Одеська Політехніка
Інститут комп`ютерних систем
Кафедра інформаційних систем
Дисципліна «Операційні системи»

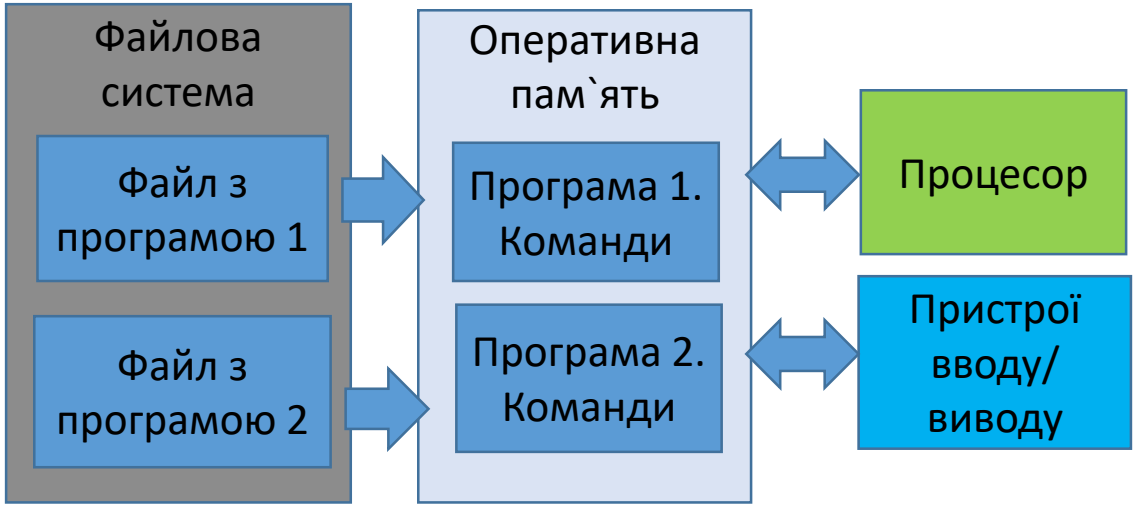
Тема 2: Керування процесами
Лекція 7: Основи операційного керування процесами в Unix-подібних ОС

Олександр А. Блажко,
доцент кафедри інформаційних систем,
E-mail: blazhko@ieee.org
Telegram-канал: t.me/Operating_Systems_IS

Одеса, 17 квітня 2023 року

Основні типи багатозадачності

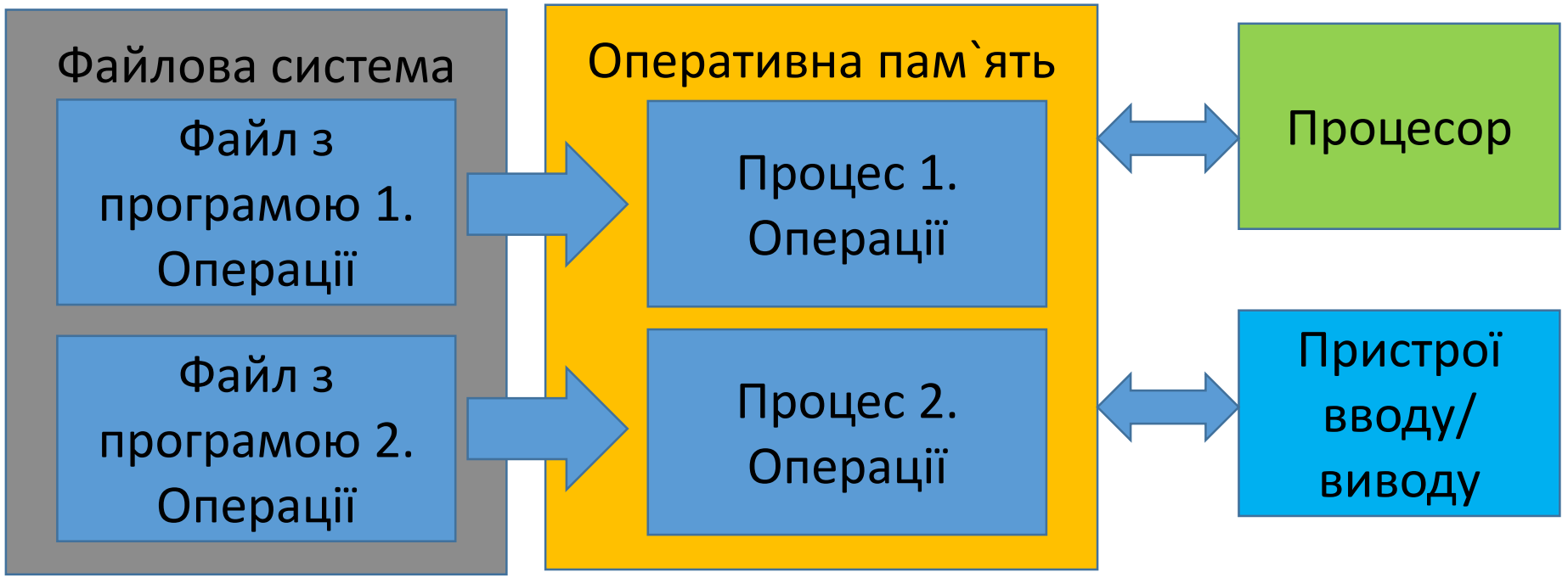
- Компонентна багатозадачність: декілька програм одночасно використовують різні апаратні компоненти (пристрої вводу/виводу, процесор), але лише одна програма використовує процесор



- Інтерактивні програми активно використовують пристрої вводу/виводу під час взаємодії з користувачем
- Невитісняюча багатозадачність (*Non-preemptive multitasking*): кожна програма за бажанням користувача запускається на процесорі та звільняє його за власним бажанням або за бажанням користувача
- Витісняюча багатозадачність (*Preemptive multitasking*): кожна програма виконується на процесорі лише обмежений час, ОС слідкує за часом та періодично перериває роботу програми, надаючи процесор іншій

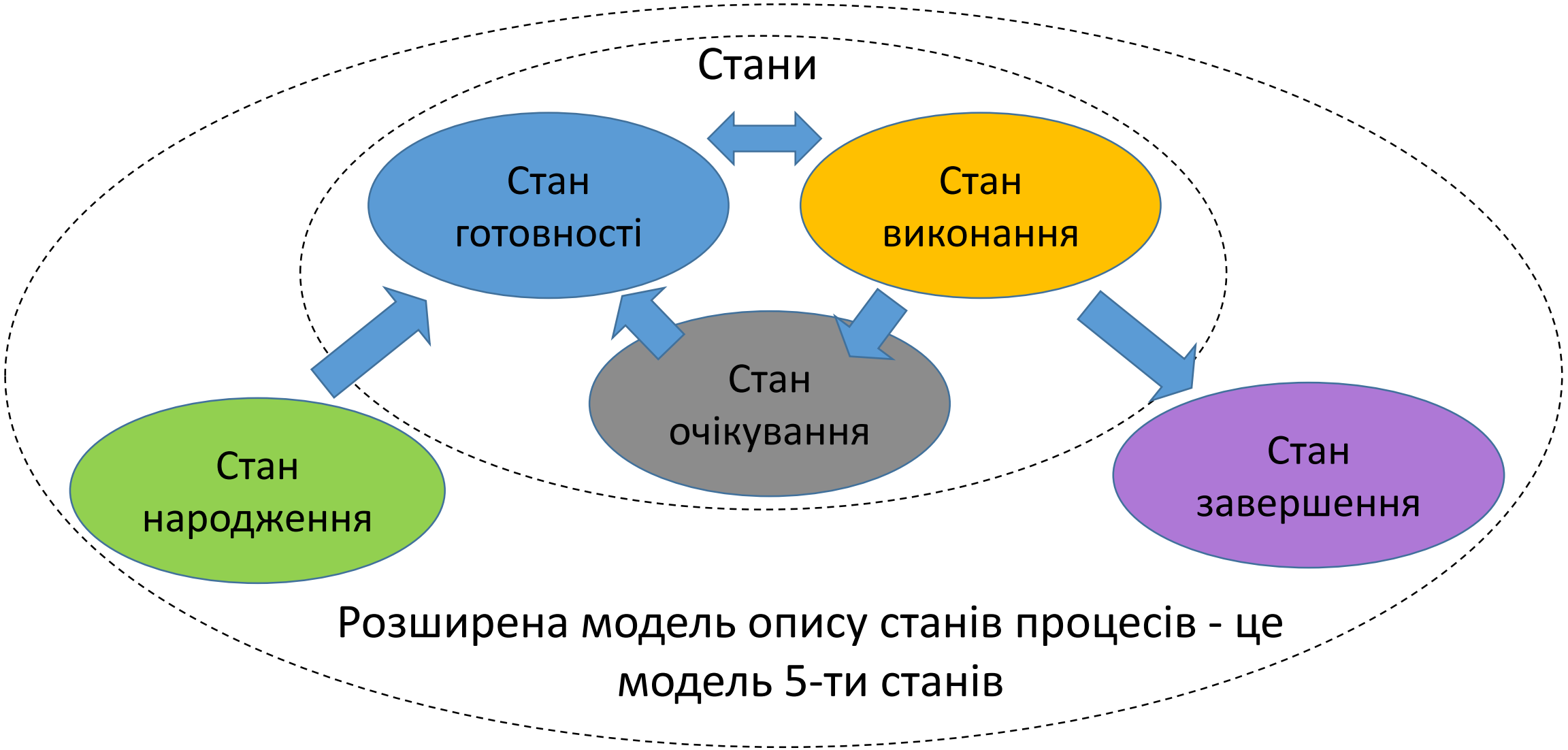
Програма -> Процесор -> Процес

- Термін "процес" вперше з'явився при розробці ОС *Multix*
- Процес - це програма на стадії виконання або "об’єкт", якому виділено процесорний час на процесорі



Модель процесу

Найпростіша модель опису станів процесів - це модель 3-х станів

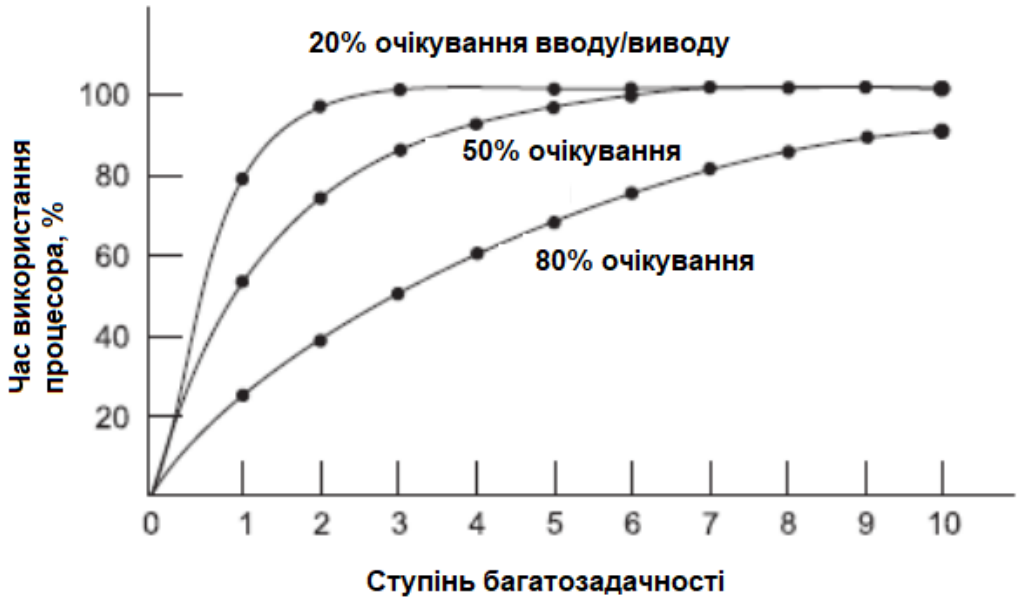


Розширена модель опису станів процесів - це модель 5-ти станів

Оцінка завантаженності процесора в режиму витісняючої багатозадачності. Моделювання

Припущення моделювання (спрощення багатозадачності при абстрагуванні):

- процес проводить частину свого часу *p* в очікуванні завершення операцій вводу/виводу;
- при одночасній присутності в пам'яті *n* процесів ймовірність того, що всі *n* процесів очікують завершення вводу/виводу, = p^n ;



- завантаженність процесору (час використання процесора) = 1 – p^n

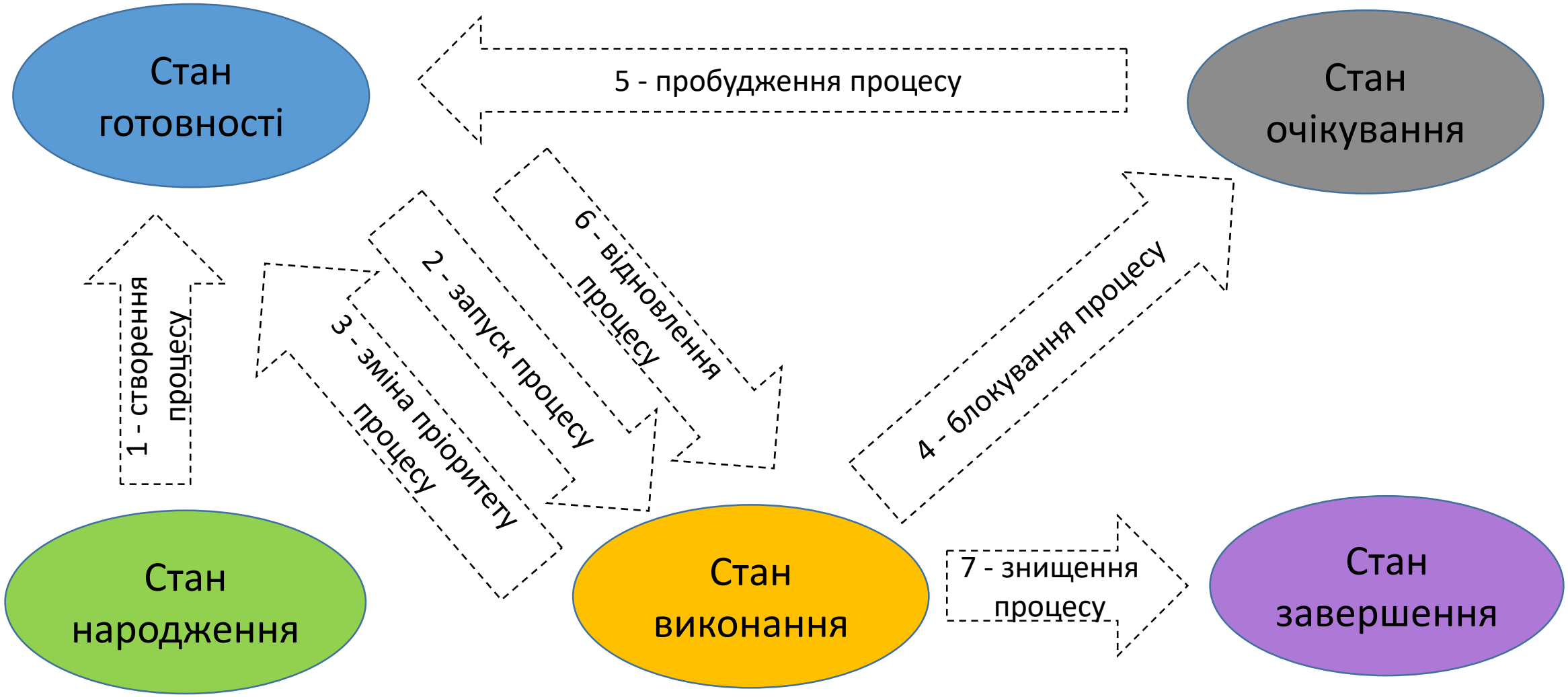
Приклад: ОП = 8 Гбайт, ОС та її таблиці процесів <= 2 Гбайт, кожна програма <= 2 Гбайт (3 програми в ОП). При очікуванні вводу/виводу = 80%, завантаженість процесора, 1 – $0{,}8^3$ = 49%.

Керування об’єктами ОС

- Відомо, що будь-яке керування будь-яким об'єктом включає в себе чотири основні операції (*CRUD*-операції):
  - ***C****reate* – створення об’єкта,
  - ***R****ead* – перегляд значень атрибутів об'єкта;
  - ***U****pdate* – зміна значень атрибутів об'єкта;
  - ***D****elete* – видалення об'єкта.
- Відомі об’єкти ОС: файл, користувач, група, псевдотермінал
- Об’єкт типу «процес ОС» має особливості
- Для процесів ОС вказані операції створення, зміни і видалення представляються у вигляді 7-ми додаткових операцій, які забезпечують перехід процесів між 5-ма станами

Операції керування процесами

Для процесів операції створення, зміни і видалення представляються у вигляді 7-ми операцій



Ієрархія процесів в *Unix*-подібній ОС

Процес не може взятися з порожнини – його запускає інший процес:

- процес, який запустив (породив) інший процес, – це *parent*-процес;
- процес, запущений (породжний) іншим процесом, – це *child*-процес.

У кожного процеса є два атрибути:

- *PID* (*Process ID)* – ідентифікатор процесу;
- *PPID – Parent Process ID*, ідентифікатор *parent*-процесу.

Процеси створюють ієрархію у вигляді дерева:

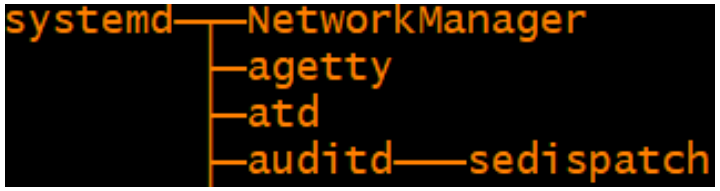
child-процес має лише один *parent*-процес;

parent-процес може мати багато *child*-процесів;

процес з *PID* = 1 – найголовніший процес з назвою *init* або *systemd*

Команда перегляду дерева процесів:

- *pstree [опції] [аргумент]*, наприклад, *pstree -up*



Таблиця процесів ОС

Приклад таблиці процесів як результат виконання команди *PS*

```
[oracle@vpsj3IeQ ~]$ ps -p 9,873,21133,30174,28848 -o pid,stime,time,%cpu,%mem,vsz,rss,cmd
  PID STIME     TIME %CPU %MEM    VSZ   RSS CMD
    9  2021 09:28:53  0.2  0.0      0     0 [rcu_sched]
  873  2021 00:01:27  0.0  5.3 375448 100864 /usr/sbin/named -u named -c /etc/named.conf
21133 08:37 00:00:00  0.0  0.1 115680  2168 -bash
28848 03:18 00:00:00  0.0  0.2 280196  4368 php-fpm: pool index
30174 10:24 00:00:00  0.0  0.1 161724  2360 sshd: miroshnichenko_maksim@pts/1
```

VSZ (Virtual Size), RSS (Resident Set Size)

Приклад таблиці процесів як результат виконання команди *TOP*

```
top - 20:42:43 up 162 days,  5:30,  4 users,  load average: 0.50, 1.17, 0.94
Tasks: 128 total,   1 running, 127 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  0.5 sy,  0.0 ni, 98.5 id,  0.0 wa,  0.0 hi,  0.5 si,  0.2 st
KiB Mem :  1881836 total,   883128 free,   145088 used,   853620 buff/cache
KiB Swap:  4194300 total,  3609888 free,   584412 used.  1687644 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
  488 root      20   0  476436   1140    452 S   1.7  0.1   2935:47 NetworkManager
  274 root      20   0       0      0      0 S   0.3  0.0   7:14.28 jbd2/vda1-8
 2021 mysql     20   0 1272260   1076      0 S   0.3  0.1 170:33.48 mysqld
27079 oracle    20   0  162104   2280   1592 R   0.3  0.1   0:00.11 top
```

PR (Priority), NI (Nice), VIRT (Virtual memory), RES (Resident memory size), SHR (Shared Memory size), S – Process Status

Керування процесами через сигнали ОС

Unix-подібні ОС керують процесами через сигнали, як засіб передачі повідомлення про деякі події в ОС

Процеси теж можуть генерувати сигнали для передачі повідомлень іншим процесам

Команда передачі сигналів:

- *kill [-сигнал] PID [PID ..]*,
- сигнал – номер/ символічне ім'я сигналу

№	Назва	Опис	Комбінація клавіш
1	*HUP*	*Hangup*. Відміна роботи процесу	
2	*INT*	*Interrupt*. У разі виконання простих команд викликає припинення виконання, в інтерактивних програмах - припинення активного процесу	*<Ctrl>+<C>* або *<Ctrl>+<Del>*
3	*QUIT*	Як правило, сильніше сигналу *Interrupt*	*<Ctrl>+<\>*
9	*KILL*	Завершення роботи процесу	
15	*TERM*	*Software Termination*. Вимагання завершити процес (програмне завершення)	
18	*CONT*	Продовжити виконання призупиненого процесу	
19	*STOP*	Призупинення виконання процесу	
20	*TSTR*	Сигнал призупинення процесу, генеруємий клавіатурою. Переводить процес у фоновий режим	*<Ctrl>+<Z>*

Ступінь свободи процесів в *Unix*-подібній ОС

Мінімальний ступінь свободи – режим *foreground* («передній план»):

- процес є результатом запуску програми з командного рядку;
- процес "прив'язується" до псевдотерміналу, з якого він запущений;

Середній ступінь свободи – режим *background* («задній план»):

- процес є результатом запуску програми з командного рядку із завершальним символом & (фоновий режим);

Середній ступінь свободи – режим «*background without hanging up*» :

- процес ігнорує сигнал *HUP (*команда *nohup)*.

Максимальний ступінь свободи – *daemon*-процес («даймон», не демон):

- «не прив’язані» до псевдотерміналу;
- найчастіше для взаємодії з іншими процесами надають свої копії;
- найчастіше запускаються під час старту ОС;
- найчастіше у назві мають останню літеру *d.*

Визначення станів процесів в *Unix*-подібній ОС

R (Running або Runnable) – процес у стані виконання або готовності;
S (Interruptible Sleeping) – процес у стані «сну», не у стані очікування, пов’язаного з операціями вводу/виводу, а, наприклад, у стані очікування завершення свого *child*-процесу
D (Uninterruptible Sleeping) – процес у стані очікування завершення операцій вводу/виводу;
T (Stopped) – процес у стані тимчасового призупинення;
Z (Zombie) – процес-зомбі – помилковий процес, який знаходиться у стані завершення, але від нього не очищено таблицю процесів.
Додаткові властивості станів:
s – процес є лідером псевдотерміналу – головним *parent*-процесом у ланцюжку запуску *child*-процесів власником псевдотерміналу;
+ (плюс) – процес в інтерактивному режимі свого псевдотерміналу, запущений користувачем з командного рядку не у фоновому режимі.

Визначення пріоритетів процесів в *Unix*-подібній ОС

- Пріоритет процесу пов’язано з англійським словом «*nice*» - «милий», «ввічливий».
- «Ввічливий» процес завжди «поступається дорогою» іншим процесам.
- «Грубий» процес намагається сам обійти всі процеси, що знаходяться у стані виконання або готовності.
- *nice = +20* – найнижчий пріоритет процесу, який виконується рідше за інших і тільки тоді, коли ніщо інше не займає процесор;
- *nice = -19* – найвищий пріоритет процесу, який виконується частіше за інших.
- За замовчуванням для *child*-процесу встановлюється значення *nice*, збільшене на +10 у порівнянні із *nice* його *parent*-процесу.
- Негативні значення *nice*, які забезпечать більший приоритет може встановлювати тільки привілейований користувач.

Одеська Політехніка
Інститут комп`ютерних систем
Кафедра інформаційних систем
Дисципліна «Операційні системи»

Дякую за увагу!
Запитання?

Олександр А. Блажко,
доцент кафедри інформаційних систем,
E-mail: blazhko@ieee.org
Telegram-канал: t.me/Operating_Systems_IS

Одеса, 17 квітня 2023 року