ZARZĄDZANIE PROCESAMI

Lekcja

Temat: Procesy

1. Procesy

Proces jest wykonywanym programem. Wykonanie procesu musi przebieać w sposób sekwencyjny Oznaza to że w dowolnej chwili na zamówienie danego procesu może być wykonywany co najwyżej jden rozkaz kodu programu.

Proces to nie tylko kod programu. W pojeciu procesu mieści się również bieżąca czynność reprezentowana przez wartość licznika rozkazów oraz zawartość rejestrów procesora.

Zasoby są przydzialena procesowi w chwili jego tworzenia lub podczas późniejszego działania.

1. Diagram stanów procesu [ wstaw pszczółke ]
2. Stan procesu
3. Blok kontrolny procesu

Kżdy proces jest reprezentowany w systemie operacyjnym przez blok kotrolny procesu ( ang.Process control block – CPB)

1. Przełączanie procesora do różnych procesów

Temat: Stany procesów

1. Dwustanowy model procesu

Jeśli wykonanie procesu zostało anulowane lub zakończone to opuszcza on system, a program przydzielający wybiera kolejny proces z kolejki który zostanie wykonany. W modelu dwustanowym kolejka działa na zasadzje FIFO a procesor wykonuje procesy cyklicznie

1. Model procesu w systemie Unix
   * wykonywany w trybie urzytkownika – działąjąc w trybie użytkownika
   * wykonywany w trybie jądra – działający w trybie jądra
   * gotowy do uruchomienia
   * śpiący w pamięci – niemożliwy do uruchomienia dopóki nie zajdzie odpowiednie zdarzenie
   * gotowy do uruchomienia w obszarze wymiany – gotowy ale musi być przeniesiony do pamięci głównej
   * śpiący w obszarze wymiany – proces przeniesiony do obszary wymiany i przeniesiony do obszaru wymiany i oczekujący na wystąpenie pewnego zdarzenia (zalokowany)
   * wywłaszczony – wywłaszczony w celu uruchomienia innego procesu
   * stworzony – nowo utworzony proces
   * zombi – proces któy przestał istnieć ale pozostawił zapis o swoim procesie rodzicielskim

Palniści

Planista krótkoterminowy musi bardzo często wybierać nowy proces dla procesora.

Planista długoterminowy działa o wiele rzadziej. Między utworzeniem nowych procesów w systemie mogą upływać minuty.

Planista długoterminowy nadzoruje stopień wieloprogramowości czyli liczbę procesów w pamięci.

Jeśli stopień wieloprogramowości jest stabilny to średnia liczba utworzoych procesów musi się równać średniej liczbie procesów usuwaych.

Czyli planista długoterminowy może btć wywołany tylko wtedy gdy jakiś proces opuszcza system.

Planista długoterminowy może w niektórych systemach być nieobecny lub mocno zredukowany. Np. procesy z podziałem czasu często nie mają żadnego planisty długoterminowego i umieszczają każdy nowy proces w pamięci pod opieką lanisty krótkoterminowego

W niktórych systemach operacyjnyh takich jak systemy z podziałem czasu możę występowaćdodatkowy pośredni poziom planowania

Pośredni poziom planowania to planista średnioterminowy

Podstawowym argumentem przemawiającym za użyciem planisty średnioterminowego jest fakt, że niekiedy może okazać się korzystne usunięcie procesów z pamięci w celu zmniejszenia stopnia wieloprogramowości.

Usunięte procesy można póniej znów wprowadzić do pamięci a ich wykonanie kontynuować do tych miejsc w których je przerwano

Posępowanie takie częso nazywane jest swappingiem (wymianą)

Działania na procesach

Procesy w systemie mogą uć wykonywane współbieżnie oraz dynamicznie tworzone i usuwanie. Sytem operacyjny musi więc zawierać mechanizm tworzenia i kończenia procesu.

Twoezenie procesu:

Proces może tworzyć nowe procesy za poomocą wywołania systemowego „utwórz proces”.

Proces tworzący nazywa się procesem macierzystym, a utworzone przez niego nowe procesy nazywają sięj ego potomkami.

Każdy nowy proces możetworzyć kolejne procesy, wskutek czego powstaje drzewo procesów

Do wypełnienia swoich zadań proces potrzebuje pewnych zasobów.

Gdy proces tworzy podproces ten drugi może otrzymać swoje zasoby bezpośrednio od systemu operacyjnego bądz od procesu macierzystego

Proces macierzsty może rozdzielać własne zasoby między procesy potomne albo powodować że niektóre zasoby będą przez otomków żywane wspólnie

Kiedy proes stworzy nowy proces wtedy:

* proces macierzysty kontynuuje działanie współbieżne ze swoimi potomkami
* proces macierzysty oczekuje na zakończenie działań niektórych lub wszystkich swoich procesów potomnych

Proces końćzt się wówczas gdy wykona swoją ostatnią instrukcjęi za pomocąwywołąnia funcji exit poprosi system operacyjny aby go usunął

Oprócz tego istnieją inne przyczyny zakończenia procesu.

Proces może spowodować zakończenie innego procesu za pomoną odpowiedniej funkcji systemowej.

Funkcję tę można wywołać tylko przodek procesu, któy ma być zakończony. W przeciwnym razie użytkownicy mogliby likwidować sobie wzajemnie dowolne zadania.

Proces macierzysty może zakończyć któryś ze swoich procesów potomnych z różnych przyczyn:

* potomek nadużył któregoś z przydzelonych mu zasobów
* wykonywane przez potomka zadanie stało się zbędne
* proces macierzysty kończy sięa w tej sytuacji system operacyjny nie pozwala ptomkowi na dalsze zadania

Lekcja

Temat: Wątki

1. Wątek (ang. thread) – nazywane czasami procesem lekkim, jest podstawowąjednostką wykorzystania procesora, w któej skład wchodzą:licznik rozkazów, zbió rejestrów i obszar stosu
2. Wątek współużytkuje wraz z inymi równorzędnymi wątkami sekcję kodu, sekcję danych oraz zasoby, np. otwarte pliki
3. Proces Tradycyjny, czyli ciężki jest równoznaczny z zadaniem z jednym wątkiem
4. Dzielenie zasobów powoduje, że przełączanie procesora międz równorzędnymi wątkami, jak równieżtworzenie wątkó jest tanie w porónaniu z przełączaniem kontekstu między procesami ciężkimi.
5. Przełączanie kontekstu między wątkami wymaga przełączania zbioru jestrów, ale nie rzeba wykonywać żadnych prac związanych z zarządzaniem pamięcią.
6. Wątki mogą znajdować sięw jedym z kilku stanów: gotości, zablokowania, aktywności lunb kończenia. Podobnie jak procesy, atki używają wspólnie jednostkę centralną i w danej chwili tylko jeden wątek jest akywny (wykonywany).
7. Wykonywanie wątku w procesie przebiega sekwencyjnie, a każdy wątek ma własy stos i licznik rozkazów. Wątki mogątworzyć wątki potomne i mogą blokować się do czasu zakońceznia wywołań systemowych. W odróżnieniu od procesów wątki nie są niezależne od siebie.
8. Wielowątkowośćoznacza zdolność systemu operacyjnego do obsłgi wielu wątków w ramach jednego procesu.
9. Główne korzyści z zastosowania wątków dotyczą wydajności:
   * Stworzenie noweo watku zajmuje mniej czasu niż stworzenie noweggo procesu
   * zamknięcie wątku trwa krócej niż zamknięcie procesu
   * Przelączanie między wąkami jest szybsze niżmiędzy procesami
   * wątki podnoszą szybkość komunikacji między wykonywanymi zadaniami
10. Istnieją dwie kategorie wątków: wątki poziomu użytkownika oraz wątkipoziomu jądra (wątki wspierane przez jądro, procesy lekkie).
11. Wątkami poziomu użytkownika nazywamy atki, któe są zarządzae wyłącznie przez aplikacje, a jądro nie jest odpowiedzialne za ich istnienie. Dowolną aplikację można zaprogramować wielowątkowo, korzystając z biblioteki wątków, czyli zestawu procedur do zarządzania wątkami użytkownika.
12. Wątki poziomu jądra nazywamy takie atki, które sązarządzane wyłącznie przez jądro.

Główną wadą stosowania wątków poziomu jądra jest konieczność przełączania na tryb jądra za każdym razem kiedy trzeba przełączyć sterowanie między wątkami tego samego procesu.

1. Rozwiązanie mieszane – Tworzenie wątku odbywa się w przestrzeni użytkownika, a wiele wątków poziomu użytkownika z pojedynczego programu użytkownika zostaje odwzorowanych na większą lub mniejszą liczbę wątków poziomu jądra dla konkretnej aplikacji i dla konkretnej maszyny tak, aby osiągnąć najlepszy wynik.

Lekcja

Temat: Komunikacja międzyprocesorowa

1. Komunikacja międzyprocesorowa

Procesy współpracujące mogą siękomunikować w środowisku ze wspólną pamięcią . Wymfa to dzilenia przez procesy wspólnego bufora. Inna możliwośc polega na tym, że SO zawiera środki pozwalające współpracującym procesom na kontaktowanie się ze sobą.

Oprogramowanie komunikacji międzyprocesowej tworzy mechanizm umożliwiający procesom łącznośći synchroniację działań. Komunikację międzyprocesową najlepiej realizuje się za pomocąsystemu przekazywania komunikatów.

Pamięć dzelona i schematy łączności systemów komunikatów nie wykluczają sięwzajemnie i mogą btć stosowane jednocześnie w obrębie jednego OS lub nawet w jednym procesie.

Komunikaty wysyłane przez procesmogą mieć stałą lub zmienną długość.

1. Lomunikacja bezpośrednia

Łączenie w tym schemacie ma następujące właściwości:

- łącze jest ustawiane automatycznie między parą procesó; do komunikowania sięwystarczy, aby procesy znały swoje identyfikatory

- łącze dotyczy dokładnie dwóch procesów

- między każdą parą procesów istnieje dokładnie jedno łącze

1. Komunikacja pośrednia

W komunikacji pośredniej komunikaty są nadawane i odbierane za pośrednictwem skrzynek pocztowych, nazywanych także portami. Skrzynka jest obiektem, w którym procesy mogąumieszczać komunikaty i z którego komunikaty mogą być pobierae. Każda skrzynka pocztowa ma jednoznaczną identyfikację.

W tej metodzie proces może skomunikować się z innym procesem,gdy mają one jakąś wspólną skrzynkę:

- nadaj (A, komunikat) – nadaj komunikat do skrzynki A

- odbierz (A, komunikat) – odbierz komunikat ze skrzynki A

Łącze komunikacyjne w tym schemacie ma następujące właściwości:

- łącze jest ustawiane tylko wtedy, gdy dzuelą one jakąś skrzynkę

- łącze może być związane z wieloma procesami

- każda para procesów może mieć kilka różnych łączy, z któych każde odpowiada jakiejś skrzynce

Skrzynka może być własnością procesu albo systemu. Jeśli skrzynka należy do procesu, to rozróżnia się jej właściciela (który może tylko odbierać komunikaty) i użytkownika (który może tylko nadawać komunikaty)

Skrzynka należąca do systemu operacyjnego istnieje ez inicjatyw procesu, Jest niezależna i nie przydziela się jej do żadnego konkretnego procesu. OS dostarcza mechanizmów pozwalających na tworzenie mowej skrzynki, nadawanie i odbieranie komunikatów i likwidowanie skrzynki.

Proces, na którego zamówienie jest tworzona nowa skrzynka, staje się jej właścicielem. Początkowo właściciel jest jedynym procesem, który może odbierać komunikaty. Przywilej własności i odbierania komunikatów może zostać przekazany innym procesom za pomocą odpowiedniejfunkcji systemowej.

Łącze ma pewną pojemność określającą liczbę komunikatów, które mogą w nim czasowo przebywać, Cechata może być przestrzgana jako kolejka komunikatów przypisanych do łącza.

Są trzy podstawowe metody implementacji takiej kolejki:

- pojemność zerowa: Maksymalna długość kolejki wynosi 0, czyli łącze nie dopuszcza by czekałw nim jakikolwiek komunikat. W tym przypadku nadawca musi czekać aż odbiorca odbierze komunika. Aby moża było przysłać komunikaty, oba procesy muszą być zsynchronizowane.

- pojemność ograniczona: Kolejka ma skończoą długość n, może w niej pozostawać co najwyżej komukoatów, Jeśli w chwili nadania komunikaty koleja nie jest pełna to nowy komunikat zostaje w niej umieszczony i nadawca może kontynuować działanie bez czekania. Jeśli kolejka jest pełna nadawca musi czekać ażzwolni się miejsce.

1. Buforowanie

------------------

1. Wzajemne wykluczanie

Przynajmniej jeden zasób musi być niepodzielny czyli tego zasobu może używać w danym czasie tylko jeden proces.

Jeśli inny proces zamawia dany zasób to musi być opóźniany do czasu aż zasób zostanie zwolniony

1. Przetrzymywanie i oczekiwanie

Musi istnieć proces któremu przydzielono co najmniej jeden zasób i który oczekuje na przydział dodatkowego zasobu przetrzymywanego właśnie przez inny proces

1. Brak wywłaszczeń

Zasoby nie podlegają wywłaszczaniu co oznacza że zasób może zostać zwolniony tylko z inicjatywy przetrzymującego go procesu, po zakończeniu pracy tego procesu.

1. Czekanie cykliczne

Musi istnieć zbiór {P1, P2 … Pn} czekających procesów takich że P1 czeka na zasób przetrzymywany przez proces P2, P2 czeka na zasób przetrzymywany przez P3 … Pn-1 czeka na zasób Pn a Pn czeka na zasób przetrzymywany przez proces P1

Lekcja

Temat: Likwidacja zakleszczenia

* 1. Można poinformować operatora o wystąpieniu zakleszczenia i pozwolić mu na ręczne jego usunięcie
  2. Inną możliwością jest pozwolenie systemowi, aby automatycznie usuwał zakleszczenia
  3. Są dwa sposoby likwidowania zakleszczenia:
* Pierwszy to usunięcie jednego lub kilku procesów w celu przerwania czekania cyklicznego.
* Drugi sposób polega na odebraniu pewnych zasobów jednemu lub kilku procesom których dotyczy zakleszczenie ( poprzez wywłaszczenie)
  1. Aby zlikwidować zakleszczenie przez zaniechanie procesu używa się jednej z dwóch metod. W obu metodach system odzyskuje wszystkie zasoby przydzielone zakończonym procesom:
* Zaniechanie wszystkich zakleszczonych procesów:

1. Metoda ta likwiduje cykl zakleszczenie, lecz ponoszone są znaczne koszty (najbardziej chodzi o czas)

* Usuwanie procesów pojedynczo, aż do wyeliminowania cyklu zakleszczenia:

1. Ta metoda wymaga sporego nakładu pracy na przetwarzanie wykonania algorytmu wykrywania zakleszczenia po każdym usunięciu procesu w celu sprawdzenia czy pozostałe procesy nadal są zakleszczone.
   1. W celu usunięcia zakleszczenia za pomocą wywłaszczania zasobów stopniowo odbiera się pewne zasoby jednym procesom i przydziela się innym, aż cykl zakleszczenia zostanie przerwany.
   2. Jeśli do zwalczania zakleszczeń stosuje się wywłaszczenia, to należy uwzględnić następujące kwestie:

* Wybór ofiary:

Należy wybrać które zasoby i które procesy mają ulec wywłaszczeniu. Porządek wywłaszczeń powinien minimalizować koszty.

* Wycofanie:

Trzeba wycofać proces do bezpiecznego stanu, z którego będzie go można wznowić. Najprostszym rozwiązaniem jest całkowite wycofanie i późniejsze wykonanie go od początku.

* Głodzenie:

W systemie. W którym wybór ofiary opiera się przede wszystkim na ocenie kosztów, może się zdarzyć, że w roli ofiary będzie wybierany wciąż ten sam proces.

W skutek tego proces nigdy nie zakończy swojej pracy – jest to zjawisko głodzenia, Najprostszym rozwiązaniem jest uwzględnienie liczny wycofań przy ocenie kosztów