OS a large piece of sophisticated software that provides an abstract computing environment

An OS manages resources and supplies computational services

An OS hides low-level hardware detailed from programmers

OS software is among the most complex ever devised

Example services an OS supplies

Support for concurrent execution

Process synchronization

Process to process communication mechanism

Process to process message passing and asynchronous events

Management of address spaces and virtual memory support

Protection among users and running applications

High level interface for I/O devices

File systems and file access facilities

Internet communication

The system interface

A single copy of OS runs at any time

Hidden from users

accessible only to application programs

the application program interface API

define services OS makes available

defines arguments of the services

provides access to

interface to system services

appreas to operate like a function call mechanism

OS makers set of functions available to applications

Application supplies argument using standard mechanism

Application calls an OS function to access a service

#include <xinu.h>

Void main(void){

putc(CONSOLE,’h’);

putc(CONSOLE,’i’);

putc(CONSOLE,’\n’);

}

Most services employ a set of several system calls

Examples

Process management service includes function to suspend and then resume a process

Socket API used for internet communication includes many functions

System calls used with I/O

Open close read write paradigm

Open to connect to a file

Write data or read

Calls close to terminate use

The set of I/ functions coordinate

Open returns a descriptor, d

Read and write operate on descriptor d

Concurrent processing

Real concurrency is only achieved when hardware operates in parallel

I/O devices operate at same time as processor

Multiple processors/cores each operate at the same time

Apparent concurrency is achieved with multitasking

Multiple programs appear to operate simultaneously

How multitasking works

Users start multiple compuj…

program consists of static code and data

A function is a unit application program code

A process (a thread of execution) is an active computation (execution of a program)

A process is an OS abstraction

Can be created when needed

Is manded sent..

Difference between function call and process creation

A normal function call only involves a single computation executes synchronously

The create system call starts a new process and returns

Both the old process and the new one process to run after the call

Two processes running the same code

Distinction between a program and

Storage allocation when multiple processes execute

Different stack

Different Local variables

Different copies of argument

A process May have private heap storage as well