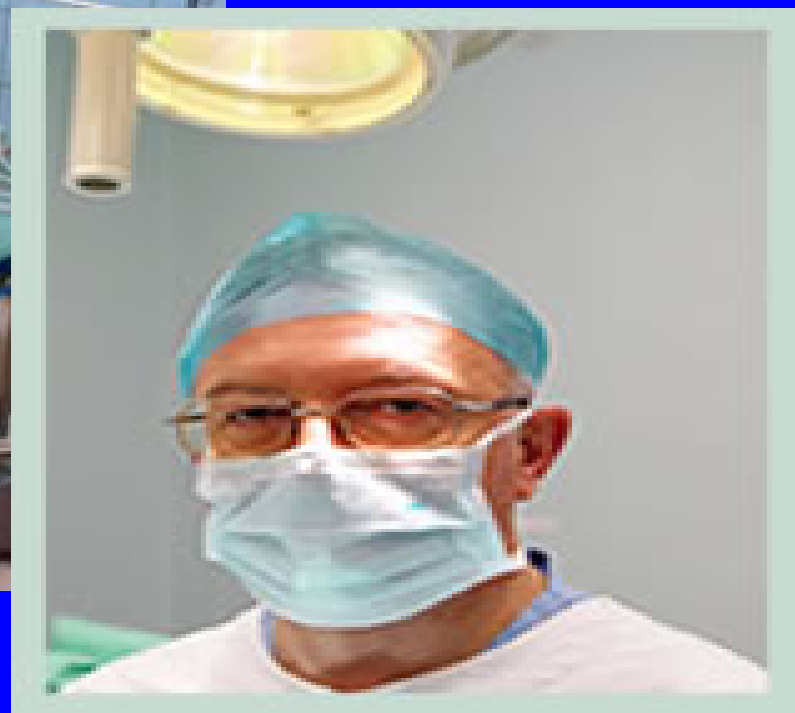


Systemy operacyjne

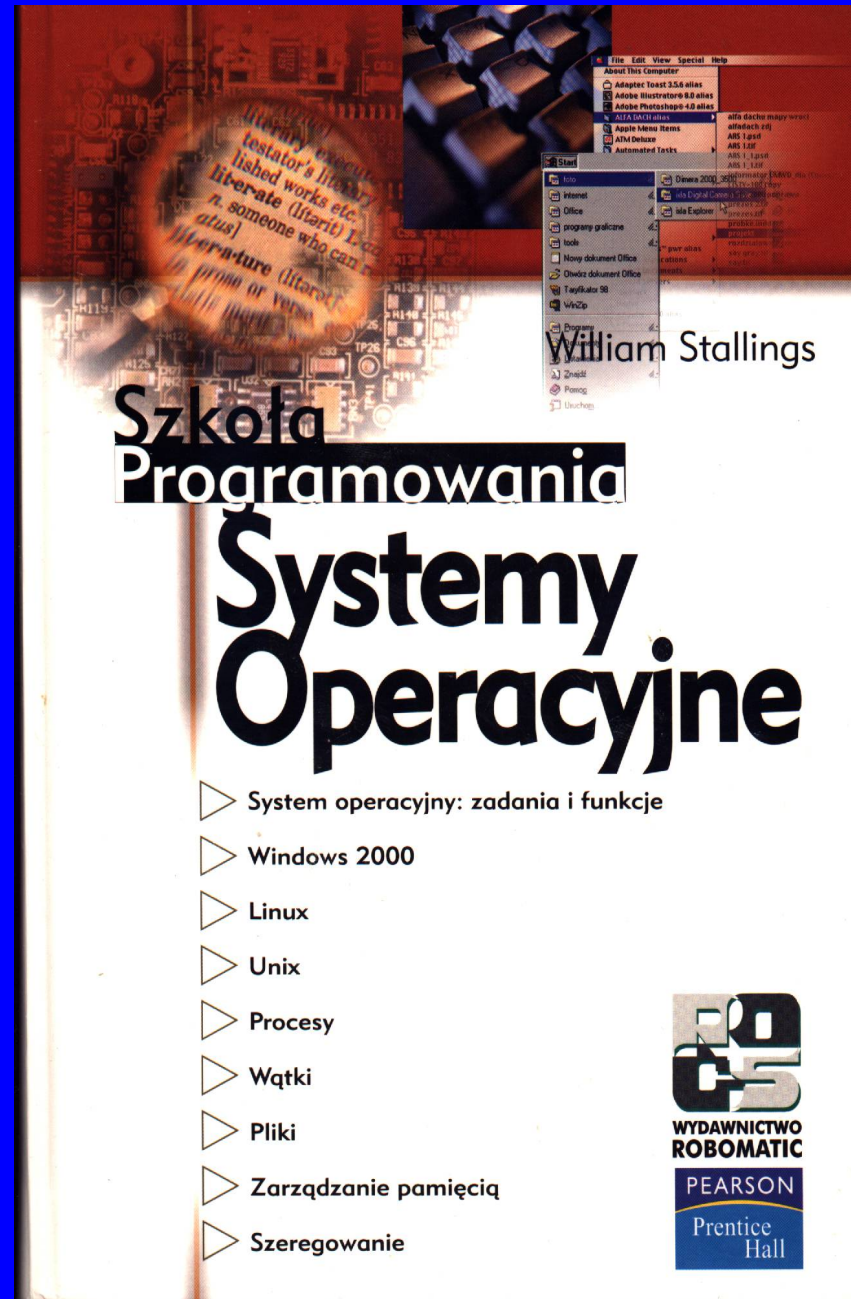


Opis przedmiotu

- Wykład (30 godz.)
 - program wykładu
- Laboratorium (30 godz.)
 - LINUX
 - serwer / stacja robocza?
- Egzamin na koniec semestru (test)
 - materiał z wykładu
 - ok. 20 pytań, na które należy udzielić odpowiedzi (punktowane 0, 1 lub 2 punkty)

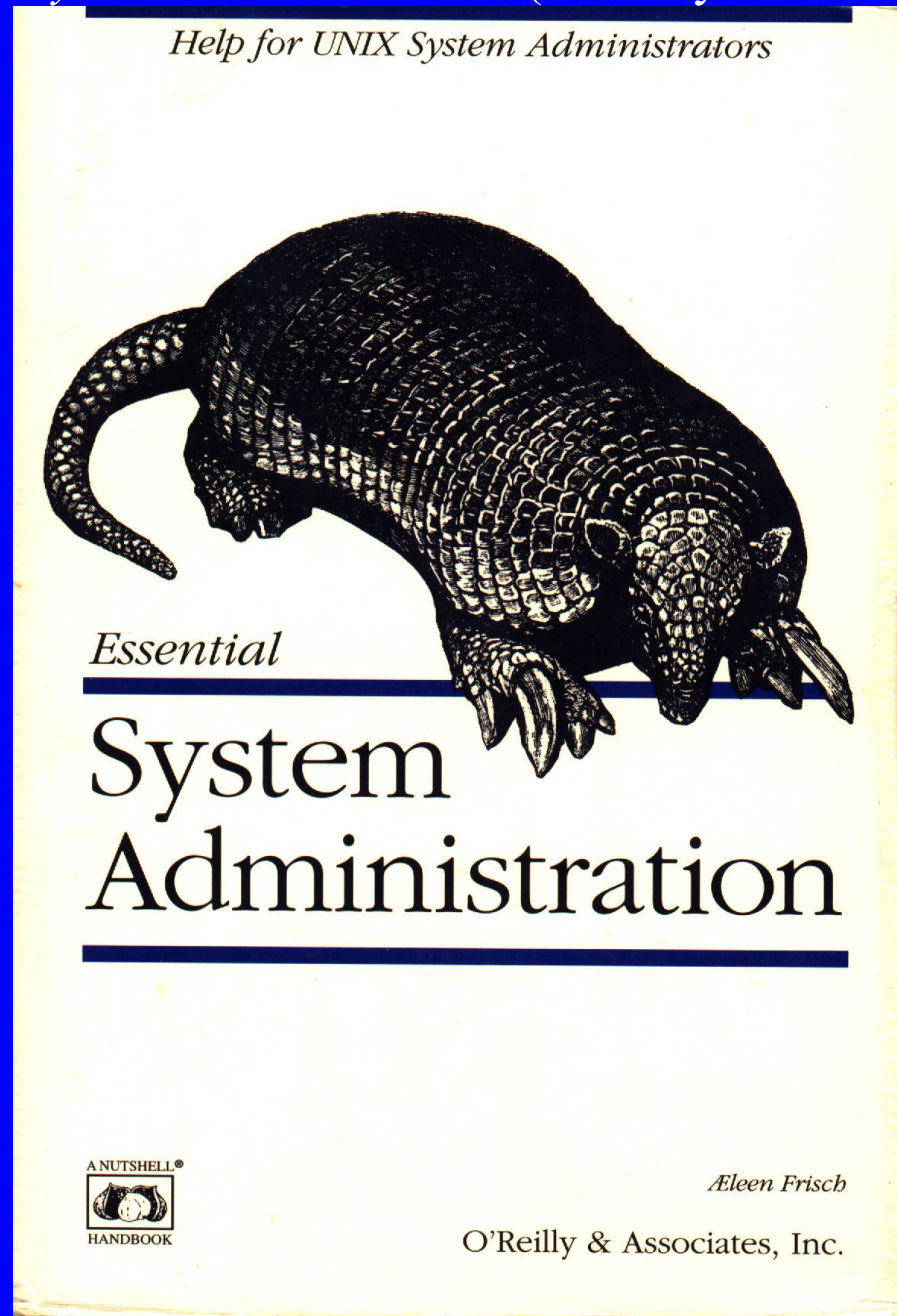
Literatura dodatkowa:

William Stallings - *Systemy operacyjne* (seria *Szkoła programowania*, ok. 700 str.)

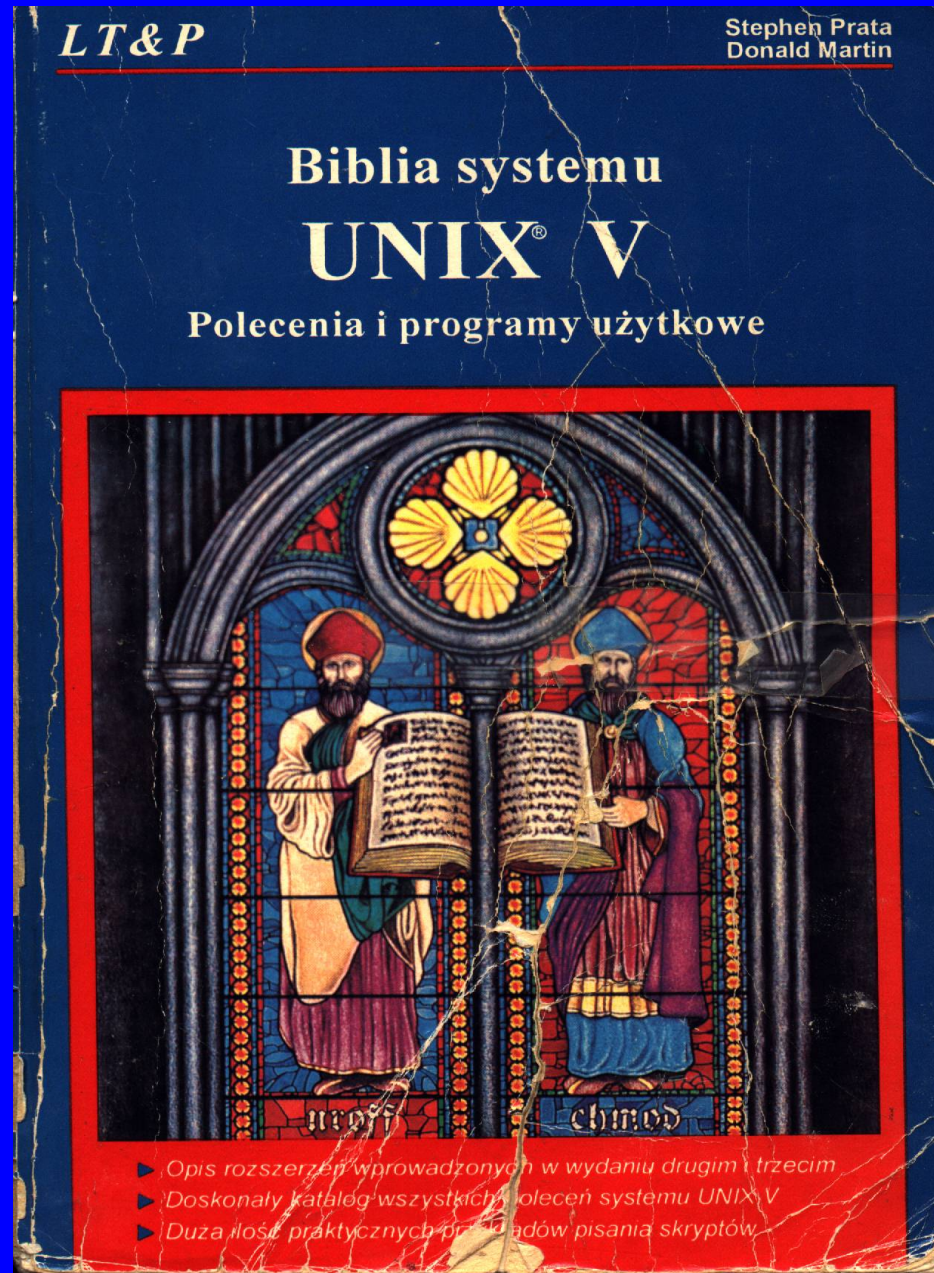


Literatura dodatkowa:

Aleen Frisch – *Essential System Administration* (O'Reilly & Associates, ok. 450 str.)

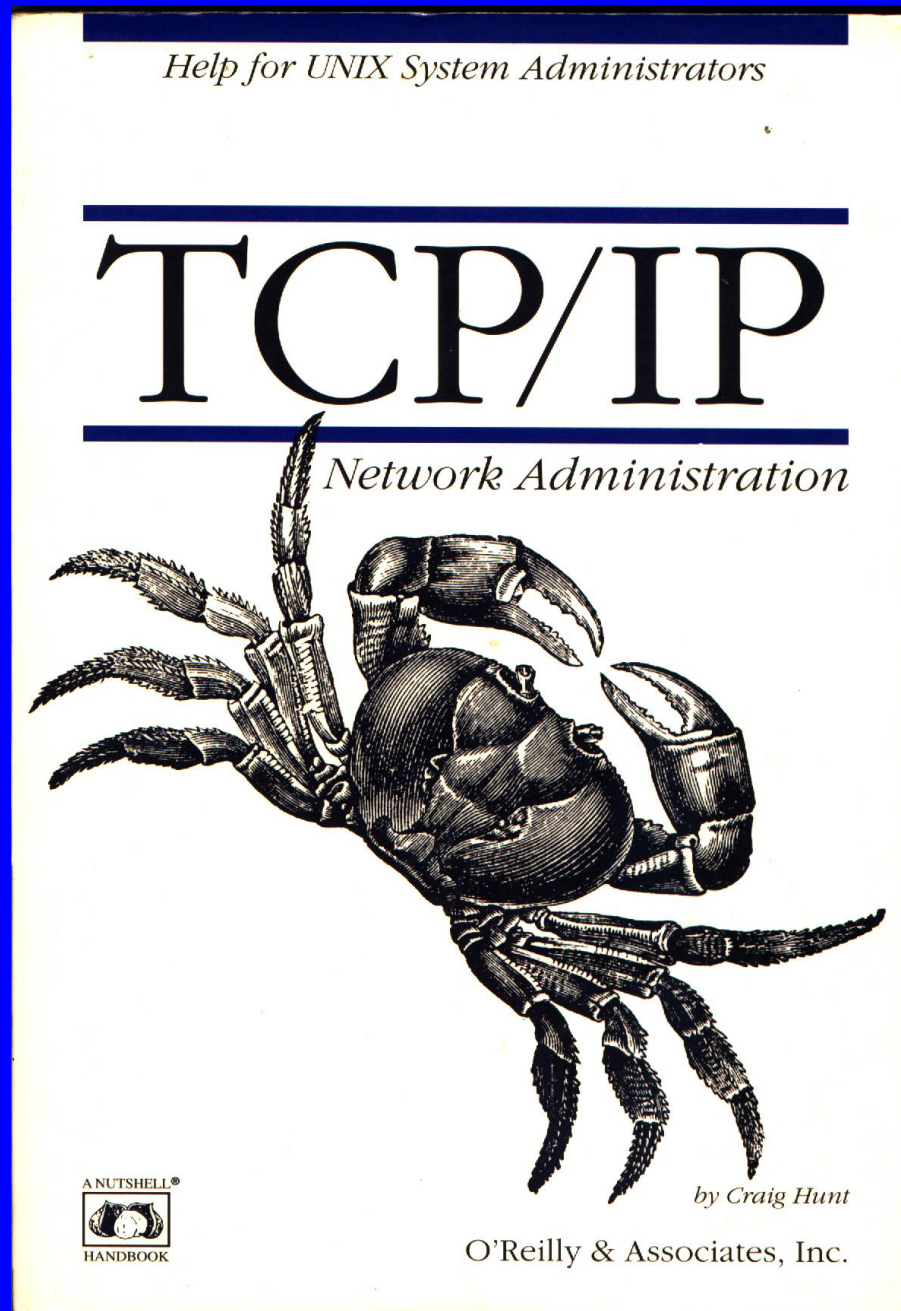


Literatura dodatkowa:
Stephen Prata, Donald Martin – *Biblia systemu UNIX V. Polecenia i programy użytkowe*
(rok wyd. 1994, ok. 450 str.)



Literatura dodatkowa:

Craig Hunt – *TCP/IP Network Administration* (O'Reilly & Associates, ok. 470 str.)



Literatura dodatkowa:

Z ostatniej chwili – powstał „ważny” serwis internetowy:

<http://wazniak.mimuw.edu.pl>

(...)

Przekazujemy do rąk wykładowców i studentów informatyki materiały dydaktyczne przygotowane w ramach projektu Opracowanie programów nauczania na odległość na kierunku studiów wyższych – Informatyka. Główne cele projektu to:

Opracowanie programów nauczania dla studiów informatycznych I i II stopnia.

Przygotowanie elektronicznych materiałów dydaktycznych, w tym pakietów multimedialnych do prowadzenia zajęć zdalnych.

Ministerstwo Edukacji Narodowej zakwalifikowało projekt do realizacji w maju 2006 roku. Wykonanie projektu sfinansowano ze środków Europejskiego Funduszu Społecznego z programu Sektorowy Program Operacyjny Rozwój Zasobów Ludzkich 2004 - 2006.

Tylko uczelnie o różnorodnych i uzupełniających się profilach oraz dużych doświadczeniach dydaktycznych były w stanie zrealizować tak złożony i trudny projekt. Dlatego do realizacji projektu powołano partnerstwo czterech, czołowych polskich uczelni - Uniwersytetu Warszawskiego, Uniwersytetu Jagiellońskiego, Politechniki Poznańskiej i Politechniki Warszawskiej.

(...)

Program wykładu

- Ogólna charakterystyka systemów Unixowych.
 - Rys historyczny.
 - Identyfikacja – użytkownicy, grupy.
 - Struktura katalogów - polecenia: *cd*, *ls*, *mkdir*, *rm*, *rmdir*, *cp*, *mv*, *pwd*.
 - System plików – prawa dostępu, polecenia: *chmod*, *chown*, *chgrp*. Bity SUID i SGID.
 - Dowiązania statyczne i symboliczne.

Program wykładu - c.d.

- Wykorzystanie mechanizmu *pipeline* oraz redirectioni do budowania potoków poleceń.
 - Użycie poleceń:
echo, cat, more, less, tee, head, tail, wc, sort, uniq, rev.
- Archiwizacja i kompresja danych przy pomocy standardowych narzędzi systemu Unix.
 - Standardowe programy archiwizujące: *tar, cpio.*
 - Przegląd standardowych programów kompresujących: *gzip, compress, pack, zip.*
 - Kodowanie plików binarnych do postaci tekstowej przy pomocy programów *uuencode* i *uudecode.*

Program wykładu - c.d.

- Nazwy wieloznaczne oraz wyrażenia regularne.
 - Opis znaków specjalnych powłoki.
 - Definicja pełnych i ograniczonych wyrażen regularnych.
 - Użycie programów: *grep*, *fgrep*, *egrep*.
- Przegląd podstawowych narzędzi CLI (command line interface).
 - Opis możliwości i zastosowań programów: *find*, *tr*, *sed*, *awk*, *diff*, *patch*.

Program wykładu - c.d.

- Skrypty powłoki systemowej.
 - Przegląd konstrukcji interpretera poleceń *Bash*.
 - Użycie zmiennych środowiskowych i cytowania poleceń.
- Zarządzanie procesami.
 - Opis pojęcia procesu w systemie Unix.
 - Użycie polecenia *ps*.
 - Wysyłanie sygnałów do działających procesów – polecenia *kill* i *nice*.
 - Praca w tle – polecenia *jobs*, *fg*, *nohup*.

Program wykładu - c.d.

- Proces startu systemu Unix.
 - Etapy startu systemu Unix.
 - Ładowanie jądra.
 - Standardowe skrypty startowe różnych odmian Unixa.
 - Poziomy pracy systemu.
- Partycje w systemie plików Unixa.
 - Typy partycji.
 - Tworzenie i formatowanie partycji.
 - Montowanie partycji w systemie plików. System plików NFS.
 - Zestaw narzędzi *ltools* do bezpośredniego dostępu do napędów dyskieta w systemie Linux.

Program wykładu - c.d.

- Standardowe demony.
 - Opis działania standardowych demonów: *inetd*, *atd*, *crond*.
- Podstawowe protokoły i usługi sieciowe.
 - Wprowadzenie do zagadnień sieci komputerowych.
 - Identyfikacja i przepływ informacji w sieci Internet.
 - Użycie klientów protokołów: *telnet*, *ftp*, *http*.

Program wykładu - c.d.

- Podstawowe własności i działanie systemu X-Window.
 - Przegląd podstawowych własności graficznego interfejsu X-Window.
- Rutynowe prace administratora systemu Unix.
 - Instalacja i aktualizacja systemu.
 - Administracja kontami użytkowników oraz zasobami dyskowymi.

Czym jest system operacyjny?

- “An elephant is a mouse with an Operating System.” - D. Knuth
- oprogramowanie nadzorujące pracę systemu komputerowego, które:
 - przyjmuje zlecenia przekazywane przez programy użytkowe (lub samych użytkowników)
 - wykonuje je (przydzielając im zasoby komputera: pamięć, czas procesora, urządzenia zewnętrzne, itp.)
 - działa zawsze (program startuje jako pierwszy i kończy pracę jako ostatni)

Fragment historii systemu UNIX

- 1960s – AT&T Bell Labs. + MIT = MULTICS
- 1969 – Ken Thompson, Dennis Ritchie, Brian Kernighan – UNIX jako przeciwieństwo (DEC PDP-7, single user)
- 1973 – D. Ritchie (język programowania C)
- AT&T UNIX kontra BSD (standard POSIX)
- 1991 – Fin Linus Torvalds – LINUX
 - FSF, GNU, GPL

Fragment historii systemów MicroSoft Windows

- 1981 – MSDOS 1.0, IBM 8086
- 1983 – DOS 2.0, PC XT – dysk twardy
- 1984 – DOS 3.0, PC AT 80286 – adresowanie pamięci nie wykorzystane przez DOS
- DOS 3.1 – obsługa sieci
- 1987 – PS/2 80386, DOS 3.3
- 1990 – WINDOWS 3.0 – dorównywał Macintosh
- 1993 – NT 3.1 – IBM OS/2 – podział
- NT 4 / WINDOWS 95 – ten sam interface graficzny
- 98, Millenium, 2000, XP, 2003, Vista,...

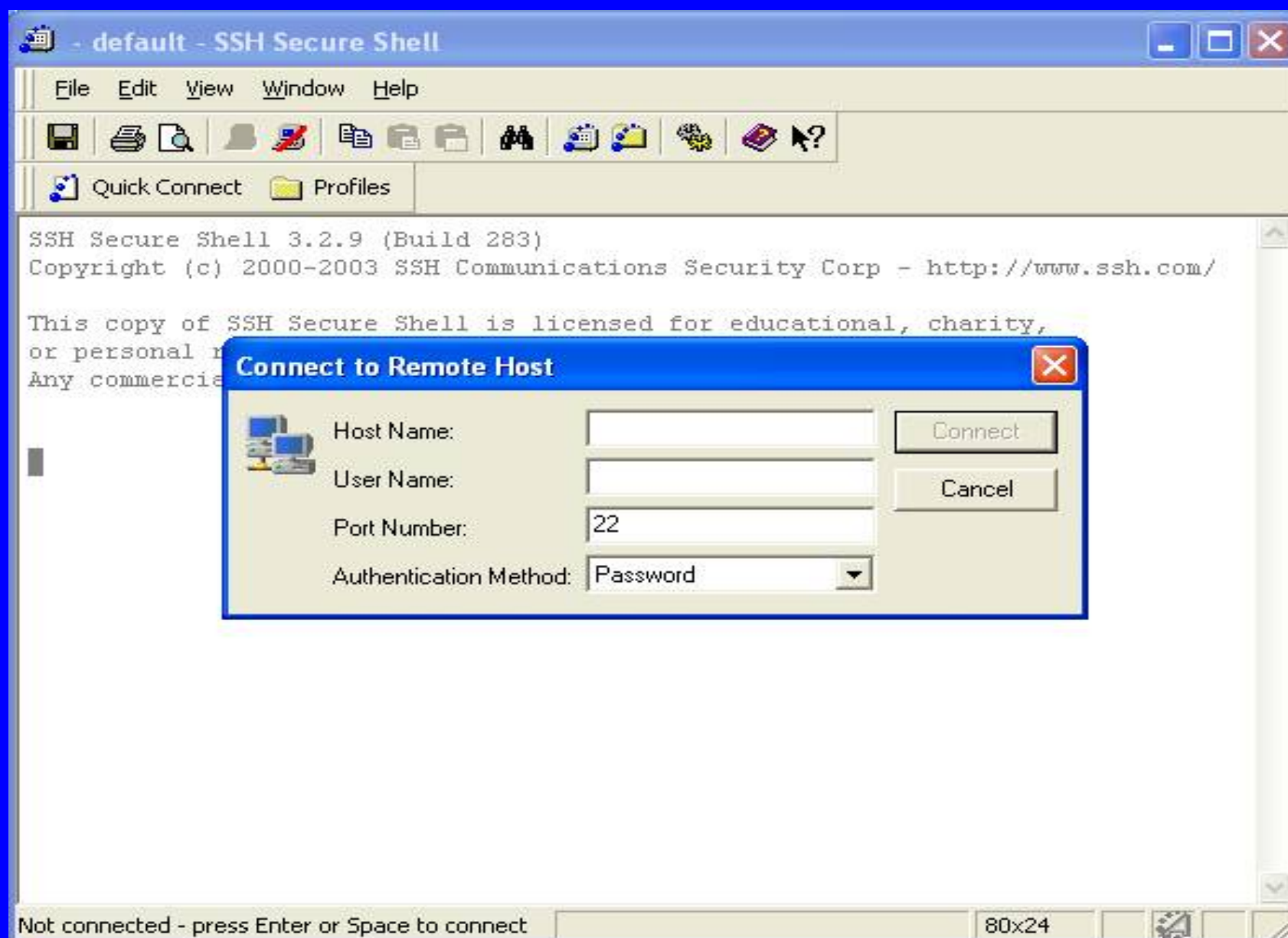
Charakterystyka systemu UNIX

- wieloplatformowość
 - PC <-> Cray
- wielozadaniowość
- wielodostępność
 - wielu użytkowników
 - praca lokalna i zdalna (sieć)

Elementy filozofii systemu UNIX

- Reguła KISS
 - “Keep it small, stupid!”
 - “Keep it small and simple”
 - BUZI:
 - “Bez Udziwnień Zapisu, Idioto!”
- “Wszystko jest plikiem”

Logowanie do systemu



Interface użytkownika

The screenshot shows an SSH Secure Shell window titled "212.33.73.194 - math - SSH Secure Shell". The window has a menu bar (File, Edit, View, Window, Help) and a toolbar with various icons. Below the toolbar, there are tabs for "Quick Connect" and "Profiles". The main area displays a file listing in a terminal-like font, showing permissions, user, group, size, date, time, and filename. At the bottom, there is a terminal prompt showing the user's current directory and a command to change to the public_html directory.

```

-rw-r--r-- 1 adamn staff 3902 Feb 14 08:19 why.miz
-rw-r--r-- 1 adamn staff 2981 Apr 22 2005 wiesiek.txt
-rw-r--r-- 1 adamn staff 66048 Apr 22 2005 wiesiek2004.doc
-rw-r--r-- 1 adamn staff 3294 Apr 23 2004 winda.pas
-rw-r--r-- 1 adamn staff 131584 Dec 3 2004 worsearches.doc
-rw-r--r-- 1 adamn staff 2201600 Oct 20 22:39 wsap.doc
-rw-r--r-- 1 adamn staff 14454 Feb 1 2005 wtol0.zip
-rw-r--r-- 1 adamn staff 80734 Jan 14 2005 wyjscie.jpg
-rw-r--r-- 1 adamn staff 18 Apr 21 2005 wyjscie.txt
-rw-r--r-- 1 adamn staff 3748 Sep 29 15:38 yahoo.txt
-rw-r--r-- 1 adamn staff 777 Aug 3 2004 yellow_4.miz
-rw-r--r-- 1 adamn staff 604482 Mar 31 2004 zad_1-2.zip
-rw-r--r-- 1 adamn staff 21504 Feb 15 2005 zadania.doc
-rw-r--r-- 1 adamn staff 55296 Feb 15 2005 zadania2.doc
-rw-r--r-- 1 adamn staff 21504 Feb 15 2005 zadania3.doc
-rw-r--r-- 1 adamn staff 92610 Apr 23 2004 zadania_an.zip
-rw-r--r-- 1 adamn staff 22528 Sep 9 15:49 zadost_wroclaw_cz.doc
-rwxr-xr-x 1 adamn staff 26112 Sep 9 17:14 zadost_wroclaw_czl.doc
drwxr-xr-x 2 adamn staff 512 Oct 19 08:39 zaklad
-rw-r--r-- 1 adamn staff 879104 Aug 4 2004 zaproszenie.pps
-rw-r--r-- 1 adamn staff 50959 Apr 15 2004 znicz_hypermova.jpg
adamn@math:~ >cd public_html/
adamn@math:~/public_html >

```

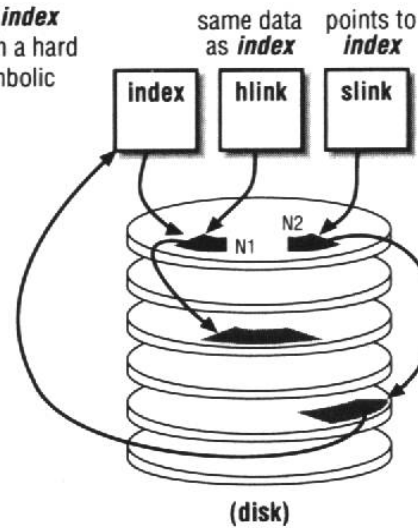
Connected to 212.33.73.194 SSH2 - aes128-cbc - hmac-md5 - none 80x23

Podstawowe polecenia UNIX / DOS

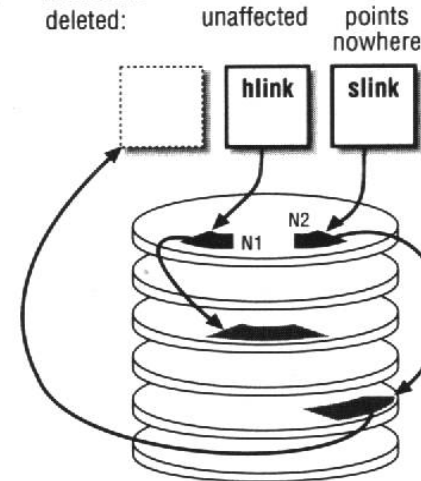
UNIX	DOS
cd	CHDIR (cd)
mkdir	MKDIR (md)
ls	DIR
rm / rmdir	DEL / ERASE / RMDIR (rd) / DELTREE
cp	COPY
mv	MOVE / RENAME
ln	-

Dowiązania statyczne i symboliczne

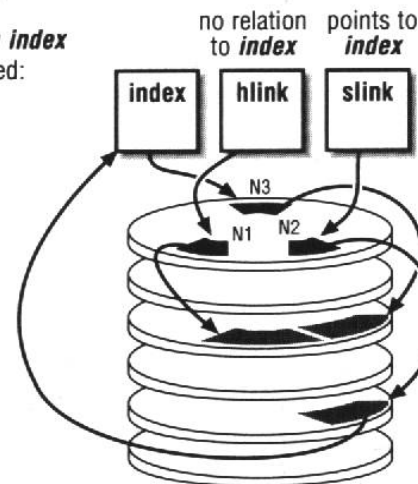
The file **index** has both a hard and symbolic link:



If **index** is deleted:



If a new **index** is created:

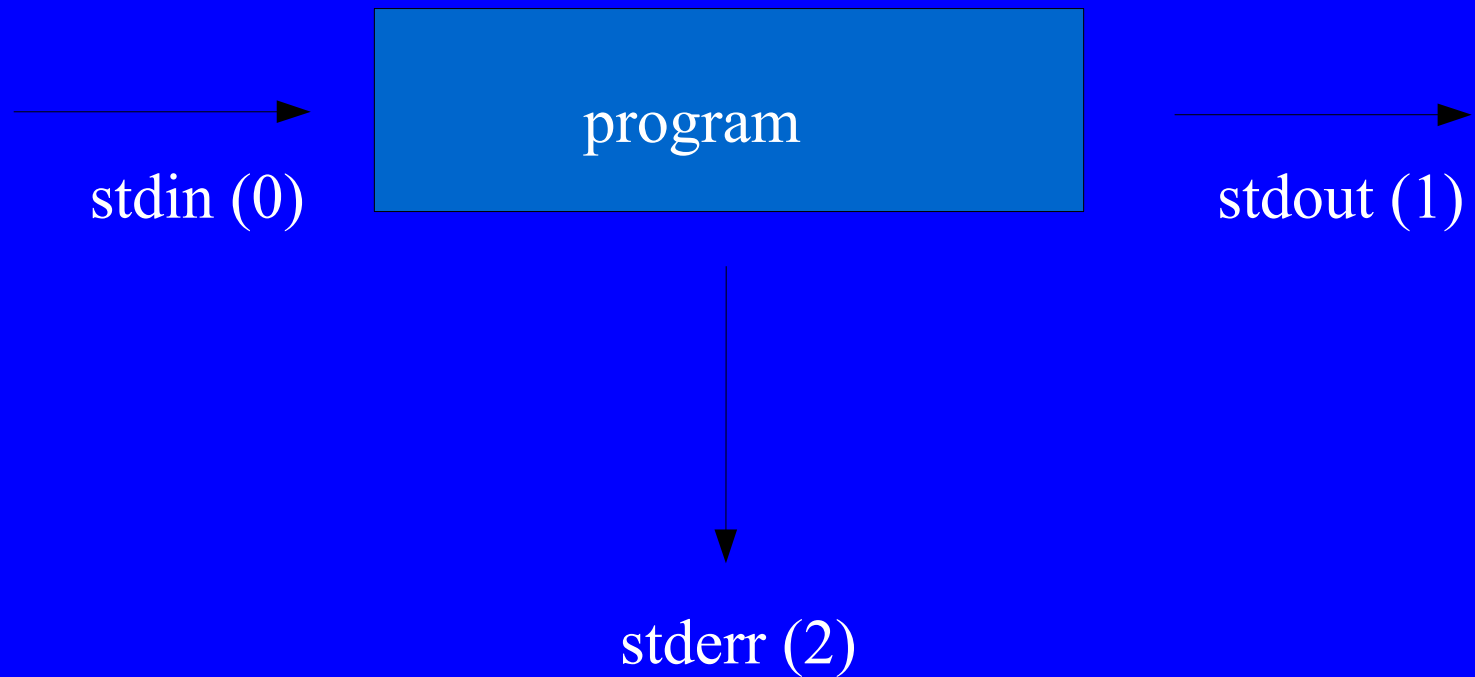


■ Inode
■ Data Block

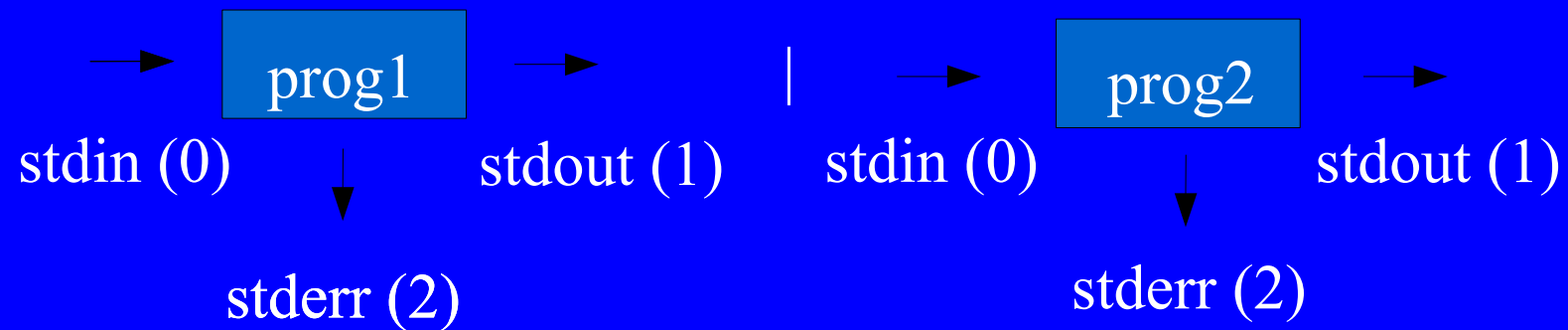
Prawa dostępu do plików

	USER	GROUP	OTHERS
	rwX	rwX	rwX
0-7	0-7	0-7	0-7
SUID=4			
SGID=2			
Sticky=1			

Standardowe deskryptory plikowe



Łączenie programów/filtrów w potok



Archiwizacja danych

- Korzyści
 - kopie bezpieczeństwa
 - kopiowanie / przenoszenie pomiędzy systemami
 - łączenie wielu plików w jedno archiwum (jeden i-node)
 - porządkowanie danych (biblioteki)
- Standardowe narzędzia
 - ar (biblioteki procedur)
 - cpio (drzewo katalogów)
 - tar (napędy taśmowe)

Kompresja danych

- Historia
 - 1951 - Huffman coding (prefix-free code)
 - MIT PhD student (Robert M. Fano) - egzamin/praca
 - *A Method for the Construction of Minimum-Redundancy Codes*
 - lepszy niż Shannon-Fano coding (bottom-up / top-down)

- Historia - c.d.
 - 1977 - Abraham Lempel, Jacob Ziv
 - LZ77
 - *A Universal Algorithm for Sequential Data Compression*
 - 1978 – LZ78 (słownik)
 - 1984 – Terry Welch
 - LZW (szybsza obsługa słownika)
 - 1985,1986 (IBM) – patenty
 - 1987 – GIF (CompuServe)
 - 1994 – orzeczenie: trzeba płacić za LZW w GIF

- Podstawowe narzędzia
 - pack (unpack, pcat)
 - Huffman
 - ~50% kompresji (opcja -f)
 - rozszerzenie .z
 - compress (uncompress, zcat)
 - LZW
 - ~50-60% kompresji
 - rozszerzenie .Z
 - gzip (gunzip, gzcat)
 - LZW w wersji PKZIP
 - ~60-70% kompresji
 - rozszerzenie .gz

Idea algorytmu Huffmana

1. Start with as many leaves as there are symbols.
2. Push all leaf nodes into the heap.
3. While there is more than one node in the heap:
4. Remove two nodes with the lowest weight from the heap.
5. If the heap was storing copies of node data rather than pointers to nodes in final storage for the tree, move these nodes to final storage.
6. Create a new internal node, with the two just-removed nodes as children (either node can be either child) and the sum of their weights as the new weight.
7. Update the parent links in the two just-removed nodes to point to the just-created parent node.
8. Push the new node into the heap.
9. The remaining node is the root node; the tree has now been generated.

Idea algorytmu LZW

(Więcej na ten temat: <http://www.dogma.net/markn/articles/lzw/lzw.htm>)

STRING = get input character

WHILE there are still input characters DO

 CHARACTER = get input character

 IF STRING+CHARACTER is in the string table then

 STRING = STRING+character

 ELSE

 output the code for STRING

 add STRING+CHARACTER to the string table

 STRING = CHARACTER

 END of IF

END of WHILE

output the code for STRING

Nazwy wieloznaczne (metaznaki)

- * - dowolny ciąg znaków (także pusty)
 - znak . na początku musi być podany bezpośrednio
 - znak . po / musi być podany bezpośrednio
 - znak / musi być podany bezpośrednio
- ? - jeden dowolny znak
- [lista] – dowolny znak z podanej listy
 - wielkość liter
 - zakresy, np. [a-z]
 - łączenie, np. [A-Zabc]
- [!lista] – dowolny znak spoza listy

Przykłady użycia metaznaków

.profile	bagfile	bgfile	canfile
ashcan	bigfile	boogfile	sleepy

- *file
- .*file
- *ashcan
- b?gfile
- *can*
- b[aou]gfile
- [ac]*
- [c-t]*
- [!ac]*

Wyrażenia regularne (regular expressions)

- podstawowe wyrażenia regularne
- ograniczone wyrażenia regularne
- pełne wyrażenia regularne

Uwaga: ograniczone $\not\subseteq$ pełne

Podstawowe wyrażenia regularne

- c – znak niebędący znakiem specjalnym
 - znaki specjalne: \, [, . oraz w pewnych kontekstach], *, ^, \$
- . - jeden dowolny znak
- \ - pozbawia znak specjalny jego znaczenia
- [lista] – jeden ze znaków z listy
 - [aeiou], [a-z], [AEIOUa-m]
 - l[ao]s -> 'las' lub 'los' ale nie 'laos'
- [^lista] – jeden znak spoza listy
 - k[^oa]t -> 'kit', 'ket', ... ale nie 'kot' ani 'kat'
- * - zero lub więcej wystąpień poprzedzającego znaku
- ^ - początek linii (jeśli jest pierwszym znakiem wyrażenia)
- \$ - koniec linii (jeśli jest ostatnim znakiem wyrażenia)

Przykłady

- `[Kk]ot`
- `[Kk][ao]t`
- `s[aeiou]*k`
- `^\.`
- `^$.*\.$`
- `^[a-z]* 100`

Ograniczone wyrażenia regularne (ed, sed, grep)

- Podwyrażenia
 - `\( i \)`
 - odwołanie poprzez `\1`, `\2`, itd.
 - `Bim \(\Bom\) \1`
 - `Bim Bom Bom`
 - `\(hau\) \(\miau\) \2 \1`
 - `hau miau miau hau`
- rozszerzenia
 - `\{n\}` - dokładnie `n` wystąpień znaku poprzedzającego
 - `\{n,\}` - przynajmniej `n` wystąpień
 - `\{n,m\}` - od `n` do `m` wystąpień

Pełne wyrażenia regularne (egrep, awk)

- + - jedno lub więcej wystąpień poprzedniego wyrażenia
 - go+d -> 'god', 'good', 'goood', itd., ale nie 'gd'
- ? - zero lub jedno wystąpienie poprzedniego wyrażenia
 - go?d -> 'gd' lub 'god', ale nie 'good'
- | - jedno z dwóch rozdzielonych wyrażeń
 - kot|pies
- (oraz) - grupowanie wyrażeń
 - go(go)? -> 'go' lub 'gogo'
 - gogo? -> 'gog' lub 'gogo'
- priorytet (od najsilniejszego): nawiasy, *, ?, +, konkatenacja, |

Przykłady

- [a-z] +[0-9]
- [a-z]+[0-9]?
- Tony|Anthony
- Tony|Anthony Quinn
- (Tony|Anthony) Quinn
- (so?)+
 - tak: 'so', 'sos', 'sso', 'soso', 'sosssssosos', itd.
 - nie: 'soo', 'soos', itd.

Podstawowe narzędzia CLI (command line interface)

- Edytory:
 - ed
 - sed
 - vi
- Porównywanie plików:
 - diff (DOS: fc.exe)
 - patch
 - rcs/cvs/svn
- Przetwarzanie danych tekstowych
 - awk
 - perl

Edytor ed

- składnia: ed [-p prompt] [file] [< commands]
- tryby pracy (polecenia/tekstowy) – obsługa ograniczonych wyrażeń regularnych, praca na buforze
- znaczniki (adresy)
 - . - bieżąca linia, \$ - ostatnia linia, n – linia o numerze n, 'c - mark
 - /wyr/ - pierwsza linia odpowiadająca wyrażeniu, // - powtarza ostatnio użyty wzorzec
 - ?wyr? - poszukuje w tył
 - +, -, +n, -n (adresy wzgl. bieżącej linii) adr+n, adr-n (wzgl. adr)
 - adr1,adr2 – zakres, adr1;adr2 – zakres i bieżąca:=adr1
 - , = 1,\$
 - ; = .,\$
 -

Edytor ed - c.d.

- przykłady poleceń
 - [.]a – tryb tekstowy, pojedynczy znak '.' w linii kończy
 - [.]i – wstawia tekst przed linią
 - [..]d – kasuje
 - e file – wstawia plik
 - f name – zmienia nazwę
 - [.]kc – markuje jako c
 - [..]p, [..]l, [..]n – wypisuje linie, (l – znaki specjalne ósemkowo, n – z numerami)
 - [1,\$]r file, [\$]r !prog – wczytuje plik lub output programu
 - [1,\$]w file, [\$]r !prog – zapisuje do pliku lub jako input programu
 - ! prog – uruchamia program

Edytor ed - c.d.

- [1,\$]g/wyr/lista poleceń (każde polecenie w osobnej linii zakończonych \ - bez ostatniej)
- [1,\$]v/wyr/lista poleceń – działa gdy nie znajdzie wyr w linii
- [1,\$]s/wyr/napis/[g] – wymienia wyr na napis (z g – wszystkie wystąpienia)
- [.,.]m adr – move pod adres
- [.,.]t adr – copy za adr
- u – undo
- q - quit

Edytor strumieniowy sed

- składnia: `sed [-e 'skrypt'] [-f plik_skryptu] [-n (drukowanie tylko z p)]`
- bufor wzorca i cykl
- komendy:
 - `a\, c\, i\` - tekst za \
 - `: etykieta, b etykieta` – umożliwia skoki
 - `h (hold), g (get), x (exchange)` - obsługa bufora przechowywania
 - `s/wyr/napis/[1-512,g,p (print),w file` – dodaje bufor wzorca do pliku jeśli nastąpiła wymiana]
 - `y/napis1/napis2/ = tr`
 - GNU sed: adresowanie `first~step`, np. `sed -n -e '1~3 p'`

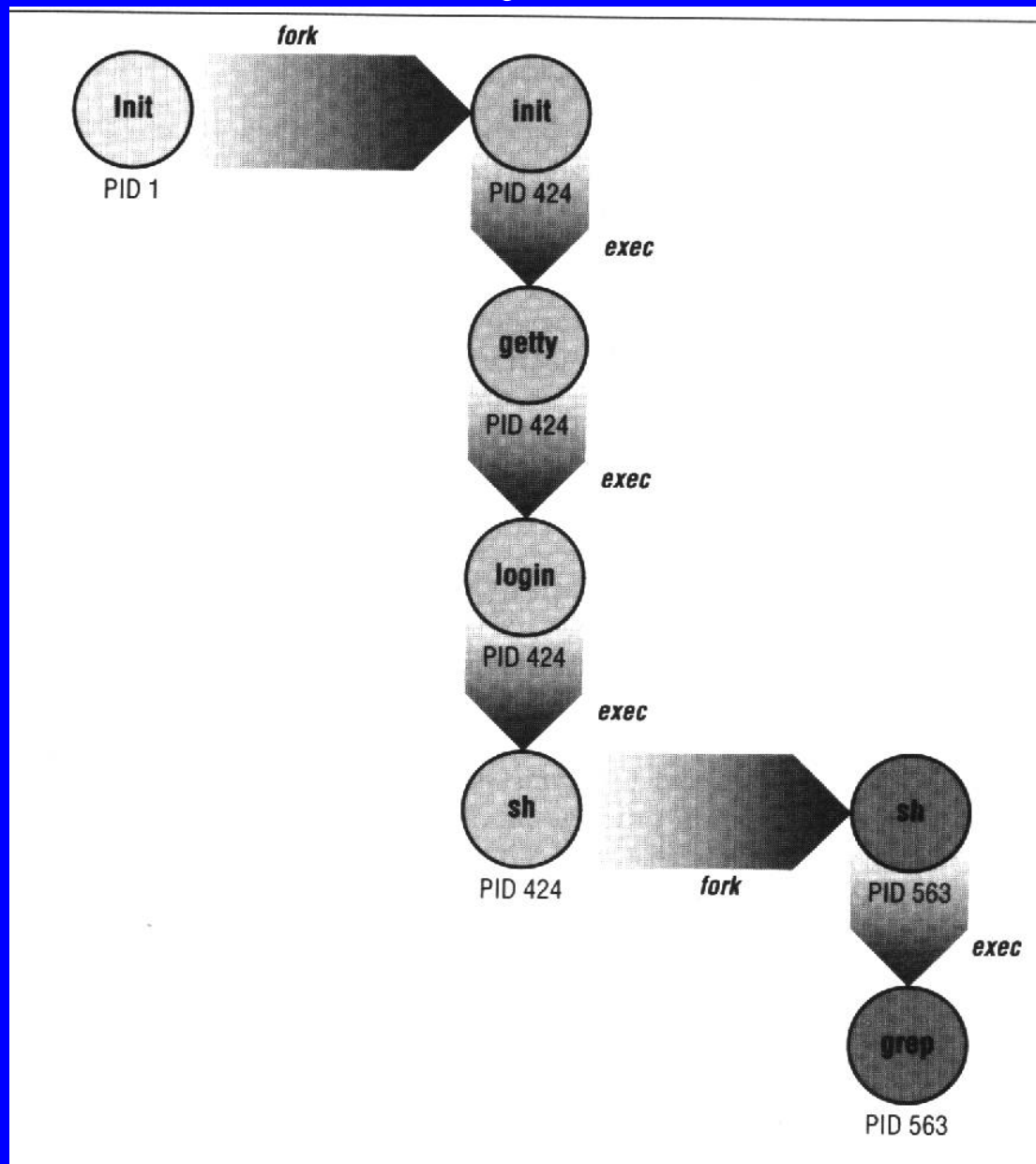
Procesy

- Proces: fragment programu posiadający własną przestrzeń adresową
 - np. ls – jeden proces
 - |, &&, || - każdy segment
- Typy procesów w systemie UNIX
 - interaktywne
 - terminal, foreground, background
 - batch
 - kolejka (que), nie są skojarzone z terminalem
 - daemon
 - czekają aż jakiś proces będzie wymagał ich akcji

Procesy - c.d.

- Cechy procesu:
 - PID
 - PPID
 - USER
 - UID (RUID, EUID), GID (RGID, EGID)
 - TTY
 - %CPU
 - %MEM
 - TIME
 - STIME
 - PRI
 - NI
 - STAT

Procesy - c.d.



Proces startu systemu UNIX

- bootstrapping (booting)
 - “The computer must pull itself up by its own bootstraps”
 - ROM
 - power ON
 - komenda wydana na konsoli systemowej
 - block 0
 - network
 - boot program
 - dodatkowy test hardware'u
 - czy jest dostępna pamięć
 - czy są podstawowe urządzenia zewnętrzne
 - kernel
 - inicjalizacja tablic
 - kompletny test pamięci i urządzeń (detekcja, tworzenie plików specjalnych)
 - fork – init: proces 1

Konfiguracja programu *init*

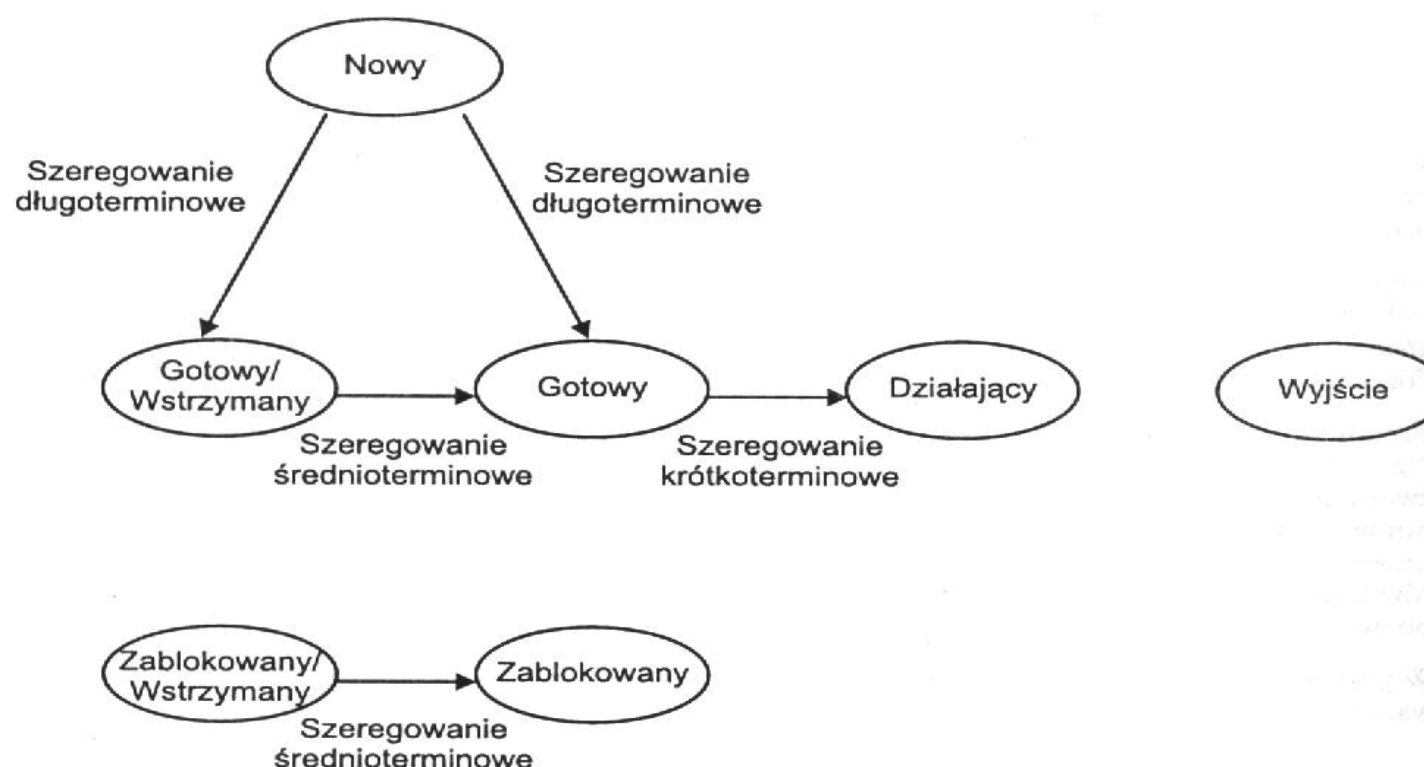
- single-user mode <-> multi-user mode
- BSD
 - /etc/rc
 - /etc/rc.local
- SunOS
 - /etc/rc.boot
 - /etc/rc.single
 - /etc/rc.net itd.
- Unix V
 - runlevels
 - /etc/inittab

Zarządzanie procesami nieinteraktywnymi

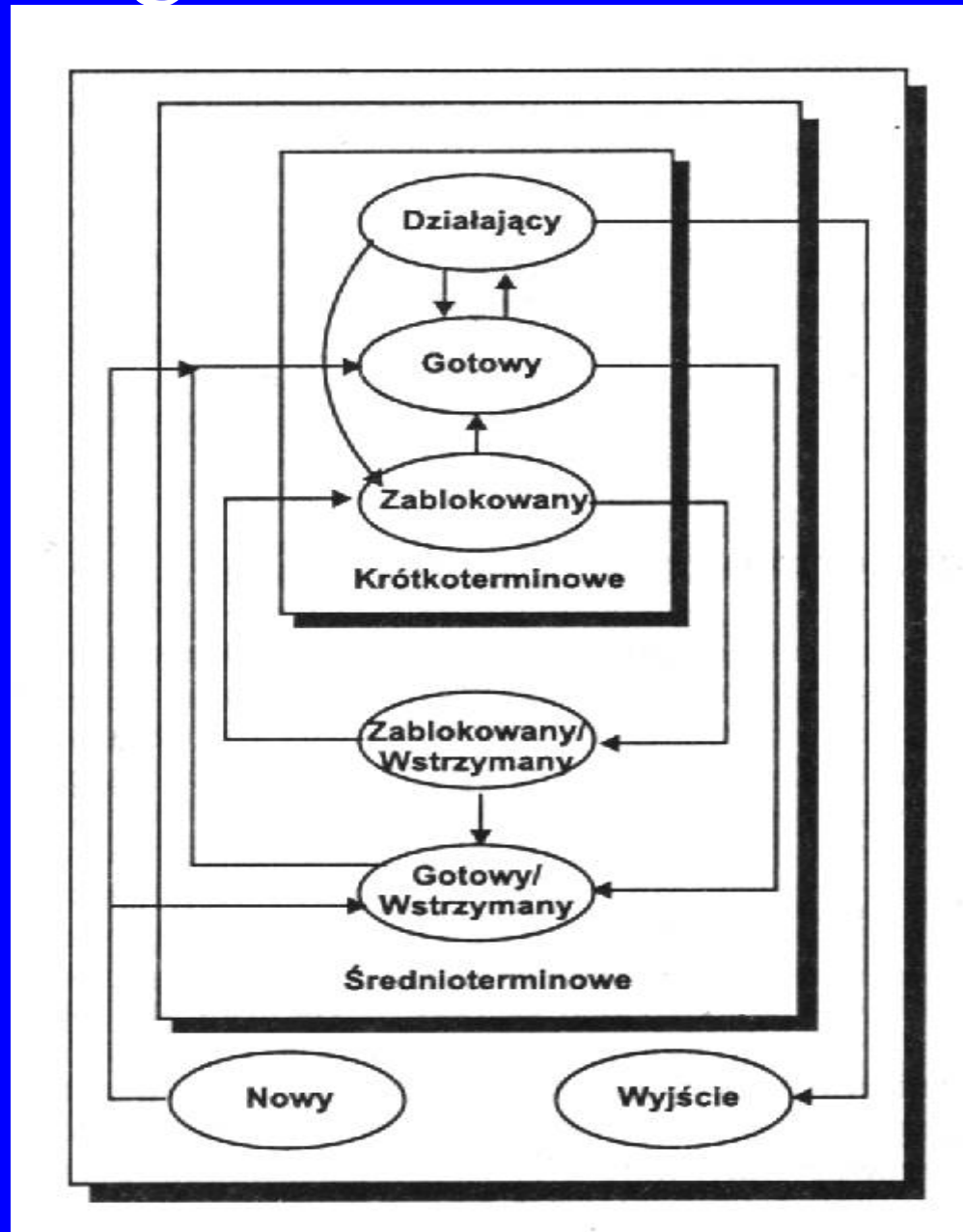
- **nohup**
 - SIGHUP, nohup.out
- **screen**
 - ^A + D (detach), -r
- **batch**
 - stdin, load average
- **at**
 - time + increment, atq, atrm, /etc/at.{allow,deny}
- **cron**
 - crontab, -l, -r, -e, /etc/cron.{allow,deny}

Szeregowanie zadań

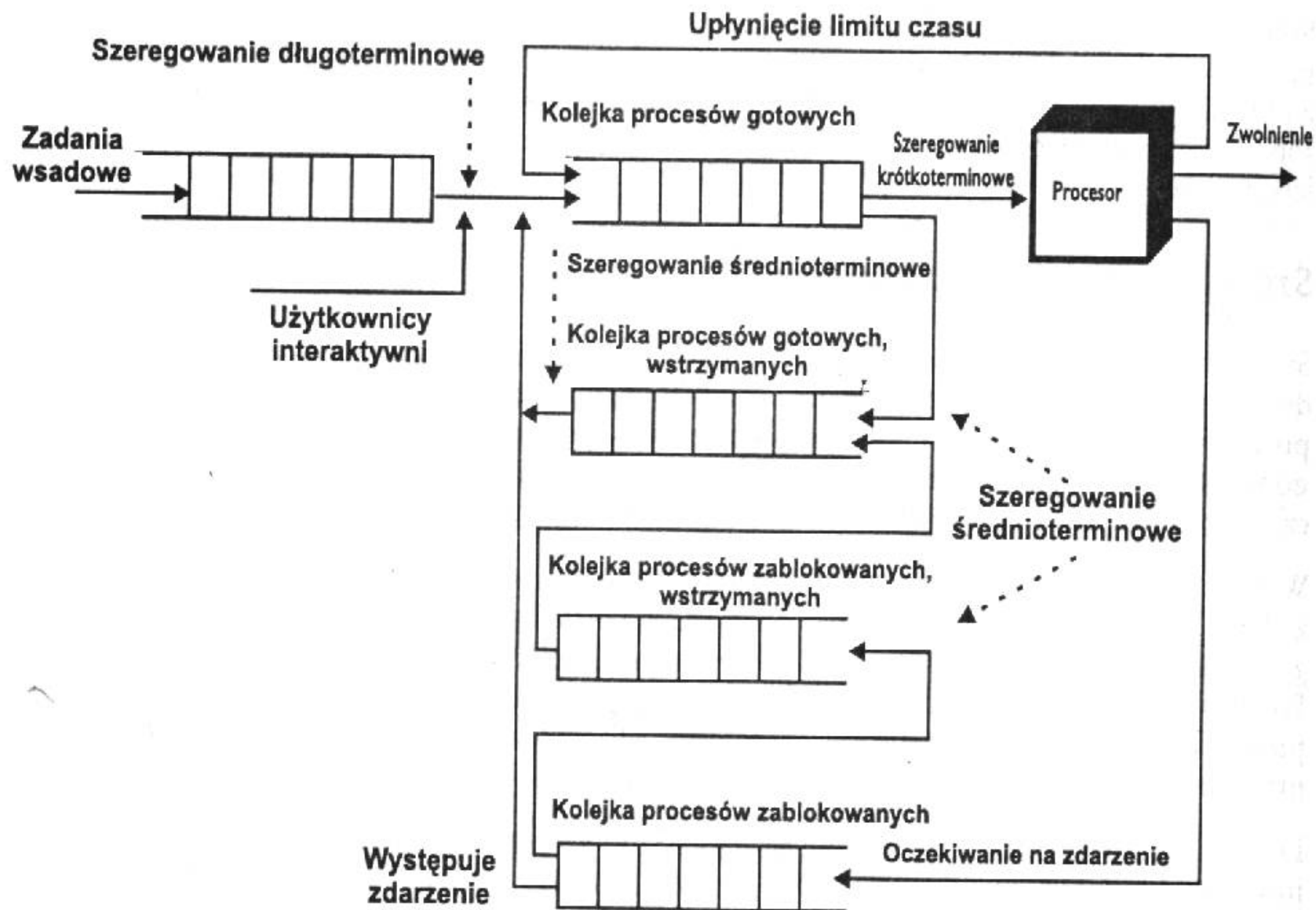
Szeregowanie długoterminowe	Decyzja o dodaniu procesu do puli procesów przeznaczonych do wykonania.
Szeregowanie średnioterminowe	Decyzja o dodaniu procesu do tych, które częściowo lub w całości znajdują się w pamięci głównej.
Szeregowanie krótkoterminowe	Decyzja dotycząca tego, który z dostępnych procesów będzie wykonany przez procesor.
Szeregowanie we-wy	Decyzja dotycząca tego, które z oczekujących żądań wejścia-wyjścia, zgłoszonych przez procesy, będzie obsłużone przez dostępne urządzenie wejścia-wyjścia.



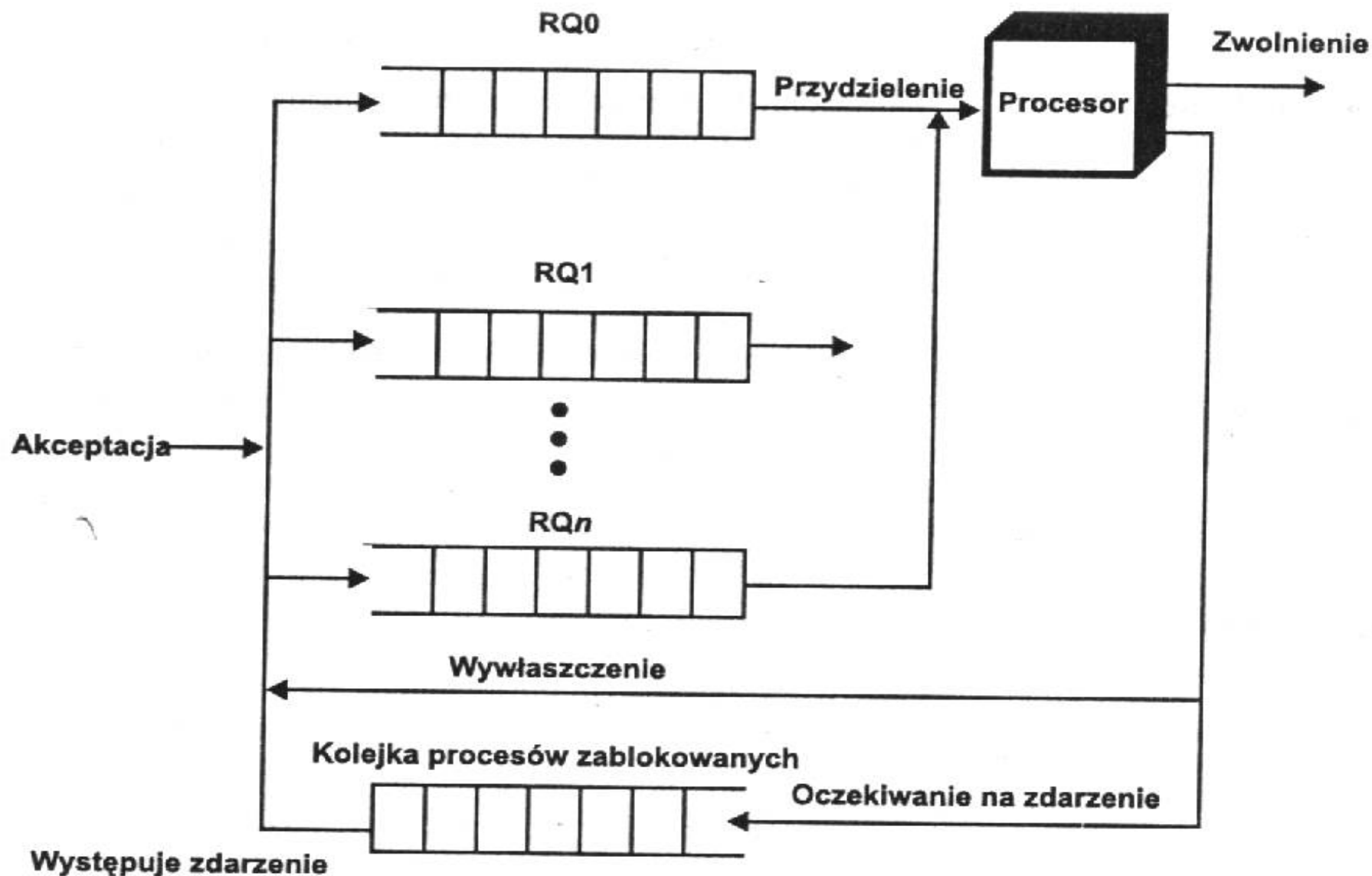
Szeregowanie zadań - c.d.



Kolejkowanie przy szeregowaniu



Kolejkowanie priorytetowe



Strategie szeregowania

Odnoszące się do użytkownika, związane z wydajnością

Czas przetwarzania

Przedział czasu od zgłoszenia procesu do jego finalizacji. Obejmuje rzeczywisty czas wykonywania oraz czas spędzony w oczekiwaniu na zasoby, w tym na procesor. Jest to odpowiednia miara dla zadania wsadowego.

Czas reakcji

W przypadku procesu interaktywnego jest to czas od zgłoszenia żądania do rozpoczęcia udzielania odpowiedzi. Często proces może zacząć wyświetlać jakieś wyniki, kontynuując przetwarzanie żądania. Dlatego z punktu widzenia użytkownika jest to lepsza miara niż czas przetwarzania. Reżim szeregowania powinien dążyć do zapewnienia krótkiego czasu reakcji i zmaksymalizowania liczby interaktywnych użytkowników, którzy uzyskują akceptowalny czas reakcji.

Nieprzekraczalne terminy

Jeśli da się określić nieprzekraczalne terminy finalizacji procesów, reżim szeregowania powinien podporządkowywać inne cele maksymalizacji odsetka terminów, których udało się dotrzymać.

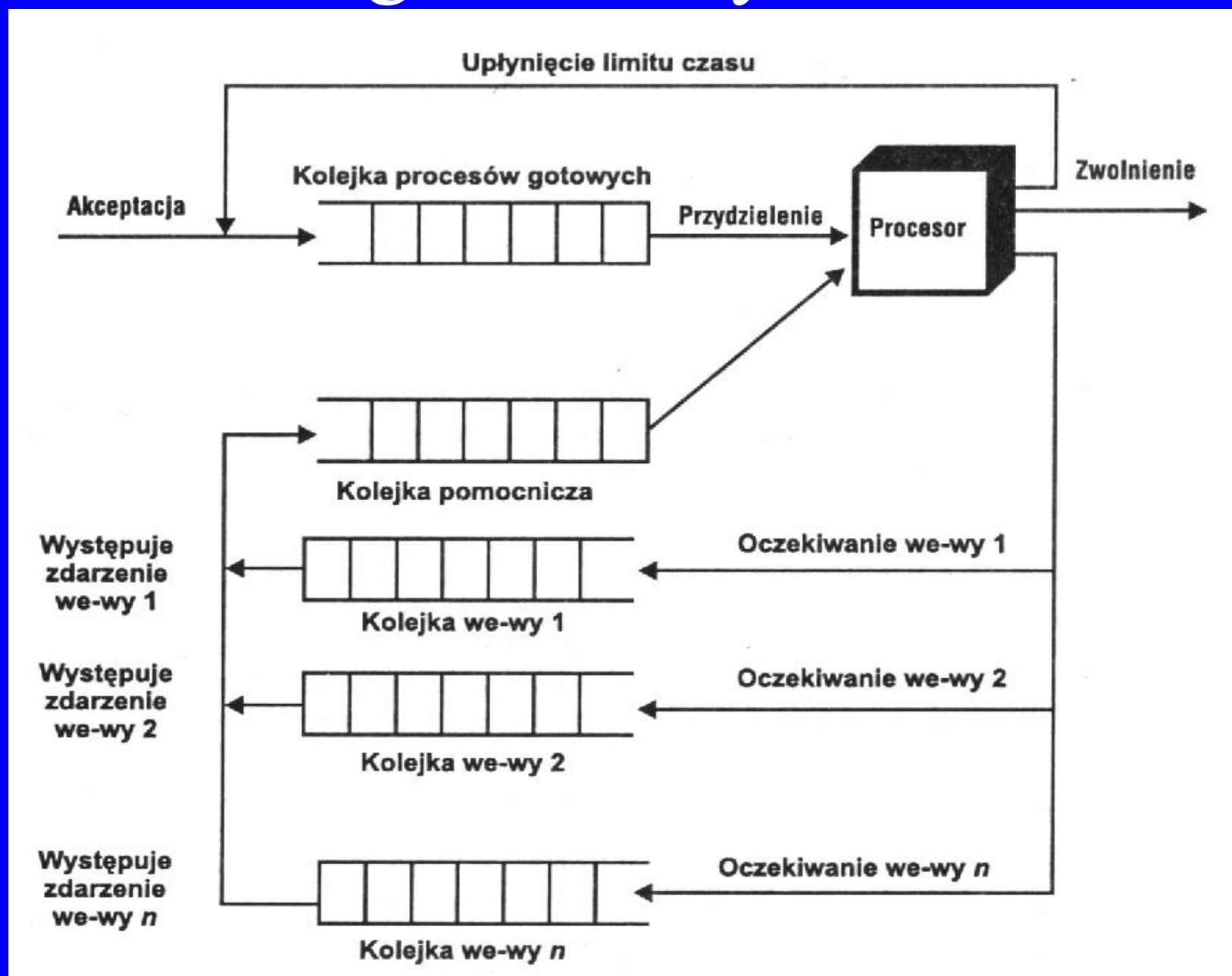
Strategie szeregowania - c.d.

	Odnoszące się do użytkownika, pozostałe
Przewidywalność	Dane zadanie powinno być wykonywane mniej więcej w tym samym czasie i mniej więcej tym samym kosztem bez względu na obciążenie systemu. Bardzo zróżnicowane czasy reakcji lub czasy przetwarzania przeszkadzają użytkownikom w pracy. Może to sygnalizować dużą zmienność obciążenia systemu albo konieczność dostrojenia systemu w celu wyeliminowania niestabilności.
	Odnoszące się do systemu, związane z wydajnością
Przepustowość	Strategia szeregowania powinna maksymalizować liczbę procesów finalizowanych w danej jednostce czasu. Jest to miara ilości wykonywanej pracy. Zależy ona oczywiście od długości procesu, ale w pewnej mierze także od strategii szeregowania, która może wpływać na wykorzystanie zasobów.
Wykorzystanie procesora	Jest to procent czasu, przez który procesor pozostaje zajęty. Kryterium to ma duże znaczenie w drogich systemach współdzielonych. W systemach jednodostępnych i niektórych innych, takich jak systemy czasu rzeczywistego, bywa mniej ważne od pozostałych kryteriów.
	Odnoszące się do systemu, pozostałe
Sprawiedliwość	Bez specjalnych wskazówek ze strony użytkownika lub systemu operacyjnego procesy powinny być traktowane jednakowo i żaden nie powinien ulegać zagłodzeniu.
Egzekwowanie priorytetów	Jeśli procesom przydziela się priorytety, strategia szeregowania powinna faworyzować procesy o wyższym priorytecie.
Równoważenie wykorzystania zasobów	Strategia szeregowania powinna dążyć do zrównoważonego wykorzystania zasobów systemu. Należy faworyzować procesy, które będą w minimalnym stopniu korzystać ze szczególnie obciążonych zasobów. To kryterium ma również znaczenie w szeregowaniu średnio- i długoterminowym.

Przykłady algorytmów szeregowania

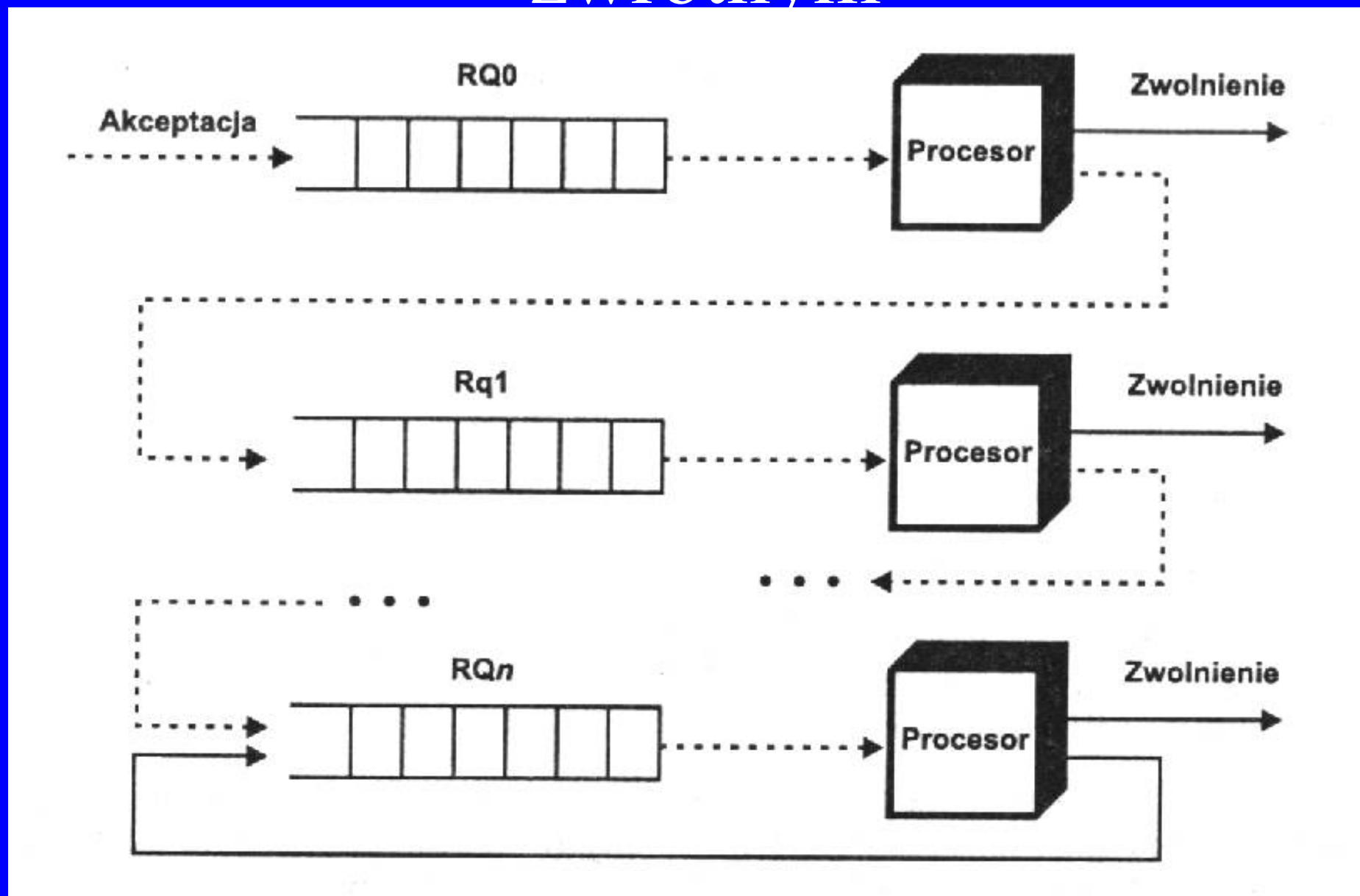
- ♦ Kto pierwszy, ten lepszy (*First-come-served*): Wybrać proces, który najdłużej czeka na obsługę.
- ♦ Szeregowanie cykliczne (*Round robin*): Użyć „cięcia czasu”, aby ograniczyć czas procesora przydzielany każdemu procesowi, i cyklicznie wykonywać wszystkie gotowe procesy.
- ♦ Najkrótszy proces jako następny (*Shortest process next*): Wybrać proces z najkrótszym przewidywanym czasem przetwarzania i nie wywłaszczać go.
- ♦ Najkrótszy pozostały czas (*Shortest remaining time*): Wybrać proces z najkrótszym przewidywanym czasem przetwarzania. Proces może zostać wywłaszczony, kiedy inny proces znajdzie się w stanie gotowości.
- ♦ Największy stosunek reaktywności jako następny (*Highest response ratio next*): Podjąć decyzję o szeregowaniu na podstawie szacunkowego znormalizowanego czasu przetwarzania.
- ♦ Sprzężenie zwrotne (*Feedback*): Utworzyć zbiór kolejek szeregowania i kierować procesy do kolejek na podstawie historii wykonywania i innych kryteriów.

Szeregowanie cykliczne



Szeregowanie ze sprzężeniem zwrotnym

10:58:12



Charakterystyka wybranych algorytmów

	Funkcja wyboru	Tryb decyzyjny	Przepustowość	Czas reakcji	Koszt dodatkowy	Wpływ na procesy	Zagłódzenie
FCFS	$\max[w]$	Niewywłaszczający	Nieakcentowana	Bywa długi, zwłaszcza jeśli istnieją duże różnice w czasie wykonywania procesów	Minimalny	Krzywdząca dla procesów krótkich oraz powiązanych z we-wy	Nie
Cykliczna	stała	Wywłaszczający (w odcinku czasu)	Może być mała, jeśli odcinek czasu jest za krótki	Dobry czas reakcji dla krótkich procesów	Minimalny	Sprawiedliwe traktowanie	Nie
SPN	$\min[s]$	Niewywłaszczający	Duża	Dobry czas reakcji dla krótkich procesów	Może być duży	Krzywdząca dla długich procesów	Możliwe
SRT	$\min[s - e]$	Wywłaszczający (w chwili przybycia)	Duża	Dobry czas reakcji	Może być duży	Krzywdząca dla długich procesów	Możliwe
HRRN	$\max\left(\frac{w + s}{s}\right)$	Niewywłaszczający	Duża	Dobry czas reakcji	Może być duży	Dobrze zrównoważona	Nie
Sprzężenie zwrotne	(patrz tekst)	Wywłaszczający (w odcinku czasu)	Nieakcentowana	Nieakcentowany	Może być duży	Może faworyzować procesy powiązane z we-wy	Możliwe

w = czas spędzony do tej pory w systemie (na oczekiwaniu i wykonywaniu)

e = czas spędzony do tej pory na wykonywaniu

s = całkowity czas obsługi wymagany przez proces, łącznie z e

Tradycyjne szeregowanie UNIX'a

$$CPU_j(i) = \frac{CPU_j(i-1)}{2}$$
$$P_j(i) = Baza_j + \frac{CPU_j(i-1)}{2} + nice_j$$

gdzie

$CPU_j(i)$ = Miara wykorzystania procesora przez proces j w okresie i

$P_j(i)$ = Priorytet procesu j na początku okresu i ; niższe wartości oznaczają wyższy priorytet

$Baza_j$ = Bazowy priorytet procesu j

$nice_j$ = Współczynnik ustalany przez użytkownika

Szeregowanie w LINUX'ie

- **SCHED_FIFO**: Wątki czasu rzeczywistego, pierwszy na wejściu – pierwszy na wyjściu
- **SCHED_RR**: Wątki czasu rzeczywistego, szeregowanie cykliczne
- **SCHED_OTHER**: Inne wątki, nie wykonywane w czasie rzeczywistym

W każdej klasie można używać wielu priorytetów, przy czym priorytety w klasach czasu rzeczywistego są wyższe niż w klasie **SCHED_OTHER**. W przypadku wątków FIFO obowiązują następujące reguły:

1. System nie przerywa wykonującego się wątku FIFO, z wyjątkiem poniższych przypadków:
 - a. Inny wątek FIFO o wyższym priorytecie staje się gotowy.
 - b. Wykonujący się wątek FIFO blokuje się w oczekiwaniu na zdarzenie, np we-wy.
 - c. Wykonujący się wątek FIFO dobrowolnie odstępuje procesor po wywołaniu funkcji pierwotnej *sched_yield*.
2. Kiedy wykonujący się wątek FIFO zostanie przerwany, jest umieszczany w kolejce związanej ze swoim priorytetem.
3. Kiedy wątek FIFO staje się gotowy i ma wyższy priorytet niż obecnie wykonujący się wątek, wówczas wykonujący się wątek zostaje wywłaszczony i wykonuje się gotowy wątek FIFO o najwyższym priorytecie. Jeśli więcej niż jeden wątek ma ten najwyższy priorytet, wybiera się wątek, który czekał najdłużej.

Przykład

	Proces A		Proces B		Proces C	
Czas	Priorytet	Licznik procesora	Priorytet	Licznik procesora	Priorytet	Licznik procesora
0	60	0 1 2 . . 60	60	0	60	0
1	75	30	60	0 1 2 . . 60	60	0
2	67	15	75	30	60	0 1 2 . . 60
3	63	7 8 9 . . 67	67	15	75	30
4	76	33	63	7 8 9 . . 67	67	15
5	68	16	76	33	63	7

Przykład

A	Minimalny
B	Średni
C	Średni
D	Maksymalny

a) Względne priorytety wątków



b) Przepływ z szeregowaniem FIFO



c) Przepływ z szeregowaniem RR

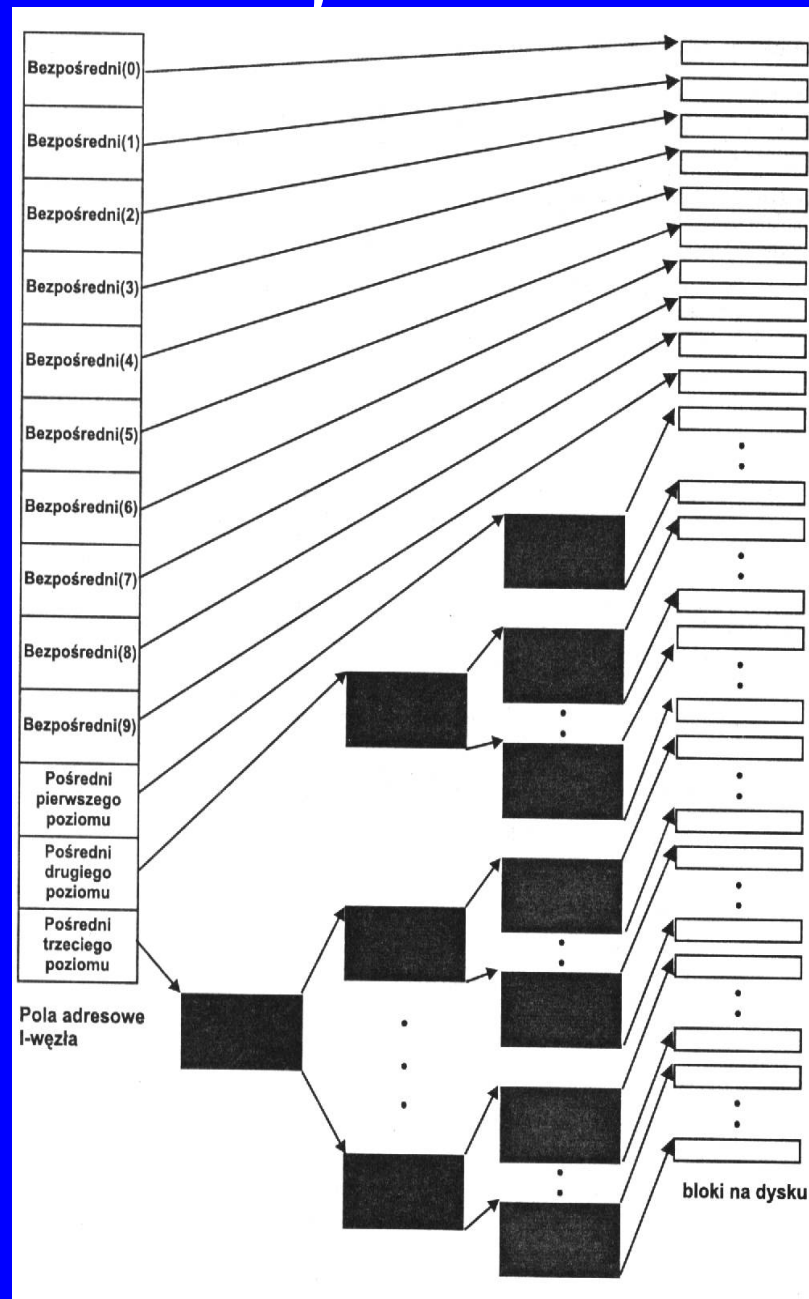
Szeregowanie w UNIX V

Klasa priorytetu	Globalna wartość	Sekwencja szeregowania
Czas rzeczywisty	159	Pierwszy
	•	
	•	
	•	
	•	
	100	
Jądro	99	
	•	
	•	
	60	
Podział czasu	59	↓
	•	
	•	
	•	
	•	
	0	
		Ostatni

System plików - struktura i-node'u

Tryb pliku	16-bitowy znacznik określający uprawnienia dostępu i wykonania danego pliku. 12-14 Typ pliku (regularny, katalog, znakowy lub specjalny blokowy, potok FIFO). 9-11 Znaczniki wykonania. 8 Uprawnienia odczytu właściciela. 7 Uprawnienia zapisu właściciela. 6 Uprawnienia wykonania właściciela. 5 Uprawnienia odczytu dla grupy. 4 Uprawnienia zapisu dla grupy. 3 Uprawnienia wykonania dla grupy. 2 Inne uprawnienia odczytu. 1 Inne uprawnienia zapisu. 0 Inne uprawnienia wykonania.
Licznik odwołań	Liczba odwołań katalogów do tego I-węzła.
ID właściciela	Indywidualny właściciel pliku.
ID Grupy	Grupowy właściciel pliku.
Wielkość pliku	Liczba bajtów pliku.
Adres pliku	Informacje adresowe, 39 bajtów.
Ostatni dostęp	Data ostatniego dostępu do pliku.
Ostatnia modyfikacja	Data ostatniej modyfikacji pliku.
Modyfikacja I-węzła	Data ostatniej modyfikacji tego I-węzła.

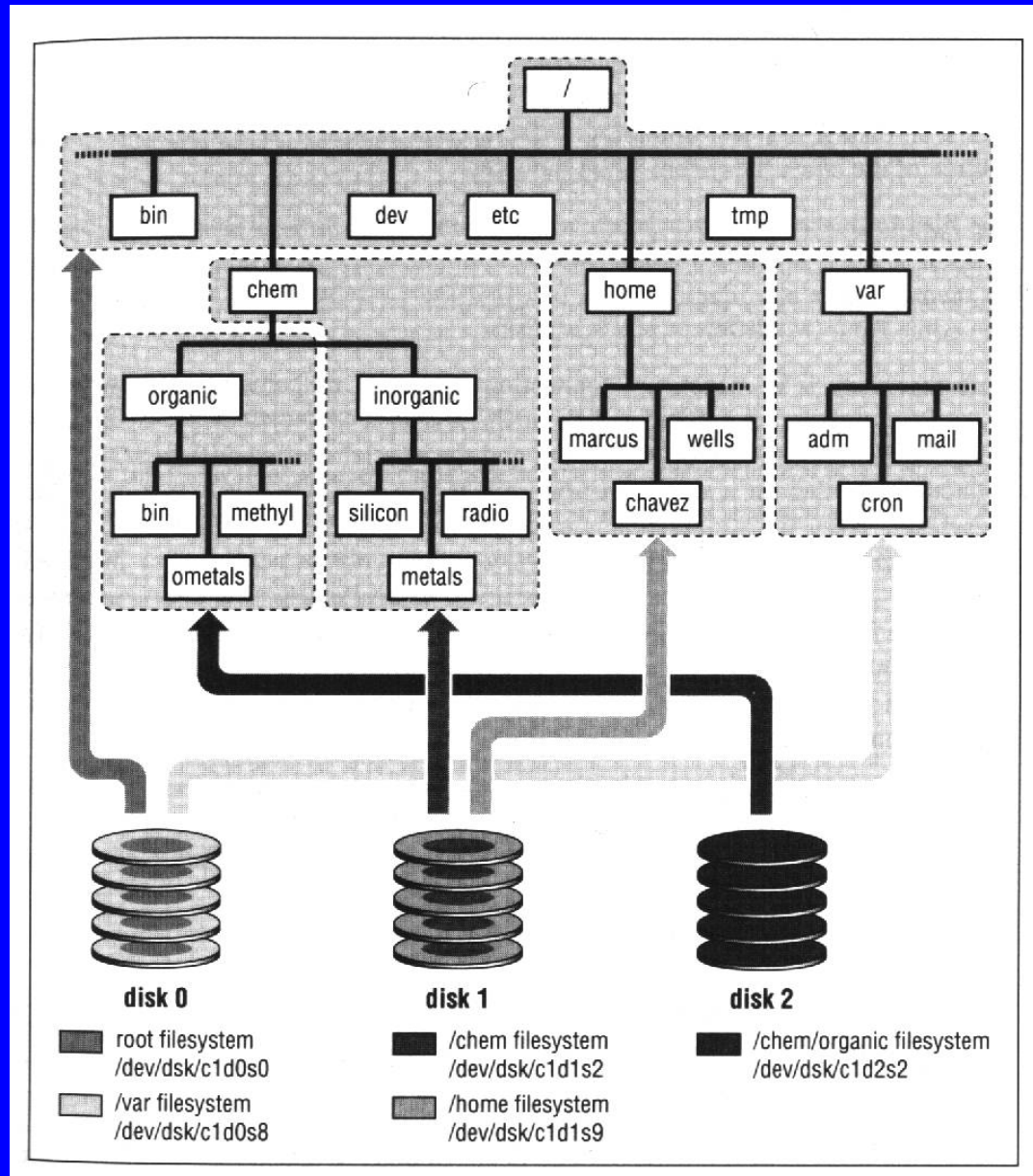
Alokacja i-node'ów



Alokacja - c.d.

Poziom	Liczba bloków	Liczba bajtów
Bezpośredni	10	10K
Pierwszy poziom pośredni	256	256 K
Drugi poziom pośredni	$256 \times 256 = 65 \text{ K}$	65 M
Trzeci poziom pośredni	$256 \times 65 \text{ K} = 16 \text{ M}$	16 G

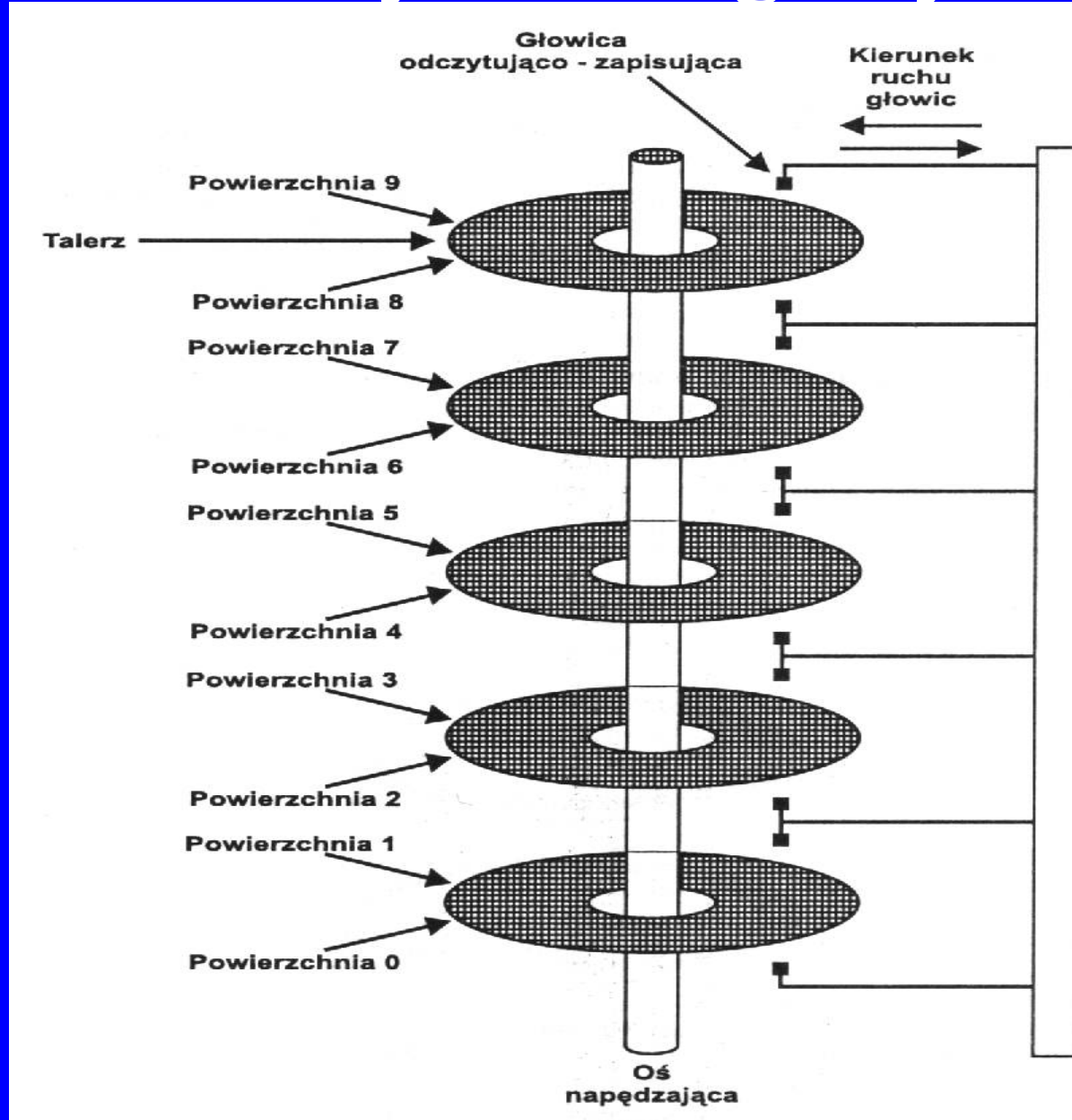
Montowanie



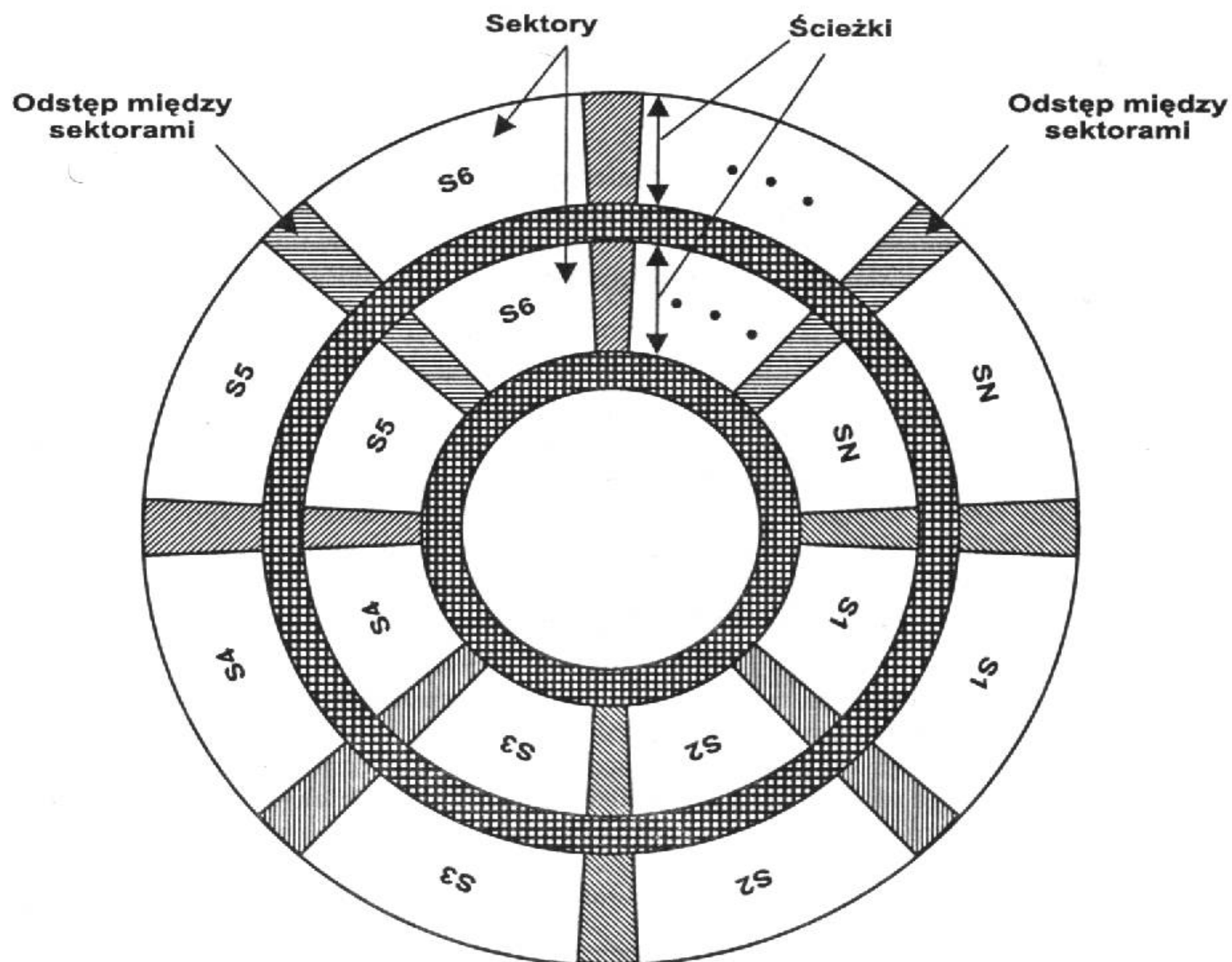
Zarządzanie systemem plików

- fdisk
- format (DOS format), mkfs (UnixV), newfs (BSD), mknod (MAKEDEV)
- fsck
- mount
- umount
- stat
- df
- du
- mtools (ltools)

Budowa dysku magnetycznego



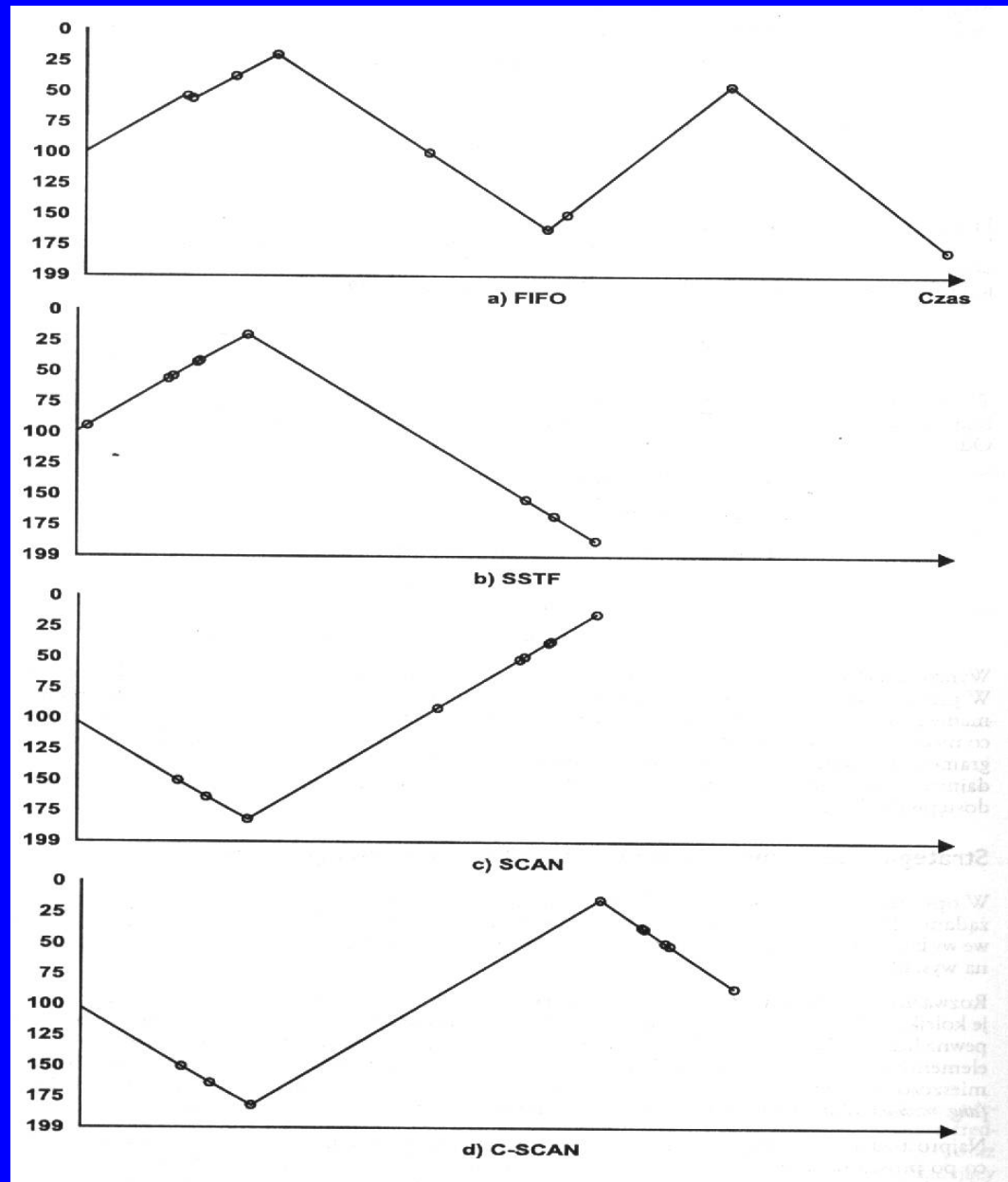
Budowa dysku -c.d.



Przykładowe strategie buforowania

Nazwa	Opis	Uwagi
Wybór według żądającego		
RSS	Szeregowanie losowe	Do analiz i symulacji
FIFO	Pierwszy na wejściu, pierwszy na wyjściu	Najsprawiedliwszy ze wszystkich
PRI	Według priorytetu procesu	Poza kontrolą oprogramowania zarządzającego kolejką dysku
LIFO	Ostatni na wejściu, pierwszy na wyjściu	Maksymalizuje lokalność i poziom wykorzystania zasobów
Wybór według żądanego elementu		
SSTF	Najpierw zadanie o najkrótszym czasie obsługi	Wysoki poziom wykorzystania, krótkie kolejki
SCAN	Tam i z powrotem przez cały dysk	Lepsza dystrybucja obsługi
C-SCAN	W jedną stronę z szybkim powrotem	Mniejsza zmienność poziomu usług
N-step-SCAN	SCAN N rekordów jednocześnie	Gwarancja obsługi
FSCAN	N-step-SCAN, gdzie N = rozmiar kolejki na początku cyklu SCAN	Wrażliwy na obciążenie

Buforowanie - c.d.



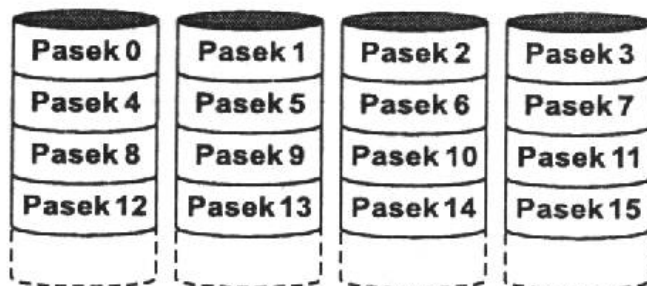
Buforowanie - c.d.

(a) FIFO (zaczynając od ścieżki 100)		(b) SSTF (zaczynając od ścieżki 100)		(c) SCAN (zaczynając od ścieżki 100, w kierunku ścieżek o wyższych numerach)		(d) C-SCAN (zaczynając od ścieżki 100, w kierunku ścieżek o wyższych numerach)	
Następna ścieżka	Liczba przebytych ścieżek	Następna ścieżka	Liczba przebytych ścieżek	Następna ścieżka	Liczba przebytych ścieżek	Następna ścieżka	Liczba przebytych ścieżek
55	45	90	10	150	50	150	50
58	3	58	32	160	10	160	10
39	19	55	3	184	24	184	24
18	21	39	16	90	94	18	166
90	72	38	1	58	32	38	20
160	70	18	20	55	3	39	1
150	10	150	132	39	16	55	16
38	112	160	10	38	1	58	3
184	146	184	24	18	20	90	32
Średni czas wyszukiwania	55,3	Średni czas wyszukiwania	27,5	Średni czas wyszukiwania	27,8	Średni czas wyszukiwania	35,8

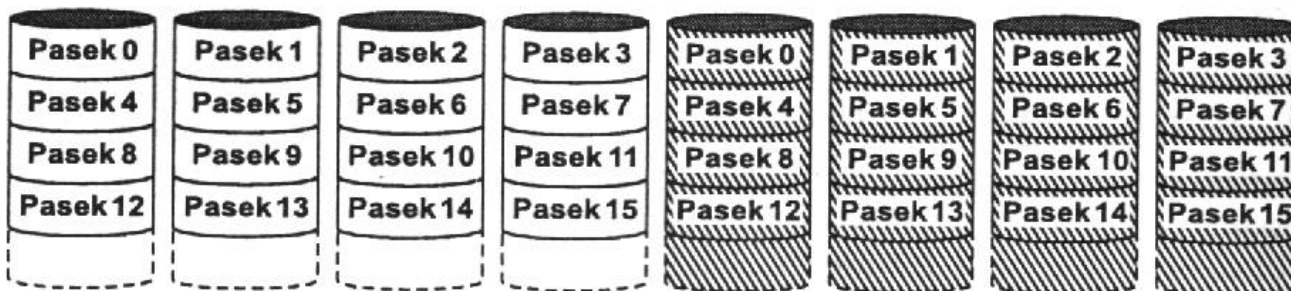
Poziomy RAID

Kategoria	Poziom	Opis	Tempo żądań we-wy (zapis/ odczyt)	Szybkość transmisji danych (zapis/odczyt)	Typowe zastosowanie
Paskowanie	0	Brak nadmiarowości	Duże paski: doskonałe	Małe paski: doskonała	Aplikacje wymagające wysokiej wydajności przetwarzania niekrytycznych danych
Kopiowanie lustrzane	1	Kopia lustrzana	Dobre/średnie	Średnia/średnia	Dyski systemo- we, pliki krytyczne
Dostęp równoległy	2	Nadmiarowość z wykorzysta- niem kodu Hamminga	Niskie	Doskonała	
	3	Parzystość z przeplotem bitów	Niskie	Doskonała	Aplikacje z dużymi żądania- mi we-wy, np. przetwarzanie obrazów, CAD
Dostęp niezależny	4	Parzystość z przeplotem bloków	Doskonałe/ średnie	Średnia/niska	
	5	Rozproszona parzystość z przeplotem bloków	Doskonałe/ średnie	Średnia/niska	Duże tempo żądań, dużo operacji odczy- tu, wyszukiwa- nie danych
	6	Podwójnie rozproszona parzystość z przeplotem bloków	Doskonałe/ niskie	Średnia/niska	Aplikacje wymagające bardzo wysokiej niezawodności

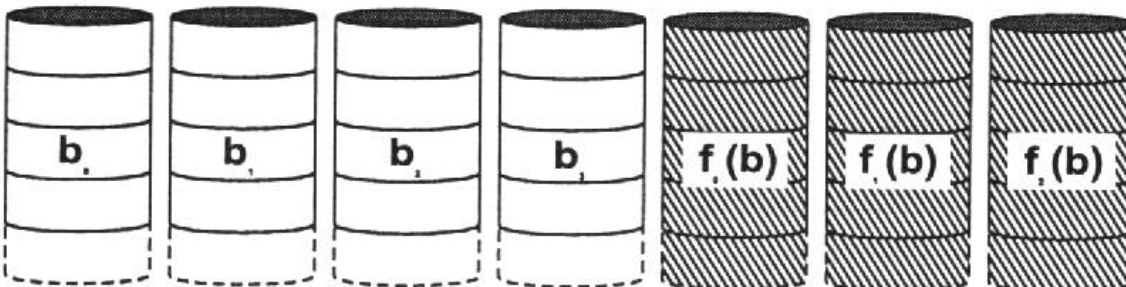
RAID 0-2



a) RAID 0 (brak nadmiarowości)

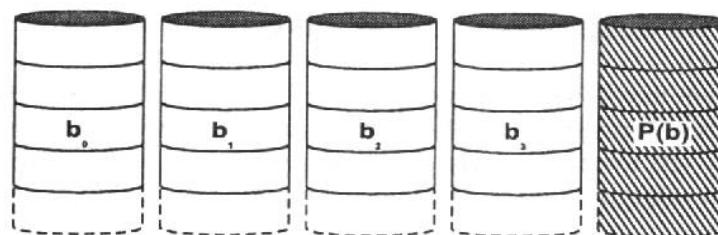


b) RAID 1 (kopia lustrzana)

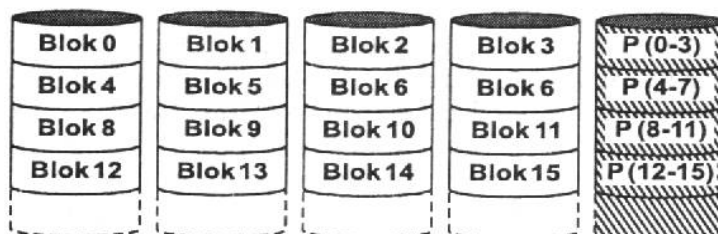


c) RAID 2 (nadmiarowość z wykorzystaniem kodu Hamminga)

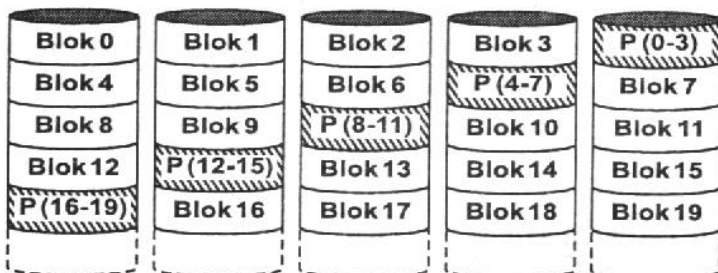
RAID 3-6



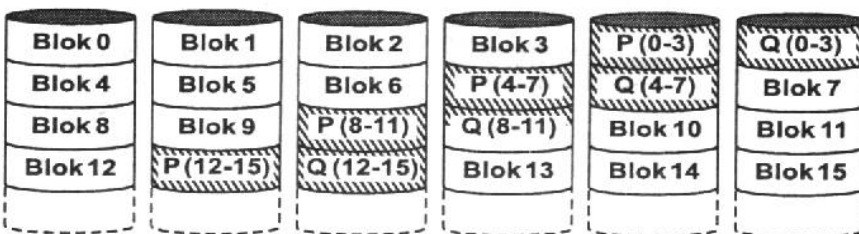
d) RAID 3 (parzystość z przeplotem bitów)



e) RAID 4 (parzystość z przeplotem bloków)



f) RAID 5 (rozproszona parzystość z przeplotem bloków)



g) RAID 6 (podwójna nadmiarowość)

UNIX'owe narzędzia sieciowe

- diagnostyka
 - ping, traceroute, netstat, nslookup, nmap, tcpdump
- konfiguracja
 - loopback (127.0.0.1)
 - ifconfig, route, hostname, dhcp(c)d
- pliki
 - /etc/hosts
 - /etc/resolv.conf
- zdalne połączenie
 - telnet, rsh (ssh)
 - ftp (sftp)

Rutynowe prace administracyjne w UNIX'ie

- root
 - su, sudo (/etc/sudoers)
- użytkownicy
 - passwd, useradd (adduser), userdel (deluser), group, etc.
- komunikacja
 - write, wall, mesg, who, w, finger, last
 - /etc/motd (.hushlogin)
 - mail, talk
- oprogramowanie
 - ./configure, make, gcc
- logi systemowe (np. w /var/log/)