SISTEMAS INFORMÁTICOS, PRÁCTICA 1

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación.
Departamento de Ingeniería Informática.
Escuela Politécnica Superior.
Universidad Autónoma de Madrid.

INTRODUCCIÓN A GNU/LINUX

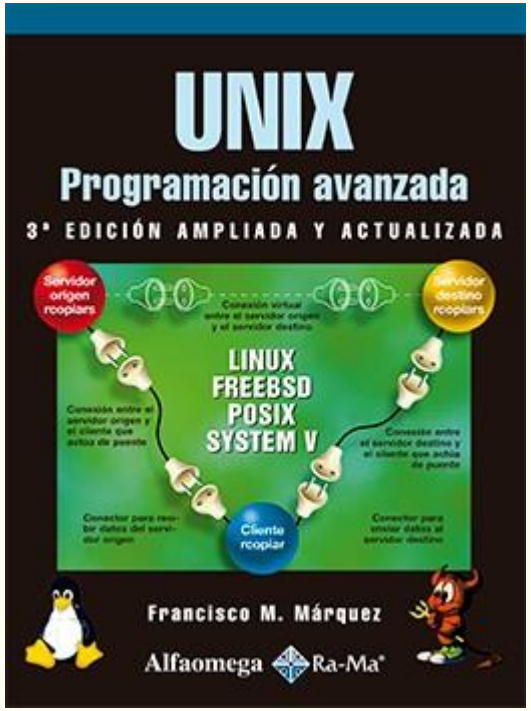
Práctica 1

Bibliografía para las prácticas centradas en Sistemas Operativos

Escuela Politécnica Superior

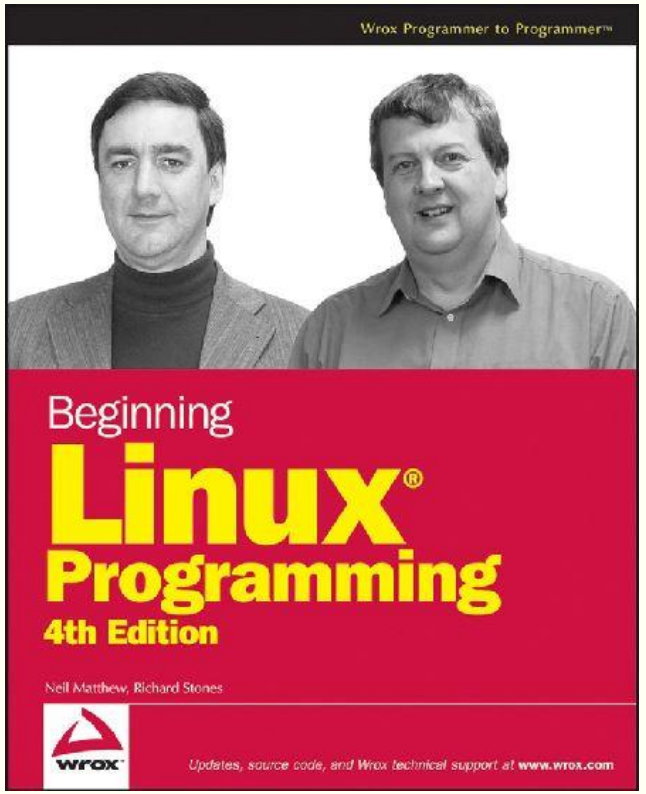
Unix: programación avanzada

Márquez



Beginning Linux Programming

Matthew & Stones



Enlace al vídeo para verlo fuera del aula

GNU : «GNU No es Unix»

- GNU es un sistema operativo (SO) de tipo Unix, lo cual significa que se trata de una colección de muchos programas: aplicaciones, bibliotecas, herramientas de desarrollo, etc.
- Un SO GNU es un sistema operativo de software libre, es decir, respeta la libertad de los usuarios. El software libre permite que los usuarios ejerzan el control de sus propias tareas de computación.
- **Software libre significa que los usuarios tienen la libertad de ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software.**

El software libre es una cuestión de libertad, no de precio. Para entender el concepto, debe pensarse en «libre» como en «libertad de expresión», no como en «cerveza gratis».

Vea SW libre y educación.

Código abierto (*open source*)

El código abierto (*open source*) es una metodología de programación, el software libre es un movimiento social.

- Software libre: imperativo ético, respeto esencial por la libertad de los usuarios.
- Código abierto: cuestiones en términos de cómo «mejorar» el software, en sentido meramente práctico. Sostiene que el software privativo no es una solución óptima para los problemas prácticos que hay que resolver.

En la mayoría de los casos, cuando se discute sobre «código abierto» no se toma en consideración el bien y el mal sino únicamente la popularidad y el éxito.

Por ejemplo, Ubuntu provee repositorios específicos de software que no es libre, y Canonical promueve y recomienda explícitamente, bajo el nombre de Ubuntu, software que no es libre en algunos de sus canales de distribución.

Enlace al vídeo para verlo fuera del aula

Linux

- **Linux** es un núcleo/*kernel* de libre distribución y mayormente libre semejante al núcleo de Unix. Está desarrollado por colaboradores de todo el mundo.
- El núcleo/kernel es un *software* que constituye una parte fundamental del sistema operativo. Se ejecuta en modo privilegiado (conocido también como modo núcleo). Es el principal responsable de facilitar a los distintos programas acceso seguro al *hardware* de la computadora. También se encarga de decidir qué programa podrá usar un dispositivo de hardware y durante cuánto tiempo.

Enlace al vídeo para verlo fuera del aula

Introducción al intérprete de comandos (shell)
CTRL+ALT+T

- Limpiar el intérprete de comandos:
  $ clear
- Cómo completar un comando: pulsad “tabulador” repetidamente.
- Comandos anteriores: pulsad “flecha arriba” repetidamente.
  Comandos posteriores: pulsad “flecha abajo” repetidamente.
- Buscador de comandos ejecutados anteriormente: pulsad CTRL+R y escribid parte del comando ejecutado en un pasado muy lejano, o no tan lejano. Pulsad CTRL+R repetidamente para seguir viendo comandos del pasado.
- Salir: $ exit

Tarea guiada por el profesor de prácticas

Ayuda, navegación, visualización, gestión de procesos y sistema

- $ man
- $ cd
- $ ls
  - $ ls -la
- $ find
- $ cat
- $ ps
  - $ ps -a
  - $ ps -l
- $ pstree
- $ df
- $ du
- $ free

Trabajo con archivos y expresiones regulares

- Búsqueda de patrones: $ grep
- 2º plano: $ … &
- Tubería/*Pipe*: $ … | …
- Redirección de salida a archivo: $ … > …
- Expresión regular …*…
- Etc.

Máquinas virtuales

Oracle VM VirtualBox

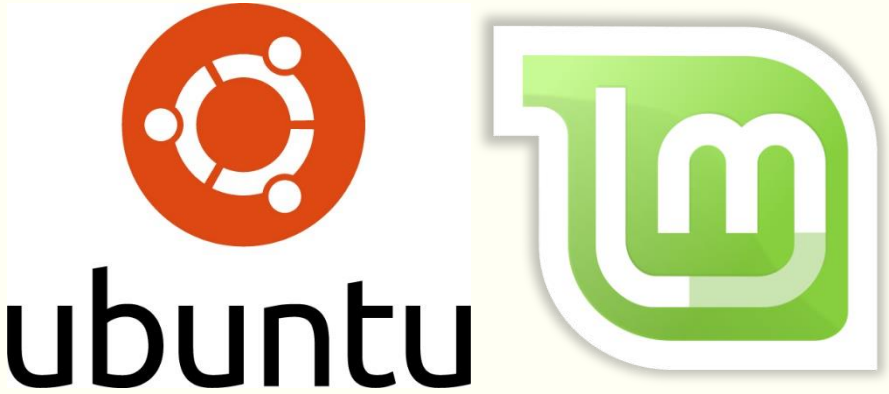
VMware

¿Imagen del SO EPS_UAM o imagen nueva de un SO? ¿O quizá una partición en el disco duro?

Escuela Politécnica Superior

Máquina virtual EPS-UAM

(Ubuntu 14.04 con entorno de escritorio Cinnamon, más propio de Linux Mint)



SO instalado de cero

(Lubuntu = Ubuntu con entorno de escritorio ligero LXDE, descargad Lubuntu 14.04 para que el compilador gcc tenga la misma versión que en los laboratorios)

No todo es SW libre en esta vida, ni únicamente SW

Escuela Politécnica Superior

UAM DotNet Club

Club de Robótica-Mecatrónica

COMANDO MAKE Y ARCHIVOS MAKEFILE

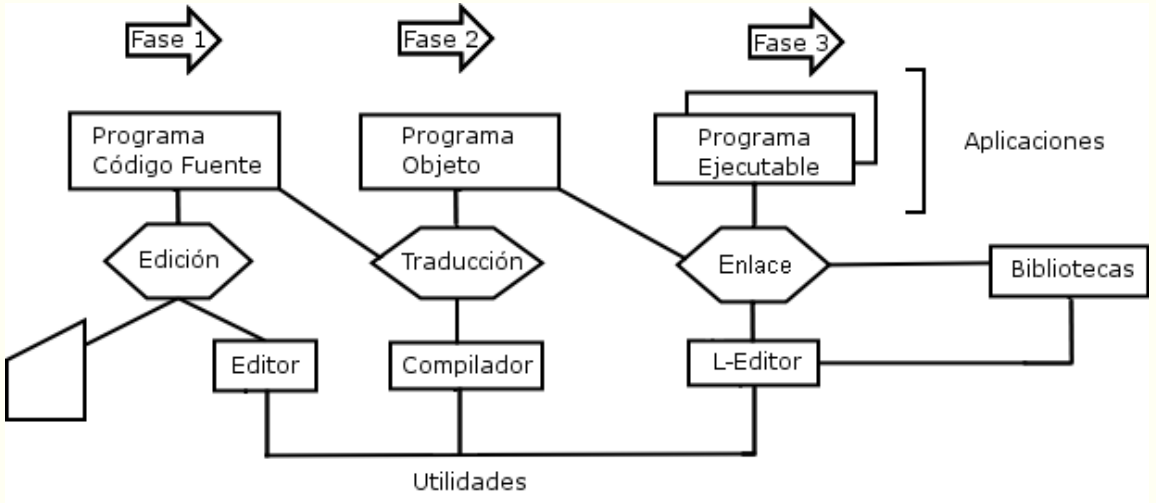
Práctica 1

Make, código objeto y ejecutables

Make es una herramienta de gestión de dependencias que existen entre los archivos que componen el código fuente de un programa para dirigir su recompilación o generación automáticamente.

Código objeto es el que resulta de la compilación del código fuente.

Un **enlazador** (*linker*) es un programa que toma los objetos generados en los primeros pasos del proceso de compilación, la información de todos los recursos necesarios (biblioteca), quita aquellos recursos que no necesita, y enlaza el código objeto con su(s) biblioteca(s) con lo que finalmente produce un fichero ejecutable o una biblioteca.



"Código objeto," Wikipedia, the free encyclopedia

Código objeto

Ventajas:

1. Velocidad de compilación.
2. Compresión de librerías objeto.
3. Compartir librerías y funciones sin liberar código fuente original.
4. Puede permitir a diferentes leguajes de programación compartir funciones sin reescribir el código fuente.

Error común:

1. Un objeto importa funciones de otro que ha sido modificado, llamando con parámetros incorrectos a una función cuyos argumentos han sido cambiados.
2. El compilador no lo detecta ya que el código objeto no es verificado, únicamente enlazado.
3. Solución: reescribir el código de manera correcta y recompilarlo a código objeto.

Tarea

- Fuera del directorio p1 se os proporciona un Makefile de plantilla para futuros proyectos.
- La tarea consiste en conocer cómo funciona dicho Makefile.
  - Macros CC, etc.
  - Flags -c, etc.
  - Macros especiales como $@ $< etc.

REPASO DE PUNTEROS EN C

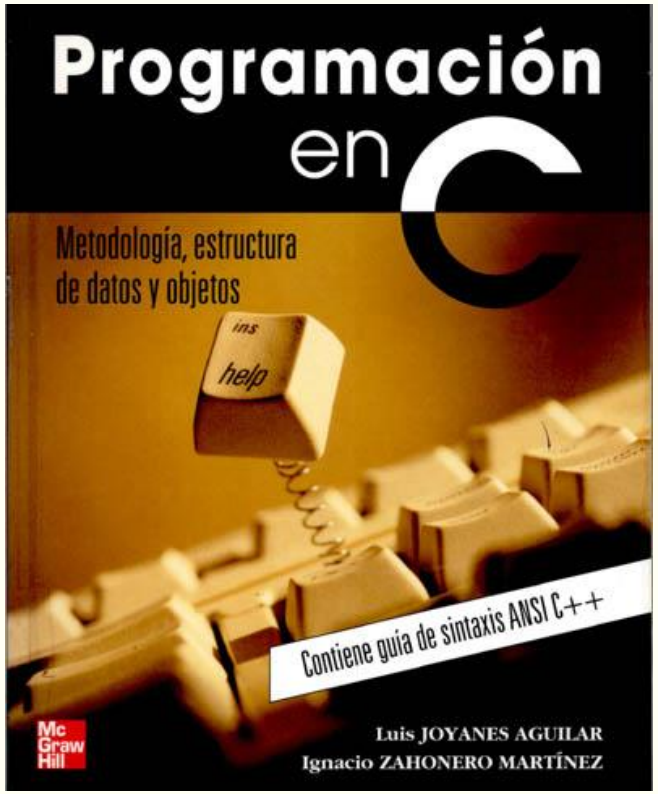
Práctica 1

Bibliografía en castellano de programación en C

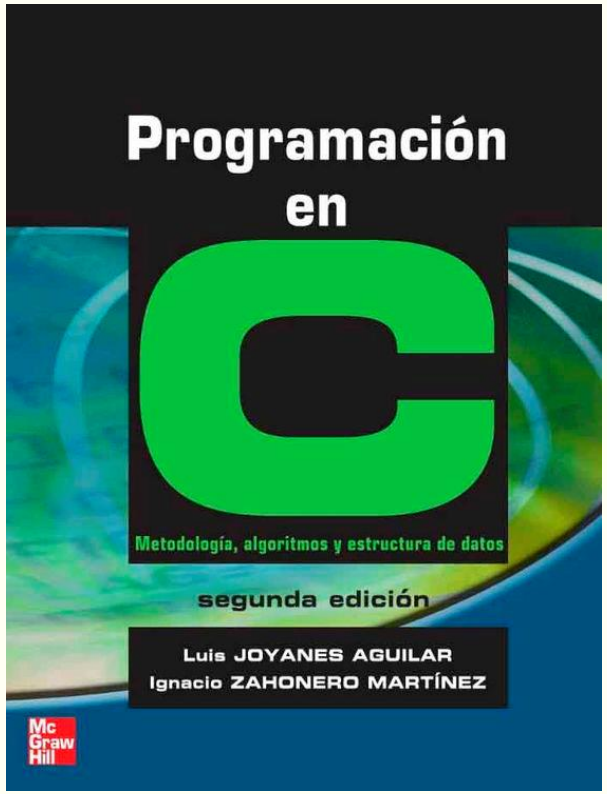
"*Programación en C*". Joyanes & Zahonero.

Escuela Politécnica Superior

Primera edición (igual de completa que la segunda edición)



Segunda edición



Herramientas

- Las siguientes herramientas serán explicadas brevemente en clase por el profesor de prácticas:
  - Doxygen: generación de documentación.
    $ make doxygen
  - Valgrind: usado para detección de problemas liberando memoria reservada.
    $ make valgrind ARGS="./<executable> [<arguments>]"
  - Nemiver: depuración de código en vivo (mejor no ejecutarse en segunda plano, …&)
    $ nemiver ./<executable> [<arguments>]
  - nmon para comprobar en un futuro el uso de cores por distintos procesos
    $ nmon (pulsando ‘c’ a continuación, ‘c’ de nuevo para volver o ‘q’ para terminar)
- Para las prácticas se aconseja Geany como IDE y Terminator como multi-terminal (éste último no instalado en los laboratorios)
- Estas herramientas además del compilador gcc pueden ser instaladas ejecutando
  $ make dev-essential

Tarea

- Se os adjuntan 4 documentos adicionales, leedlos detenidamente:
  - C, lectura de archivos
  - Doxygen, introducción
  - Valgrind, tutorial
  - Valgrind, utilización
- Entra en el directorio p1, comprende el archivo Makefile y genera los códigos ejecutables con el comando make.
- Comprende el código fuente y rellena la documentación Doxygen de los archivos fuente. Pregunta a tu profesor cuestiones concretas.
- Genera la documentación ejecutando $ make Doxygen
- Comprueba el resultado, por ejemplo, pulsando en el archivo index.html de la carpeta html.
- Comprueba con el comando
  $ make valgrind ARGS="./<executable> [<arguments>]"
  la liberación de memoria del la función *ej3* comentando y descomentando *free(ac)*.

SISTEMAS INFORMÁTICOS, PRÁCTICA 1

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación.
Departamento de Ingeniería Informática.
Escuela Politécnica Superior.
Universidad Autónoma de Madrid.