Práctica 2. Modelo Cliente / Servidor
"Administrador de Procesos y Mecanismo de Comunicación Básico"

Objetivo

Implementar una simulación de los servicios para comunicación básica entre procesos, las operaciones send y receive, que proporciona un núcleo de sistema operativo distribuido de estructura tipo micro núcleo y efectuar la interacción entre este y los procesos de nivel usuario (procesos clientes y servidores).

Requerimientos

La simulación del modelo Cliente/Servidor deberá cumplir con lo siguiente:

1. Todos los requerimientos especificados para la práctica 1.

2. Mandar a imprimir en la interfaz gráfica (IG) correspondiente al núcleo, cada suceso significativo en que éste se encuentre involucrado:

a) "Buscando en listas locales el par (máquina, proceso) que corresponde al parámetro dest de la llamada a send"
b) "Enviando mensaje de búsqueda del servidor" (imprimirlo si el destinatario no se conoce, que es el caso para el proceso servidor la primera vez que se le busque; para esta práctica, se simulará la búsqueda tomando de la IG los datos de la ubicación del servidor
c) "Recibido mensaje que contiene la ubicación (máquina, proceso) del servidor"
d) "Completando campos de encabezado del mensaje a ser enviado"
e) "Enviando mensaje por la red"
f) "Recibido mensaje proveniente de la red"
g) "Buscando proceso correspondiente al campo dest del mensaje recibido"
h) "Copiando el mensaje hacia el espacio del proceso"
i) "Proceso destinatario no encontrado según campo dest del mensaje recibido"

3. Los mensajes que los procesos han de enviarse entre sí, sean en forma de un arreglo de bytes, el cual tenga la estructura siguiente:

a) Primeros 2 bytes (campo origen): Identificador del emisor (ID de proceso emisor, completado por el núcleo)
b) Siguientes 2 bytes (campo destino): Identificador del receptor (ID de proceso receptor; completado por el núcleo)
c) Siguientes 1020 bytes: Datos relativos a la operación solicitada o respuesta, en su caso; se pueden reservar para ello de 1 hasta 1020, dependiendo de la información a enviar.

4. El núcleo incluya las primitivas send(dest,messagePtr) y receive(addr,messagePtr), a través de las cuales se transfieran los mensajes entre cliente y servidor

dest = identificador del proceso destinatario.

addr = ID del proceso que espere recibir un mensaje.

messagePtr = apuntador a un arreglo de bytes definido en el espacio del proceso (cliente ó servidor).

5. Las primitivas send(dest,messagePtr) y receive(addr,messagePtr) sean utilizadas exclusivamente por los procesos.

6. La primitiva send(dest,messagePtr) sea implementada contemplando sólo el caso remoto; esto es, no realice alguna verificación por encontrar al proceso destinatario de forma local para efecto de envío del mensaje.

7. El proceso que invoque a send recupere el control después de que haya sido enviado el mensaje por la red (primitivas por bloqueo).

8. El proceso que invoque a receive, recupere el control después de que el mensaje recibido en el espacio del núcleo haya sido copiado a su espacio de memoria (primitivas por bloqueo).

9. Las primitivas send(dest,messagePtr) y receive(addr,messagePtr) hagan uso de ambos parámetros recibidos y únicamente utilicen variables externas al procedimiento si tienen que ver con alguna estructura de datos involucrada en el mecanismo de comunicación.

10. Para el envío del mensaje, no se intente averiguar la naturaleza del proceso que desea enviar el mensaje (no verificar si es cliente o servidor)

11. Todo proceso sea ubicado remotamente mediante el par (máquina, proceso), donde máquina es una dirección IP y proceso es un ID de proceso en la máquina remota; para esto, el parámetro dest proporcionado en la llamada a send(dest,messagePtr) sea verificado por el núcleo para determinar el par (máquina, proceso) que le corresponde a este dest, de la siguiente forma:

a) El núcleo verifique en alguna estructura de datos propia (una tabla), si se encuentra una entrada cuya clave primaria sea igual a dest
b) Si se encuentra, se extraigan los campos IP e ID de proceso
c) Si no se encuentra, se envíe un mensaje de búsqueda del proceso destinatario, esto es, haciendo la simulación de una posterior implementación de búsqueda en red ya sea por multitransmisión (direcciones ralas de procesos) o mediante un servidor de nombres; (estas implementaciones corresponden a una práctica posterior); la forma de ubicar al proceso remoto para los fines de esta práctica, será tomando de la interfaz gráfica del núcleo los datos correspondientes a la IP y al ID de proceso del destinatario.
d) Una vez que se hayan obtenido la IP e ID de proceso del destinatario, el ID de proceso destino sea colocado en el campo destino del mensaje a enviar, el ID del proceso emisor se coloque en el campo origen y la IP se utilice para enviar el mensaje por la red.
e) Los ID de proceso correspondientes a los campos origen y destino del mensaje a enviar sean colocados en el arreglo de bytes respetando orden de magnitud, esto es, el byte de mayor orden del tipo de dato entero de 4 bytes sería colocado a la izquierda del byte de orden inmediato menor y así sucesivamente (para esta práctica basta con colocar el byte de menor orden del ID de proceso en el lugar correspondiente).
f) La interfaz gráfica sólo se utilice para localizar al servidor, no al cliente.

12. La IP del emisor o del receptor nunca sea incluida en el arreglo de bytes proporcionado en la llamada a send ni en el enviado por la red (esto excluye el uso de APIs, librerías o instrucciones de bajo nivel requeridas para el verdadero envío de los mensajes por la red).

13. El proceso servidor solicite al núcleo el envío de la respuesta, otorgando como argumento la información del campo origen de la solicitud procesada.

14. Para la recepción de un mensaje por parte de un proceso, se recurra a una estructura de datos (una tabla) dedicada a los procesos que desean recibir un mensaje, aquellos que han invocado a la primitiva receive(addr,messagePtr); utilizar a addr como la llave que identifique la intención de recepción de mensaje, dada por el proceso que invoque a dicha primitiva y proporcionarle a la mencionada estructura de datos, la llave y el apuntador al arreglo de bytes messagePtr correspondiente a fin de que pueda ser obtenido por parte del hilo de control descrito en el siguiente punto.

15. El núcleo designe un hilo de control particular que se dedique a la recepción de mensajes por la red, el cual:

a) Se designe un búfer en el espacio de direcciones del núcleo para recepción de mensajes provenientes de la red, el cual siempre se localice en la misma dirección de memoria relativa.
b) Se prepare para la recepción de un mensaje proveniente de la red.
c) A la llegada de un mensaje, muestre en su IG la IP e ID de proceso del emisor.
d) Para un mensaje recibido, revise el campo dest del mensaje.
e) Verificar si el proceso destino al que hace referencia el mensaje se encuentra entre los que este núcleo administra.
f) Si se encuentra el destinatario según el punto anterior y tal proceso espera recibir un mensaje, el hilo del núcleo copie el mensaje al espacio de memoria del proceso; para esto se requiere de la estructura de datos mencionada en el punto anterior.
g) Extraiga la dirección IP del emisor a partir del mensaje recibido.
h) Guarde la IP e ID de proceso del emisor del mensaje en la misma estructura de datos que sea consultada en la primitiva send, a fin de que se pueda localizar al destinatario (el cliente) para cuando el servidor envíe la respuesta; lo anterior llevado a cabo sí y solo sí fue determinado en el inciso “f” que el destinatario esperaba recibir un mensaje; con la información guardada se debe poder distinguir entre varios clientes pendientes de respuesta;
i) Si determina que el proceso al que hace referencia el mensaje no se encuentra, envíe un mensaje al núcleo emisor de la solicitud, indicando la ausencia del destinatario (paquete AU, dirección desconocida); se notifique en la IG de cada uno de los núcleos involucrados (cliente y servidor) la respectiva falla.
j) Sea desbloqueado el proceso al cual se haga referencia en el paquete AU recibido, si este se encuentra bloqueado por llamada a receive.
k) Este hilo de control termine su ejecución al momento de solicitar desde la interfaz gráfica el cierre de la aplicación.
l) Los mensajes/paquetes núcleo a núcleo (AU, TA, ACK, AYA, IAA) sean del tamaño mínimo indispensable para que los núcleos se entiendan entre sí; dichos mensajes no incluyan información a interpretar como caracteres, ni información de los mensajes REQ ni REP más allá de los campos origen y destino.

16. Ejecución de la práctica a prueba de fallas tanto en núcleo como en procesos considerando el universo de posibilidades en la elaboración de mensajes por parte de los procesos y el núcleo; esto es, que sea cual sea el contenido de los mensajes elaborados tanto por los procesos como por el núcleo, siempre que respeten el protocolo de requerimientos especificados (en esta práctica, anteriores y subsecuentes), no

causen fallo en el cumplimiento del paso de mensajes. Así mismo, el mecanismo de comunicación debe contemplar la ejecución de varios servidores y clientes simultáneos.

Interfaz Gráfica. Debe ser implementada utilizando SWING.

Núcleo:

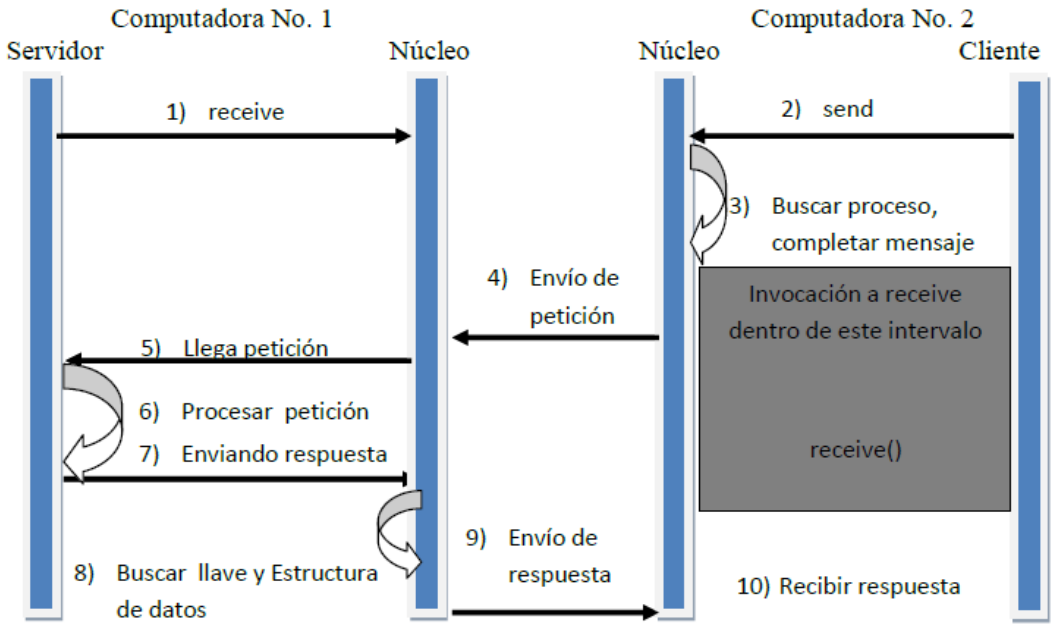
- Un JTextArea para visualizar los eventos que está realizando con la propiedad activada de sólo lectura.
- Un botón para crear a los servidores.
- Un botón para crear a los clientes.
- Un botón para cerrar la aplicación.
- Un botón para eliminar procesos, la cual debe mostrar una ventana que contenga la dirección IP y el ID de todos los procesos, para poder seleccionar el proceso a eliminar.

Cliente:

- Un JTextArea para visualizar los eventos que está realizando con la propiedad activada de sólo lectura.
- Un botón para cerrar la aplicación.
- Un botón para enviar datos a un servidor.
- De uno a dos JtextField, que serán los operandos para una operación definida por el usuario.
- Un botón que abre otra ventana, la cual debe mostrar la dirección IP y el ID proceso.
- Un comboBox para seleccionar el tipo de operación

Servidor:

- Un JTextArea para visualizar los eventos que está realizando con la propiedad de solo lectura.
- Un botón que abre otra ventana, la cual debe mostrar la dirección IP y el ID proceso.
- Un botón para cerrar la aplicación localmente.



Actividades

1. Describa los tipos de paquetes utilizados en los protocolos cliente-servidor (AU, TA, ACK, AYA, IAA, REQ, REP).

Criterios de Evaluación

- La práctica se revisa en clase la fecha señalada.
- Fecha de entrega de la práctica: 29 de Abril de 2015.
- La práctica se subirá a la plataforma de moodle en la fecha indicada, deberá subirse un archivo comprimido, el cual contendrá: Código Fuente (indicando en la parte superior de **cada archivo** el nombre del alumno(a), la sección y el número de práctica), Reporte en electrónico, Ejecutable. El nombre del archivo zip deberá estar integrado por su apellido paterno, nombre, número de práctica y sección.

* Ejemplo Nombre Archivo Comprimido:

Gutierrez_Luis_D04_J2_TSOA16B.zip