5.5 СТАНИ ПРОЦЕСІВ

Необхідно відрізняти системні управляючі процеси, які представляють роботу супервізора ОС і займаються розподілом та управлінням ресурсів, від усіх інших процесів: системних оброблювальних процесів, які не входять в ядро ОС, і процесів користувача. Для системних управляючих процесів у більшості ОС ресурси розподіляються спочатку і однозначно. Тому виконання системних управляючих програм не прийнято називати процесами.

У якому стані знаходиться процес залежить від планування обчислювального процесу. Усі види планування, використовувані в сучасних ОС, будуть детальніше розглянуті в розділі «Планування процесів». Види планування залежно від часового масштабу, діляться на довгострокове, середньострокове, короткострокове і планування введення-виведення.

Розглядаючи частоту роботи планувальника, можна сказати, що довгострокове планування виконується порівняно нечасто, середньострокове дещо частіше. Короткостроковий планувальник, який часто називають диспетчером (dispatcher), працює, визначаючи, який процес або потік виконуватиметься наступним. Нижче наведений короткий перелік функцій, що виконуються планувальником кожного виду.

1. Довгострокове. Рішення про додавання процесу в пул виконуваних в системі процесів.
2. Середньострокове. Рішення про додавання процесу до числа процесів, повністю або частково розміщених в основній пам'яті.
3. Короткострокове. Рішення про те, який з доступних процесів виконуватиметься процесором.
4. Планування введення-виведення. Рішення про те, який із запитів процесів (потоків) на операцію введення-виведення виконуватиметься вільними пристроями введення-виведення.

Процес може знаходитися в активному і пасивному стані. В активному стані процес може брати участь у конкуренції за використання ресурсів, а в пасивному – він тільки відомий системі, але не бере участі в конкуренції (хоча його існування в системі

зв'язане з представленням йому оперативної і/або зовнішньої пам'яті). У свою чергу, активний процес може знаходитися в одному з трьох основних станів (рис. 5.5):

- виконання – усі ресурси, що зажадалися процесом, виділені. У цьому стані в кожен момент часу може знаходитися тільки один процес, якщо йдеться про однопроцесорну обчислювальну систему;
- готовності до виконання – ресурси надані, працездатний, але тимчасово процес призупинений, щоб дати можливість виконання іншому процесу;
- блокування або очікування – ресурси, що зажадалися, не можуть бути надані, або не завершена операція введення-виведення.

У стані виконання в однопроцесорній системі може знаходитися тільки один процес, а в кожному із станів очікування і готовності – декілька процесів. Ці процеси утворюють черги відповідно очікуючих і готових процесів.

Як показано на рис. 5.5, між цими трьома станами можуть бути чотири переходи. Життєвий цикл процесу розпочинається зі стану готовності, коли процес готовий до виконання і чекає своєї черги (працездатний, але тимчасово призупинений, щоб дати можливість виконання іншому процесу).

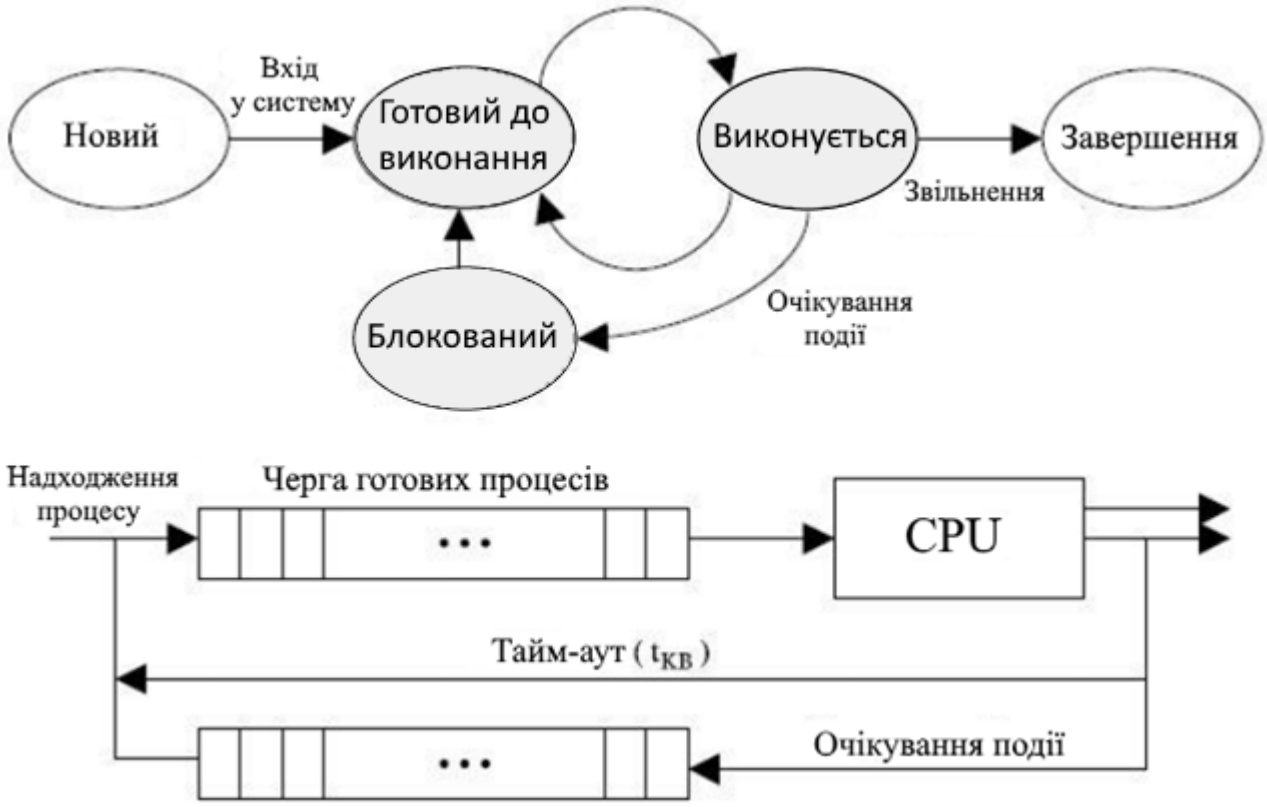

Рисунок 5.5 – Діаграма станів процесу

При активізації процес переходить в стан виконання і знаходиться в ньому до тих пір, поки він сам звільнить процесор, перейшовши в стан очікування якої- небудь події. В стан очікування він може також перейти за допомогою системного виклику.

Процес може бути насильно «витіснений» з процесора, наприклад, унаслідок вичерпання відведеного цьому процесу кванта процесорного часу. В останньому випадку процес повертається в стан готовність. У цей же стан процес переходить із стану очікування після того, як очікувана подія станеться.

У більшості ОС стан блокування (очікування), у свою чергу, поділяється на певну кількість станів очікування, що відповідають певному виду ресурсу, через відсутність якого процес переходить в заблокований стан.

Призупинені процеси

Три основні стани процесів, описані раніше (готовий, виконується і блокований), дозволяють змоделювати поведінку процесів і отримати уявлення про реалізацію ОС. Багато ОС створені на основі тільки цих трьох станів.

Можна навести переконливі аргументи на користь додавання в модель процесів і інших станів. Щоб усвідомити, які переваги можуть дати ці нові стани, розглянемо систему, яка не використовує віртуальну пам'ять, в якій кожен процес перед виконанням потрібно завантажити в основну пам'ять. Таким чином, усі процеси, які представлені моделлю з 3-ма станами, повинні знаходитися в основній пам'яті.

Причиною розробки такої системи було повільне, в порівнянні з обчисленнями, виконання операції введення-виведення, яке призводило до простоїв CPU в однозадачній системі. Але організація роботи відповідно до схеми з трьома станами повністю цю проблему не вирішує. Звичайно, при роботі з такою моделлю в пам'яті знаходиться декілька процесів, і доки одні процеси чекають завершення операції введення-виведення, процесор може перейти до виконання інших процесів. Але процесор працює настільки швидше за виконання операцій введення-виведення, що незабаром усі процеси, що знаходяться в пам'яті, виявляються в стані очікування.

Таким чином, CPU може простоювати навіть у багатозадачній системі. Якщо збільшити розмір основної пам'яті для розміщення в ній більшої кількості процесів, то це, по-перше, призведе до подорожчання пам'яті і системи в цілому, а по-друге, сучасні

технології створення програм дозволяють створювати все складніші і тому великі за розміром програми.

Іншим рішенням проблеми є свопінг, який включає перенесення частини процесів з основної пам'яті на диск. Якщо в основній пам'яті немає жодного готового до виконання процесу, ОС переводить один з блокованих процесів на диск (здійснює його свопінг), розміщуючи його в чергу призупинених (блокованих) процесів, які тимчасово витягнуті з основної пам'яті. Далі ОС завантажує інший процес з черги призупинених, після чого продовжує його виконання.

Але свопінг сам по собі є операцією введення-виведення, тому існує ризик погіршити ситуацію, замість її удосконалення. Завдяки тому, що обмін з диском виконується швидше за інші операції введення-виведення (наприклад, виведення на друк), то свопінг найчастіше підвищує продуктивність системи в цілому.

Якщо в модель процесів ввести свопінг, то необхідно ввести новий стан – стан призупиненого процесу. Коли всі процеси в основній пам'яті знаходяться в блокованому стані, ОС може призупинити один з процесів, переводячи його в призупинений стан. Простір, що звільнився в основній пам'яті, потім використовується для завантаження іншого процесу.

Після того, як ОС вивантажила один з процесів на диск, вона має дві можливості вибору процесу для завантаження в основну пам'ять: вона може або створити новий процес, або завантажити процес, який був призупинений перед цим. Може здатися, що краще було б завантажити для обробки раніше призупинений процес, що не призведе до збільшення навантаження на систему. Але це не зовсім так. Усі процеси,

перш ніж були призупиненими, знаходилися в блокованому стані.

Зрозуміло, що повернення в пам'ять блокованого процесу не дає ніяких результатів, оскільки він все одно не готовий до виконання. Адже кожен процес в призупиненому стані блокований в очікуванні на якусь певну подію. Якщо ця подія відбувається, процес перестає бути блокованим і можна продовжити його виконання.

Це слід врахувати при розробці ОС. Існують дві незалежні ситуації: чи чекає процес якої-небудь події (тобто, блокований він або ні); чи вивантажений процес з основної пам'яті (тобто, призупинений він чи ні). Маємо 2x2 можливі комбінації, тому необхідні чотири перераховані нижче стани.

1. Готовий. Процес, який знаходиться в основній пам'яті і готовий до виконання.

2. Блокований. Процес, який знаходиться в основній пам'яті і чекає на певну подію.

3. Блокований/Призупинений. Процес, який знаходиться в зовнішній пам'яті (диск) і чекає на певну подію.

4. Готовий/Призупинений. Процес, який знаходиться в зовнішній пам'яті (диск), але готовий до виконання, його потрібно тільки завантажити в основну пам'ять.

Перш ніж скласти діаграму переходів станів, де враховуються два нові призупинені стани, потрібно згадати ще одну обставину. Досі ми не враховували наявність віртуальної пам'яті. Вважалося, що процес знаходиться або повністю в основній пам'яті, або повністю в зовнішній пам'яті. За наявності віртуальної пам'яті з'являється можливість виконувати процес, який завантажений в основну пам'ять частково.

Якщо відбувається звернення до відсутнього в основній пам'яті адреси процесу, то ця частина процесу може бути завантажена. Здавалося б, що використання віртуальної пам'яті позбавляє необхідності явного свопінгу, оскільки будь-яку потрібну адресу будь-якого процесу можна перенести в основну пам'ять або з неї за допомогою апаратного забезпечення CPU, керуючого пам'яттю.

Проте, як дізнаємося далі щодо віртуальної пам'яті, за наявності досить великої кількості активних процесів, які повністю або частково знаходяться в основній пам'яті, продуктивність віртуальної пам'яті може виявитися недостатньою. Тому, навіть за наявності віртуальної пам'яті, ОС час від часу вимагається явно і повністю вивантажувати процеси з основної пам'яті заради підвищення загальної продуктивності.

На рис. 5.6 показана діаграма станів процесу, модифікована з урахуванням операцій призупинення і відновлення.

Рисунок 5.6 – Діаграма стану процесу з операціями призупинення і відновлення

У діаграму введені два нові стани, а саме «призупинений_готовий» і «призупинений_блокований», стану «призупинений_виконується» не буває.

Ініціатором призупинення може бути або сам процес, або інший процес. В однопроцесорній машині процес, що виконується, не може працювати одночасно з ним, щоб видати сигнал призупинення. У мультипроцесорній машині процес, що виконується, може бути призупинений і іншим процесом, що виконується на іншому процесорі.

Процес, що знаходиться в стані готовності, може бути призупинений тільки іншим процесом.

Розглянемо детальніше діаграму станів процесу з операціями призупинення і відновлення, найважливіши з яких наводяться нижче.

Блокований → Блокований/Призупинений. Якщо до виконання не готовий жоден процес, то, принаймні, один блокований процес вивантажується з пам'яті, щоб звільнити місце для іншого процесу, який не є блокованим. Цей перехід можна виконувати і за наявності готових до виконання процесів, якщо ОС визначить, що для процесу, що виконується зараз, або процесу, управління до якого перейде найближчим

часом, треба збільшити об'єм основної пам'яті для забезпечення високої продуктивності.

Блокований/Призупинений → Готовий/Призупинений. Процес у стані блокованого/призупиненого переходить у стан готового до виконання призупиненого процесу, якщо відбувається подія, якої чекав цей процес. Для цього необхідно, щоб ОС мала доступ до інформації про стан призупинених процесів.

Готовий/Призупинений → Готовий. Коли в основній пам'яті немає готових до виконання процесів, ОС для продовження обчислень вимагається завантажити процес в пам'ять. Може статися і так, що в готового до виконання призупиненого процесу більш високий пріоритет, ніж у будь-якого іншого з готових до виконання процесів. У такій ситуації розробник ОС може вирішити, що важливіше забезпечити пріоритет процесу, чим мінімізувати свопінг.

Готовий → Готовий/Призупинений. ОС вважає за краще призупиняти не готовий до виконання процес, а заблокований процес, оскільки до виконання готового процесу можна приступити негайно, а блокований процес тільки даремно займає основну пам'ять, оскільки не може бути виконаний. Але іноді виявляється, що єдиний спосіб звільнити досить великий розмір основної пам'яті – це призупинити готовий до виконання процес. ОС може також замість блокованого процесу з вищим пріоритетом призупинити готовий до виконання процес з нижчим пріоритетом, якщо блокований процес досить скоро буде готовий до виконання.

Новий → Готовий/Призупинений і Новий → Готовий (на рисунку не зображений). Після створення нового процесу цей процес може бути доданий або в чергу готових до виконання процесів, або в чергу готових до виконання призупинених процесів. У будь-якому з цих випадків ОС повинна створити таблиці для управління процесом і виділити йому адресний простір. Краще виконувати ці дії на ранніх етапах, щоб мати більший запас неблокованих процесів. Але якщо дотримуватися цієї стратегії, то в основній пам'яті може не вистачити місця для нового процесу. З цієї причини передбачений перехід нового процесу в стан призупиненого готового до виконання. З іншого боку, створення процесу в «останню мить» призводить до зменшення непродуктивних витрат і дозволяє ОС виконувати свої обов'язки зі створення процесів навіть тоді, коли вона переповнена блокованими процесами.

Блокований/Призупинений → Блокований. На перший погляд може здатися, що враховувати такий перехід немає сенсу. Для чого завантажувати в пам'ять процес, який не готовий до виконання? Але розглянемо ситуацію: завершився деякий процес, звільнивши при цьому певну частину основної пам'яті. У черзі заблокованих призупинених процесів знаходиться процес, пріоритет якого вищий, ніж у будь-якого процесу з черги готових до виконання, але призупинених процесів. Крім того, ОС має в розпорядженні аргументи на користь того, що скоро станеться подія, яка зніме блокування з цього високопріоритетного процесу. При таких обставинах доречно віддати перевагу блокованому процесу перед готовим до виконання, завантаживши в основну пам'ять саме його.

Виконується → Готовий/Призупинений. Процес, що виконується, в якого вийшов відведений йому час, переходить в стан готового до виконання. Проте за наявності процесу з вищим пріоритетом, який знаходиться в черзі блокованих призупинених процесів, і тільки що був розблокований, ОС може віддати перевагу саме йому. Щоб звільнити частину основної пам'яті, вона може перевести процес, що виконується, безпосередньо в стан готового до виконання призупиненого процесу.

Довільний стан → Завершення (на рисунку не зображений). Завершується процес, що виконується зараз, – це відбувається або через те, що він виконаний до кінця, або через помилки при виконанні. Але в деяких ОС процес може завершитися процесом, що створив його, або разом із завершенням батьківського процесу. Таке завершення можливе за умови, що процеси з будь- якого стану можуть переходити в стан завершення.

У конкретних операційних системах стани процесу можуть бути ще більше деталізовані, можуть з'явитися деякі нові варіанти переходів з одного стану в інший. Так, наприклад, модель станів процесів для операційної системи Windows NT містить 7 різних станів, а для операційної системи Unix – 9. Проте так чи інакше, усі операційні системи підпорядковуються викладеній вище моделі.