

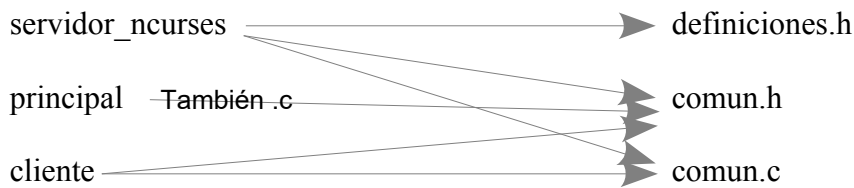
Índice

1.Ficheros.....	1
2.Eschema de creación.....	1
3.Eschema de sincronización.....	1
4.Eschema de comunicación.....	2
5.Funciones especiales.....	2
6.Procesos.....	4

1. Ficheros

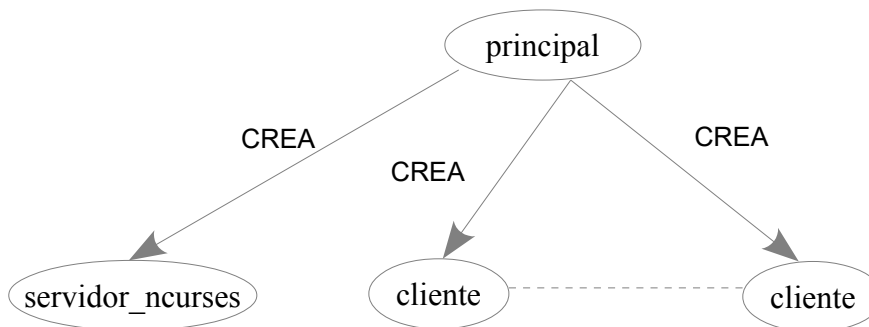
Procesos

Librerías



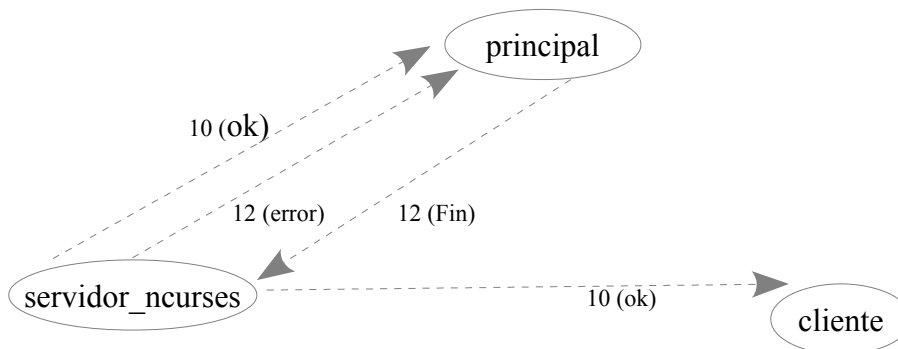
fichcola.txt → Para crear cola de mensajes

2. Esquema de creación

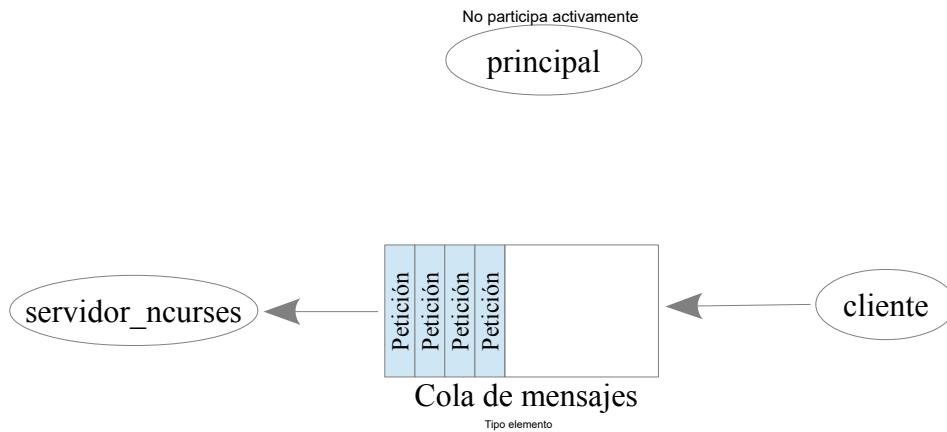


Bucle desde 1 hasta 30 (maxclientes)

3. Esquema de sincronización



4. Esquema de comunicación



5. Funciones especiales

Función creaproceso

(Va en principal.c)

```
int creaproceso(const char nombre[])
{
    int vpid;

    vpid=fork();
    if(vpid==0)
    {
        execl(nombre,nombre,NULL);
        perror("error de execl");
        exit(-1);
    }
    else if (vpid==-1)
    {
        perror("error de fork");
        exit(-1);
    }
    return vpid;
}
```

Función creaservigraf

(Va en principal.c)

```
int creaservigraf(int ultimaparada)
{
    int vpid;
    char cadparada[10];

    sprintf(cadparada,"%d", ultimaparada);

    vpid=fork();
    if(vpid==0)
    {
        execl("servidor_ncurses","servidor_ncurses",cadparada,NULL);
        perror("error de execl");
        exit(-1);
    }
    else if (vpid==-1)
    {
        perror("error de fork");
        exit(-1);
    }
    return vpid;
}
```

Función creacola

(Va en comun.c)

```
int crea_cola(key_t clave)
{
    int identificador;
    if(clave == (key_t) -1)
    {
        printf("Error al obtener clave para cola mensajes\n");
        exit(-1);
    }

    identificador = msgget(clave, 0600 | IPC_CREAT);
    if (identificador == -1)
    {
        printf("Error al obtener identificador para cola mensajes\n");
        perror("msgget");
        exit (-1);
    }

    return identificador;
}
```

Función visualiza

(Va en cliente.c)

```
void visualiza(int cola, int parada, int inout, int pintaborra,
int destino)
{
    struct tipo_elemento peticion;

    peticion.tipo=2; //Los clientes son tipo 2, el autobus tipo 1
    peticion.pid=getpid();
    peticion.parada=parada; //ventana en la que se muestra.0 es bus
    peticion.inout=inout; //IN es que llega. OUT es que se baja
    peticion.pintaborra=pintaborra; // PINTAR es para visualizarlo
                                // BORRAR es para borrarlo
    peticion.destino=destino; //parada de destino

    msgsnd (cola, (struct tipo_elemento *) &peticion,sizeof(struct
tipo_elemento)-sizeof(long),0);

    if(pintaborra==PINTAR)
    {
        if(!llega10) pause();//Espero conformidad de que me han
pintado,
        llega10=0;           //sino me mataran
    }

}
```

6. Procesos

principal.c

```
main()
{
    Prepararse para recibir señales 10 y 12;
    Crear servidor_ncurses con creaservigraf;
    Inicializamos semilla;
    Esperar señal de ok o de error del servidor gráfico;
    for (i=1;i<=MAX_CLIENTES;i++)
    {
        Crear cliente con creaproceso;
        Espera aleatoria antes de crear otro cliente;
    }
    Espera fin de sus hijos;
    Avisa a servidor grafico con señal 12;
}
```

cliente.c

```
main()
{
    Prepararse para recibir señal 10;
    Crear la cola de mensajes;
    Inicializamos semilla;
    Generar parada de llegada y de bajada distintas aleatorias;
    Se visualiza en la parada de llegada;
    Se espera un tiempo;
    Se borra;
    Se visualiza en la parada de bajada;
}
```