**Wielozadaniowość**

**Wielozadaniowość** – cecha systemu operacyjnego umożliwiająca mu równoczesne wykonywanie więcej niż jednego procesu. Zwykle za poprawną realizację wielozadaniowości odpowiedzialne jest jądro systemu operacyjnego.

Wielozadaniowość zapewniona jest między innymi przez planistę, czyli część systemu operacyjnego realizującą algorytm szeregowania zadań w kolejce do przyznania czasu procesora.

Równoczesność jest pozorna, gdy system ma dostępnych mniej procesorów niż zadań do wykonania. Wówczas dla uzyskania wrażenia wykonywania wielu zadań jednocześnie, konieczne staje się dzielenie czasu.

Systemy wielozadaniowe można podzielić na oferujące i nie oferujące wywłaszczania. W systemach z wywłaszczaniem może nastąpić przerwanie wykonywania procesu, odebranie mu procesora i przekazanie sterowania do planisty. Pełne wywłaszczanie zapewniają mechanizmy sprzętowe działające niezależnie od oprogramowania (np. dołączanie wywłaszczania do procedury obsługi przerwania zegarowego). W systemach bez wywłaszczania procesy powinny same dbać o sprawiedliwy podział czasu, co często uzyskuje się pośrednio - proces dokonując wywołania systemowego, oddaje sterowanie procesowi jądra, lub jednemu z procesów systemowych i w ten sposób zrzeka się procesora. Program nie wykonywany pozostaje "w uśpieniu" do momentu, gdy znów zostanie mu przydzielony czas procesora.

**Procesy, wątki i wielozadaniowość**

**Proces** - jedno z najbardziej podstawowych pojęć w informatyce, definiowane jako egzemplarz wykonywanego programu, jednak każdy nowo powstały proces otrzymuje unikalny numer, który go jednoznacznie identyfikuje, tzw. numer PID (ang. *process IDentifier*).

W celu wykonania programu system operacyjny przydziela procesowi zasoby (pamięć, czas procesora i inne), ale także może być konieczne współbieżne wykonywanie pewnych fragmentów programu. Aby to zrealizować program może zażądać utworzenia określonej liczby wątków, wykonujących wskazane części programu - o ich współbieżne wykonanie dba system operacyjny (albo sam program, wówczas mówi się o zielonych wątkach). Wątki współdzielą prawie wszystkie zasoby zarezerwowane dla danego procesu, wyjątkiem jest **czas procesora**, który jest przydzielany indywidualnie każdemu wątkowi.

Za zarządzanie procesami odpowiada jądro systemu operacyjnego, sposób ich obsługi jest różny dla różnych systemów operacyjnych. W systemie operacyjnym każdy proces posiada proces nadrzędny, z kolei każdy proces może, poprzez wywołanie funkcji systemu operacyjnego, utworzyć swoje procesy potomne; w ten sposób tworzy się swego rodzaju drzewo procesów. Każdy proces otrzymuje od systemu operacyjnego odrębne zasoby, w tym odrębną przestrzeń adresową, listę otwartych plików, urządzeń itp.

W skład procesu wchodzi:

* kod programu,
* licznik rozkazów,
* stos,
* sekcja danych.

Każdemu procesowi przydzielone zostają zasoby, takie jak:

* procesor,
* pamięć,
* dostęp do urządzeń wejścia-wyjścia,
* pliki.

Użytkownik za pomocą powłoki zleca uruchomienie programu, proces wywołujący wykonuje polecenie *fork*, lub jego pochodną.

* System operacyjny tworzy przestrzeń adresową dla procesu oraz strukturę opisującą nowy proces w następujący sposób:
	+ wypełnia strukturę opisującą proces,
	+ kopiuje do przestrzeni adresowej procesu dane i kod, zawarte w pliku wykonywalnym,
	+ ustawia stan procesu na działający,
	+ dołącza nowy proces do kolejki procesów oczekujących na procesor (ustala jego priorytet),
	+ zwraca sterowanie do powłoki użytkownika.

Dany proces rozpoczyna wykonywanie w momencie przełączenia przez Jądro systemu operacyjnego przestrzeni adresowej na przestrzeń adresową danego procesu oraz takie zaprogramowanie procesora, by wykonywał kod procesu. Wykonujący się proces może żądać pewnych zasobów, np. większej ilości pamięci. Zlecenia takie są na bieżąco realizowane przez system operacyjny.

Wykonanie procesu musi przebiegać sekwencyjnie. Może przyjmować kilka stanów:

* aktualnie wykonywany przez procesor,
* czekający na udostępnienie przez system operacyjny zasobów,
* uśpiony
* przeznaczony do zniszczenia,
* proces zombie,
* właśnie tworzony itd.

Proces wykonuje ostatnią instrukcję - zwraca do systemu operacyjnego kod zakończenia. Jeśli proces zakończył się poprawnie zwraca wartość 0, w przeciwnym wypadku zwraca wartość kodu błędu.

* W momencie zwrotu do systemu operacyjnego kodu zakończenia, system operacyjny ustawia stan procesu na przeznaczony do zniszczenia i rozpoczyna zwalnianie wszystkich zasobów, które w czasie działania procesu zostały temu procesowi przydzielone.
* System operacyjny po kolei kończy wszystkie procesy potomne w stosunku do procesu macierzystego.
* System operacyjny zwalnia przestrzeń adresową procesu. Jest to dosłowna śmierć procesu.
* System operacyjny usuwa proces z kolejki procesów gotowych do uruchomienia i szereguje zadania. Jest to ostatnia czynność wykonywana na rzecz procesu.
* Procesor zostaje przydzielony innemu procesowi.

**Wątek** (ang. *thread*) - część programu wykonywana współbieżnie w obrębie jednego procesu; w jednym procesie może istnieć wiele wątków.

Różnica między zwykłym procesem a wątkiem polega na współdzieleniu przez wszystkie wątki działające w danym procesie przestrzeni adresowej oraz wszystkich innych struktur systemowych (np. listy otwartych plików, gniazd, itp.) - z kolei procesy posiadają niezależne zasoby.

Ta cecha ma dwie ważne konsekwencje:

1. Wątki wymagają mniej zasobów do działania i też mniejszy jest czas ich tworzenia.
2. Dzięki współdzieleniu przestrzeni adresowej (pamięci) wątki jednego zadania mogą się między sobą komunikować w bardzo łatwy sposób, niewymagający pomocy ze strony systemu operacyjnego. Przekazanie dowolnie dużej ilości danych wymaga przesłania jedynie wskaźnika, zaś odczyt (a niekiedy zapis) danych o rozmiarze nie większym od słowa maszynowego nie wymaga synchronizacji (procesor gwarantuje atomowość takiej operacji).

Wątki są udostępniane wprost przez system operacyjny MS Windows

W systemach wieloprocesorowych, a także w systemach z wywłaszczaniem, wątki mogą być wykonywane równocześnie (współbieżnie). Równoczesny dostęp do wspólnych danych grozi jednak utratą spójności danych i w konsekwencji błędem działania programu.

**Wielowątkowość** (ang. *multithreading*) – cecha systemu operacyjnego, dzięki której w ramach jednego procesu może wykonywać kilka wątków lub jednostek wykonawczych. Nowe wątki to kolejne ciągi instrukcji wykonywane oddzielnie. Wszystkie wątki tego samego procesu współdzielą kod programu i dane.

**Wielowątkowość** może także odnosić się do samych procesorów. W takim wypadku oznacza możliwość jednoczesnego wykonywania wielu wątków sprzętowych na pojedynczej jednostce wykonawczej – rdzeniu (ang. *core*). Wielowątkowość w procesorach możliwa jest dzięki temu, że nie wszystkie części jednostki wykonawczej są w jednakowym stopniu wykorzystywane przez pojedynczy wątek (ciąg instrukcji). Nieaktywne części jednostki wykonawczej mogą w tym czasie wykonywać inny wątek zwiększając efektywność wykorzystania całego procesora. W zależności od rodzaju technik zastosowanych do obsługi dodatkowych wątków sprzętowych spotyka się od 2 (najczęściej) do nawet 8 wątków sprzętowych na pojedynczy rdzeń procesora (*core*).

### Cechy wielowątkowości

* wszystkie wątki wykonują się w ramach tylko jednego programu (procesu) - innymi słowy jeden proces posiada wiele instancji wykonawczych (wątków)
* wątki zostały wprowadzone aby umożliwić przetwarzanie współbieżne, np gdy zachodzi potrzeba wykonania wielu zadań jednocześnie. Może się to wiązać również ze zwiększeniem wydajności o ile istnieją odpowiednie zasoby sprzętowe (wiele procesorów lub obsługa wielowątkowości przez pojedynczy procesor). Należy pamiętać, iż wprowadzenie wątków może obniżyć wydajność ponieważ najczęściej wymagane jest przy tym wprowadzenie odpowiednich mechanizmów synchronizacji;
* wszystkie wątki procesu współdzielą tę samą wirtualną przestrzeń adresową (mają dostęp do tych samych "egzemplarzy" zmiennych, obiektów i struktur) i korzystają z tych samych zasobów systemowych;
* komunikacja między wątkami w odróżnieniu od procesów jest bardzo łatwa do wykonania – w przypadku wątków wystarczy odwoływać się do tych samych zmiennych i obiektów – komunikacja między procesami wymaga zastosowania mechanizmów IPC (ang. *InterProcess Communication*);
* współdzielenie wirtualnej przestrzeni adresowej niesie zagrożenie – jeden "wadliwy" wątek może zagrozić wykonaniu całego programu;
* każdy wielowątkowy system operacyjny zapewnia specyficzne metody synchronizacji wątków, które z wyżej wymienionych przyczyn należy bezwzględnie zaimplementować.

Systemy wielowątkowe to m.in. BeOS, Microsoft Windows 95, Windows NT, Unix, Linux.

**Wielozadaniowość** – cecha systemu operacyjnego umożliwiająca mu równoczesne wykonywanie więcej niż jednego procesu. Zwykle za poprawną realizację wielozadaniowości odpowiedzialne jest jądro systemu operacyjnego.

Wielozadaniowość zapewniona jest między innymi przez planistę, czyli część systemu operacyjnego realizującą algorytm szeregowania zadań w kolejce do przyznania czasu procesora.

Równoczesność jest pozorna, gdy system ma dostępnych mniej procesorów niż zadań do wykonania. Wówczas dla uzyskania wrażenia wykonywania wielu zadań jednocześnie, konieczne staje się dzielenie czasu.

Systemy wielozadaniowe można podzielić na oferujące i nie oferujące wywłaszczania. W systemach z wywłaszczaniem może nastąpić przerwanie wykonywania procesu, odebranie mu procesora i przekazanie sterowania do planisty. Pełne wywłaszczanie zapewniają mechanizmy sprzętowe działające niezależnie od oprogramowania (np. dołączanie wywłaszczania do procedury obsługi przerwania zegarowego). W systemach bez wywłaszczania procesy powinny same dbać o sprawiedliwy podział czasu, co często uzyskuje się pośrednio - proces dokonując wywołania systemowego, oddaje sterowanie procesowi jądra, lub jednemu z procesów systemowych i w ten sposób zrzeka się procesora. Program nie wykonywany pozostaje "w uśpieniu" do momentu, gdy znów zostanie mu przydzielony czas procesora.

**Dzielenie czasu**, zwane **podziałem czasu**, w wielozadaniowych systemach operacyjnych, jest mechanizmem pozwalającym na wykonywanie jednocześnie wielu zadań (zwykle dotyczy procesów, czasem również wątków) w obrębie jednego procesora w pewnej, umownej, jednostce czasu.

Za mechanizm dzielenia czasu odpowiedzialny jest planista. Decyduje on o kolejności przełączania zadań oraz o wyznaczaniu okresu, na jaki danemu zadaniu przydziela się procesor. Takie postępowanie może sprawiać wrażenie równoległego wykonywania tychże zadań (przy czym każde z nich ma do dyspozycji tylko część, uzależnionej od planisty, mocy obliczeniowej procesora).

Aby dzielenie czasu funkcjonowało właściwie, konieczne jest zapewnienie możliwości wywłaszczania zadań.

W przypadku wieloprocesorowych systemów komputerowych, zadania mają do dyspozycji więcej niż jeden procesor i w takim wypadku mogą rzeczywiście wykonywać się jednocześnie.

**Wywłaszczenie** - to technika używana w środowiskach wielowątkowych, w której algorytm szeregujący (scheduler) może wstrzymać aktualnie wykonywane zadanie (np. proces lub wątek), aby umożliwić działanie innemu. Dzięki temu rozwiązaniu zawieszenie jednego procesu nie powoduje blokady całego systemu operacyjnego. W systemach bez wywłaszczenia zadania jawnie informują scheduler, w którym momencie chcą umożliwić przejście do innych zadań. Jeżeli nie zrobią tego w odpowiednim czasie, system zaczyna działać bardzo wolno. Oprócz tego wywłaszczanie umożliwia szczegółowe określanie czasu, w jakim dany proces może korzystać z procesora. Wywłaszczanie w niektórych systemach operacyjnych może dotyczyć nie tylko programów, ale także samego jądra - przykładem takiego systemu jest Linux.

Wywłaszczanie jest często ograniczane, na przykład procedury odpowiedzialne za obsługi przerwań sprzętowych są zwykle niewywłaszczalne, co znacznie upraszcza ich konstrukcje ale wymusza też zadbanie o to, żeby szybko się kończyły umożliwiając działanie innym procesom.