

DISEÑO DETALLADO SISTEMA DE TAQUILLA VIRTUAL

BORJA GONZÁLEZ ENRÍQUEZ
FRANCISCO GARCÍA VÁZQUEZ
JUAN FERNÁNDEZ OTERO



MODIFICACIONES

MODIFICACIONES

- Reducción en el número de colas : los procesos de gradas y eventos pasan a formar parte de la misma cola.
- Por tanto , el número de colas se reduce de 5 a 4.
- El cuanto para las colas que no son la de *pagos* , es el cuanto por defecto de Linux .

DISEÑO DETALLADO

COLA DE PAGOS

- Es la más prioritaria .
- Se planifica con un algoritmo Round Robin , con cuanto bastante alto de manera que se asemeje a un FIFO .
- Se utiliza una prioridad de tiempo real (rt_priority) , mayor que el de las otras colas .



COLAS DE ANULACIONES

- Es la segunda cola más prioritaria.
- Se utiliza también de un algoritmo Round Robin , aun que en este caso se utiliza el cuanto por defecto de Linux .
- Tiene un valor de prioridad de tiempo real (`rt_priority`) ,más alto que el resto de las colas pero más bajo que el de la cola de pagos.



COLAS DE PRE-RESERVAS

- Es la tercera cola más prioritaria .
- Se utiliza también de un algoritmo Round Robin , aun que en este caso se utiliza el cuanto por defecto de Linux .
- Tiene un valor de prioridad de tiempo real (`rt_priority`) ,más alto que gradas y eventos pero mas baja que pagos y anulaciones.

COLAS DE GRADAS Y EVENTOS

- Es la cola menos prioritaria .
- Se utiliza también de un algoritmo Round Robin , aun que en este caso se utiliza el cuanto por defecto de Linux .
- Tiene un valor de prioridad de tiempo real (`rt_priority`) , es el más bajo de todas la colas .



MANUAL DE REFERENCIA

MANUAL DE REFERENCIA

- Modificamos la función `do_exit()` que se encuentra en el archivo `exit.c` de Linux para llevar una cuenta de los procesos que hay en nuestro sistemas .
- Round-Robin: utilizaremos la constante `SCHED_RR` implementada en Linux , con el valor 2.
- Cada proceso de diferentes colas es diferenciado por el campo `comm` del `task_struct`.

MANUAL DE REFERENCIA

- **Vamos a utilizar las siguientes funciones de Linux :**
 - `fork()` : Para crear par nuevos procesos
 - `schedule()`: Para escoger el siguiente proceso a ejecutar
 - `sched_setscheduler()` : Asocia los procesos de nuestro sistema con el algoritmo Round Robin.
 - `Schedduler_tick()`: Se ejecuta cuando el proceso agota su cuanto y en ella se comprueba si se ha terminado los cuantos máximo asignados.

MANUAL DE REFERENCIA

- Implementamos las siguiente funciones :
 - Crear_colas() : Definimos la prioridad de cada proceso.
 - Validar_requisitos: Comprobamos si un nuevo proceso puede entrar en el sistema . También su correspondiente liberación de espacio si fuese necesaria.

PLAN DE PRUEBAS


PLAN DE PRUEBAS

Trabajamos sobre estas suposiciones :

- Puede haber como máximo 10 procesos
- El cuanto de la cola de pagos es 8
- Los procesos de pre-reservas , anulaciones ,gradas y eventos tienen el cuanto definido por Linux .
- El numero máximo de cuantos que se le pueden asignar a un proceso de pagos son 3
- El numero máximo de cuantos que se le pueden asignar al resto de procesos es n.



PLAN DE PRUEBAS

1. No hay ningún: proceso pendiente y llega un proceso eventos .Resultado : Se atiende al proceso de eventos
 2. El sistema está trabajando con un proceso de gradas y llega un proceso de pagos .Resultado : Se pone en espera el proceso de gradas y se atiende el de pagos
 3. El sistema esta atendiendo un proceso de pagos y llega otro proceso de pagos .Resultado :se sigue atendiendo al proceso de pagos y se pone al final de la cola al nuevo proceso de pagos
- 

PLAN DE PRUEBAS

4. El sistema esta trabajando con un proceso de anulaciones y llega un proceso de eventos. **Resultado : Se sigue trabajando con el proceso de anulaciones y se pone en espera el nuevo proceso al final de la cola de eventos.**
5. Se esta atendiendo un proceso de pre-reservas ,transcurre un tiempo de nC , y hay mas procesos en espera. **Resultado : Se elimina ese proceso y se atiende al siguiente más prioritario .**
6. El sistema lleva atendiendo a un proceso pagos 18 segundos(2 cuantos de 8) y el sistema hay mas procesos en espera de tipo gradas . **Resultado: Se sigue trabajando con el proceso de pagos hasta los 24 segundos (3 cuantos de 8)**

PLAN DE PRUEBAS

7. En el sistema hay ya 10 procesos y llega un proceso de tipo Gradas. **Resultado : se descarta el nuevo proceso .**
 8. Un proceso de tipo pagos termina y entra un proceso de tipo pre-reservas : **Resultado : el proceso pagos se elimina y se atiende al proceso pre-reservas**
 9. En el sistema hay 4 procesos de pagos, 2 pre-reservas ,2 eventos y 2 de gradas y llega un nuevo proceso de pagos . **Resultado: se descarta el último de la cola de gradas-eventos y se introduce al final de la cola de pagos**
- 