Robótica y automatización

Introducción a los sistemas operativos
Caso de estudio Lunix

Facultad de Ingeniería
Instituto de Computación

Contenido

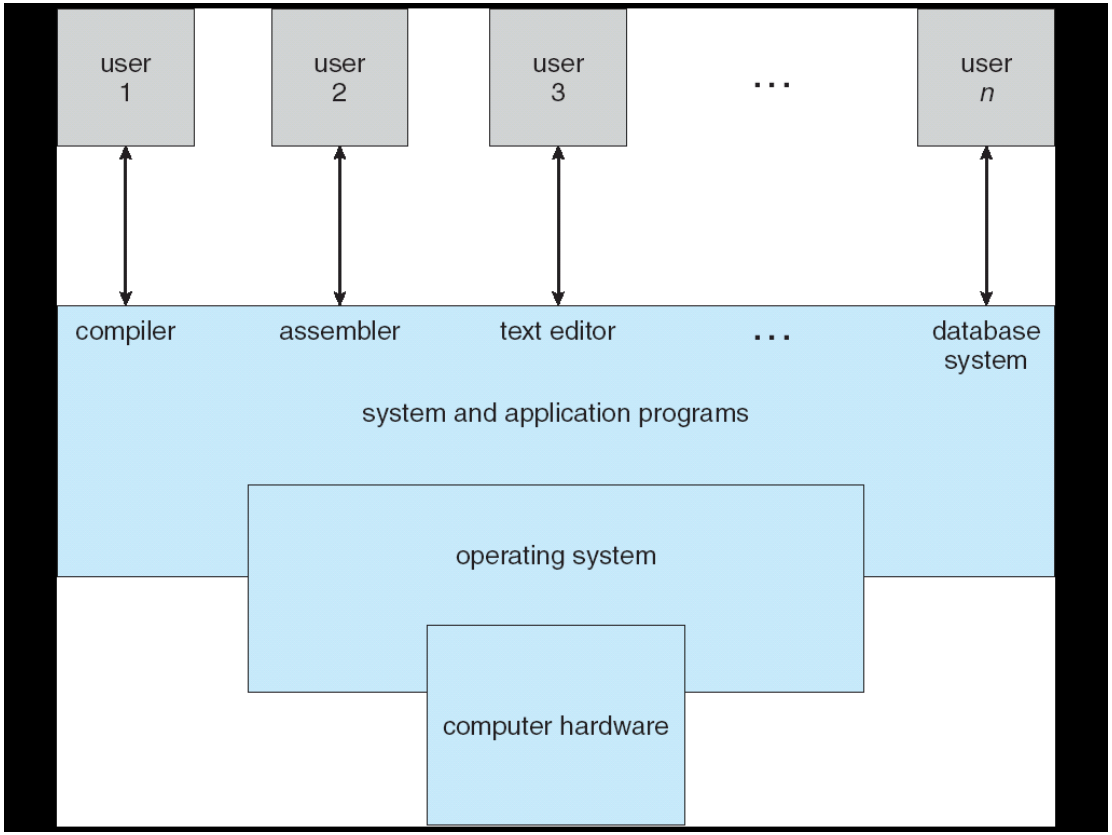
- Introducción a los sistemas operativos.
- Introducción a Linux y Linux embebido.

Definición de sistema operativo

- ¿Qué sistemas operativos conoces?
- ¿Qué es un sistema operativo?
- Programa o conjunto de programas que realizan funciones básicas y permiten el desarrollo de otros programas.

Definición de sistema operativo

- Es un programa
- Funciona como intermediario entre el usuario y los programas y el hardware



Detalles de los SO

- Metas
  - Brindar un entorno para que los usuarios puedan ejecutar programas en forma conveniente
  - Brindar un entorno para que los programas usen el hardware con facilidad
  - Administrar el hardware de forma eficiente y equitativa
  - Proveer un entorno sin interferencias a cada usuario
- Todas las aplicaciones requieren un conjunto de operaciones comunes que son incorporadas al sistema operativo
- Funciones básicas:
  - Administración de procesos
  - Manejo de interrupciones
  - Administración de memoria
  - Manejo del sistema de archivos
  - Administración de seguridad
  - Control de entrada/salida

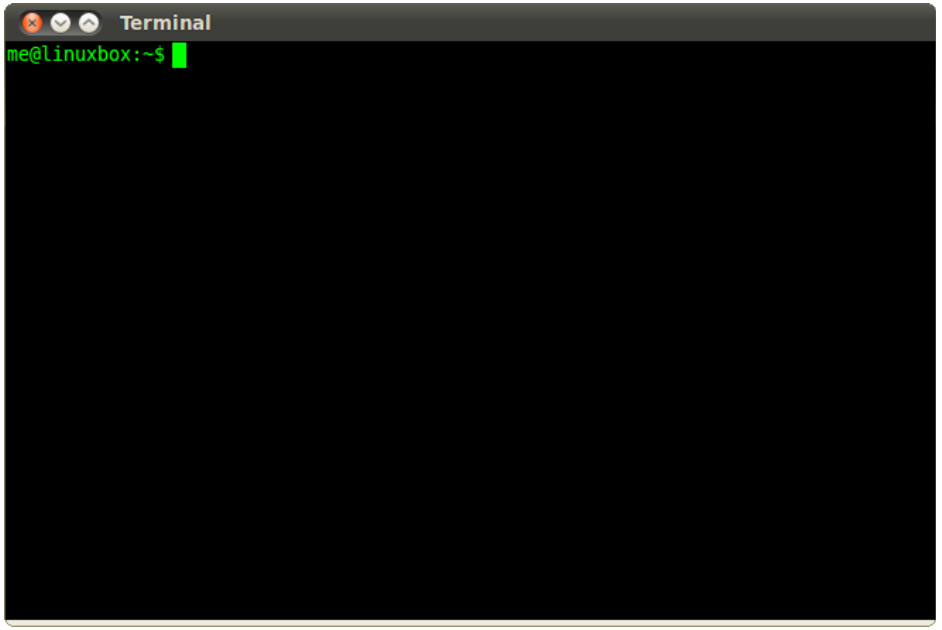
Interfaz de usuario

- La interfaz de usuario controla como se ingresan los datos y las instrucciones, y como se presenta la información en la pantalla.
- Tipos
  - Textuales (Línea de comandos)
  - Basadas en menú
  - Gráficas

Línea de comando

- La línea de comando es una interfaz textual.
- Propone una interacción entre ella computadora y el usuario donde la computadora responde a los comandos ingresados por el usuario.

- Ejemplo:
  - Usuario (me)
  - Máquina (linuxbox)
  - Directorio (~)



Aspectos importantes

- Los sistemas Linux distinguen entre mayúsculas y minúsculas.
- Se tiene al uso de las minúsculas
- Los comandos deben escribirse con exactitud.
- El uso de la tecla tabulador (Tab) minimiza los errores de tipeo.
- La ejecución de un comando puede detenerse digitando Ctrl-C.

Estructura de un comando

nombre opciones argumentos

- opciones, controlan el comportamiento del comando
- argumentos, indican sobre que elemento actúa el comando

- Ejemplo:
  - ls
  - ls -l
  - ls -l /tmp

Primeros comandos

- echo Hola mundo
- echo $LOGNAME
- echo $HOMEDIR
- hostname
- whoami
- who
- id
- id --help
- pwd
- date
- ls -l /tmp
- ps
- ps aux
- ps aux | less
- ps aux | wc -l
- man whoami
- man man
- cat /etc/hostname
- ping
- ifconfig
- cd
- exit

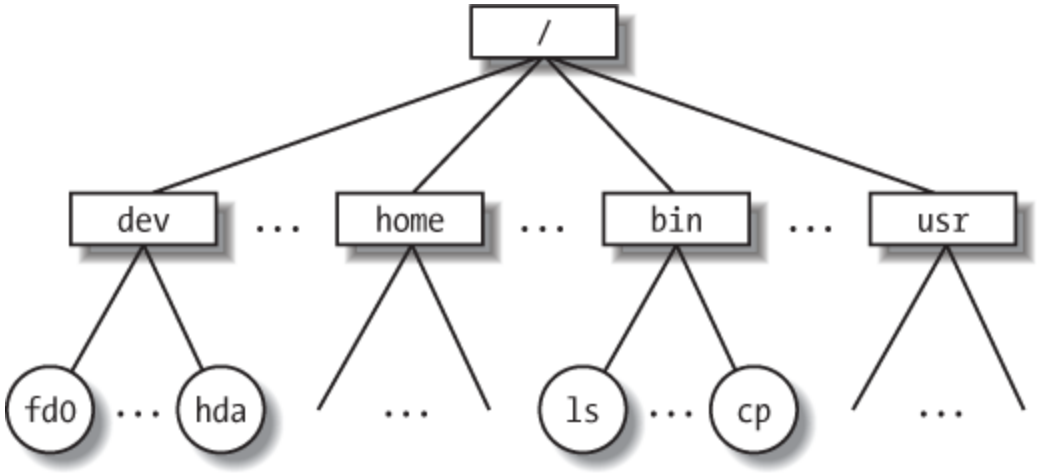
Página man

- El comando man permite acceder a las páginas man.
- Las páginas man son un manual de referencia de los comandos del sistema operativo, bibliotecas, entre otros.
- Ejemplo:

  man ls
- La página man describe la sintaxis de un comando. Esto incluye las distintas formas que toma el comando y sus argumentos.
- Su contenido es técnico y totalmente abarcativo.

Directorios

- Un directorio es un contenedor de archivos y otros directorios.
- Esta organización da lugar a una estructura ordenada denominada árbol de directorios que tiene como raiz (origen) al directorio /.



Comandos

- cd
- pwd
- mkdir

Archivos

- Listado de archivos
  - ls, ls -l, ls -l /bin
  - ls -a, ls -la, ls -la /var
- Contenido de un archivo
  - man man > man.txt
  - cat man.txt
  - less man.txt
  - head man.txt
  - tail -5 man.txt
- Crear, borrar y mover archivos
  - touch vacio.txt; touch .vacio.txt; ls -la
  - rm vacio.txt
  - mv .vacio.txt vacio.txt
-

Ejercicio 1

- cambiar hacia el directorio /etc
- verificar mostrando el nombre del directorio actual
- desplegar la lista de archivos en ese directorio
- volver al directorio propio
- verificar mostrando directorio actual

Usuarios y grupos

- Usuarios
  - who
  - whoami
  - passwd
  - less /etc/passwd
  - adduser
- Grupos
  - groups
  - addgroup
  - less /etc/group
  - id

Ejercicio 2

- Crear un archivo en blanco en el directorio home del usuario.
- Determinar el valor de filtro para que al ejecutar el comando ls -l filtro se muestre únicamente el archivo recientemente creado.
- Investigar cada uno de los datos presentados luego de la ejecución del comando anterior.

Caracteres especiales

- Ctrl-C
- Ctrl-D
- ~
- *
- $

Documenta los comandos

- Durante las prácticas registra los comandos ejecutados en un archivo de texto plano.
- Incluye comentarios sobre su aplicación.
- Permite reutilizar comandos.
- Repetir un proceso.
- Compartir comandos con otros.
- Comando útil:

```
gedit ~/lab-history.txt
```

Memento de comandos

- Las hojas de memento contienen los principales comandos y ejemplos de su uso.
- Descargar memento del siguiente link

https://bootlin.com/doc/legacy/command-line/command_memento.pdf

- En una consola no olvides usar:
  - la tecla “Tab / Tabulador”
  - flechas (arriba y abajo)
  - Ctrl + r

Editar archivos

- gedit
- vi
  - Muy util. Ampliamente disponible.
  - No parece muy amigable pero con unos pocos comandos es suficiente para el día a día.
  - Memendo para vi

    https://bootlin.com/doc/legacy/command-line/vi_memento.pdf

Ejercicio 3

- Crea una carpeta ejercicios
- Crea la carpeta ejercicios dos archivos con diferencias menores entre ellos
- Utiliza el comando diff para ver esas diferencias

Historia del software libre

- 1983, Richard Stallman, lidera el proyecto GNU y la cultura de software libre. Inicia el desarrollo de gcc, gdb glibc y otras herramientas importantes para su desarrollo.
- 1991, Linus Torvalds, desarrolla el kernel de Linux.
- 1995, Linux se vuelve popular en servidores.
- 2000, Linux se vuelve cada vez más popular en sistemas embebidos.
- 2008, Linux se vuelve cada vez más popular en dispositivos móbiles.
- 2010, Linux se vuelve cada vez más popular en teléfonos.

Software libre

- Un programa es considerado libre si su licencia permite:
  - Libertad de ejecutar el software para cualquier propósito.
  - Libertad de estudiarlo y modificarlo.
  - Libertad de distribuir copias.
  - Libertad de distribuir copias de versiones modificadas.
- Estas libertades son tanto para uso comercial como no comercial.

Linux embebido

- Se refiere al uso del kernel de linux y otros componentes de código abierto (open-source) en sistemas embebidos.

Ventajas de Linux y software libre

- Reutilización de componentes.
- Proporciona un ecosistema de componentes estándares (drivers, red, multimedia, criptografía, etc).
- Permite el prototipado o desarrollo de productos complejos a partir de componentes existentes.
- No es necesario re-desarrollar un kernel, stack TCP/IP, stack USB u otros.
- Permite focalizarse en el producto que se desea desarrollar.

Ventajas de Linux y software libre

(Bajo costo)

- El software libre puede colocarse en tantos dispositivos como se desee sin cargo.
- Costo cero en licencias.
- Sin embargo, usar Linux no es gratuito:
  - Se necesita bastante aprendizaje.
  - Esfuerzo de ingeniería para lograr los objetivos.
- Permite distribuir los costos hacia el hardware y el incremento en la capacitación

Ventajas de Linux y software libre

(Control total)

- Se dispone del código fuente de todos los componentes.
- Se permiten ilimitadas modificaciones, cambios, ajustes, debuging, optimizaciones por un período ilimitado de tiempo.
- Sin trabas o dependencias de otros vendedores.
- Permite tener el control total sobre el software que conforma el sistema desarrollado

Ventajas de Linux y software libre

(Calidad)

- Muchos componentes del mundo del software libre son usados por millones de sistemas.
- Usualmente tienen más calidad que los desarrollos propios aún frente a las soluciones propietarias.
- Es claro que no todos tiene buenas propiedades pero si los más usados.
- Permite diseñar sistemas con componente de base de alta calidad.

Ventajas de Linux y software libre

(Fácil de testar nuevas funcionalidades)

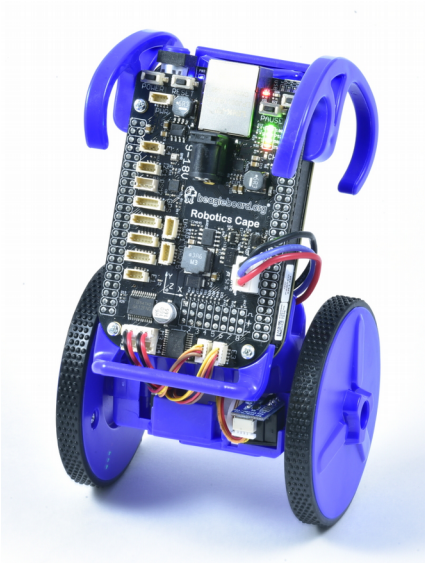
- Al estar libremente disponible es fácil obtener una porción de código y evaluarla.
- Permite evaluar varias alternativas mientras se toma la decisión.
- Mucho más simple que los procedimientos de compra y demostración de las soluciones propietarias.
- Permite fácilmente explorar nuevas posibilidades y soluciones

Ventajas de Linux y software libre

(Soporte comunitario)

- Los componentes de software libre son desarrollados por la comunidad (desarrolladores y usuarios).
- La comunidad da soporte de alta calidad.
- Generalmente el soporte es mejor pero debe tenerse en cuenta como funciona.
- Permite aumentar la velocidad con la que se resuelven los problemas.

Sistemas que usan Linux embebido



Arquitectura

- El kernel de Linux soporta un rango amplio de arquitecturas de 32 y 64 bits (x86, ARM, RISC, MIPS, PowerPC, entre otros).
- Linux no está diseñado para pequeños microcontroladores.
- El toolchain, bootloader y kernel dependen de la arquitectura.

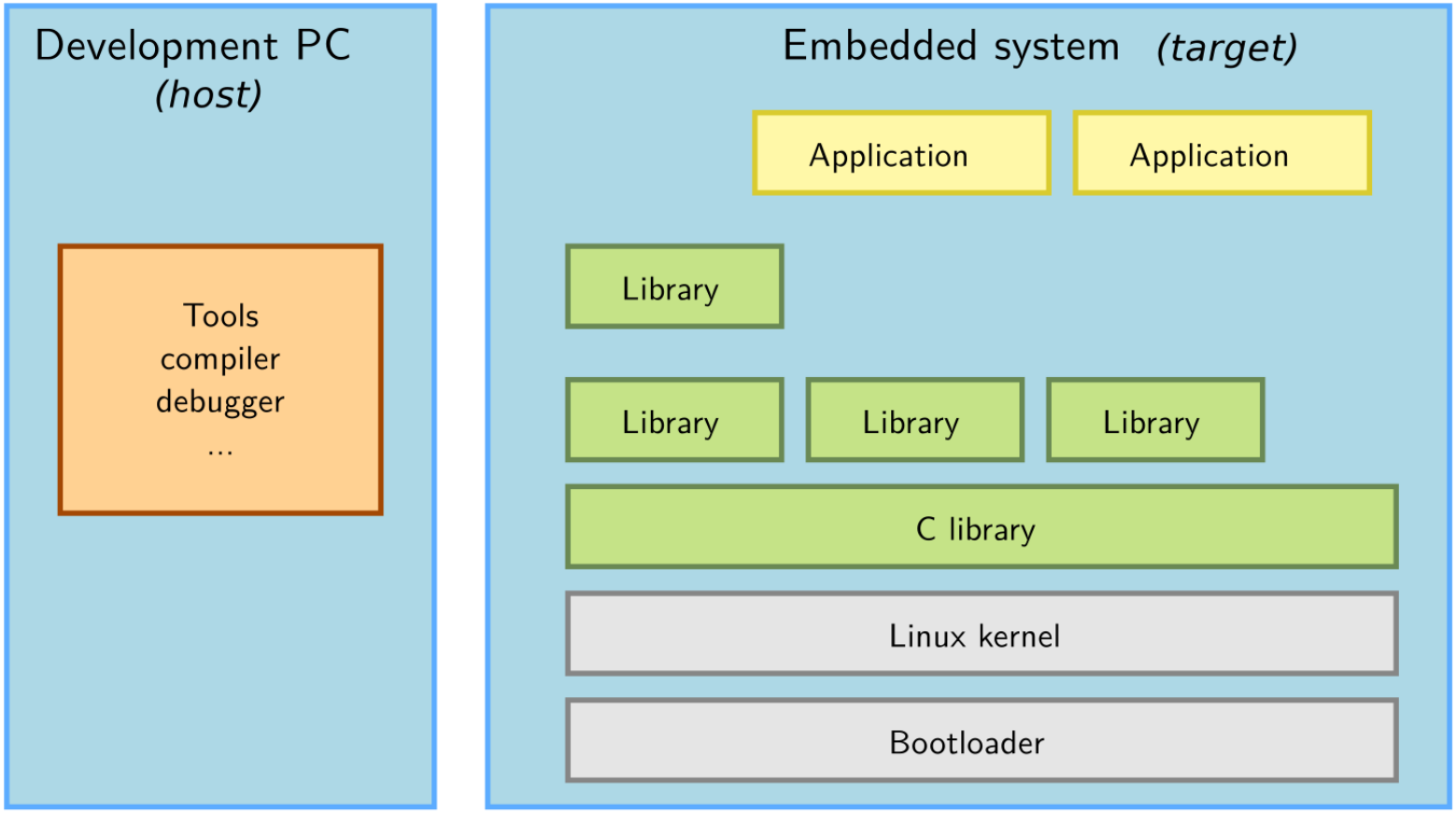
Memoria y almacenamiento

- RAM (mínimo 8MB, recomendado 32MB)
- Almacenamiento, mínimamente 4MB.

Comunicación

- Linux tiene soporte para los protocolos más comunes
  - I2C
  - SPI
  - CAN
  - 1-wire
  - SDIO
  - USB
- Y un amplio soporte de red
  - Ethernet, WiFi, Bluetooth, entre otros.
  - IP, UDP, TCP, etc.
  - Firewall, ruteo, multicast.

Arquitectura del sistema

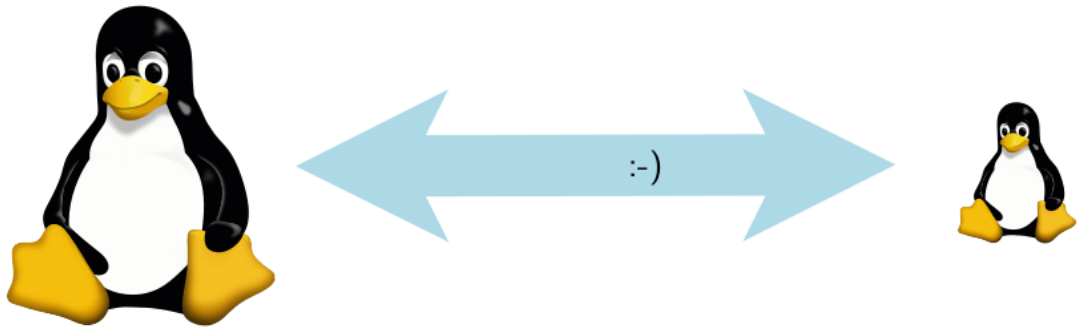


Componentes de software

- Cross-compilation toolchain
  - Compilador que ejecuta en la maquina de desarrollo pero genera código para la máquina objetivo
- Bootloader
  - Inicializado por el hardware, responsable de una inicialización básica y responsable de pasarle el control al kernel
- Kernel Linux
  - Se encarga de la gestión de procesos y memoria, el stack de red, drivers de dispositivos y proporciona servicios a las aplicaciones de usuario.
- Biblioteca de C
  - Es la interfaz entre el kernel y es espacio de aplicaciones de usuario
- Bibliotecas y aplicaciones (de terceros o propias).

SO para el desarrollo

- Es recomendable usar Linux como SO en las computadoras usadas para desarrollo (hosts).
- Todas las herramientas se desarrollan y diseñan para ejecutar en Linux.
- Linux corre en el sistema embebido.



Distribuciones de Linux

- Puede usarse cualquier distribución buena y reciente (LTS) de Linux como puesto de desarrollo.
  - Ubuntu, Debian, Fedora, openSUSE, Red Hat, etc.
- Ubuntu es una buena opción por ser ampliamente usado y fácil de usar.

Tipos de usuarios

- Linux es un sistema multi-usuario
  - El usuario root es el administrador.
- En los sistemas ubuntu no es posible logearse como root.
- Los usuarios pueden (si sus permisos lo permiten) ejecutar instrucciones no privilegiadas a través del comando sudo.

  Ejemplo: sudo mount /dev/sda2 /mnt/disk

Paquetes de software

- Linux proporciona una forma central y coherente de instalar, actualizar y eliminar aplicaciones y bibliotecas (paquetes).
- Los paquetes incluyen bibliotecas o aplicaciones, y datos asociados (versión, dependencias, entre otros).
  - .deb para Debian y Ubuntu. .rpm para RedHat, Fedora y OpenSUSE.
- Los paquetes se almacenan en repositorios en servidores http o ftp.

Gestionando paquetes

- La lista de repositorios se encuentra en el archivo /etc/apt/sources.list
- Para actualizar la lista de paquetes se utiliza el comando

  sudo apt update
- Para encontrar en nombre de un paquete se pude usar la página http://packages.ubuntu.com o el comando

  apt-cache search <keyword>

Gestionando paquetes

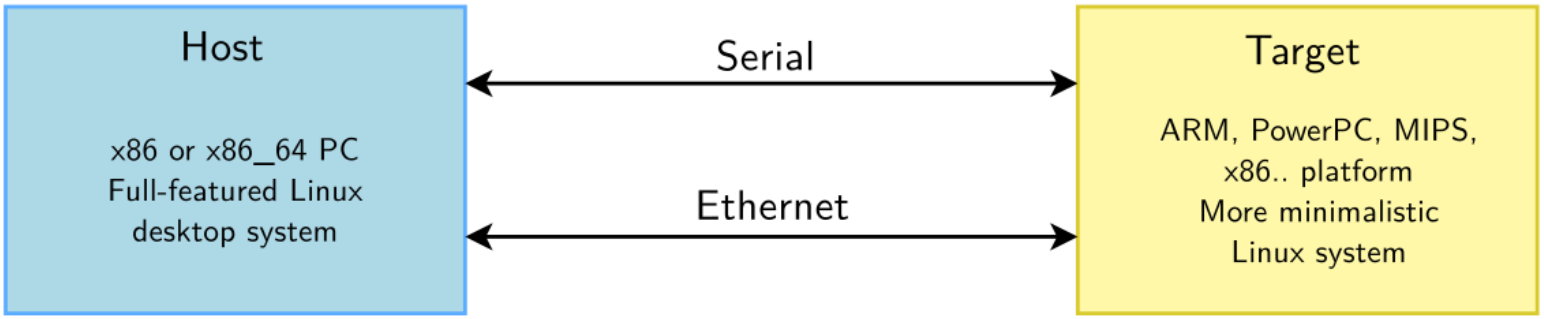
- Para instalar un paquete:
  sudo apt install <package>
- Para eliminar un paquete:
  sudo apt remove <package>
- Para instalar todos los paquetes disponibles:
  sudo apt dist-upgrade
- Para obtener información sobre un paquete:
  apt-cache show <package>
- Interfaces gráficas
  - Synaptic para GNOME
  - KPackageKit para KDE

Actualizar los paquetes del PC (host)

mina
network management | artificial intelligence

Host vs. target

- Cuando se trabaja en sistemas embebidos hay una clara división entre maquinas
  - Host
  - Target
- Estás máquinas deben estar conectadas de alguna forma.
- Pueden conectarse por la interfaz JTAG para debuggin de bajo nivel.



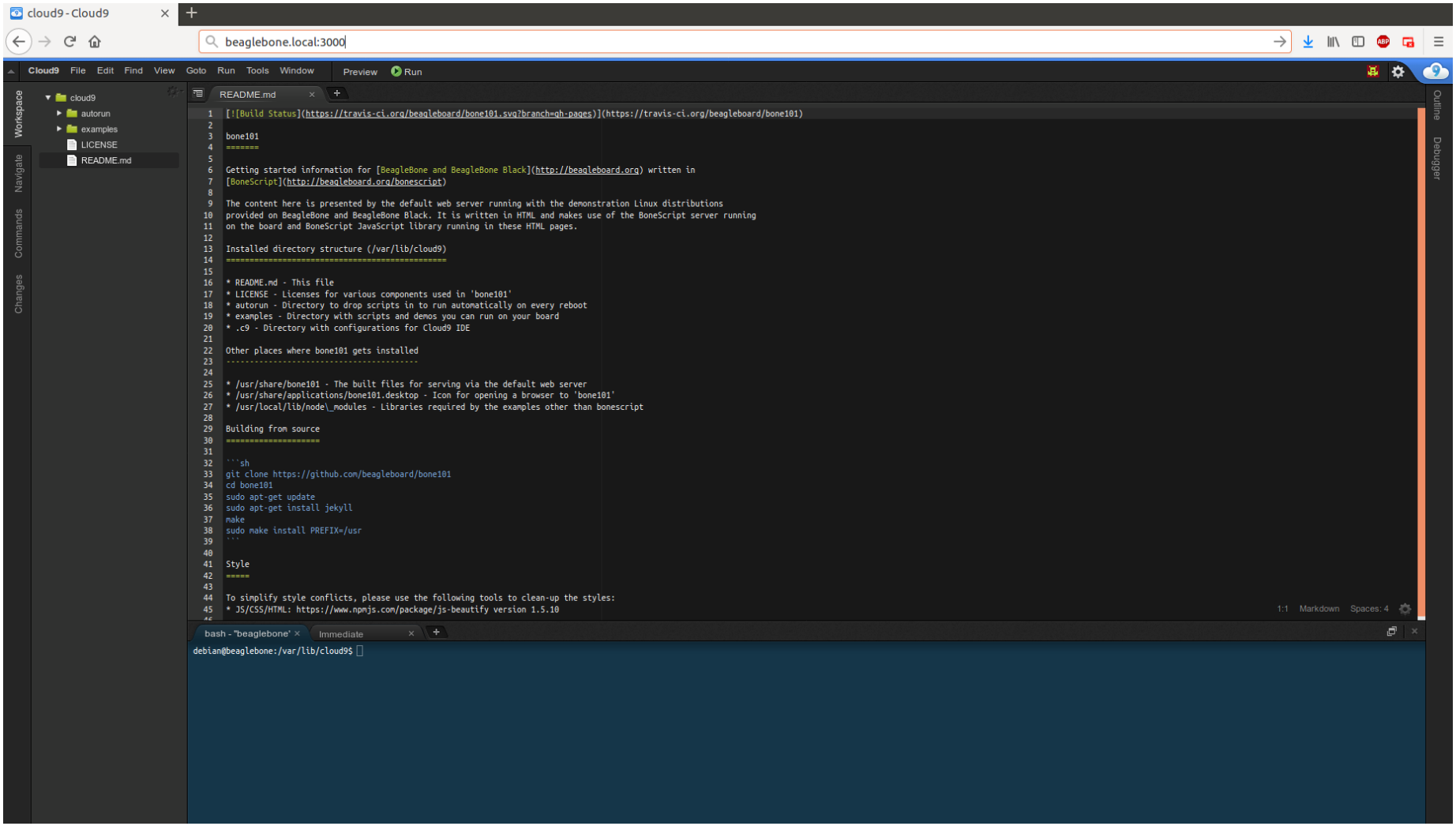
Conectarse a la placa

- Al alimentar la placa es posible conectarse a ella por:
  - ssh
  - navegador web.
- El protocolo ssh proporciona una conexión segura encriptada entre dos máquinas sobre una red potencialmente insegura.
- La conexión segura puede usarse para acceder a una terminal, transferir archivos o armar un túnel entre aplicaciones. Las aplicaciones gráficas pueden ser ejecutadas remotamente sobre SSH.
- Comando ssh

  ssh machine

  ssh user-name@machine

Conectarse a la placa



mina
network management | artificial intelligence

Trabajo práctico

(Conectarse a la placa por ssh)

- Conectar la placa a la computadora usando el cable micro USB.
- Conectarse a la placa por:
  - Ssh (192.168.7.2, debian, temppwd) o
  - Navegador web (http://beaglebone.local:3000)
- Conectarse a la wifi

  https://github.com/beagleboard/beaglebone-blue/wiki/Wifi

- Actualizar el sistema operativo

Bibliografía

[Kernighan1995] – El entorno de programación Unix. Kernighan/Pike, 1995.

[González2019] – Tutorial básico de Unix,

http://iie.fing.edu.uy/~vagonbar/unixbas/tutorial.htm

visitada en Mayo de 2019.

¿Preguntas?