Linux Internals & Networking

System programming using Kernel interfaces

Team Emertxe

Contents

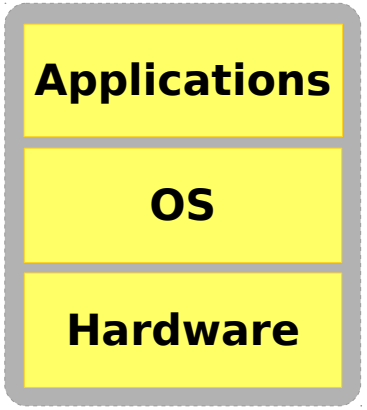
Linux Internals & Networking
Contents

Introduction

Introduction
Let us ponder...

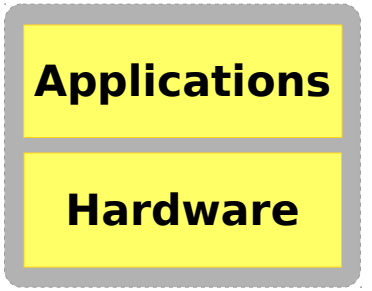
- What exactly is an Operating System (OS)?
- Why do we need OS?
- How would the OS would look like?
- Is it possible for a team of us (in the room) to create an OS of our own?
- Is it necessary to have an OS running in a Embedded System?
- Will the OS ever stop at all?

OS: What?



OS is an interface between application and hardware
Which abstracts H/W layer from user

Is it possible to make an embedded without OS.?



OS : Its Need

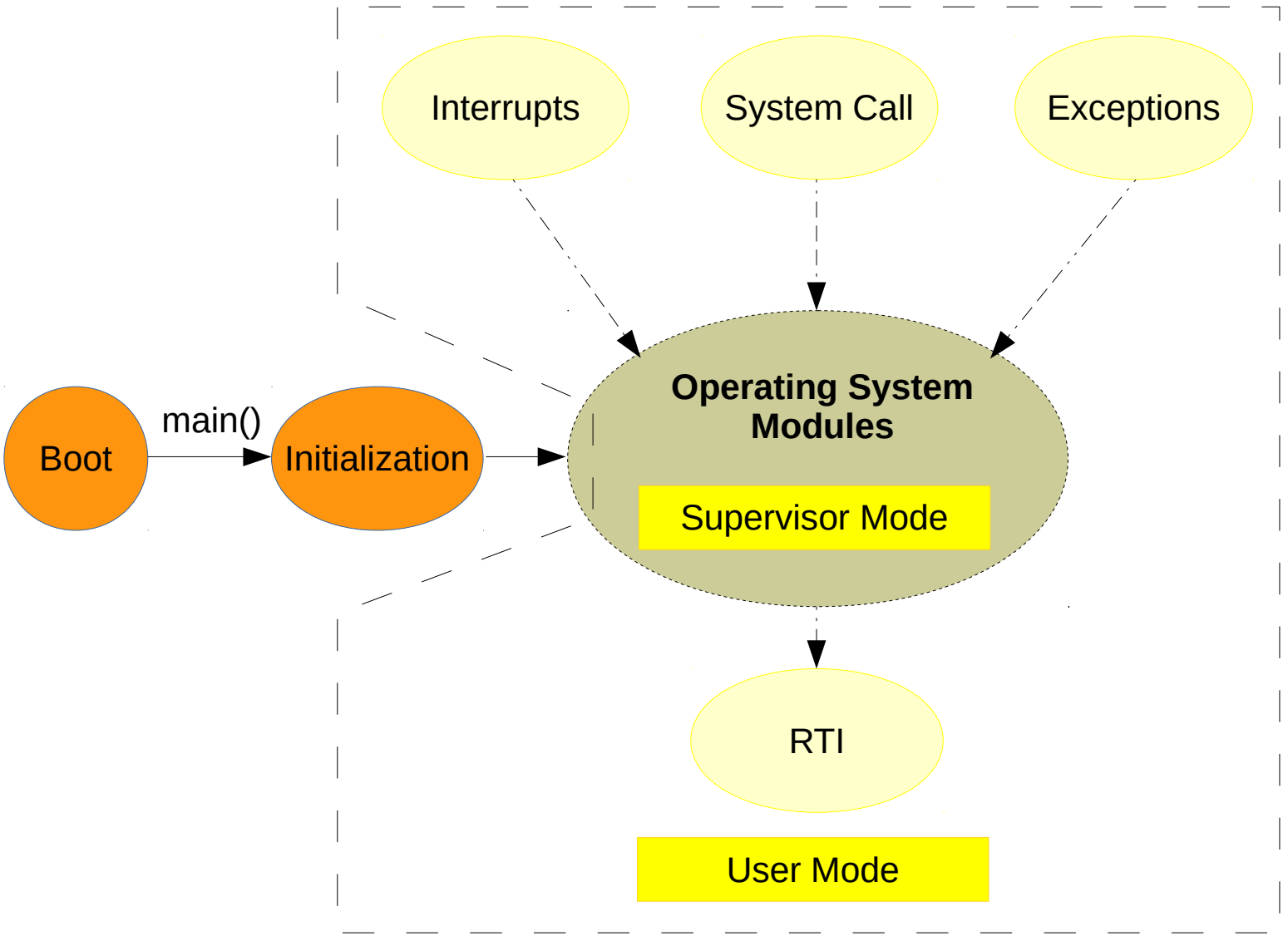
- Multitasking
- Multi user
- Scheduling
- Memory management etc ...

Linux Booting
Sequence

System Start-up	BIOS / Boot Monitor
Stage 1 Boot Loader	Master Boot Record
Stage 2 Boot Loader	LILO, GRUB, etc
Kernel	Linux
Init	User Application

EMERTXE

OS: Control flow



History

1940 - 1955	ENIAC, Mechanical switches, Mainframe computers
1955 - 1970	Concept OS, FORTRAN, IBM OS/360, Multiprogramming, Minicomputers.

Vertical

Horizontal

1970 - 1980	UNIX, Microprocessors(intel), Personal computers age
1980 - 1990	First computer IBM 5150, DOS, Apple & Windows with GUI
1990 - NOW	Linux, ios, Android.....

EMERTXE

Vertical vs Horizontal

- Vertical
  - Hardware & Software made by same company
  - OS was integrated part of the Hardware
  - Applications were propitiatory
- Horizontal
  - Hardware & Software made by different company
  - OS is an independent software, that can run on diversified set of hardware
  - Applications are developed by everybody (propitiatory or open source)

Quiz

- How would the OS look like ?

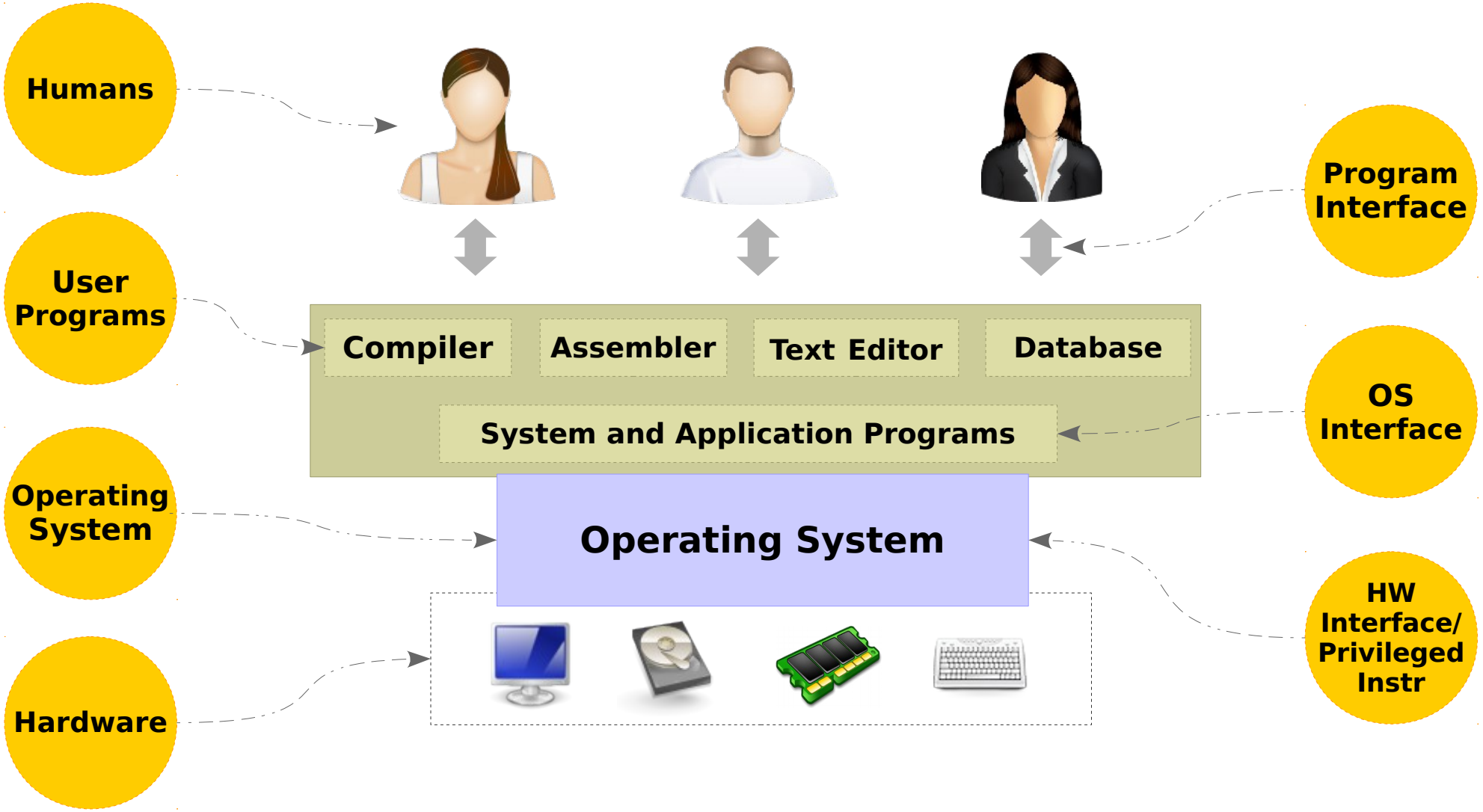
  a) H/W only

  b) S/W only

  c) S/W + H/W

- How big is OS ?

Introduction
Operating System



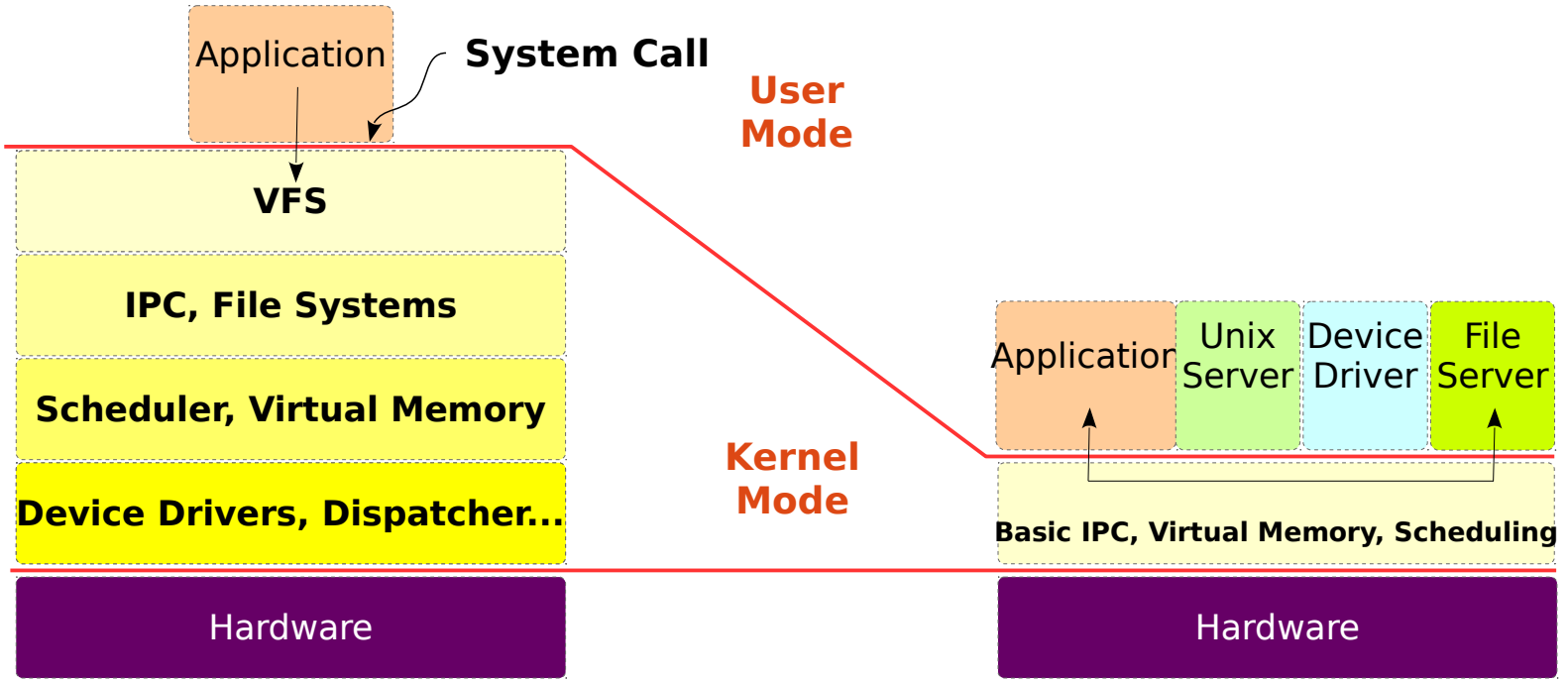
Introduction
Kernel Architecture

- Most older operating systems are monolithic, that is, the whole operating system is a single executable file that runs in 'kernel mode'
- This binary contains the process management, memory management, file system and the rest (Ex: UNIX)
- The alternative is a microkernel-based system, in which most of the OS runs as separate processes, mostly outside the kernel
- They communicate by message passing. The kernel's job is to handle the message passing, interrupt handling, low-level process management, and possibly the I/O (Ex: Mach)

Introduction
Kernel Architecture

Monolithic Kernel
Based Operating System

Micro Kernel
Based Operating System



Introduction
Monolithic vs Micro

Monolithic	Micro
• Kernel size increases because kernel + kernel subsystems compiled as single binary	• Kernel size is small because kernel subsystems run as separate binaries.
• Difficult to extend or fix the bug	• Easily extensible and bug fixing.
• Need to compile entire source code.	• No Need.
• Faster, run as single binary	• Slower due to complex message passing between services
• Communication between services is faster.	• Communication is slow
• No crash recovery.	• Easily recoverable from crash.
• More secure	• Less secure
• Bad maintainability	
• Eg: Windows, Linux etc	• Eg: MacOS, WinNT

RTOS

- Real time means fast..?
- An operating system that guarantees a certain capabilities within a specified time constraint.
- Must also able to respond predictably to unpredictable events
- Uses: where time is crucial.
  - In Aircraft,
  - Nuclear reactor control systems
- Eg: LynxOS, OSE, RTLinux, VxWorks, Windows CE

Stay Connected

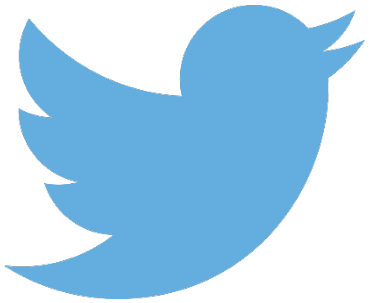
About us: Emertxe is India's one of the top IT finishing schools & self learning kits provider. Our primary focus is on Embedded with diversification focus on Java, Oracle and Android areas

Emertxe Information Technologies,

No-1, 9th Cross, 5th Main,
Jayamahal Extension,
Bangalore, Karnataka 560046

T: +91 80 6562 9666
E: training@emertxe.com

https://www.facebook.com/Emertxe



https://twitter.com/EmertxeTweet

https://www.slideshare.net/EmertxeSlides

EMERTXE

Thank You