Unidad Didáctica 1. Arranque y procesos del sistema

Administración de Sistemas Operativos - 1ª Evaluación (RA 2 – CE a, b, c)

Alvaro Vazquez Vazquez

September 28, 2025

I.E.S. Fernando Aguilar Quignon
C/Conil de la Frontera, 3
CP 11010, Cádiz

Abstract

Este documento recoge las respuestas a las preguntas de la Unidad Didáctica 1 sobre arranque y procesos del sistema. Las respuestas han sido elaboradas utilizando el lenguaje técnico aprendido en clase y organizadas de manera didáctica para facilitar su estudio y comprensión.

Pregunta 1. Explica qué es un programa y un proceso, realizando una comparación de ambos.

Respuesta: Un **programa** es un conjunto de instrucciones almacenadas en un archivo ejecutable, mientras que un **proceso** es una instancia de ese programa en ejecución. El sistema operativo carga el programa en memoria y crea un proceso que incluye el código, datos y estado de ejecución.

Pregunta 2. ¿Qué elementos contiene una imagen de proceso? Define cada uno de ellos.

Respuesta: La imagen de proceso contiene:

a) **Bloque de control de proceso (BCP):** Estructura de datos que mantiene información sobre el proceso (punteros, estado, número de proceso, etc.)

b) **Instrucciones del programa:** Código máquina del programa en ejecución

c) **Datos estáticos:** Constantes y variables inicializadas

d) **Pilas de usuario y kernel:** Para llamadas a funciones

e) **Heap o montículo:** Memoria para datos creados dinámicamente

Pregunta 3. ¿Qué estructura utiliza el sistema operativo para gestionar, de forma global, los procesos que están corriendo?

Respuesta: El sistema operativo utiliza la **tabla de procesos del sistema**, que contiene punteros a los BCP de todos los procesos activos, permitiendo localizar rápidamente la información de cualquier proceso.

Pregunta 4. ¿Qué elementos componen el BCP?

Respuesta: El Bloque de Control de Proceso contiene:

- Puntero al siguiente proceso
- Estado del proceso
- Número de proceso (PID)
- Contador de programa
- Registros de la CPU
- Espacio de direcciones
- Lista de archivos abiertos
- Datos de contabilidad y prioridad

Pregunta 5. ¿Para qué usa el sistema el CP?

Respuesta: El **Contador de Programa (CP)** almacena la dirección de memoria de la siguiente instrucción a ejecutar por la CPU, permitiendo que el procesador sepa qué instrucción ejecutar a continuación.

Pregunta 6. ¿Por qué es útil almacenar el estado de los registros de la CPU?

Respuesta: Es esencial para los **cambios de contexto**. Cuando un proceso con mayor prioridad entra en la CPU, se debe almacenar el estado del proceso anterior para que, cuando se reanude, la CPU pueda continuar exactamente donde lo dejó.

Pregunta 7. ¿Qué implica que un proceso corra en modo kernel? ¿Qué ocurrirá si se ocasiona un error durante dicha ejecución?

Respuesta: Un proceso en **modo kernel** tiene acceso sin restricciones al hardware y comparte espacio de memoria con el sistema operativo. Si ocurre un error, podría escribir en memoria crítica, causando un **kernel panic** que bloquea todo el sistema.

Pregunta 8. Relacionado con lo anterior, ¿cuál es el mensaje típico que muestra el sistema?

Respuesta: El sistema muestra un mensaje de **"Kernel panic"** seguido de información técnica sobre el error. Los drivers son procesos comunes que ejecutan en modo kernel y su fallo puede causar este problema.

Pregunta 9. ¿Para qué sirven las llamadas a sistema?

Respuesta: Las **llamadas a sistema** permiten que procesos en modo usuario soliciten servicios del sistema operativo de forma segura, ya que no tienen acceso directo al hardware. El SO realiza la operación requerida en modo kernel.

Pregunta 10. En relación con el espacio de direcciones virtuales que puede utilizar un proceso, ¿qué diferencia existe entre el modo usuario y el modo kernel?

Respuesta: Los procesos en **modo kernel** comparten un espacio de direcciones con el SO, mientras que los procesos en **modo usuario** tienen espacios de direcciones privados y aislados, garantizando seguridad y estabilidad.

Pregunta 11. ¿Qué implicaciones tiene el hecho de que un proceso se esté ejecutando en primer plano? ¿Y en segundo?

Respuesta:

- **Primer plano:** El proceso ocupa la terminal y requiere interacción del usuario. Debemos esperar a que finalice para usar la terminal nuevamente.
- **Segundo plano:** El proceso ejecuta sin bloquear la terminal, permitiendo seguir trabajando.

Ejemplo de ejecución en segundo plano:

```
firefox &
```

Pregunta 12. ¿Qué términos anglosajones utilizamos para referirnos a los procesos que corren en primer plano? ¿Y en segundo?

Respuesta:

- **Primer plano:** *Foreground processes*
- **Segundo plano:** *Background processes*

Pregunta 13. ¿Qué principal diferencia existe entre un trabajo y un daemon?

Respuesta: Un **trabajo** está asociado a una terminal y requiere interacción del usuario, mientras que un **daemon** es un proceso desvinculado de cualquier terminal que ejecuta en segundo plano de forma permanente.

Pregunta 14. ¿En qué plano corre un daemon? ¿Y un trabajo?

Respuesta: Los **daemons** ejecutan exclusivamente en segundo plano, mientras que los **trabajos** pueden ejecutar en primer o segundo plano según cómo sean iniciados.

Pregunta 15. ¿Cómo pasamos un proceso que está corriendo en primer plano a segundo plano?

Respuesta: Se utiliza `Ctrl+Z` para detener el proceso y liberar la terminal, luego `bg` para reanudarlo en segundo plano:

```
# Detener proceso con Ctrl+Z
# Ver trabajos
jobs
# Reanudar en segundo plano (ejemplo para trabajo
   n mero 1)
bg %1
```

Nota: `bg` reanuda en segundo plano, `fg` en primer plano.

Pregunta 16. ¿Cómo se llama la señal que hemos utilizado para llevar a cabo el proceso anterior?

Respuesta: La señal **SIGSTOP** que envía el proceso a segundo plano y lo detiene. `Ctrl+Z` equivale a ejecutar `kill -SIGSTOP [PID]`.

Pregunta 17. Cuando examinamos el estado de los procesos vía terminal, ¿qué símbolo tienen los daemons en el apartado de terminal (TT)?

Respuesta: Los daemons muestran **?** en el campo TTY, indicando que no tienen terminal asociada.

Pregunta 18. ¿Qué eventos principales provocan la creación de procesos?

Respuesta: Los eventos principales son:

- Inicio del sistema
- Ejecución de un programa por usuario
- Llamada al sistema `fork()` desde proceso existente
- Solicitud de servicio por proceso existente

Pregunta 19. ¿Qué ocurre cuando el algoritmo de planificación asociado a una CPU es de tipo apropiativo? ¿Y no apropiativo?

Respuesta:

- **Apropiativo:** Permite que procesos de mayor prioridad interrumpan (cambio de contexto) procesos en ejecución.
- **No apropiativo:** Un proceso mantiene la CPU hasta que finaliza voluntariamente, pudiendo formarse "convoyes" de procesos.

Pregunta 20. Explica los estados relacionados con el modelo de 5 estados.

Respuesta: Los estados del modelo de 5 estados son:

- **Nuevo:** Proceso recién creado, esperando admisión
- **Listo:** Admitido, esperando asignación de CPU
- **Ejecución:** Ejecutándose en la CPU
- **Bloqueado:** Esperando evento (E/S), libera CPU
- **Terminado:** Finalizado pero aún en memoria

Pregunta 21. Dentro del ámbito de la gestión de procesos, ¿qué planificadores existen? ¿Cuál es su cometido?

Respuesta: Existen tres tipos de planificadores:

- **Corto plazo:** Asigna CPU a procesos listos
- **Medio plazo:** Gestiona swapping procesos entre memoria y disco

- **Largo plazo:** Admite nuevos procesos al sistema

Pregunta 22. ¿Qué relación existe entre la memoria virtual y la gestión de procesos?

Respuesta: La memoria virtual permite que cada proceso tenga su propio espacio de direcciones, facilitando la gestión y protección entre procesos. El SO asigna memoria física según necesidad mediante paginación.

Pregunta 23. ¿Qué estados se añaden en el modelo de 7 estados?

Respuesta: Se añaden **Suspendido listo** y **Suspendido bloqueado**, para procesos que han sido movidos a disco (swapping) por falta de memoria.

Pregunta 24. Bajo qué circunstancias y desde qué estados se alcanzan los nuevos estados?

Respuesta: Los estados suspendidos se alcanzan cuando el SO necesita liberar memoria:

- De **Listo** a **Suspendido listo**
- De **Bloqueado** a **Suspendido bloqueado**

Pregunta 25. ¿Qué dos señales del sistema están relacionadas con los anteriores estados?

Respuesta: Las señales **SIGSTOP** (suspender) y **SIGCONT** (reanudar) gestionan la transición entre estados activos y suspendidos.

Pregunta 26. Dentro de los sistemas GNU/Linux, ¿qué simbolizan los estados S, D, R y T?

Respuesta:

- **S (Sleeping):** Proceso esperando evento
- **D (Disk sleep):** Bloqueado esperando E/S de disco
- **R (Running):** Ejecutándose o listo para ejecutar
- **T (Stopped):** Detenido por señal

Pregunta 27. En el caso del formato BSD, ¿qué significan los modificadores <, N, s, l y +?

Respuesta:

- **<:** Proceso de alta prioridad
- **N:** Proceso de baja prioridad
- **s:** Líder de sesión
- **+:** Proceso en primer plano

- **l:** Usa páginas de memoria bloqueadas en RAM

Pregunta 28. Define trabajo, tarea y proceso.

Respuesta:

- **Proceso:** Instancia de programa en ejecución
- **Trabajo:** Proceso iniciado desde terminal
- **Tarea:** Subproceso o hilo dentro de un trabajo

Pregunta 29. ¿Qué relación existe entre una tarea y un trabajo?

Respuesta: Un trabajo se compone de múltiples tareas que se ejecutan secuencial o concurrentemente para lograr un objetivo común.

Pregunta 30. ¿Una tarea puede ser un trabajo? ¿Y viceversa?

Respuesta: Una tarea no puede ser un trabajo (es una parte), pero un trabajo puede considerarse una tarea dentro de un contexto más amplio.

Pregunta 31. ¿Por qué decimos que una tarea se puede considerar un subproceso?

Respuesta: Porque un trabajo (proceso) puede dividirse en múltiples tareas (subprocesos) que ejecutan concurrentemente.

Pregunta 32. ¿Qué diferencia existe entre un trabajo y un proceso?

Respuesta: Todo trabajo es un proceso, pero no viceversa. Los trabajos están vinculados a terminales, mientras que otros procesos (daemons) pueden ejecutar independientemente.

Pregunta 33. ¿Cuándo consideramos que un proceso es un trabajo?

Respuesta: Cuando es iniciado desde y asociado a una terminal específica.

Pregunta 34. En relación con las terminales, ¿qué ámbito o contexto tienen los trabajos?

Respuesta: Los trabajos tienen ámbito local a la terminal donde fueron iniciados. Al cerrar la terminal, normalmente se envían señales de terminación a sus trabajos.

Pregunta 35. ¿De qué dos maneras, vía terminal, podemos revisar los trabajos?

Respuesta:

- `jobs` - muestra trabajos de la terminal actual
- `ps aux | grep $USER` - muestra todos los procesos del usuario

Pregunta 36. ¿Qué es un hilo de ejecución?

Respuesta: Un **hilo** es la unidad mínima de ejecución que puede ser planificada. Comparte espacio de memoria con otros hilos del mismo proceso pero tiene su propia pila y registros.

Pregunta 37. ¿Qué concepto está relacionado con dichos hilos?

Respuesta: El concepto de **programación multihilo** o **multithreading**, que permite ejecutar múltiples hilos concurrentemente dentro del mismo proceso.

Pregunta 38. ¿Qué problema se deriva de la no existencia de programación multihilo?

Respuesta: Ineficiencia en aplicaciones que requieren concurrencia, ya que se necesitarían múltiples procesos (más pesados) en lugar de hilos (más ligeros).

Pregunta 39. ¿Qué ocurre con la imagen del proceso cuando tenemos varios hilos corriendo del mismo proceso?

Respuesta: Los hilos comparten la misma imagen de proceso (código, datos, archivos) pero cada hilo tiene su propia pila y contexto de ejecución.

Pregunta 40. ¿Por qué los hilos aumentan la eficiencia de la comunicación entre programas que están en ejecución?

Respuesta: Porque al compartir memoria, la comunicación entre hilos es más rápida que entre procesos (que requieren mecanismos IPC más lentos).

Pregunta 41. ¿Por qué es más liviano el cambio de contexto entre dos hilos de un mismo proceso?

Respuesta: Porque solo se necesita cambiar pila y registros, no el espacio completo de direcciones (que es compartido).

Pregunta 42. Define el concepto de interrupción.

Respuesta: Señal asíncrona que alerta a la CPU sobre eventos externos que requieren atención inmediata.

Pregunta 43. Define el concepto de excepción.

Respuesta: Evento síncrono causado por la ejecución de instrucciones (división por cero, acceso inválido a memoria).

Pregunta 44. ¿Cuándo se da una interrupción por software? ¿En qué consisten?

Respuesta: Son generadas por instrucciones específicas (como `int` en x86) para solicitar servicios del sistema operativo (llamadas al sistema).

Pregunta 45. ¿Cuándo se da una interrupción por hardware? ¿En qué consisten?

Respuesta: Generadas por dispositivos hardware (teclado, disco, red) para indicar que necesitan atención del procesador.

Pregunta 46. ¿Cuál fue la primera técnica que se implementó para las interrupciones?

Respuesta: El **polling** o sondeo, donde la CPU verificaba periódicamente el estado de los dispositivos.

Pregunta 47. ¿Cómo se aplicaba la técnica anterior?

Respuesta: La CPU gastaba ciclos revisando registros de estado de dispositivos, resultando ineficiente para sistemas con múltiples dispositivos.

Pregunta 48. ¿Qué ocurre con los modos de ejecución cuando se lanza una excepción por software?

Respuesta: Se produce un cambio de **modo usuario** a **modo kernel** para que el sistema operativo maneje la excepción de forma segura.

Pregunta 49. ¿Por qué decimos que las excepciones son un mecanismo de protección que permite garantizar la integridad de los datos almacenados tanto en el espacio del usuario como en el espacio del kernel?

Respuesta: Porque detectan operaciones inválidas y permiten al SO intervenir antes de que causen daño, manteniendo el sistema estable y seguro.

Pregunta 50. ¿Bajo qué circunstancias se produce una excepción?

Respuesta: Cuando ocurren condiciones excepcionales durante la ejecución: errores aritméticos, accesos inválidos a memoria, instrucciones privilegiadas en modo usuario, etc.