1 What is an Operating System?

A search query goes to a DNS server, then goes through the Internet to its address. Along the way, a load balancer that uses algorithms to manage many machines that give partial responses that are put back together and sent back. Every single device involved in this process has an operating system.
Operating System: special layer of software that provides application software access to hardware resources. These resources include processors, screen, etc. An application may want to access many of these resources - you must provide some level of abstraction to application developers. The OS "visualizes the hardware." The OS does the following:

- Manage Resources: generally more than 1 application running on a device - protect against overload (Referee)
- Provide Abstraction: Different iPhones can run the same software from abstraction (Illusionist)
- Implements specific algorithms and techniques: scheduling, concurrency, transactions, security, etc. (Glue)

1.1 Examples of Operating Systems Design

Process: Some allocation of resources on a machine. Typical process: load the software into memory using OS, then begin using the processor. When you click on the screen, another process begins in the processor (**Context Switch**) when it finishes the new process, the OS goes back to the old application. More and more processes allocate different memory and the OS must manage these and protect them from one another in memory.

Protection Boundary: prevents the hardware from messing with the OS. Sometimes, however, devices will directly access memory without going to the processor so as to not put more load on it (**direct memory access**)

1.2 What Makes OS So Exciting

Moore's Law and microprocessors have increased in power and efficiency. All this computing power is also becoming cheaper. However, to make chips faster, we increase the frequency they are running at, but this increases power and pushes to physical limitations. The answer is to have more cores: ManyCore Chips. Parallelism must be exploited at all levels. Storage capacity is still growing incredibly fast (exponential). Network capacity is also growing exponentially. Where you have a processor, you have an OS.

1.3 Challenge of Complexity

Applications consist of a variety of software modules that urn on a variety of devices that implement different hardware architectures, run competing appli-

cations, fail in unexpected ways, can be under a variety of attacks. More and more cores also means these problems multiply. It isn't feasible to test software for all possible environments and combinations of components and devices - the question is how serious are the bugs, not if there are bugs.

Taming this complexity is complicated: every piece of computer hardware different - the OS insulates the application from the hardware and provides the abstraction to make this possible and easy to program

1.4 Virtual Machine

Software emulation of an abstract machine that give programs illusion they ow the machine. Make it look like hardware has features you want. **Process VM** support execution of a single program; functionality type provided by OS. **System VM** supports execution of entire OS and its applications