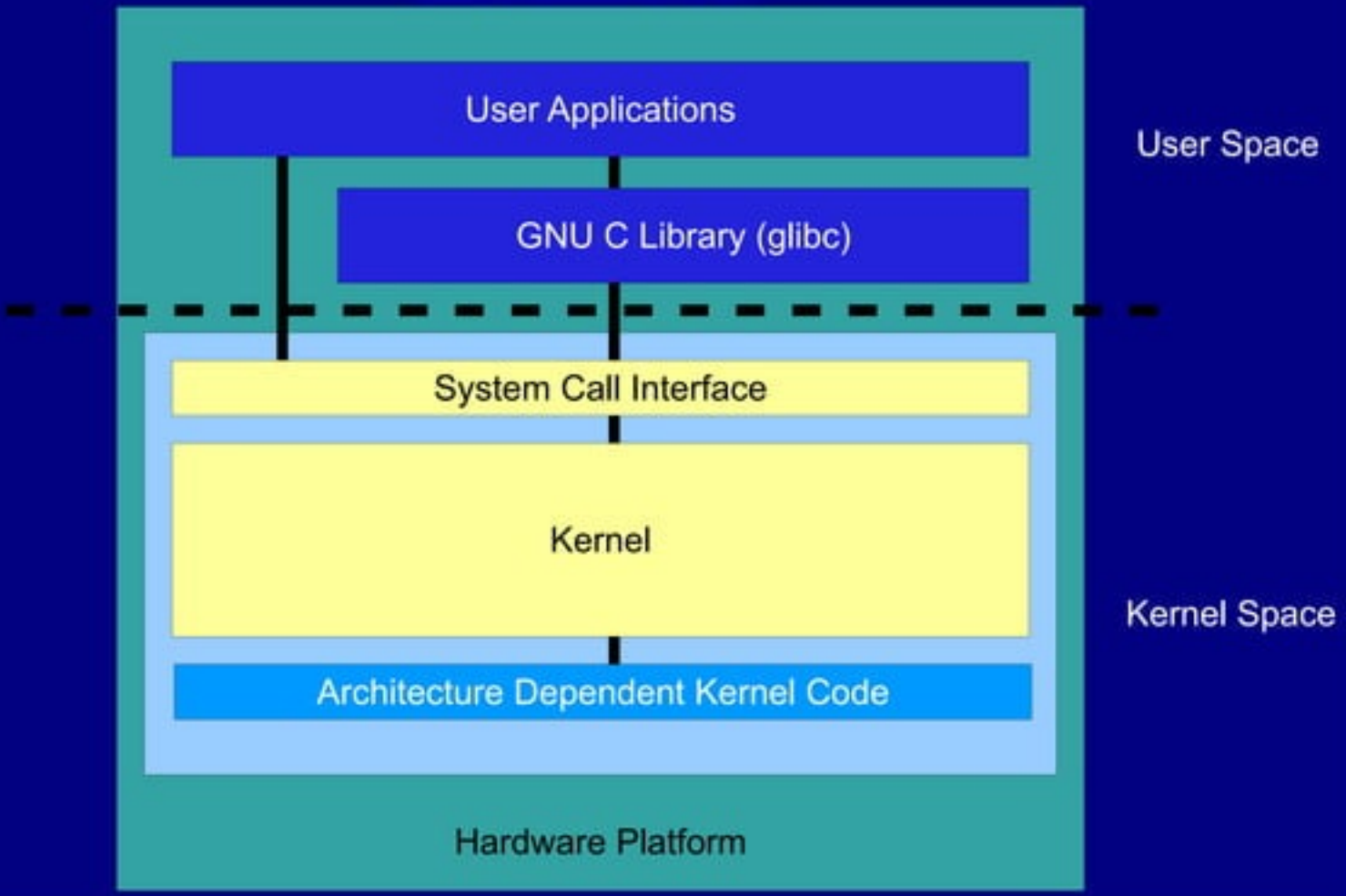
Architecture of the Linux Kernel

by Dominique Gerald M Cimafranca
dominique.cimafranca@gmail.com

Fundamental Architecture



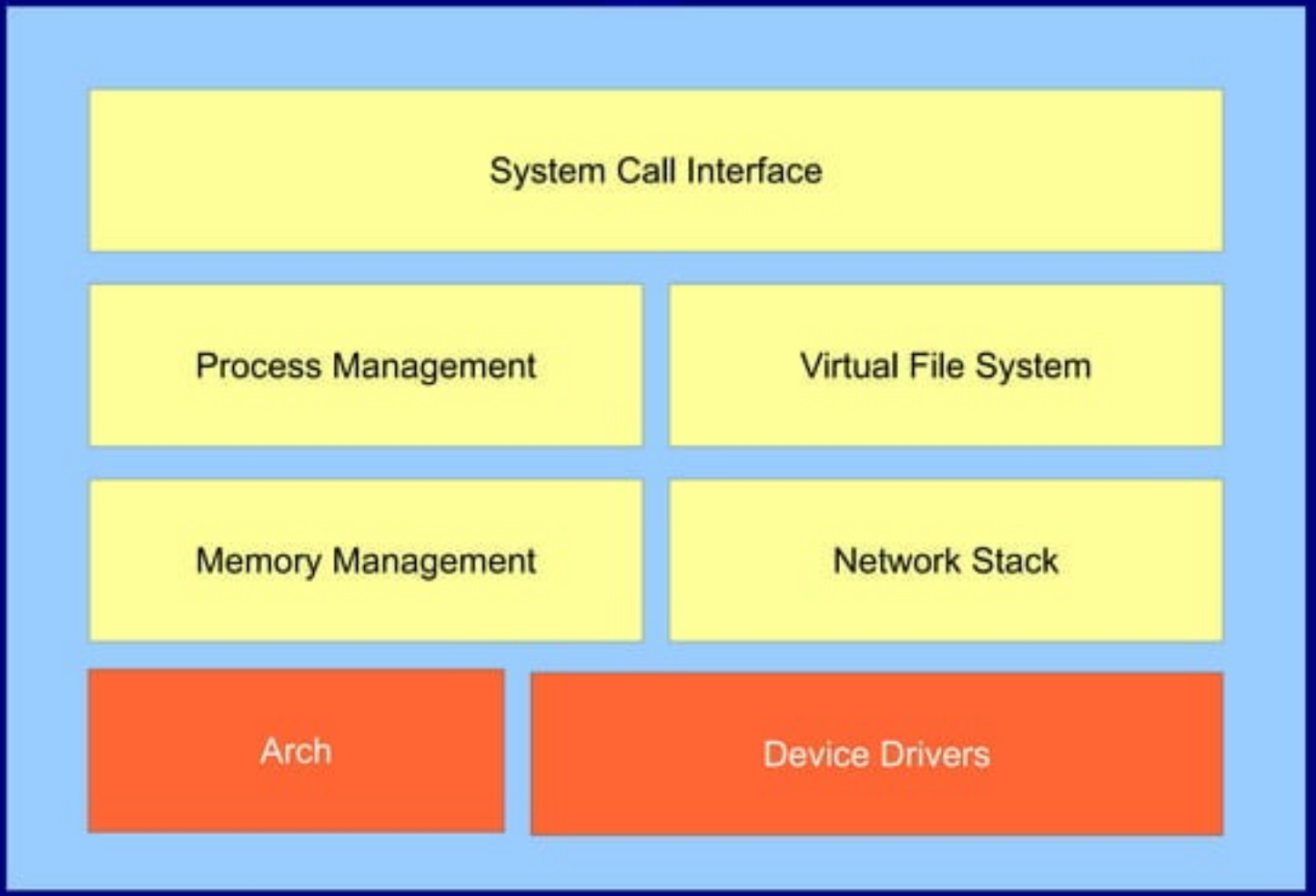
User Space Components

- User applications
- glibc
  - Provides the system call interface that connects to the kernel
  - Provides the mechanism to transition between user-space application and kernel
- Each user space process occupies its own virtual address space (vs the kernel which runs on the single address space)

Kernel Space Components

- System Call Interface
  - Provides the basic functions such as **`read()`** and **`write()`**
- Kernel
  - Architecture-independent kernel code
  - Common to all processor architectures supported by Linux
- Architecture-Dependent Code
  - Processor- and platform-specific code
  - Also known as **Board Support Package**

Kernel Subsystems



System Call Interface

- Provides the means to perform function calls from user space into the kernel.
- This interface can be architecture dependent, even within the same processor family.
- Can be found in
  - ./linux/kernel
  - ./linux/arch

Process Management

- Focused on the execution of processes
- Each has an individual virtualization of the processor (thread code, data, stack, and registers)
- Kernel provides API through SCI to start, stop, and communicate with processes
- Processes are managed by a scheduler

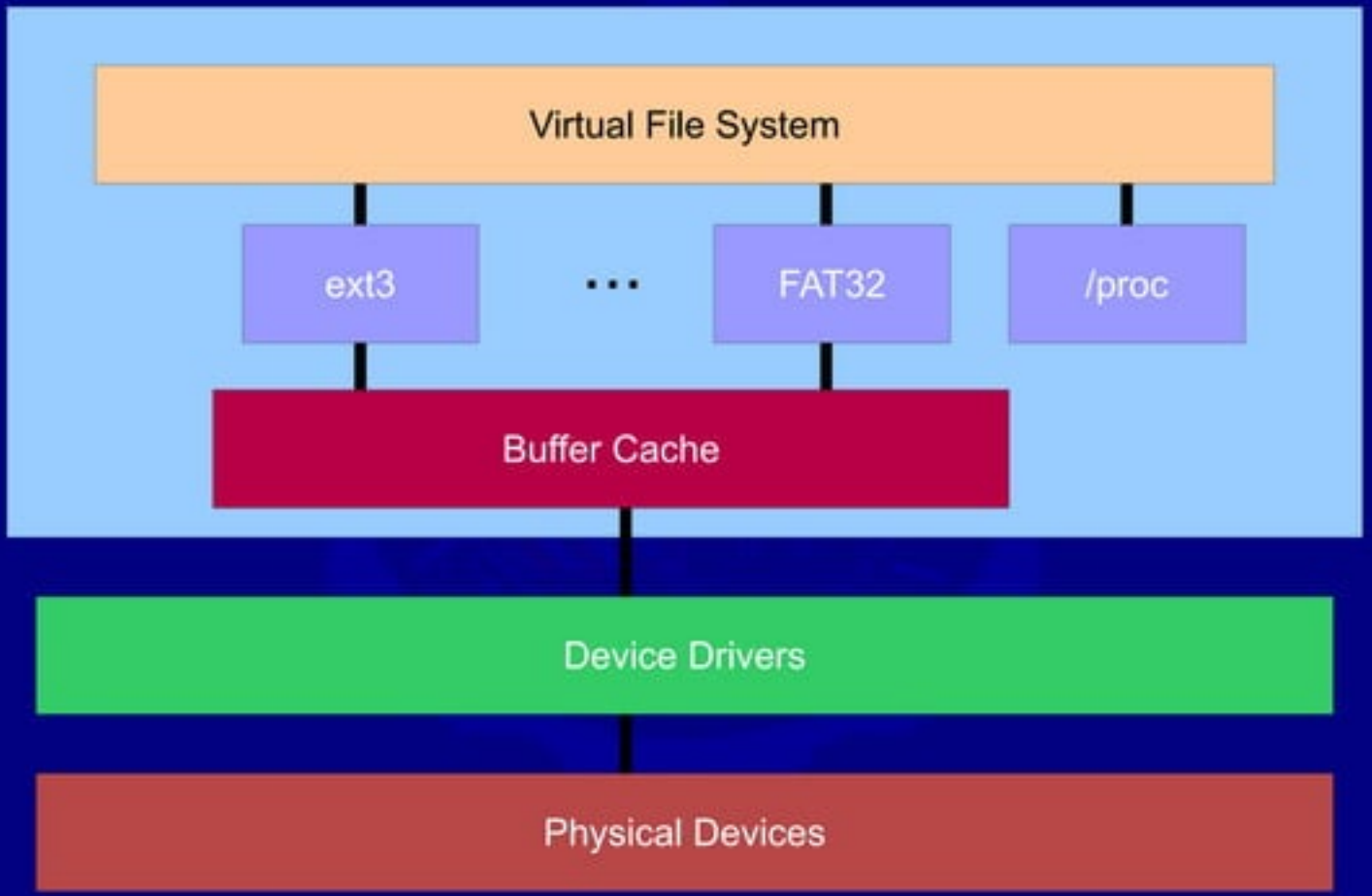
Scheduler

- Kernel implements a scheduling algorithm
  - Operates in constant time, regardless of threads
  - O(1), meaning, same time to schedule one thread or many threads
- Can be found in
  - ./linux/kernel
  - ./linux/arch

Memory Management

- Memory is managed in *pages*
  - Typically 4KB per page for most architectures
  - Can be adjusted
- Support for hardware mechanisms for physical and virtual mappings, e.g. MMU on Pentium
- Keeps tracks of which pages are full, partially used, or empty
- Or if physical memory runs out, *swap* to disk
- Can be found in ./linux/mm

Virtual File System



Virtual File System

- Presents a common API abstraction of functions such as open, close, read, and write
- Translates to abstractions specific to a file system
- Support for over 50 different file systems
- Can be found in ./linux/fs

Buffer Cache

- Caching layer that optimizes access to the physical devices by keeping data around for a short time
- Provides a common set of functions to the file system layer (independent of any particular file system)

Network Stack

- Follows a layered architecture modeled after the TCP/IP protocols
- TCP layer communicates with SCI via sockets
- Sockets provide a standard API to the networking subsystem
  - Manage connections
  - Move data between endpoints
- Can be found in ./linux/net

Device Drivers and Architecture-Dependent Code

- Most of the Linux kernel source code consists of device drivers
  - Can be found in ./linux/drivers
- While Linux kernel is mostly architecture
  - Can be found in ./linux/arch

Questions?

References

- Anatomy of the Linux Kernel, M. Tim Jones, IBM Developerworks (http://www.ibm.com/developerworks/linux/library/l-linux-kernel/)

Architecture of the Linux Kernel

by Dominique Gerald M Cimafranca
dominique.cimafranca@gmail.com