# Wprowadzenie do Informatyki

Notatki do wykładu
dla I roku astronomii

Wersja 0.98.2, 2006-10-31



Tomasz Kwiatkowski
Obserwatorium Astronomiczne
UAM, Poznań

# Spis treści

# Rozdział 1

# Wstęp

## 1.1 Wiek informacji

- Żyjemy w wieku informacji, przemiany dotykają wszystkich aspektów życia
- Zagadnieniem centralnym staje się zdobywanie, przechowywanie, przekazywanie, przetwarzanie i interpretowanie informacji
- Pewną analogią z przeszłości może być rozpowszechnienie druku w średniowiecznej Europie, co umożliwiło radykalne zmiany w światopoglądzie ludzi

## 1.2 Przykładowe zagadnienia wieku informacji

- Globalna ekonomia: transakcje handlowe w sieci, outsourcing, giełda (zakupy akcji on-line)
- Globalna nawigacja: system GPS
- Globalna wioska: łatwość komunikowania się (Internet łączy się z telefonią komórkową)
- Prawo Matcalfa: użyteczność sieci wzrasta z kwadratem liczby jej użytkowników
- Lawinowo wzrasta ilość informacji w sieci
- Konieczność sprawnego wydobywania potrzebnych informacji z zalewu danych (wyszukiwarki sieciowe, data mining, virtual observatories)

- Problem prywatności (szyfrowanie, podpis elektroniczny)
- Problem cenzury (terroryzm, rasizm, pornografia)
- Problem własności informacji: programów, literatury, muzyki, filmów (nielegalne kopiowanie informacji);
  IP (Intellectual Property): copyrights, patents, trademarks
- Problem wiarygodności informacji: (coraz więcej nieprawdziwych danych; wikipedia vs. citizendium)

# Rozdział 2

# Informacja i sposób jej zapisu

## 2.1 Podstawowe pojęcia

- Każdy czynnik zmniejszający stopień niewiedzy o jakimś zjawisku czy obiekcie nazywamy *informacją*.
- Naukę zajmującą się zagadnieniami pobierania, przechowywania, przesyłania, przetwarzania i interpretowania informacji nazywamy *informatyką*.
- Komputer gromadzi i przetwarza informację zapisaną wyłącznie w postaci liczbowej, w systemie dwójkowym.

## 2.2 Systemy liczbowe

### 2.2.1 System dziesiętny

- 10 cyfr: 0,1,2,...,9
- zapis liczby naturalnej:

$$L_{(10)} = c_0 \cdot 10^0 + c_1 \cdot 10^1 + ... + c_n \cdot 10^n \tag{2.1}$$

  gdzie: $c_0$, $c_1$,...,$c_n$ — cyfry układu

- przykład:

$$2548 = 8 \cdot 10^0 + 4 \cdot 10^1 + 5 \cdot 10^2 + 2 \cdot 10^3 \tag{2.2}$$

### 2.2.2 System dwójkowy

- 2 cyfry: 0,1
- zapis liczby naturalnej:

$$L_{(2)} = c_0 \cdot 2^0 + c_1 \cdot 2^1 + ... + c_n \cdot 2^n \tag{2.3}$$

gdzie: $c_0$, $c_1$,...,$c_n$ — cyfry układu

- przykład:

$$\begin{aligned} 10110_{(2)} &= 0 \cdot 2^0 + 1 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 + 1 \cdot 2^4 && (2.4) \\ &= 0 + 2 + 4 + 0 + 16 = 22_{(10)} && (2.5) \end{aligned}$$

| Zapis w systemie dwójkowym | | | | | | | | Dziesiętny | Szesnastkowy |
|---|---|---|---|---|---|---|---|---|---|
| $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | 2 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 3 | 3 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 4 | 4 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 5 | 5 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 6 | 6 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 7 | 7 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 8 | 8 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 9 | 9 |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 10 | A |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 11 | B |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 12 | C |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 13 | D |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 14 | E |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 15 | F |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 16 | 10 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 127 | 7F |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 255 | FF |

### 2.2.3 System szesnastkowy

- 16 cyfr: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

- zapis liczby naturalnej:

$$L_{(16)} = c_0 \cdot 16^0 + c_1 \cdot 16^1 + ... + c_n \cdot 16^n \tag{2.6}$$

gdzie: $c_0$, $c_1$,...,$c_n$ — cyfry układu

- przykład:

$$\begin{aligned} 400_{(16)} &= 0 \cdot 16^0 + 0 \cdot 16^1 + 4 \cdot 16^2 & (2.7) \\ &= (0 + 0 + 4 \cdot 256)_{(10)} = 1024_{(10)} & (2.8) \\ FF_{(16)} &= F \cdot 16^0 + F \cdot 16^1 & (2.9) \\ &= (15 + 15 \cdot 16)_{(10)} = 255_{(10)} & (2.10) \end{aligned}$$

## 2.3 Konwersja liczb pomiędzy różnymi systemami

### 2.3.1 Dwójkowy –> szesnastkowy

- rozbijamy liczbę na czteroznakowe fragmenty:

$$100110101001111_{(2)} = (0100)\ (1101)\ (0100)\ (1111).$$

W powyższym przykładzie dodano na początku zero, by dopełnić ostatni fragment do pełnej czwórki (co nie zmienia samej liczby).

- korzystając z tabeli, przedstawiającej zapis liczb w systemie dwójkowym, możemy przypisać każdej czwórce odpowiadająca jej liczbę dziesiętną, a następnie szesnastkową:

$$\begin{array}{cccc} 0100_{(2)} & 1101_{(2)} & 0100_{(2)} & 1111_{(2)} \\ 4_{(10)} & 13_{(10)} & 4_{(10)} & 15_{(10)} \\ 4_{(16)} & D_{(16)} & 4_{(16)} & F_{(16)} \end{array}$$

- podobnie postępujemy, chcąc zamienić liczbę z systemu szesnastkowego, na system dwójkowy: każdą cyfrę szesnastkową zamieniamy na ciąg czterech zer lub jedynek.

### 2.3.2 Dziesiętny –> dwójkowy

- dzielimy z resztą przez 2
- przykład:

$$\begin{array}{rcrcl}
43:2 & = & 21 & \text{r.} & 1 \\
21:2 & = & 10 & \text{r.} & 1 \\
10:2 & = & 5 & \text{r.} & 0 \\
5:2 & = & 2 & \text{r.} & 1 \\
2:2 & = & 1 & \text{r.} & 0 \\
1:2 & = & 0 & \text{r.} & 1
\end{array}$$

Czytając prawą kolumnę zer i jedynek od dołu w górę, otrzymujemy wynik: $43_{(10)} = 101011_{(2)}$

## 2.4 Kodowanie informacji

- *bit* — podstawowa jednostka informacji, kodowana poprzez znaki 0,1
- *bajt* — 8 bitów

### 2.4.1 Kodowanie tekstu

**ASCII**

- kody ASCII (*American Standard Code for Information Interchange*)
- zapisywane w jednym bajcie, można w ten sposób zakodować 256 różnych znaków
- ASCII obejmuje:
  - 26 małych liter alfabetu łacińskiego
  - 26 dużych liter alfabetu łacińskiego
  - 10 cyfr
  - spację
  - znaki specjalne, np. `! "#$% &`
  - znaki sterujące (kody ASCII od 0 do 31), np. przejdź do nowego wiersza (oznaczenie LF od *Line Feed*), powrót karetki do początku wiersza (CR, od słów *Carriage Return*), tabulator, koniec tekstu (EOT, od słów *End of Text*)
- kody ASCII powyżej 127 to tzw. zestaw rozszerzony; zapisuje się w nim znaki narodowe i znaki semigrafiki (symbole, pozwalające tworzyć na ekranie ramki itp.)

**Unicode**

- Rozszerza kodowanie znaków na wszystkie dostępne alfabety i zestawy symboli (matematycznych, muzycznych itp.). Zakodowano nawet symbole fikcyjnego alfabetu Tengwar z „Władcy pierścienia” Tolkiena.
- Unicode to zestaw tablic przypisujących znakom graficznym liczby całkowite. UCS-2 (Universal Character Set-2) i UCS-4 to kodowania dwubajtowe i czterobajtowe
- Przykład: znak `©` ma kod `U+00A9`
- Kodowanie znaków ASCII:

| Znak | ASCII | UCS-2 | UCS-4 |
|---|---|---|---|
| `A` | `0x41` | `0x00 0x41` | `0x00 0x00 0x00 0x41` |

- Problemy z UCS-2 i UCS-4: w Unixie istnieje wiele programów, które działają w oparciu o 7. bitowe ASCII
- Dlatego stworzono UTF-8 (UCS Transformation Format)
  Znaki UCS od U+0000 do U+007F odpowiadają wiernie znakom ASCII od 0x00 do 0x7F (czyli są jedno-bajtowe)
- Znaki USC powyżej kodu U+007F kodowane są na dwu i więcej bajtach (w praktyce rzadko więcej, niż trzech)
- Przykład:

| Znak | UCS-2 | UTF-8 |
|---|---|---|
| `©` | `U+00A9` | `0xC2 0xA9` |
| $\neq$ | `U+2260` | `0xE2 0x89 0xA0` |

### 2.4.2 Kodowanie liczb całkowitych

- Różne systemy kodowania, np. system ZM (znak-mantysa; Sign Magnitude Notation) oraz U1 (uzupełnienie do jednego; 1’s complement) i U2 (uzupełnienie do dwóch; 2’s complement)
- 
- Przykład zapisu ZM dla liczb jedno-bajtowych: 7 bitów na liczbę i 1 bit na znak (0 dla $+$, 1 dla $-$)

- Zamiana z ZM na U1: zostaw bit znaku, dla liczb ujemnych zamień 0 na 1, a 1 na 0
- Zamiana z ZM na U2: zamień na U1 a potem do mantysy dodaj 1 (eliminuje to podwójne oznaczenie dla zera)
-

| Liczba | ZM | U1 | U2 |
|---|---|---|---|
| $-128$ | brak | brak | `1 0000000` |
| $-127$ | `1 1111111` | `1 0000000` | `1 0000001` |
| ... | .......... | .......... | .......... |
| $-1$ | `1 0000001` | `1 1111110` | `1 1111111` |
| 0 | `1 0000000` | `1 1111111` | brak |
| 0 | `0 0000000` | `0 0000000` | `0 0000000` |
| 1 | `0 0000001` | `0 0000001` | `0 0000001` |
| ... | .......... | .......... | .......... |
| 126 | `0 1111110` | `0 1111110` | `0 1111110` |
| 127 | `0 1111111` | `0 1111111` | `0 1111111` |

- Typowa długość liczb całkowitych:

| | | |
|---|---|---|
| *short int* | — | 2 bajty |
| *long int* | — | 4 bajty |

- minimalna i maksymalna liczba, którą można zapisać na $n$ bitach:

$$L_{min} = -2^{n-1}, \quad L_{max} = 2^{n-1} - 1$$

- dla *short int*:

$$L_{min} = -2^{16-1} = -32768, \quad L_{max} = 2^{16-1} - 1 = 32767$$

- dla *long int*:

$$L_{min} = -2^{32-1} = -2147483648, \; L_{max} = 2^{32-1} - 1 = 2147483647$$

### 2.4.3 Kodowanie liczb rzeczywistych

- Zapis liczby $L$ w postaci: $L = M \times P^w$,

| | | |
|---|---|---|
| $M$ | — | mantysa |
| $P$ | — | podstawa systemu liczbowego |
| $W$ | — | wykładnik |

- Przykład dla systemu dziesiętnego:

$$
\begin{aligned}
32.246 &= 0.32246 \times 10^2 \\
0.000183 &= 0.183 \times 10^{-3} \\
-84239253.5 &= -0.842392535 \times 10^8
\end{aligned}
$$

- Przykład dla systemu dwójkowego: $625.625$

  1. $625 = 1001110001$
  2. $0.625 = 0.5 + 0.125 = 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} = 0.101$
  3. $625.625 = 1001110001.101$
  4. $625.625 = 0.1001110001101 \times 2^{10}$

- Standard IEEE 754 zapisu liczb rzeczywistych:

| Nazwa | Długość | | | |
|---|---|---|---|---|
| | Całkowita | Znak | Wykładnik | Mantysa |
| float | 4 bajty | 1 bit | 8 bitów | 23 bity |
| double | 8 bajtów | 1 bit | 45 bity | 48 bitów |

- Liczby float nazywa się czasem liczbami w pojedynczej precyzji, a double — w podwójnej

- Po zamianie na liczby dziesiętne, float udostępnia 7 lub 8 cyfr znaczących, a double: 15 lub 16 cyfr

### 2.4.4 Błędy zaokrągleń

- Zaokrąglenia przy zamianie liczb w systemie dziesiętnym na liczby w systemie dwójkowym (i odwrotnie):

- Przykład: $0.1$ nie można przedstawić w systemie dwójkowym w postaci skończonej ilości cyfr:

  $1/16 + 1/32 + 1/256 + 1/512 + 1/4096 + 1/8192 + 1/65536 = 0.0999908$

- Zaokrąglenia przy wykonywaniu obliczeń

- W trakcie długotrwałych rachunków zaokrąglenia mogą sie kumulować, powodując spadek dokładności końcowego wyniku
- W większości sytuacji spotykanych w obliczeniach naukowych wystarcza podwójna precyzja liczb zmiennoprzecinkowych

### 2.4.5 Kodowanie informacji z nadmiarem

- Stosowane w celu wykrycia błędów, powstających np. w czasie transmisji informacji
- Przykład: CRC (*Cyclic Redudancy Check*)
- Przesyłana w postaci strumienia bitów informacja dzielona na ciągi o długości 1024 bity
- Każdy ciąg 1024-bitowy dzielony na 16-cie 64 bitowych podciągów z których każdy można zinterpretować jako jakąś liczbę całkowitą
- Sprawdzamy parzystość każdej z 16-tu liczb: dla parzystych piszemy 1, dla nieparzystych: 0
- Dla każdego ciągu o długości 1024 bity dostajemy w ten sposób 16 bitów (czyli 2 bajty), które dopisujemy na jego koniec; zamiast ciągu 1024 otrzymujemy ciąg o długości 1024+16=1040 bitów (wzrost długości o ok. 1%)
- Po przesłaniu informacji następuje proces odwrotny: strumień danych dzieli się na ciągi o długości 1040 bity, odejmuje 16 bitów parzystości i sprawdza, czy pasują do pozostałych 1024 bitów danych
- Jest to jeden z najprostszych sposobów sprawdzania poprawności transmisji, nie zawsze wykrywa przekłamania; stosowany powszechnie z uwagi na prostotę

### 2.4.6 Reprezentacja kolorów

- Różne częstości światła widzialnego mają różne barwy
- Każdą z nich można uzyskać łącząc 3 kolory: czerwony, zielony i niebieski (system RGB)
- RGB stosowany w telewizji, monitorach komputerowych
- Do druku stosuje się system odbiciowy

- Biały papier odbija wszystkie barwy, czarny pochłania wszystkie
- Barwniki pochłaniają wybrane barwy, odbijają pozostałe
- System CMY: barwniki cyan, magenta, yellow
- Suma C+M+Y teoretycznie daje barwę czarną, w praktyce nieładny, ciemnobrązowy kolor
- Dlatego: CMYK (dodatkowo czarny barwnik)
- Obrazy kodowane w CMYK'u nieładnie wyglądają na monitorze; kodowane w RGB nie są najlepsze do druku
- Barwa światła wpływa na kolory widziane na wydruku

### 2.4.7 Kodowanie grafiki 2D

- Grafika rastrowa
  - Obraz złożony z kropek (pikseli), zwany bitmapą
  - Barwa każdego piksela kodowana na określonej ilości bitów
  - System indeksowany: każdy kolor ma swój numer
  - System RGB i CMY: każdy kolor złożony jest z 3 barw podstawowych, można zmieniać ich intensywność by uzyskać żądany kolor wypadkowy
  - 8 bitów – 256 kolorów, 16 bitów – 65536 kolorów, 24 bity – 16.8 milionów kolorów (tzw. true color)
  - Większa ilość bitów (np. 32) stosowana wtedy, gdy obraz ma podlegać obróbce (np. wydobyciu niewidocznych szczegółów)
  - Przy powiększaniu rozmiarów bitmapy jakość się pogarsza
  - Formaty rastrowe: GIF, PNG, JPEG, TIFF
  - GIF (Graphics Interchange Format), 8 bitów, bezstratna kompresja, system indeksowany (każdemu kolorowi z palety 256 barw przypisano określony numer)
  - Umożliwia animację, kiepsko odwzorowuje barwy
  - Tworzenie GIFów kiedyś wymagało opłat licencyjnych, dlatego stworzono format zastępczy: PNG (Partable Network Graphics, wymawiaj: ping)

- PNG koduje obrazy na 1-49 bitach, bezstratna kompresja
- JPEG (Joint Photographic Experts Group)
- Zaletą jest 24 bitowe kodowanie oraz kompresja z utratą danych (im gorsza jakość, tym mniejszy plik wynikowy; w praktyce stosuje się parametr jakości 75
- JPEG może zapisać kolory w systemie RGB lub CMYK

• Grafika wektorowa
  - Obraz złożony z wektorów (odcinek kodują współrzędne początku, końca i barwa)
  - Okrąg: współrzędne środka, promień i barwa
  - Grafikę wektorową można przeskalowywać (oraz deformować) bez utraty jakości
  - Rysunek w formacie wektorowym zajmuje znacznie mniej miejsca, niż w postaci bitmapy, ale zdjęcia lepiej zapisywać jako bitmapy
  - Programy pracujące z bitmapami często nazywają się malarskimi (np. PaintShopPro), grafiką wektorową — rysunkowymi (np. CorelDraw)

• Grafika 3D — przedstawiana na płaszczyźnie jako rzut 3 wymiarowej sceny; zaawansowane techniki pozwalają na zmianę projekcji, na „poruszanie się" w prezentowanej przestrzeni (np. gry komputerowe, standardem w Internecie jest format VRML *Virtual Reality Markup Language* i X3D)

• Animacja — formaty MPEG, QuickTime, AVI

• Kodowanie filmów: Video CD (MPEG 1), DVD Video (MPEG 2 ), DivX (MPEG 4)

### 2.4.8 Kodowanie dźwięku

• Fala dźwiękowa to sygnał analogowy, komputer przetwarza sygnały cyfrowe; potrzebna jest transformacja

• Karta dźwiękowa określa natężenie dźwięku w danym momencie i zapisuje je w postaci liczby 8 lub 16-to bitowej (jest to tzw. rozdzielczość próbkowania)

• Pomiary te trzeba powtarzać tym częściej, im większe częstotliwości występują w fali (jest to tzw. częstotliwość próbkowania)

- Ucho odbiera dźwięki do ok. 22 kHz; by nie tracić jakości należy stosować częstotliwość próbkowania 44 kHz (kryterium Nyquist'a)
- Przykłady: digitalizacja 10s dźwięku
  - 8 bit, 8 kHz, 78 KB (jakość rozmowy telefonicznej)
  - 16 bit, 44 kHz, 860 KB (jakość CD)
- Początkowo istniały 3 konkurencyjne formaty:

  wav platforma MS Windows

  aiff platforma Apple Macintosh

  au platforma Unix (Solaris)
- Obecnie popularny przenośny format mp3 (MPEG Audio Layer 3); zaleta: kompresja dźwięku
- Przy jakości CD mp3 daje 12-krotną kompresję, przy niższej jakości — jeszcze większą
- mp3 fragmentem standardu MPEG-1 kodowania filmów (stosowany do warstwy dźwiękowej)
- standard MPEG-2 określa lepszy format AAC (MPEG-2 Advanved Audio Coding)
- Ogg Vorbis — następca mp3?
- mp3 objęty prawem autorskim, za jego użycie trzeba płacić (sierpień 2002: opłaty licencyjne obejmują teraz nie tylko kodowanie do mp3, ale również odtwarzanie mp3)
- mp3pro = mp3 + SBR (Spectral Band Replication), znacznie lepszy od mp3, ale kompatybilny z nim (niestety, opłaty licencyjne)
- WMA (Windows Media Audio), bez licencji, ale kontrolowany przez Microsoft (po zdominowaniu rynku MS może wprowadzić opłaty)
- Ogg Vorbis innym standardem, dostępnym za darmo na licencji GPL
- Format MIDI — zapis nutowy, jedynie dla muzyki (a nie mowy, śpiewu, szumów); analogia do grafiki wektorowej
- Zaleta MIDI — wyjątkowo małe rozmiary pliku: kilkanaście minut muzyki to 30-40 kB

- Przykłady kodowania muzyki z płyty CD:
  - z CD na dysk: `cdparanoia 1 piosenka1.wav`
  - do formatu ogg: `oggenc -b 128 piosenka1.wav`
  - do formatu mp3: `lame -b 128 piosenka1.wav piosenka1.mp3`

### 2.4.9 Szyfrowanie informacji

**Po co szyfrować?**

- ochrona prywatności (korespondencja)
- identyfikacja nadawcy (podpis elektroniczny, transakcje finansowe)

**Jak szyfrować?**

- Juliusz Cezar szyfrował swoje listy stosując podstawienie: a-d, b-e, itp (przesunięcie o 3)
- Metoda przesunięcia to stosowany algorytm, a liczba 3 — klucz szyfrujący
- Odszyfrowanie polega na odwróceniu procesu szyfrowania
- Obecnie sekret szyfrowania leży w nieznajomości stosowanego klucza; algorytmy szyfrujące są jawne
- Przykład: algorytm szyfrujący DES ma $10^{17}$ możliwych kluczy szyfrujących (DES – Data Encription Standard)
- Klucz szyfrujący to zestaw kilku liczb – im większe liczby, tym bezpieczniejsze szyfrowanie (ale proces szyfrowania trwa dłużej)

**Szyfrowanie z kluczem publicznym**

- Klasyczne metody: jeden klucz; trzeba go znać, by odszyfrować wiadomość
- Ale jak go przekazać odbiorcy?
- Szyfrowanie z kluczem publicznym wykorzystuje 2 klucze: prywatny i publiczny
- Klucz prywatny znany tylko odbiorcy
- Klucz publiczny, otrzymywany z klucza prywatnego, dostępny każdemu

- Nadawca szyfruje wiadomość kluczem publicznym odbiorcy
- Odbiorca odszyfrowuje ją swoim kluczem prywatnym; nikt inny nie może jej odszyfrować, gdyż nie zna klucza prywatnego

**Podpis cyfrowy**

Odwrócenie sytuacji szyfrowania z kluczem publicznym

- Piszemy e-mail otwartym tekstem
- Szyfrujemy ten tekst swoim kluczem prywatnym, wynik dopisujemy na końcu listu – to jest nasz podpis cyfrowy
- Odbiorca bez problemu odczyta nasz list
- Jeśli chce zweryfikować nadawcę pobiera jego klucz publiczny (np. z jego strony WWW)
- Odszyfrowuje podpis, otrzymuje pierwotną treść listu i sprawdza, czy odpowiada ona treści e-maila

**Jednostronna funkcja znacznikowa (hash function)**

- Im lepsze szyfrowanie, tym dłużej ono trwa (dłuższe klucze szyfrujące)
- Szyfrowanie całego listu w celu jego podpisania jest nieefektywne
- Stosujemy jednostronną funkcję znacznikową, która przetwarza teksty o różnej długości w stałej długości liczbę znacznikową, będącą swoistym „odciskiem palca” całego tekstu
- Odwrotna operacja jest niemożliwa
- Chcąc podpisać list wystarczy zaszyfrować jego znacznik i zamieścić wynik na końcu tekstu – to będzie podpis cyfrowy
- Odbiorca musi wiedzieć, jaką funkcję znacznikową stosowaliśmy; stosując ją obliczy znacznik tekstu listu i porówna z odszyfrowanym podpisem
- Przykład: funkcja znacznikowa MD5 (Message-Digest algorithm 5)
  - Obliczamy znacznik tekstu niniejszego podrozdziału (dotyczącego jednostronnej funkcji znacznikowej)
  - Wynik: 018730606ea6a27ff9d8ef873059b35b

  - Zmieniamy zdanie: „Im lepsze szyfrowanie ...” na zdanie: „Im gorsze szyfrowanie ...”
  - Wynik: 0c74b73b5d8840b39e668f3294646477
  - Oba znaczniki wyraźnie się różnią

- Marzec 2006 – dowolny znacznik md5 może zostać złamany na komputerze PC Intel Pentium 4 (3.2 GHz) średnio w ciągu 17 s

- istnieją inne, bezpieczniejsze funkcje znacznikowe (np. SHA, Secure Hash Algorithm)

- kolejne wersje SHA (SHA-0, SHA-1, SHA-512) używane m.in. w SSL (Secure Socket Layer); SSL używany przy szyfrowaniu operacji bankowych

**Zastosowania podpisu cyfrowego**

- prywatność korespondencji
- transakcje bankowe
- oddawanie głosu w wyborach
- obieg dokumentów w transakcjach handlowych (np. faktury)
- weryfikacja oprogramowania ściąganego z sieci (nieautoryzowane oprogramowanie może zawierać konie trojańskie)

# Rozdział 3

# Budowa i zasada działania komputera

## 3.1 Zarys historii komputerów

### 3.1.1 Tradycyjny podział na 5 generacji

Historia komputerów jest często prezentowana w powiązaniu z rozwojem układów elektronicznych, które decydowały o ich szybkości działania. Generacja pierwsza związana była z lampą elektronową, druga — z tranzystorem, trzecia z układem scalonym, czwarta z mikroprocesorem (układ scalony o wielkiej skali integracji, VLSI), piąta z rozwojem Internetu.

**Generacja zerowa: lata 1936-46**

- budowane w oparciu o przekaźniki elektro-mechaniczne
- komputery przypominają szybkie, programowalne kalkulatory elektryczne
- 1944 MARK I, komputer zbudowany w USA dla obliczania toru pocisków balistycznych, jedna operacja trwa 3-5 sekund, nie można zmieniać raz ustalonej sekwencji obliczeń

**Generacja pierwsza: lata 1940-59**

- budowane w oparciu o lampy elektronowe, można je nazwać elektronicznymi maszynami cyfrowymi
- programowane w języku wewnętrznym (assembler)
- komputery: ABC (1942), ENIAC (1946), UNIVAC (1951), IBM 650 (1954)

- ABC

  Przez długi czas za pierwszą *elektroniczną maszynę cyfrową* uważano komputer ENIAC z 1946 r. Jednak decyzja sądu z 1973 r palmę pierwszeństwa przyznała Johnowi Atanasoff, który razem z Cliffordem Berry znacznie wcześniej, bo w latach 1939-1942, zbudowali pierwszy prawdziwy komputer.

  > „To w czasie wieczoru pełnego Scotch [Whiskey] i jazdy samochodem z prędkością 100 mil/godz pojawił mi się pomysł elektronicznie sterowanej maszyny, która używałaby liczb w systemie dwójkowym (binarnych) zamiast tradycyjnych liczb w systemie dziesiętnym..."

  Atanasoff-Berry Computer (ABC): rozmiar biurka, ponad 300 lamp, ponad kilometr kabli, szybkość obliczeń: jedna operacja na 15 sekund.

- ENIAC: (Electronic Numerical Integrator and Calculator), 18 tys. lamp elektronowych, liczy 1000 razy szybciej niż MARK I (zegar o częstotliwości 60-125 kHz), można go programować jedynie zmieniając połączenia kabli na panelu sterowania

- koncepcja Johna von Neumanna: program i dane w pamięci operacyjnej

- UNIVAC: zegar 2.25 MHz, tylko 5400 lamp elektronowych, operacja dzielenia 6 razy szybciej niż ENIAC, realizuje koncepcje von Neumana – łatwy do programowania, dostępny komercyjnie, wyprodukowano 46 egzemplarzy, zyskuje popularność po przewidzeniu wyniku wyborów prezydenckich w USA w 1952 r.

- IBM 650: dodawanie/odejmowanie 1.65 ms, dzielenie 16.9 ms, pamięć bębnowa 2000 słów (1 słowo to liczba 10-cio cyfrowa lub 2 instrukcje programu), na panelu wiele mrugających światełek (co zwiększało atrakcyjność maszyny), dane wyprowadzane na kartach dziurkowanych (kompatybilne z innymi maszynami IBM), sprzedano ok. 2000 tych maszyn (najpopularniejszy komputer w tych latach)

**Generacja druga: lata 60-te**

- budowane w oparciu o tranzystory

- programowane w językach wysokiego poziomu: Fortranie, Algolu, Cobolu

- przetwarzanie wsadowe

- Komputery: IBM 360 (od 1964), ODRA 1204 (polski, 1967)
- IBM 360: tranzystory, modele od 360/20 (wersja mini, 16KB RAM) do 360/65 i później 360/95 (supercomputer, 1024 KB), systemy operacyjne: DOS/360 (nie PC DOS!), OS/360, architektura oparta o 8-bitowy bajt, 16 bitowe rejestry, 24-bitową przestrzeń adresową; pierwszy komputer ogólnego przeznaczenia – poprzednio każdy nowy typ komputera wymagał dostosowanych do siebie urządzeń zewnętrznych oraz programów, IBM 360 zapoczątkował rodzinę kompatybilnych maszyn, spopularyzował prace na odległość (terminale podłączone przez linie telefoniczne), do 1968 sprzedano 14 tys.
- ODRA 1204: pamięć 16-64 K słów 24 bitowych, wyprodukowano 179 maszyn, 114 wyeksportowano

**Generacja trzecia: lata 70-te**

- Budowane w oparciu o układy scalone
- Praca wielodostępna pod nadzorem systemu operacyjnego (UNIX)
- Języki Pascal, C
- Minikomputery: PDP 11 (wzorowana na nim polska MERA 300)

**Generacja czwarta: lata 80-te**

- Układy scalone o bardzo wielkiej skali integracji (VLSI)
- Mikrokomputery Apple
  - Pierwsze modele już od 1976 (Apple I), ale w jeszcze dość prymitywne
  - Apple Macintosh, procesor Motorola 68000, 8 MHz, RAM 125 KB (lub 512), napęd dyskietek 400 KB, zintegrowany monitor, mysz
  - Pierwszy okienkowy system operacyjny
- Mikrokomputery IBM PC (Personal Computer)
  - IBM PC, procesor Intel 8080, RAM 16kB (rozszerzalne do 256 kB), napęd dyskietek 160kB, monitor
  - Otwarta architektura (cały komputer można złożyć z łatwo dostępnych części jak domek z klocków Lego), ujawniono plany konstrukcyjne, każdy mógł składać te komputery

- Cena: ok. 4000 USD (z uwzględnieniem inflacji)
- System operacyjny dla IBM PC: MS-DOS (Microsoft Disk Operating System); Microsoft kupił system QDOS i rozwinął go w MS-DOS

- Minikomputery VAX (zastępują PDP), pracują pod UNIXem
- Superkomputery Cray

**Generacja piąta: lata 90-te**

- Rozwój Internetu
- Przetwarzanie równoległe
- Programowanie obiektowe, Virtual Reality

### 3.1.2 Cztery paradygmaty historii komputerów

| Dekada | lata 60-te | lata 70-te | lata 80-te | lata 90-te |
|---|---|---|---|---|
| Użytkowanie | wsad | wielodostęp | PC | sieć |
| Użytkownicy | informatycy | specjaliści | poj. osoby | grupy osób |
| Lokalizacja | sala | pokój | biurko | globalna wioska |

### 3.1.3 Trendy pierwszej dekady XXI wieku

**Powrót do terminali**

- Użytkownicy otrzymują znacznie „odchudzone" komputery, pozbawione twardych dysków, z niezbyt szybkimi procesorami, za to z b. szybkim dostępem do sieci
- Programy uruchamiane są na silnych serwerach sieciowych, płacimy nie za sam program, tylko za czas jego wykorzystywania
- Interfejsem może być np. przeglądarka WWW
- Użytkownik nie musi się martwić o wirusy, o instalacje nowych wersji programów itp.
- Przykładem *thin client* jest telefon komórkowy — obecnie obsługujący protokół WAP, pozwalający np. na dostęp do konta bankowego
- Poczta elektroniczna poprzez interfejs WWW

**Mobilne komputery**

- Spada sprzedaż stacjonarnych komputerów PC, wzrasta notebooków
- Notebooki można podłączać do stacjonarnych klawiatur, myszy, monitorów
- W przyszłości stacjonarnymi komputerami będą tylko serwery – komputerami osobistymi będą notebooki

**Przetwarzanie rozproszone**

- Wiele sieci komputerowych jest w pełni wykorzystywane tylko czasami (nocą, w wakacje wiele serwerów ma wolne moce przerobowe)
- *Clusters* — komputery połączone szybką siecią, wspólnie wykonujące zadany program
- Moc obliczeniowa może być dzielona tak, jak energia elektryczna; można ją „przesyłać” szybką siecią w te miejsca globu, gdzie jest potrzebna

**Wolnodostępne oprogramowanie**

- Wzrasta znaczenie programów, udostępnianych razem z kodem źródłowym (zwanych Open Source Software — OSS)
- Licencja GPL (Gnu Public License):
  - programy można sprzedawać i/lub rozdawać za darmo, ale jednocześnie trzeba udostępniać nieodpłatnie kod źródłowy (jeśli ktoś nie chce kupować, może sobie sam program skompilować)
  - wiele firm woli kupować programy GPL, gdyż wtedy uzyskuje wsparcie techniczne (mając źródła same mogą je modyfikować)
  - każdy nowy program, zawierający w sobie fragmenty programów na licencji GPL, też musi być objęty licencją GPL
  - przykładem zestawu programów na licencji GPL jest system operacyjny GNU/Linux
  - wiele programów astronomicznych jest udostępnianych na licencji GPL
- Licencja BSD:
  - to samo, co GPL poza jednym: nie wymaga się, by nowy program, w którym wykorzystano fragment programu BSD, też był objęty licencją BSD

  - przykłady: system graficzny X Window, jedna z odmian Uniksa – FreeBSD

- Szybki rozwój Internetu był możliwy dzięki OSS (programy bind, sendmail, Apache itp.)

- Dla porównania: MS Windows XP EULA

  - EULA – End User Licence Agreement
  - MS Windows XP nie jest sprzedawany, jest licencjonowany (nie posiadasz MS Windows XP na własność)
  - Bez zgody MS nie wolno ujawniać nikomu wyników testów wydajności Win XP (a nuż okaże się, że jest gorszy od poprzedniej wersji Windows. . . )
  - Kto kupił używany komputer z zainstalowanym MS Windows XP, ten nie może dalej odsprzedać komputera bez usunięcia systemu

## 3.2 Budowa i działanie komputera

Komputer składa się z 3 zasadniczych części:

- procesor
- pamięć
- urządzenia WE/WY

połączonych magistralą systemową.

### 3.2.1 Procesor

- Układ arytmetyczno-logiczny

  - wykonuje działania matematyczne: 4 podstawowe oraz wiele dodatkowych funkcji, często potrafi obliczać wartości złożonych funkcji, np. trygonometrycznych
  - wykonuje działania logiczne, np. negacja, porównanie
  - rozkazy sterujące

- Rejestry

  - Do przechowywania liczb

  - Ich długość określa architekturę wewnętrzną procesora
  - Procesory 8-mio bitowe posiadały rejestry o długości 8-miu bitów (np. słynne mikrokomputery Atari, Amiga, Spectrum)
  - Procesory Intela stosowane w komputerach IBM PC XT oraz AT były 16-to bitowe
  - Współczesne procesory Intela klasy Pentium, procesory firmy AMD czy Cyrix występują w wersji 32 bitowej
  - Procesory innych producentów: np. DEC Alpha (64 bity), Sun Microsystems 64 bity; Intel Itanium 64 bitowy
- Szyna danych
- Szyna adresowa
- Zegar — określa szybkość pracy procesora
- Wiele innych komponentów, które pomijamy

### 3.2.2 Pamięć

- Pamięć ROM (*Read Only Memory*)
  jest zawartość jest stała i nie może być zmieniania; w PC pamięcią taką jest BIOS
- Pamięć operacyjna (RAM — *Random Access Memory*)
- Pamięć masowa
  - Dyskietka (popularna 3.5 cala, pojemność 1.44 MB; są też dyskietki 100, 250 MB, np. Iomega ZIP, wymagają specjalnego napędu)
  - Twardy dysk (pojemność typowego dysku 40-400 GB, szybko rośnie)
  - CD ROM (pojemność 650 MB; Compact Disk Read Only Memory; tylko do odczytu)
  - CD-R (Compact Disk Recordable; możliwy jednokrotny zapis w nagrywarce)
  - CD-RW (Compact Disk Rewritable; możliwy wielokrotny zapis)
  - DVD (pojemność 4.7 GB, nowe technologie dają 8.5 GB lub 17 GB w wersji dwustronnej; Digital Versatile Disc)
  - Taśmy magnetyczne (np. standard DAT, pojemności 10–80 GB, wychodzą z użycia)

- HD DVD and Blueray (podczerwony laser z DVD zastąpiono laserem niebieskim)
- Pojemności (jedna/dwie warstwy): HD DVD 15/30 GB, Blueray 25/50 GB
- Pamęci flash (np. w postaci USB Pen, typowe pojemności do 4 GB)

### 3.2.3 Urządzenia WE/WY

- Urządzenia wejścia
  - klawiatura
  - mysz, trackball, touchpad
  - mikrofon
  - kamera
  - skaner
- Urządzenia wyjścia
  - drukarka
  - monitor
  - głośnik

# Rozdział 4

# Systemy operacyjne

System operacyjny to pakiet programów, niezbędnych do funkcjonowania komputera. Rozróżniamy:

- systemy jednozadaniowe (w danej chwili może działać tylko jeden program, np. MS-DOS) i wielozadaniowe (w danej chwili może działać kilka programów, np. MS Windows 9x/Me [oraz NT/2000/XP/2003], Mac OS, Unix, BeOS, VMS)
- systemy jednodostępne (obsługują tylko jednego użytkownika na raz, np. MS-DOS, Windows 9x/Me, BeOS) oraz wielodostępne (wielu użytkowników na raz, np. MS Windows NT/2000/XP/2003, Mac OS X, Unix, VMS)

## 4.1 System operacyjny DOS

- DOS pojawił się na początku lat 80-tych, wraz z serią komputerów IBM PC
- Obecnie dostępnych jest kilka odmian DOS'a: MS DOS (Microsoft), PC DOS (IBM), darmowy DR DOS (Caldera) oraz wolny Free DOS (w trakcie tworzenia)
- DR DOS 7 zapewnia wielozadaniowość
- DOS składa się z 3 zasadniczych części: BIOS'u, jądra systemu i interpretera poleceń

### 4.1.1 BIOS

BIOS to podstawowy system obsługi wejścia i wyjścia (**B**asic **I**nput **O**utput **S**ystem). Jest on zapisany w pamięci stałej ROM komputera i zawiera:

- procedury obsługi klawiatury i monitora (oznaczenie CON), stacji dysków (A:, B:), twardego dysku (C:), drukarki (LPT1), łącza komunikacji szeregowej (COM1)

- procedury POST (**P**ower **O**n **S**elf **T**est); po uruchomieniu komputera testują one mikroprocesor, pamięć operacyjną RAM, sterowniki monitora, klawiatury i stacji dyskietek

- procedury ładowania, które wczytują z dyskietki lub dysku twardego dalsze części systemu operacyjnego

### 4.1.2 Jądro systemu

Zawarte w 2 zbiorach systemowych, znajdujących się na dyskietce systemowej lub w katalogu podstawowym dysku twardego:

- `IO.SYS`
  rozszerza BIOS o sterowniki dodatkowych urządzeń (np. dysku twardego, karty dźwiękowej itp., zainstalowanych w systemie i wskazanych w pliku konfiguracyjnym `CONFIG.SYS`

- `MSDOS.SYS`
  moduł łączący procedury niskiego poziomu, zawarte w BIOS'ie i pliku `IO.SYS` z poleceniami systemu operacyjnego; zarządza on plikami na dyskach twardych i dyskietkach

### 4.1.3 Interpreter poleceń — `COMMAND.COM`

- interpretuje polecenia wprowadzane z klawiatury

- jeśli polecenia nie ma na liście poleceń wewnętrznych systemu DOS, `COMMAND.COM` szuka go w katalogach zdefiniowanych w zmiennej `PATH` — jeśli znajdzie, wczytuje plik do pamięci RAM i uruchamia

- jeśli nie znajdzie, wyświetla komunikat o błędzie:
  `Bad command or file name`

- komendy DOS'a, rozpoznawanych przez `COMMAND.COM`, nazywamy komendami wewnętrznymi — tak jak sam `COMMAND.COM`, rezydują one w pamięci operacyjnej RAM

- pozostałe komendy DOS'a, zwane komendami zewnętrznymi, są osobnymi programami umieszczonymi na dysku systemowym, zwykle w katalogu

`C:\DOS`; po ich wywołaniu przez użytkownika, `COMMAND.COM` wczytuje je do pamięci RAM i uruchamia

- przykłady komend wewnętrznych: `del`, `dir`, `copy` i zewnętrznych: `xcopy`, `tree`, `format`

### 4.1.4 Etapy ładowania DOS'u

- po włączeniu komputera do prądu rozpoczyna prace BIOS
- wykonuje się procedura POST
- procedura ładująca szuka w pierwszym sektorze (tzw. *boot sector*) dyskietki lub twardego dysku program ładujący (*bootstrap*)
- *bootstrap* sprawdza, czy na dysku znajduje się jądro systemu (nie musi to być DOS — w podobny sposób uruchamiają się też inne systemy operacyjne, np. GNU/Linux, Windows 98, OS/2, BeOS)
- jeśli *bootstrap* nie odnajdzie jadra systemu, wyświetla komunikat:
  `Non-system disk or disk error`
  `Replace and strike any key when ready`
- jeśli znajdzie jądro DOS'a, wczytuje najpierw `IO.SYS`, a następnie `MSDOS.SYS`
- system szuka pliku konfiguracyjnego `CONFIG.SYS` i wczytuje zawarte w nim ustawienia
- `MSDOS.SYS` szuka `COMMAND.COM`, wczytuje go do pamięci RAM uruchamia
- `COMMAND.COM` szuka pliku `AUTOEXEC.BAT` i wykonuje zawarte tam polecenia startowe; jeśli go nie znajdzie, żąda podania daty i czasu
- zostaje wyświetlony znak zgłoszenia systemu (tzw. PROMPT), po czym system oczekuje na polecenia

### 4.1.5 Plik konfiguracyjny CONFIG.SYS

- wczytywany przy starcie systemu przez `MSDOS.SYS`
- zawiera zestaw parametrów, definiujących konfigurację oraz wskazujących dodatkowe sterowniki urządzeń zewnętrznych
- przykładowa zawartość:

```
LASTDRIVE=Z \\
FILES=40 \\
DEVICE=C:\DOS\HIMEM.SYS \\
```

### 4.1.6 Zmienne środowiskowe

Zmienne środowiskowe to parametry systemowe lub definiowane przez użytkownika, przechowywane w pamięci operacyjnej , np.:

- `PATH` — zmienna systemowa określająca katalogi, do których zagląda `COMMAND.COM` w poszukiwaniu programów do wykonania (czyli plików o rozszerzeniu `*.COM`, `*.EXE`, `*.BAT`)
- `PROMPT` — określa wygląd znaku zgłoszenia systemu (wyglądającego np. tak: `C:\>`)
- zmienne środowiskowe można definiować wprost z klawiatury komendą `SET`
- te, które chcemy definiować przy każdym starcie systemu, umieszcza sie w pliku `AUTOEXEC.BAT`, który jest wykonywany automatycznie przez `COMMAND.COM` na końcu inicjacji systemu

## 4.2 System operacyjny Unix

### 4.2.1 Model warstwowy systemu

W modelu tym dzieli się system na 4 warstwy: jądro, biblioteki, powłoka i programy.

#### Jądro

- zawiera m.in. program szeregujący oraz sterowniki urządzeń
- zarządza pamięcią operacyjną
- w nowszych systemach sterowniki rzadziej występujących urządzeń dostępne są w postaci zewnętrznych modułów, które można wybiórczo załadować do jądra — dzięki temu jego rozmiary są mniejsze
- jadro ma bezpośredni dostęp do wszystkich zasobów komputera

**Biblioteki**

- zawierają zestawy podprogramów (zwykle napisanych w języku C), wykonujących różne, często stosowane, operacje (mogą to być np. zestawy funkcji matematycznych, procedury, umożliwiające obsługę monitora czy też operacje na dysku)
- biblioteki te są zwykle dołączane do programów na etapie ich konsolidacji (program, np. w języku C, przed uruchomieniem zostaje poddany kompilacji, a następnie konsolidacji)
- taka *statyczna* konsolidacja powoduje, że dana biblioteka jest dołączana do każdego z korzystających z niej programów — nawet wtedy, gdy uruchamiane są one jednocześnie w systemie; powoduje to znaczną zajętość pamięci operacyjnej
- **biblioteki współdzielone** (ang. *shared libraries*) składają się z 2 części:
  - małej, zawierającej jedynie odwołania do biblioteki;
  - zasadniczej biblioteki, umieszczonej w innym miejscu na dysku, niż korzystający z niej program i wczytywanej do pamięci operacyjnej dopiero w chwili startu samego programu
- biblioteki współdzielone umożliwiają wielu programom korzystanie ze znajdujących się w nich procedur, co wydatnie oszczędza pamięć
- inną zaletą bibliotek współdzielonych jest możliwość wymiany starszej wersji biblioteki na nowszą, bez konieczności rekonsolidacji korzystających z niej programów (których może być bardzo dużo)

**Powłoka**

- nazwa pochodzi stąd, że warstwa ta oddziela wewnętrzna część systemu operacyjnego od użytkownika
- powłoka zawiera **interpreter poleceń**, który umożliwia komunikację z użytkownikiem (jest to odpowiednik programu `command.com` z DOS'a)
- interpreter poleceń uruchamia polecenia systemu operacyjnego oraz programy użytkowe

**Programy**

- procesy uruchamiane przez użytkownika
- zarządzane przez program szeregujący jądra
- mogą być przerwane w dowolnym momencie, np. komendą `kill`
- każdy ma przydzielony odpowiedni obszar pamięci i priorytet
- jeśli proces użytkownika próbuje dostać sie do cudzego obszaru pamięci, zostaje przerwany, a system wyświetla komunikat:

  `segmentation fault`

- bieżąca zawartość pamięci procesu może zastać zapisana na dysku w pliku o nazwie `core` (nazwa od słów *core dump*, czyli **zrzut pamięci**)
- analiza zawartości tego pliku może pomóc programiście w wykryciu przyczyny wystąpienia błędu

### 4.2.2 System plików

- system plików zarządza danymi zapisanymi na dysku
- istnieje wiele odmian systemów plików, większość współczesnych ma strukturę hierarchiczną, złożona z katalogów i plików (rysunek)
- katalog, w którym znajduje się użytkownik po zalogowaniu to **katalog domowy** (np. `/home/jasio`)
- DOS nadaje napędom dyskowym i partycjom dysków twardych oznaczenia literowe, np. `A:, B:, C:, D:`
- Unix łączy je w jedna strukturę, w której nie można rozróżnić partycji czy dysków
- nośniki zewnętrzne, np. dyskietki, CD-ROM'y, taśmy magnetyczne, należy przed użyciem zamontować (polecenie `mount`, dostępne często tylko dla administratora systemu)

**Prawa dostępu**

- ponieważ Unix jest systemem wielodostępnym, każdy plik ma określone prawa dostępu
- wyróżnia się 3 rodzaje użytkowników pliku: właściciel (*user*), grupa (*group*), pozostali (*other*)
- są też 3 rodzaje praw dostępu: czytanie (*read*), pisanie (*write*) i wykonywanie (*execute*)
- do pokazania praw dostępu plików służy komenda `ls -l`, a do ich zmiany komendy `chown`, `chgroup`, `chmod`
- podobne prawa nadaje się katalogom

**Dowiązania**

- jeśli plik ma być dostępny w 2 różnych miejscach systemu plików, można go skopiować; to jednak powoduje stratę miejsca na dysku
- lepiej stworzyć dowiązanie: twarde (*hard link*) lub miękkie (*symbolic link*)
- dowiązania twarde można stosować miedzy plikami tylko w ramach jednej partycji na dysku; miękkie między dowolnymi partycjami oraz katalogami
- dowiązania miękkie (częściej stosowane) tworzy się komendą `ln -s`, np.:

  ```
  ln -s ~/teksty/plik.txt ~/teksty/dla_stud/plik.txt
  ```

  gdzie pierwszy argument określa istniejący plik, a drugi argument — dowiązanie

**Urządzenia**

- dyski i urządzenia wejścia//wyjścia odwzorowywane jako pliki w katalogu `/dev`
- lista wybranych plików z katalogu `/dev`:

| | |
|---|---|
| `/dev/console` | konsola systemu |
| `/dev/mouse` | mysz szeregowa |
| `/dev/hda` | pierwszy dysk IDE |
| `/dev/hda1` | pierwsza partycja pierwszego dysku |
| `/dev/hda2` | druga partycja pierwszego dysku |
| `/dev/hdb` | drugi dysk IDE |
| `/dev/hdb1` | pierwsza partycja drugiego dysku IDE |
| `/dev/fd0` | pierwsza dyskietka |
| `/dev/lp0` | pierwszy port drukarki |
| `/dev/null` | urządzenie puste (do testów) |
| `/dev/ttyN` | wirtualny terminal (lokalny) |
| `/dev/ptyN` | pseudoterminal do logowania przez sieć |

### 4.2.3 Powłoki

**Standardowe**

- shell Bourne'a (`sh`); pierwszy interpreter poleceń Unix'a, dziś już przestarzały
- shell Korna (`ksh`); rozszerzenie shella Bourne'a
- shell C (`csh`); shell oparty o składnię języka C, wygodny w pracy interaktywnej

**Rozszerzone**

- Linuks używa rozszerzonych wariantów wymienionych powłok
- stworzono je na licencji GNU, są nieodpłatnie dostepne dla każdej wersji Unix'a stąd stały się b. popularne
- istnieją też wersje dla innych stystemów operacyjnych, np. dla Windows 95/98/NT
- Bourne Again Shell (`bash`) zastępuje shell Bourne'a
- `tcsh` wyparł standardowy shell C (`csh`)

**Praca interakcyjna**

- po zalogowaniu wydajemy polecenia, które są interpretowane przez powłokę
- najważniejsze polecenia:

| Polecenie | Opis | Przykład |
|---|---|---|
| `ls` | wyświetla listę plików i katalogów | `ls` |
| `pwd` | pokazuje nazwę bieżącego katalogu | `pwd` |
| `cd` | zmienia bieżący katalog | `cd katalog` |
| `cp` | kopiuje plik do nowego katalogu | `cp plik katalog` |
| `mv` | przesuwa plik do nowego katalogu | `mv plik katalog` |
| `rm` | usuwa plik | `rm plik` |
| `mkdir` | tworzy nowy katalog | `mkdir nowy_katalog` |
| `rmdir` | usuwa katalog | `rmdir katalog` |
| `more` | wyświetla zawartość pliku na konsoli w częściach | `more plik` |
| `man` | wypisuje na konsoli opis polecenia | `more plik` |
| `passwd` | pozwala zmienić hasło dostepu | `passwd` |
| `exit` | kończy sesje powłoki | `exit` |
| `logout` | kończy sesję pierszej powłoki tzw. *login shell* | `logout` |

**Kanały i potoki**

- podstawowe kanały WE/WY:

| Nazwa | Skrót | Nr | Urządzenie |
|---|---|---|---|
| Standardowe wejście | `stdin` | 0 | konsola (klawiatura) |
| Standardowe wyjście | `stdout` | 1 | konsola (monitor) |
| Standardowy kanał błędów | `stderr` | 2 | konsola (monitor) |

- standardowo powłoka czyta klawiaturę, a wyniki wykonanego programu oraz ew. błędy wykonania wyświetla na konsoli (czyli ekranie monitora)

- można to zmienić używając znaków `>`, `<`, `>>`

- standardowo polecenie `ls` wypisuje na konsolę (monitor) pliki i katalogi znajdujące się bieżącym katalogu; jego wejścia nie można zmieniać (`ls` zawsze czyta zawartość bieżącego katalogu), natomiast jego wyjście można skierować w inne miejsce niż do `stdout`

- przekierowanie wyjścia polecenia `ls` z konsoli (monitora) do pliku:

```
ls > list.txt
```

- chcąc dopisać zawartość do już istniejącego pliku piszemy:

```
ls >> list.txt
```

- polecenie `cat` czyta dane z `stdin` i wypisuje je na `stdout`
- przekierowanie wejścia z konsoli (klawiatury) na plik (program nie będzie czytał klawiatury tylko przyjmie zawartość pliku):

  ```
  cat < pliki.txt
  ```

- przekierowanie zarówno `stdout` jak i `stdin`

  ```
  cat < plik1.txt > plik2.txt
  ```

- przekierowanie kanału błędu z konsoli do pliku:

  ```
  cat list2.txt 2> error.txt
  ```

- przekierowanie kanałów: wyjściowego i błędu do tego samego pliku:

  ```
  ls > wynik 2>&1
  ```

- można też podłączyć wyjście jednego programu do wejścia drugiego, np. polecenie:

  ```
  ls | sort
  ```

  wyświetli nam posortowana listę plików w danym katalogu

- potoki można stosować wielokrotnie; jeśli lista plików z poprzedniego polecenie jest zbyt długa, by się zmieścić na ekranie, można użyć polecenia:

  ```
  ls | sort | more
  ```

- potoki można rozwidlać: z jednego potoku tworzą się wówczas dwa, które można kierować do 2 różnych kanałów wyjściowych:

  ```
  who | sort | tee osoby.txt | more
  ```

  polecenie to spowoduje wypisanie identyfikatorów osób, pracujących w systemie (komenda `who`), wynik posortuje a następnie rozwidli potok i jeden z nich wyśle do pliku `osoby.txt`, a drugi wyświetli na konsoli komendą `more`

- potoki dają powłoce ogromne możliwości, odzwierciedlają też naczelną zasadę Unix'a: dużo prostych programów, które można łączyć w miarę potrzeb

**Rozwijanie nazw plików**

- podobnie jak w DOS'ie, w linii poleceń można stosować tzw. **znaki globalne** (*wildcards*), np.:

```
ls *.txt
cp galaktyka?.fits.* ./obserwacje
cat dane[123].dat >> wszsytkie-dane.dat
```

- najważniejsze znaki globalne:

| Znak | Funkcja |
|---|---|
| `*` | zastępuje dowolna liczbę znaków (także ich brak) |
| `?` | zastępuje dowolny pojedynczy znak |
| `[abc...]` | zastępuje jeden z wymienionych znaków<br>*tych 3 kropek się nie pisze; oznaczają one, że można podać więcej znaków* |
| `[a-z]` | kreska oznacza zakres (tu: litery od a do z) |
| `[!abc...]` | zastępuje wszystkie nie podane znaki |

**Tworzenie aliasów, cytowanie**

- często używane polecenia można zastapić skrótami, stosując polecenia `alias`
- `alias l="ls -l"` definiuje nowe polecenie o nazwie `l`, które wyswietla listę plików w długim formacie
- zastosowane znaki cudzysłowu to tzw. **cytowanie**, które ochrania zawartość przed powłoką (inaczej powłoka by ją zinterpretowała!)
- samo polecenia `alias` wyświetla listę już zdefiniowanych aliasów

**Uruchamianie programów w tle**

- uruchamiając programy można odłączyć ich kanały: wejściowy, wyjściowy i błędu od consoli, co spowoduje wykonywanie w tle; konsola pozostaje wtedy dostępna do prac interakcyjnych
- przykład:

```
cd /; du > ~/wyniki 2> ~/errors &
```

albo:

```
cd /; du > /dev/null 2>&1  &
```

- inny sposób: uruchamiamy program na pierwszym planie:

```
cd /; du > /dev/null 2> ~/errors
```

- mimo, iż potok wyjściowy i błędu zostały przekierowane, terminal pozostaje zablokowany
- zatrzymujemy wykonanie programu: `Ctrl-Z`
- powłoka wypisze np:

```
[5]+  Stopped                 du > /dev/null 2> ~/errors
```

- przenosimy program w tło komendą `bg` (od słowa *background*):

```
bg 5
```

- program rusza ponownie i kontunuuje pracę w tle
- możemy go (bez zatrzymywania) ponownie przenieść na pierwszy plan komendą `fg`:

```
fg 5
```

- do przerywania działania programów (bez możliwości wznowienia ich pracy) służy `Ctrl-C`

**Zmienne lokalne**

- są innym sposobem przekazywania parametrów do programów i powłoki (ich nazwy pisze się tradycyjnie wielkimi literami)
- przyk. zdefiniowania nowej zmiennej lokalnej (proszę zwrócić uwagę na cytowanie):

```
NAZWISKO="Jan Kowalski"
```

- do wyświetlania wartości zmiennej używamy polecenia `echo`:

```
echo $NAZWISKO
```

- tak zdefiniowana zmienna będzie widoczna tylko w tej powłoce, w której ją zdefiniowano

- chcąc powiadomić następne uruchamiane powłoki o naszej zmienne, używamy polecenia `export`:

```
export NAZWISKO
```

**Zmienne środowiska**

- określają parametry pracy powłoki, ustawiane są przez administratora dla wszystkich użytkowników systemu
- niektóre z nich użytkownik może modyfikować
- zmienne użytkownika zawarte są w 2 plikach: `.bash_profile` oraz `.bashrc`
- zmienne z `.bash_profile` są ustawiane zaraz po zalogowaniu sie do systemu, dopiero potem uruchamiana jest powłoka
- zmienne z `.bashrc` są ustawiane poźniej; są one odświeżane przy każdym starcie nowej powłoki użytkownika
- aktualne wartości zmiennych środowiska (oraz zmiennych lokalnych) wypisuje komenda `set`

**Skrypty**

- aby nie powtarzać zestawów poleceń, często wydawanych powłoce, można zapisać je w pliku w postaci skryptu
- przyk.

```
#!\bin\bash
#
# Przykladowy skrypt
#
cd ~
echo "Jestesmy w katalogu domowym"
echo "Oto pliki w nim zawarte:"
ls -l
```

### 4.2.4 Demony

- są to ważne porgramy systemowe, pracujące w tle
- dzięki nim jądro systemu może być mniejsze
- demony pozwalają na zmiany konfiguracji poszczególnych usług (np. drukowania, poczty elektronicznej, serwera WWW) bez zatrzymywania całego systemu
- **lpd** — demon drukowania; w regularnych odstępach czasu sprawdza zawartość katalogu `/var/spool` szukając prac do wydrukowania i kierując je na odpowiednie drukarki
- **cron** — demon czasu (od greckiego boga, Kronosa); w regularnych odstępach czasu sprawdza tablice (o nazwie `crontab`), w których zapisano chwile, w których mają być uruchomione różne programy;
- **syslog** — demon protokołujacy; zbiera raporty wysyłane przez pozostałe demony i zapisuje je w specjalnych plikach, wysyła pocztą elektroniczna do administratora lub wyświetla na konsoli systemu

### 4.2.5 Drukowanie

- jakość obrazu zależu od jego rozdzielczości, określanej jako ilość punktów na cal (dpi, *dot per inch*)
- w środowisku graficznym obraz na monitorze ma zwykle rozdzielczość ok. 75 dpi
- typowe drukarki laserowe zapewniają obecnie 600 dpi
- mimo braku kolorów, łatwiej czytać teksty (zwłaszcza dłuższe) na papierowym wydruku
- najczęściej pod Unixem stosuje się drukarki PostScriptowe
- PostScript to zaawansowany język opisu grafiki na stronie; drukarka PostScritpowa musi posiadać wbudowany interpreter PostScriptu, pozwalający jej stworzyć obraz do wydruku (nie każda drukarka laserowa go posiada)
- PostScript można drukować praktycznie na każdej drukarce (igłowej, atramentowej) korzystając z programu, zamieniającego PostScripot na kody danej drukarki — jednym z takich programów jest GhostScript (dostępny nieodpłatnie na licencji GNU)

- drukowanie pod Unixem odbywa się pod nadzorem demona drukarki, programu `lpd`
- kopiuje on plik, przeznaczony do wydruku, do kolejki spoolera (ang.*System Peripherals Operating On-Line*)
- czekające w kolejce spoolera pliki wysyłane są po kolei na drukarkę
- taki system odciąża komputer od zadań związanych z wydrukiem
- drukowanie pliku PostScriptowego (standardowe rozszerzenie: ps) następuje po wydaniu komendy:
  `lpr plik.ps`
- inne, przydatne komendy:

  `lpq` — wyświetla zadania, oczekujące w kolejce na wydruk
  `lprm` — pozwala usunąc zadanie o podanym numerze z kolejki

- w razie kłopotów z wydrukiem, demon `lpd` wysyła do użytkownika list pocztą elektroniczną
- w praktyce pliki, wysłane do kolejki poleceniam `lpr`, przechodzą przez specjalny program filtrujący, który rozpoznaje ich typ i, jeśli nie są to pliki PostScriptowe, przerabia je na PostScript
- w OA stosujemy filtr, umożliwiający wydruk plików tekstowych (kody ASCII) oraz graficznych (gif, jpeg, tiff, png, fits itp.); poprawne są zatem komendy:

```
lpr list.txt
lpr obrazek.jpeg
```

- wydruk tekstów ASCII z polskimi literami:

```
a2ps -Xlatin2 plik.txt
```

- wydruk polskich tekstów w formacie html z Netscape:

```
ogonkify -CT -N | lpr
```

## 4.3 Cechy Linuksa

### 4.3.1 Wirtualne konsole

- standardowo po zalogowaniu do systemu można utworzyć do 6-ciu wirtualnych konsoli (nazwa *wirtualna* oznacza, że nie są to rzeczywiste monitory)
- przełączanie odbywa się klawiszami `Alt+F1`, `Alt+F2`, ..., `Alt+F6`
- maksymalna liczba konsoli jest parametrem jądra systemu — można ją zmieniać przy jego kompilacji
- w środowisku XWindows klawisz `Alt` ma specjalne znaczenie, dlatego przełączanie na konsolę odbywa się klawiszami `Alt+Ctrl+Fn` (gdzie `Fn` oznacza `F1`, `F2`, ..., `F6`)

### 4.3.2 Linuksowe systemy plików

**Ext2**

- drugi rozszerzony system plików (ext2)
- obsługuje partycje o rozmiarze do 2 GB, nazwy plików o długości do 255 znaków
- zawiera mechanizm zapobiegający znacznej fragmentacji dysku
- rozpoznanie uszkodzonych plików nastepuje już przy starcie systemu
- utracone sektory dysku zapisywane są w katalogu `lost+found` (co oznacza: biuro rzeczy znalezionych)

**Ext3**

- przykład systemu plików z księgowaniem (ang. journaling)
- do systemu ext2 dodaje dokładny zapis zmian na dysku – w razie nagłego wyłączenia systemu umożliwia on odbudowanie integralności systemu plików
- przykład: po awarii zasilania serwer z kilkoma dużymi dyskami z ext2 potrzebował ok. 10 min na odbudowanie spójności systemu plików; po zamianie na ext3 ta sama operacja trwała 3 sekundy

**Inne systemy plików**

- **msdos**: kompatybilny z MSDOS, OS/2, Windows 95/98/NT
- **umsdos**: rozszerzenie systemu msdos o długie nazwy, prawa dosstępu itp.; stosowany w systamie GNU/Linux, instalowanym bezpośrednio na partycji dosowej (np. popularny Monkey Linux)
- **iso9660**: standardowy system plików na CD-ROM'ach
- **nfs**: sieciowy system plików
- systemy plików innych odmian Unix'a

### 4.3.3 Komputery z kilkoma systemami operacyjnymi

- na dysku jednego komputera klasy IBM PC mogą rezydować różne systemy operacyjne (na osobnych partycjach)
- może to być np. kombinacja: MSDOS, Windows 98, OS/2 i GNU/Linux
- wybór systemu następuje po właczeniu komputera
- uruchamia sie wtedy menadżer startowy (*boot manager*), może być nim np. LILO (*Linux Loader*)
- LILO wyświetli znak zachęty `LILO boot:`, po którym należy wpisać nazwę systemu operacyjnego, który chcemy uruchomić
- naciśnięcie tabulatora pokaże nam wszystkie dostępne na danym komputerze systemy

### 4.3.4 Obsługa dyskietki z systemem msdos

- dyskietkę taką można zamontować w Linuxie poleceniem:

```
mount -t msdos /dev/fd0 /mnt/floppy
```

- gdy np. administrator uniemożliwił użytkownikom montowanie dyskietek, można posłużyć się pakietem `mtools`
- udostępnia on zestaw poleceń, takich jak:

  `mdir`, `mcopy`, `mdel`

### 4.3.5 Konwersja tekstu z systemu msdos

- Unix nie różnicuje plików tesktowych i binarnych: w obu koniec wiersza oznacza sie znakiem LF (line feed – wysów wiersza)
- w systemie plików msdos, w plikach tekstowych wiersze kończą się 2 znakami: CR LF (*carriage return* + *line feed*)
- konwersję można uzyskać np. stosując polecenia pakietu mtools z opcją `-t`, np.:

```
mcopy -t plik.txt a:\teksty\list.txt
```

### 4.3.6 Emulatory innych systemów operacyjnych

**Dosemu**

- nie jest epełnym emulatorem — zapewnia jedynie przestrzeń dla uruchomienia systemu DOS w środowisku GNU/Linux'a (na dysku trzeba posiadać kopię DOS'a)
- aktualne wersje pozwalaja uruchamiać bardzo dużo programów dosowych, zarówno na konsoli, jak i w X Window System
- uruchomienie na konsoli: `dos`; powrót do powłoki: `exitemu`
- uruchomienie w X Window: `xdos`; powrót do powłoki: `exitemu`

**Wine**

- Pełny emulator: nie trzeba mieć kopii MS Windows na dysku
- Tłumaczy wywołania funkcji Windows API na odpowiadajace im wywołania funkcji X Window (dzięki czemu np. można używać programów zdalnie poprzez sieć)
- Aktualna wersja nie nadaje się jeszcze do normalnego użycia
- Istnieje komercyjna wersja Wine, CrossOver Office, optymalizowana pod kątem pakietu MS Office i Lotus Notes (cena wersji 1.3: 55 USD)
- CrossOver Office działa bezbłędnie, nie wymaga instalacji MS Windows – wystarczą tylko same programy

**Win4Lin**

- Do działania wymaga instalacji pełnej wersji MS Windows
- Uruchamia MS Windows w środowisku Linuksa; w razie zawieszenia MS Windows można je zabić jak inne procesy
- Nie powoduje spowolnienia działania aplikacji MS Windows
- Cena wersji 4.0: 90 USD

**VMWare**

- Emulacja sprzętu: procesora, płyty głównej itp
- Umożliwia uruchomienie w środowisku GNU/Linux innych syetmów
- Przykład: w Linuxie możemy uruchamić jednocześnie DOS'a, MS Windows 95, MS Windows XP oraz Free BSD
- Emulacja sprzętowa powoduje znaczne spowolnienie działania systemów – potrzebny jest mocny komputer
- VMWare stosowany tam, gdzie pisze się programy jednocześnie na kiilka systemów operacyjnych
- Cena wersji 3.2: 300 USD

# Rozdział 5

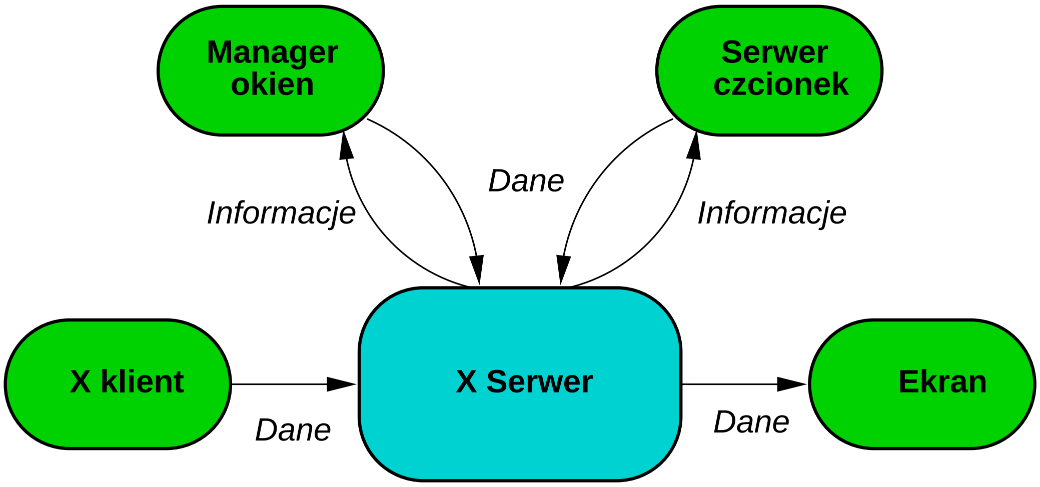
# X Window System

## 5.1 Ogólna charakterystyka X Window System

- Pojawienie się graficznych stacji roboczych w latach 80-tych spowodowało zapotrzebiowanie na graficzne interfejsy użytkownika (GUI — *Graphocal User Interface*)
- W 1986 r. naukowcy z MIT, w ramach projektu Athena, stworzyli jednolite, niezależne od sprzętu środowisko nazwane X Window System
- W 1987 r. pojawia się wersja 11 X Window, stosowana do dziś (kolejne jej modyfikacje oznacza się dodatkowymi numerami, np. obecnie mamy X11R6 (R od słowa: *Release*)
- Oficjalne nazwy systemu (stosowane zamiennie): X Window System, X11, X
- Najpopularniejszą wersją X na systemu uniksowe jest XFree86, rozpowszechnianie na zasadach licencji BSD
- Aktualnie najnowsza wersja to XFree86 4.0

### 5.1.1 Cechy systemu X Window

- **otwartość** — niezależność od producentów, od sprzętu, dostępny nieodpłatnie kod źródłowy; spowodowało to rozpowszechnienie X Window na praktycznie każdą platformę sprzętową (od superkomputerów CRAY, poprzez komputery mainframe w rodzaju IBM4381, Unix'owe stacje robocze, komputery VAX pracujące pod VMS'em aż po PC-ty z MS Windows czy DOS'em)




Rysunek 5.1: Przepływ danych w systemie X Window

- **architektura klient-serwer** — na stacji roboczej czy terminalu graficznym pracuje X serwer, zarządzający sposobem wyświetlania informacji na ekranie; programy użytkowe (klienci) pracują oddzielnie, przesyłając dane do X serwera
- **przezroczystość sieciowa** serwer i klienci porozumiewają się ze soba za pomocą X protokołu, który można przesyłać siecią; dzięki temu klienci mogą pracować na innych maszynach niż X serwer

### 5.1.2 Sposób działania X Window

- po uruchomieniu X serwera, przejmuje on kontrolę nad ekranem, klawiatura i myszą; użytkownik nie może odwołać się do systemu inaczej jak tylko za jego pośrednictwem (rys. 5.1)
- każdy z uruchomionych z X serwera programów jest X klientem — przesyła on dane do X serwera, a ten wyświetla je w odpowiednim oknie na ekranie, korzystając z **managera okien** (*window manager*)
- manager okien to moduł zarządzający oknami: decyduje on o wyglądzie tzw. **widgetów** oraz interakcji okien z klawiaturą i myszką
- widgety to złożone obiekty graficzne: przyciski, ramki, menu wyboru

- przed wyświetleniem tekstu, otrzymanego od X klienta, X serwer wysyła do tzw. **serwera czcionek** rządanie dostarczenia odpowiednich krojów czcionek; serwer czcionek przelicza rozmiar czcionek, dostosowując je do aktualnej rozdzielczości ekranu, po czym przesyła do X serwera odpowiednie mapy bitowe
- korzystając z managera okien, X serwer wyświetla w danym oknie informacje, otrzymane od X klienta

## 5.2 Typy managerów okien

- jedna z głównych zasad filozofii systemu X Window jest modularna budowa: system składa się z wielu łatwo wymienialnych modułów, dzięki czemu można go dostosować do indywidualnych potrzeb
- widać to po mnogosci różnych managerów okien: wiele z nich wyposażonych jest we własne zbiory widgetów
- najpopularniejsze 3 zbiory widgetów:
  - Athena (najstarszy, z pierwszych wersji X Window)
  - OpenLook (opracowany przez firmę Sun Microsystems dla swoich stacji roboczych)
  - Motif (opracowany przez *Open Software Foundation* jako przeciwwaga do OpenLook'a)
- typowe managery okien:

| | |
|---|---|
| twm | *Tab Window Manager* — pierwszy z manager z MIT, stworzony w oparciu o widgety Atheny; spartański w użyciu |
| olvwm | *OpenLook Virtual Window Manager* — produkt Sun'a, oparty o widgety OpenLook |
| mwm | *Motif Window Manager* — produkt OSF, oparty o widgety Motif |
| fvwm2 | *F.. Virtual Window Manager 2* Roberta Nationa, znaczne ulepszenie twm, dostępny na licencji GNU, autor zapomniał już, co oznacza litera F w nazwie |
| fvwm95 | wersja fvwm2, naśladujaca wygląd desktopu MS Windows 95 |
| AfterStep | Wzorowany na window managerze komputerów NeXTStep |
| KDE | Nowość — b. rozbudowany desktop manager (konkurent Gnome) |
| Gnome | Nowość — b. rozbudowany desktop manager (konkurent KDE) |

## 5.3 Praca w systemie X Window

### 5.3.1 Rozpoczynanie i kończenie

- z konsoli X'y na ogół uruchamia się komendą `startx`
- korzysta ona z ustawień pliku `.xinitrc`, znajdującego się w katalogu domowym; .xinitrc podaje, jakich klientów należy uruchomić przy starcie oraz jakiego managera okien użyć
- umieszczona w .xinitrc komenda `xrdb -merge nazwa_pliku` pozwala wczytać dodatkowe parametry pracy managera okien
- w niektórych systemach X Window chodzi bez przerwy, tj. logujemy się do systemu nie na konsoli lecz od razu w środowisku graficznym (sesję taka nadzoruje nie powłoka lecz tzw. *X Display Manager* — xdm)
- uruchamiając X serwer komendą `startx` można podać mu dodatkowe opcje, np. ilość bitów do odwzorowania kolorów, numer ekranu:

  `startx -- -depth 16`
  uruchomi X serwer w trybie 16-to bitowym
  `startx -- :1 -depth 8`
  uruchomi X serwer na drugim ekranie (pierwszy ma numer 0) w trybie 8-mio bitowym

- przed wylogowaniem dobrze jest pozamykać te okna, które otwarliśmy w czasie trwania sesji (niektóre startują samoczynnie po uruchomienia managera okien — te możemy pozostawić)
- jeśli tego nie zrobimy, X'y mogą odmówić zakończenia pracy wypisując komunikat o „stopped jobs"
- wychodzimy z X'ów klikając na właściwy dla danego managera okien przycisk — w wyjątkowym przypadku można przesłać X serwerowi komendę `kill`, naciskając jednocześnie `Ctrl-Alt-Backspace`

### 5.3.2 Okna i X klienci

- widoczna na ekranie przestrzeń ekranu to tzw. pulpit (biurko) — na nim umieszczone są okna X klientów
- istnieją 2 sposoby rozmieszczania nieużywanych w danej chwili okien: minimalizacja do ikony lub rozmieszczanie na kilku wirtualnych pulpitach

- w przypadku braku pamięci operacyjnej, nieaktywne programy zostają przesuniete do przestrzeni swap'u na dysku
- za pomocą myszy można kopiować i wklejać fragmenty okien: kopiowanie do schowka następuje po zaznaczeniu fragmentu okna myszą, przy wciśniętym lewym klawiszu
- wklejenie po naciśnięciu środkowego klawisza myszy (uwaga: myszy komputerów PC nie maja aktywnego 3 klawisza — jego użycie symuluje się naciskając *jednocześnie* oba skrajne klawisze)
- czasem dana aplikacja wykorzystuje kliknięcia myszą do innych celów – wówczas należy wykonać te same operacje przy wciśniętym klawiszu *Shift*
- standardowym X klientem, uruchamianym w każdej sesji X Window, jest xterminal (lub jego odpowiednik, np. rxvt, kvt, ...)
- uruchamianie innych programów możliwe jest poprzez klikanie odpowiednich przycisków na pulpicie lub wpisywaniem nazw programów w oknie xterminala (ten drugi sposób jest ogólniejszy – nie każdy manager okien ma zdefiniowany przycisk uruchamiający potrzebny nam program)
- by nie blokować sobie xterminala, zaleca sie uruchamiać programy w tle, tj. zakończone znakiem `&`
- przykłady:

  `xterm &`
  spowoduje otwarcie nowego okna xterminala
  `xterm -bg linen -fg black &`
  xterminal z kremowym tłem i czarnymi literami
  `mozilla &`
  uruchomi Mozillę

### 5.3.3 X terminal

W poniższych przykładach użyjemy komend `telnet` i `ssh`, służących do zdalnej pracy poprzez sieć. Zostaną one omówione później.

- Komputer z X serwerem można użyć w charakterze X terminala

- X terminal (terminal graficzny) różni się od zwykłego terminala tekstowego (konsoli) tym, że pozwala wyświetlać nie tylko kody ASCII, ale i grafikę
- By rozpocząć sesję z naszego komputera (Vesta) na innej maszynie (Juno) należy:

  1. Umożliwić Juno dostęp do X serwera na Veście:

     ```
     [tkastr@vesta tkastr]$ xhost +juno
     juno being added to access control list
     [tkastr@vesta tkastr]$
     ```

  2. Zalogować się z Vesty na Juno, używając polecenia `telnet` lub `ssh`

     ```
     [vesta ]$ ssh juno
     tkastr@juno's password:
     Last login: Wed Nov 22 19:39:29 from vesta

     [tkastr@juno tkastr]$
     ```

  3. Będąc już na Juno, powiadomić ją, by jej X serwer przekierował całe wyjście graficzne na Vestę:

     ```
     [tkastr@juno tkastr]$ export DISPLAY=vesta:0.0
     ```

  4. Od tej chwili programy graficzne, uruchamiane na Juno, będą otwierały swoje okienka na monitorze Vesty.

- Zaletą tego rozwiązania jest możliwość zdalnej pracy na serwerach, o większej mocy obliczeniowej, niż nasz komputer.
- Możemy też korzystać z zainstalowanego tam komercyjnego oprogramowania, które z uwagi na ograniczenia licencyjne nie może być uruchamiane lokalnie na naszym komputerze.

## 5.4 Przegląd aplikacji X Window

Z konieczności dość wyrywkowy...

### 5.4.1 Przetwarzanie tekstu

Panuje tu zamieszanie, często myli się różne sposoby przetwarzania tekstu, brakuje odpowiednich tłumaczeń terminów angielskich, które lepiej różnicują funkcje programów.

1. Edytory tekstu

   - tekst zapisuje się w postaci kodów ASCII (7-mio lub 8-mio bitowych)
   - możliwość pracy na terminalu znakowym (konsoli) lub w X terminalu
   - wykorzystywane również do pisania kodu źródłowego programów (czyli instrukcji opisujących algorytm, które po odpowiednim przetworzeniu będa wykonane przez procesor)
   - przykłady: `vi`, `emacs`, `joe`, `mcedit` (pracują zarówno na konsoli, jak i X terminalu)
   - graficzne: `xedit` (widgety Atheny, siermiężny), kedit (z KDE), gEdit (z Gnome)

2. Word processors (często zwane po polsku edytorami tekstu)

   - praca wyłącznie w środowisku graficznym
   - poza kodami ASCII program wstawia kody formatujące tekst (np. oznaczenia kroju czcionek, ich wielkości, kody formatujące akapit, stronę itp.)
   - rezultat zwykle widoczny na ekranie w formie WYSIWYG (**W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et)
   - wykorzystywane zwykle w pracy biurowej do pisania krótkich tekstów, nie nadają sie do profesjonalnego składu np. czasopism, książek
   - obecnie wchodzą zwykle w skład Pakietów Biurowych

3. Text processors (procesory tekstu)

   - służą do profesjonalnego składu publikacji
   - w środowisku X Window dostępne są m.in. darmowe pakiety `groff` i $\TeX$; ten drugi jest standardem w naukach ścisłych i informatyce

### 5.4.2 Programy graficzne

1. Do grafiki rastrowej (bitmapowej)
   - najlepszy jest Gimp (licencja GNU), który wg. słów fachowców pobił już klasykę w braży: PhotoShop'a
   - prościutki `xpaint`
   - z pakietu StarOffice:
2. Do grafiki wektorowej
   - klasyczny `xfig` o olbrzymich możliwościach
   - rzadziej spotykany `tgiff`
   - z pakietu StarOffice:
   - z pakietu KOffice: Killustrator
   - przeglądarka PostScriptu: `gv`, `kghostview` (z KDE),
   - przeglądarki PDF (portable Document Format): `xpdf`, `acroread`

### 5.4.3 Zintegrowane pakiety biurowe

- Applixware (komercyjny)
- StarOffice (darmowy, własność firmy Sun Microsystems); używany np. w Sejmie RP, zdobywa popularność bo: dobry, bezpłatny, działa na różnych platformach (od MS Windows, poprzez GNU/Linux'a, na Solarisie kończąc)
- KOffice (wchodzi w skład KDE, obecna wersja ma jeszcze dość ograniczone możliwości)
- każdy z nich zawiera: procesor tekstu, arkusz kalkulacyjny, program grafiki wektorowej, program grafiki rastrowej (bitmapowej),

### 5.4.4 Multimedia

- programy z pakietów KDE i Gnome (do słuchania muzyki, przeglądania filmów)
- osobne programy, np. `mpeg_play`, `xanim`

# Rozdział 6

# Sieci komputerowe

## 6.1 Podstawowe pojęcia

- sieć komputerowa — zbiór wzajemnie połączonych, autonomicznych komputerów (zwanych stacjami roboczymi)
- centralny serwer i połączone z nim terminale (mogą nimi być np. pozbawione dysków twardych komputery PC) to nie sieć, tylko system wielodostępny
- stacje robocze często mogą pełnić jednocześnie funkcje terminali
- rodzaje sieci:
  1. LAN (*Local Area Network*)
     lokalna sieć komputerowa, zawiera do kilkuset stacji, rozmieszczonych na niewielkim obszarze, np. budynku (przyk.: sieć OA UAM)
  2. MAN (*Metropolitan Area Network*)
     miejska sieć komputerowa, obejmuje większy obszar np. osiedla, miasta (w Polsce: PozMan, LodMan, WarMan itp.)
  3. WAN (*Wide Area Network*)
     rozległa sieć komputerowa, obejmuje obszar kraju, kontynentu, cały świat (np. Internet)

## 6.2 Sieci lokalne

### 6.2.1 Korzyści

- umożliwiają połączenie zasobów komputerowych w jeden system, wzajemną komunikację, dzielenie mocy obliczeniowej, centralizację danych i programów
- zasoby komputerowe: dyski twarde, drukarki, plotery, czytniki CD-ROM, napędy taśm magnetycznych (streamery) mogą być wykorzystywane przez wszystkie komputery w sieci
- komunikacja między komputerami np. za pomocą poczty elektronicznej (bardzo wygodne w pracy grupowej)
- dzielenie mocy obliczeniowej komputerów:
  - możliwość uruchomienia programów nie tylko na swojej stacji roboczej, ale również na innych
  - przetwarzanie rozproszone — programy wymagające długotrwałych rachunków można uruchamiać jednocześnie na wielu komputerach sieciowych
  - istnieją programy, które automatycznie dzielą program obliczeniowy na fragmenty i uruchamiają je na innych stacjach roboczych w sieci
  - w ten sposób łatwo z setek gotowych, tanich komputerów PC stworzyć superkomputer
  - przyk.: efekty specjalne do filmu *Titanic* wykonywano na połączonych w sieć komputerach PC z procesorem DEC Alpha, pracujących pod Linuxem

### 6.2.2 Części składowe sieci lokalnych

- komputery
- przewody (koncentryk, skrętka, światłowód) i złącza
- karty sieciowe (obecnie najczęściej typu Ethernet)
- zasoby sieci (drukarki, plotery, streamery, dyski, modemy)
- regeneratory sygnałów (ang. *repeaters*) — wzmacniają sygnały gdy odległości między stacjami są duże

- oprogramowanie sieciowe (np. Novell Netware; Unix jest z natury rzeczy systemem sieciowym i nie wymaga dodatkowego oprogramowania)

### 6.2.3 Techniki transmisji danych

- transmisja analogowa i cyfrowa (wzmacniacze i regeratory sygnału)
- szybkość transmisji cyfrowej: szybkość modulacji (body) i transmisji danych (bity/sekundę)
- transmisja cyfrowa po łączach analogowych (modem)
- zestawianie połączeń w sieci (komutacja łączy i pakietów)

## 6.3 Sieci lokalne typu Ethernet

### 6.3.1 Koncentryk

- typ 10base2, kable koncentryczne, prędkość transmisji 10 MB/s,
- topologia magistrali
- gniazda typu BNC

### 6.3.2 Skrętka

- typ 10base-T, kabel: skrętka (4 skrecone przewody we wspólnej izolacji, 1 para do nadawania, druga do odbioru), prędkość 10 MB/s
- topologia gwiazdy
- stosując tzw. skretkę klasy 5 oraz karty nowsze karty sieciowe, można osiągnąć prędkość transmisji 100 MB/s
- gniazda typu RJ-45, podobne do gniazd telefonicznych (na przewodach sieciowych moga działać telefony, ale nie odwrotnie)

## 6.4 Warstwowe modele sieci

- Architekturę sieci opisuje model warstwowy
- Model sieci TCP/IP (stowrzonej w latach 60-tych dla potrzeb Departamentu Obrony USA):

| | | |
|---|---|---|
| Poziom aplikacji | np. telnet, ftp | NFS |
| Poziom transportowy | TCP | UDP |
| Poziom bramki | IP | |
| Poziom sieciowy | np. Ethernet, PPP | |

- Poziom sieciowy: kable i karty sieciowe Ethernet (lub innych sieci np. IBM Token Ring, Local Talk itp. choć Ethernet jest obecnie najpopularniejszy) lub modemy połączone linią telefoniczną, korzystające z protokołu PPP (**P**oint to **P**oint **P**rotocol)

- Poziom bramki: protokół IP (**I**nternet **P**rotocol), potrafi przesyłać dane w pakietach o zmiennej długości (nie większej niż 65 kB), nie posiada mechanizmów kontroli błędów

- Poziom transportowy: protokół TCP (**T**ransmission **C**ontrol **P**rotocol), w praktyce występuje razem z protokołem IP, uzupełniając go o korekcję błędów, wysyłanie potwierdzeń dotarcia pakietów, porządkowaniem kolejności dochodzących pakietów itp.

- Poziom transportowy: protokół UDP (**U**ser **D**atagram **P**rotocol), umożliwia proste przesyłanie pakietów bez potwierdzenia odbioru

- Poziom aplikacji: różne usługi, dostępne poprzez sieć — telnet, ftp, smtp (poczta elektroniczna), nntp (grupy newsowe), http (World Wide Web) i in.

- Poziom aplikacji: NFS (**N**etwork **F**ile **S**ystem) protokół udostępniania dysków sieciowych, działa w oparciu o UDP czyli bez potwierdzenia odbioru

# Rozdział 7

# Internet

- Internet to sieć typu WAN o zasięgu światowym (może wkrótce zostanie rozszerzona na przestrzeń kosmiczną)
- działa w oparciu o protokół TCP/IP
- jest siecią sieci, gdyż łączy różne sieci lokalne LAN, pozwalając im sie komunikować dzięki standardowi TCP/IP
- sieci lokalne mogą używać innych protokołów na poziomie bramki: zamiast IP może to być np. AppleTalk (protokół komputerów Apple Mcintosh) lub IPX (protokół sieci Novell Netware)
- najpierw była sieć ARPANET (**A**dvanced **R**esearch **P**roject **A**gency **NET**work), wykorzystywana przez amerykańskie uczelnie oraz Departament Obrony USA; ARPANET przekształciła się w Internet

## 7.1 Przestrzeń adresowa

- protokół IP działa w oparciu o 32-bitowe adresy (IPv4), choć w przygotowaniu są adresy 128-bitowe (IPv6), które pozwolą na podłączanie do sieci nie tylko komputerów, ale również np. samochodów, pralek, lodówek, tosterów itp.
- każdy komputer w Internecie ma swój 32-bitowy numer, np.:
  10010110 11111110 1000010 111100
- aby ułatwić jego zapamiętanie, przedstawia się go w postaci 4 liczb dziesiętnych (z których każda odpowiada jednemu bajtowi):
  150.254.66.60

- w celu łatwiejszego kierowania ruchem pakietów w Internecie, część adresu komputera określa sieć (lokalną lub rozległą), reszta — jego numer w tej sieci
- z uwagi na różne wielkości LAN'ół i MAN'ów, objetych przez Internet, wprowadzono 3 klasy sieci, różniące się ilością komputerów:

| Klasa sieci | Adres |
|---|---|
| A | `0XXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX` |
| B | `10XXXXXXXXXXXXXX    XXXXXXXXXXXXXXXX` |
| C | `110XXXXXXXXXXXXXXXXXXXXX    XXXXXXXX` |

- przykład: sieć klasy C w OA UAM
  - adres sieci (network address): 150.254.66.0
    ten adres identyfikuje całą sieć
  - adres rozgłoszeniowy (broadcast address): 150.254.66.255
    pakiet wysłany na ten adres dotrze do wszystkich komputerów w sieci
  - adres sieci i adres rozgłoszeniowy zmniejszają o 2 liczbe adresów, które można przypisać komputerom w danej sieci
- zapisując adresy sieci w postaci 4 liczb dziesiętnych:

| Klasa sieci | Pierwsza liczba numeru | Maks. liczba komputerów w sieci |
|---|---|---|
| A | 1–126 | $2^{24} - 2 \sim$ 16 mln |
| B | 128–191 | $2^{16} - 2 \sim$ 65 tys. |
| C | 192–254 | $2^{8} - 2 \sim$ 254 |

- przykład sieci klasy B:
  - sieć AMUNET (**A**dam **M**ickiewicz **U**niversity **NET**work)
  - adres sieciowy: 150.254.0.0, adres rozgłoszeniowy: 150.254.255.255
- przykład sieci klasy A: firma IBM (NETWORK=9.0.0.0, BROADCAST=9.255.255.255)
- centralną instytucją, zajmującą się rejestracją adresów, jest INTERNIC w USA (Network Information Center)
- `ping` — sprawdza, czy komputer o podanym adresie jest w sieci
- `traceroute` — pokazuje drogę, którą wędruja pakiety w sieci

## 7.2 Adresy domenowe

- w celu łatwiejszego zapamiętania adresów, wprowadzono adresy symboliczne (domenowe), np. astro.amu.edu.pl
- zamiana adresów cyfrowych na symboliczne (i odwrotnie) zajmują sie wyznaczone komputery, będące serwerami nazw (DNS — **D**omain **N**ame **S**erver)
- do zamiany nazw służy komenda `nslookup`, która łączy nas z najbliższym DNS'em, np.: `nslookup 150.254.66.60`

### 7.2.1 Domena poznan.pl

- podmiotem administrującym domeną jest JM Rektor Uniwersytetu im. Adama Mickiewicza
- administratorem jest Ośrodek Informatyki UAM

## 7.3 Usługi

Każdej usłudze odpowiada odpowiedni protokół:

- zdalne logowanie — protokół telnet
- transfer plików — protokół ftp
- poczta elektroniczna — protokół smtp
- grupy dyskusyjne — protokół nntp
- światowa pajęczyna — protokół http