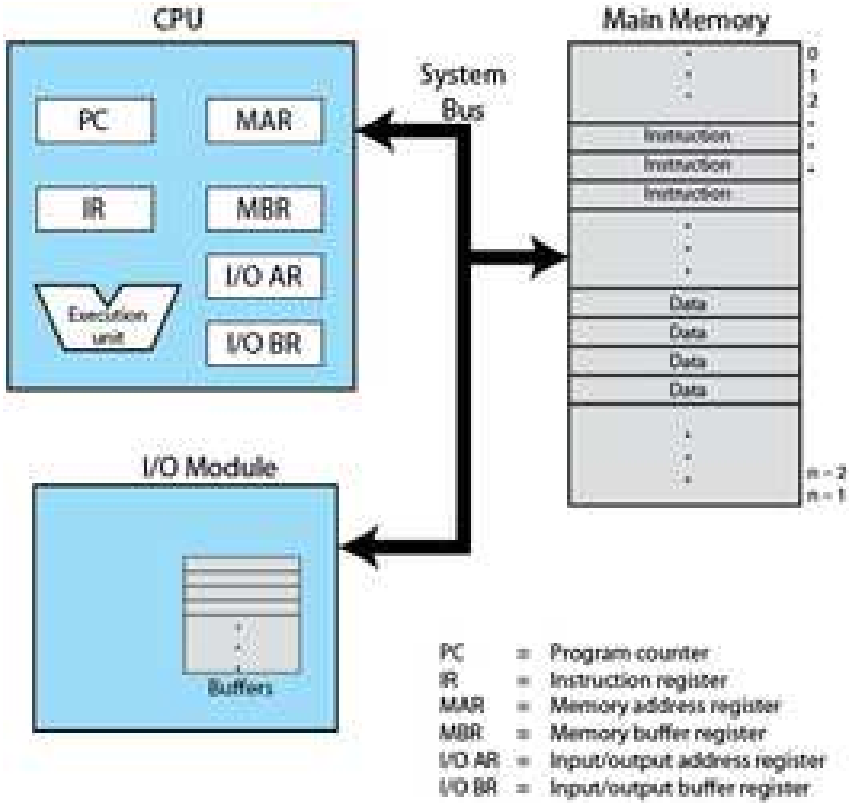
Wieloprogramowy system komputerowy

- sprzęt: procesor(y), pamięć, łącza i magistrale komunikacyjne, urządzenia wejścia/wyjścia
- system operacyjny: obsługuje i zarządza sprzętem, umożliwia pracę i korzystanie ze sprzętu programom użytkowym
- programy użytkowe: wykonują obliczenia korzystając z przydzielonych zasobów komputera: procesora, pamięci, urządzeń wejścia/wyjścia
- użytkownicy (opcjonalnie): programiści, administratorzy, zwykli użytkownicy

Ogólnie system operacyjny ma za zadanie umożliwić, ułatwić, i maksymalnie usprawnić korzystanie z takiego systemu komputerowego programom aplikacyjnym i użytkownikom (jeśli są).

Procesor

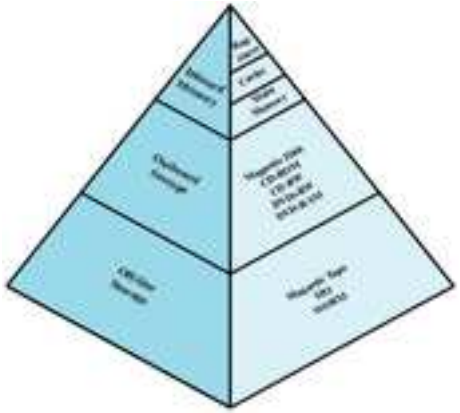
- Rejestry
  - dostępne dla (programu) użytkownika
    - rejestry adresowe
    - rejestry danych
  - systemowe
    - licznik rozkazów (PC)
    - rejestr rozkazów (IR)
- Cykl wykonywania rozkazów
  - pobranie
  - dekodowanie
  - wykonanie
- Przerwania
  - generowane przez zegar procesora
  - generowane przez procesory wejścia/wyjścia
  - generowane przez jednostki kontrolujące (błędy procesora, pamięci)
  - przetwarzanie przerwania — przerwania wielopoziomowe



Pamięć

Istnieje wiele rodzajów pamięci, różniących się od siebie parametrami czasu dostępu, technologią (np. pamięć trwała czy ulotna), kosztu, sprawności energetycznej, i in. Zapotrzebowanie na pamięć w systemie komputerowym ma preferencje do:

- uzyskiwania dużych pojemności,
- uzyskiwania krótkich czasów dostępu,
- obniżania zużycia energii,
- wydłużania czasu zachowania danych,
- obniżenia kosztu w przeliczeniu na jednostkę danych.



Jako uniwersalne rozwiązanie stosuje się wiele warstw pamięci, składających się na tzw. **piramidę pamięci**. Na szczycie piramidy są jednostki bardzo szybkiej pamięci (rejestry i pamięć podręczna *cache*) pozwalającej procesorowi pracować wydajnie, bez nadmiernych opóźnień. Poniżej są pamięci pozwalające przechowywać większą ilość danych przez nieco dłuższy czas. Na spodzie piramidy są urządzenia pamięci zewnętrznej, o większych czasach dostępu, lecz mniejszym koszcie i zużyciu energii.

Taka struktura godzi w pewnym stopniu sprzeczne wymagania pamięciowe dzięki tzw. **zasadzie lokalności**. Dla zapewnienia współpracy wszystkich pięter piramidy pamięci stosowane są sterowniki obsługujące automatyczną wymianę danych między piętrami.

Urządzenia wejścia/wyjścia

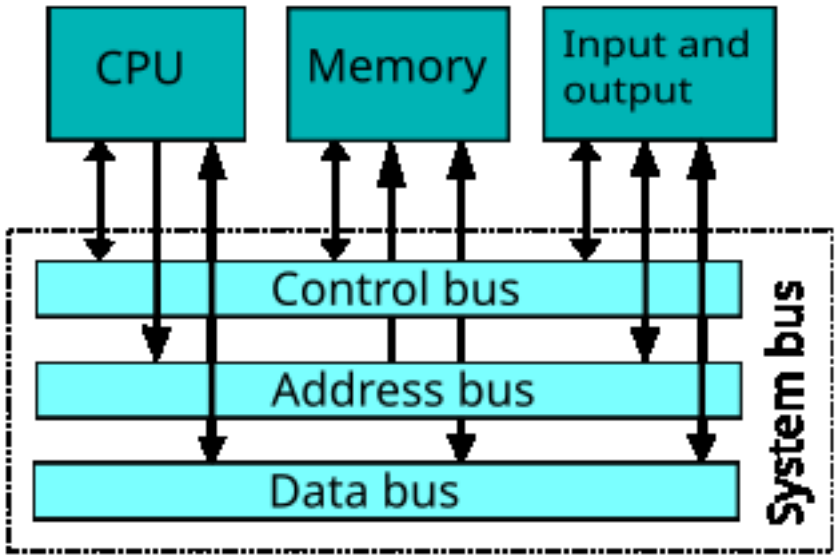
Wszystkie programy korzystają z urządzeń i operacji wejścia/wyjścia (inaczej nie byłoby żadnych efektów ich działania). Podstawowym urządzeniem I/O jest system dyskowy, który tradycyjnie w systemach operacyjnych pełni rolę urządzenia systemowego. W nim przechowywane są wszystkie pliki systemowe, z niego system ładuje się przy starcie (zatem musi być dostępny również dla elektroniki systemu startowego), w nim przechowują dane użytkownicy.

W odniesieniu do systemowych urządzeń I/O system operacyjny pełni rolę interfejsu. Sam korzysta z tych urządzeń, a wykonującym się zadaniom udostępnia te urządzenia w postaci usług, np. tworzenia i operacji na plikach dyskowych. (Innym typowym urządzeniem systemowym są interfejsy sieciowe.)

Dalsze rodzaje urządzeń I/O to kanały komunikacji użytkowników: urządzenia wyświetlające, terminale, klawiatury, itp. W odniesieniu do tych urządzeń system operacyjny typowo pełni rolę przydziału i rezerwacji.

Magistrale komunikacyjne

Do połączenia procesora z systemem pamięci, oraz modułami wejścia/wyjścia istnieją w systemach komputerowych łącza zwane **magistralami** (*bus*). Ideą magistrali jest połączenie wszystkich urządzeń wspólnym medium komunikacyjnym, z koniecznością kontroli dostępu do niego. Zajmuje się tym oddzielny moduł zarządzający magistralą.

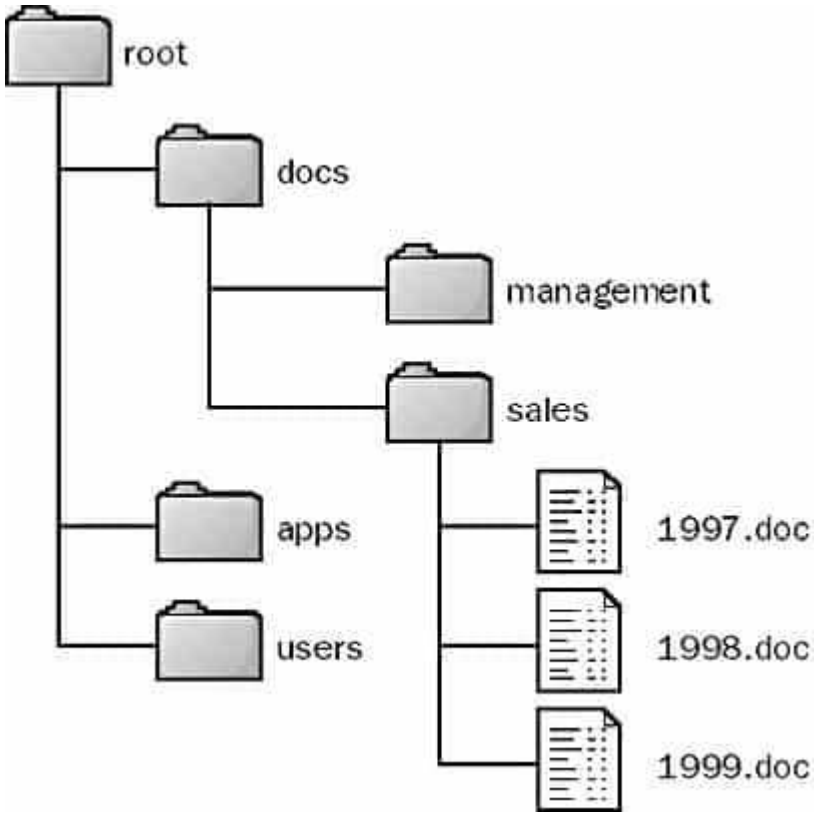


Magistralą zarządza oddzielny moduł zezwalający na dostęp do niej. W rozbudowanych systemach komputerowych może istnieć kilka/wiele magistral pełniących różne funkcje i obsługujących różne moduły. Ich niezależna i równoległa praca zwiększa możliwość efektywnego przetwarzania przez system wielu programów jednocześnie.

Systemy plików

Systemy komputerowe przetwarzają duże ilości danych zorganizowanych w **pliki**. Gdy system posiada wiele plików sensowne jest zorganizowanie ich w **systemy plików**.

Obsługa systemów plików jest kolejną funkcją realizowaną przez system operacyjny, dzięki czemu programy mają zapewniony wygodny i efektywny dostęp do potrzebnych im danych.



System plików stanowi warstwę abstrakcji uwalniającą programy użytkownika od bezpośredniego zarządzania urządzeniami pamięci trwałej (dyskami) przez wprowadzenie takich obiektów jak:

- pliki,
- nazwy, prawa dostępu, i inne atrybuty plików,
- katalogi plików,
- hierarchiczne drzewo katalogów,
- itp.

Sieci komputerowe

Sieci komputerowe pojawiły się w systemach komputerowych w latach 70-tych XX-ego wieku, gdy architektura systemów operacyjnych była już mocno ukształtowana. Przez szereg lat technologie sieciowe były rozwijane i integrowane z systemami operacyjnymi, często kosztem gwałtownych i gruntownych zmian, powodujących wiele problemów użytkownikom komputerów.

Jednak w dzisiejszych czasach systemy komputerowe pozbawione dobrodziejstw połączenia z siecią komputerową stanowią rzadkość.

Można patrzeć na sieć komputerową jako na jeden z zasobów, z których mogą korzystać programy i użytkownicy komputera, oraz jako jedną z technologii, które może wykorzystywać sam system komputerowy.

Ten drugi punkt widzenia jest związany z technologiami rozproszonych systemów komputerowych.

W związku z integracją systemów operacyjnych z oprogramowaniem sieciowym pojawiły się pojęcia **sieciowych systemów operacyjnych** oraz **rozproszonych systemów operacyjnych**. Pierwszy oznacza system dostarczający użytkownikom usługi sieciowe, a drugi system integrujący zasoby sieciowe i zasoby lokalne, prezentujące je użytkownikom w spójny sposób.

Rozproszone systemy komputerowe

Zalety systemów rozproszonych:

- współdzielenie zasobów: file-serwery, bazy danych, specjalistyczne urządzenia (np. drukarki),
- przyspieszenie obliczeń, zmniejszenie czasu reakcji,
- możliwość rozproszenia obliczeń, również automatycznie (*load balancing*),
- niższy koszt systemu przy wzrastającym obciążeniu,
- możliwość budowania zwiększonej niezawodności

Systemy komputerowe i systemy operacyjne — historia

I generacja 1945–1955 / lampy elektronowe — brak systemu operacyjnego

proste obliczenia, programowanie za pomocą tablicy połączeń kablowych

II generacja 1955–1965 / tranzystory — systemy wsadowe

algorytmiczne języki programowania (Fortran, Cobol, Lisp), obliczenia naukowe i inżynierskie, czytniki kart i taśm, zadania przetwarzane po jednym na raz, rola systemu operacyjnego (np. FMS) sprowadzała się do odczytania opisu zadania

III generacja 1965–1980 / układy scalone — systemy wieloprogramowe

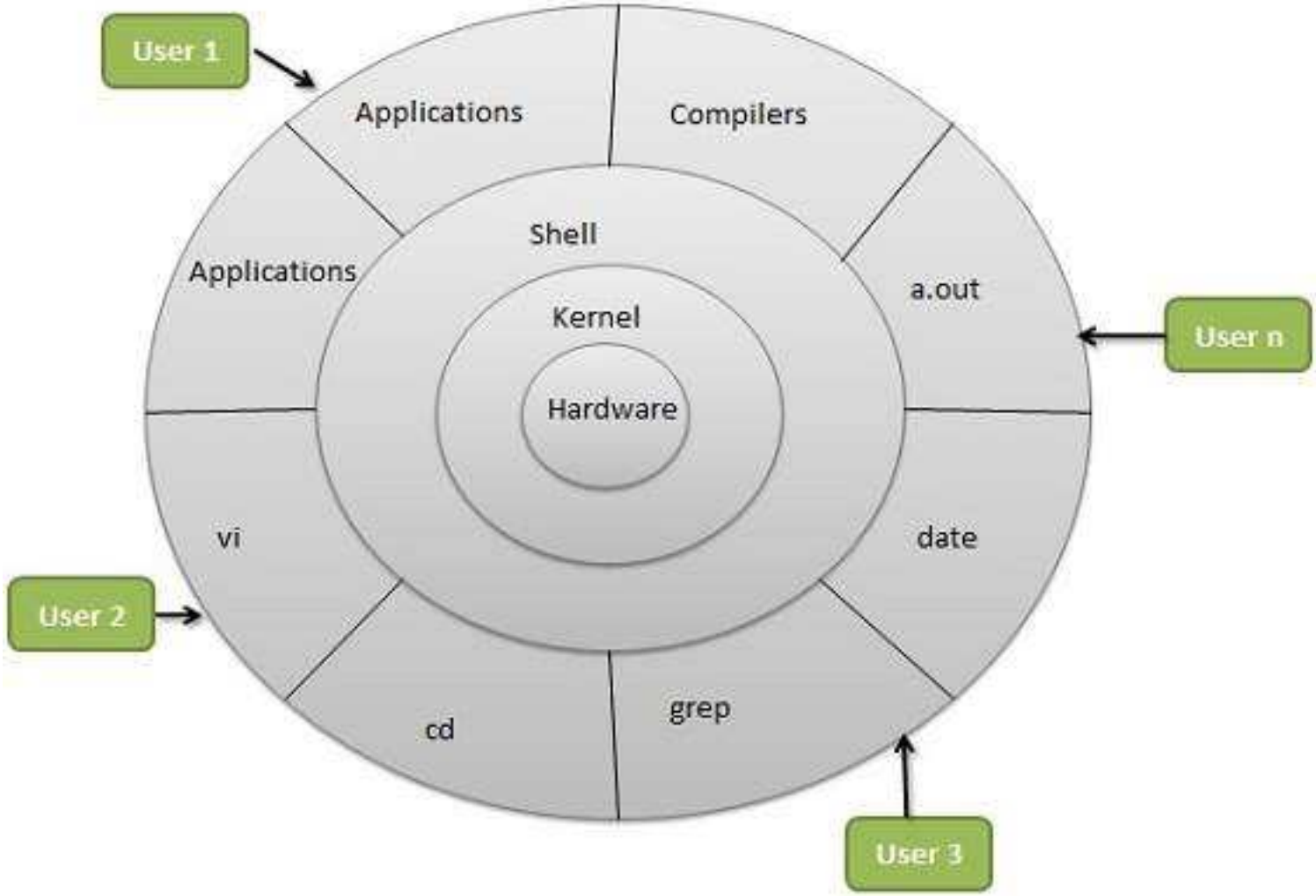
OS/360 (IBM) pozwalał na naprzemienne wykonywanie kilku programów umieszczonych jednocześnie w pamięci, spadek cen komputerów, ogólny rozwój technologii cyfrowej

IV generacja 1980–teraz / układy VLSI, mikroprocesory — systemy obecne

duże systemy komputerowe (*mainframe*) i systemy operacyjne z nimi związane: MULTICS, VM/CMS (OS/370), oraz minikomputery, system Unix, a potem komputery osobiste; nowe technologie: sieci komputerowe, graficzne interfejsy użytkownika (GUI),

Architektura systemu operacyjnego

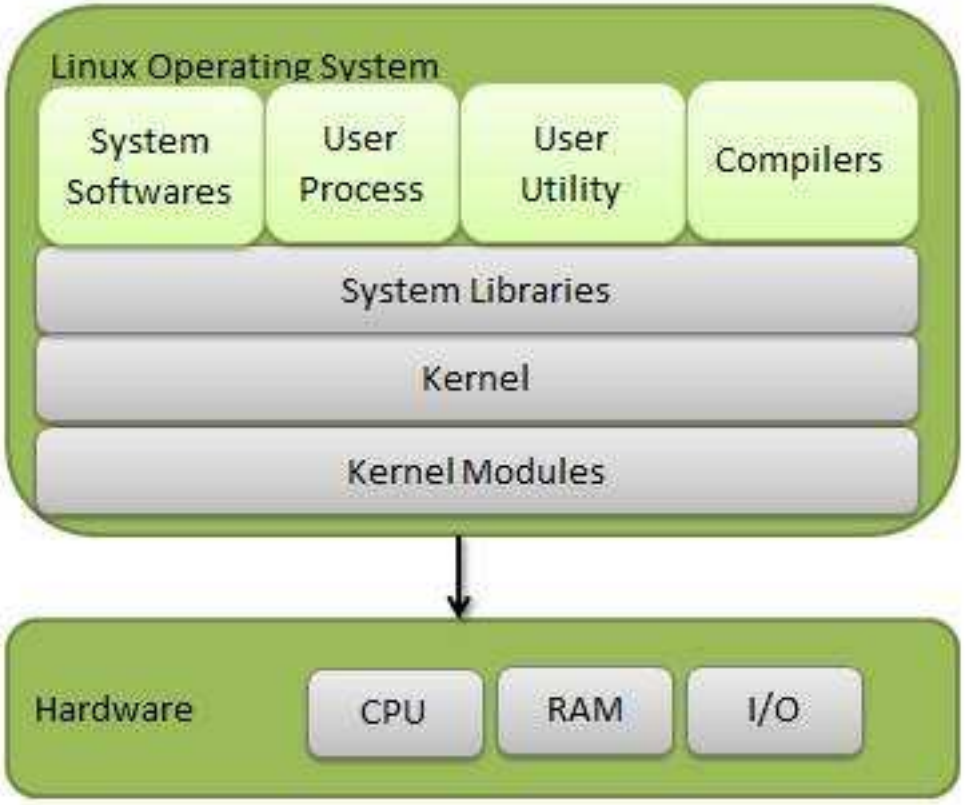
Często prezentowany jest następujący obrazek ilustrujący architekturę systemu operacyjnego jako składającego się z **jądra systemu** (*kernel*), obsługującego sprzęt, i **powłoki** (*shell*) stanowiącej interpreter poleceń umożliwiający uruchamianie programów aplikacyjnych oraz interakcyjną pracę użytkowników:



Architektura systemu operacyjnego (2)

W rzeczywistości jednak architektura systemu wygląda nieco inaczej. Jądro systemu składa się zarówno ze sterowników urządzeń, zasadniczego kodu realizującego funkcje systemu, jak również biblioteki funkcji umożliwiających programom aplikacyjnym dostęp do usług systemu.

Natomiast w skład systemu operacyjnego wchodzi również szereg programów dodatkowych, realizujących dodatkowe funkcje. Do nich należą właśnie interpretery poleceń, ale również programy komunikacyjne, edytory, kompilatory, system drukowania, itp.



Funkcje systemu operacyjnego

- **wygoda** użytkowania komputera: dostęp, konfiguracja, przechowywanie programów, danych, itp., uruchamianie programów w zależności od potrzeb
  - w szczególności: wygoda programowania — biblioteki przydatnych funkcji
- **efektywność** wykorzystania zasobów: procesor, pamięć, kanały wejścia/wyjścia
- **usługi** systemowe: dostęp do funkcji niedostępnych dla programów w ramach języka programowania, np. informacja o dacie/czasie, dostęp do urządzeń peryferyjnych, arbitraż dostępu (zapobieganie konfliktom), itp.
- **zabezpieczenie** jednych elementów komputera i jego oprogramowania przed niepoprawnym dostępem/nadużyciem przez inne elementy, bądź wskutek błędu, bądź świadomego działania
- **„rozwojowość”:** umożliwianie tworzenia nowych mechanizmów i funkcji, bez przerw i zakłóceń w normalnej pracy systemu

System operacyjny — dwie perspektywy

Można postrzegać system operacyjny z różnych punktów widzenia:

***top-down*: jako warstwę abstrakcji** — aby program/użytkownik systemu mógł posługiwać się np. abstrakcją plików i katalogów danych, albo abstrakcją komunikacji sieciowej za pomocą protokołu wysokiego poziomu, a nie musiał bezpośrednio sterować dyskiem albo kartą sieciową,

Wykonywanie programów też wykorzystuje pewne abstrakcje, a mianowicie wykonywanie programu napisanego w języku programowania wysokiego poziomu, zamiast sterowania rejestrami procesora; do budowy tej abstrakcji należą takie narzędzia jak interpretery, kompilatory, linkery dynamiczne, i systemy *run-time* odpowiednich języków programowania, które z tego punktu widzenia stanowią część systemu operacyjnego.

***bottom-up*: jako zarządcę zasobów** — system komputerowy składa się z wielu różnych podzespołów: procesorów, pamięci, zegarów, timerów, interfejsów, urządzeń peryferyjnych, itp., i system operacyjny ma za zadanie zarządzać nimi w taki sposób, aby cały system poprawnie wykonywał swoje zadania

Budowa systemu operacyjnego

Istnieje szereg możliwości konstrukcji systemu operacyjnego jako programu zarządzającego komputerem:

system monolityczny — system operacyjny jest jednolitym programem, kompilowanym i konsolidowanym ze wszystkimi sterownikami

jest to najwcześniejsza historycznie i najprostsza metoda budowy systemu

system modularny — system operacyjny składa się z głównego programu, do którego mogą się dynamicznie ładować moduły w razie potrzeby; np. sterowniki urządzeń, gdy takie urządzenie zostanie włączone w trakcie pracy systemu

jest to obecnie typowa budowa systemów operacyjnych komputerów

system oparty na mikrojądrze — system operacyjny składa się z głównego programu, któremu towarzyszą inne programy realizujące część funkcji systemu, ale uruchamiane jako oddzielne procesy; dzięki temu jądro systemu jest mniejsze (mikrojądro), a dodatkowe procesy mogą podlegać szeregowaniu, wywłaszczaniu, itp. w razie potrzeby

tego typu konstrukcja stosowana jest w wielu systemach czasu rzeczywistego i systemach wbudowanych

Specyfika systemów operacyjnych

W systemach operacyjnych występują pewne specyficzne zagadnienia, które są kluczowe dla ich funkcjonowania:

wieloprogramowość i wielozadaniowość — zdolność wykonywania wielu zadań „jednocześnie”

przydział zasobów — zarządzanie zasobami sprzętowymi komputera oraz zasobami programowymi systemu, i przydział tych zasobów użytkownikom systemu

szeregowanie zadań — praktyczna realizacja wielozadaniowości i jednocześnie przykład funkcji przydziału zasobów

wywołania systemowe — udostępnianie programom aplikacyjnym określonych usług

dwa tryby wykonywania — uprzywilejowany tryb jądra i bezpieczny tryb użytkownika

zabezpieczenia — dla zapewnienia bezawaryjnej pracy całego systemu

Wieloprogramowość i wielozadaniowość

Starsze systemy komputerowe miały pojedynczy procesor, na którym mógł wykonywać się tylko jeden program na raz. Jednak pojawił się mechanizm **wieloprogramowości** pozwalający na wykorzystania czasu procesora, gdy wykonujący się program wywołał operację I/O i musiał czekać na jej zakończenie. Zdolność załadowania do pamięci kilku programów pozwalała wykorzystać czas procesora przez przełączenie się na wykonywanie innego programu. Wieloprogramowość nie oznacza jednak zarządzania czasem wykonywania się programów. Pojedynczy program mógł wykonywać się dowolnie długo, co mogło prowadzić do **zagłodzenia** innych programów.

W późniejszych systemach coraz ważniejsza stała się obsługa wielu użytkowników, którzy nie chcieli czekać na siebie nawzajem. Pojawiło się zapotrzebowanie na **wielozadaniowość**, czyli efektywne wykonywanie wielu zadań na raz, nawet jeśli system komputerowy nie posiadał dostatecznie wielu procesorów lub rdzeni, żeby wszystkie zadania rzeczywiście liczyły się jednocześnie.

Zatem wielozadaniowość nie oznacza rzeczywiście jednoczesnego, czyli równoległego, wykonywania wszystkich zadań. Zamiast tego system realizuje ich ***quasi*-równoległe** wykonywanie, co oznacza wykonywanie po kolei wybranych zadań przez mały **kwant** czasu. Ponieważ dzieje się to w kółko i powtarzalnie, z punktu widzenia tych zadań są one wykonywane (jakby) ciągle, chociaż odpowiednio wolniej.

Szeregowanie

Prawie wszystkie współczesne systemy komputerowe są wielozadaniowe, to znaczy, że jednocześnie wykonują wiele zadań, zwanych czasami procesami lub wątkami. Jednak zarazem prawie wszystkie współczesne systemy komputerowe mają więcej zadań do jednoczesnego wykonywania, niż mają procesorów lub rdzeni.

Koncepcja quasi-równoległego wykonywania jest dobra i powszechnie stosowana. Jednak wybór zadań, które powinny być w ten sposób wykonywane nie jest oczywisty i zależy od strategii systemu i ważności oraz pilności poszczególnych zadań. Ta procedura wyboru nazywa się **szeregowaniem**, albo **planowaniem**, i stanowi jedną z ważnych czynności wykonywanych przez system operacyjny.

Pojęcie **szeregowania** pochodzi z wcześniejszych jednoprocesorowych architektur komputerowych, gdzie procesy musiały jakby czekać na swoją kolejność wykonania. Jednak we współczesnych systemach wieloprocesorowych/wielordzeniowych, nadal występuje czekanie i kolejkowanie.

Wywołania systemowe

Zbiór wywołań systemowych stanowi interfejs (API) udostępniający programom aplikacyjnym funkcje, których nie jest w stanie zapewnić język programowania, a może zapewnić jedynie system operacyjny.

Przykładem może być dostęp do informacji o czasie rzeczywistym. Innym przykładem może być mechanizm wzajemnego wykluczania pewnych operacji. Ze względu na szeregowanie zadań, programy aplikacyjne nigdy nie mogą mieć pewności w jakiej kolejności operacje różnych zadań będą wykonywane. Programy działające w przestrzeni użytkownika nie są w stanie niezawodnie zrealizować tego typu operacji.

Tryb jądra i tryb użytkownika

System operacyjny zarządza pracą całego systemu komputerowego. Aby zapewnić mu zdolność zapanowania nad sprzętem oraz programami aplikacyjnymi, musi on mieć pewne „ustawowe zwierzchnictwo" nad tymi programami. (Czasami mówi się o realizowaniu funkcji policyjnej.)

Takim mechanizmem jest podwójny tryb pracy: uprzywilejowany tryb jądra, z pełnym dostępem do warstwy sprzętowej, oraz tryb użytkownika, bezpieczny, bez pełnego dostępu do, i prawa programowania urządzeń.

Praca w tych dwóch trybach jest realizowana przez procesor, który wspiera przełączanie trybów pracy. Dzięki temu jądro systemu operacyjnego jest w stanie zagwarantować, że programy aplikacyjne nie są w stanie samodzielnie przejść do trybu jądra. Mogą to zrobić jedynie za pomocą odpowiednich wywołań systemowych, które zapewniają im potrzebne im usługi.

Zabezpieczenia

Mechanizmy zabezpieczeń w systemach operacyjnych:

- zabezpieczenie systemu operacyjnego przed wykonaniem niepożądanych operacji przez programy użytkowe,
- zabezpieczenie wykonujących się programów przed błędnym działaniem innych programów,
- udostępnienie mechanizmów zabezpieczeń użytkownikom dla bezpiecznego przechowywania ich zasobów (plików),
- zabezpieczenie całego systemu i jego zasobów przed atakami.

Systemy czasu rzeczywistego

System czasu rzeczywistego RTS (*Real-Time System*) to system w którym można określić maksymalny czas wykonania poszczególnych operacji.

W systemach RTS poprawność procesu obliczeniowego zależy nie tylko od samego wyniku, ale również od czasu, w którym został on osiągnięty.

Systemy RTS są często **wbudowane** w urządzenia. Przykładowe dziedziny zastosowań: sterowanie produkcją, przemysł samochodowy, aparatura medyczna, systemy komunikacyjne, odtwarzacze multimedialne.

Systemy czasu rzeczywistego mają odmienne wymagania od zwykłych systemów komputerowych. Kluczowym wymaganiem, poza działaniem w czasie rzeczywistym, jest niezawodność. Są zatem budowane w sposób oszczędny, minimalistyczny. Często są to systemy małe, energooszczędne.

Systemy operacyjne czasu rzeczywistego (RTOS) muszą nie tylko dostarczać mechanizmów i usług dla wykonywania, planowania, i zarządzania zasobami aplikacji, ale muszą również same sobą zarządzać w sposób oszczędny, przewidywalny, i niezawodny.

Krótkie podsumowanie — pytania sprawdzające

Odpowiedz na poniższe pytania:

1. Jaka jest rola i zadania systemu operacyjnego?

2. Jakie zadania pełni system operacyjny wobec procesów?
   Pamięci RAM?
   Urządzeń wejścia/wyjścia?

3. W jakim celu/z jakiego powodu stosowana jest piramida pamięci?

4. Czym różni się wieloprogramowość od wielozadaniowości?

5. Co oznacza *quasi*-równoległe wykonywanie zadań?

6. Co oznacza szeregowanie zadań?

7. Czym różnią się wywołania systemowe od zwykłej biblioteki funkcji?

8. W jaki sposób system operacyjny wykorzystuje dwa różne tryby pracy procesora?

9. Jakie mechanizmy zabezpieczeń implementuje system operacyjny?