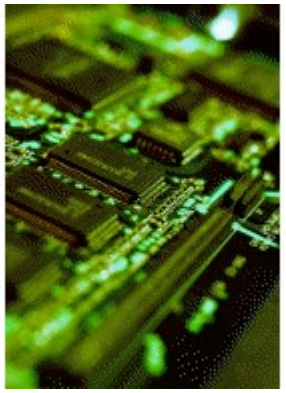
Systemy Operacyjne

(studia zaoczne)



Sprzęt komputerowy

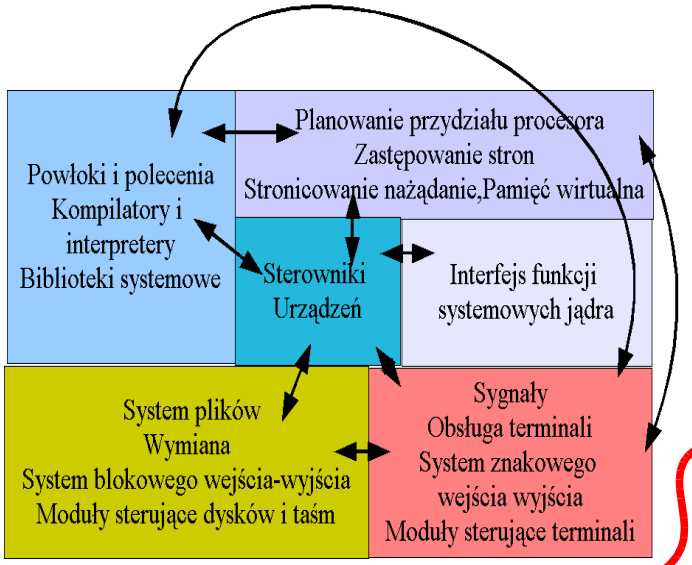
System Operacyjny
+
Programy

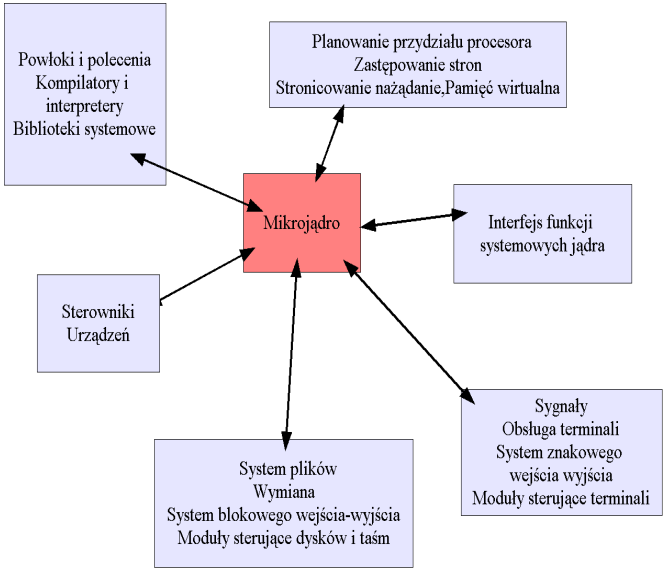
Łatwe użytkowanie

Prowadzący: *Robert Szmurło*
szmurlor@iem.pw.edu.pl
GE 229

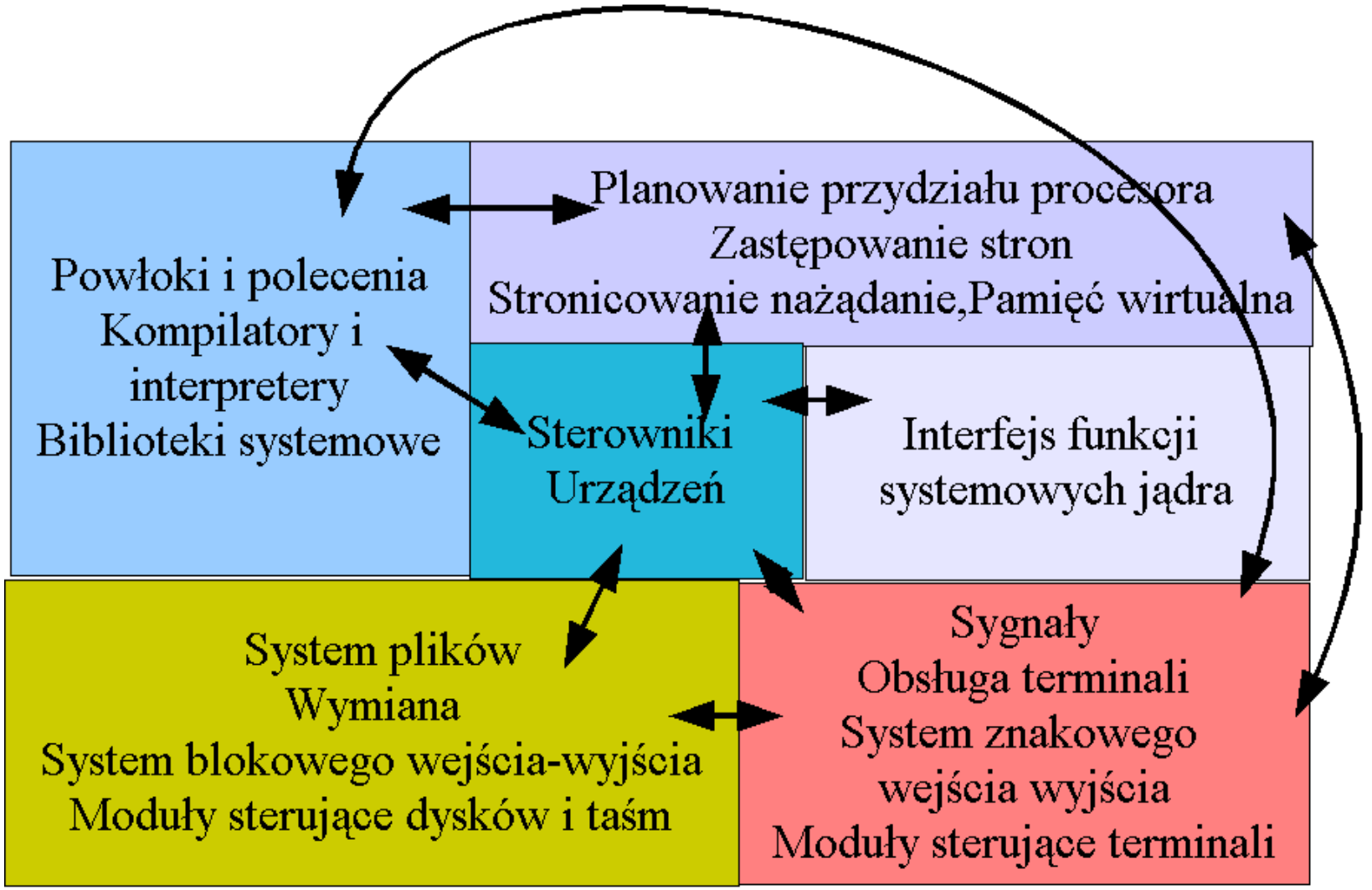
Architektury SO

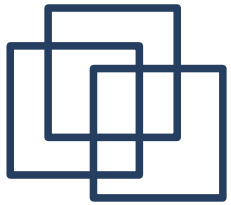
- Jednolita
- Warstwowa
- Klient-Serwer



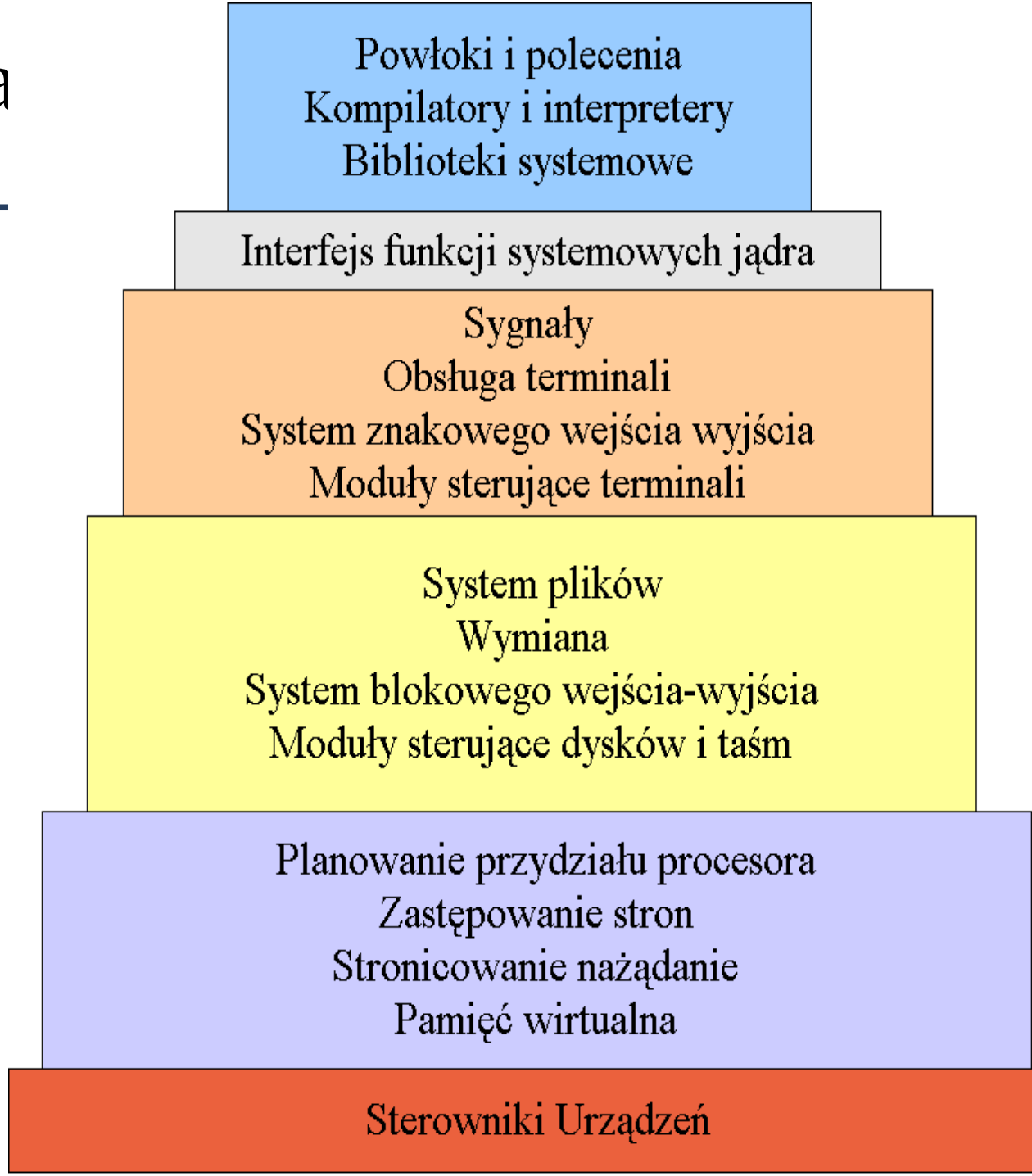


Jednolita

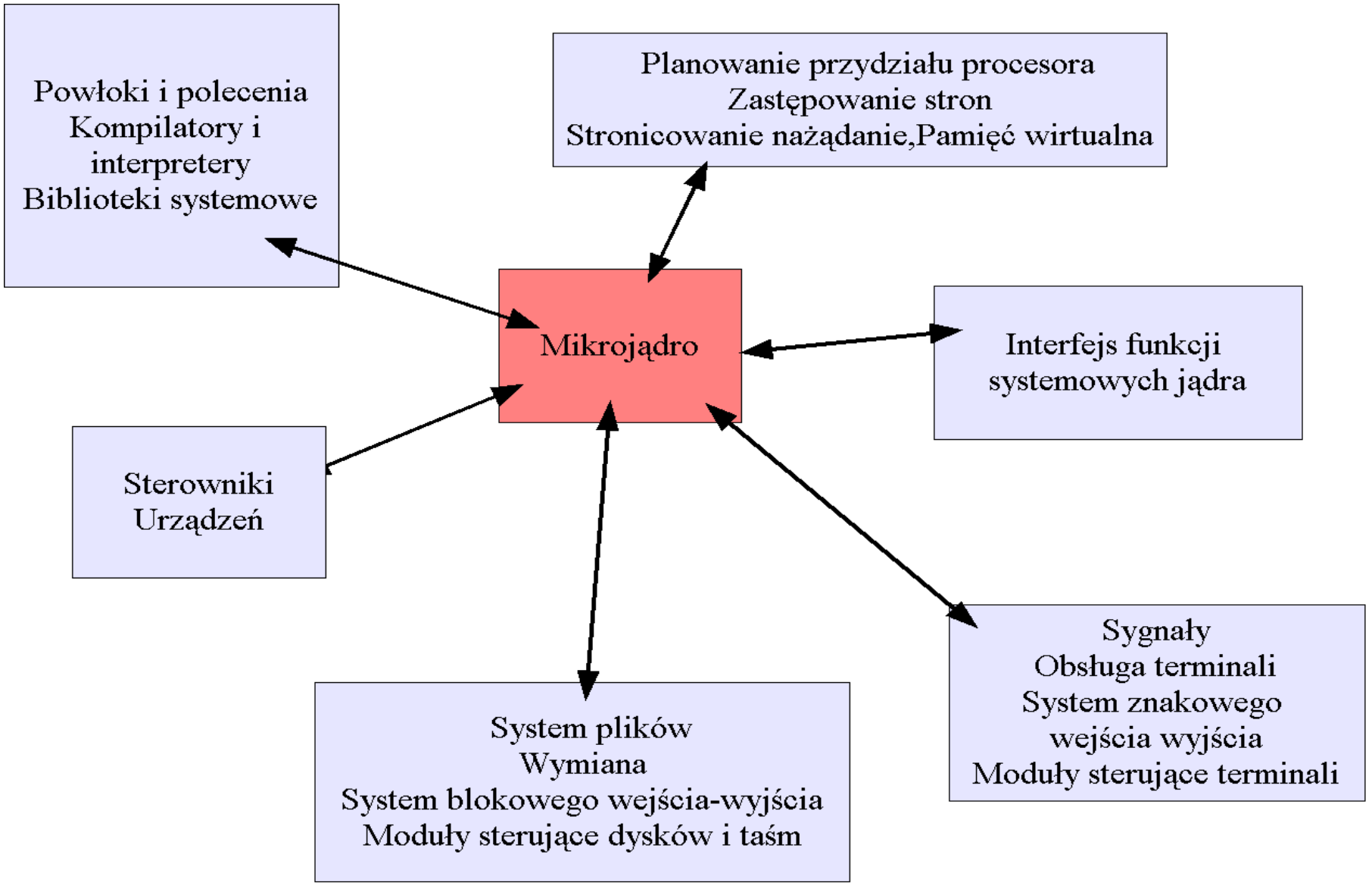

Powłoki i polecenia
Kompilatory i
interpretery
Biblioteki systemowe
Planowanie przydziału procesora
Zastępowanie stron
Stronicowanie nażądanie,Pamięć wirtualna
Sterowniki
Urządzeń
Interfejs funkcji
systemowych jądra
System plików
Wymiana
System blokowego wejścia-wyjścia
Moduły sterujące dysków i taśm
Sygnały
Obsługa terminali
System znakowego
wejścia wyjścia
Moduły sterujące terminali



Warstwowa

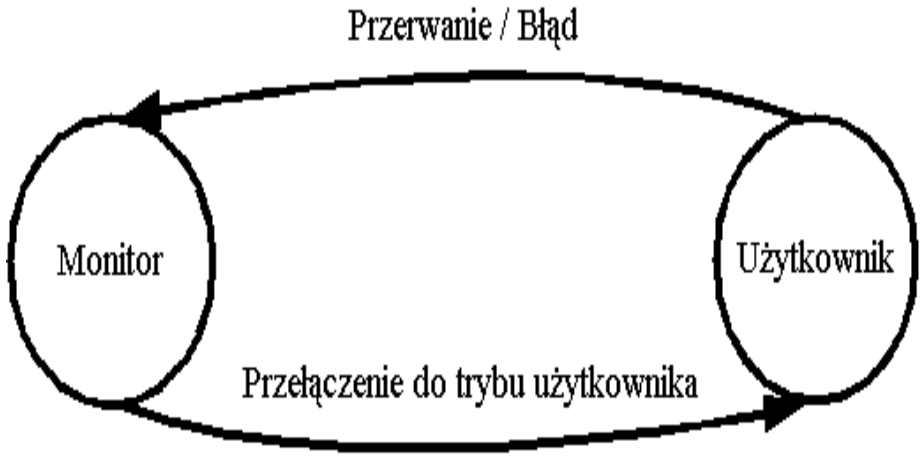

Powłoki i polecenia
Kompilatory i interpretery
Biblioteki systemowe
Interfejs funkcji systemowych jądra
Sygnały
Obsługa terminali
System znakowego wejścia wyjścia
Moduły sterujące terminali
System plików
Wymiana
System blokowego wejścia-wyjścia
Moduły sterujące dysków i taśm
Planowanie przydziału procesora
Zastępowanie stron
Stronicowanie nażądanie
Pamięć wirtualna
Sterowniki Urządzeń

Klient-serwer



Ochrona sprzętowa i dualny tryb pracy

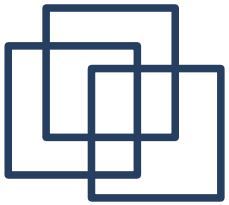
- Współdzielenie zasobów wymaga od systemu operacyjnego zapewnienia, że nieprawidłowo działający program nie spowoduje błędów w innym programie.



- Sprzęt (procesor) musi udostępniać co najmniej dwa podstawowe tryby pracy:
  - Tryb użytkownika (user mode)
  - Tryb nadzorcy, monitora (supervisor mode)
  - Sprzęt posiada przełącznik bitowy: monitor (1), użytkownik (0).
  - Instrukcje uprzywilejowane mogą być uruchamiane tylko w trybie monitora (nadzorcy).

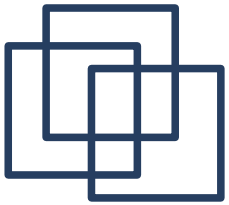
Tryb Chroniony / Rzeczywisty

- Mikroprocesor(y) Intela potrafią pracować w trybie:
  - Chronionym – z zabezpieczeniami (użytkownik)
  - Rzeczywistym – bez zabezpieczeń (system)
- Po uruchomieniu komputera w architekturze intela procesor znajduje się w trybie rzeczywistym. Dopiero system operacyjny przełącza procesor do trybu chronionego.
- W trybie chronionym:
  - ochrona zasobów komputera (przerwania, porty)
  - ochrona pamięci (I/O permission bitmaps)
  - wywłaszczanie procesów (wielozadaniowość)
  - pamięć wirtualna (stronicowanie)



Procesy

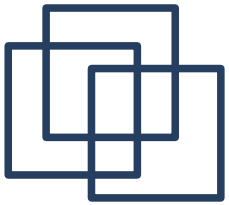
- **Proces jest elementarną jednostką pracy (aktywności) zarządzaną przez system operacyjny, która ubiega się o zasoby systemu komputerowego w celu wykonania programu.**
- Proces - program w trakcie wykonywania, który do wykonania określonego zadania potrzebuje pewnych zasobów: procesor, pamięć, pliki, urządzenia wejścia-wyjścia (klawiatura,ekran,skaner,karta sieciowa,port szeregowy lub równoległy itp.)
- Synonimami procesu, które są stosowane w literaturze są: zadanie (task) lub praca (job).
  - Zadanie – odnosi sie żazwyczaj do systemów wsadowych, w danej chwili może być wykonywane tylko jedno,
  - Praca - systemy z podziałem czasu (czas wykorzystania zasobów w tym procesora) jest dzielony na wiele prac (multitasking).



Elementy składowe procesu

- program — definiuje zachowanie procesu, („*Proces jest czymś więcej niż samym kodem programu (sekcją tekstu - text section)*“).
- dane — zbiór wartości przetwarzanych oraz wyniki,
  - Stos procesu (przechowuje dane tymczasowe).
  - Sterta - Sekcja danych (zawiera zmienne globalne).
- zbiór zasobów tworzących środowisko wykonawcze, (np. zawartość rejestrów procesora.)
- blok kontrolny procesu (PCB, deskryptor) — opis
- bieżąca czynność reprezentowana przez wartość licznika rozkazów. (rejestr PC – program counter).

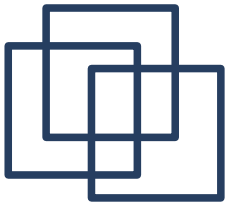
- ***Program jest obiektem pasywnym, natomiast proces jest obiektem akywnym.***



Blok kontrolny procesu

- Struktura w której system operacyjny przechowuje informacje o procesie:
  - Stan procesu
  - Licznik rozkazów (pozycja aktualnie wykonywanej instrukcji)
  - Rejestry procesora (akumulatory,rejestry indeksowe,wskaźniki stosu)
  - Informacje o planowaniu przydziału procesora (np. priorytet procesu)
  - Informacje o zarządzaniu pamięcią (rejestry graniczne, tablice stron, lub tablice segmentów)
  - Informacje do rozliczeń (ilość zużytego procesora i czasu rzeczywistego, ograniczenia czasowe,numery kont,numery zadań,numery procesów)
  - Informacje o stanie wejścia-wyjścia

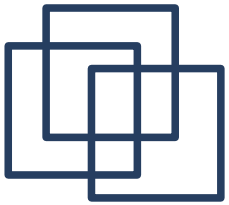
Wskaźnik	Stan procesu
Numer procesu (PID)	
Licznik rozkazów (PC)	
Rejestry	
Ograniczenia pamięci	
Wykaz otwartych plików	
… … …	



Wielozadaniowość?

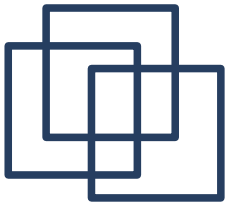
- Systemy jednego użytkownika (single-user: S), wieloużytkownikowe (multi-user: M).
- Systemy jednozadaniowe (single-task: S), wielozadaniowe (multi-task: M).
- QM - quasi-multi

System Operacyjny	Użytkownicy	Zadania	Procesory
MS/PC DOS	S	S	1
Windows 3x	S	QM	1
Macintosh Syst. 7.*	S	QM	1
Windows 95/98/ME	S	M*	1
AmigaDOS	S	M	1
Unix-like	M	M	n
VMS	M	M	n
NT-like	S/M	M	n
Windows 2000/XP	M	M	n
OS390 (zOS)	M	M	n



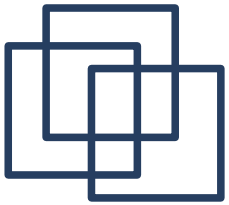
Powłoki i Interpretery

- Wszystkie współczesne SO posiadają interfejs graficzny, który umożliwia wykonywanie większości zadań.
- Właśnie: Większość zadań, czyli tylko te wcześniej przewidziane przez programistów. Interfejsy graficzne bardzo trudno, o ile to w ogóle możliwe rozszerzać.
- Każdy współczesny system posiada również powłokę udostępniającą wiekszą wolność dla administratorów.
- W powłokach możemy 'wygodnie' automatyzować zadania za pomocą skryptów lub plików wsadowych.
- Wiele badań nad użytecznością wskazuje, że praca w trybie tekstowym JEST WYDAJNIEJSZA!



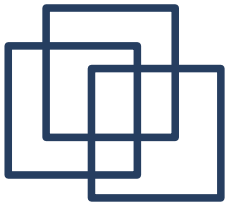
Wybrane podstawowe koncepcje SO

- Log i Audyt (obserwowalność i sterowalność systemu)
- Bezpieczeństwo (zewnętrzne i wewnętrzne)
- Uprawnienia użytkowników (poziomy)
- System plików (architektura i wydajność)
- Zdalna administracja (dostęp do maszyny przez sieć)
- Struktura katalogów (dostęp do plików)
- Typy plików (rozpoznawanie typów plików)



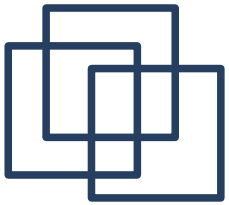
Zestawienie Koncepcji Unix i Windows

OS Unixowy	Windows
Wiele menadżerów okien	Jedno Windows GUI
GUI Administracyjne (nie-standard)	GUI Administracyjne (Standard)
Dowolny model klient-server	Model z serwerem centralnym
rsh (remote shell)	ograniczona implementacja w serwerze
Mnogość darowego oprogram.	Ograniczone darmowe programy
Perl	Perl + moduł WIN32
Skrypty	Skrypty
Powłoki (bash,zsh,sh,csh,...)	Okno komend DOS
Primitywny model bezpiecz.	Primitywny model bezpiecz.
Pliki z kropką z konfiguracją	Rejestr systemowy
Potoki za pomocą: comm1 | comm2	Potoki za pomocą: comm1 | comm2
Konfiguracja w plikach tekstowych	Konfiguracja w binarnej bazie



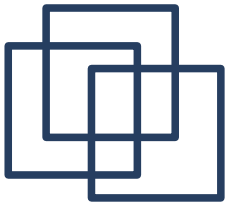
Zestawienie Koncepcji Unix i Windows

OS Unixowy	Windows
Biblioteki Standardowe	WIN32 API
Biblioteki Unixowe	Biblioteka kompatyb. z Posix
Symboliczny/Twardy	Tylko Twarde linki (short cuts)
Procesy	Procesy
Wątki	Wątki
Długie nazwy	Długie nazwy na NTFS
Montowanie dysku do katalogu	Montowanie litera A: B: itp.
endl jest jednym znakiem LF	endl jest dwoma CR LF
UID (User ID)	SID (Subject ID)
Grupy (groups)	Grupy (groups)
ACLs (nie standardowe)	ACLs
Permission bits	(Tylko w ACLs lub w Cygwinie)
Biblioteki dzielone	DLL'e
Zmienne środowieskowe	Zmienne środowieskowe
Daemons/services/init	Menadżer usług
DNS/DHCP/bootp (w standardzie)	DNS/DHCP (NT server)
X Window	X Window



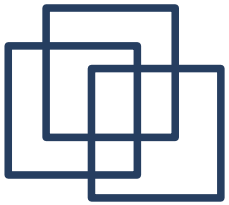
Log i Audyt

- Audyty oraz logowanie sięga historią mainframów, kiedy użytkowniki płacili za czas użytkowania systemu.
- Potrzeba zatem trzymać listę transakcji (log transakcji), aby w przyszłości można było sprawdzić co się działo w ścisle określonym okresie. (dmesg, /var/log/*)
- W dziesiejszych czas logowanie ponownie ma wielkie znaczenie w aspektach bezpieczeństwa systemu komputerowego.
- Jądro systemu operacyjnego zarządza zasobami oraz udostępnia usługi, musimy mieć możliwość diagnozowania stanu jądra.



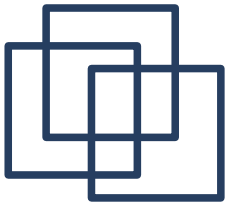
Bezpieczeństwo!

- Rozwój komputerów osobistych wyprzedził o kilka lat rozwój sieci. Pierwsze systemy operacyjne zupełnie zignorowały problem bezpieczeństwa: DOS, Windows (3.x, 95/98/Milenium), Amiga-OS, MacIntosh.
  - *No limits*
- Unix oraz Windows NT były projektowane z uwzględnieniem wielu użytkowników oraz sieci komputerowej.
- Bezpieczeństwo bezwględne jest nieosiągalnym ideałem i w rzeczywistości jest rozważane w kategoriach poziomu ryzyka.



Systemy Plików

- Pliki i system plików to integralna część systemu operacyjnego! Praktycznie wszystkie czynności administracyjne wymagają modyfikacji w plikach.
- Techniczne aspekty systemów plików, czyli fizyczna reprezentacja danych na dyskach jest bardzo różnorodna.
  - Unixie: drzewo i-node'ów,
  - Windows NT: podobnie jak w Unix, drzewo
  - DOS, Windows 3.x: struktura tablicowa (tablica FAT – file allocation table)
- Systemy transakcyjne i nie transakcyjne:
  - Nie transakcyjne: Linux: ext2, Windows: FAT32
  - Transakcyjne: Linux: ext3, ReiserFS, Windows: NTFS

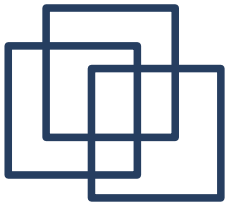


Porównując Unix z Windows

- Pomimo różnych struktur plików i katalogów oba systemy posiadają takie same podstawowe narzędzia (chociaż inaczej się nazywają :-))

Zestawienie komend z Unixa i Windows

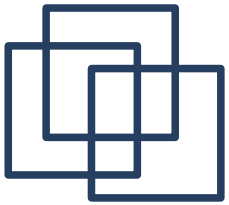
SO Unixowy	Windows
chmod	CACLS
chown	CACLS
chgrp	No direct equivalent
emacs	Wordpad or emacs in GNU tools
kill	kill command in Resource Kit
ifconfig	ipconfig
lpq	lpq
lpr	lpr
mkfs/newfs	format and label
mount	net use
netstat	netstat
nslookup	nslookup
ps	pstat in Resource Kit
route	route
setenv	set
su	su in resource kit
tar	tar command in Cygnus tools
traceroute	tracert



Porównanie strukturę katalogów

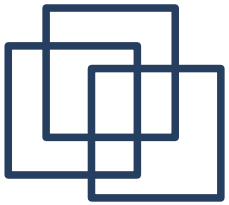
- Znak rozdzielajacy katalogi: \ oraz /
- W unixie jest jeden korzeń całego systemu. W Windows każdy dysk ma własny korzeń.

OS Unixowy	Windows
/usr	%SystemRoot% zazwyczaj wskazuje na C:\WinNT lub C:\Windows
/bin or /usr/bin	%SystemRoot%\System32
/dev	%SystemRoot%\System32\Drivers
/etc	%SystemRoot%\System32\Config
/etc/fstab	Brak odpowiednika.
/etc/group	%SystemRoot%\System32\Config\SAM* (binarny)
/etc/passwd	%SystemRoot\%\System32\Config\SAM* (binarny)
/etc/resolv.conf	%SystemRoot%\System32\DNS*
/tmp	C:\Temp
/var/spool	%SystemRoot%\System32\Spool



Struktura Katalogów - Unix

- **/bin** - podstawowe programy narzędziowe, wykorzystywane zarówno podczas uruchamiania systemu w trybie single user jak i multi user. Wykorzystywane są one do tworzenia otoczenia systemowego.
- **/etc** - większość plików konfiguracyjnych, pliki z konfiguracją systemu używaną do procesu bootowania, tzn rc.conf, rc.local.
- **/usr** - większość aplikacji i programów użytkowych dla użytkowników. Programy tutaj są mniej krytyczne dla działania systemu.
  - /usr/bin
  - /usr/sbin
  - /usr/local
- **/sbin** – większość programów narzędziowych przeznaczonych dla administratorów systemu



Struktura Katalogów - Unix

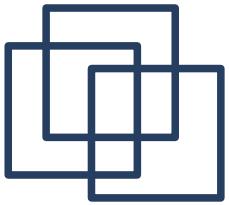
- **/sys** lub **/usr/src** – źródła jądra systemu
- **/dev -** bloki i pliki urządzeń wykorzystywane przez system.
  - W Unixie, każde urządzenie reprezentowane jest jako specjalny plik w tym katalogu. Komunikacja z urządzeniami realizowana jest za pomocą operacji zapisu i odczytu z pliku.
- **/home** – zwyczajowa lokalizacja katalogów użytkowników systemu
- **/root** – katalog domowy administratora systemu (pamiętajmy, że root w unixie może 'wszystko')
- **/var** – katalog gdzie przechowywane są logi o pracy serwera, bufory różnych kolejek itp.
  - /var/mail – skrzynki pocztowe użytkowników
  - /var/log – logi różnych serwisów uruchomionych na serwerze

Struktura Katalogów - Unix

- Każdy katalog w unixie posiada specjalne katalogi wirtualne oznaczone kropką '.' i podwójną kropką '..'. Jedna kropka oznacza katalog bieżący, dwie kropki katalog piętro wyżej w hierarchii.

Struktura Katalogów - Unix

- Nawigacja po katalogach w Unixie.
- Jeden korzeń / dla całego i wszystkich systemów plików.
- Ścieżka względna: nie rozpoczynająca się od znaku /
- Ścieżka bezwzględna: rozpoczynająca się od znaku /



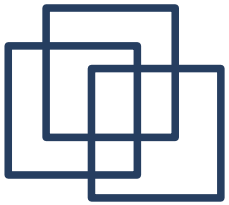
Acykliczny Graf Katalogów

- WINDOWS - skrót do programu lub katalogu, który jest interpretowany jedynie przez program Explorer.
- UNIX - linki są zaimplementowane na poziomie systemu operacyjnego. W unixie występują dwa typy linków:
  - linki symboliczne - przechowują jedynie ścieżkę oraz nazwę do pliku oryginalnego. Przykład użycia:

    ```
    ln -s ../jaki/plik.txt nowy_link.txt
    ```

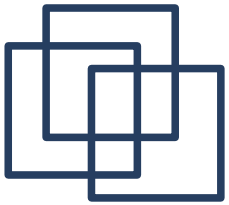
  - linki twarde - kopia wpisu struktury informacyjnej pliku do nowego miejsca (nowa struktura nadal odowolueje się do tego samego miejsca na dysku fizycznym) Przykład użycia:

    ```
    ln ../jaki/plik nowy_link_twardy.txt
    ```



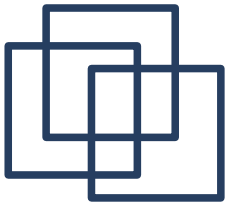
Co nam pokazuje ls?

```
volt% ls -la
total 1
drwxr-x---   3 szmurlor  prac    512 11 paź 21:42 .
drwx------  12 szmurlor  prac   1536 11 paź 21:20 ..
drwxr-x---   2 szmurlor  prac    512 11 paź 21:40 katalog
-rw-r-----   1 www        prac      0 11 paź 21:41 plik_pusty
-rw-r-----   1 szmurlor  prac    296 11 paź 21:41 plik_z_zawartoscia
-rw-r-----   1 szmurlor  prac    353 11 paź 21:41 program
-r-xr-x---   1 szmurlor  prac  15324 11 paź 21:42 program1
-rwxr-x---   1 szmurlor  prac    411 11 paź 21:41 program2
volt%
```



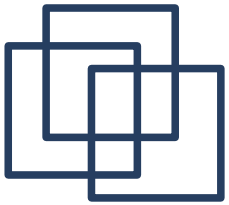
Zdalna Administracja (przez sieć)

- W Windows nie ma zdalnej konsoli tekstowej (w wersji podstawowej).
- W obu systemach jest zdalna praca w trybie graficznym.
  - Unix: X Window
  - Windows: Terminal Server
  - Unix i Windows: VNC



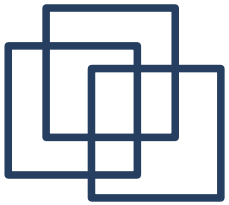
Rozpoznawanie typów plików Windows

- na podstawie rozszerzenia nazwy pliku:
  .exe, .bat, .btm, .sys, .com, .txt, etc.
- struktura danych wewnątrz pliku:
  - .com - plik zawiera tylko kod programu w postaci skompilowanej. Pierwszy bajt pliku to konkretna instrukcja maszynowa.
  - .exe - plik jest w specjalnym formacie. Znakiem rozpoznawczym pliku exe dodatkowo są pierwsze dwa bajty które są zawsze równe: MZ. Pliki exe zawierają oprócz kodu nagłówek, informacje o stosie, stercie oraz pamięci operacyjnej która jest wymagana przez proces.
  - .bat – plik zawiera skrypt, stanowiący zbiór komend



Rozpoznawanie typów plików Unix

- Rozszerzenie ma znaczenie tylko umowne. System nie rozpoznaje typu pliku po rozszerzeniu.
- atrybuty plików:
  - + r - prawo do czytania
  - + w - prawo do zapisywania
  - + x - plik wykonywalny
- pierwsza linia w plikach tekstowych:
  - #!/bin/zsh - oznacza że plik jest skryptem napisanym w języku zsh.
  - pliki wykonywalne zapisywane są w specjalnych formatach. Np w linuxie obowiązuje standard ELF. Wstandardzie tym piewsze cztery bajty są zawsze takie same: [0x7f],E,L,F

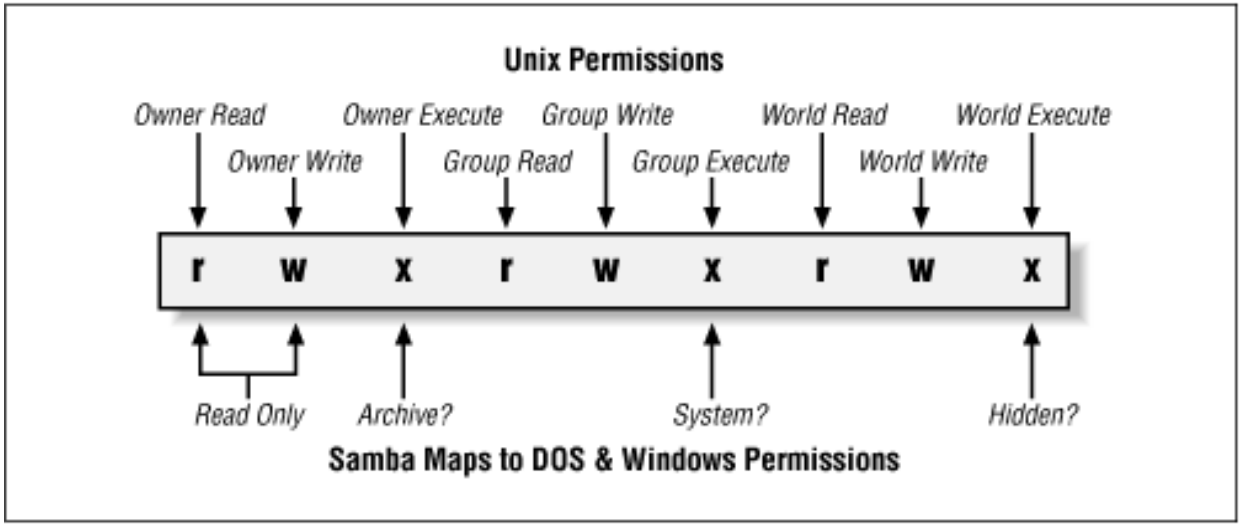


Poziomy Uprawnień Użytkowników

- Konto uprzywilejowane:
  - Unix: root (ma automatycznie dostęp do absolutnie wszystkiego)
  - Windows: Administrator (musi mieć udostępnione prawo do danego obiektu, całe szczęście Administrator może udostępniać prawa do absolutnie wszystkich obiektów)
- Konta uprzywilejowane NIGDY nie powinny być używane podczas normalnej pracy użytkownika.
- Otrzymanie uprzywilejowanego konta kojarzy się często z otrzymanie władzy nad innymi użytkownikami, tymczasem zostało ono stworzone ponieważ użytkownicy nie chcieli posiadać zbyt szerokich uprawnień ze względu na odpowiedzialność.

Uprawnienia w Unix

- W Unix każdy proces (uruchomiony program) posiada trzy główne identyfikatory:
  - PID (Process ID) – liczba integer określająca identyfikator procesu,
  - UID (User ID) – liczba integer określająca właściciela procesu,
  - GID (Group ID) – liczba integer określająca grupę procesu.
- Dostęp do zasobów jest określany na podstawie UID i GID.
- Zasoby w Unix posiadają trzy poziomy praw dostępu:
  - dla właściela zasobu (user lub owner),
  - dla grupy będącej właścicielem (group),
  - dla reszty (others).



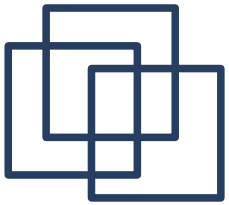
Zarządzanie Uprawnieniami w Unix

- Zmiana właściciela zasobu:

```
volt% ls -l plik_pusty
-rw-r-----  1 szmurlor  prac  0 11 paź 21:41 plik_pusty
volt% sudo chown www plik_pusty
volt% ls -l plik_pusty
-rw-r-----  1 www  prac  0 11 paź 21:41 plik_pusty
volt% sudo chown :www plik_pusty
volt% ls -l plik_pusty
-rw-r-----  1 www  www  0 11 paź 21:41 plik_pusty
volt%
```

- Zmiana atrybutów:

```
volt% chmod a+rx plik_pusty
volt% ls -l plik_pusty
-rwxr-xr-x  1 szmurlor  www  0 11 paź 21:41 plik_pusty
volt% chmod o-rx plik_pusty
volt% ls -l plik_pusty
-rwxr-x---  1 szmurlor  www  0 11 paź 21:41 plik_pusty
volt% chmod g-x plik_pusty
volt% ls -l plik_pusty
-rwxr-----  1 szmurlor  www  0 11 paź 21:41 plik_pusty
volt% chmod 754 plik_pusty
volt% ls -l plik_pusty
-rwxr-xr--  1 szmurlor  www  0 11 paź 21:41 plik_pusty
volt%
```

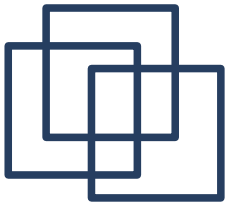


Uprawnienia w Unix szczegóły

- Komendy:
  - **`chmod [-R] [uprawnienia] [nazwa_pliku]`**
    gdzie [uprawnienia] mogą być postaci:
    - wyrażenia: a+, u+, g+, o+ oraz a-, u-, g-, o- połączonymi z uprawnieniami rwx,
    - trzema liczbami reprezentującymi binarną postać uprawnień:
      - 777, pierwsza liczba to właściciel, druga grupa, trzecia reszta.
      - Binarnie: 111 jest równe dziesiętnie 4 + 2 + 1 = 7
      - Przykłady:
        - r-x = 101 = 5,
        - rw- = 110 = 6,
        - --x = 001 = 1
        - rw-xr--r-x = 110 100 101 = 645

Rekursywnie przejdź po katalogach

001	1	--x
010	2	-w-
100	4	r--
110	6	rw-
101	5	r-x
-	644	rw-r--r--

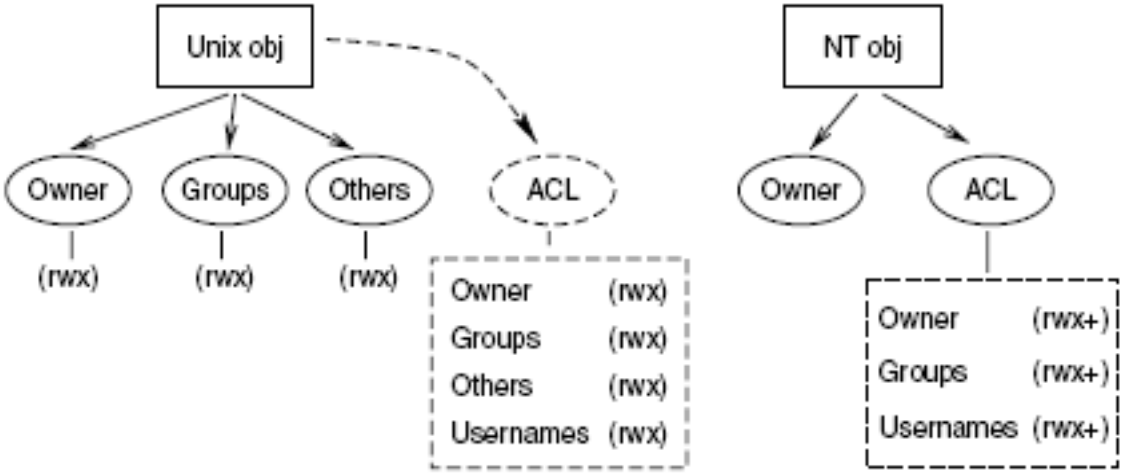


Uprawnienia w Unix szczegóły

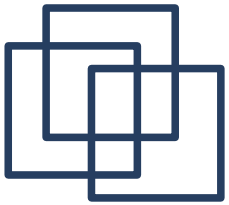
- Przykładowe czynności:
  - pozwól zapisyć wszystkim: **`chmod a+w mojplik`**
  - Pozwól właścicielowi uruchamiać: **`chmod u+x mojplik`**
  - Pozwól wszystkim czytać i urucham.: **`chmod 755 mojplik`**
  - Ustaw s-bit dla grupy: **`chmod g+s mojplik`**
  - Przejdź rekursywnie po podkatalogach: **`chmod -R a+r .`**
- umask – odwrotność chown – zawsze usuwa zaznaczone bity. (Uwaga, nie zmienia bitów nie zaznaczonych).
  - **`umask 077`** -> powoduje, że plik może mieć rwx --- ---
  - **`umask 022`** -> odbiera grupie i reszcie prawo pisania czyli odbiera: --- -w- -w-

Access Control Lists

- Każdy użytkownik i grupa mogą mieć indywidualne uprawnienia do danych plików.



- W Unix ACL funkcjonują równolegle do standardowych uprawnień i ze względu na skomplikowaną obsługę i kontrolę nie zyskały popularności.



Interakcja

- Jeżeli coś cię zainteresowało i chciałbyś aby na następnym wykładzie zostało rozszerzone, powtórzone, omówione dokładniej, to nie krępuj się i napisz maila:

  szmurlor@iem.pw.edu.pl

- Jeżeli coś było nie jasne, napisz maila:

  szmurlor@iem.pw.edu.pl

- Jeżeli coś cię znudziło, napisz maila:

  szmurlor@iem.pw.edu.pl