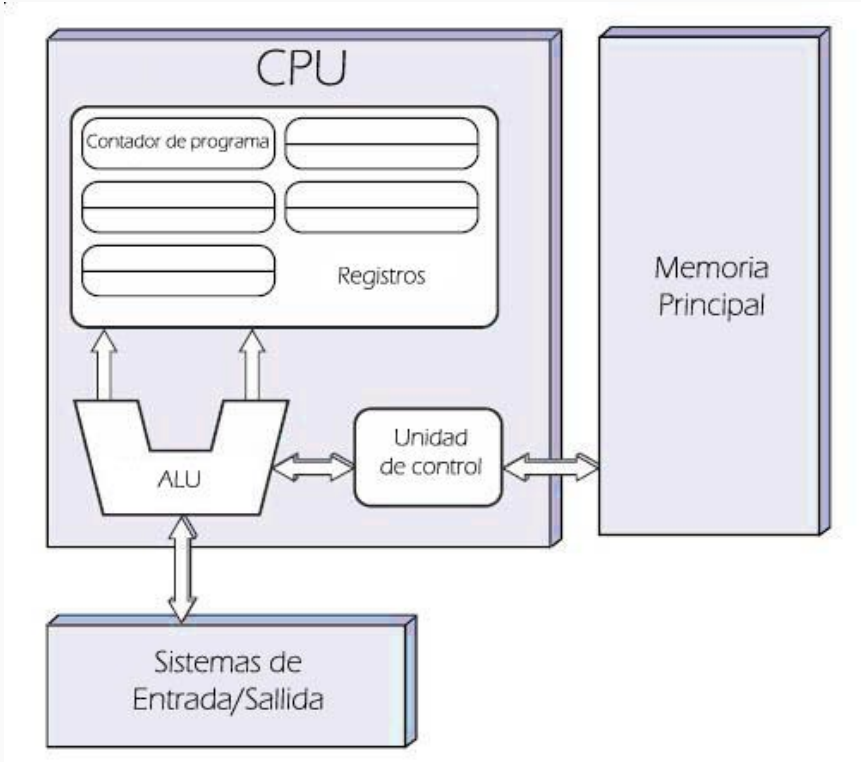
Introducción

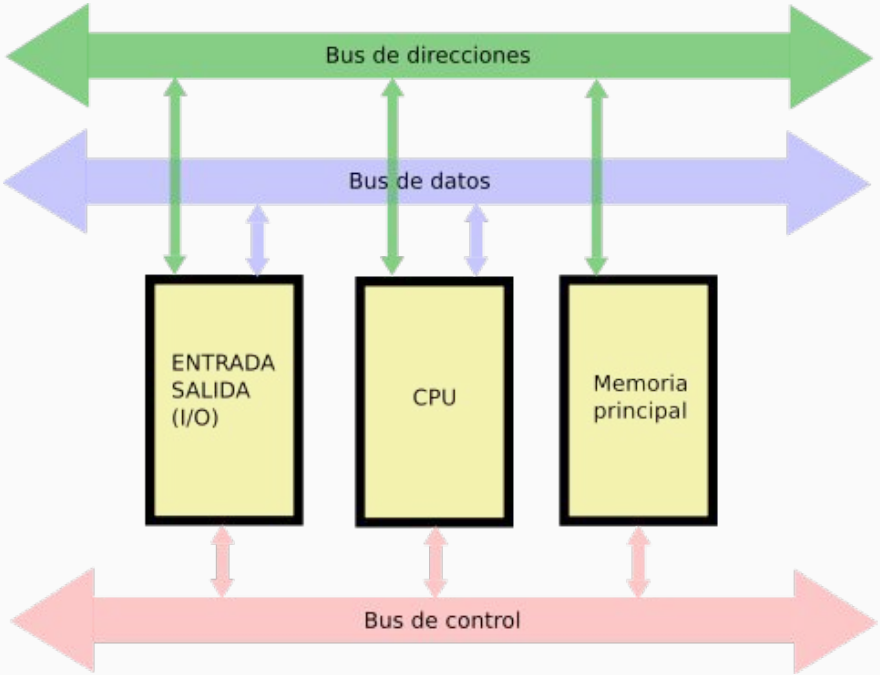
Introducción

- Hardware
- Gnu/Linux
- Tipos de lenguajes
- Introducción a C/C++

Hardware

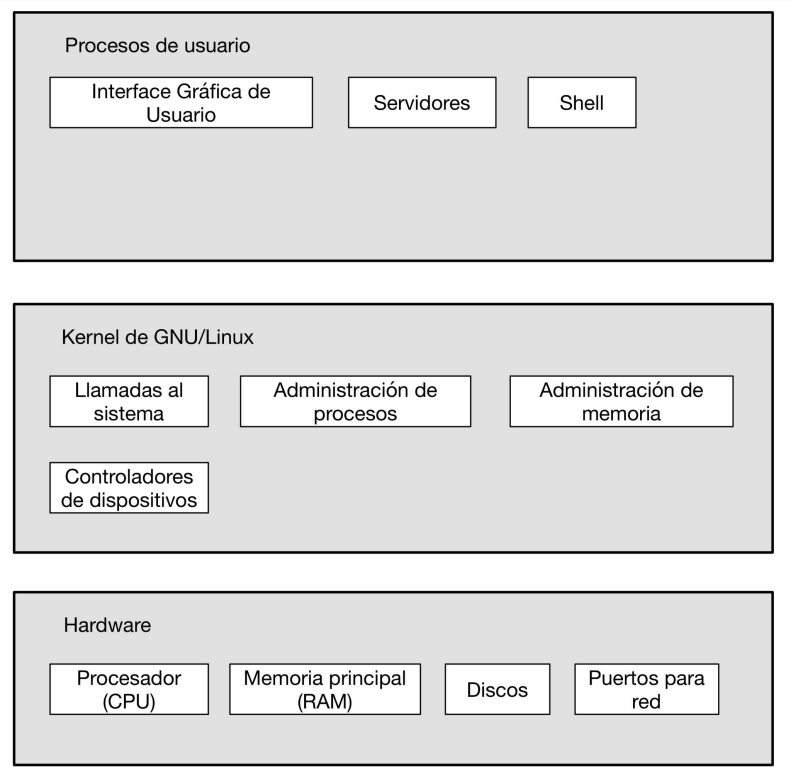


Imágen tomada de
https://es.wikipedia.org/wiki/Arquitectura_de_Von_Neumann#/media/Archivo:Arquitecturaneumann.jpg

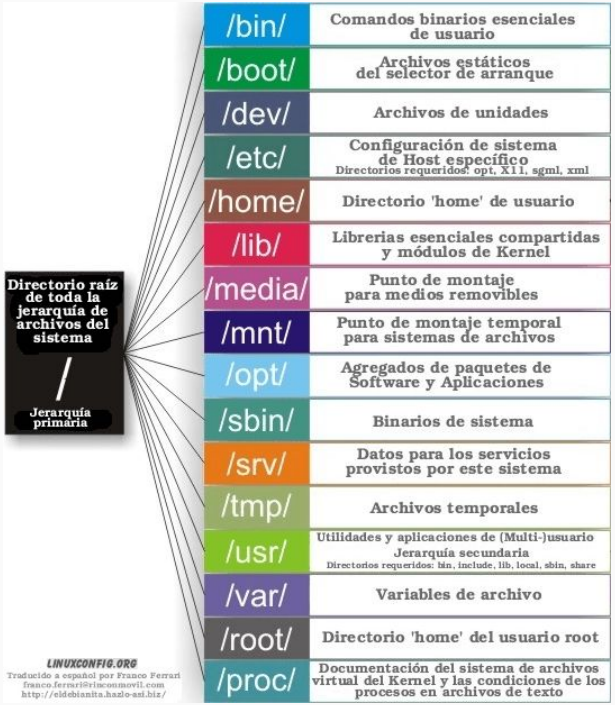


Imágen tomada de
https://elpuig.xeill.net/Members/vcarceler/c1/didactica/apuntes/ud2/na1

Sistema operativo



Imágen tomada de
https://www.uv.mx/personal/rcarrera/programacion-de-sistemas/3-el-sistema-operativo-linux/4-introduccion-a-gnulinux/



Imágen tomada de
https://blog.desdelinux.net/estructura-de-directorios-en-linux/

Tipos de lenguajes

Interpretados

- Son más flexibles
- Son más portables
- Son más lentos
- Requiren el interprete presente en cada OS

Compilados

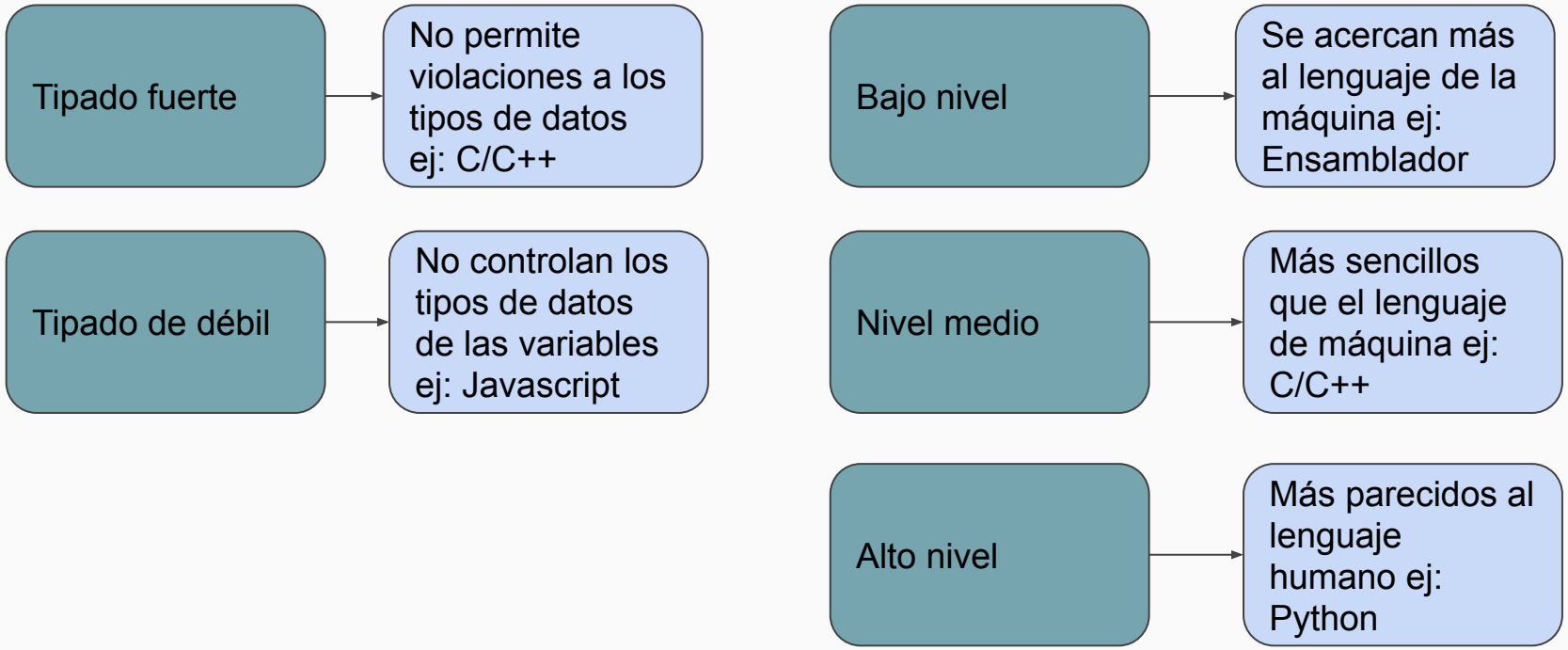
- Son más rápidos
- Se requiere compilar para cada OS y arquitectura de CPU (menos portables)

Intermedio

- Parecidos a los interpretados pero se requiere un paso intermedio de compilación.

Imágen tomada de
https://qph.cf2.quoracdn.net/main-qimg-672dd724b781aa3faa74456daea9d861-lq

Tipos de lenguajes



C/C++

C es un lenguaje de programación compilado de propósito general desarrollado por Dennis Ritchie entre 1969 y 1972 en los laboratorios Bell.

C++ es un lenguaje de programación orientado a objetos cuyo objetivo es extender C y ser interoperables, fue desarrollado por Bjarne Stroustrup en 1979.

C/C++ son lenguajes compilados de nivel intermedio fuertemente tipado.

C/C++

- Variables/Constantes
  - Tipos datos
  - Punteros
- Operadores
- Condicionales
- Bucles
- Funciones
- Orientación a objetos

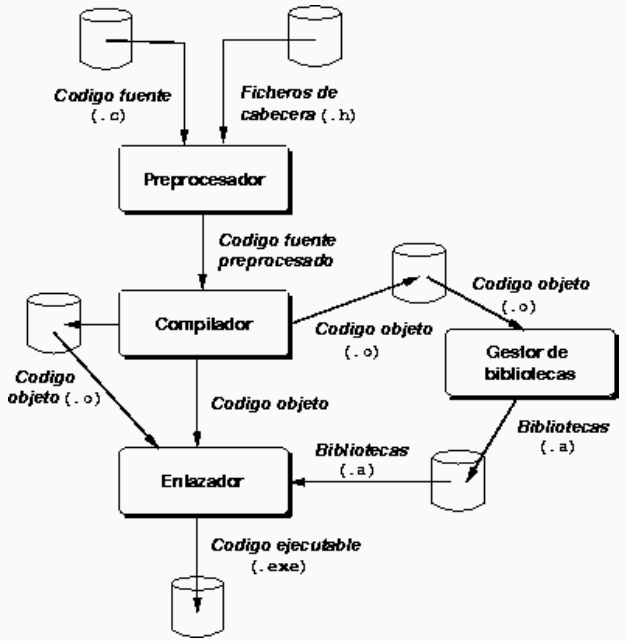
- Compilación

Hola mundo y Compilación

```
#include<iostream>
int main()
{
    std::cout<<"hola mundo"<<std::endl;
    return 0;
}
```

g++ hola.cxx ./a.out

g++ hola.cxx -o hola ./hola



https://es.quora.com/Se-puede-crear-un-lenguaje-de-programaci%C3%B3n-sin-crear-su-respectivo-compilador-o-interprete

Tipos de datos

Fundamentales : int, float, double, bool, char, and void.

Especiales: enum, typedef

Modificadores de tipos: const, signed, unsigned, short, long, long long

Tarea: volatile, mutable, static, register

Definidos por el usuario: class, struct, union

varoles espaciales: NULL, nullprt

https://www.dremendo.com/cpp-programming-tutorial/cpp-data-types-and-modifiers

Tipos de datos

Fundamentales : int, float, double, bool, char, and void.

Especiales: enum, typedef

Modificadores de tipos: const, signed, unsigned, short, long, long long

Tarea: volatile, mutable, static, register

Definidos por el usuario: class, struct, union

varoles espaciales: NULL, nullprt

https://www.dremendo.com/cpp-programming-tutorial/cpp-data-types-and-modifiers

Tipos de datos

Operadores aritméticos [editar]

Nombre del operador	Sintaxis	Sobrecargable	Incluido en C
Suma	`a + b`	✓ Sí	✓ Sí
Suma y asignación	`a += b`	✓ Sí	✓ Sí
Resta	`a - b`	✓ Sí	✓ Sí
Resta y asignación	`a -= b`	✓ Sí	✓ Sí
Multiplicación	`a * b`	✓ Sí	✓ Sí
Multiplicación y asignación	`a *= b`	✓ Sí	✓ Sí
División	`a / b`	✓ Sí	✓ Sí
División y asignación	`a /= b`	✓ Sí	✓ Sí
Módulo	`a % b`	✓ Sí	✓ Sí
Módulo y asignación	`a %= b`	✓ Sí	✓ Sí
Más unario (promoción entera)	`+a`	✓ Sí	✓ Sí
Menos unario (opuesto)	`-a`	✓ Sí	✓ Sí
Incremento prefijo	`++a`	✓ Sí	✓ Sí
Incremento postfijo	`a++`	✓ Sí	✓ Sí
Decremento prefijo	`--a`	✓ Sí	✓ Sí
Decremento postfijo	`a--`	✓ Sí	✓ Sí

Operadores de comparación [editar]

Nombre del operador	Sintaxis	Sobrecargable	Incluido en C
Menor que	`a < b`	✓ Sí	✓ Sí
Mayor que	`a > b`	✓ Sí	✓ Sí
Menor o igual que	`a <= b`	✓ Sí	✓ Sí
Mayor que o igual que	`a >= b`	✓ Sí	✓ Sí
Igual que	`a == b`	✓ Sí	✓ Sí
Diferente que / No igual que	`a != b`	✓ Sí	✓ Sí
Comparación a tres sentidos	`a <=> b`	✓ Sí	✗ No

Operadores lógicos [editar]

Nombre del operador	Sintaxis	Sobrecargable	Incluido en C
Negación lógica (NOT)	`!a`	✓ Sí	✓ Sí
Y lógico (AND)	`a && b`	✓ Sí	✓ Sí
O lógico (OR)	`a || b`	✓ Sí	✓ Sí

https://es.wikipedia.org/wiki/Anexo:Operadores_de_C_y_C%2B%2B

https://gist.github.com/omazapa/65d02b6c9eef26c4ced5de0b2b50f714