

[75.54] Sistemas Distribuidos I

2do cuatrimestre 2017

TP primera entrega

|  |  |  |
| --- | --- | --- |
| **Nombre y apellido** | **Padrón** | **Email** |
| Tobias Bianchi di Carcano | 96724 | tobias.bianchi93@gmail.com |
| Marcos Pernica | 94349 | marcospernica@gmail.com |

**Breve explicación de la estructura del programa**

El programa está compuesto por un proceso ‘Cine’ que se encarga (a modo de punto de entrada a un servidor) de atender solicitudes de operaciones mediante una cola y por cada solicitud disparar un proceso que se encargue de atender al cliente.

El proceso ‘CineHijo’ toma las solicitudes y respuestas del usuario y las envía al proceso ‘administrador’ para ser evaluadas. Luego recibe la respuesta y se la envía al cliente. Además hace un seguimiento sencillo de las solicitudes del usuario para implementar un sistema de timeout dado que si el usuario deja de responder se necesitan liberar los asientos reservados y no comprados.

El proceso ‘administrador’ es el encargado y unido responsable de acceder y modificar la base de datos y procesar todas las solicitudes de los clientes de forma secuencial. Si como resultado de una consulta se modifica el estado de una asiento en una sala entonces se envía mediante una cola especial a tal fin el nuevo estado de todos los asientos de esa sala a todos los usuarios que estén en ese momento operando sobre ella.

La base de datos esta implementada por varias estructuras una de las cuales es un arreglo tridimensional que indexa primero por numero de sala, luego por fila y finalmente por columna de la sala y se obtiene el estado de ese asiento. Por otra parte se tienen dos estructuras, una de ‘usuariosPorSala’ que indica que usuarios están viendo una determinada sala y otra ‘salaPorUsuario’ que informa que sala está mirando un determinado usuario, estas estructuras sirven para facilitar el proceso de entrada y salida de las salas así como la liberación de asientos reservados por un usuario. Se decidió esta redundancia para priorizar tiempo de procesamiento por sobre memoria utilizada.

El proceso ‘ClienteAsinc’ es el encargado de tomar estos mensajes asíncronos y actualizar la memoria compartida intermedia del ‘Cliente’ lo que permite que al momento de intentar reservar un asiento que se sabe ya tomado por otro cliente no se necesite una petición al servidor para saber dicho inconveniente.

**Compilación y ejecución**

Para la compilación se debe ir con la terminar a la carpeta raíz del programa y escribir ´make all´ lo que generara los archivos ejecutables ´client´ y ´cine´. Finalmente lanzar primero ‘cine’ y luego el ´client´ siguiendo los menús de este para acceder a la reserva de asientos.