**CC41B: Sistemas Operativos**
**Tarea 2 - Semestre Primavera'2005**
**Plazo de entrega: Jueves 27 de Octubre**
**Prof.: Luis Mateu**

## Multicast en nSystem

En esta tarea Ud. debera incorporar a nSystem un sistema de multicast. Este permite enviar el mismo mensaje a varias tareas receptoras, denominadas oyentes. El siguiente ejemplo muestra el sistema en acción:

| *Emisor* | *Oyentes* |
|---|---|
| `nTask oyentes[3]; int i, j;`<br>`nTask yo= nCurrentTask(); /* emisor */`<br>`for (i= 0; i<3; i++)`<br>`  oyentes[i]= nEmitTask(oyente, yo);`<br>`for (j=1; j<=10;j++)`<br>`  nMulticast(oyentes, 3, &j, -1);`<br>`for (i= 0; i<3; i++)`<br>`  nWaitTask(oyentes[i]);` | `int oyente(nTask emisor) {`<br>`  int j, *pmsg;`<br>`  do {`<br>`  pmsg= (int*)nListen(emisor);`<br>`  j= *pmsg;`<br>`  nReplyMulticast(emisor);`<br>`  nPrintf("%d\n", j);`<br>`  } while (j!=10);`<br>`  return 0;`<br>`}` |

Cada una de los 3 oyentes despliega 1, 2, 3, ..., 10 en pantalla, en una mezcla impredescible. La API es la siguiente:

- `int nMulticast(nTask *oyentes, int numOyentes, void *msg, int timeout)`: emite un mensaje multicast. El parámetro `oyentes` es un arreglo de `numOyentes` elementos con las tareas a las que se destina el mensaje `msg`. El procedimiento nMulticast se bloquea hasta que todos los oyentes que recibieron el mensaje con `nListen`, lo hayan respondido con `nReplyMulticast`. El parámetro `timeout` es el tiempo máximo de espera para que los destinatarios reciban el mensaje con `nListen`. Después de ese plazo, las nuevas invocaciones de nListen deben esperar una futura llamada de nMulticast.
- `void *nListen(nTask emisor)`: recibe y retorna un mensaje emitido por `emisor`. Si el mensaje ya fue emitido, `nListen` retorna de inmediato el mensaje. En caso contrario se bloquea indefinidamente hasta que se emita. Un mensaje se recibe una sola vez, es decir que si al momento de invocar `nListen` el emisor todavía se encuentra bloqueado en un `nMulticast` del cual ya se recibió el mensaje, entonces `nListen` se bloquea a la espera de una futura llamada de `nMulticast`.
- `void nReplyMulticast(nTask emisor)`: responde un multicast emitido por `emisor`.

## Requerimientos

Ud. debe implementar el sistema de multicast usando los procedimientos de bajo nivel de nSystem para la implementación de herramientas de sincronización (START_CRITICAL, END_CRITICAL, PushTask, GetTask, ResumeNextReadyTask, ProgramTask, CancelTask, etc.). Ud. *no puede usar* las herramientas de alto nivel (monitores, mensajes, semáforos, etc.).

Para implementar la API solicitada Ud. debe modificar los siguientes archivos:

- include/nSystem.h: agregue los encabezados de los procedimientos de la API.
- src/nSysimp.h: agregue al descriptor de tarea los campos que Ud. estime convenientes para implementar la API. Defina además los estados de proceso que estime necesarios.
- src/nMulticast.c: agregue este archivo y coloque en el la implementación de los procedimientos de la API.
- src/Makefile: agregue nMulticast.o a la variable NSYSTEM.

**Recursos**

En `http://www.dcc.uchile.cl/~lmateu/CC41B` Ud. encontrará testmcast.c, el programa de prueba de esta tarea. Este programa le dirá si su tarea funciona.

**Plazo de entrega**

La tarea se entrega en U-cursos. Para ello entregue un archivo tar comprimido con gzip que incluya todos los archivos de nSystem que Ud. modificó. No incluya archivos binarios. El plazo de entrega vence el Jueves 27 de Octubre. Se descontará medio punto por día hábil de atraso.