2.1 The process concept

The process concept

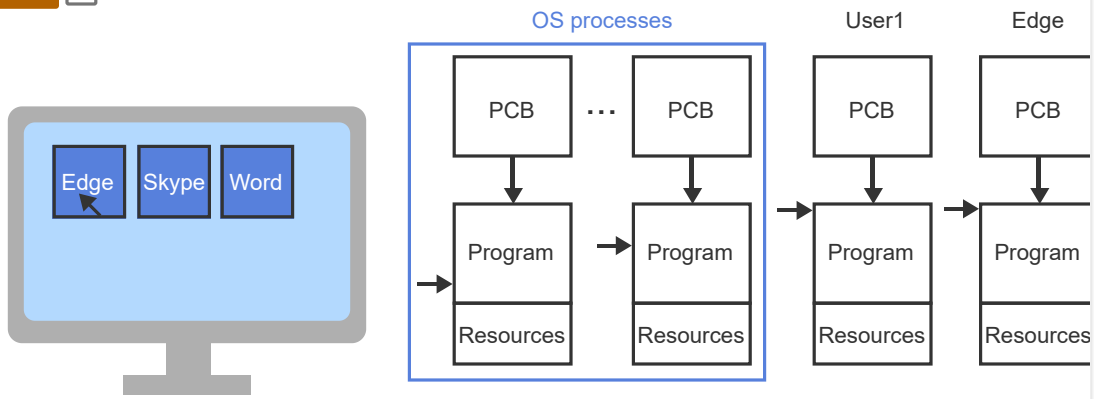
A ***process*** is an instance of a program being executed by an OS. Ex: When a user opens a new application like a web browser or text editor, the OS creates a new process.

The OS itself is organized as a collection of processes.

The OS keeps track of each process using a ***process control block*** (***PCB***): A data structure that holds information for a process, including the current instruction address, the execution stack, the set of resources used by the process, and the program being executed. The PCB is the concrete representation of a process.

PARTICIPATION ACTIVITY 2.1.1: OS creates a new PCB for every new process.

Start 2x speed



Captions

1. Before a user logs in, only OS processes are running. Each process is represented by a PCB, which points to a program and the resources controlled by the process. Horizontal arrows represent program counters.
2. When User1 logs in, a new PCB is created. The PCB points to User1's program, which typically executes the graphical user interface, and to initial resources. The program counter points to the start of the program.
3. User1 is running. The screen shows the icons of different applications.
4. When the user clicks the Microsoft Edge icon, a new PCB for Microsoft Edge is created by the OS.

Feedback?

PARTICIPATION ACTIVITY 2.1.2: Basic principles of a process.

1) An OS uses a PCB to represent a process.
   - (•) True
   - ( ) False

   Correct
   The OS implements every process as a data structure called PCB, the process control block.

2) The PCB is created by the process when execution starts.
   - ( ) True
   - (•)

   Correct
   The PCB is a data structure created and maintained only by the OS, not by the process itself.

False

3) The PCB becomes part of the program being executed by a process.
   - ○ True
   - ◉ False

Correct

The PCB is an independent data structure managed by the OS that is destroyed when the process terminates. The program remains unchanged.

4) Two processes can be executing the same program.
   - ◉ True
   - ○ False

Correct

A program can be executed multiple times simultaneously, each instance being a process. Each PCB has a separate program counter to keep track of each process' execution location in the program.

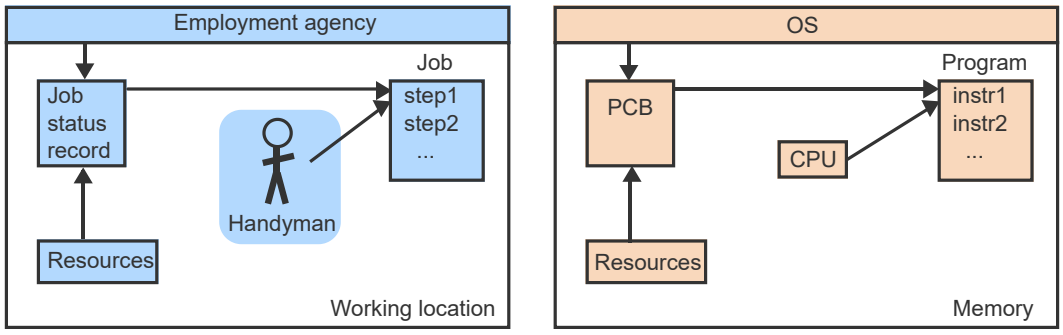
Feedback?

The process concept by analogy

A good way to understand the concept of a process is to consider an analogy from real life.

PARTICIPATION ACTIVITY | 2.1.3: Correspondence between the process concept and the employment agency analogy.

Start ☐ 2x speed



Captions ^

1. An agency employs a handyman to perform various jobs. By analogy, the OS uses the CPU to execute different programs.
2. The handyman performs jobs consisting of multiple steps, analogous to the CPU, which executes programs consisting of multiple instructions.
3. The agency sends the handyman to a location where the job takes place. The work location corresponds to the memory where the program execution takes place.
4. The agency creates a job status record, which is the dynamic representation of the job's progress, analogous to the PCB, which is the representation of the process.
5. The job status record also keeps track of the resources assigned to the handyman in the same way as the PCB keeps track of resources assigned to the process.

Feedback?

PARTICIPATION ACTIVITY | 2.1.4: Understanding the process concept via a student library analogy.

A process can be compared to the act of reading a book checked out by a student from the reserved books section of a library.

Select the definition that matches each term

1) CPU
   - The student corresponds to the ____.

Correct
The student is the active entity performing the task of reading.

2) program
   - The book corresponds to the ____.

Correct
The book contains the words to be read, just like the program contains the instructions to be executed.

3) memory
   - The library reading room corresponds to the ____.

Correct
The reading takes place in the reading room, just like the program execution takes place in the memory.

4) OS
   - The librarian corresponds to the ____.

Correct
The librarian is in charge of signing out the book and keeping track of the loan, just like the OS is in charge of creating the PCB and keeping track of the process.

5) PCB
   - The check-out record corresponds to the ____.

Correct
The check-out record represents and keeps track of the book loan, just like the PCB represents and keeps track of a process.

Reset

Feedback?

Process states and transitions

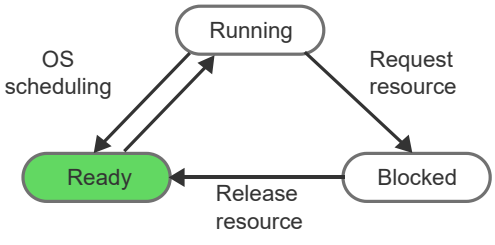
A process is always in one of several possible states. The number and type of states vary between different OSs, but most will implement at least the following 3 basic states:

- A process is in the ***running state*** (or ***running***) when the process has all necessary resources and the CPU is actively executing the program's instructions.
- A process is in the ***ready state*** (or ***ready***) when the process has all necessary resources to run but the CPU is currently unavailable.
- A process is in the ***blocked state*** (or ***blocked***) when the process is waiting on a currently unavailable resource.

A transition from one state to another is performed by the OS but can be caused by the process itself, by some other process, or by the OS.

PARTICIPATION ACTIVITY 2.1.5: Process states and transitions.

1 2 3 4 5 6 2x speed



When some other process releases the resource needed by the blocked process, the OS moves the blocked process back to ready.

Captions

1. A newly created process starts in the ready state.
2. When the OS selects the process for execution, the process's state changes to running.
3. When the OS interrupts the process, for example to let some other process use the CPU, the current process's state changes back to ready.
4. When the OS again restarts the process, the process's state changes back to running.
5. When the process requests a currently unavailable resource, such as a file already open by another user, the process's state changes to blocked.
6. When some other process releases the resource needed by the blocked process, the OS moves the blocked process back to ready.

Feedback?

PARTICIPATION ACTIVITY 2.1.6: Process states and transitions.

1) The transition (ready -> running) of a process p is caused by _____.
   - (•) the OS
   - ( ) the process p itself
   - ( ) some other process

   Correct
   Only the OS can decide when a process should continue to run.

2) The transition (running -> blocked) of a process p is caused by _____.
   - ( ) the OS
   - (•) the process p itself
   - ( ) some other process

   Correct
   p causes the transfer when requesting a currently unavailable resource.

3) The transition (running -> ready) of a process p is caused by _____.
   - (•) the OS
   - ( ) the process p itself
   - ( ) some other process

   Correct
   The OS may wish to let some other process use the CPU.

4) The transition (blocked -> ready) of a process p is caused by _____.
   - ( ) the OS
   - ( ) the process p itself
   - (•) some other process

   Correct
   p is waiting for a resource held by some other process q. The release of the resource by q causes the transition of p back to ready.

Feedback?

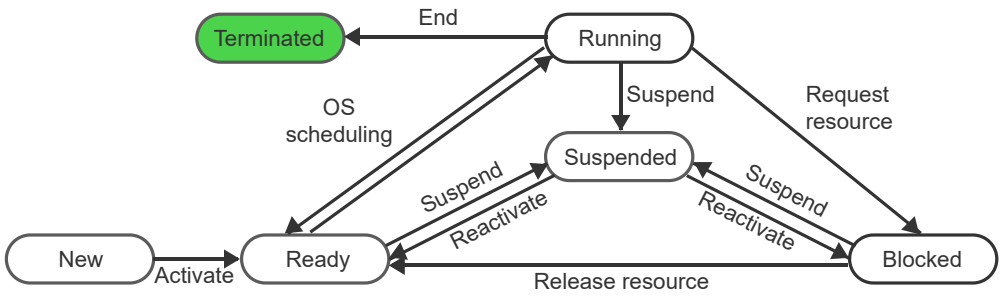
Additional process states and transitions

Some OSs provide a more complex set of states and transitions. The additional states may include the following:

- A newly created process is placed into the ***new state*** before the process is allowed to compete for the CPU. Ex: The OS may want to regulate the number of processes competing for the CPU.
- A process is placed into the ***terminated state*** when execution can no longer continue but before the PCB is deleted. Ex: The OS may want to examine the final state of a process that committed a fatal error.
- A process may be placed into the ***suspended state*** even though the CPU and all resources are available. Ex: The OS may want to stop a process to allow debugging or to regulate performance.

PARTICIPATION ACTIVITY 2.1.7: Additional state transitions.

Start 2x speed



Captions

1. A newly created process starts in the new state.
2. The OS moves the process from new to ready when the process is allowed to compete for the CPU. The transition is called "Activate".
3. From ready, the process can move repeatedly between running, blocked, and ready as before.
4. In addition, the OS can move the process from the running, ready, or blocked state to a suspended state to temporarily prevent the process from making any progress.
5. If the process was suspended in the blocked state, then upon reactivation the process returns to the blocked state.
6. If the suspension occurred in the running or ready state, then upon reactivation the process is returned to the ready state.
7. When a running process finishes all work or causes a fatal error, the process's state changes to terminated. The OS can also perform the transition to terminated from any other state.

Feedback?

PARTICIPATION ACTIVITY 2.1.8: Additional process states and transitions.

1) When a newly created process p is ready to compete for the CPU, p moves itself from the new to the ready state.
   - ○ True
   - ◉ False

   Correct
   The purpose of the new state is to allow the OS to control how many processes are competing for the CPU and thus only the OS can cause a transition from new to ready.

2) When a ready process p wishes to temporarily stop competing for the CPU, p moves itself to the suspended state.
   - ○ True
   - ◉ False

   Correct
   A process in the ready state is not running program code and so cannot perform any actions. Only the OS or p's creator (parent) can move p to the suspended state.

3) The transition of a process p

from running to terminated can be caused by p itself.

- (●) True
- ( ) False

Correct

Completing all work or causing a fatal error causes p to transition from running to terminated.

4) When a blocked process p is suspended and the resource needed by p becomes available, p is moved to the ready state.

- ( ) True
- (●) False

Correct

p is suspended for a reason different from waiting for a resource. Thus p must remain in the suspended state until explicitly reactivated. Only then is the process moved to the ready state.

Feedback?

PARTICIPATION ACTIVITY 2.1.9: Process reactivation.

1) When a process p is moved from the current state (running, ready, or blocked) to the suspended state, then upon reactivation, _____.

- ( ) p must be moved back to the same state where the suspension occurred
- ( ) p can always safely be moved to the ready state
- ( ) p is moved to either the ready or the blocked state

Feedback?

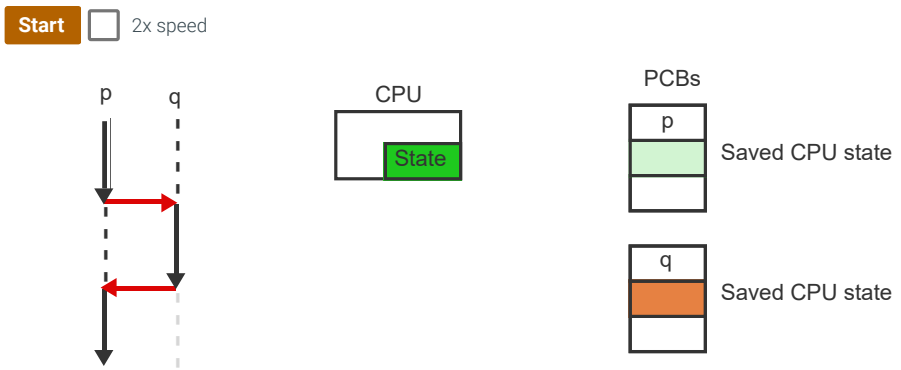
The context switch

The CPU is always running one process at a time. Multiple processes can share a single CPU by taking turns. A ***context switch*** is the transfer of control from one process to another.

Each time one process stops executing to allow another one to resume, the OS must save all information about the stopped process. This information is restored when the process again gets a chance to run. The temporary interruption is fully transparent to the process.

The information that needs to be saved is called the CPU state. The ***CPU state*** consists of all intermediate values held in any CPU registers and hardware flags at the time of the interruption. One of the registers is the program counter, which determines the next instruction to execute after the process is restored.

PARTICIPATION ACTIVITY 2.1.10: Context switches between processes p and q.



Captions

1. Process p is running. The current state of p is in the CPU, while p's PCB contains an out-of-date copy.
2. A hardware interrupt transfers control to the OS, which saves the current CPU state in p's PCB.
3. If the OS chooses process q to continue, the current CPU state is overwritten with q's last state from q's PCB.
4. As q continues running, the saved state in q's PCB gets out of date. q's current state is maintained in the CPU.
5. Another interrupt causes the current CPU state to be saved in q's PCB. The OS then may restart p by restoring p's saved state in the CPU.
6. As p continues running, the saved state in the PCB gets out of date.

Feedback?

PARTICIPATION ACTIVITY 2.1.11: Who causes a context switch.

A context switch for a currently running process p may be caused by _____.

1) process p itself
   - (•) True
   - ( ) False

   Correct
   This context switch may occur when process p requests a currently unavailable resource.

2) the OS
   - (•) True
   - ( ) False

   Correct
   The OS may wish to let another process q use the CPU.

3) some other process q
   - ( ) True
   - (•) False

   Correct
   The OS can take the CPU away from a running process p, or p can give up the CPU by requesting an unavailable resource. But some other process q cannot cause a context switch for process p.

Feedback?

PARTICIPATION ACTIVITY 2.1.12: Current CPU state versus copy of CPU state in PCB.

1) The PCB contains an up-to-date copy of the CPU state of a process p _____ .
   - ( ) when p's state is running
   - (•) when p's state is ready or blocked
   - ( ) at all times

   Correct
   At this point, p's PCB contains an exact copy of the CPU state at the point when p transitioned out of the running state. This CPU state is restored when p transitions back to the running state.

Feedback?

CHALLENGE ACTIVITY 2.1.1: State transitions.

371440.2479476.qx3zqy7

Start

1

Which of the following state transitions are possible?

- [ ] Running -> Suspended
- [ ] Blocked -> Ready
- [ ] Ready -> Blocked
- [ ] Suspended -> Running
- [ ] Running -> Ready
- [ ] Blocked -> Running

2 3 4 5

1 2 3 4 5

Check Next

Feedback?

EXERCISE 2.1.1: State transitions.

Indicate whether each series of state transitions is valid or invalid.

(a) new ➛ ready ➛ suspended ➛ blocked ➛ suspended.

Solution

Invalid. A process that went from ready to suspended must return to ready rather than blocked.

(b) suspended ➛ blocked ➛ suspended ➛ blocked ➛ ready ➛ running.

Solution

Valid. A suspended process goes back to blocked if the suspension occurred in the blocked state. Next, the process may be suspended again and later return back to the blocked state. When the resource becomes available, the process transitions to ready. Finally, from ready, the process may go into running.

(c) running ➛ blocked ➛ ready ➛ blocked ➛ suspended.

Solution

Invalid. A process cannot go from ready to blocked because only a running process can request a resource.

(d) new ➛ ready ➛ running ➛ ready ➛ new.

Solution

Invalid. Only a newly created process is ever placed into the new state.

Feedback?

How was this section? **Provide feedback**