REPORTE DE INVESTIGACIÓN Nº1

Cristian Herrera, Paul Beltrán.

1) SERVICIOS

1.1. Process Manager (PM)

PM tiene su propia tabla de procesos, la cual es la contraparte en modo usuario de la tabla de procesos del kernel.

La tabla de procesos de PM tiene dos atributos relacionados a la calendarización: mp_nice y mp_scheduler.

PM es el único proceso en modo usuario que sabe quien está siendo calendarizado por quien. Esta información es guardada en mp_scheduler, el que apunta al endpoint del calendarizador.

1.2. Scheduler (SCHED)

En esencia el calendarizador bloquea, ya sea mensajes de control desde PM o mensajes de fuera de quantum desde los procesos. Cuando se recibe un mensaje de control, el calendarizador tomará el control o renunciará a calendarizar un proceso en particular.

Cuando el calendarizador recibe un mensaje de fuera de quantum necesita recalendarizar el proceso usando “sys_schedule”. Aquí es donde la política de calendarización entra en juego. La política dictará en que cola de prioridad el proceso debe ser calendarizado y que quantum debe ser asignado.

La actual implementación del calendarizador (SCHED) imita la política que fue previamente implementada en el kernel. Cada vez que a un proceso se le acaba el quantum, este será degradado de prioridad en uno. Después periódicamente el calendarizador correrá por todos los procesos que han sido degradados y los pondrá en cola.

2) ADMINISTRADOR DE CALENDARIZADORES

Los procesos pueden ser calendarizados según distintas políticas. las que podemos entender como dominios de calendarización múltiple. PM por consiguiente le da el control del proceso al administrador de calendarizadores y este se encarga de asignarle el calendarizador adecuado según un determinado proceso.

3) PLANIFICACIÓN DEL PROYECTO

Se ha revisado la implementación de:

http://giteit.udp.cl/so-2016-1/Minix/blob/master/minix/servers/sched/

y con más énfasis el archivo "schedule.c" dentro de esta misma dirección para entender el funcionamiento del calendarizador.

Note la importancia de la estructura "schedproc" implementada en:

http://giteit.udp.cl/so-2016-1/Minix/blob/master/minix/servers/sched/schedproc.h

junto a la cual se define una tabla de estructuras "schedproc" que contiene información asociada a cada proceso.

```
EXTERN struct schedproc {
        endpoint_t endpoint;    /* process endpoint id */
        endpoint_t parent;      /* parent endpoint id */
        unsigned flags;         /* flag bits */

        /* User space scheduling */
        unsigned max_priority;  /* this process' highest allowed priority */
        unsigned priority;              /* the process' current priority */
        unsigned time_slice;            /* this process's time slice */
        unsigned cpu;           /* what CPU is the process running on */
        bitchunk_t cpu_mask[BITMAP_CHUNKS(CONFIG_MAX_CPUS)]; /* what CPUs is the
                                                                process allowed
                                                                to run on */
} schedproc[NR_PROCS];
```

Dentro de esta estructura se definen variables importantes como el id del proceso actual y el id del proceso padre. Estas nos ayudarán a diferenciar los procesos hijos para poder ser delegados a el calendarizador que se guía por la política FCFS.

Dentro del archivo "schedule.c" se define una función importante que es "do_start_scheduling" que es la que maneja las solicitudes para calendarizar un proceso según alguna política.

Otras funciones importantes implementadas en "schedule.c" son "schedule_process" y "sys_schedule".

Además cabe destacar el uso de la estructura "message" en la mayoría de las funciones, ya que los procesos se valen de esta.

Como equipo de trabajo hemos hecho este análisis para tener una base correcta y poder generar de forma adecuada la implementación que requiere este tercer proyecto.

Dentro de la siguiente semana se evaluarán ideas sobre que estructuras de datos serán las más convenientes para la programación del proyecto.