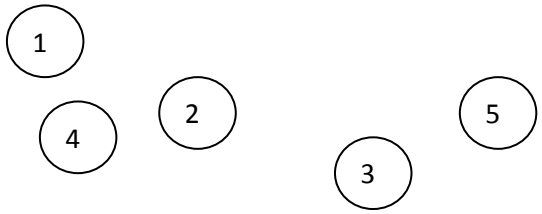
UNIVERSIDAD CARLOS III DE MADRID
AREA DE ARQUITECTURA Y TECNOLOGÍA DE COMPUTADORES
GRADO EN INGENIERÍA INFORMÁTICA. SISTEMAS DISTRIBUIDOS

Para la realización del presente examen se dispondrá de **1 hora y 15 minutos. NO** se podrán utilizar libros, apuntes **ni** calculadoras de ningún tipo.
Alumno: _____**Grupo:** _____

Ejercicio 1 (2,5 puntos). Responda a las siguientes preguntas cortas justificando brevemente su respuesta:

a) Describir brevemente qué es un servicio web.
b) En una transacción HTTP identificar los principales campos de la línea de petición y de respuesta.
c) Definir brevemente en qué consiste el multicast atómico.
d) ¿Cuál es la función del servicio *portmapper*?
e) Dado el siguiente conjunto de n nodos, con n=5, que usa el método de votación dinámica (quórum) para mantener la consistencia de las réplicas.



Suponga que en un momento de la ejecución se produce un fallo en la red que da lugar a dos particiones: {1,2,4} y {3,5}. En ese momento los nodos tienen los siguientes valores NV y SC para una determinada réplica:

	Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
SC	5	5	5	5	5
NV	3	3	3	3	3

NV= número de versión SC=cardinalidad de actualización

En esta situación: ¿se podría actualizar en la partición {1,2,4}? Justifique su respuesta y complete la siguiente tabla:

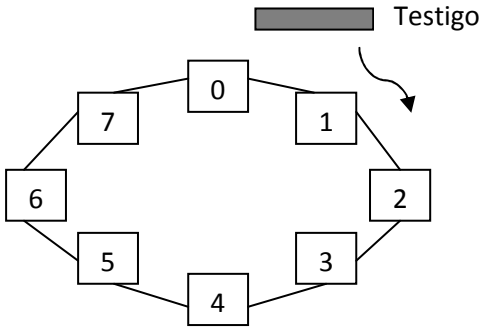
	Nodo 1	Nodo 2	Nodo 3	Nodo 4	Nodo 5
SC					
NV					

f) Un cliente de NFS realiza la siguiente operación de apertura de un fichero remoto:
   - fd=open("/home/user/data.txt");

   Describir el conjunto de llamadas NFS necesarias para abrir ese fichero en el servidor NFS.

g) ¿Cuál es la función del servicio *mount*?
h) ¿Cuáles son las tres principales características de la computación grid según el artículo "What is the Grid? A Three Point Checklist" de Ian Foster (2002).

Ejercicio 2 (3 puntos). Se desea usar el algoritmo del anillo con testigo para implementar exclusión mutua en un sistema local. En el algoritmo del anillo con testigo un conjunto de N procesos (i = 0..N-1) se organizan en forma de anillo, de manera que un proceso i sólo puede comunicar con el proceso i+1 y el proceso i-1. Por el anillo circula un testigo (token) de manera que sólo puede entrar a ejecutar la sección crítica (SC) aquel proceso que tenga el testigo. El sentido en que circula el anillo es unidireccional. Si un proceso recibe el testigo pero no quiere entrar a ejecutar en la sección crítica, reenvía el testigo al siguiente proceso. Si por el contrario quiere entrar en la SC debe esperar a recibir el token; una vez obtenido entra en la SC y a la salida simplemente reenvía el token al siguiente proceso. Se pide implementar la función **anillo** para obtener exclusión mutua usando el algoritmo previamente descrito y las colas de mensajes de UNIX. Considere que comienza a ejecutar el proceso con identificador 0.



Utilice el siguiente código de apoyo correspondiente a la función principal main:

```
#define    N    8
pthread_mutex_t mutex;
pthread_cond_t copiado;
int seguir = FALSE;

void main(){
      pthread_t threads[N];
      pthread_mutex_init(&mutex);
      pthread_cond_init(&mutex);

      // Crear los procesos del anillo
      for (int i=0;i<N;i++){
            pthread_mutex_lock(&mutex);
            pthread_create(&threads[i], NULL, anillo, &i)
            while(!seguir){
                  pthread_cond_wait(&mutex, &copiado);
            }
            seguir = FALSE;
            pthread_mutex_unlock(&mutex);
      }
      for (i=0;i<N;i++)
            pthread_join(threads[i]);

      pthread_mutex_destroy(&mutex);
      pthread_cond_destroy(&copiado);
}
```

Ejercicio 3 (3 puntos) Se quiere diseñar un servicio de almacenamiento de imágenes. El servicio deber permitir:

- Almacenar una nueva imagen. El cliente que accede al servicio puede almacenar en el servidor una nueva imagen. Una imagen viene dada por su nombre, su fecha de creación y el archivo de la imagen. El contenido de este archivo se enviará al servidor para su almacenamiento.
- Borrar una imagen. El servicio borrará una imagen dado su nombre. El cliente enviará al servidor el nombre de la imagen y el servidor eliminará la imagen asociada.
- Obtener una imagen. El cliente puede recuperar del servidor una imagen. El cliente enviará al servidor el nombre de la imagen y servidor enviará la imagen al cliente para que este almacene la imagen en un fichero local.

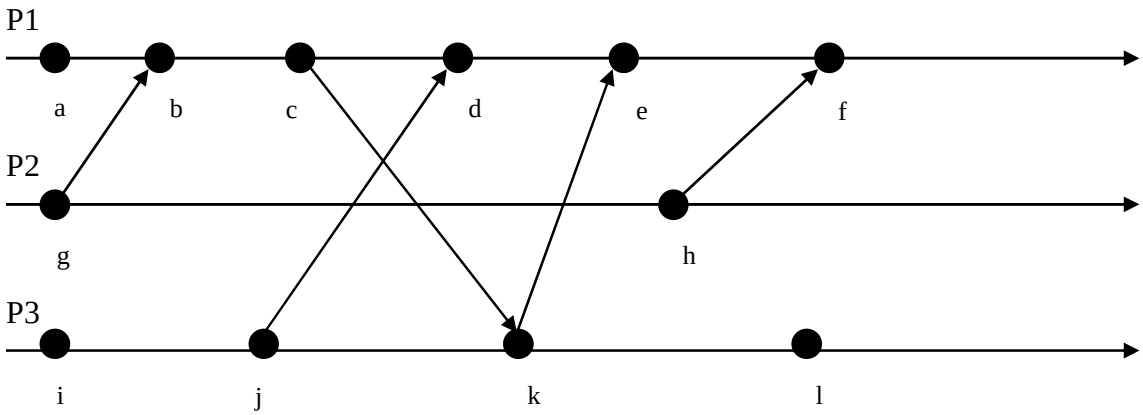
Tenga en cuenta que el archivo de la imagen puede tener cualquier tamaño.

Se pide:

a) Diseñe la aplicación cliente-servidor anterior utilizando sockets, indicando y especificando todos los aspectos necesarios para su diseño.

b) De acuerdo al diseño anterior, indique qué llamadas a la biblioteca de sockets utilizaría en el cliente y en el servidor y en qué orden.
c) Considerando que se emplean RPC con un lenguaje de definición de interfaces similar a la sintaxis del lenguaje C, defina la interfaz necesaria para poder implementar la aplicación cliente-servidor anterior.
d) Especifique la misma interfaz utilizando las RPC de Sun.

Ejercicio 4 (1.5 puntos). El siguiente diagrama muestra una serie de eventos que ocurren entre estos tres procesos. Estos eventos tienen una marca de tiempo asignada (*timestamp*).



Se pide:

a) Utilizando relojes lógicos de Lamport, indique las marcas de tiempo para los eventos que aparecen en la figura.
b) ¿Qué significa que dos eventos sean concurrentes? Indicar cuáles son los eventos concurrentes en el ejemplo de la figura.
c) Responda a las siguientes preguntas:
    1. ¿Obtenemos una relación de orden total con los relojes de Lamport? Razone la respuesta.
    2. Si no es así, ¿cómo podemos conseguir que exista una relación de orden total entre los procesos?
d) Utilizando relojes vectoriales, indique las marcas de tiempo para los eventos que aparecen en la figura.