ບົດທີ 3

ແນວຄິດກ່ຽວກັບຂະບວນການ (Process Concept)

ເນື້ອໃນຫຍໍ້

- ສະເໜີເບື້ອງຕົ້ນ
- ສະຖານະພາບຂອງຂະບວນການ (Process States)
- ການບໍລິຫານຈັດການຂະບວນການ (Process Management)
- Interrupts
- ການສື່ສານລະຫວ່າງຂະບວນການ
- ກໍລະນີສຶກສາ: ຂະບວນການຂອງ UNIX

ສະເໜີເບື້ອງຕົ້ນ

- ຄອມພິວເຕີເຮັດວຽກຫຼາຍຢ່າງພ້ອມກັນເຊັ່ນ
  - ແປໂປຣແກຣມທີ່ຂຽນດ້ວຍພາສາລະດັບສູງໄປເປັນພາສາເຄື່ອງ, ສົ່ງຟາຍໄປພິມອອກທາງເຄື່ອງພິມ, ສະແດງໜ້າ ເວບເພດຈ໌, ຫຼິ້ນເພັງ, ຮັບ e-mail
  - ຂະບວນການເຮັດໃຫ້ລະບົບສາມາດເຮັດວຽກ ແລະ ຕິດຕາມບັນດາກິດຈະກຳຕ່າງໆໄປພ້ອມໆກັນໄດ້
  - ຂະບວນການສາມາດປ່ຽນຈາກສະຖານະພາບໜຶ່ງໄປຫາສະຖານະພາບອື່ນ
  - ລະບົບປະຕິບັດການສາມາດປະຕິບັດຕໍ່ຂະບວນການຕ່າງເຊັ່ນ: ສ້າງ, ທຳລາຍ, ຢຸດເຮັດຊົ່ວຄາວ, ສືບຕໍ່ເຮັດວຽກ ແລະ ປຸກໃຫ້ຕື່ນ

ສະເໜີເບື້ອງຕົ້ນ

- ນິຍາມຂະບວນການ
  - ຂະບວນການແມ່ນໂປຣແກຣມທີ່ກຳລັງເຮັດວຽກຢູ່
  - ແຕ່ລະຂະບວນການຈະມີເນື້ອທີ່ໜ່ວຍຄວາມຈຳເປັນຂອງຕົນເອັງ
    - Text region ເປັນບ່ອນເກັບລະຫັດຄຳສັ່ງທີ່ໜ່ວຍປະມວນຜົນໃຊ້ໃນການປະມວນຜົນ
    - Data region ໃຊ້ເກັບຂັ້ນຕາຄ່າຂອງຕົວປ່ຽນ ແລະ ຊຶ່ງໄດ້ຖືກຈັດສັນໃຫ້ຕາມຄວາມຕ້ອງການ
    - Stack region ໃຊ້ເກັບຂັ້ນຕາຄຳສັ່ງ ແລະ ຕົວປ່ຽນພາຍໃນຟັ່ງຊັ່ນໃດໜຶ່ງທີ່ກຳລັງເຮັດວຽກຢູ່

ສະຖານະພາບຂອງຂະບວນການ: ວົງຈອນຊີວິດ

- ຂະບວນການຈະເຮັດວຽກຕາມສະຖານະພາບຕ່າງໆດັ່ງນີ້:
  - *Running* state
    - ຂະບວນການຢູ່ໃນສະຖານະທີ່ກຳລັງເຮັດວຽກຢູ່ໃນໜ່ວຍປະມວນຜົນ
  - *Ready* state
    - ຂະບວນການກຽມພ້ອມທີ່ຈະເຂົ້າເຮັດວຽກຢູ່ໃນໜ່ວຍປະມວນຜົນ ຖ້າວ່າໜ່ວຍປະມວນຜົນຫວ່າງ
  - *Blocked* state
    - ຂະບວນການລໍຖ້າເຮັດການໃດໜຶ່ງເພື່ອຈະເຮັດວຽກຕໍ່ໄປ
- ລະບົບປະຕິບັດການຈະມີລາຍການ *ready* ແລະ ລາຍການ *blocked* ທີ່ອ້າງອີງໄປຫາຂະບວນການທີ່ບໍ່ເຮັດວຽກ

ການບໍລິຫານຈັດການຂະບວນການ

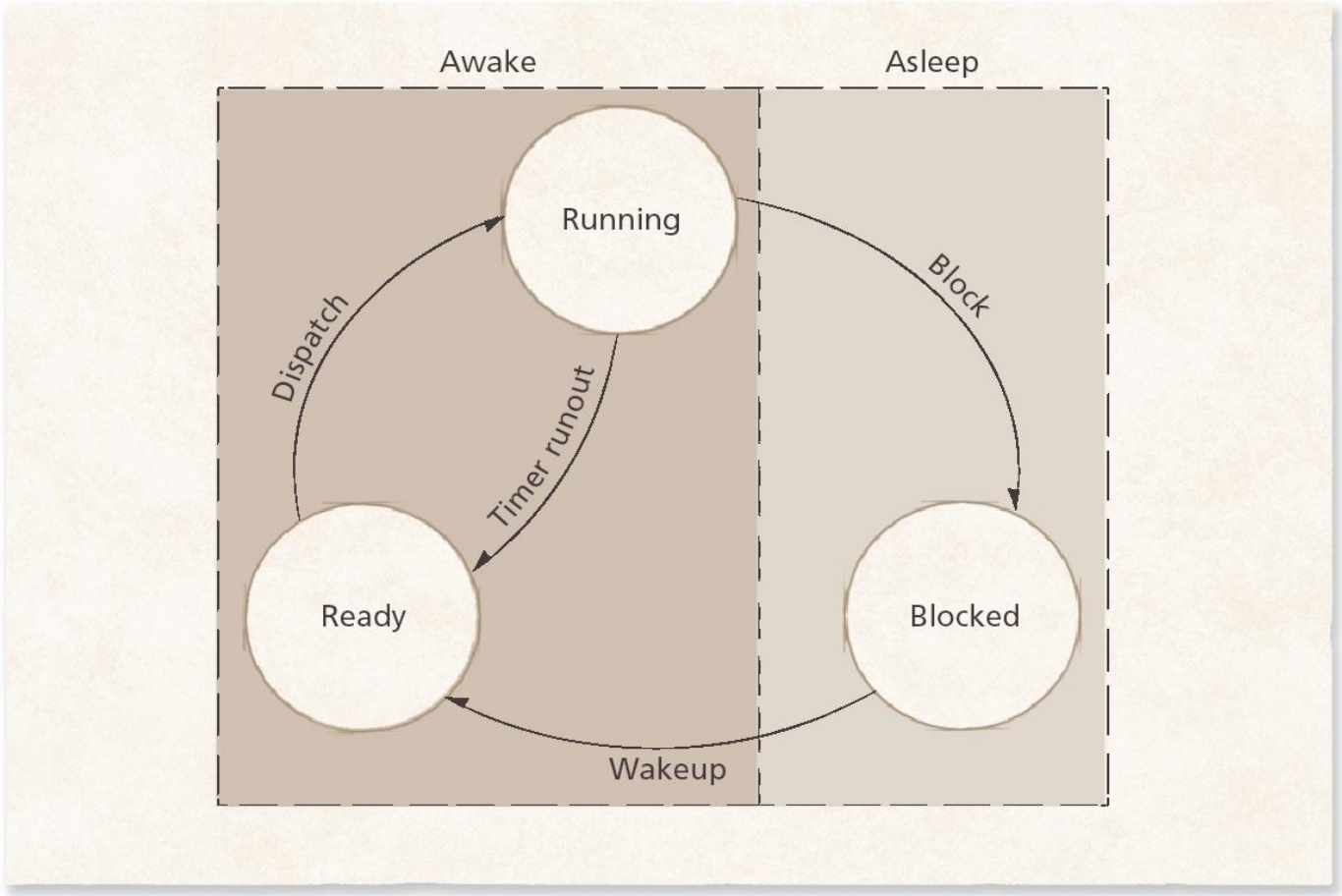
- ລະບົບປະຕິບັດການເປັນຜູ້ຈັດການຂັ້ນພື້ນຖານຕໍ່ກັບຂະບວນການດັ່ງນີ້
  - ສ້າງຂະບວນການ (Creating processes)
  - ທຳລາຍ (Destroying processes)
  - ຢຸດເຮັດວຽກຊົ່ວຄາວ (Suspending processes)
  - ສືບຕໍ່ເຮັດວຽກ (Resuming processes)
  - ປ່ຽນລຳດັບຄວາມສຳຄັນ (Changing a process's priority)
  - ກັນຂະບວນການ (Blocking processes)
  - ປຸກໃຫ້ຕື່ນ (Waking up processes)
  - ມອບໜ່ວຍປະມວນຜົນໃຫ້ (Dispatching processes)
  - ຈັດການການສື່ສານ (Interprocess communication (IPC))

ການບໍລິຫານຈັດການຂະບວນການ

- ສະຖານະພາບ ແລະ ການປ່ຽນສະຖານະພາບຂອງຂະບວນການ
  - ສະຖານະພາບຂອງຂະບວນການ
    - ການມອບໝ່ວຍປະມວນຜົນໃຫ້ແກ່ຂະບວນການທຳອິດໃນຄິວກຽມພ້ອມເອີ້ນວ່າ dispatching ຊຶ່ງຜູ້ມອບເອີ້ນວ່າ dispatcher
    - ລະບົບປະຕິບັດການຈະກຳໜົດຊ່ວງເວລາໃນການມອບໝ່ວຍປະມວນຜົນໃຫ້ແກ່ຂະບວນການຊຶ່ງເອີ້ນວ່າ Time slice, Time quantum
  - **ການປ່ຽນສະຖານະພາບຂອງຂະບວນການຈະມີ 4 ກໍລະນີ**
    - ເມື່ອຂະບວນການໃດໜຶ່ງຖືກມອບໝ່ວຍປະມວນຜົນໃຫ້ມັນຈະປ່ຽນສະຖານະພາບຈາກ *ready* ໄປເປັນ *running*
    - ເມື່ອໝົດເວລາທີ່ກຳໜົດໃຫ້ ມັນຈະປ່ຽນຈາກສະຖານະ *running* ໄປເປັນ *ready*
    - ເມື່ອຂະບວນການລໍຖ້າເຫດການໃດໜຶ່ງ ມັນຈະປ່ຽນສະຖານະຈາກ *running* ໄປເປັນ *blocked*
    - ເມື່ອໄດ້ຮັບເຫດການ ມັນຈະປ່ຽນສະຖານນະຈາກ *blocked* ໄປເປັນ *ready*

ການບໍລິຫານຈັດການຂະບວນການ

- ສະຖານະພາບ ແລະ ການປ່ຽນສະຖານະພາບຂອງຂະບວນການ



ການບໍລິຫານຈັດການຂະບວນການ

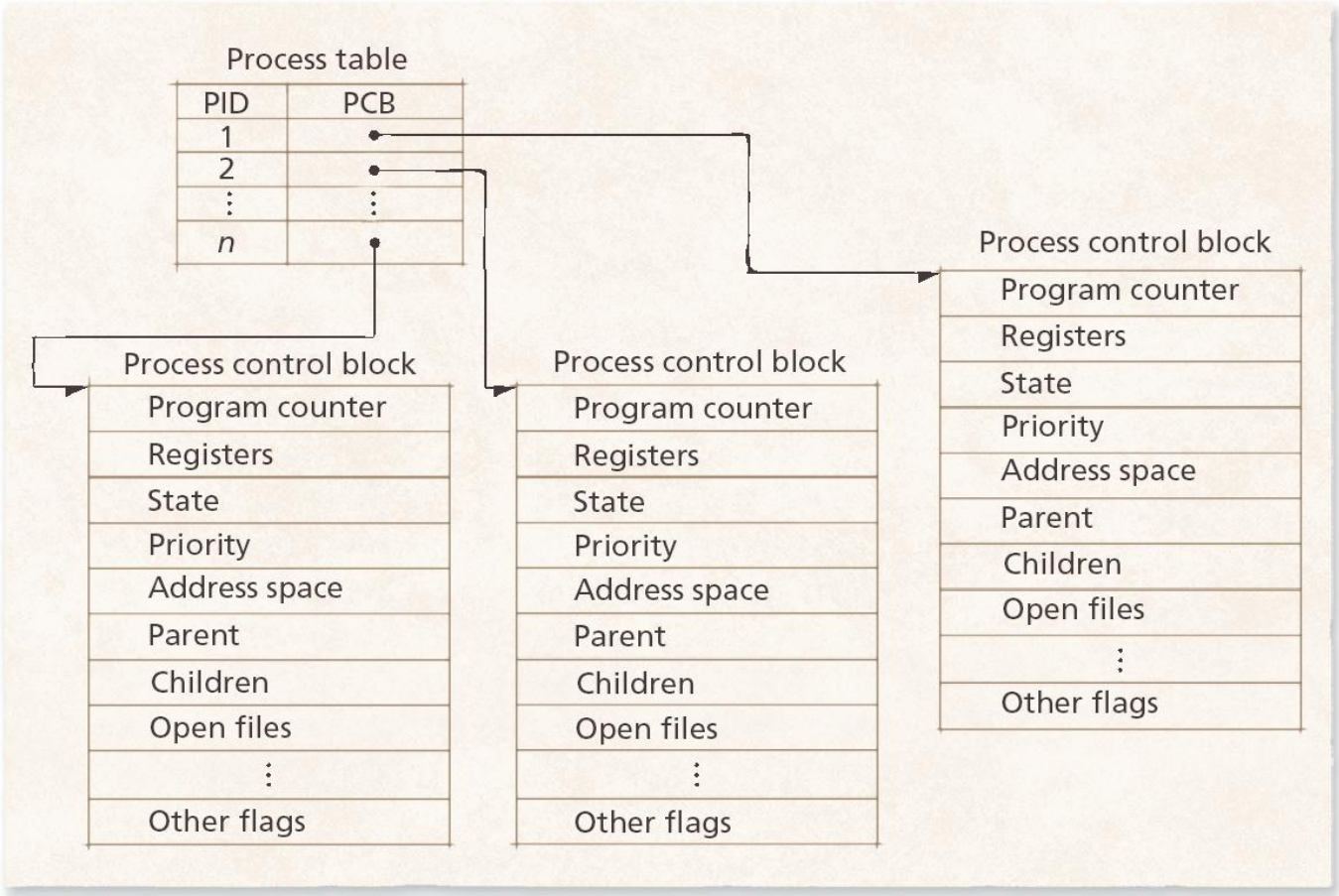
- ກ່ອງຄວບຄຸມຂະບວນການ (PCB)
  - ເປັນບ່ອນເກັບຂໍ້ມູນຂອງຂະບວນການທີ່ລະບົບປະຕິບັດການຕ້ອງການ ເພື່ອໃຊ້ບໍລິຫານຈັດການຂະບວນການ ໂດຍປົກກະຕິຈະມີຂໍ້ມູນດັ່ງນີ້
    - Process identification number (PID)
    - Process state
    - Program counter
    - Scheduling priority
    - Credentials
    - A pointer to the process's parent process
    - Pointers to the process's child processes
    - Pointers to locate the process's data and instructions in memory
    - Pointers to allocated resources (such as file)

ການບໍລິຫານຈັດການຂະບວນການ

- ກ່ອງຄວບຄຸມຂະບວນການ
  - ຕາຕະລາງເກັບກຳຂະບວນການ (Process Table)
    - ເປັນຕາຕະລາງເກັບກຳ ກ່ອງຄວບຄຸມຂະບວນການ (PCB)ຂອງບັນດາຂະບວນການ
    - ເຮັດໃຫ້ງ່າຍຕໍ່ການເຂົ້າໃຊ້ກ່ອງຄວບຄຸມຂະບວນການ
    - ເມື່ອຂະບວນການສິ້ນສຸດການເຮັດວຽກ ລະບົບປະຕິບັດການຈະລຶບຂະບວນການອອກຈາກຕາຕະລາງ ກູ້ຄືນບັນດາຊັບພະຍາກອນທີ່ຂະບວນການນັ້ນໃຊ້ຢູ່

ການບໍລິຫານຈັດການຂະບວນການ

- ກ່ອງຄວບຄຸມຂະບວນການ

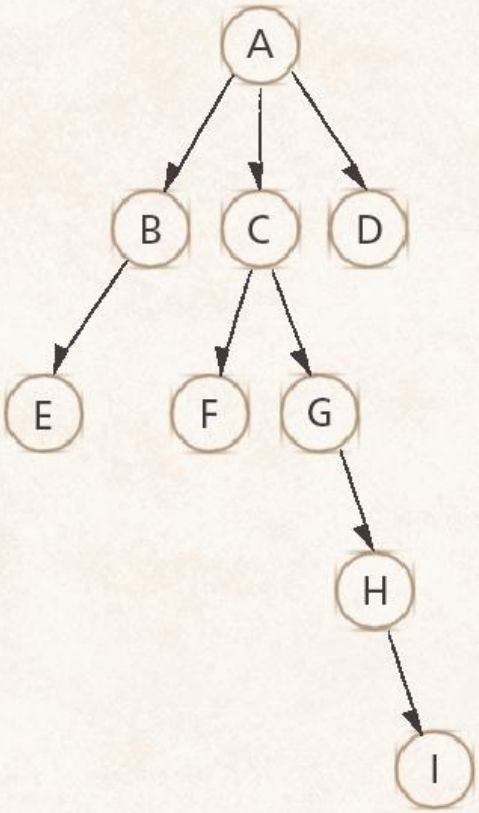


ການບໍລິຫານຈັດການຂະບວນການ

- ການດຳເນີນການຂອງຂະບວນການ
  - ຂະບວນອາດຈະສ້າງຂະບວນລູກອອກມາຫຼາຍອັນ
    - ຂະບວນການທີ່ສ້າງຂະບວນການອື່ນອອກມາເອີ້ນວ່າຂະບວນການພໍ່ແມ່
    - ຂະບວນການທີ່ຖືກສ້າງມາຈາກຂະບວນການອື່ນເອີ້ນວ່າຂະບວນການລູກ
    - ຂະບວນການລູກແຕ່ລະອັນຈະເກີດຈາກພໍ່ແມ່ດຽວເທົ່ານັ້ນ
    - ເມື່ອຂະບວນການທີ່ເປັນພໍ່ແມ່ຖືກທຳລາຍ ລະບົບປະຕິບັດການຈະປະຕິບັດຢ່າງໃດໜຶ່ງດັ່ງນີ້:
      - ທຳລາຍຂະບວນການລູກທັງໝົດຂອງພໍ່ແມ່ດັ່ງກ່າວ
      - ປ່ອຍໃຫ້ຂະບວນການລູກສືບຕໍ່ເຮັດວຽກຕໍ່ໄປໂດຍບໍ່ຂຶ້ນກັບພໍ່ແມ່ຂອງມັນ

ການບໍລິຫານຈັດການຂະບວນການ

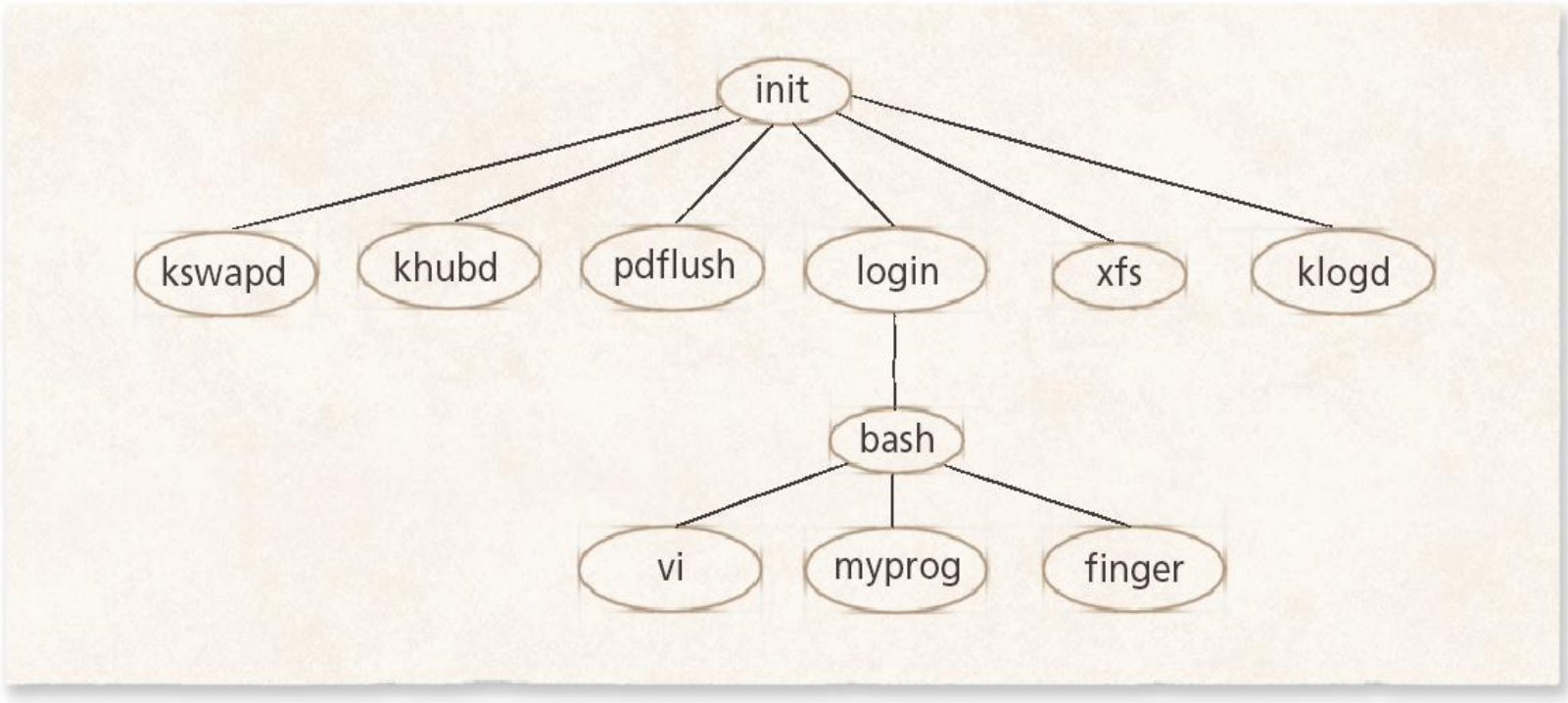
- ການດຳເນີນການຂອງຂະບວນການ



ລຳດັບຊັ້ນການສ້າງຂະບວນການລູກຂອງຂະບວນການ

ການບໍລິຫານຈັດການຂະບວນການ

- ການດຳເນີນການຂອງຂະບວນການ



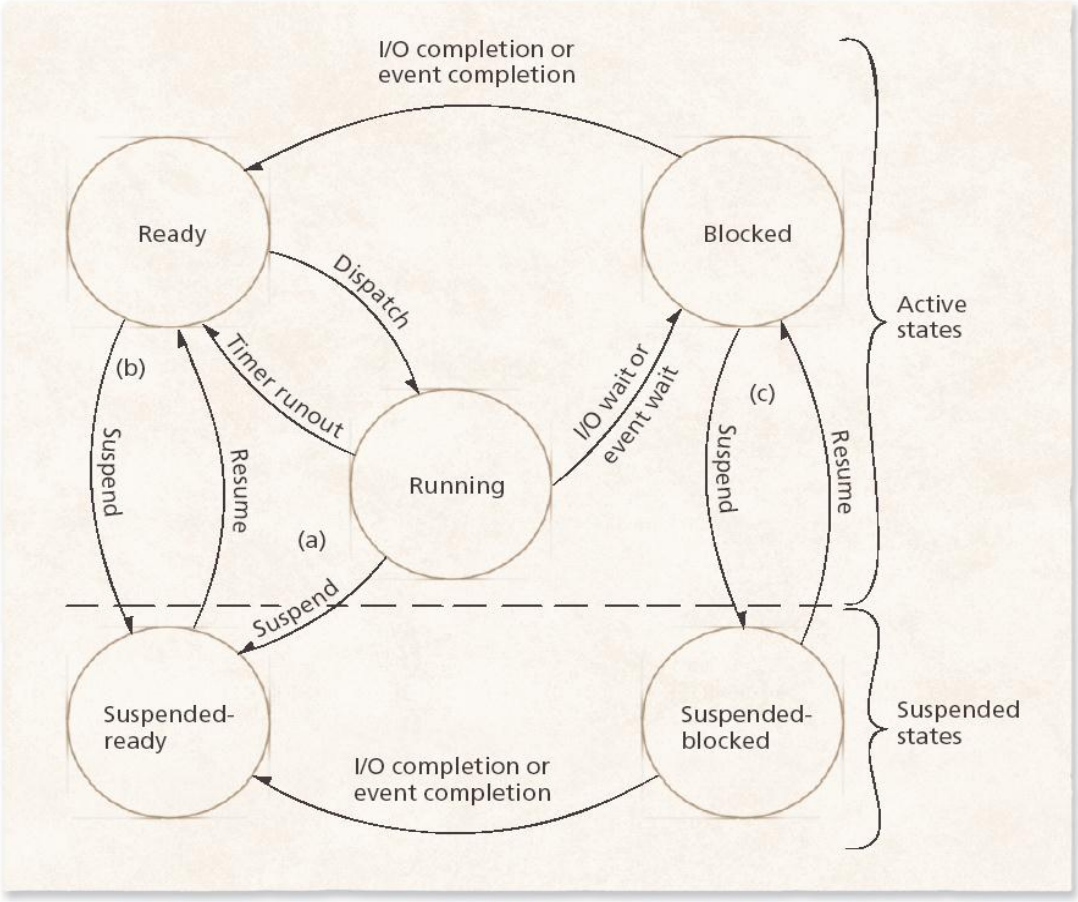
ລຳດັບຊັ້ນການສ້າງຂະບວນການລູກຂອງຂະບວນການຂອງ Linux

ການບໍລິຫານຈັດການຂະບວນການ

- ການຢຸດຊົ່ວຄາວ ແລະ ການສືບຕໍ່ປະຕິບັດ
  - ການຢຸດຂະບວນການຊົ່ວຄາວ (Suspending a process)
    - ເປັນການເອົາຂະບວນການອອກຈາກການເຮັດວຽກໃນໜ່ວຍປະມວນຜົນໂດຍບໍ່ມີກຳນົດ ແຕ່ວ່າຍັງບໍ່ທັນໄດ້ທຳລາຍມັນເທື່ອ
    - ການຢຸດຊົ່ວຄາວຂອງຂະບວນການໃດໜຶ່ງອາດຈະເຮັດໂດຍຂະບວນການຂະບວນການທີ່ຢຸດຊົ່ວ ຫຼື ຂະບວນການອື່ນໆ
    - ມີປະໂຫຍດຕໍ່ການກວດສອບໄພຸກຄາມລະບົບຄວາມປອດໄພ ແລະ ການການກວດສອບຫາຂໍ້ບົກພ່ອງຂອງຊອບແວຣ໌
    - ການເຮັດໃຫ້ຂະບວນການທີ່ຢຸດຊົ່ວຄາວກັບມາເຮັດວຽກຄືນໃໝ່ຕ້ອງແມ່ນຂະບວນການອື່ນເປັນຜູ້ເຮັດ
    - ມີ 2 ສະຖານະພາບ:
      - *suspendedready*
      - *suspendedblocked*

ການບໍລິຫານຈັດການຂະບວນການ

- ການຢຸດຊົ່ວຄາວ ແລະ ການສືບຕໍ່ປະຕິບັດ

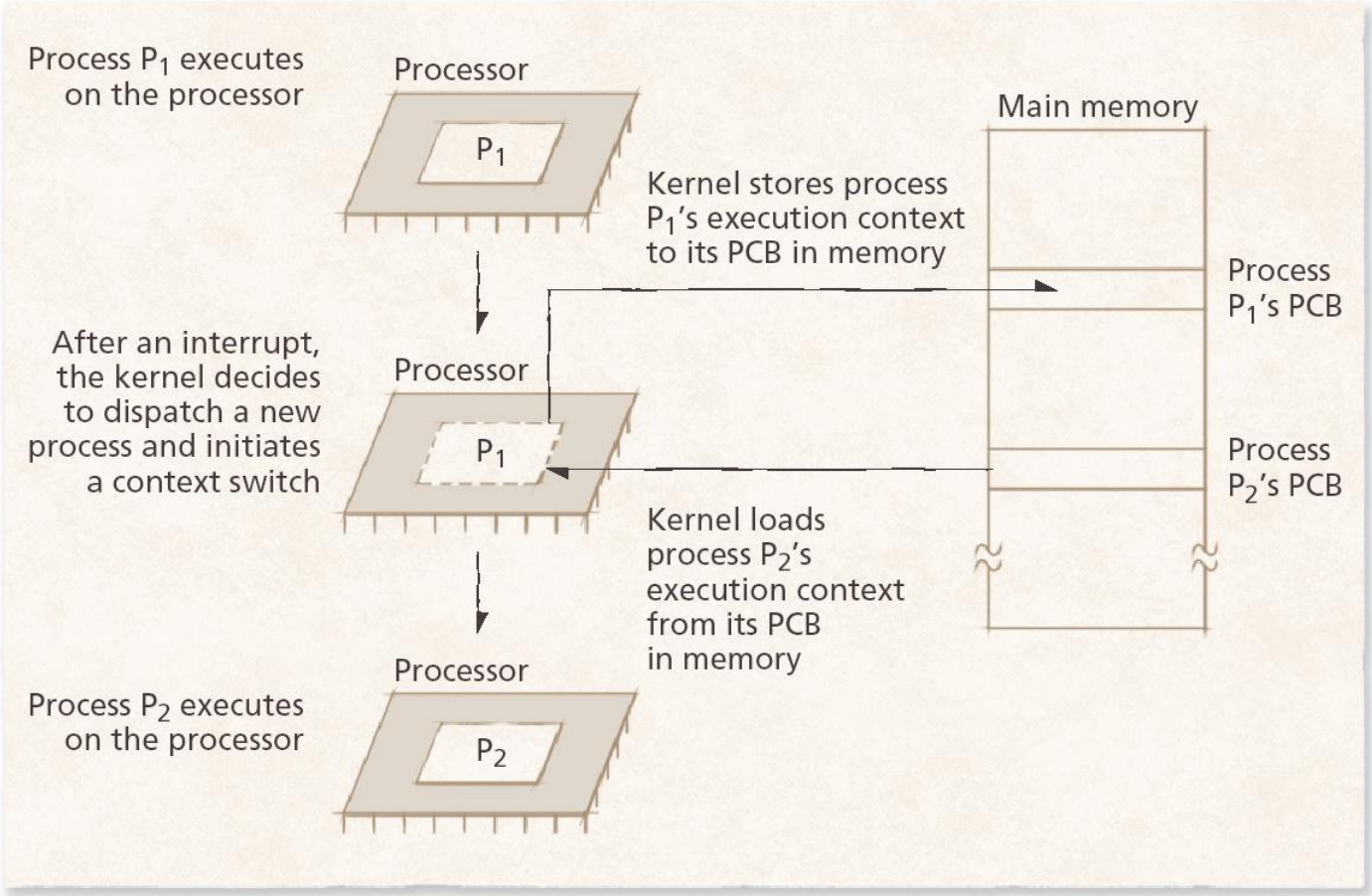


ການບໍລິຫານຈັດການຂະບວນການ

- ການສັບປ່ຽນຂະບວນການເຂົ້າອອກ(Context Switching)
  - ເຮັດໂດຍລະບົບປະຕິບັດການເພື່ອຢຸດການເຮັດວຽກຂອງຂະບວນການທີ່ກຳລັງປະຕິບັດການຢູ່ ແລະ ເອົາຂະບວນການໃໝ່ທີ່ກຳລັງລໍຖ້າຢູ່ໃນຄິວກຽມພ້ອມເຂົ້າມາປະມວນຜົນ
  - ບັນທຶກສະຖານະພາບຂອງຂະບວນການໄວ້ໃນກ່ອງຄວບຄຸມຂະບວນການກ່ອນຈະຢຸດມັນ
  - ໂລດເອົາສະຖານະຂອງຂະບວນການທີ່ກຽມພ້ອມຢູ່ທາງໜ້າຂອງຄິວກຽມພ້ອມ
  - ບໍ່ໃຫ້ໜ່ວຍປະມວນຜົນປະຕິບັດການຄຳນວນບາງຢ່າງ
    - ລະບົບປະຕິບັດການຈະເຮັດຊັບປ່ຽນໃຫ້ໄວທີ່ສຸດເທົ່າທີ່ເປັນໄປໄດ້
  - ໃນບາງລະບົບ ແມ່ນເຮັເດຍຮາດແວຣ໌

ການບໍລິຫານຈັດການຂະບວນການ

- ການສັບປ່ຽນຂະບວນການເຂົ້າອອກ(Context Switching)



Interrupts

- ເປັນສັນຍານຂັດຈັງຫວະຈາກຮາດແວຣ໌ເພື່ອສັ່ງໃຫ້ຊອບແວຣ໌ປະຕິບັດການໃດໜຶ່ງ
- ການຂັດຈັງຫວະອາດຈະສ້າງຈາກການເຮັດວຽກຂອງຂະບວນການໃດໜຶ່ງ
  - ສັນຍານຂັດຈັງຫວະປະເພດນີ້ເອີ້ນວ່າ trap ຊຶ່ງເກີດຈາກການເຮັດວຽກຂອງຂະບວນການ
  - ເຊັ່ນ: ການຫານຈຳນວນໜຶ່ງດ້ວຍສູນ, ອ້າງອີງຫາໜ່ວຍຄວາມຈຳທີ່ບໍ່ໄດ້ຮັບອະນຸຍາດ
- ອາດຈະສ້າງຈາກເຫດການໃດໜຶ່ງທີ່ກ່ຽວຂ້ອງຫຼືບໍ່ກ່ຽວຂອງກັບຂະບວນການທີ່ກຳລັງເຮັດວຽກຢູ່ເຊັ່ນ: ການກົດແປ້ນພິມຄີໃດໜຶ່ງ

Interrupts

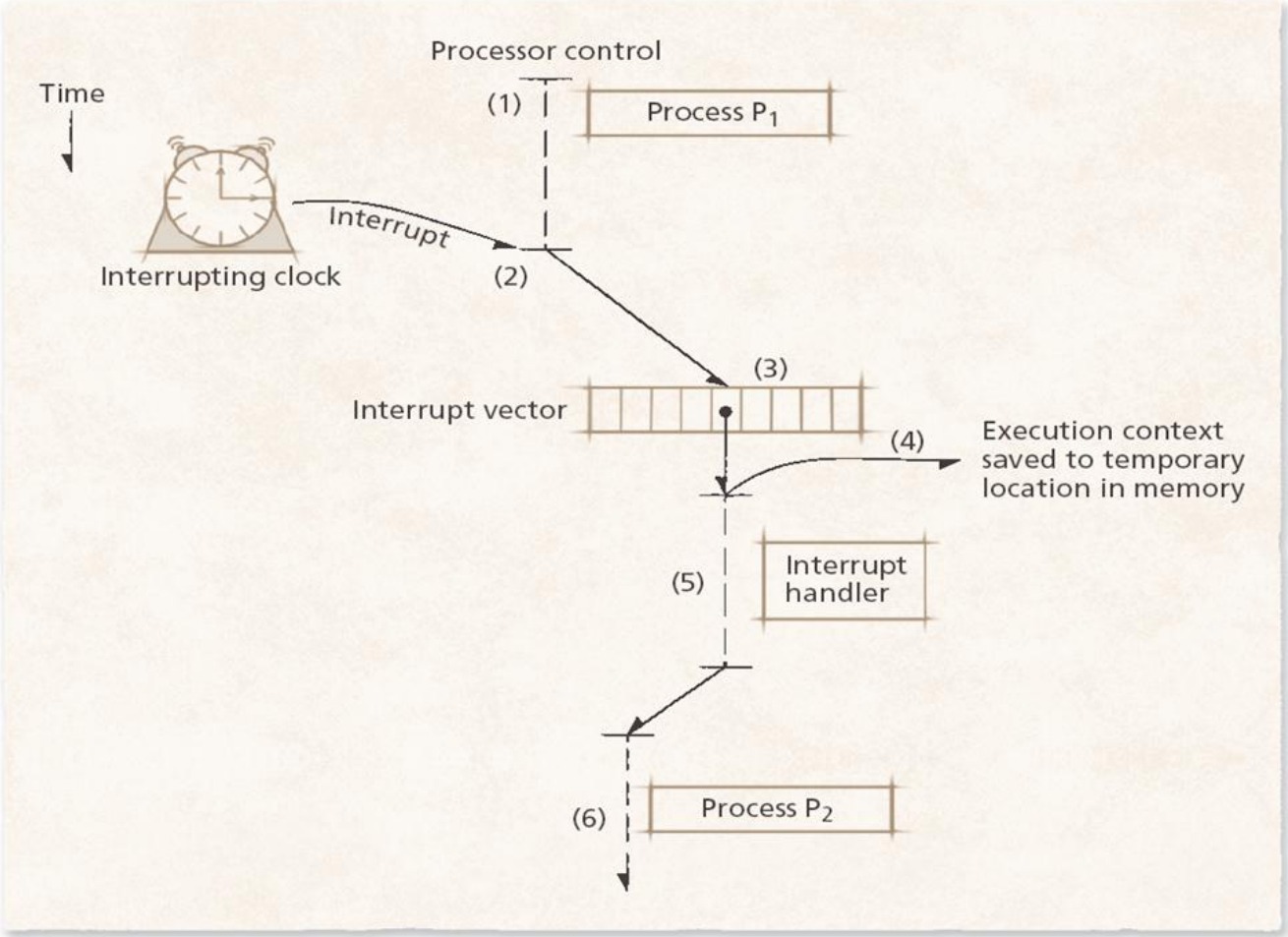
- ການສ້າງການຂັດຈັງຫວະຈະບໍ່ສິ້ນເປືອງ
- ວິທີການໜຶ່ງໃນການສັ່ງໃຫ້ຊອບແວຣ໌ປະຕິບັດການອັນໃດໜຶ່ງ ແມ່ນ Polling
  - ເປັນວິທີການແບບເກົ່າ
  - ໜ່ວຍປະມວນຜົນຈະກວດສອບສະຖານະພາບຂອງແຕ່ລະອຸປະກອນຫຼັງຈາກປະຕິບັດຄຳສັ່ງແຕ່ລະຄຳສັ່ງ
  - ສະຫຼັບຊັບຊ້ອນ ແລະ ສິ້ນເປືອງ

Interrupts

- ການຈັດການກັບ Interrupt
  - ຫຼັງຈາກໄດ້ຮັບສັນຍານຂັດຈັງຫວະ ໜ່ວຍປະມວນຜົນຈະຢຸດການເຮັດວຽກປະຈຸບັນໄວ້ຊົ່ວຄາວ
  - ໜ່ວຍປະມວນຜົນເຮັດວຽກຕາມຄຳສັ່ງຂັດຈັງຫວະ
  - Interrupt handler ຈະເປັນຕົວກຳໜົດວ່າຈະປະຕິບັດຕໍ່ການຂັດຈັງຫວະນັ້ນແນວໃດ
  - Interrupt handlers ຈະຖືກເກັບໄວ້ໃນ array ຂອງ pointers ທີ່ຊື່ວ່າ interrupt vector
  - ຫຼັງຈາກປະຕິບັດຕໍ່ສັນຍານຂັດຈັງຫວະແລ້ວ ໜ່ວຍປະມວນຜົນຈະກັບມາເຮັດວຽກທີ່ໂຈະໄວ້ ຫຼື ເຮັດວຽກຕໍ່ໄປ

Interrupts

- ภาพจัดภาพกับ Interrupt



Interrupts

- Interrupt Classes
  - Interrupt ແຕ່ລະປະເພດຈະຂຶ້ນຢູ່ກັບສະຖາປັດຕະຍະກຳທີ່ເລືອກໃຊ້
  - ການຂັດຈັງຫວະມີ 2 ປະເພດ
    - Interrupts
      - ເປັນການແຈ້ງໃຫ້ໜ່ວຍປະມວນຜົນຮູ້ຈັກວ່າມີຂະເຫດການອັນໃດໜຶ່ງເກີດຂຶ້ນ ຫຼື ສະຖານະພາບຂອງອຸປະກອນໃດໜຶ່ງໄດ້ປ່ຽນແປງໄປ
      - ສ້າງໂດຍອຸປະກອນຕ່າງໆ
    - Exceptions
      - ສະແດງໃຫ້ເຫັນວ່າມີຂໍ້ຜິດພາດເກີຂຶ້ນໃນຮາດແວຣ໌ໃດໜຶ່ງ ຫຼື ເກີດຈາກຂອບແວຣ໌ໃດໜຶ່ງ
      - ແບ່ງອອກເປັນ faults, traps ຫຼື aborts

Interrupts

Interrupt Classes

Interrupt Type	Description of Interrupts in Each Type
I/O	These are initiated by the input/output hardware. They notify a processor that the status of a channel or device has changed. I/O interrupts are caused when an I/O operation completes, for example.
Timer	A system may contain devices that generate interrupts periodically. These interrupts can be used for tasks such as timekeeping and performance monitoring. Timers also enable the operating system to determine if a process's quantum has expired.
Interprocessor interrupts	These interrupts allow one processor to send a message to another in a multiprocessor system.

Interrupts

Interrupt Classes

Exception Class	Description of Exceptions in Each Class
Fault	These are caused by a wide range of problems that may occur as a program's machine-language instructions are executed. These problems include division by zero, data (being operated upon) in the wrong format, attempt to execute an invalid operation code, attempt to reference a memory location beyond the limits of real memory, attempt by a user process to execute a privileged instruction and attempt to reference a protected resource.
Trap	These are generated by exceptions such as overflow (when the value stored by a register exceeds the capacity of the register) and when program control reaches a breakpoint in code.
Abort	This occurs when the processor detects an error from which a process cannot recover. For example, when an exception-handling routine itself causes an exception, the processor may not be able to handle both errors sequentially. This is called a double-fault exception, which terminates the process that initiated it.

ການສື່ສານລະຫວ່າງຂະບວນການ

- ລະບົບປະຕິບັດການສ່ວນຫຼາຍແມ່ນມີເທັກນິກສຳຫຼັບການສື່ສານລະຫວ່າງຂະບວນການ (IPC)
- ຂະບວນການໃດໜຶ່ງຈະຕ້ອງສື່ສານກັບຂະບວນການອື່ນໃນລະບົບ multiprogrammed ແລະ ລະບົບເຄືອຂ່າຍ
  - ຕົວຢ່າງ: Web browser ໄປເອີ້ນເອົາຂໍ້ມູນຈາກ server ໃດໜຶ່ງ
- ມັນເປັນສິ່ງຈຳເປັນສຳຫຼັບຂະບວນການທີ່ຈະຕ້ອງເຮັດວຽກປະສານງານກັນເພື່ອບັນລຸເປົ້າໝາຍໃດໜຶ່ງ

ການສື່ສານລະຫວ່າງຂະບວນການ

- Signals
  - ເປັນສັນຍານຂັດຈັງຫວะຈາກຊອບແວຣ໌ໃດໜຶ່ງທີ່ແຈ້ງໃຫ້ຂະບວນການຮູ້ຈັກວ່າມີເຫດການໃດໜຶ່ງເກີດຂຶ້ນ
  - ບໍ່ອະນຸຍາດໃຫ້ມີການແລກປ່ຽນຂໍ້ມູນລະຫວ່າງຂະບວນການ
  - ຂະບວນການອາດຈະຮັບ, ບໍ່ຮັບ ຫຼື ກັນສັນຍານດັ່ງກ່າວ
- Message Passing
  - ອາດຈະສົ່ງໃນທິດທາງດຽວ ຫຼື ສອງທິດທາງ
  - ເປັນແບບ blocking/nonblocking
  - ວິທີທີ່ເປັນທີ່ນິຍົມແມ່ນ pipe ຊຶ່ງເປັນເນື້ອທີ່ຂອງໜ່ວຍຄວາມຈຳທີ່ຕ້ອງກັນໂດຍ OS ທີ່ໃຊ້ເປັນ buffer ເພື່ອໃຊ້ແລກປ່ຽນຂໍ້ມູນລະຫວ່າງຂະບວນການ

ການສື່ສານລະຫວ່າງຂະບວນການ

- IPC ໃນລະບົບກະຈາຍ (distributed systems)
  - ການຂົນສົ່ງຂໍ້ມູນອາດຈະມີຂໍ້ບົກພ່ອງ ຫຼື ສູນເສຍ
  - ຕ້ອງໃຫ້ຝ່າຍຮັບແຈ້ງບອກຝ່າຍສົ່ງຖ້າໄດ້ຮັບຂໍ້ມູນຄົບຖ້ວນແລ້ວ
  - ຖ້າຝ່າຍສົ່ງບໍ່ໄດ້ຮັບສັນຍານແຈ້ງບອກຈະຕ້ອງໃຫ້ມີສັນຍານໝົດເວລາ ເພື່ອຕັດການເຊື່ອມຕໍ່
  - ການຕັ້ງຊື່ທີ່ບໍ່ຊັດເຈນຈະເຮັດໃຫ້ການອ້າງອີງຂໍ້ມູນຜິດພາດ
  - ກ່ຽວກັບເລື່ອງຄວາມປອດໄພເປັນເລື່ອງສຳຄັນ ຕ້ອງໃຫ້ໃຊ້ຫຼັກການ authentication

ກໍລະນີສຶກສາ: ຂະບວນການຂອງ UNIX

- UNIX processes
  - ທຸກຂະບວນການຈະໄດ້ຮັບທີ່ຢູ່ຂອງໜ່ວຍຄວາມຈຳຊຸດໜຶ່ງຊຶ່ງເອີ້ນວ່າ virtual address space
  - PCB ຂອງຂະບວນການໃດໜຶ່ງແມ່ນຖືກດູແລຮັກສາໂດຍ kernel ຢູ່ໃນບ່ອນປອດໄພ ຊຶ່ງຂະບວນການຂອງຜູ້ໃຊ້ບໍ່ສາມາດເຂົ້າໄປໃຊ້ໄດ້
  - UNIX PCB ບັນຈຸ:
    - ຄ່າ registers ຂອງໜ່ວຍປະມວນຜົນ
    - PID
    - The program counter
    - The system stack
  - ທຸກຂະບວນການໄດ້ຖືກບັນທຶກໄວ້ໃນຕາຕະລາງຂະບວນການ

ກໍລະນີສຶກສາ: ຂະບວນການຂອງ UNIX

- UNIX processes
  - ທຸກຂະບວນການສື່ສານກັບລະບົບປະຕິບັດການຜ່ານທາງ system calls
  - ຂະບວນການໃດໜຶ່ງສາມາດສ້າງຂະບວນການລູກໂດຍໃຊ້ຄຳສັ່ງ fork system call, ຊຶ່ງສຳເນົາຮູບແບບແລະຂໍ້ມູນຈາກພໍ່ແມ່ມັນ
    - Child receives a copy of the parent's resources as well
  - ຄ່າລຳດັບຄວາມສຳຄັນຢູ່ລະຫວ່າງ -20 ຫາ 19 (inclusive)
    - ຄ່າຕ່ຳກ່ວາຈະມີລຳດັບຄວາມສຳຄັນສູງກ່ວາ
  - UNIX ໃຊ້ເທັກນິກ IPC, ເຊັ່ນ pipes, ເພື່ອອະນຸຍາດໃຫ້ບັນດາຂະບວນການທີ່ບໍ່ກ່ຽວຂ້ອງກັນສົ່ງຂໍ້ມູນຫາກັນ

ກໍລະນີສຶກສາ: ຂະບວນການຂອງ UNIX

- UNIX system calls

System Call	Description
fork	Spawns a child process and allocates to that process a copy of its parent's resources.
exec	Loads a process's instructions and data into its address space from a file.
wait	Causes the calling process to block until its child process has terminated.
signal	Allows a process to specify a signal handler for a particular signal type.
exit	Terminates the calling process.
nice	Modifies a process's scheduling priority.