UNIT - 3

LAYERING & VIRTUALIZATION

* A common approach to managing system complexity is to identify a set of layers w/ well-defined interfaces among them.

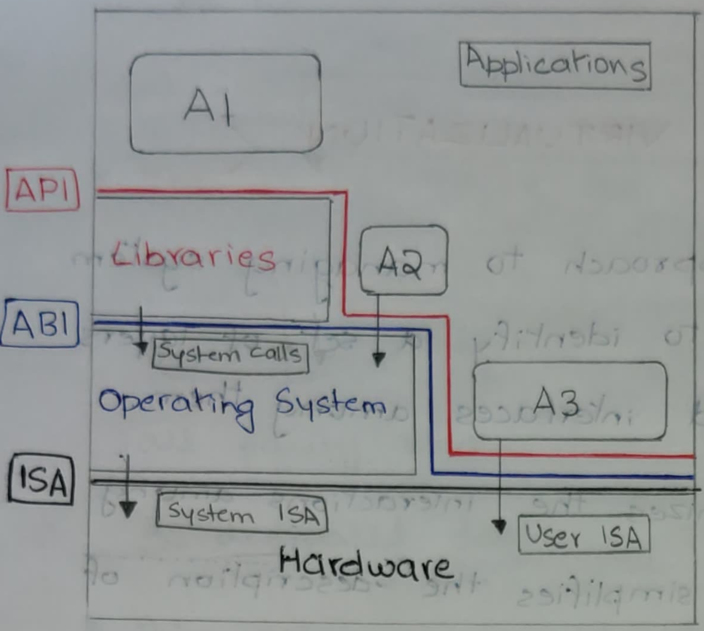
* Layering minimizes the interactions among subsystems & simplifies the description of the subsystems.

* Each subsystem is abstracted through it's interfaces w/ other subsystems.

* The ISA (Instruction Set Architecture) defines a processor's set of instructions.

- The hardware supports 2 execution modes → Kernel / privileged, → user.

- The instruction set consists of 2 sets of instructions — privileged instr. that can be executed only in the kernel mode, & nonprivileged instr. that can be executed in kernel & in user mode but that behave differently.



This diagram shows the interfaces among the software components & the hardware.

* The hardwrare consists of 1 or more multicore processors, a system interconnect (eg: buses), a memory translation unit, the main memory, & I/O devices, including 1 or more networking interfaces.

* Applications written mostly in HLL (high-level lang.) often call library modules & are compiled into object code.

* Privileged instructions, such as I/O requests, cannot be executed in user mode; instead, application & library modules issue system calls & the OS determines

red whether the privileged instructions required by the application do not violate system security or integrity & if they don't, executes them on behalf of the user.

- The 1st interface is the ISA at the boundary of the hardware & the s/ware.
- The next interface is the ABI (Application Binary Interface), which allows the applications & the library modules to access the hardware. The ABI doesn't include privileged system instructions; instead it invokes system calls. → requesting a service from the kernel of the OS.
- The API (Application Program Interface) defines the set of instructions the hardware was designed to execute & gives the application access to the ISA. It includes HLL library calls, which often invoke system calls.
- A process is the abstraction for the code of an application at execution time; a thread is a lightweight process.
- The ABI is the projection of the system seen by the process, & the API is the projection of the system from the perspective of the HLL program

* The binaries created by a compiler for a specific ISA & a specific OS are not portable. Such code cannot run on a computer w/ a diff. ISA or OS.

However, it is possible to compile an HLL program for a VM environment, where portable code is produced & distributed, & then converted by binary translators to the ISA of the host system.

by

dynamic binary translation