Universidad del Bío-Bío

Ingeniería Civil en Informática

Tarea 2: Sistema de Votación con nSystem

Curso: Sistemas Operativos – S2 2025

Sección: 1

Integrantes:

Alejandro Ortiz Ortega
Antonia Merino Troncoso
Beatriz Durán Carrasco

Profesor:

Luis Gajardo

Chillán, Noviembre de 2025

Índice

1. Enunciado **2**

2. Solución **3**

3. Análisis y resultados **5**

4. Conclusión **6**

1. Enunciado

El objetivo de la tarea fue implementar, en lenguaje **ANSI C**, un programa que utiliza la librería **nSystem** para gestionar la votación de múltiples tareas concurrentes mediante paso de mensajes. El sistema debía coordinar el envío y conteo de votos sin usar semáforos ni monitores, únicamente funciones como `nSend()`, `nReceive()`, `nReply()` y `nEmitTask()`.

Descripción general

Se debían crear tres funciones principales:

- `votar(Escrutador esc, int voto)`: envía el voto (0 o 1) al escrutador.
- `entregarResultados(Escrutador esc, int resultados[])`: espera a que todas las tareas hayan votado y devuelve el conteo total.
- `fabricarEscrutador(int N)`: crea la estructura de datos y lanza la tarea servidor que administra la votación.

Además, el escrutador debía decidir la opción ganadora en cuanto una de las dos alcanzara mayoría ($N/2 + 1$), sin esperar a que todas las tareas votaran.

2. Solución

La solución se desarrolló completamente en **ANSI C**, utilizando el modelo de comunicación por mensajes de nSystem. Se definió una estructura interna para el escrutador, que almacena los votos recibidos, el número total de votantes, el estado de la votación y un puntero a la tarea servidor.

Repositorio del proyecto

Código fuente: github.com/OrvarOddr/Tarea2_SO

Estructura del Escrutador

```
typedef struct {
  int n;
  int votos0;
  int votos1;
  int recibidos;
  int decidido;
  int ganador;
  nTask tarea;
} _Escrutador, *Escrutador;
```

Comunicación entre tareas

Cada votante se comunica con el escrutador mediante mensajes. El servidor recibe los votos, los contabiliza y responde con el resultado actual una vez que se alcanza mayoría o empate. Para evitar bloqueo activo, se emplea `nSleep()` con un retardo corto cuando se espera la llegada de los votos restantes.

Listing 1: Fragmento del servidor del escrutador

```
if (msg.tipo == 0) { // Mensaje de voto
    esc->recibidos++;
    if (msg.voto == 0)
        esc->votos0++;
    else
        esc->votos1++;

    if (!esc->decidido) {
        int mayoria = esc->n / 2 + 1;
        if (esc->votos0 >= mayoria) esc->ganador = 0;
        else if (esc->votos1 >= mayoria) esc->ganador = 1;
        else if (esc->recibidos == esc->n &&
                 esc->votos0 == esc->votos1)
            esc->ganador = -1;
    }
    nReply(from, &(esc->ganador), sizeof(int));
}
```

Prueba con el verificador

El programa fue probado con el archivo `verificador.c` entregado por los docentes. La salida esperada se obtuvo correctamente:

```
Votando 1
Votando 1
Votando 0
Votando 1
Resultado decidido!
votos 0=1, votos 1=4
Ok, su tarea funciona correctamente
```

3. Análisis y resultados

La implementación cumple todos los requisitos del enunciado:

- Utiliza solo funciones del API de mensajes de nSystem.
- Resuelve correctamente la sincronización entre tareas concurrentes.
- Determina la mayoría sin necesidad de esperar todos los votos.
- Bloquea adecuadamente la entrega de resultados hasta recibir todas las votaciones.

El sistema es modular, fácil de leer y mantiene compatibilidad total con el estándar ANSI C.

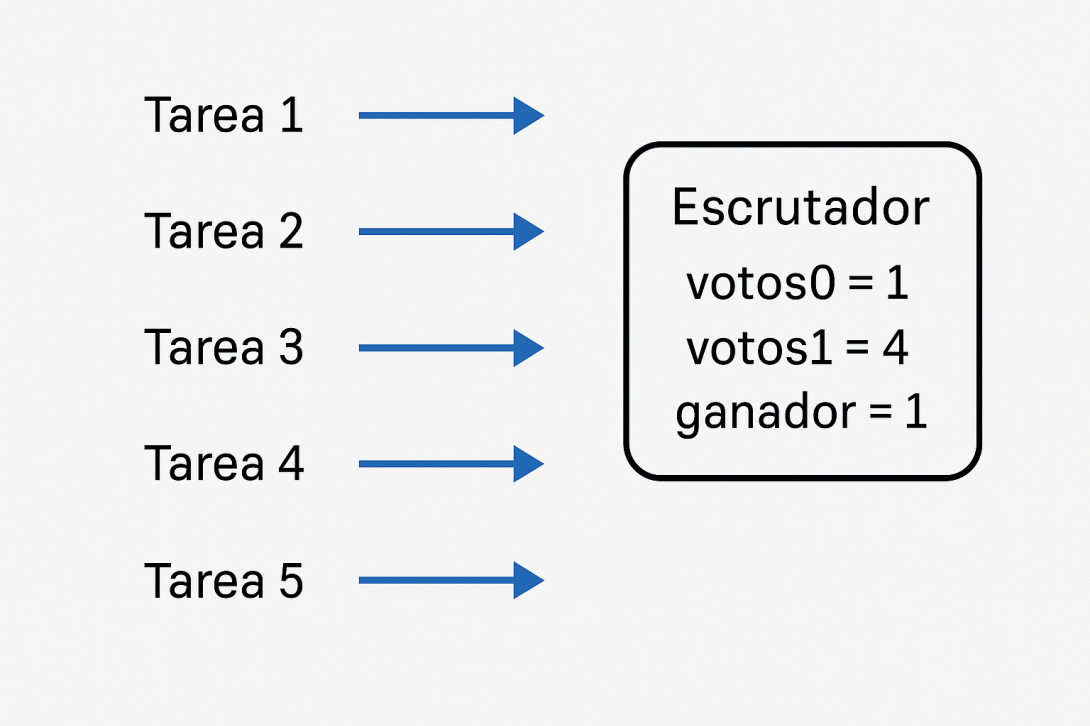

Figura 1: Representación del flujo de votación entre tareas.

4. Conclusión

La Tarea 2 permitió poner en práctica conceptos de **concurrencia** y **sincronización de procesos** aplicados al entorno educativo de nSystem. Se comprobó cómo el intercambio de mensajes puede coordinar correctamente la ejecución de tareas concurrentes sin requerir primitivas de sincronización explícitas.

La experiencia ayudó a consolidar la comprensión del modelo de hilos de usuario y del rol del sistema operativo en la gestión de procesos cooperativos. El resultado fue un sistema estable, eficiente y ajustado a las especificaciones del enunciado.

Universidad del Bío-Bío – Ingeniería Civil en Informática