CE 4303

Instituto Tecnológico de Costa Rica

Ingeniería en Computadores

Principios de Sistemas Operativos

Alejandra Bolaños

Creación y comparación de algoritmos calendarizadores

Proyecto Programado 1

# Motivación y objetivos

El alto conocimiento en tecnologías computacionales, y el amplio uso de lenguajes de programación no son suficientes para construir un conocimiento robusto sobre el diseño y comportamiento de un sistema operativo. En cambio, la construcción, diseño e implementación de partes de un sistema operativo se proyecta como una forma natural de involucrar al estudiante con el funcionamiento de los sistemas en la vida real. De esta manera el estudiante debe tomar el papel de diseñador del sistema operativo en lugar del papel acostumbrado de usuario del OS.

En este proyecto se centra en los calendarizadores de procesos de un sistema operativo. Un calendarizador es quien toma la decisión de cual proceso ejecutar, y el algoritmo que se usa es llamado el algoritmo de calendarización. Ambos elementos tienen una influencia directa sobre el desempeño del sistema operativo y de los programas que corren sobre él. Por lo que el éxito del sistema operativo depende de una manera importante del calendarizador.

Durante este proyecto el estudiante deberá diseñar y programar sus propios algoritmos de calendarizacion y calendarizador, y justificar sus decisiones de diseño y la racionalización que los llevo a la implementación realizada.

Las buenas prácticas de programación son también parte importante de este proyecto, por lo que una documentación del código es esperada tanto como un documento especificando la solución propuesta.

# Objetivo

Los estudiantes deben desarrollar cuatro tipos de calendarizadores para el sistema operativo PintOS, y proveer una plataforma de prueba y comparación para dichos calendarizadores.

# Tecnologías, herramientas y estándares.

* Se debe utilizar Github como software de versionamiento, todo el codigo de su sitema operativo debe ir en una carpeta llamada PintOS y la documentación en otra carpeta llamada Documentación. Aparte de estas dos carpetas, la carpeta principal debe contar con el documento de descripción provisto por la profesora en la carpeta de dropbox Progra/Progra1.tmp
* La metodología de trabajo debe ser Agile-Scrum, por lo que en cualquier momento del proyecto y en clase se les puede pedir a los estudiantes mostrar su backlog. Las tareas también deben ser documentadas como se especifica en el punto de documentación.
* Se debe basar el proyecto en el sistema operativo PintOS.
* El estándar de código a utilizar es el publicado por Richard Stallman en “The GNU coding standards” en Julio 25 (secciones 5 y 6).

# Requerimientos Funcionales

El proyecto consiste en crear cuatro calendarizadores distintos para el sistema operativo PintOS, también debe proveerse una plataforma de pruebas y por último debe asegurarse de que el sistema operativo pueda cargarse desde una memoria USB con la habilidad de seleccionar cuál de los cuatro calendarizadores desea utilizar el usuario.

## Calendaizadores

Implementar cuatro tipos de calendarizadores en Pintos.

El estudiante deberá investigar los archivos que debe modificar para crear un calendarizador en este sistema operativo, y programar e implementar los siguientes cuatro tipos de calendarizador vistos en clase: FCFS, SJF, RoundRobin y colas multinivel.

Para el algoritmo de Round Robin, el diseñador debe elegir el tamaño del quantum y presentar la razón de su diseño en la documentación.

En el algoritmo de colas multinivel se debe basarse en el descrito por la universidad de Stanford para el sistema operativo Pintos: https://web.stanford.edu/class/cs140/projects/pintos/pintos\_7.html#SEC131

Para los cuatro algoritmos, el programador puede incluir funcionalidades y nuevos espacios en el TCB para poder obtener los datos que se piden para las plataformas de prueba.

Incluya una variable nueva que le indique al calendarizador el tipo de algoritmo de calendarización que debe usar. (Hint: Investigue la opción –o mlfqs)

## Plataformas de prueba

### Set de Hilos

El estudiante debe proveer un programa donde puedan seleccionarse los tipos y cantidad de hilos a correr en el procesador de la siguiente manera:

Banderas:

* -t: cantidad de hilos.
* -b: 0 inidca I/O bounded, 1 indica CPU bounded.
* -l: archive donde se escribiran los logs de esta corrida.
* -p: no puede usarse con –b, indica el porcentaje de hilos I/O bounded del grupo de hilos.

./puebahilos –t 19 –b 0 –l logname.log

Por lo que la cantidad de hilos son 19, todos I/O bounded y se escribirán en el archivo logname.log

./puebahilos –t 22 –p 13 –l logname.log

Por lo que la cantidad de hilos son 22, 13% serán I/O bounded y los otros CPU bounded y se escribirán en el archivo logname.log

El programa ejecutará los hilos en el procesador y escribirá los datos necesarios en el archivo de log. Los datos son:

* Procesos en la pila al inicio de la ejecución del primer hilo.
* ~~Pid de cada hilo involucrado.~~
* ~~Tiempo promedio de espera de todos los hilos.~~
* ~~Tiempo promedio de espera de cada hilo.~~
* Cantidad de veces que cada hilo estuvo en la cola.
* ~~Los números de prioridades que tuvo en el caso del algoritmo de cola multinivel.~~

La cantidad máxima de hilos a solicitar será de 25. Y la elección de las operaciones a correr para cada tipo de hilo deben ser justificadas en la documentación en prosa.

### Programa de comparación:

El estudiante deberá proveer un programa que compare al menos cuatro logs (cuatro archivos de log de salida de la sección de pruebas de hilos). Donde los siguientes datos deben mostrarse:

-Comparación entre los tiempos de espera promedio de cada corrida.

-Algoritmo más rápido.

-Tipos de hilos involucrados (CPU Bounded, I/O bounded, porcentaje de cada uno).

Todos los datos deben ser mostrados de manera comprensible y uniforme. Se calificará la claridad con la que se muestren.

## Bootloader

El sistema operativo debe ser cargado desde una memoria USB utilizando el bootloader del computador y al mismo tiempo poder elegir el tipo de calendarizador a utilizar. Por lo que el bootloader del computador debe ser modificado para proveer esta funcionalidad. El estudiante puede elegir el bootloader a utilizar y modificar con tal fin.

# Documentación

## Documentación del Scrum:

Un documento describiendo el avance del backlog por lo menos semanalmente debe estar en la carpeta de Documentos/ del Github y podrá ser revisado en cualquier momento a partir del 31 de Marzo del presente año.

## Documento de diseño en Github:

El código debe incluir el documento de diseño cuyo template se encuentra en Dropbox-Progra1-Progra1.tmp. El campo de calendarizador debe llenarse para cada uno de los calendarizadores programados en cada una de las implementaciones de los algoritmos de calendarización.

## Documentación del código:

En la sección de Tecnologías, herramientas y estándares se especifica el estándar a seguir para documentar el código.

## Documentación en prosa:

El documento debe estar en formato IEEE transactions, debe contar con las secciones de Abstract, Introducción, Descripción del problema, Detalles de la implementación, Justificación, Comparación entre los algoritmos, Conclusiones.  
No debe extenderse a más de 3 páginas, por lo que el programa de comparación y el bootloader no deben incluirse.

Este documento se ubicará en la carpeta Documentación/ del proyecto en Github.

# Entregables y fechas

## 1er entregable: Viernes 31 de marzo.

## Crear un proyecto de Github y dentro del mismo crear dos carpetas: PintOS y Documentación.

* Subir el documento que se encuentra en Progra/Progra1.tmp a la parpeta principal del proyecto.

## Subir el código original de la versión de PintOS a utilizar en una carpeta llamada PintOS a la carpeta del proyecto de Github. Se recomienda <http://www.stanford.edu/class/cs140/projects/pintos/pintos.tar.gz>

## Crear un README dentro de la carpeta PintOS donde explique que hay en cada carpeta (no incluya las carpetas de test ni example ) así como los archivos dentro de /thread.

## 2do entregable: Martes 4 de abril

* Documento con las tareas realizadas y autores en la bitácora del backlog.

## Entrega Final: Jueves 20 de abril, sábado 22 de abril.

* Proyecto completo y documentación al día de la entrega.

# Evaluación

Documentación: 20% (toda o nada)

Interna 5%

Justificación 5%

Externa-Scrum 10%

Bootloader modificado: 8%

Algoritmos: 40%

Set de Hilos: 20%

Avance 1: 5%

Avance 2: 2%

Análisis de los datos: 5%

Total: 100%

# Aspectos administrativos

El proyecto se debe realizar en parejas y su valor sobre la nota del curso es de 20%.

## Documentación:

Si toda la documentación no es presentada (tanto de código como el documento final y los archivos especificados en este documento) se asumirá que la documentación es inexistente ya que incompleta no representa ningún aporte a los programadores, por lo que la nota de la documentación será 0 del 20%.

## Fecha de entrega:

De no presentarse un estudiante a la revisión, tendrá una nota de 0% en el proyecto, salvo por justificación médica debidamente presentada.

Funcionalidades:

Es responsabilidad del estudiante proveer al evaluador de una plataforma por medio de la cual se pueda comprobar el código escrito y las implementaciones de los requerimientos. De caso contrario la nota para el requerimiento será de cero.

## Originalidad:

Se considerará plagio el uso de cualquier código no programado por los estudiantes quienes entregan el proyecto.