OPERATING SYSTEM [CS-301]

TUTORIAL 3

[BHAGYA VINOD RANA]
U19CS012

1> What are benefits and disadvantages of each of the following? Consider both system level and programmer level.

a) Synchronous (live) and Asynchronous Communication

Message passing may be either blocking (synchronous) or non-blocking (asynchronous).

Synchronous operation blocks a process till the operation completes so control will not return to the application until completion of operation.

An Asynchronous Operation is non-blocking and only initiates the operation. This means nothing is waiting for the operation to complete and the call returns immediately while completion of operation is communicated later by setting some variable or call-back routine.

(A) SYNCHRONOUS Communication

(i) **Advantage** : You can get immediate response. Instant feedback faster communication. (Allows rendevous between sender & receiver)

(ii) **Disadvantage**: Less time to think what you want to say

The disadvantage of blocking send is that a rendezvous may not be required and message could be delivered asynchronously.

(B) ASYNCHRONOUS Communication

(i) **Advantage**: Time to think about what to say. (time not a constraint)

You can receive message whenever you're free and analyze the content carefully for meaningful reply.

(ii) **Disadvantage**: No immediate response, You may not know if other person has received your message.

U19CS012

b.) Automatic and Explicit Buffering

AUTOMATIC Buffering

Advantage: Automatic Buffering provides a queue with indefinite length, thus ensuring the sender will never have to block while waiting to copy a message.

Disadvantage: There is no specification how automatic buffering will be provided; one scheme may reserve sufficiently large memory where much of memory is wasted.

EXPLICIT Buffering

Advantage: Explicit buffering specifies how large the buffer is.

Disadvantage: In this situation, the sender may be blocked while waiting for available space in queue. However, it is less likely memory will be wasted with explicit buffering.

c.) Send by copy and send by refrence.

SEND by copy

Advantage: send by copy is better for synchronization issues.

Disadvantage: send by copy does not allow the receiver to alter the state of parameter.

SEND by reference:

Advantage: A benefit of send by reference is that it allows the programmer to write a distributed version of a centralized application. Java's RMI (Remote Method Invocation) allows both. However passing a parameter by reference requires declaring the parameter as a remote object as well. Send by reference allows receiver to alter the state of parameter.

Disadvantage: Send by refrence can compromise the security of an organisation because sensitive information can easily be distributed accidently or deliberately.

U19CS012

d) Fixed sized and variable-sized message

Fixed sized messages

Advantage: The implications of this are mostly related to buffering issues. With fixed sized messages, a buffer with specific size can hold a known number of messages.

Disadvantage: With fixed sized messages, the messages are copied from address space of the sender to the address space of the receiving process. [Windows 2000]

VARIABLE sized messages

Advantage: variable-sized message (i.e. Larger messages) use shared memory to pass the message.

Disadvantage: The number of variable-sized messages that can be held by such a buffer is unknown.

2.) With respect to the RPC mechanism, consider the "exactly once" semantics. Does the algorith for implementing this semantic execute correctly even if the ACK message back to the client is lost due to a network problem? Describe the sequence of messages and discuss whether "exactly once" is still preserved.

2.) The "exactly once" semantics ensures that a remote procedure will be executed exactly once and only once. The general algorithm for ensuring this combination~~s~~ an acknowledgement (ACK) scheme combined with timestamps (or some other incremental counter that allows server to distinguish between duplicate messages).

The general strategy is for the client to send the RPC to the server along with timestamp. The client with also start a timeout clock.

Continued → on Next Page ⟹

the client will then wait for one of two occurances:

(1) It will receive an ACK from server indicating that the remote procedure was performed
OR
(2) It will time out.
If the client times out, it assumes the server was unable to perform the remote procedure so the client invokes the RPC a second time, sending a later timestamp.

The client may not receive the ACK for one of two reasons:
(1) The orginal RPC was never received by the server.
OR
(2) The RPC was correctly received - and performed - by the server but the ACK was lost.

In situation (1), the use of ACK's allows the server ultimately to receive and perform the RPC.
In situation (2), the server will receive a duplicate RPC and it will use the timestamp to identify it as a duplicate so as not to perform the RPC a second time.
[Note: server must send second ACK back to the client to inform the client the RPC has been performed]

3.) Palm OS provides no means of concurrent processing. Discussing three major complications that concurrent processing adds to an operating system.

3.) (a) A method of time sharing must be implemented to allow each of several processes to have access to the system.
This method involves the preemption of processes that do not voluntarily give up the CPU (by using a system call, for instance)

and the kernel being reentrant (so more than one process may be executing kernel code concurrently).

b) Processes and system resources must have protections and must be protected from each other. Any given process must be limited in the amount of memory it can use and the operations it can perform on devices like disks.

c) Care must be taken in the kernel to prevent deadlocks between processes, so processes aren't waiting for each other's allocated resources.

4) Describe the actions taken by a kernel to context-switch between processes.

4) The process of context switching between the processes is accomplished by the kernel.

The process is as follows:

a) In reply to clock interrupt, operating system stores the value of the Program Counter and the user Stack Pointer of the presently implementing process, and handovers charge to the kernel clock interrupt handler.

b) The clock interrupt handler hold back the remaining registers, along with other machine* status such the status of the floating point registers in the Process Control Block of the process.

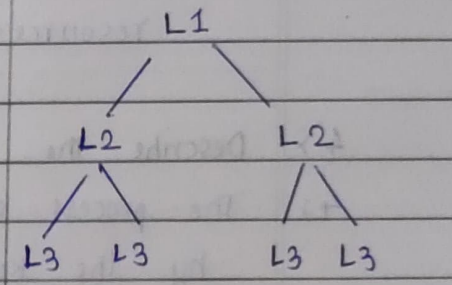
c) The Operating System call upon the scheduler to decide the next process that has to be implemented.

d) The operating system takes the status of the next process from its Process control Block and fix up the registers.

This restore task takes the processor back to the state in which the process was earlier interrupted, implementing in user code with user mode privileges.

5> Including the intial parent process, how many processes are created by the following program.

```
#include <stdio.h>
#include <unistd.h>
int main()
{   /* fork a child process */
    fork(); // Line 1 → Create 1 child process
    /* form another child process */
    fork(); // Line 2 → Create 2 child process
    /* and fork another */
    fork(); // Line 3 → create 4 child process
    return 0;
}
```



[Note: ↑ child processes] per line

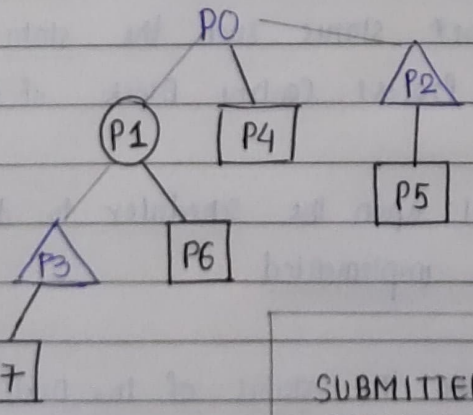
∴ main Process : P0

Processes created by first fork: P1

Processes created by second fork: P2, P3

Processes created by 3rd fork : P4, P5, P6, P7

Ans:



Total 8 Processes = 1 (original Process) + 7 (new child process)

[∵ In General, Total no of processes $= 2^n$; n = number of fork system calls]

here, $n = 3$ ∴ $2^3 = 8$ processes

SUBMITTED BY: BHAGYA VINOD RANA

U19CS012