario, la causa que produjo la imposibilidad del inicio de la misma.

Aclaraciones:

- En caso de existir una sesión previa, el Mshell deberá (internamente) ejecutar un logout sobre el ADS y luego **iniciar sesión sobre el ADS** con el *login* indicado por el usuario.
- La conexión establecida con el ADS perdurará el tiempo que viva la SESION.

Presentar al usuario la información provista por el ADS

Toda información proveniente del ADS que haya sido generada como resultado de la ejecución de los PPCBs creados en la sesión deberá ser desencriptada e impresa en pantalla. En caso que múltiples PPCBs en forma simultánea emitan información, se deberá imprimir previamente un mensaje que identifique al PPCB de manera tal que se pueda apreciar fácilmente que información genera cada uno.

Encriptar la información

[2] Ante el ingreso de un comando no permitido, el Mshell deberá informar dicha situación por pantalla.

3 El concepto de *secuencial* hace referencia a que no podrán existir dos sesiones simultáneas en un mismo Mshell.

Todo flujo de datos enviado desde el Mshell al ADS, a excepción del correspondiente al comando *login <nombreUsuario>*[4], deberá ser encriptado en base a la especificación del **Anexo - Encripción/Desencripción de la información del canal Mshell-ADS**.

Desencriptar la información

Todo flujo de datos proveniente del ADS, a excepción del correspondiente a la respuesta al comando login <nombreUsuario>, deberá ser desencriptado en base a la especificación del **Anexo - Encripción/Desencripción de la información del canal Mshell-ADS**.

Parámetros configurables

Parámetro	Por archivo de config.	En tiempo de ejecución
IP y puerto donde el ADS escucha conexiones Mshell.	X	
Path absoluto del directorio donde residen las claves de (des)encripción de los usuarios.	X	

ADS (Administrador de sesión)

Proceso que actúa como interfaz entre los MShell conectados a él y el ACR. El ADS gestionará las sesiones de los usuarios establecidas mediante los procesos MShell.

Especificación funcional

El ADS tendrá bajo su responsabilidad las siguientes funciones:

Negociar el logueo de un usuario

Cuando el ADS reciba del Mshell la información sobre el intento de logueo de un usuario, éste deberá en primer lugar verificar la existencia del mismo, para ello consultará el archivo de usuarios[5]. Luego el ADS informará al Mshell el resultado de dicha búsqueda, el cual podrá ser "usuario válido" en caso de encontrarlo ó "usuario inexistente" en caso que no exista (manteniendo la conexión abierta para un nuevo ingreso de usuario).
En caso que el usuario exista, el ADS deberá recibir del Mshell el password del usuario y, nuevamente, validará el mismo consultando el archivo de usuarios e informará al Mshell el resultado de dicha validación. Esta puede generar dos posibles resultados:

- Password correcto, el ADS iniciará la sesión y estará disponible para recibir comandos del usuario.
- Password incorrecto, el ADS cerrará la conexión establecida con el Mshell que intento iniciar sesión.

Aclaraciones:

- Un mismo usuario podrá generar tantas sesiones como desee (siempre que las realice mediante diferentes procesos Mshell). Esto significa por ejemplo que el usuario "pepe" podrá tener activas 3 sesiones en el ADS, por lo cual deberá haber ejecutado 3 procesos Mshell.

Finalizar una sesión

Cuando el ADS reciba del Mshell el evento para finalizar la sesión (sea tanto por el comando *logout*, *exit* ó por una finalización inesperada del proceso Mshell), el ADS deberá cerrar la conexión en caso de encontrarse abierta. Acto seguido deberá informarle dicha situación al ACR para que éste finalmente libere los recursos y la carga asociada a esa sesión (PPCBs).

Entregar al Mshell correspondiente la información provista por el ACR

[4] La clave de encripción/desencripción se encuentra asociada al usuario, por lo tanto no hay manera de desencriptar el flujo de datos en el ADS si aún éste no conoce al usuario en cuestión.
5 Para conocer el formato del archivo de usuarios dirigirse a la sección **Anexo - Formato de archivos**.

El ADS deberá enviar al Mshell correspondiente la información enviada por el ACR.

Comunicar al ACR la ejecución de un programa

Cuando el ADS reciba del Mshell el comando *exec <nombrePrograma>* deberá retransmitirlo[6] al ACR para que éste pueda **iniciar la ejecución del programa**. Luego el ADS retransmitirá la respuesta del ACR hacia el Mshell que invocó la ejecución.

Encriptar la información

Todo flujo de datos enviado desde el ADS al Mshell, a excepción del correspondiente a la respuesta al comando login <nombreUsuario>, deberá ser encriptado en base a la especificación del anexo **Anexo - Encripción/Desencripción de la información del canal Mshell-ADS**.

Desencriptar la información

Todo flujo de datos proveniente del Mshell, a excepción del correspondiente al comando *login <nombreUsuario>*, deberá ser desencriptado en base a la especificación del anexo **Anexo - Encripción/Desencripción de la información del canal Mshell-ADS**.

Volcar información sobre el estado del ADS

El ADS al recibir la señal SIGUSR1 (10) deberá escribir[7] en un archivo específico de estado del ADS la siguiente información actual del sistema:

- Por cada sesión generada:
  - Su identificador único de sesión (*Podrá ser, por ejemplo, el descriptor del socket*).
  - El usuario asociado a la sesión (aquél que la generó).
  - El Mshell asociado a la sesión (IP).

Parámetros configurables

Parámetro	Por archivo de config.	En tiempo de ejecución
IP y puerto donde el ACR escucha la conexión del ADS.	X	
Path absoluto donde residen las claves de (des)encripción de los usuarios.	X	
Path absoluto del archivo de usuarios.	X	

ACR (Administrador de carga y recursos)

Proceso que se encarga de gestionar los PPCBs en lo que a su creación, balanceo y otorgamiento de recursos se refiere.
El ACR tendrá conocimiento de la totalidad de los PPCBs que se encuentren activos en el sistema y el nodo de red donde resida cada uno de ellos.

Características generales

Listas de procesos

El ACR hará uso de una lista de procesos, a saber:

- **LPP (Lista de PPCBs pendientes) – *Planificación: Algoritmo FIFO***

  En esta lista se coloca a aquellos PPCBs que aún no migraron a un nodo de red de carga, por lo tanto residen físicamente en el nodo de red del ACR.
  Los PPCBs que se encuentren en la misma tendrán un tiempo máximo de vida definido por configuración, esto es, si el tiempo que transcurrió desde que un PPCB ingresó a la lista es mayor al tiempo máximo de vida, el PPCB deberá ser eliminado del sistema e informar al ADS dicha situación.

[6] Recordar que sólo el canal Mshell-ADS estará encriptado, los restantes canales de comunicación transmitirán la información en formato plano (legible sin necesitar desencripción previa).
[7] En caso de ya existir el archivo, se deberá agregar la información al final.

*Cada vez que un ADP se conecte al ACR (es decir, se adicione un nodo de red de carga), se deberá llevar a cabo la función **migrar PPCB a un nodo de red de carga** con cada uno de los PPCBs que se encuentren en la LPP.*
Para mayor detalle en las transiciones de estado/lista dirigirse a **Anexo – Estados de un PPCB / Planificación de PPCBs**.

Recursos compartidos[8] del sistema

El ACR contará con tres recursos que podrán ser solicitados por los PPCBs que se encuentren en el sistema.
Los recursos serán ***Impresora***, ***Disco*** y ***Cinta***, cada uno de ellos tendrá una cantidad de instancias que serán consumidas al ser cedidas a PPCBs y restauradas al ser devueltas por aquellos que las tomaron.

Ejemplo:
Disco=3 Indica la existencia de 3 discos compartidos. Estos residen en el nodo del ACR.

Especificación funcional

El ACR tendrá bajo su responsabilidad las siguientes funciones:

Iniciar la ejecución de un programa

Cuando el ACR reciba del ADS el comando *exec <nombrePrograma>*, éste en primer lugar verificará la existencia del programa en cuestión, para ello buscará en el directorio definido por configuración el archivo de nombre "nombrePrograma.emu", en caso de no existir se informará dicha situación al ADS.

Si el archivo es hallado, el ACR creará un proceso PPCB y le transferirá toda la información relativa al programa a ejecutar, esto es su código ejecutable obtenido desde el archivo, un identificador único en todo el sistema de red (PPCB ID), el usuario y comando que lo creó.
Finalizada la creación del PPCB, el ACR deberá migrarlo a un nodo de red de carga, para ello llevará a cabo la función **migrar PPCB a un nodo de red de carga**.

Migrar PPCB a un nodo de red de carga

Para realizar la migración de un PPCB a un nodo de red carga, el ACR en primer lugar deberá asegurarse que el PPCB se encuentre en su nodo de red[9], en caso de no estarlo deberá traerlo (migrar hacia el nodo del ACR). En segundo lugar lo encolará en la LPP.
Acto seguido, deberá **determinar el nodo de red de carga sobre el cuál migrar un PPCB**. La ejecución de esta función puede generar dos posibles resultados, a saber:

- No se encuentre un nodo de carga, bajo este escenario el ACR mantendrá al PPCB en la LPP.
  *Aclaración importante: durante todo el tiempo que el PPCB permanezca en la LPP, el ACR deberá contar con alguna forma de comunicación con el mismo, sea tanto mediante un socket, memoria compartida u otra. Luego que el PPCB sea quitado de la LPP, esta comunicación deberá ser **finalizada**.*
- Se encuentre un nodo de carga, entonces el ACR le enviará al PPCB la información del nodo de red de carga al cual debe migrar, haciendo que el PPCB lleve a cabo la función **migrar hacia un nodo de carga**.

Determinar nodo de red de carga sobre el cuál migrar un PPCB

Para realizar esto el ACR deberá recolectar información sobre cada nodo de red de carga disponible, para ello obtendrá de cada uno de los procesos ADP que se encuentren conectados a él la siguiente información:

- La cantidad de PPCBs que puede albergar.
- La memoria máxima que dispone el nodo de red de carga.
- La carga promedio real[10] del nodo de red de carga.

[8] Se denominan compartidos ya que podrán ser solicitados por un PPCB desde cualquier nodo de red de carga. Dichos recursos son ficticios, es decir, no representan recursos reales del sistema operativo.
[9] Si el PPCB fue recién creado entonces sí estará en el nodo del ACR, pero esta función, como se verá más adelante en la especificación del PPCB, es reutilizada, y en esa situación el PPCB se encontrará en otro nodo con lo cual el ACR deberá traerlo previamente a migrarlo a un nodo de carga.
[10] Para conocer en detalle cómo obtener dicha información dirigirse a la especificación funcional del proceso ADP.

En base a la información recogida, el ACR desechará a aquellos nodos de carga que no posean espacio disponible para albergar al PPCB ó la memoria máxima sea inferior a la requerida por el PPCB.
Acto seguido, el ACR seleccionará entre los nodos de carga válidos a aquél que posea la menor carga promedio real. En caso de existir dos nodos con igual valor, seleccionará a aquél que pueda albergar mayor cantidad de PPCBs.

Planificar la LPP

El ACR se encargará de administrar la LPP según lo especificado en el **Anexo – Estados de un PPCB / Planificación de PPCBs**.

Administrar los recursos compartidos y gestionar su otorgamiento

Cada vez que el ACR reciba desde un ADP la solicitud de una instancia de recurso compartido, el ACR deberá chequear los permisos sobre los recursos que posee el usuario que ejecutó el PPCB[11], esto puede generar dos escenarios:

- El usuario no tiene permiso, el ACR indicará al ADP que finalice la ejecución del PPCB[12] que solicitó el recurso, luego se informará dicha situación al ADS.
- Si el usuario tiene permiso, el ACR lo otorgará si la cantidad disponible de instancias es mayor a cero. Una vez aceptada o rechazada la solicitud se informará al ADP la decisión y luego se procederá a descontar en uno la cantidad de instancias disponibles del recurso. De esta forma, y al igual que en un semáforo, dependiendo del signo ese valor tendrá los siguientes significados:
  - Mayor a cero: Indica la cantidad de instancias disponibles del recurso.
  - Igual a cero: No existen instancias del recurso y, adicionalmente, ningún PPCB se encuentra a la espera del mismo.
  - Menor a cero: Indica que no existen instancias disponibles del recurso y el valor absoluto de ese número representa la cantidad de PPCBs a la espera de instancias del recurso.

Dicho valor será incrementado en uno por cada mensaje de ADP indicando que un PPCB liberó una instancia de ese recurso.

Aclaraciones:

- Cuando el ACR reciba el evento indicando la finalización (normal o anormal) de un PPCB, éste deberá liberar todas las instancias de recursos tomadas por el PPCB que aún no fueron explícitamente liberadas.
- Cada vez que se libere una instancia de un recurso, el ACR deberá analizar si existe algún PPCB bloqueado a la espera de éste, en caso de existir deberá otorgarle la instancia y luego informarle al ADP correspondiente sobre el desbloqueo de un PPCB bajo su control.
  La búsqueda de PPCB bloqueados por recurso deberá respetar el orden de ingreso, es decir se aplicará el algoritmo de FIFO.

Volcar información sobre el estado del sistema

El ACR al recibir la señal SIGUSR1 (10) deberá escribir[13] en un archivo específico de estado del ACR la siguiente información actual del sistema:

- Por cada PPCB en el sistema[14]:
  - Su identificador único.
  - El usuario que lo ejecutó.
  - El Nodo de red en el cual reside (IP).
  - El estado del PPCB (Ver **Anexo – Estados de un PPCB / Planificación de PPCBs**).
  - El comando completo que provocó su ejecución.
  - El identificador de sesión que generó la ejecución del PPCB.
- Por cada recurso del sistema:

[11] Esta información la obtendrá del archivo de usuarios.
[12] La terminación del proceso se realiza en base al criterio "Protection error", para más información leer tabla 3.2 de Sistemas operativos (Stallings, 5ta edición).
[13] En caso de ya existir el archivo, se deberá agregar la información al final.
[14] Cierta información estática del proceso (como por ejemplo el usuario que lo ejecutó y otras) podrá ser almacenada en el ACR de manera tal de no solicitarla al ADP y evitar la transferencia de mensajes variables.

- La cantidad de instancias disponibles.
- La cantidad de instancias totales.
- Los PPCB que se encuentran a la espera de una instancia.
- Los PPCB que poseen una instancia.

- De la LPP:
  - Los PPCB encolados (respetando el orden).

Parámetros configurables

Parámetro	Por archivo de config.	En tiempo de ejecución
Tiempo máximo de vida de un PPCB en la lista LPP [segundos].	X	X
Cantidad de instancias de los recursos Impresora, Disco y Cinta. *Por ejemplo: Impresora=1, Disco=3*.	X	
Path absoluto del directorio donde residen los programas (*.emu) *Sólo el ACR tendrá acceso a estos archivos*.	X	
Path absoluto del archivo de usuarios.	X	

ADP (Administrador de PPCBs)

Proceso que al ejecutarse en un nodo de red le confiere al mismo la capacidad de recibir carga por parte del ACR, convirtiéndolo de esta manera en un nodo de red de carga.
El ADP se encargará de proporcionar información en tiempo real del nodo de red de carga y tendrá la responsabilidad principal de albergar los procesos PPCBs que migren hacia él y gestionar su ejecución.

Características generales

Memoria[15] de nodo de red de carga

El ADP, mediante el archivo de configuración, conocerá la cantidad de memoria de la cual dispone el nodo de carga para ejecutar procesos.
Esta memoria será la que defina el grado de multiprogramación del ADP, esto es, la cantidad de PPCBs que se pueden ejecutar simultáneamente en un determinado instante.

Aclaraciones:

- *Sólo consumirán memoria aquellos PPCBs que se encuentren en la LPE.*
- *La memoria no sufrirá un proceso de fragmentación. A efectos prácticos, deberá simplemente incrementarse o decrementarse sin generar "huecos" ante la salida de PPCBs de la LPE.*

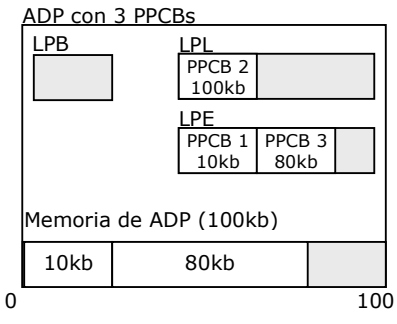
Listas de procesos

El ADP hará uso de tres listas de procesos principales, a saber:

- **LPL (Lista de PPCBs listos)**

  En esta lista estarán aquellos PPCBs que se encuentren listos para ejecutar pero que aún el ADP no les haya concedido su quantum.
  El algoritmo de planificación consistirá en recorrer hasta el FINAL la lista en el orden en que los PPCBs ingresaron otorgando un quantum a aquellos cuya memoria requerida sea inferior o igual a la memoria actual disponible por el nodo.

  Cada vez que un PPCB ingrese a la LPL o salga de la LPE se deberá llevar a cabo este procedimiento. Esto significa que el ADP deberá tratar en todo momento de ocupar al máximo posible la memoria disponible.



- **LPE (Lista de PPCBs en ejecución)**

[15] La memoria no se corresponde con la memoria real con la que cuenta la PC, sino que es ficticia y solamente utilizada para limitar la cantidad de PPCBs que se pueden ejecutar concurrentemente.

En esta lista se colocará a los PPCBs que se encuentren actualmente en ejecución, es decir a aquellos a los cuales se les otorgó el quantum[16].

- **LPB (Lista de PPCBs bloqueados)**

En esta lista se colocará a aquellos procesos que solicitaron un recurso pero éste no se encontraba disponible, por lo tanto están a la espera del mismo y no pueden continuar su ejecución.

Especificación funcional

El ADP tendrá bajo su responsabilidad las siguientes funciones:

Verificar ante su creación la no existencia de otro ADP en el nodo

Cuando se inicie[17] un proceso ADP, previo a conectarse al ACR, deberá verificar que no exista otro proceso ADP ejecutándose en ese nodo de red, en caso de existir deberá finalizar su ejecución informando dicha situación por pantalla.

Gestionar la recepción de un PPCB

Cuando un PPCB que reside en el ACR se conecte al ADP y le indique su deseo de migrar hacia él, el ADP creará el proceso PPCB con la información que le transfiera el proceso que migra y luego lo ubicará en la lista correspondiente en función de su estado (LPL ó LPB) para, finalmente, comenzar a planificar su ejecución.

Planificar la ejecución de los PPCBs que residan en el nodo de carga

El ADP se encargará de planificar la ejecución de aquellos PPCBs que se encuentren dentro de su nodo de red, para conocer cómo llevará a cabo esta función dirigirse al **Anexo – Estados de un PPCB / Planificación de PPCBs**.

Proporcionar información en tiempo real del nodo de red de carga

Cuando el ACR solicite información actual del nodo de carga al ADP, éste deberá proporcionarle la siguiente:

- La memoria máxima que dispone el nodo de red de carga, este es el valor que se obtiene del archivo de configuración.
- La carga promedio real del nodo de red de carga, este es el valor que se visualiza al ejecutar el comando uptime[18] en unix.
- La cantidad de PPCBs que puede albergar, dicho valor estará en función de la carga promedio real del nodo:
  - Si el load average es superior al **Límite 1** (definido por configuración) entonces se retornará 0(cero), indicando así que no se podrán recibir PPCBs.
  - Si el load average es inferior al **Límite 1**, se retorna la resta entre la cantidad de PPCBs máxima que puede albergar el ADP y la cantidad actual que alberga.

Administrar en tiempo real el límite 2 de load average

Periódicamente (5 seg.), el ADP deberá analizar el valor de la carga promedio real de su nodo, esto con el objetivo de reaccionar oportunamente ante valores que circunden el Límite 2.
En el instante en el que el load average crece de manera tal que supera al **Límite 2**, el ADP deberá seleccionar (sólo si existe una cantidad de PPCBs mayor a 1) al PPCB que le reste la **mayor** cantidad de tiempo estimada para finalizar. Luego al PPCB seleccionado se le indicará que migre, haciendo que ejecute la función **Conectarse al ACR para indicar el deseo de migrar**.

[16] El quantum no se aplicará sobre todo el conjunto de PPCBs que se encuentre en la LPE, sino que en particular sobre cada PPCB, esto posibilita que por ejemplo exista uno con quantum restante 3s y otro con 2s.
[17] Dicha verificación podrá ser realizada mediante un script en bash (dicho script se encargaría de invocar o no al binario del ADP) ó dentro del mismo proceso.
[18] El valor a retornar será el primer campo de "load average", para obtener información acerca de cómo conseguir ese valor leer *man proc.*

Retransmitir al ACR la solicitud de recursos compartidos que le realicen sus PPCBs en ejecución

Cada vez que el ADP reciba de un PPCB la solicitud de un recurso compartido, éste deberá retransmitirla al ACR. Si la respuesta al pedido es negativa, el ADP deberá quitar de la LPE al PPCB y colocarlo en la LPB, en caso de ser positiva, continuará ejecutando.

Retransmitir al ACR la información generada por los PPCBs

Toda información de salida que genere un PPCB producto de la ejecución de una sentencia deberá ser recibida por el ADP y retransmitida al ACR.

Volcar información sobre el estado del nodo de red de carga

El ADP al recibir la señal SIGUSR1 (10) deberá escribir[19] en un archivo específico de estado del nodo de carga la siguiente información actual del sistema:

- Por cada PPCB en el nodo de carga:
  - Su identificador único.
  - El usuario que lo ejecutó.
  - El estado del PPCB (Ver **Anexo – Estados de un PPCB / Planificación de PPCBs**).
  - El comando completo que provocó su ejecución.
  - Si se encuentra en ejecución, el quantum RESTANTE para ser quitado de la LPE.
- Por cada lista:
  - Los PPCBs encolados (respetando el orden).

Parámetros configurables

Parámetro	Por archivo de config.	En tiempo de ejecución
IP y puerto donde el ACR escucha las conexiones de ADPs.	X	
Memoria máxima de la cual dispone el nodo de carga.	X	
Cantidad máxima de PPCBs que puede albergar el nodo de carga.	X	
Quantum que se le otorgará a cada PPCB al ingresar a la LPE [segundos].	X	X[20]
Límite 1 de carga promedio real (loadavg) [decimal].	X	
Límite 2 de carga promedio real (loadavg) [decimal]. *Este valor será siempre mayor ó igual al Límite 1.*	X	

PPCB (Proceso PCB)

Este proceso representa una instancia de un programa creado por el ADP. Su ejecución será planificada por el ADP.
El PPCB se encargará de ejecutar sentencias siempre que posea el quantum.

Características generales

Elementos de un PPCB

El PPCB, asemejándose a un PCB de Unix, estará compuesto por los siguientes elementos:

- Identificador único de PPCB (PPCB ID).
- Nombre de usuario que lo creó.
- Código ejecutable (aquél conjunto de sentencias que le proporcionó el ACR al crearlo, obtenido del "<nombrePrograma>.emu").
- IP (Instruction pointer): Número de sentencia próxima a ejecutar.
- Stack o pila de datos: Estructura de datos del tipo LIFO donde se almacenarán valores.
- Estado de PCBB: Estado actual en el que se encuentra el PPCB.

[19] En caso de ya existir el archivo, se deberá agregar la información al final.
[20] Los cambios realizados sobre el quantum en tiempo de ejecución tendrán efecto la próxima vez que los PPCBs ingresen a la LPE, es decir, no afectará inmediatamente a aquellos que se encuentran en ejecución.

Sentencias de un programa PPCB

A continuación se detallarán las sentencias que podrán formar parte de un programa:

Sentencia (sintaxis)	Descripción general	Tiempo insumido
MEM <cantidad>	Esta será la primer sentencia de todo programa de EMULASO, indicará la cantidad de memoria requerida por el PPCB para ejecutarse. El ACR solicitará, al inicio del PPCB, esta información para decidir el nodo sobre el cual migrar.	-
OPER <cantSegundos>	Esta sentencia al ser ejecutada simulará[21] una operación compleja que requiere una cantidad de segundos para finalizar. **Aclaración importante***: Si el quantum del PPCB finaliza mientras se está ejecutando la sentencia OPER, al volver a obtener el quantum el PPCB deberá continuar hasta terminar el tiempo restante.* *Sobre las demás sentencias no aplica dicho comportamiento.*	cantSegundos
SOL <nombreRecurso>	Solicita una instancia de recurso compartido. La ejecución de esta sentencia implica el envío de un mensaje por parte del PPCB hacia el ADP solicitando el recurso.	1 segundo
DEV <nombreRecurso>	Devuelve una instancia de recurso compartido que posee el PPCB. La ejecución de esta sentencia implica el envío de un mensaje por parte del PPCB hacia el ADP indicando la devolución del recurso.	1 segundo
IMP <texto>	Al ejecutar esta sentencia el PPCB deberá enviar al ADP un mensaje de contenido <texto> para que sea impreso en el Mshell que ejecutó el programa.	1 segundo
PUSH <letraONumero>	Esta sentencia apilará en el stack el valor <letraONumero>, este valor corresponderá a un único byte.	1 segundo
POP	Esta sentencia obtiene (y quita) del stack el último valor agregado y lo envía al ADP para que finalmente sea impreso en el Msheel.	1 segundo

Un ejemplo de programa podría ser:

SENTENCIAS	**DETALLE**
MEM 10	El PPCB requiere, para ejecutar, 10 unidades de memoria.
SOL Impresora	Solicita a su ADP el recurso compartido Impresora.
OPER 15	Ejecuta una operación que insumirá **como mínimo** 15 segundos.
DEV Impresora	Devuelve el recurso compartido Impresora.
IMP Ya imprimí!!	Imprimirá en el Mshell el texto "Ya imprimí!!".

Al finalizar la ejecución de sentencias, el PPCB deberá ejecutar internamente un *IMP "Proceso <PPCBID> finalizado exitosamente."* y luego terminar su ejecución.

Especificación funcional

El PPCB tendrá bajo su responsabilidad las siguientes funciones:

Migrar hacia un nodo de carga

Cuando el ACR provea al PPCB la información del nodo de carga (ADP) sobre el cual migrar, éste deberá en primer lugar establecer una conexión temporal de migración hacia el ADP residente en el nodo de carga indicado y luego transmitirle su información para que el ADP lleve a cabo la función **Gestionar la recepción de un PPCB**.

Ejecutar sentencias siempre que se posea el quantum

Cuando el PPCB reciba del ADP el mensaje indicando que puede ejecutar, entonces éste deberá comenzar (o continuar en el caso de un *OPER* interrumpido) la ejecución de sentencias.

Cuando el PPCB reciba del ADP el mensaje indicando la finalización del quantum, deberá interrumpir su ejecución.

Conectarse al ACR para indicar el deseo de migrar

Cuando el PPCB detecte que el ADP que gestiona su ejecución finalizó inesperadamente ó cuando reciba la señal SIGUSR2 (12)[22], el PPCB deberá conectarse al

[21] "Simulará" hace referencia a que la ejecución de la sentencia no implica un procesamiento complejo real, sino que será utilizada para introducir demoras en la ejecución de un PPCB.
[22] Y no se encuentre en el nodo de red del ACR.

ACR e indicarle su deseo de migrar, para que luego el ACR ejecute la función **migrar PPCB a un nodo de red de carga**.

Proveer información acerca del tiempo estimado para finalizar

El PPCB deberá proveer al ADP, cuando éste lo requiera, del tiempo estimado que le resta para finalizar su ejecución. Este valor resultará de sumar el **tiempo insumido** de cada una de las sentencias que resten ejecutar. Esto significa que sólo se considerarán las sentencias de número mayor igual al Instruction pointer.

Volcar información de estado del PPCB

El PPCB al recibir la señal SIGUSR1 (10) deberá escribir[23] en un archivo específico de estado del PPCB la siguiente información actual del sistema:

- Usuario que ejecutó el PPCB.
- El comando completo que provocó su ejecución.
- Sentencias del programa del PPCB.
- Instruction pointer (ó en su defecto una marca sobre la próxima sentencia a ejecutar).
- Valores almacenados en el stack.

[23] En caso de ya existir el archivo, se deberá agregar la información al final.

Fases del trabajo práctico

Nota preliminar

Antes de comenzar con el desarrollo de las fases, se recomienda al alumno dedicar el tiempo necesario para leer y comprender la totalidad del trabajo práctico. El desarrollo del mismo es mayoritariamente modular, por lo tanto la buena comprensión los ayudará a optimizar el desarrollo y los tiempos, como así también hacerlo más fácil y llevadero.

Es obligatorio para todo alumno que realice el trabajo práctico leer y aceptar las NTP (Normativas del trabajo práctico) en su versión 1.6. Este documento se encontrará disponible en el grupo de la materia.

Es requisito que en todas las fases se graben en un archivo Log los eventos significativos (hitos, errores, información, etc.) acontecidos durante la ejecución del proceso. Esta información será de utilidad para el alumno y ayudante para comprender el comportamiento del proceso.
El log deberá respetar el formato establecido en el **Anexo – Formato de archivos de log.**

Definición de fases

EMULASO estará compuesto de las siguientes fases:

Fase 1 - Mshell y ADS completos	
Alcance	Se deberá desarrollar la totalidad de las funcionalidades de los procesos Mshell y ADS, salvo aquellas que requieran la existencia del ACR.
Tiempo estimado	2 semanas.
Fecha de finalización	15/09/2007

Fase 2 – ACR y ADP, balanceo de carga.	
Alcance	Se deberán desarrollar las funcionalidades del ACR y ADP necesarias para **determinar el nodo de red de carga sobre el cual migrar**. Para esta fase no será necesaria la creación de procesos PPCB ni, por consiguiente, la migración.
Tiempo estimado	1 semana.
Fecha de finalización	22/09/2007

Fase 3 – PPCB completo (ENTREGA OBLIGATORIA)	
Alcance	Se deberá desarrollar el PPCB completo. Adicionalmente, permitir al usuario que ejecute un programa desde el Mshell y que el PPCB creado en el ACR migre al nodo de red de carga. No es necesaria la planificación del PPCB dentro del ADP, sólo se requiere su migración.
Tiempo estimado	2 semanas.
Fecha de finalización	06/10/2007

Fase 4 – ADP completo	
Alcance	Se deberá desarrollar al ADP en su totalidad. Esto posibilitará la planificación de los PPCBs bajo su mando (en su nodo). El pedido de recursos no debe ser desarrollado.
Tiempo estimado	3 semanas.
Fecha de finalización	27/10/2007

Fase 5 – ACR, gestión y otorgamiento de recursos (ENTREGA OBLIGATORIA)	
Alcance	Se deberá desarrollar la totalidad de las funciones del ACR que permitan llevar a cabo la administración y otorgamiento de recursos compartidos. En esta fase un usuario podrá ejecutar múltiples programas y visualizar los resultados que ellos generen.
Tiempo estimado	2 semanas.
Fecha de finalización	10/11/2007

Fase 6 – Integración final (ENTREGA OBLIGATORIA Y PRESENCIAL)	
Alcance	Desarrollar por completo todos los procesos e integrarlos. El sistema debe quedar totalmente operativo.
Tiempo estimado	2 semanas.
Fecha de finalización	24/11/2007

Anexos

Encripción/Desencripción de la información del canal Mshell-ADS

Un hecho conocido por personas del ambiente informático es la gran cantidad de riesgos de seguridad que presentan los sistemas de red como es el caso de la internet, por esta razón se observa cada día más la necesidad de implementar mecanismos que permitan mitigar estos riesgos. Con el paso del tiempo han surgido muchos de ellos, en EMULASO se decidió implementar uno de los más conocidos y utilizados mecanismos de seguridad en la información, la encripción simétrica de datos.

Implementación en EMULASO

La encripción simétrica se caracteriza por los siguientes 5 componentes:

1. **Texto plano:** Estará representado por el flujo de bytes que se transmita desde un proceso a otro (Mshell -> ADS o ADS -> Mshell).
2. **Algoritmo de encripción:** Consistirá básicamente en la aplicación de una operación de XOR entre cada byte a enviar y la clave de (des)encripción.
   A continuación se ejemplifica el pseudo código de dicho algoritmo:

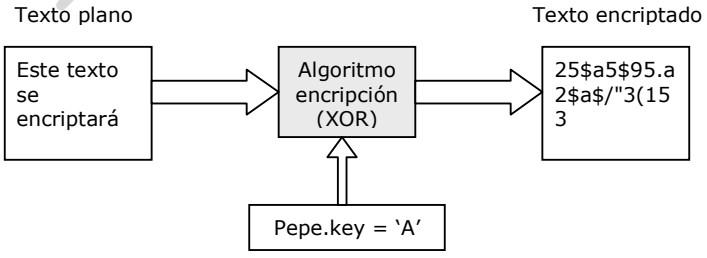
   ```
   char clave = leerClave("pepe.key")
   Por cada byte del bufferAEnviar {
           bufferAEnviar[índice] = bufferAEnviar[índice] XOR clave
   }
   send(bufferAEnviar)
   ```

3. **Clave de (des)encripción:** Estará almacenada en un archivo cuyo nombre tendrá la nomenclatura "<nombreUsuario>.key" por ejemplo "pepe.key". Dentro de este archivo figurará un único byte[24] que será el utilizado por el algoritmo de encripción/desencripción para realizar la operación de XOR.

   Como lo indica el algoritmo, esta clave deberá encontrarse físicamente en todos aquellos nodos de red que requieran encriptar o desencriptar la información.

   Como se puede apreciar, existirá una clave por usuario del sistema, la misma será generada automáticamente por el script de creación de usuarios[25] y deberá ser copiada manualmente[26] donde se la requiera.
4. **Texto encriptado:** Estará representado por el flujo de bytes resultante de aplicar sobre el texto plano el algoritmo de encripción en conjunto con la clave del usuario.

   Ejemplo de texto plano y texto encriptado:

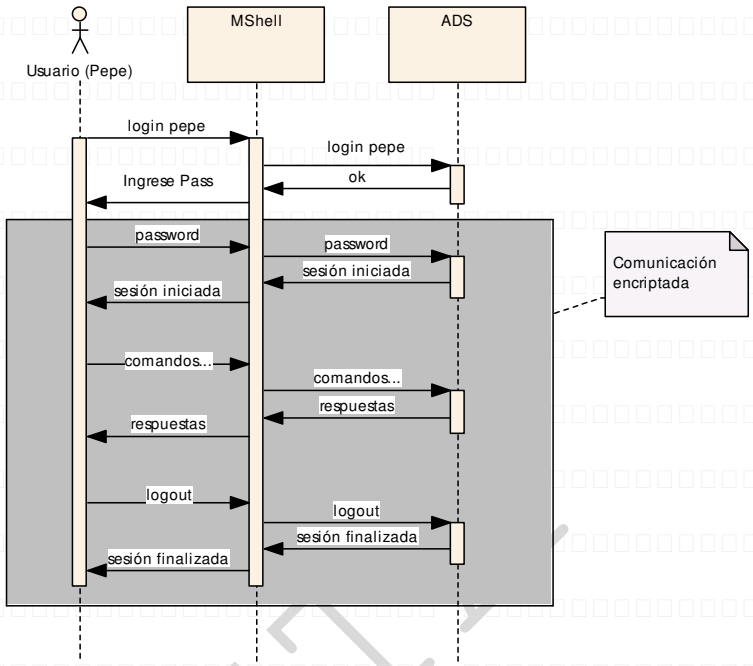


[24] Como el alumno puede apreciar, la complejidad del algoritmo de encripción es muy baja, esto hace que la seguridad provista por el mecanismo de encripción sea muy pobre. La idea de EMULASO es que el alumno aplique de manera práctica los conceptos teóricos de seguridad enseñados en la materia, por lo tanto el objetivo de realizar una operación XOR está más bien ligado a aspectos didácticos que a cuestiones reales de seguridad.
[25] El script de creación de usuarios será proporcionado por la cátedra de Sistemas operativos.
[26] La idea que subyace en la copia manual de la clave está detallada en el Capítulo 16 de sistemas operativos. En él se habla acerca de las diferentes formas de distribución de claves. En EMULASO se optó por la distribución manual (por ejemplo con diskettes) ya que evita el uso del canal de comunicación normal (socket) previniendo así los riesgos de seguridad propios de él.

5. **Algoritmo de desencripción:** Consiste en aplicar nuevamente la operación XOR tal como se lo especificó en el algoritmo de encripción. La única diferencia será que en lugar de aplicarlo sobre el texto plano se aplicará sobre el texto encriptado.

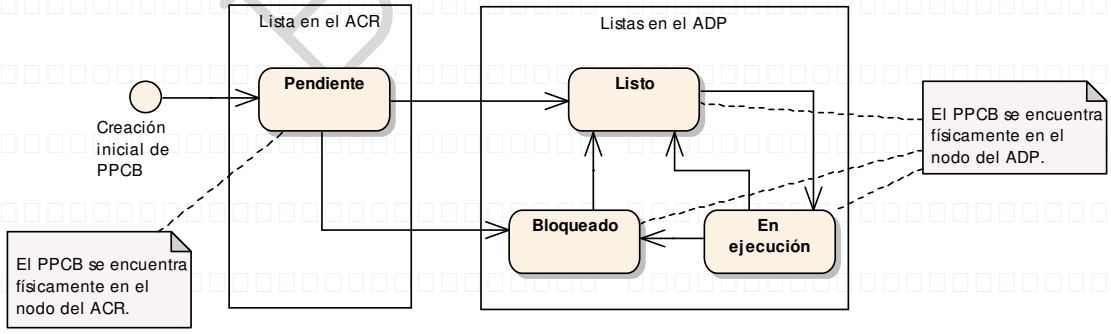
Aclaración importante: Recordar que sólo el canal establecido entre el ADS y Mshell deberá ser encriptado. Los restantes canales, como por ejemplo ADS-ACR; ACR-ADP; ADP-PPCB, trabajarán con flujos de información planos, es decir, sin encriptar.

Diagrama de secuencia que ejemplifica la encripción



Estados de un PPCB / Planificación de PPCBs

A continuación se presenta el diagrama de estados de un PPCB, cada estado se corresponderá con una lista (LPP, LPB, LPL o LPE).



Detalle de los estados

- Pendiente: El PPCB aún no ha sido migrado hacia un nodo de red de carga.
- Bloqueado: El PPCB se encuentra a la espera del otorgamiento de un recurso.
- Listo: El PPCB se encuentra listo para ejecutar, pero aún no posee el quantum.
- En ejecución: El PPCB se encuentra en ejecución, posee el quantum.

Detalle de las transiciones

- Inicio -> Pendiente: Cuando un PPCB es creado por primera vez en el ACR, se inicia en el estado Pendiente.
- Pendiente -> Listo: La transición se produce cuando el PPCB (que no viene del estado Bloqueado) migra desde el ACR hacia un nodo de carga. Esto puede deberse a:
    1. El PPCB fue recién creado y luego migrado.
    2. Se adicionó un ADP al sistema y se cumplen las restricciones para que migre hacia él.
    3. Se modificó el estado de un ADP ya existente y se cumplen las restricciones para que migre hacia él. Esto puede ser provocado por la finalización de un PPCB con lo cual se liberó espacio en el ADP.

  El ACR deberá tener conocimiento de estos eventos para poder realizar la transición en el momento oportuno.
  Cuando el ACR detecte alguno de estos eventos deberá recorrer TODA la LPP (en el orden en el que los PPCB ingresaron) e intentar migrar la mayor cantidad posible de PPCBs.
- Pendiente -> Bloqueado: Esta transición es equivalente a la de Pendiente -> Listo, sólo que este PPCB en el instante previo a pasar a Pendiente se encontraba en estado Bloqueado, con lo cual al salir de Pendiente debe continuar Bloqueado como es lógico.
- Listo -> En ejecución: El ADP le concede el quantum al PPCB. Para ello le envía un mensaje al PPCB indicando que puede ejecutar sentencias.
- En ejecución -> Listo: Se finaliza el quantum del PPCB. Para realizar esto el ADP le enviará un mensaje al PPCB indicando que no puede ejecutar más sentencias.
- En ejecución -> Bloqueado: Un PPCB solicitó una instancia de un recurso que no poseía instancias disponibles. El ADP enviará un mensaje al PPCB indicando que no posee más el quantum (no puede ejecutar más sentencias).
- Bloqueado -> Listo: Se liberó una instancia del recurso que esperaba el PPCB y el ACR informó al ADP para que lo cambie a Listo.

Aclaración importante:
Siempre será posible que desde los estados *Listo*, *En ejecución* y *Bloqueado* se realice una transición al estado Pendiente[27], esto se producirá cuando por alguna causa el PPCB deba ejecutar la función **Conectarse al ACR para indicar el deseo de migrar.**

Formato de archivos

Archivo de usuarios

El formato del archivo de usuarios (de manera análoga al archivo /etc/passwd de linux) deberá ser como se especifica a continuación:

usuario:password:recursosPermitidos

Donde:
1. usuario: Nombre de usuario que podrá acceder al EMULASO.
2. password: Password del usuario (sin encriptar).
3. recursosPermitidos: Lista de recursos compartidos a los cuales el usuario podrá tener acceso (Separados por coma). Esta lista introduce el concepto de Capability List en EMULASO.

Ejemplo:
sole:sole:Impresora,Disco
invitado:invitado:
superusuario:root:Impresora,Disco,Cinta

La inserción de registros sobre el archivo estará a cargo del script de creación de usuarios proporcionado por la cátedra. Esto significa que para añadir un usuario al sistema deberá ejecutarse dicho script y automáticamente se creará el registro en el archivo de usuarios y la clave de (des)encripción asociada al mismo.

Archivos de log

El formato del archivo log deberá ser como se especifica a continuación:

Fecha NombreProceso [PIDProceso] TipoEvento: Data

[27] En el diagrama no se expresa para no dificultar la comprensión.

Donde:

1. Fecha: Fecha del sistema en formato Hora:Minuto:Segundo:Milisegundo.
2. NombreProceso: Nombre del proceso que escribe en el log.
3. PIDProceso: Identificador del proceso que escribe en el log.
4. TipoEvento: "Info", "Error", "Advertencia" o lo que se considere apropiado.
5. Data: Información a escribir.

Ejemplo:
15:20:30:3 PPCB 150 INFO: Se detectó la caída del ADT.
15:20:31:2 PPCB 150 INFO: Se intentará migrar a otro nodo de carga.

Renombramiento de procesos

Cada instancia del proceso PPCB deberá respetar el siguiente nombre:

"PPCB (<IdDelPPCB>, <nombrePrograma>)"

De esta manera al ejecutar el comando 'ps –fea' cada PPCB deberá aparecer de la siguiente forma:

guest@localhost:~# ps –fea

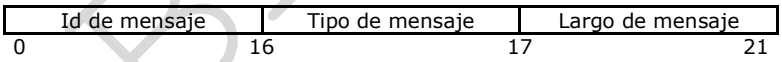
UID	PID	PPID	TTY	CMD
tipi	2005	2004	tty3	vi /etc/passwd
root	3473	3447	tty8	bash
tipi	3900	3801	tty3	nmap www.google.com
guest	5000	4500	tty1	ADP
guest	5005	5000	tty1	**PPCB (3, programa1)**
guest	5006	5000	tty1	**PPCB (5, programa2)**

Tipos de comunicación entre procesos

La comunicación del Mshell-ADS, ACR-ADP y ACR-PPCB (Conexión TEMPORAL que se establece al iniciar un pedido de migración por parte del PPCB) deberá ser realizada mediante sockets del tipo AF_INET.
Los restantes canales de comunicación podrán ser seleccionados libremente por el grupo siempre y cuando no quiten o modifiquen las funcionalidades especificadas en el documento.

Definición de Protocolo de comunicación

La cátedra de Sistemas operativos recomienda a los alumnos el uso de la siguiente cabecera (header) para cada mensaje enviado:



Donde:

- Id de mensaje (16 bytes): Valor de 16 bytes que identifica unívocamente al mensaje en toda la red.
- Tipo de mensaje (1 byte): Valor que clasifica al mensaje. *Por ejemplo: Pedido de migración (0x01)*.
- Largo de mensaje (4 bytes): Valor que indica la cantidad de bytes que tendrá el cuerpo del mensaje (la información propiamente dicha).

Glosario

- Sesión: Relación lógica que se establece entre un usuario y una aplicación servidor mediante la cuál se lleva a cabo la comunicación.
- Nodo de red: Computadora conectada físicamente a la red, se la identifica de manera unívoca mediante un número denominado IP.
- Nodo de red de carga: Nodo de red sobre el cuál se ejecutó un ADP. Este tipo de nodo es susceptible a recibir carga (PPCBs) por parte del ACR.
- Proceso: Instancia de un programa ejecutando en un computador. Un proceso es administrado (controlado y planificado) por el sistema operativo. Para una definición más precisa ver el capítulo 3 de Sistemas operativos (Stallings, 5ta edición).

- Meta: Prefijo utilizado para indicar que un elemento se encuentra por sobre otro elemento del mismo tipo. En el caso de Meta Shell se hace referencia a que el proceso es un tipo especial de shell que se crea y ejecuta por sobre el shell (bash) provisto por el sistema operativo.
- Shell: Sistema que actúa como interfaz entre el usuario y la aplicación (generalmente el sistema operativo). Traduce los eventos del usuario a eventos comprensibles por la aplicación destino y viceversa.
- Carga de un sistema operativo: Conjunto formado por aquellos procesos y threads que existen en el sistema operativo y consumen recursos del mismo.
  Para el caso de EMULASO la carga será representada por los PPCBs.
- Balanceo de carga: Técnica por la cuál se intenta distribuir, en base a un determinado criterio, la carga de un sistema a lo largo de una serie de recursos que aceptan carga.
  Ejemplo de balanceo puede ser un sistema operativo que distribuye procesos o threads a lo largo de diversas CPUs (SMP – Capítulo 4 Stallings) ó, en el caso de EMULASO, un sistema que distribuye PPCBs a lo largo de nodos de red de carga (Ver una analogía a EMULASO en Clusters – Sección 14.4 Stallings).
- Encripción simétrica: Mecanismo por el cual un texto plano procesado por un algoritmo de encripción que utiliza una clave produce como salida un texto cifrado que podrá ser reconvertido si se aplica un algoritmo de encripción "inverso" con la misma clave. Para profundizar sobre el tema leer el capítulo 16 de Sistemas operativos (Stallings, 5ta edición).
- Semáforo: Mecanismo utilizado para sincronizar procesos y/o permitir la mutua exclusión. Para conocer más información dirigirse al Capítulo 5 de Sistemas operativos (Stallings, 5ta edición).
- Señal: Mecanismo que provee el sistema operativo para notificar a un proceso de la ocurrencia de un evento asincrónico. Para profundizar leer sección 6.7 de Sistemas operativos (Stallings, 5ta edición).
- Migración: Operación por la cual un proceso que reside en una máquina M1 se traslada a una máquina M2 conservando el estado que tenía en el instante inmediatamente anterior a la migración. Si la operación es exitosa el proceso de M1 es eliminado. Para mayor información, Sección 15.1 de Sistemas operativos (Stallings, 5ta edición).
- Quantum: Período máximo de tiempo por el cual un proceso podrá hacer uso de la CPU. El quantum es utilizado por el algoritmo Round Robin. Para mayor información, Capítulo 9 de Sistemas operativos (Stallings, 5ta edición).
  Para el caso de EMULASO el quantum representa el tiempo máximo concedido por el ADP por el cual un PPCB podrá ejecutarse de manera continua.
- PCB (Process control block): Estructura de datos que utiliza el sistema operativo para administrar procesos. Para más información leer Sección 3.2 de Sistemas operativos (Stallings, 5ta edición).
- CL (Capability list): Lista que indica para un determinado usuario los objetos (recursos en EMULASO) a los cuáles tendrá permiso. Para más información leer Sección 16.2 de Sistemas operativos (Stallings, 5ta edición).