

Uniwersytet Śląski



Wydział Techniki

*Instytut Informatyki Stosowanej*

## PRACA MAGISTERSKA

Temat:

Emulacja drukarki i plotera w trybie graficznym

Wykonali:

Ryszard Czekaj  
Paweł Szymik

Promotor:

prof. dr hab. inż. Dariusz Badura

Konsultant:

mgr Ireneusz Gościński

SOSNOWIEC 1996

## Wstęp

Nowoczesne drukarki, oprócz wydruku prostych tekstów, umożliwiają używanie różnorodnych skalowalnych czcionek, a nawet umieszczanie na stronie obrazów graficznych. Plotery, podobnie jak drukarki, sterowane są specjalnymi sekwencjami, które w wielu wypadkach można traktować jako języki sterujące wydrukiem. Strumień kodów sterujących, wraz z danymi do wydruku, może zostać przesłany do drukarki, lub zapisywany w pliku, na dysku komputera. Taki plik, zawierający dane dla drukarki lub plotera nazywany jest plikiem wydruku. Plik ten może być później przesłany do urządzenia podłączonego do innego stanowiska komputerowego. Plik wydruku zawiera sekwencje sterujące, właściwe tylko dla konkretnego typu urządzenia, przez które będzie później właściwie interpretowany.

Celem tej pracy jest opracowanie programu komputerowego emulującego działanie wybranej drukarki i plotera, który pozwoli na prezentację graficzną pliku wydruku na ekranie komputera.

Przy realizacji tego celu przyjęte zostały następujące założenia:

- istnieje plik wydruku wybranych drukarek i plotera w odpowiednim dla nich formacie
  - znany jest model drukarki lub plotera, dla których plik wydruku został stworzony
- plik wydruku może zawierać proste dane tekstowe, oraz obraz graficzny
- opracowywany program możliwie wiernie będzie prezentował obraz zawarty w pliku
- program umożliwi wydruk danych na drukarce innej, niż ta, dla której został przygotowany plik wydruku
- do budowy programu zaleca się wykorzystać środowisko graficzne Windows 3.xx oraz pakiet narzędzi programistycznych Borland C++.



# Techniczne aspekty pracy drukarek i ploterów

## 1. Klasyfikacja drukarek

### 1.1. Drukarki mozaikowe (drukarki uderzeniowe)

W drukarce mozaikowej obraz tworzony jest za pomocą głowicy zaopatrzonej w serię igieł (zwykle 9, 24 lub 48), które uderzają w taśmę barwiącą nasyconą atramentem, podczas gdy głowica jest przesuwana w przód i w tył przed papierem. Igły ustawione są w jednym lub w kilku pionowych rzędach.

Inna, mniej znana technologia, nazywana „Comb Matrix” lub „Shuttle Matrix” posługuje się poziomym rzędem igieł (optymalnie jedna igła dla każdego znaku poziomego), ponieważ pojedynczy rząd punktów potrafi lepiej odwzorować znak. W przypadkach, gdy nie ma tylu igieł, ile jest znaków poziomych, „grzebień” porusza się nieznacznie do przodu i do tyłu, aby wydrukować niezbędne punkty. Przy tej technice papier może przesuwać się prawie bezustannie i dlatego jest ona często używana w szybkich drukarkach mozaikowych. Technologia ta jest znacznie sprawniejsza, niż drukowanie pojedynczych symboli.

Uderzeniowe drukarki mozaikowe są wszechstronne, tzn. matryca z igłami może być ustawiana do druku różnych czcionek, a nawet grafiki.

Drukarki mozaikowe mogą być używane do drukowania kopii razem z oryginałem, ponieważ obrazy tworzone są poprzez uderzanie w taśmę barwiącą. W drukarkach tych stosuje się papier ciągły, chociaż w większości modeli możliwe jest użycie pojedynczych kartek oraz etykiet.

Ze względu na metodę drukowania, drukarki mozaikowe są znacznie głośniejsze niż inne modele (nawet w tzw. trybie „cichej pracy”).

Prędkość drukowania, podawana zwykle w cps (znaki na sekundę), jest różna, w zależności od używanych czcionek (Draft/LQ) oraz różnych wartości rozdzielczości cpi (punkty na cal) [16], [15], [8].

### 1.2. Drukarki Daisywheel i maszyny do pisania

W drukarkach typu Daisywheel (drukarki rozetkowe) wydruk tworzony jest za pomocą bębna, na którym wyłoczone są wszystkie litery i symbole. Główica przesuwa się

po papierze w przód i w tył, a elektromagnes zaczyna pracować, gdy bęben jest ustawiony w pozycji potrzebnej litery.

Drukarki Daisywheel popularne były ze względu na jakość wydruku lepszą, niż we wcześniejszych modelach, jak np. w dziewięcioigłowych drukarkach mozaikowych.

Główne ich wady to: mała szybkość drukowania, ograniczona ilość symboli na bieżnie, oraz brak możliwości tworzenia grafiki.

Drukarki Daisywheel należą do typu drukarek uderzeniowych. Umożliwia to drukowanie kopii jednocześnie z oryginałem. Drukowanie jest możliwe na papierze ciągłym, jak również na pojedynczych arkuszach oraz etykietach.

Prędkość drukowania podaje się zwykle w cps (znaki na sekundę) [8].

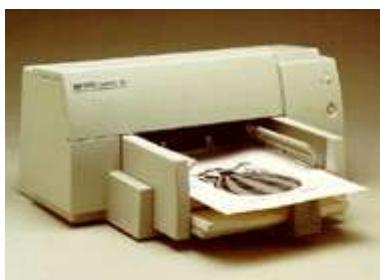
### 1.3. Drukarki atramentowe

Drukarki atramentowe działają podobnie jak mozaikowe, które tworzą wydruk przez układanie punktów na papierze. W drukarkach atramentowych krople atramentu rozpryskiwane są bezpośrednio na papier. Znacznie gęstszy, niż w drukarkach mozaikowych wzór z punktów pozwala uzyskać wydruk lepszej jakości.

W nowszych modelach atramentowych atrament nie jest wyrzucany elektrostatycznie lub poprzez drobne „pompki”, lecz zamieniany w parę przez podgrzewanie i, pod ciśnieniem rozszerzającej się pary, nanoszony jest na kartkę.

Modele atramentowe nie należą do typu drukarek uderzeniowych, więc kopie drukuje się oddziennie. Drukowanie jest możliwe na papierze ciętym, jak w przypadku fotokopiarki. Niektóre modele, dla lepszego wydruku wymagają specjalnej jakości papieru, chociaż dla większości drukarek wystarcza papier standardowy.

W niektórych drukarkach można używać papieru ciągłego i można nawet drukować etykiety, pod warunkiem, że są one wykonane z papieru dobrze wchłaniającego atrament.



rys.A HP DeskJet 600 C

Drukarki atramentowe są cichsze i znacznie szybsze od drukarek uderzeniowych. Parametr szybkości wydruku podaje się w cps (znaki na sekundę), czasami także, dla różnych czcionek w ppm (strony na minutę).

Pomimo, że atramenty tych drukarek są rozpuszczalne w wodzie, różnią się bardzo pod względem „zamaczania”. Może to mieć znaczenie, jeśli wydruk ma być użyty do prezentacji w środowisku o dużej wilgotności.

Niektóre rodzaje atramentu nie są odporne na działanie wody. Do takich atramentów zalicza się: wszystkie Canon'y, czarny DEC 520ic oraz atramenty TI Micromarc. Natomiast czarny atrament HP DeskJet (rys.1) jest prawie tak wodooodporny jak wydruk laserowy. Najbardziej wodooodporne wydają się być czarne i kolorowe atramenty Epson'a: litery „rozpraszają” się, ale nie rozmazują pod wpływem wilgoci.

Wydruki kolorowe (atramentowe) niestety nie są tak długotrwałe jak fotografie. Te z nich, które są wytrzymałe na światło, często niszczy wilgoć, podczas kiedy inne szybko płowieją w jasnym świetle. Trwałość wydruku może być przedłużona przez laminowanie. Niestety, nowoczesne kolorowe drukarki atramentowe nie są w użyciu dostatecznie długo, by zbadać długowieczność najnowszych atramentów [8], [10], [11].

#### **1.4. Drukarki laserowe i drukarki LED**

Modele laserowe i LED nie należą do typu drukarek uderzeniowych, tak więc kopie muszą być drukowane oddziennie. Używane są w nich zwykle pojedyncze arkusze papieru. Można w nich drukować etykiety wytrzymałe na podgrzewanie podczas „wypalania” tonera.

Aby powstał wydruk w drukarce laserowej, specjalny bęben zostaje naładowany elektrostatycznie, a rysunek strony powstaje przez rozładowywianie bębna promieniem lasera lub „linią” LED (ang. Light Emitting Diode) w odpowiednich punktach. Do rozładowanych punktów przylega toner. Punkty te są lustrzany odbiciem miejsc, w których nośnik ma się ukazać na wydruku końcowym.



rys.B HP LaserJet 4V

Papier mija drut Corona, który nadaje mu ładunek elektryczny przeciwny do ładunku punktów na bębnie, następnie przechodzi przez walec, z którego toner jest zatrzymywany na jego powierzchni. Ostatecznie, przesuwany jest pod elementem cieplnym, który, stapiając toner, łączy go z papierem.

Większość obecnych drukarek laserowych nie posiada drutu Corona, a papier elektryzowany jest za pomocą wałka. Drukarki laserowe i LED tworzą bardzo wyraźny, precyzyjny wydruk, o rozdzielczości powyżej 300 dpi (osiągając nawet 1200 dpi). W niektórych drukarkach zastosowano różnorodne techniki, jak np. technologia rozszerzonej rozdzielczości (ang. Resolution Enhancement) w celu kontroli rozlożenia punktów dla wygładzenia konturów.

Drukarki laserowe i LED są stosunkowo ciche. Wydruk modeli laserowych i LED jest mierzony w ppm (strony na minutę) w pozycji „copy mode” (tzn., obraz strony, który jest już w pamięci drukarki, kopowany jest na papier możliwie szybko) [8], [10], [11].

## 1.5. Drukarki kolorowe

W drukarkach kolorowych zastosowano kilka metod służących do tworzenia kolorowego wydruku.

Niezależnie od metody używanej do produkcji kolorowego wydruku, istnieją trzy różne sposoby wyszczególniania kolorów:

- RGB (ang. red-green-blue): metoda standardowa, używana we współpracy z urządzeniami takimi, jak monitory i telewizory. Jest to „dodający” proces mieszania kolorów (kolor biały uzyskuje się przez wymieszanie wszystkich trzech kolorów z tą samą intensywnością).
- CMYK (ang. cyan-magenta-yellow-black): metoda używana szczególnie przy drukowaniu dla celów handlowych. Jest to „odejmujący” proces mieszania kolorów (kolor czarny uzyskuje się przez wymieszanie wszystkich trzech kolorów z tą samą intensywnością lub przez posiadanie dodatkowego koloru czarnego, dającego bogatszą, ciemniejszą czerń).
- Roztrząsanie (ang. dithering): metoda, w której kolory nie są naprawdę mieszane, lecz przez umieszczenie punktów różnych kolorów w różnych „ścieśnionych” miejscach uzyskuje się obraz sprawiający wrażenie użycia więcej niż czterech kolorów. Metoda ta jest podobna do metody otrzymywania skali szarości przez ustawienie więcej lub mniej czarnych punktów.

W kolorowych drukarkach mozaikowych stosuje się kolorową taśmę. W drukarkach atramentowych znajdują się pojemniki z kolorowym atramentem. Niektóre z nich tworzą kolor czarny poprzez mieszanie kolorów (HP DeskJet 600 C rys.1), w innych znajduje się oddzielny pojemnik z czarnym atramentem (HP DeskJet 660 C).

W drukarkach termicznych stosuje się film z czterema kolorami wosku. Drukarka przesuwa papier cztery razy, jeden raz dla każdego koloru. Ponieważ każdy kolor jest przenoszony na papier, powstaje pełny kolorowy obraz.

Niektórzy producenci wprowadzili ostatnio na rynek kolorowe drukarki laserowe, posiadające cztery oddzielne pojemniki na toner, jeden na każdy kolor podstawowy.

W kolorowych drukarkach laserowych stosuje się zwykle cztery kolory tonera: trzy główne „odejmujące” (cyan, magenta i yellow) oraz czarny.

Widzialne spektrum jest ciągłe, można więc tylko przybliżyć kolory widzialne przez wymieszanie barw podstawowych, „odejmujących” (mieszając razem tonery cyan, magenta i yellow uzyskuje się właściwie rodzaj szarości, a nie mocną czerń). Dodanie czarnego tonera pozwala uzyskać więcej kolorów oraz znacznie wyrazistszą czerń. We wszystkich kolorowych drukarkach laserowych zakres kolorów tworzy się przez „mieszanie” tonerów na jeden z dwóch podstawowych sposobów:

1) W drukarkach o tonowaniu ciągłym można zmieniać ilość tonera dla każdego koloru w każdym punkcie. Drukarki te są dość kosztowne, ale umożliwiają jednocześnie reprodukcje o jakości zbliżonej do fotografii. Dla drukarki kolorowej 32-bitowej, o tonowaniu ciągłym każdy piksel może mieć 4,294,967,296 różnych kombinacji tonera. Drukowanie ich wszystkich nie jest konieczne, gdyż np. kombinacja mocnej czerni z C, M i Y wygląda identycznie, a użytkownik ponosi jedynie straty tonera. Ponadto, przejście z 24-bitowej palety RGB na 32-bitową paletę CMYK nie jest zazwyczaj oczywiste, użytkownik ma więc właściwie dostęp do 16,777,216 kolorów.

Nie wszystkie kolory, które mogą być przedstawiane na ekranie w palecie RGB powstają przez zmieszanie czterech tonerów. Niektóre z nich będą poza skalą i można je osiągnąć przez zastąpienie kolorem drukowalnym, postrzeganym podobnie. W ten sam sposób będzie możliwe drukowanie kolorów, których nie da się odtworzyć na ekranie RGB.

2) W drukarkach o tonowaniu stopniowym nie można zmieniać ilości tonera przypadającego na kolor w obrębie każdego piksela. Są one stosunkowo tańsze, ale ich wydruk jest dużo niższej jakości, szczególnie przy odtwarzaniu obrazów z rzeczywistości. Każdy z czterech kolorów może znajdować się, lub nie, w każdym pikselu, tak więc każdy punkt może mieć tylko 16 różnych kombinacji tonera. Ponadto, kolor czarny, zmieszany z jakimkolwiek innym, będzie wyglądać jak czarny, więc 8 z tych kombinacji będzie wyglądać tak samo, i każdy piksel może w rzeczywistości przybrać 9 różnych kolorów. Kolory, które nie mogą być reprezentowane bezpośrednio, są pozorowane przez roztrząsanie (ang. dithering), co powoduje, że oko ludzkie odbiera kolory pośrednie w taki sam sposób, jak półtonowe fotografie w gazetach. Drukowanie nie jest możliwe w 16,777,216 kolorach prezentowanych przez 24-bitowe RGB. Można drukować tylko 9 różnych kolorów, a więc konieczne jest używanie procesu roztrząsanego dla pokazania 24-bitowych kolorów. Roztrząsanie pozwala na przedstawienie kolorów pośrednich, w sposób porównywalny do tonu ciągłego. Do drukarek o tonowaniu stopniowym można dostarczać 24-bitowe dane, co nie wpływa na możliwość mieszania różnych ilości każdego tonera dla równego zakresu kolorów każdego punktu [8].

## 1.6. Inne rodzaje drukarek

Istnieją inne technologie druku np. technologia drukarek liniowych, wykorzystywana szczególnie w systemach mainframe. Drukowanie następuje za pomocą obracającego się cylindra lub łańcucha zawierającego wszystkie symbole.

W przypadku zastosowania cylindra rotacyjnego wszystkie kolumny zawierają cały zestaw znaków, cylinder obraca się w kierunku przesuwania się papieru, a papier jest zatrzaszkowany między taśmą i cylindrem, kiedy symbol do wydrukowania znajduje się we właściwej pozycji. Jeden obrót cylindra drukuje całą linijkę tekstu.

W przypadku zastosowania łańcucha rotacyjnego zestaw symboli jest powtarzony na łańcuchu kilka razy. Projektant drukarki ustala ilość powtórzeń każdego symbolu, w zależności od tego jak często pojawia się on w tekście. W amerykańskim angielskim jest to sekwencja ETAON.... Jeśli używane symbole powtarzane są częściej, wtenczas drukowanie całości jest szybsze. Łańcuch obraca się prostopadle do kierunku przesuwania się papieru, a papier jest zatrzaszkowany między taśmą i łańcuchem, kiedy symbol do wydrukowania znajduje się we właściwej pozycji. Wariantem drukarki łańcuchowej jest drukarka typu „Belt Printer”. Jakość wydruku jest zbliżona do jakości wydruku drukarki łańcuchowej, przy jednocześnie niższym poborze mocy i mniejszym natężeniu hałasu. W standardowej drukarce „belt” używa się wysokiej jakości zestawu w łańcuchu. W drukarkach IBM 3203 i 1403 każde „uderzenie” powoduje wydrukowanie trzech symboli z łańcucha. Użycie łańcucha wyrazów jest celowe: trzy symbole na moduł podczas obrotu łańcucha następują po sobie.



rys.C IBM 3203

W drukarce „belt” symbole ustawione są osobno na „palcach” (ustawienie podobne do symboli na „daisywheel”) umieszczonych na gumowym pasie. Podobnie jak w drukarce łańcuchowej, nie wszystkie symbole są powtarzone z tą samą częstotliwością. W odróżnieniu od drukarek cylindrowych i łańcuchowych, pas jest zatrzaszkowany pomiędzy taśmą i papierem.

Prędkość drukowania dla tych drukarek podawana jest zwykle w lps (linie na sekundę) lub nawet w pps (strony na sekundę). Drukarki „belt”, drukarki łańcuchowe oraz drukarki z cylindrem rotacyjnym (do 70 lps) polecane są dla zwiększenia szybkości wydruku.

Istnieją również drukarki mozaikowe termiczne, umożliwiające drukowanie symboli i grafiki (np. barcodes) przez podgrzewanie papieru (punktów): miejsca podgrzane stają się

czarne.

W dziedzinach związanych z tworzeniem wydruków wykorzystuje się jeszcze kilka innych „egzotycznych” technologii np. papier pokryty aluminium stosowany przez Sinclair’a, a także technologia „rozładowanie iskry” (ang. spark discharge). Są one jednak bardzo rzadkie i w większości przestarzałe [8].

## 2. Klasyfikacja ploterów

### 2.1. Plotery tablicowe

Plotery tablicowe są urządzeniami o dość pokaźnych gabarytach, zbliżonych do maksymalnego formatu rysunku. Na przykład, ploter formatu A1 lub A0 miał wygląd ogromnego stołu, na którym arkusze kalki mocowano elektrostatycznie. Po tablicy poruszały się pisaki na stalowych szynach, a wykreślenie najdrobniejszego nawet elementu trwało bardzo długo, ponieważ pisaki musiały całą drogę tam i z powrotem przebyć po szynach [11].

### 2.2. Plotery pisakowe (bębnowe)

Znacznie doskonalsze są plotery pisakowe (rys. 4). Urządzenia te zajmują o wiele mniej miejsca, a arkusz kalki mocowany jest już tylko w dwu punktach.



rys.D Ploter pisakowy

Zasadę działania można najprościej przedstawić w postaci ramy z rozpiętą kalką poruszającą się po rolkach w płaszczyźnie pionowej, podczas gdy karetka z pisakami porusza się w płaszczyźnie poziomej. Wymiary gabarytowe plotera pisakowego musiały być dostosowane wyłącznie do szerokości formatu. Były to urządzenia znacznie mniejsze, cichsze i szybsze.

Szybkość pracy była stale zwiększana dzięki optymalizacji programowej ruchu karetki z pisakiem.

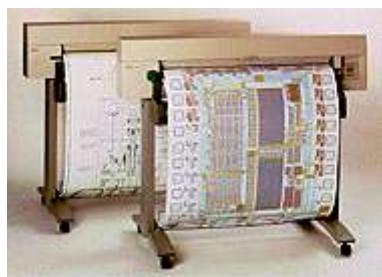
W ploterach pisakowych nie udało się przezwyciężyć następujących wad: możliwości kolorystyczne ograniczały się do wymiany pisaków, a wypełnianie pól sprowadzało się do ich zakreskowania. Rysowanie bitmap, było możliwe tylko teoretycznie. Jakość rysunku zależała od szeregu czynników, jak np. jakość

tuszu i pisaka oraz szybkość przesuwu pisaka. W przypadku skomplikowanych, wielowarstwowych rysunków większych formatów występowali niedokładności rzędu milimetrów, spowodowane kumulowaniem się mechanicznych odchyleń przesuwu pisaka.

Rozwój ploterów pisakowych (laboratoria firmy Houston Instruments) doprowadził z czasem do powstania plotera z głowicą skanującą. Zamontowanie takiej głowicy przekształcało ploter w wielkoformatowe urządzenie skanujące [11].

### 2.3. Plotery atramentowe

Równolegle z ploterami pisakowymi rozwijały się rastrowe plotery atramentowe. Urządzenie to działa na zasadzie rozpylania na kalce kropelek tuszu, podobnie jak w drukarce atramentowej. Plotery atramentowe są znacznie dokładniejsze i łatwiejsze obsłużyć, zaś efekty ich pracy są bardziej przejrzyste.



rys.E DesignJet 350 C

Obecnie wiele firm wprowadza na rynek wielkoformatowe plotery rastrowe. Najnowszym produktem tego rynku jest produkt firmy Hewlett-Packard DesignJet 750C. Jest to ploter formatu A0, kolorowy, dysponujący niezwykle wysoką rozdzielczością true 600 dpi w systemie monochromatycznym. Jak większość nowoczesnych wyrobów Hewlett-

Packarda, jest to urządzenie całkowicie bezobsługowe. Po załadowaniu rolki papieru i włożeniu na swoje miejsce naboju atramentowych ploter można włączyć. DesignJet (rys.5) już samodzielnie „troszczy się”, by wszystkie dysze prawidłowo działały, by papier ustawiony był właściwie, sam sprawdza parametry nośnika i dostosowuje się do podanego formatu.

Oryginalnym wkładem Hewlett-Packarda w rozwój koncepcji plotera rastrowego jest zastosowanie tzw. tuszu pigmentowego. Do tej pory głównym argumentem przeciwko ploterom atramentowym, było zjawisko zasychania dysz. W HP zadbane, by produkt sam „sprawdzał”, czy jakaś dysza się nie zatkała, i sam próbował awarię usunąć. Dopiero zastosowanie tuszu pigmentowego przyniosło radykalne efekty i przyczyniło się do podniesienia jakości wydruku. Tusz ten, w odróżnieniu od innych stanowi zawiesinę ciała stałego. Naboju z takim tuszem nie można już uzupełniać tzw. refilami.

Efektem zastosowania tuszu pigmentowego jest zwiększenie wyrazistości wykreślonej linii bądź powierzchni. Napylona na papier warstewka substancji zabezpieczającej przed wilgocią i podnosi trwałość oraz odporność rysunku.

Drobnoziarnista struktura tuszu umożliwia uzyskanie rozdzielczości rzędu 600 dpi.

Innym ciekawym rozwiązaniem zastosowanym w modelu 750C jest kompensacja luzu na zębatkach toru karetki. Usuwa ono efekt poszarpanych linii pionowych, który w przypadku niektórych innych urządzeń, zwłaszcza przy rysowaniu cienkimi liniami dużych elementów tworzył mało estetyczny efekt wizualny.

DesignJet 750C jest urządzeniem niezwykle cichym i szybkim. Choć przy prostych rysunkach z zasady nie może konkurować z ploterami pisakowymi, w przypadku skomplikowanych, z czasem rysowania formatu A1 rzędu 4 minut, przewyższa je.

Dużym udogodnieniem jest możliwość wyposażenia plotera w tzw. kartę HP JetDirect. Jest to karta sieciowa, dzięki której ploter widziany jest w sieci jako urządzenie dostępne dla wszystkich pracowników i nie wymaga ustawienia dodatkowego serwera. 8 MB RAM, w które standardowo wyposażony jest DesignJet, pozwala na kreślenie bardzo dużych rysunków (nawet o wielkości kilkunastu megabajtów).

Sterowniki rozwiązyano w taki sposób, by drukowanie odbywało się w trybie ciągłym w pamięci komputera, równolegle z przetwarzaniem wektorów na bitmapę. Dodatkowa pamięć potrzebna jest dopiero wówczas, gdy rozbudowujemy system o funkcje postscriptowe [11].

### 3. Języki sterowania drukarek i ploterów

Nowoczesne drukarki mogą tworzyć bardzo złożone rysunki, niekiedy konkurujące z fotografiemi lub z dokumentami otrzymywany mi z drukarni. Sposób przesyłania strumienia danych od drukarki jest sam w sobie formatem zapisu obrazu w pliku. Istnieją dwa ogólne typy formatów plików wydruku dla drukarek: rozszerzone formaty tekstowe (ang. extended text format) i języki opisu strony (ang. page description languages).

#### 3.1. Rozszerzone formaty tekstowe

Formaty takie włączają informacje graficzne do konwencjonalnego strumienia danych tekstowych. Zwykły tekst jest drukowany, a sekwencje sterujące wprowadzają elementy nietekstowe. Szeroko rozpowszechnionym formatem jest Printer Control Language (PCL). Stał się on standardem dla drukarek laserowych niskiej i średniej wydajności [6], [15], [16], [8].

### 3.1.1. IBM Proprinter

Język drukarki IBM ProPrinter początkowo używany do drukarek IBM (IBM Graphics Printer 5152, IBM ProPrinter XL 4201/4202, IBM ProPrinter X24/XL24 4207/4208), jest też używany obecnie przez wiele drukarek mozaikowych jak w przypadku Epson'a ESC/P i ESC/P2. Funkcjonalność jest różna w zależności od zastosowanej wersji lub wyselekcyjowanej podczas ustawienia drukarki (XL, XL24 lub AGM są przykładami takich wersji).

Język sterowania IBM ProPrinter w drukarkach od różnych producentów może się nieznacznie różnić co do funkcjonalności. Zwykle różnice są dodatkowymi funkcjami nieuwzględnionymi w standardzie.

### 3.1.2. Epson ESC/P, ESC/P2

ESC/P (Epson Standard Code).

ESC/P2 (Epson Standard Code, Level 2).

Język drukarki ESC/P został stworzony przez Epson'a do stosowania we wczesnych drukarkach mozaikowych. Obecnie używany jest także przez atramentowe i laserowe drukarki Epson'a jak również przez wiele innych drukarek mozaikowych znajdujących się na rynku. ESC/P2 jest ulepszeniem ESC/P, posiada np. nowe funkcje do skalowania czcionek, dla drukowania grafiki rastrowej, itp.

Drukarki ESC/P lub ESC/P2 różnych producentów mogą się nieznacznie różnić w swojej funkcjonalności. Zwykle różnice te, to dodatkowe funkcje niedostarczone z wersją oryginalną Epson'a.

Informacje o stosowanej wersji ESC/P i ESC/P2 znajdują się w dokumentacji drukarek poszczególnych producentów. Niektóre pozycje to krótkie wyliczenie dostępnych funkcji z przykładami w BASIC'u. Można też znaleźć tam tablice z zestawem symboli i tablice z szerokością czcionek. „ESC/P2 Reference Manual” firmy Epson, zawiera wykaz kodów ESC/P i ESC/P2, jak również kompletny opis różnic w rozkazach każdej drukarki. Najnowsza wersja pochodzi z sierpnia 1992.

## 3.2. Języki opisu strony

Innym podejściem do sterowania drukarką jest zdefiniowanie całkowicie nowego języka opisu wydruku. W przeszłości używano kilku takich języków, ale obecnie standardem stał się PostScript [8], [6].

### 3.2.1. PostScript

PostScript jest językiem „opisu strony” produkowanym przez Adobe Systems Inc. od początku lat 80-tych. Adobe zostało utworzone w 1982 r przez dr John E. Warnock i dr Charles M. Geschke. Dostarcza on instrukcji do opisu strony informacji. Ponieważ wymaga więcej pamięci, niż większość języków „opisu strony”, był pierwszym, szeroko dostępnym produktem, który kontrolował większą liczbę czcionek i grafiki.

Pierwsza wersja opublikowana w 1985 r jest nazywana Level I, obecne ulepszenie jest nazywane Level II (nie należy mylić z: wersją PostScript 47.0 lub 2011.110, ani z liczbą w linii początkowej jakiegokolwiek wydruku PostScript, jak „%!PS-Adobe-3.0”). Poziom PostScript’u i rozszerzenie wersji o interpreter, poszerzają możliwe operacje.

W obiegu znajduje się kilka klonów PostScript’u, lecz, ze względu na koszty opłaty licencyjnej za Adobe interpreter; najbardziej znanym jest GhostScript. Inne, wbudowane bezpośrednio do drukarek laserowych lub dostosowywane przez dodatkową kasetę (ang. cartridge) to: Phoenix Page, BrotherScript, Page Styler, True Image, Turbo PS, PDL i KPDL. Mówią się o nich, że są całkowicie kompatybilne z PostScript’em, lecz ta kompatybilność kończy się czasami przy ładowaniu czcionki, przy manipulowaniu czcionką (tj. wstawianiu tabeli metrycznej lub nowych symboli) oraz przy innych operacjach. Przy użyciu klonów nie ma jednak problemu z drukowaniem prostego tekstu czy grafiki.

Zestaw symboli PostScript jest stosowany do pisania programu PostScript, a nie symboli drukowanych czcionką PostScript’u.

Adobe zaleca użycie tylko drukownego podzestawu symboli ASCII w programie PostScript’u, spacje, tabulatory i symbole CR oraz LF. PostScript nie zabrania użycia symboli poza tym zestawem, lecz stosowanie ich może spowodować problemy z przesyaniem (np. transfer 8-mio bitowych symboli przez 7-mio bitową linię seryjną do drukarki). Dla przedstawienia 8-mio bitowych symboli poza ciągiem należy użyć formuły „\ddd”.

Plik przeznaczony dla drukarki jest właściwie programem komputerowym w języku PostScript, który rysuje żądany obraz. PostScript używany jest do wykonywania operacji ścisłe związanych z generacją obrazu.

Programy PostScriptowe rysują grafikę zasadniczo na dwa sposoby. Łatwiejszy sposób polega na narysowaniu mapy pikseli. Jeden z podstawowych operatorów PostScriptu wczytuje sekwencję pikseli i wyświetla je w prostokątnym obszarze na stronie. To podejście nadaje się do dołączania do dokumentów obrazów zeskanowanych lub pobranych z obrazu monitora, lub dowolnych innych obrazów, które już istnieją w formacie mapy bitowej.

Alternatywnym sposobem użycia PostScirptu jest tworzenie rysunku w formacie wektorowym lub metapliku. Istnieją operatory do rysowania linii, okręgów, krzywych, prostokątów itp., i z tych elementów można budować grafikę.

Można również łączyć te dwa podejścia. Dla przykładu: operatory linii, prostokąta i krzywej mogą posłużyć do wyznaczenia obszaru, który zostanie następnie użyty jako maska wycinania do sterowania wyświetleniem mapy bitowej [8], [11].

### 3.2.2. HP PCL i PJL

Język PCL został stworzony przez firmę Hewlett-Packard na potrzeby produkowanych przez siebie drukarek (laserowych i atramentowych). Wersje języka PCL są numerowane od 1, aż do obecnej wersji 5e.

Krótką historię języka PCL (na podstawie „HP's Printer Language Technical Reference Manual”):

- PCL 1 - Funkcjonalność druku i przestrzeni jest bazą dla funkcji dostarczonych dla prostego, wygodnego stanowiska pracy pojedynczego użytkownika.
- PCL 2 - Funkcjonalność EDP (ang. Electronic Data Processing - elektroniczne przetwarzanie danych), jest ulepszeniem PCL 1. Zostały dodane funkcje dla celów ogólnych, system drukowania dla wielu użytkowników.
- PCL 3 - Funkcjonalność „Office Word Processing” jest ulepszeniem PCL 2. Zostały też dodane funkcje dla podwyższenia jakości wydruku, obróbka dokumentów biurowych (drukarki z rodziny HP DeskJet).
- PCL 4 - Formatowanie strony jest ulepszeniem PCL 3. Zostały dodane funkcje nowych możliwości drukowania strony (drukarki: HP LaserJet, HP LaserJet IIP (PCL 4,5))
- PCL 5 - Funkcjonalność „Office Publishing” jest ulepszeniem PCL 4. Nowe możliwości publikowania włączają skalowanie czcionki i grafikę HP-GL/2 (Drukarki HP LaserJet III, HP LaserJet 4 (PCL 5e)).

Wersje PCL różnią się pod względem funkcjonalności (np. typ czcionek, czcionki mapy bitowej, czcionki skalowalne (Intellifonts, True Type), metody kompresji grafiki, wsparcie grafiki przez HP LaserJet III).

PCL jest najbardziej rozpowszechnionym językiem drukarek na obecnym rynku drukarek laserowych. Większość producentów drukarek laserowych stosuje do nich języki PCL 4 lub PCL 5.

PJL (ang. Printer Job Language) został także stworzony przez HP, aby dostarczyć metody na zmianę parametrów poziomu pracy i stanu odczytu pomiędzy drukarką, a głównym komputerem. PJL może być stosowany na początku drukowania do ustawienia niektórych parametrów jak język drukarki (PCL, PostScript lub inne), rozdzielcość (300 lub 600 dpi), ilość kopii, itp.

PJL jest obecnie stosowane w następujących drukarkach HP: LaserJet IIISi, rodzina LaserJet 4, PrintJet XL 300 oraz DesignJet.

PJL jest również stosowane w serii „5” drukarek LaserJet [11], [7], [10].

### 3.3. Inne języki sterowania drukarek

Na rynku znajduje się wiele innych unikalnych języków sterowania drukarkami. Następująca lista nie jest więc kompletna (intencją jest tylko wspomnienie o nich, a nie dokładny opis). Kolejność wymienionych języków nie ma nic wspólnego z ich ważnością na rynku.

- Advanced Function Printing (AFP): jest używany w IBM Mainframes dla drukarek „stronicowych”. Jest to funkcja prezentacji zestawu Mixed Object Document Content Architecture (MO:DCA), który jest częścią IBM System Application Architecture.
- Właściwie nie drukuje się za pomocą MO:DCA, stosuje się IPDS (ang. Intelligent Printer Data Stream).
- Informacje obejmują PTOCA (ang. Print Text Object Content Architecture), GOCA (ang. Graphic Object Content Architecture), IOCA (ang. Image Object Content Architecture) jak też inne.
- IPDS jest językiem druku IBM SAA. Działa on z różnymi czcionkami bitmapowymi, z prostą podstawową grafiką, oraz z wizerunkami bitmapowymi. Ze względu na prostotę „wyobrażanego” modelu, może być użyty do kierowania-obsługi szybkich drukarek laserowych.
- Diablo 630: początkowo używany z drukarkami rozetkowymi i maszynami do pisania. Toleruje tylko sekwencje tabulacji, odstępy linii i symboli, selekcję atrybutów (ang. bold, double-strike, underline - pogrubiony, podwójne uderzenie, podkreślony), ruchy poziome w obu kierunkach, proporcjonalne odstępy i automatyczne wyśrodkowanie i justyfikację - między innymi. Ten język jest czasami używany przez innych producentów jako baza dla ich specyficznych emulacji.
- CaPSL: (ang. Canon Printing System Language) był poprzednim językiem standardowym dla laserowych drukarek Canon'a. Inna występująca nazwa to LIPS

(ang. Laser-beam Image System). Za CaPSL kryje się pewna historia - Canon produkuje silniki do drukarek dla HP, nie miał jednak licencji na stosowanie HP PCL w swoich własnych drukarkach. Stąd powstała potrzeba znalezienia własnego języka drukarek. Lasery Canona tradycyjnie zawierały CaPSL, IBM ProPrinter, ESC/P i emulacje PostScript, ale nie PCL). Ta część kontraktu między Canon'em a HP najwidoczniej uległa przedawnieniu, ponieważ obecnie Canon oferuje drukarki z PCL 4 i PCL 5.

- LIPS wspomaga Diablo 630 (ustawienie fabryczne dla trybu polecenia - ang. command mode), tryb ISO (dla drukowania tekstu i grafiki rastrowej) oraz tryb VDM (dla grafiki wektorowej i drukowania symboli)..
- RENO: jest to standardowy język sterowania dla drukarek Agfa (P400, P3400, itp.). RENO jest rodzajem języka opisu strony. Jego funkcjonalność jest ogromna: oprócz drukowania tekstu z różną skalą czcionek, można rysować linie, zapełniać ikony (Windows) wzorami, używać „wyrażeń” programowania (if-then-else, repeat-until, set, use and print variables, operacje push i pop), może ładować i drukować swoje własne symbole i przenosić dane do pamięci operacyjnej drukarki lub na twardy dysk, czy dyskietkę jeśli jest dołączona.
- Prescribe: jest to język opisu strony stworzony przez Kyocera. Jego zaletą jest fakt, że można go osadzić w innej obecnej emulacji drukarki na urządzeniach Kyocera. Drukarki Kyocera wspomagają HP PCL, klon HP-GL nazywany KC-GL, Epson ESC/P (tryb LQ-850), IBM ProPrinter X24E, Diablo 630, ogólną emulację drukarki liniowej, i jako opcję KPDL, klon PostScript.
- DEC: DEC posiada swoje własne unikalne języki dla drukarek laserowych (LN03, LN06).
- ANSI: Dan McGowan z Mannesmann Tally twierdzi, że: "Drukarki Mannesmann Tally, które posiadają ANSI dały podstawę dla notacji ANSI 3.64. Jest to raczej luźna specyfikacja pokrywająca ogólne funkcje peryferyjne. Drukarki produkowane w U.S.A., w większości posiadają wszystkie rozkazy ANSI 3.64 należące do funkcji drukarki. Drukarki produkowane w Niemczech są serią drukarek głowicowych (ang. flying serial head printers). Posiadają MTPL (ang. Mannesmann Tally Printer Language), który jest oparty na ANSI 3.64 lecz zawiera dodatkowe polecenia" [8].

### 3.4. Języki sterowania ploterów

Nazwa HP-GL (ang. Hewlett-Packard Graphics Languages - język graficzny firmy Hewlett-Packard) oznacza język stosowany pierwotnie do obsługi ploterów firmy Hewlett-

Packard. Za datę powstania tego języka przyjmuje się rok 1976, w którym pojawiły się na rynku wspomniane plotery. Wraz z udoskonaleniem tych urządzeń język HP-GL został wzbogacony o nowe polecenia, a obecnie jego druga wersja jest oznaczona symbolem HP-GL/2.

Język HP-GL posiada także obsługę krojów wektorowych, co oznacza w praktyce możliwości sterowania urządzeniem, podobne do tych jakie daje PostScript. Duża część języka HP-GL/2 została włączona do piątego poziomu języka PCL, w którym istnieją specjalne polecenia przełączające sterowanie z „klasycznego” PCL na język HP-GL i odwrotnie.

Składnia języka HP-GL/2 jest prosta. Wszystkie instrukcje są dwuznakowymi skrótami ich nazw. Po skrócie następują parametry. Mogą one być obowiązkowe lub opcjonalne. Jako separator oddzielający od siebie parametry może służyć odstęp lub przecinek. Preferowanym separatorem jest przecinek. Każdą instrukcję kończy terminator. Terminatorami mogą być: średnik, pierwszy znak następnej instrukcji lub odstęp [10], [7], [6].

# Systemy tłumaczące

## 1. Konwersja między typami plików

Ponieważ istnieje wiele formatów języków sterowania drukarką, jest rzeczą oczywistą, że często występuje potrzeba przetwarzania formatu pliku wydruku na inny. Zależnie od rodzajów formatu źródłowego i docelowego może to być zadanie trywialne, ale może też być niemożliwe do wykonania. Obrazy zapisane w pliku wydruku mogą mieć różną postać. Może to być obraz bitmapowy, wektorowy lub wręcz zwykły tekst ASCII. Uwzględniając format języka rozumianego przez drukarkę może zachodzić potrzeba konwersji formatu obrazu zapisanego w pliku wydruku [6].

### 1.1. Mapa bitowa na mapę bitową

Przetworzenie jednego typu mapy bitowej na inny jest zwykle łatwe. Jeśli przebrniemy już przez szczegóły kodowania pliku, będziemy mieć do czynienia z pikselami, które są niemal zawsze takie same, więc konwersja jest prosta.

Jeśli przebiega z formatu bardziej do mniej deskryptywnego, np. z kolorowego do czarno-białego lub ze stopniowaniem szarości, istnieją znane metody przekształcania z zachowaniem najlepszej możliwej jakości obrazu.

### 1.2. Format wektorowy na wektorowy

Podstawowym zagadnieniem przy przekształcaniu pomiędzy formatami wektorowymi jest dostosowanie nieco różniących się semantyk poszczególnych formatów i także, w pewnym stopniu, układów współrzędnych. W najprostszym przypadku rozkazy są tłumaczone jeden do jednego, np. polecenie rysowania linii na takie samo polecenie w pliku wynikowym. Problemy pojawiają się, gdy dwa formaty nie mają odpowiadających sobie rozkazów. Jeśli format oryginalny ma polecenie rysowania elipsy, a w formacie docelowym ono nie występuje, to trzeba zastosować jedną z dostępnych metod translacji. Można przybliżyć elipsę okręgiem lub łamaną złożoną z krótkich odcinków. Jeśli format przeznaczenia ma możliwość oddzielnego skalowania osi x i y, to dobrą metodą może być ustalenie na nich różnych jednostek i narysowanie okręgu, który zostanie przeskalowany w elipsę.

### 1.3. Format wektorowy na mapę bitową

Przetwarzanie obrazu z postaci wektorowej do mapy bitowej zwane jest rastryzacją. Polega ono na znalezieniu zbioru pikseli odpowiadającego każdemu wektorowi obrazu oryginalnego. Podstawowy algorytm rastrowania jest znany od 1963 roku, kiedy to został opublikowany przez Bresenhama; okręgi i łuki mogą być rysowane przy użyciu tej samej metody.

Ten rodzaj konwersji dotyczy takich samych zagadnień jak rysowanie obrazu wektorowego na ekranie rastrowym czy drukarce laserowej, więc często daje się przystosować kod wyświetlania do konwersji na mapę bitową.

### 1.4. Mapa bitowa do formatu wektorowego

Konwersja mapy bitowej na postać wektorową jest daleko bardziej złożona niż dowolne z poprzednich przekształceń. Obecnie istnieją zadowalające algorytmy wykrywania krawędzi (ang. edge detection). Mogą one posłużyć do znalezienia linii w mapie bitowej, ale tylko w najprostszych przypadkach. Problem znajdowania linii w zeskanowanym obrazie (np. w dokumencie przesłanym faksem) pozostaje nierozwiążany.

## 2. Programy konwertujące

### 2.1. Zamiana PostScript'u na inne standardy

Najbardziej znanym programem do interpretacji PostScript'u jest GhostScript. Przez wywołanie GS z opcją -help można uzyskać informację o dostępnych urządzeniach. W GhostScript możliwa jest emulacja następujących systemów: epson, epsonc, necp6, laserjet, ljetplus, ejet2p, ljet3, paintjet, bj10e, djet500, djet500c, pjetxl,lbp8 (są to nazwy używane w programie). Możliwe jest także dodanie do programu nowych sterowników [8].

### 2.2. Zmiana innych standardów na PostScript

W przypadku tekstu ASCII zmiana standardu na PostScript nie jest skomplikowana. Istnieje kilka ogólnodostępnych narzędzi służących do tego celu. Najprostszym z nich jest program o nazwie a2ps.

W przypadku tekstów innych od ASCII (ISO 8859-1 lub PC 437), lub szczególnych sekwencji kontrolnych drukarki (ang. printer specific control sequences), konwersja może być skomplikowana. Do zmiany HP PCL na PostScript służy narzędzie lj2ps, które jest

bardzo pomocne w przypadku czcionek nieproporcjonalnych (jak dla HP LaserJet II). Problem komplikuje się jednak przy konwersji grafiki. Istnieje również konwerter dla HPGL (hpgl2ps), ale jest on rzadko spotykany. Do zmiany Epson'a na PostScript służy filtr epson2ps [8].



## Aplikacja EmuLator

Program EmuLator jest przeznaczony dla środowiska Windows od wersji 3.1. O wyborze tego systemu zadecydowało głównie prostota w jego użyciu i komunikacji z użytkownikiem za pośrednictwem bogatego interfejsu graficznego. Łatwość obsługi programu jest możliwa dzięki złożoności programowania przy pomocy funkcji API (ang. Application Programming Interface) - zestawu funkcji, z których programiści muszą korzystać, pisząc aplikacje dla Windows. Windows API zawiera ponad 600 funkcji. Pomijając wspomniane niedogodności, należy powiedzieć o korzyściach, wynikających z programowania dla Windows [13]:

- System Windows zapewnia niezależność od sprzętu. Ten sam program może wyświetlać informacje na różnych monitorach (EGA, VGA itd.) i drukować na różnych drukarkach, od mozaikowych do laserowych.
- Z punktu widzenia programisty, Windows oferuje wiele gotowych elementów interfejsu użytkownika, jak np. przyciski ekranowe, menu, okienka dialogowe, listy i okienka edycyjne.
- Windows zawiera obszerny interfejs urządzeń graficznych GDI (ang. Graphics Device Interface), służący do wyświetlania tekstu i grafiki. W szczególności, interfejs ten pozwala rysować we własnym układzie współrzędnych.

Językiem programowania został wybrany C++ w wersji dla najnowszego w chwili tworzenia programu, kompilatora Borlanda 4.52 [4]. To narzędzie pozwala na obiektowe podejście do projektowanej aplikacji [2] oraz umożliwia wykorzystanie bibliotek klas OWL i CLASSLIB [1].

Zasadniczym przeznaczeniem biblioteki klas OWL (ang. Object Windows Library) jest dostarczenie programistie kompletnego szkieletu aplikacji dla Windows. Wykorzystana została wersja 2.5 OWL Borlanda, która pozwoliła na szybkie stworzenie interfejsu użytkownika z atrakcyjnymi, graficznymi elementami sterującymi.

Biblioteka kontenerowa Borlanda CLASSLIB dostarczyła wyjściowych obiektów do przechowywania oraz przetwarzania danych.

## 1. Opis programu

Program ma za zadanie rozwiązanie problemu interpretacji obrazu graficznego zawartego w pliku wydruku na ekranie komputera. Duża liczba języków sterowania drukarkami i ploterami oraz rozmaite formy ich zapisu skłaniają programistę do

interpretacji wybranego języka drukarki. Takie podejście można zaobserwować np. w programie GhostScript interpretującym język PostScript. Praca ta polega na napisaniu programu który będzie interpretował różne języki sterowania.

Przyjęto założenie, że plik wydruku zawiera różne rozkazy powodujące akcje graficzne typu: wypisanie tekstu lub narysowanie grafiki oraz sekwencje zmieniające parametry drukarki. Wyniki można zobaczyć na wydrukowanej kartce, a więc można je także przedstawić na ekranie komputera. Poszczególne języki sterowania powodują to samo działanie przy pomocy różnych kombinacji kodów. W celu zasymulowania pracy drukarki lub plotera, program zawiera zbiór podstawowych operacji wykonywanych przez te urządzenia, oraz umożliwia wykonanie tych operacji.

Istotnym problemem było wykorzystanie przez program dowolnego języka drukarki w sposób przystępny dla użytkownika, a jednocześnie pozwalający w każdej chwili na dodawanie nowych języków sterowania. EmuLator otwiera sterownik drukarki, który jest plikiem tekstowym w ustalonym formacie, sprawdzając równocześnie czy nie zawiera on błędów w zapisie. Plik ten zawiera także ustawienia początkowe dla danej drukarki lub plotera. Użytkownik może sam stworzyć sterownik dla każdej drukarki, korzystając z zawartych w programie operacji, bez potrzeby rekompilacji programu. Jeżeli dana drukarka używa specyficznych dla siebie mechanizmów, nie ujętych w omawianym programie, sprawny programista potrafi szybko rozbudować program o nowe funkcje.

W aplikacji wykorzystane są mechanizmy MDI (ang. Multiple Document Interface), które pozwalają na przeglądanie kilku dokumentów jednocześnie. Program pozwala interpretować wydruki tylko dla jednego rodzaju drukarki w danym momencie.

Efekt końcowy można oglądać na ekranie w skali 1:1 (przy uwzględnieniu niedokładności ekranu monitora), w podglądzie wydruku całej strony, lub wydrukować na dowolnej drukarce. Wykorzystując niezależność sprzętową kontekstu urządzenia GDI, program pozwala na dokonanie konwersji znanego mu formatu drukarki lub plotera na format dowolnego urządzenia dostępnego w systemie Windows.

## 1.1. Budowa

Aplikacja powstała w oparciu o architekturę MVC (ang. Model-View-Controller), powszechnie wykorzystywaną w języku Smalltalk-80 z uwzględnieniem specyficznych możliwości biblioteki OWL 2.5. Architektura MVC dzieli aplikację na trzy warstwy:

1. Model oznacza warstwę aplikacji, w której umieszczone są wszystkie obiekty, zależne od tej aplikacji. W EmuLatorze są to wszystkie obiekty reprezentujące warstwę danych oraz operacje na nich wykonywane. W szczególności, może być

to programowalny słownik, umożliwiający tłumaczenie kodów pliku wydruku na operacje graficzne, oraz klasa przetwarzająca dane pliku wydruku na obraz graficzny, wykorzystany w warstwie prezentacji.

2. View jest tzw. warstwą prezentacyjną, która odpowiada za prezentację danych. Warstwa prezentacyjna odczytuje odpowiednie informacje z warstwy aplikacji i wyświetla je na ekranie. Warstwa ta zajmuje się również oknami graficznego interfejsu użytkownika.
3. Controller jest tzw. warstwą sterującą, pośredniczącą w przekazywaniu informacji od urządzeń wejściowych (klawiatura, myszka) do pozostałych dwóch warstw.

Model ten w trakcie realizacji ulegał nieznacznym modyfikacjom, oraz był podporządkowany nowym technikom znajdującym się w OWL 2.5, jak np. Document/View, w zależności od rozwiązywanego problemu.

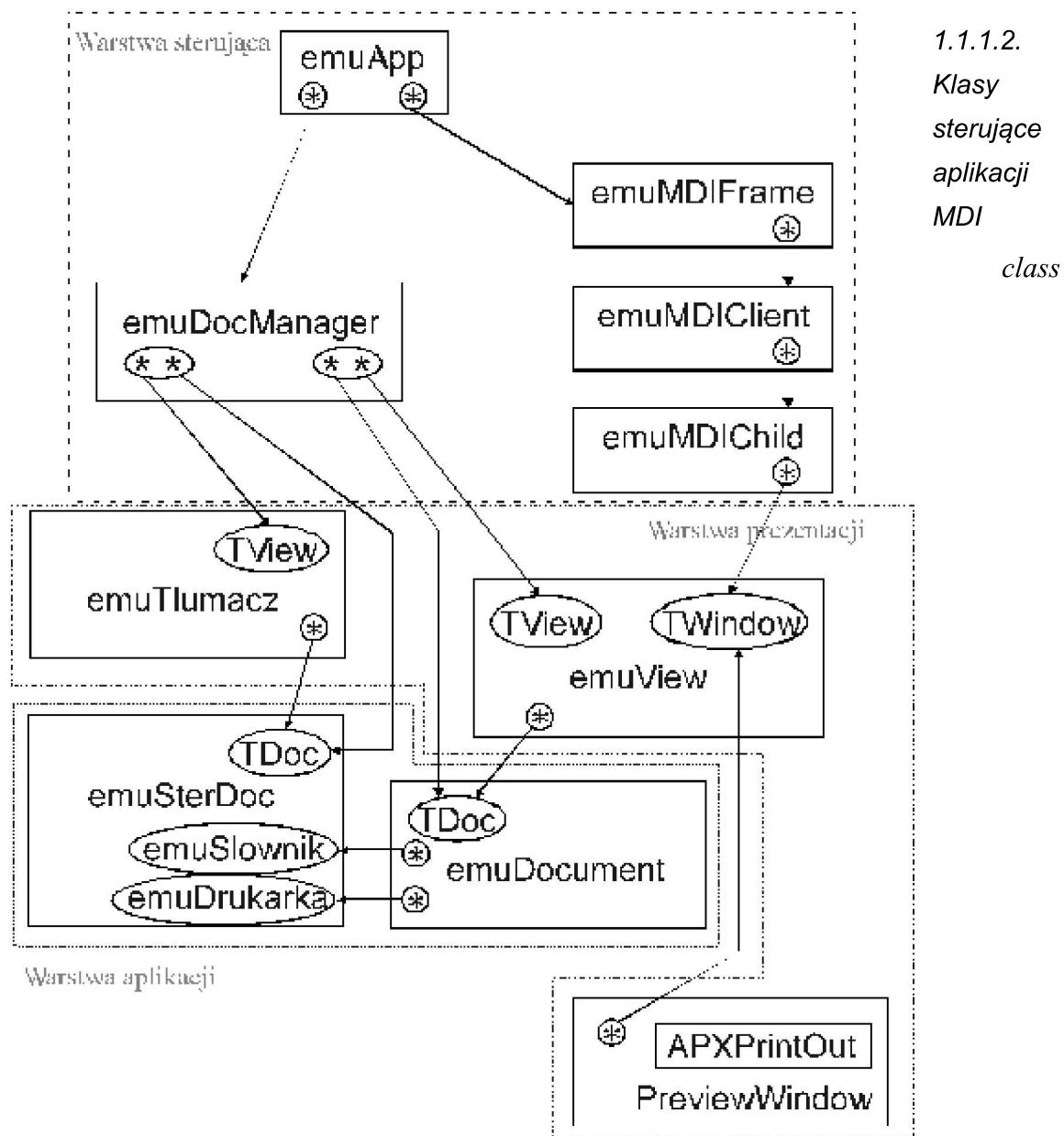
### 1.1.1. Warstwa sterująca

#### 1.1.1.1. Główna klasa aplikacji

*class emuApp: public TApplication* - najważniejsza klasa aplikacji, której implementacja znajduje się w plikach emuapp.cpp i emuapp.h, tam też zdefiniowano pomocniczą klasę *TFileDrop*, zajmującą się obsługą wczytywanych dokumentów metodą *przenieś-upuść* (ang. drag and drop). Klasa *emuApp* jest bazą dla wszystkich obiektów aplikacji. W funkcji *InitMainWindow* tworzone są obiekty klas *emuMDIFrame*, *emuMDIClient*, ładowane z zasobów elementy graficzne programu (takie jak ikony i menu), wstawiane w ramkę *emuMDIFrame* linia statusu (*TStatusBar*) i pasek narzędzi (*TControlBar*), ustawiane atrybuty głównego okna oraz uaktywniane są przestrzenne elementy dialogu funkcją *EnableCtl3d(true)*. Oprócz obsługi standardowych komunikatów Windows klasa *emuApp* jest odpowiedzialna za następujące zadania dla omawianej aplikacji:

- obsługa systemu pomocy - wczytanie pliku pomocy w trybie zwykłym lub kontekstowym;
- tworzenie nowego dokumentu i widoku - podczas otwierania nowego dokumentu i tworzenia widoku, użytkownik informowany jest jeśli plik sterownika nie jest wczytany; przy wczytywaniu nowego pliku sterownika zamykane są wszystkie okna;
- przekazanie komunikatów sterujących dla widoku dokumentu pliku wydruku (*emuView*) - klasa *emuApp* przekazuje widokowi aktualnego *emuView* żądanie przejścia do następnej lub poprzedniej strony informując w pasku narzędzi o jej bieżącym numerze;

- uaktualnianie informacji w linii statusu - klasa *emuApp* informuje o aktualnie wczytanym sterowniku drukarki lub jego braku, a w trakcie wyboru komendy z paska narzędzi lub menu wyświetlna jest podpowiedź.



Rys. F Hierarchia głównych klas aplikacji

*emuMDIFrame*: *public TDecoratedMDIFrame* (*emmdifrm.cpp*, *emmdifrm.h*) - zawiera widoczne elementy sterujące oraz obsługuje wszystkie komunikaty sterujące przeznaczone dla okien potomnych *emuMDIChild* kierując je do *emuMDIClient*.

*class emuMDIClient: public TMDIClient* (*mmdicnt.cpp*, *mmdicnt.h*) - odpowiada za zarządzanie obiektami okien *emuMDIChild* w odpowiedzi na żądania z menu Okno.

*class emuMDIChild: public TMDIChild* (mmdichld.cpp, mmdichld.h) - definiuje podstawowe zachowanie okna interfejsu MDI, zawierając w sobie wskaźnik do obiektu *TWindow*, będącego widocznym elementem widoku dokumentu.

#### 1.1.1.3. Klasa sterująca dokumentów

*class emuDocManager: public TDocManager* (emdcmngr.cpp, emdcmngr.h) - obiekt zarządzający listą bieżących, zarejestrowanych szablonów. Makro *DEFINE\_DOC\_TEMPLATE\_CLASS* tworzy szablon wiążący obiekty dokumentów z widokami, na podstawie którego *TDocManager* obsługuje standardowe komendy menu *Plik*, przekazując je odpowiednim dokumentom. Jest również odpowiedzialny za wyświetlenie elementów interfejsu użytkownika do wyboru pliku i widoku.

### 1.1.2. Warstwa prezentacji

#### 1.1.2.1. Klasa prezentacji danych sterownika drukarki

*class emuTlumacz: public TView* (emutlmcz.cpp, emutlmcz.h) - obiekt pośredniczący w dostępie do danych zawartych w dokumencie. *emuTlumacz* pozwala na uzyskanie wskaźnika do klasy słownika (*emuSlownik*) tworzonego w obiekcie dokumentu sterownika drukarki (*emuSterDoc*). Jako jedyny widok w aplikacji, *emuTlumacz* nie posiada skojarzonego ze sobą okienka i jest specjalnie obsługiwany w obiekcie głównym aplikacji. Dla każdego obiektu dokumentu pliku wydruku (*emuDocument*), *emuApp* ustawia wskaźnik do *emuTlumacz* (w danym momencie może istnieć tylko jeden obiekt *emuTlumacz*).

#### 1.1.2.2. Klasy prezentacji danych plików wydruków

*class emuView: public TWindowView* (emuview.cpp, emuview.h) - obiekt widoku dokumentu pliku wydruku dziedziczący własności dwóch klas OWL. Jako potomek *TView* obsługuje dostęp do danych *emuDocument*. Dziedzicząc z *TWindow* jest graficzną reprezentacją dokumentu w postaci widocznego okna, którego wskaźnik jest przekazywany obiekowi *emuMDIChild*. W odpowiedzi na żądania wyświetlenia kolejnej strony, *emuView* przekazuje obiekowi *emuDocument* kontekst wyświetlania do narysowania obrazu strony wydruku. Aby każda reakcja widoku na komunikat *WM\_PAINT* nie powodowała uruchamiania całego mechanizmu tłumaczenia, interpretacji i tworzenia rysunku z pliku wydruku, używany jest kontekst metapliku, pamiętany w postaci obiektu klasy *TMetaFilePict*.

*class APXPrintOut: public TPrintout* (apxprint.cpp, apxprint.h) - reprezentuje fizyczny dokument wysyłany do drukarki. Obiekt tej klasy jest odpowiedzialny za

wyrysowanie strony na drukarce fizycznej lub w oknie podglądu wydruku.

*class PreviewWindow: public TDecoratedFrame* (apxprev.cpp, apxprev.h) - tworzy ramkę podglądu wydruku oraz obsługuje wyświetlanie jednej lub dwóch stron jednocześnie w obiekcie *TLayoutWindow*. Jest również odpowiedzialny za wydruk na fizycznej drukarce w wyniku wyboru komendy *Drukuj* aplikacji.

#### 1.1.2.3. Inne klasy

*class emuEditView : public TEditView* (emueditvw.cpp, emuedtvw.h) - klasa widoku skojarzona z dowolnym typem pliku (bezpośrednio *TFileDocument*); pozwala na podstawowe operacje edycji, wydruku tekstu i wyszukiwania.

*class TBmpViewWindow* (emuabout.cpp, emuabout.h) - służy do przedstawienia początkowej wizualizacji programu w postaci bitmapy.

*class emuAboutDlg: public TDialog* (emabtdlg.cpp, emabtdlg.h) - korzystając z pomocniczej klasy *ProjectRCVersion*, podającej informacje projektu, wyświetla okienko dialogu o programie.

### 1.1.3. Warstwa aplikacji

#### 1.1.3.1. Klasy dokumentu sterownika drukarki

*class emuSterDoc: public TFileDocument* (emustrdc.cpp, emustrdc.h) - reprezentuje obiekt danych dokumentu i podaje sposób ich interpretacji widokowi. Zawiera szereg metod do obsługi fizycznego pliku na dysku, łącznie z obsługą strumieni. Każdy dokument może być skojarzony z kilkoma widokami, dlatego wspomaga komunikację z nimi, tworząc listę bieżących widoków i przesyłając do nich komunikaty o wszelkich zmianach. Dokumenty i widoki posiadają listę własności (*Property*), na podstawie których aplikacja decyduje o sposobie przetwarzania danych. *emuSterDoc*, wśród swoich własności, posiada wskaźniki do stworzonych przez niego obiektów *emuSlownik* i *emuDrukarka* w momencie otwierania dokumentu sterownika drukarki.

*class emuSlownik: public ContainerType* (emuslwnk.cpp, emuslwnk.h) - obiekt ten wywodzi się od kontenerowej klasy słownika tworzonej makrem *typedef TDictionayAsHashTable<AssociationType> ContainerType;* jest zbiorem asocjacji, wraz z metodami ich dodawania i wyszukiwania. Asocjacja (*typedef TIIAssociation<emuMyClass, GraphicalObject> AssociationType;*) stanowi parę wskaźników do obiektów *emu MyClass* i *GraphicalObject*. Obiekt klasy *emuDocument*, znajdujący kod sterujący, wykonuje operację graficzną, wywiedzioną z *GraphicalObject*. W trakcie tworzenia słownika przez

*emuSterDoc*, tworzone są dynamicznie i dodawane tylko te obiekty graficzne, które są potrzebne do interpretacji danego urządzenia.

*class emuMyClass* (*emumycla.cpp*, *emumycla.h*) - pojedynczy kod drukarki lub plotera, wraz z operatorem porównania pozwalającym na znalezienie go.

*class GraphicalObject* (*emugrobject.cpp*, *emugrobject.h*) - bazowa klasa dla operacji graficznych na kontekście urządzenia, z której są wirtualnie wywiedzione obiekty odpowiadające poszczególnym kodom sterującym. Obiekt tej klasy nie wykonuje żadnej operacji, lecz w celach diagnostycznych wypisuje nazwę operacji do wykonania.

*class GraphObjHPGL: public virtual GraphicalObject* (*emughpgl.cpp*, *emughpgl.h*) - bazowa klasa dla operacji graficznych specyficznych dla ploterów. Z niej wywodzą się klasy konkretnych operacji graficznych *class HPGL\_0xXXXX: public virtual GraphObjHPGL*, gdzie *0xXXXX* stanowi unikalny numer wewnętrzny aplikacji dla danej operacji.

*class GraphObjPCL: public virtual GraphicalObject* (*emugopcl.cpp*, *emugopcl.h*) - bazowa klasa dla operacji graficznych specyficznych dla drukarek atramentowych i laserowych. Z niej wywodzą się klasy konkretnych operacji graficznych *class PCL\_0xXXXX: public virtual GraphObjPCL*, gdzie *0xXXXX* stanowi unikalny numer wewnętrzny aplikacji dla danej operacji.

*class GraphObjIBMP*: *public virtual GraphicalObject* (plik *emugopro.cpp*, *emugopro.h*) - bazowa klasa dla operacji graficznych specyficznych dla drukarek igłowych. Z niej wywodzą się klasy konkretnych operacji graficznych *class IBMP\_0xXXXX: public virtual GraphObjIBMP*, gdzie *0xXXXX* stanowi unikalny numer wewnętrzny aplikacji dla danej operacji.

*class emuDrukarka* (*emudrkrk.cpp*, *emudrkrk.h*) - klasa zawierająca ustawienia początkowe emulowanego urządzenia, wykorzystana również w momencie tworzenia obrazu graficznego.

#### 1.1.3.2. Klasy dokumentu pliku wydruku

*class emuDocument: public TFileDocument* (*emudcmnt.cpp*, *emudcmnt.h*) - obsługuje operacje związane z fizycznym plikiem wydruku. Na żądanie *emuView* analizuje plik, szukając kodów sterujących w *emuSlownik* i wykonując operacje graficzne wirtualną funkcją *Draw()* obiektu wywodzącego się od *GraphicalObject*.

*class emuStrona* (*emustron.cpp*, *emustron.h*) - klasa wspomagająca tworzenie rysunku konkretnej strony. *emuDocument* wykorzystuje zbiór stron (*typedef TIArrayAsVector<emuStrona> emuStronki;*) do pamiętania pozycji strumienia pliku wydruku i ustawień drukarki (*emuDrukarka*) dla każdej ze stron.

## 1.2. Rozbudowa

Program stanowi doskonały szkielet do rozbudowy o nowe funkcje i moduły. Wykorzystując technikę OWL Document/View, można stworzyć inne rodzaje widoków i dokumentów dla istniejących lub nowych obiektów. Również stworzone moduły interpretujące pliki wydruku mogą zostać łatwo rozszerzone. Przy rozbudowie programu pomocne będą komentarze zamieszczone w plikach źródłowych oraz klasy pomocnicze, które nie zostały wbudowane w końcowej wersji programu.

### 1.2.1. Dodawanie obiektów widoków

Aby dodać obiekt widoku, wystarczy zdefiniować nową klasę wywodzącą się od *TView*, w której wykorzystane będą metody zawarte w obiekcie wybranego dokumentu. Jeżeli w widoku do prezentacji użyte będą specjalne formy danych, należy zdefiniować je w obiekcie dokumentu. Obiekt widoku może być powiązany z dokumentem, który posiada kilka innych sposobów prezentacji. Aby program mógł je obsłużyć, należy zdefiniować odpowiednie szablony dla *emuDocManager*, np.:

```
DEFINE_DOC_TEMPLATE_CLASS(emuSterDoc, TSterListView, DocType7);  
DocType7 __dvt7("Podgląd słownika", "*.emu", 0, "EMU", dtAutoDelete);
```

gdzie: *emuSterDoc* jest istniejącą klasą dokumentu, a *TSterListView* jest klasą widoku sterownika. Widok może występować w postaci pokazanego na ekranie okna lub w dowolnej, innej formie interpretacji danych zawartych w dokumencie.

### 1.2.2. Dodawanie obiektów dokumentów

Każdy obiekt dokumentu odpowiada za wczytanie danych i prawidłowe podanie ich widokom. Jeżeli dokument posiada kilka widoków, a jeden z nich pozwala na zmianę danych, konieczna jest jeszcze obsługa komunikatów przekazujących informacje o tym fakcie innym widokom. Komunikacja między widokami i dokumentami może przebiegać przez komunikaty, listę własności (*Prosperities*) lub bezpośrednie odwołania wskaźnikami. Więcej danych o współpracy dokumentów z widokami można znaleźć w dokumentacji kompilatora Borland C++ 4.52.

### 1.2.3. Rozbudowa funkcji graficznych

W programie EmuLator, w trakcie interpretacji pliku wydruku wykorzystana jest funkcja:

```
int emuDocument::Rysuj( TDC& strDC, emuStrona* strona )
```

gdzie: *strDC* jest kontekstem urządzenia, w którym ma powstać obraz; *strona* jest wskaźnikiem do obiektu zawierającego bieżące ustawienia strony oraz kopię ustawień drukarki (*emuDrukarka*) z poprzednio stworzonej strony, lub (w przypadku pierwszej strony) ustawienia urządzenia przeczytane przez *emuSterDoc*. Jeżeli w strumieniu dokumentu zostanie znaleziony kod sterujący, obiekt klasy *emuSlownik* zwraca wskaźnik do asocjacji, z której jest uzyskiwany wskaźnik do odpowiedniego obiektu graficznego. Operacja jest wykonywana poprzez wywołanie wirtualnej funkcji obiektu wywodzącego się od *GraphicalObject*:

```
if (znalazl)
{
    if (obiekt = znalazl->Value())
        obiekt -> Draw(strDC, strumyk, strona);
```

Funkcja *void GraphicalObject::Draw( TDC& dc, TInStream\* is, emuStrona\* str)* do wykonania zadania jest wywoływana z następującymi parametrami:

- *TDC& dc* - referencja do kontekstu urządzenia, na którym należy wykonać operację graficzną;
- *TInStream\* is* - wskaźnik do strumienia pliku wydruku, z którego mogą być pobierane dalsze dane do wykonania operacji;
- *emuStrona\* str* - wskaźnik do obiektu, w którym można znaleźć bieżące ustawienia emulowanego urządzenia (*emuDrukarka \*druka*) oraz, w przypadku wystąpienia błędu lub końca strony należy ustawić jedną z wartości zmiennej *status*:

```
enum {
    Blad=0,
    KoniecStrony,
    KoniecPliku,
    Dalej,
};
```

Aby rozszerzyć program o nowe funkcje graficzne, należy stworzyć obiekt wywodzący się wirtualnie wprost lub pośrednio (np. tak jak obiekty *GraphObjHPGL*) z klasy *GraphicalObject*, który wykonuje operację korzystając z wyżej podanych parametrów. Następnie należy wybrać unikalny numer wewnętrzny aplikacji dla operacji graficznej (z zakresu od 0x0000 do 0xFFFF) i umieścić operację dodawania obiektu w *emuslwnk.cpp*:

```
int
emuSlownik::AddItem(const string& mc, const unsigned long int & mv, const
string& mvs)
```

```

{
    int wynik=0;
    switch (mv) {
        case 0xFFFFXX: {
            AssociationType assoc( new emuMyClass(mc),
                new GraphicalObject(mvs) );
            wynik=Add(assoc);
            break;
        }
    }
}

```

Ta funkcja jest wywoływana przez *emuSterDoc*, podczas czytania pliku sterownika urządzenia z następującymi parametrami:

- *mc* - kod sterujący emulowanego urządzenia;
- *mv* - kod wewnętrzny aplikacji dla operacji graficznej;
- *mvs* - nazwa operacji wykorzystywana w celach diagnostycznych, a pobierana z pliku sterownika.

#### 1.2.4. Rozszerzenie własności emulowanych urządzeń

Jeżeli do wykonania operacji graficznej potrzebne są parametry, których nie można znaleźć w definicji *emuDrukarka* (plik emudrkrk.h), należy je dodać do tej klasy. Inicjalizacja wartości następuje w *bool emuSterDoc::GetDrukarka(TInStream\* is)* gdzie należy dodać procedurę czytania parametrów np.:

```

if (klucz.contains("PAGE SIZE")) // Rozmiar strony
{
    wiersz += SkipAll(*is);
    long x = GetLong(*is);
    wiersz++;
    if (!is->good()) goto error;
    wiersz += SkipAll(*is);
    long y = GetLong(*is);
    wiersz++;
    if (!is->good()) goto error;
    drukarka->SetRStr(x*(metric ? mm_pkt : cal_pkt),
        y*(metric ? mm_pkt : cal_pkt));
}
else

```

*klucz* jest napotkanym słowem kluczowym w pliku sterownika drukarki według stosowanej konwencji:

! Page Size

Następnie, dowolne parametry czytane są ze strumienia tekstowego `*is` odpowiednią dla nich funkcją. Funkcja `SkipAll(*is)` omija wszystkie znaki w strumieniu do napotkania znaku „!” na początku wiersza. Zmienna `wiersz` wskazuje numer wiersza w którym wystąpił błąd. Konstrukcja `x*(metric ? mm_pkt : cal_pkt)` pozwala ustawić parametry we wcześniejszej ustalonych jednostkach (obecnie `mm` lub `cal`).

### 1.2.5. Klasy pomocnicze

Podczas uruchamiania projektu można skorzystać z dołączonych do źródeł klas diagnostycznych. Należy w tym celu włączyć dołączanie kodu dla debugger'a w opcjach projektu, oraz zmienić komentarze w pliku `emuapp.cpp`, uaktywniając szablon tych widoków. Są to następujące klasy:

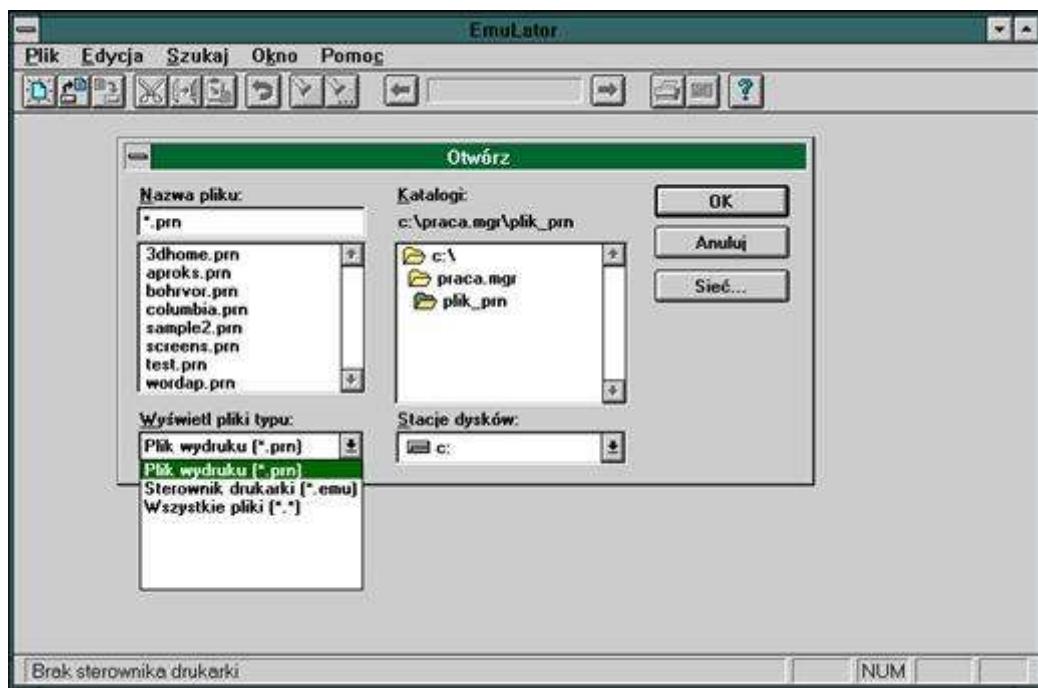
- `class TDumpView : public TListBox, public TView` (`dumpview.cpp`, `dumpview.h`) - pozwala na skojarzenie z dowolnym typem dokumentu, oraz pokazuje dokładnie jego zawartość z kodami szesnastkowymi poszczególnych bajtów pliku;
- `class TInfoView : public TWindowView` (`infoview.cpp`, `infoview.h`) - pozwala na zobaczenie listy własności (*Prosperities*) dowolnego dokumentu;
- `class TSterListView : public TListBox, public TView` (`strlisvw.cpp`, `strlisvw.h`) - widok specjalnie stworzony na potrzeby dokumentu klasy `emuSterDoc`, pokazujący zawartość słownika zainicjalizowanego plikiem sterownika drukarki.

- **Obsługa programu**
- **Instrukcja użytkowania**

Program EmuLator przed rozpoczęciem użytkowania wymaga zainstalowania na dysku twardym komputera. Jego pakiet instalacyjny znajduje się na dyskietkach typu HD 3.5". Przebieg instalacji jest typowy dla środowiska Windows.

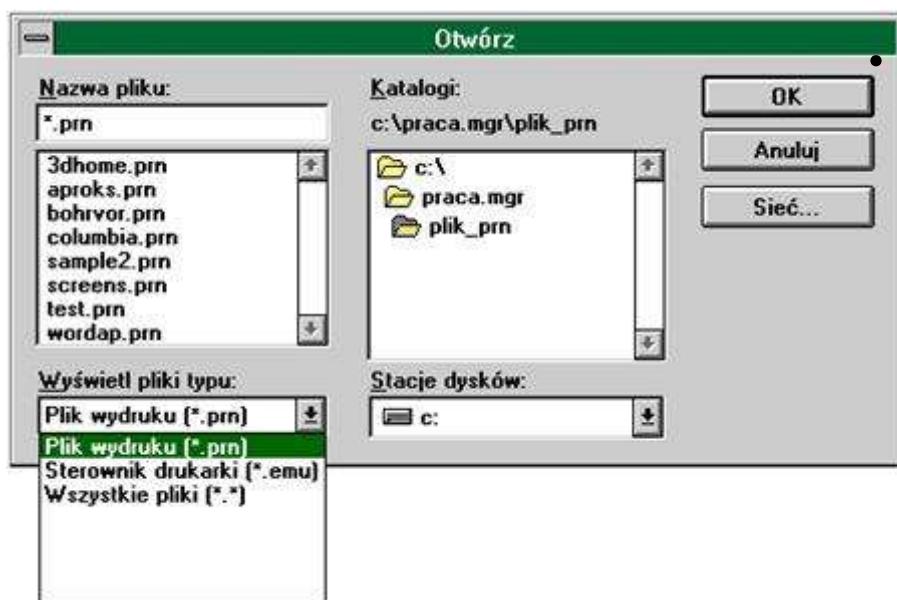
Po zainstalowaniu program EmuLator posiada w oknie Menedżera Programów swoją grupę pod nazwą „EmuLator drukarki i plotera”. Ikoną głównego programu aplikacji jest ikona o nazwie „EmuLator”.

Po uruchomieniu aplikacji na ekranie monitora ukazuje się okno aplikacji o tytule „EmuLator” (Rys. 7).



Rys. G Okno główne aplikacji

Operacje obsługi programu ograniczają się najczęściej do kilku podstawowych:



Rys. H Okno otwórz

Otwieranie dokumentu  
Otwarcie dokumentu zawierającego wydruk drukarki musi być poprzedzone wczytaniem sterownika urządzenia dla którego plik został

stworzony. Załadowanie sterownika dokonywane jest poprzez wybranie opcji *Otwórz*

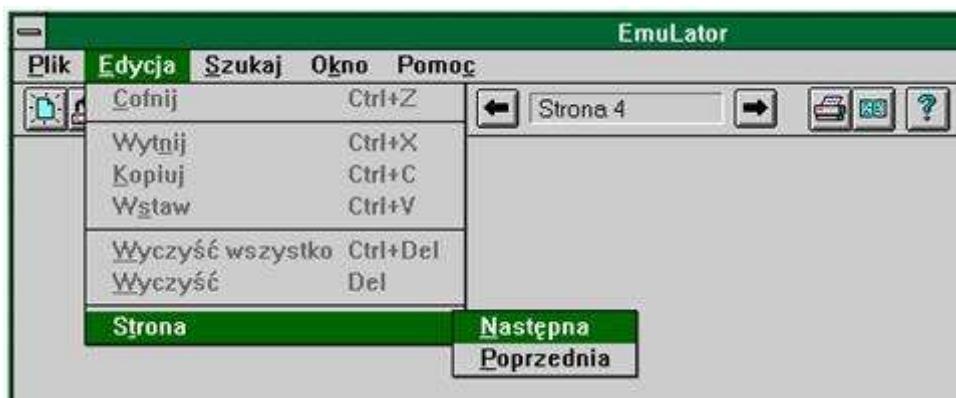
z menu *Plik* lub kliknięcie odpowiedniej ikony z paska narzędzi. Po rozwinięciu okna dialogu *Otwórz* należy ustawić typ pliku na *Sterownik Drukarki* z rozszerzeniem \*.emu i w polu dialogu *Nazwa pliku* wpisać nazwę sterownika odpowiadającego danej drukarce. Nazwa urządzenia dla którego został wczytany sterownik ukaże się w linii statusu w dolnej części okna (Rys. 9).

W przypadku błędnie zapisanego kodu rozkazu w sterowniku, podczas próby wczytywania go program wyświetli informację z numerem linii z błędnie wpisany kodem.



Rys. I Linia statusu z wczytanym sterownikiem.

Program jest przygotowany do otwierania pliku wydruku. Postępowanie jest zbliżone do powyższego, z tą różnicą, że jako typ otwieranego pliku należy wybrać *Pliki Wydruku* z rozszerzeniem \*.prn. Przy próbie ponownego otwarcia wczytanego już dokumentu program poinformuje o tym użytkownika i nie pozwoli mu na tę operację. Możliwe jest natomiast dodanie innych widoków tego samego dokumentu; opcja *Dodaj Widok* z menu *Okno*. Zawartość pliku wydruku zostanie pokazana w oknie MDI o tytule otwartego dokumentu. Uказание następnej (poprzedniej) strony dokumentu (jeżeli istnieje) następuje po wybraniu opcji *Strona* w menu *Edycja* lub przycisków strzałek w pasku narzędzi (Rys. 10).



Rys. J Przeglądanie stron dokumentu

- **Zamykanie dokumentu**

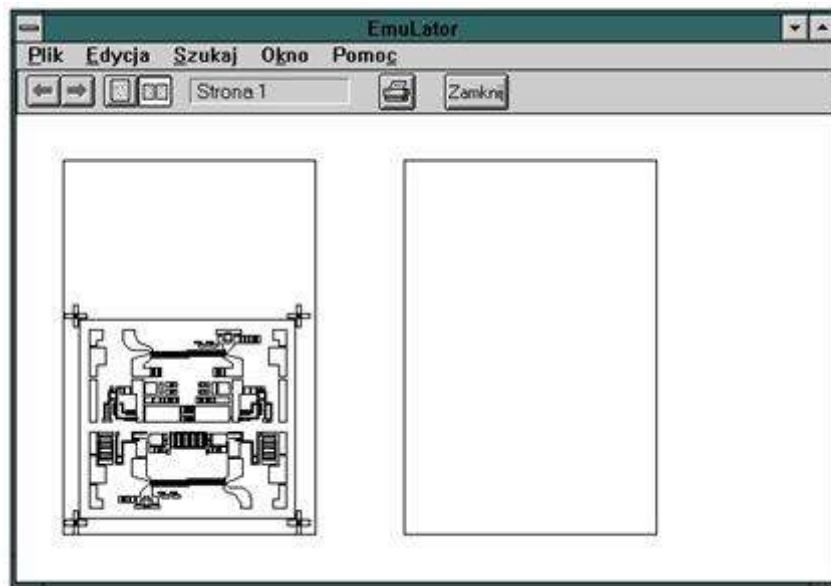
Aby zamknąć otwarty dokument należy wybrać opcję *Zamknij* w menu *Plik*. W chwili zmiany urządzenia, dla którego pokazywany jest dokument wszystkie widoki (dokumenty) zostają zamknięte automatycznie.

- **Podgląd wydruku**

Pokazanie dokumentu w takiej postaci, jaką będzie on miał na stronie po wydrukach można uzyskać poprzez wybór z menu *Plik* opcji *Podgląd Wydruku* lub odpowiedniej ikony z paska narzędzi. Nowy obraz okna dokumentu pokazuje jedną lub dwie strony pliku wydruku. Przegląd następnych (poprzednich) stron następuje po kliknięciu myszką w ikony strzałek na pasku narzędzi (Rys. 11).

- **Wydruk dokumentu**

Po przejrzeniu dokumentu można go wydrukować na urządzeniu podłączonym do stanowiska. Wydruk następuje po wybraniu opcji *Drukuj* z menu *Plik* lub po kliknięciu myszką w odpowiednią ikonę paska narzędzi. Wydruk nastąpi w aktualnych ustawieniach drukarki, które można zmienić opcją *Ustawienia Drukarki* w menu *Plik*.

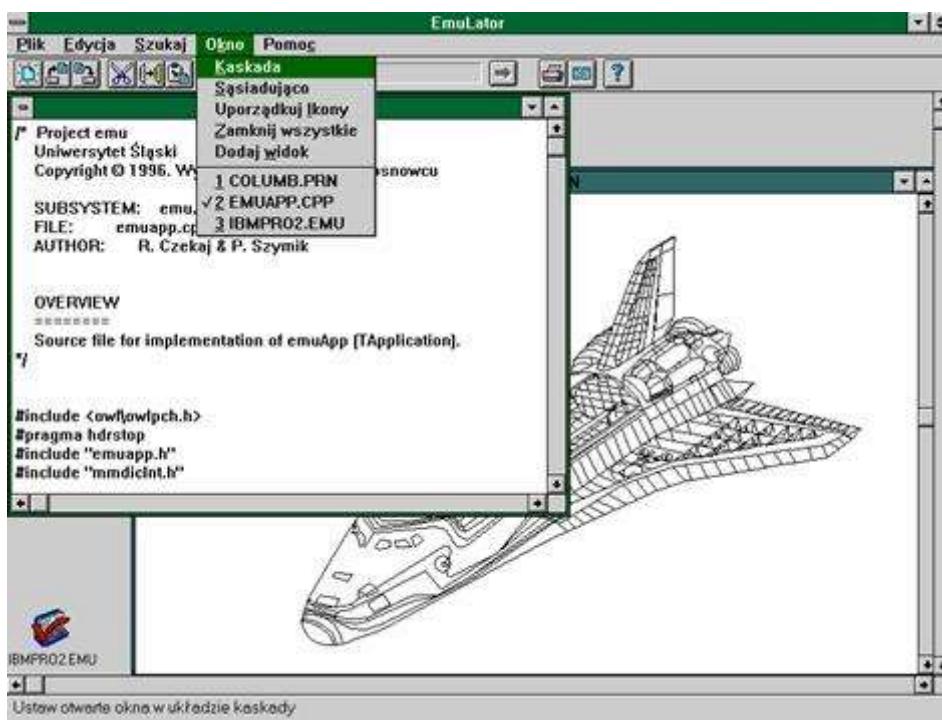


Rys. K Podgląd wydruku

### Edycja sterowników drukarek

Sterowniki drukarek mogą być modyfikowane, w celu ewentualnego dopasowania do używanego urządzenia. Muszą być one wczytane w oknie dialogu *Otwórz* przy wybranym

typie pliku *Wszystkie pliki (\*.\*)*. Operacje edycji umożliwiają opcje menu *Edycja*, *Szukaj*, *Okno*. Sterownik nie może być wczytany w trakcie modyfikacji.



Rys. L Edycja plików tekstowych

#### Uzyskanie pomocy

System pomocy dla aplikacji EmuLator można wywołać w menu *Pomoc*. Umożliwia on uzyskanie pomocy na zasadzie szukanego klucza słownego lub szukanego tematu. Po naciśnięciu kombinacji klawiszy Shift+F1, wskaźnik myszy zmienia kształt, pozwalając wskazać element sterujący, na temat którego potrzebna jest informacja.

- **Opis sterownika**

Format pliku sterownika drukarki został stworzony dla aplikacji EmuLator przy użyciu prostej konwencji zapisu. W odróżnieniu od innych programów, EmuLator wykorzystuje sterowniki, które nie są komplikowanymi programami. Zapisane są one w formacie pliku tekstowego, który nie wymaga innych dodatkowych operacji. Program posiada wbudowaną kontrolę poprawności notacji i w chwili rozpoznania nieprawidłowości użytkownik zostanie poinformowany, w której linii ona wystąpiła. Możliwa do zapisania liczba kodów sterowania drukarki jest w zasadzie nieograniczona, podobnie jak liczba możliwych ustawień początkowych drukarki lub plotera.

Każda linia, rozpoczynająca się od znaków: „!”, „#”, „\$”, „%”, traktowana jest przez moduł czytający kody sterownika jako komentarz. Każda linia rozpoczynająca się od znaku „!” jest czytana przez moduł inicjalizujący ustawienia drukarki, a każdy wiersz musi

rozpoczynać się znakiem komentarza lub zawierać poprawne dane. Od momentu napotkania znaku komentarza program „ignoruje” kolejne znaki w strumieniu, aż do napotkania znaku końca linii.

Pierwszy wiersz sterownika jest niezbędny, drugi bez komentarza oznacza nazwę urządzenia emulowanego. Jest podawana w linii statusu programu.

```
EMU Driver File
$ Plik sterownika plotera:
Ploter HP-GL 2
$ dla programu EmuLator
```

Opis rozkazu kopiowany jest przez moduł obsługi strumienia jako cały wiersz.

Poniższy fragment zawiera dane dla jednego kodu sterującego drukarki.

```
# Przykład sposobu opisu danego kodu plotera:
2          2 bajty kodu do przeczytania
0x2b      kod sterujący urządzenia
opis rozkazu  opis słowny kodu
0xfffff   kod programu EmuLator dla operacji (hex od 0x0 do
#          0xfffff)
```

Liczby można podawać dowolnie wg konwencji języka C:

22	-	DEC
0x22	-	HEX
022	-	OCT

oddzielone spacją, np. 0x56 0x53

Rozkazy podzielone są na grupy. Podział wprowadzony jest tylko ze względu na zwiększenie przejrzystości zapisu sterownika.

```
#####
#           KODY ROZKAZÓW JĘZYKA HP-GL2
#           GRUPA ROZKAZÓW "0A"
#####
```

W kolejnych wierszach pliku sterownika następuje już właściwy opis kodu rozkazu. Pierwszy parametr oznacza ilość bajtów kodu sterownika do przeczytania. Następny wiersz zawiera właściwe kody sterujące urządzenia. Po wierszu opisu, znajduje się unikalny numer operacji graficznej aplikacji EmuLator. W poniższym przypadku jest to jednobajtowy kod 0D oznaczający powrót karetki, znajdujący się w grupie 0A, a oznaczony numerem 11.

```
1
0x0D
powrót karetki
0xA11
```

Opis jednego kodu sterującego drukarki musi być oddzielony od kolejnego znakiem komentarza, np.:

```
1
0x0A
przesuw o wiersz LF
```

```
0x0A01
#
2
0x56 0x53
wybór prędkosci rysowania
0x131B
```

Lista wszystkich dostępnych operacji graficznych, wraz z odpowiadającymi numerami, znajduje się w katalogu sterowników drukarki, w pliku *emu\_nrop.txt*.

Po opisie wszystkich niezbędnych kodów sterujących drukarki lub plotera, następuje opis ustawień początkowych emulowanego urządzenia. Pierwsza z pozycji pojedynczego bloku opisującego jedno z ustawień jest słowem kluczowym, kolejne są jego wartościami.

```
#!!!!!!!!!!!!!! USTAWIENIA POCZĄTKOWE DRUKARKI !!!!!!!
#           Pierwszy wiersz rodzaj ustawien, drugi wartosc
#!!!!!!!!!!!!!! !!!!!!!!!!!!!!! !!!!!!!!!!!!!!! !!!!!!!!!!!!!!!
# Rodzaj urzadzenia
!Device
#!0 # nieznany typ
#!1 # drukarka iglowa
#!2 # drukarka atramentowa
#!3 # drukarka laserowa
!10 # ploter
#!!!!!!!!!!!!!! !!!!!!!!!!!!!!! !!!!!!!!!!!!!!! !!!!!!!!!!!!!!!
# Stosowana dalej jednostka
# mm
# cal inch
!Unit
!mm
#
```

Wiersze, które nie są zaznaczone jako komentarz inicjalizują wartości jako domyślne. Są one zmieniane w momencie wystąpienia w pliku wydruku kodu sterującego, modyfikującego daną wartość.







## Podsumowanie

Aplikacja EmuLator stworzona została z myślą o uniwersalnym rozwiązaniu problemu interpretacji plików wydruku. Znajduje się ona w wersji instalacyjnej na dyskietkach dołączonych do części opisowej pracy. Program zawiera zestaw podstawowych funkcji graficznych, pozwalających na interpretację plików wydruku drukarki igłowej IBM ProPrinter oraz plotera języka HPGL.

Plik wydruku może zawierać proste dane tekstowe oraz obraz graficzny. Program dokonuje analizy pliku wydruku, a po odnalezieniu kodu sterującego, wywołuje odpowiadającą mu funkcję graficzną, przedstawiając efekt działania na monitorze komputera. Przeszukiwanie pliku pod kątem występowania kodów sterujących polega na porównywaniu znaków pobranych ze strumienia pliku wydruku ze znanymi kodami zapisanymi w pliku tekstowym sterownika. Po rozpoznaniu kodu sterującego, następne bajty danych są traktowane, jako argumenty wywołania operacji graficznej. Nierozpoznana sekwencja znaków traktowana jest, jako prosty tekst ASCII pokazywany na ekranie. W ten sposób prezentowane są także pliki zapisane w formacie tekstowym.

Program może zostać w prosty sposób przystosowany do emulacji innych rodzajów urządzeń, poprzez edycję tekstowych plików sterowników drukarek w zakresie wbudowanych funkcji graficznych. Istnieje możliwość szybkiego stworzenia kolejnych obiektów wykonujących inne funkcje drukarki lub plotera. Wiąże się to jednak z całkowitą rekompilacją programu.

W opisywanej postaci program może być przystosowany do czytania dowolnego, nieznanego formatu plików wydruku, jak również innych plików zawierających dowolne dane możliwe do pokazania na ekranie monitora (np. graficzne wektorowe i bitmapowe, edytorów tekstów, itp.).

Trzeba przyznać, że uniwersalność stworzonego programu nie wpłynęła pozytywnie na szybkość analizy. Kody sterujące wyszukiwane są, począwszy od najdłuższego słowa znajdującego się w słowniku. Metoda ta pozwala ujednolicić obsługę różnych formatów sterowania, tzn. można czytać zarówno języki sterowania w formacie tekstowym (np. PostScript), jak również sekwencje sterujące rozpoczynające się od znaku ESC (np. PCL i ESC/P). Jednak, przy plikach zawierających w większości znaki ASCII, szybkość analizy zmniejsza się, wraz ze wzrostem długości słowa sterującego. Możliwa jest modyfikacja programu przez zastąpienie kontenerowej klasy Borlanda (z której wywiedziony jest *emuSlownik*) szybszą strukturą danych z efektywniejszą metodą wyszukiwania. Innym rozwiązaniem jest stosowanie różnych metod analizy pliku dla

każdego formatu. Powoduje to jednak konieczność pisania przez programistę osobnego modułu dla każdego z nich, podobnie jak w tradycyjnych kompilowanych sterownikach i filtrach importu.

Możliwe jest udoskonalenie programu, przez wzbogacenie go o mechanizmy wymiany danych przy pomocy Schowka, lub nawet OLE 2. Przez dopisanie odpowiednich funkcji do klas wywiedzionych z *GraphicalObject*, możliwa mogłaby być nawet kontrola i poprawa poszczególnych obiektów, np. skalowanie dla bitmap, dowolne transformacje dla rysunków wektorowych, lub zmiana rodzaju czy wielkości czcionki.

EmuLator pozwala na wydruk oglądanych plików, po uprzednim ich zinterpretowaniu. Wydruk danych możliwy jest na dowolnej drukarce, dostępnej w środowisku Windows. Ze względu na niezależność sprzętową kontekstu urządzenia GDI, program umożliwia dokonanie niejawnej konwersji dowolnego znanego mu formatu, w sposób pozwalający mu przesyłać wyniki do dowolnego urządzenia dostępnego w systemie Windows.

Biorąc pod uwagę powyższe rozważania oraz spostrzeżenia wynikające ze sposobu konstrukcji i działania aplikacji EmuLator, konstruktory uważają, że cel oraz założenia tej pracy zostały w pełni zrealizowane.



## Literatura

- [1] Barkakati N.: *Grafika i animacja w Windows*. Warszawa, Intersoftland 1994, ( tłum. z ang.).
- [2] Barteczko K.: *Programowanie obiektowe; Praktyczne wprowadzenie do programowania obiektowego w języku C++*. Warszawa, LUPUS 1993
- [3] Drożdżewicz P.: *Programowanie dla Windows w języku C*. Warszawa, Lynx-SFT 1994.
- [4] Faison T.: *Borland C++ 4.5 programowanie obiektowe*. Warszawa, Oficyna Wydawnicza READ ME 1996, ( tłum. z ang.).
- [5] Klein M.: *Przewodnik po bibliotekach DLL i sposobach zarządzania pamięcią*. Warszawa, Intersoftland 1994, ( tłum. z ang.).
- [6] Levine J.: *Programowanie plików graficznych w C/C++*. Warszawa, Translator 1994, ( tłum. z ang.).
- [7] Marciniak A.: *Język PCL*. Poznań, NAKOM 1992.
- [8] McCoy B.C.: *Summary - printers FAQ Home Page*. Usenet, comp.periph.printers Frequently Asked Question (FAQ) List, 06.08.1996
- [9] Osiak S.: *PostScript krok po kroku*. Warszawa, Agencja Wydawnicza M&M 1991.
- [10] Smith N.E.: *Drukarki laserowe*. Warszawa, ZNI MIKOM 1995, ( tłum. z ang.).
- [11] Sowiński R.: *Gwiazda plotera*. CADmania Nr 5 (11), listopad 1995
- [12] Wacławek R.: *Programowa obsługa drukarek laserowych*. Warszawa, Komputerowa Oficyna Wydawnicza HELP 1992.
- [13] Wacławek R.: *Windows od kuchni*. Warszawa, Komputerowa Oficyna Wydawnicza HELP 1993.
- [14] Zalewski A.: *Programowanie w językach C i C++ z wykorzystaniem pakietu Borland C++*. Poznań, NAKOM 1995.
- [15] *Drukarka mozaikowa D-100MPC*. Błonie, Zakłady Mechaniczno-Precyzyjne Mera-Błonie 1991.
- [16] *Instrukcja obsługi drukarki LC-20*. Warszawa, Intersoftland 1991.

## Streszczenie

W omawianej pracy przedstawione zostały techniczne aspekty działania nowoczesnych drukarek i ploterów oraz sposoby sterowania tych urządzeń za pomocą języków sterowania wydrukiem. Opisany program komputerowy powstał w wyniku analizy poszczególnych formatów języków sterujących.

Aplikacja EmuLator umożliwia emulację działania wybranej drukarki i plotera, oraz pozwala na prezentację graficzną plików wydruku na ekranie komputera. Do budowy programu wykorzystane zostało środowisko graficzne Windows 3.11 oraz pakiet narzędzi programistycznych Borland C++ w wersji 4.52.

W programie EmuLator znajduje się zestaw podstawowych funkcji graficznych, pozwalających na interpretację plików wydruku drukarki igłowej IBM ProPrinter oraz plotera języka HPGL. W zakresie tych funkcji, każdy użytkownik może łatwo przygotować aplikację do analizy plików wydruku dla innych urządzeń poprzez edycję tekstowych plików sterowników drukarek.

Dokładna dokumentacja oraz komentarze w plikach źródłowych programu pozwalają na rozbudowę programu o nowe funkcje graficzne dla innych urządzeń. Obiektywne właściwości języka C++ umożliwiają wzbogacenie aplikacji o funkcje, które nie zostały przewidziane przez twórców.

Program EmuLator, wraz z systemem pomocy, dostępny jest na dyskietkach w wersji instalacyjnej.



# Spis treści

<b>WSTĘP.....</b>	<b>3</b>
<b>TECHNICZNE ASPEKTY PRACY DRUKAREK I PLOTERÓW 4</b>	
1. KLASYFIKACJA DRUKAREK.....	4
1.1. Drukarki mozaikowe (drukarki uderzeniowe).....	4
1.2. Drukarki Daisywheel i maszyny do pisania.....	5
1.3. Drukarki atramentowe.....	5
1.4. Drukarki laserowe i drukarki LED.....	6
1.5. Drukarki kolorowe.....	7
1.6. Inne rodzaje drukarek.....	9
2. KLASYFIKACJA PLOTERÓW.....	10
2.1. Plotery tablicowe.....	10
2.2. Plotery pisakowe (bębnowe).....	10
2.3. Plotery atramentowe.....	11
3. JĘZYKI STEROWANIA DRUKAREK I PLOTERÓW.....	12
3.1. Rozszerzone formaty tekstowe.....	13
3.2. Języki opisu strony.....	14
3.3. Inne języki sterowania drukarek.....	16
3.4. Języki sterowania ploterów.....	18
<b>SYSTEMY TŁUMACZĄCE.....</b>	<b>19</b>
1. KONWERSJA MIĘDZY TYPAMI PLIKÓW.....	19
1.1. Mapa bitowa na mapę bitową.....	19
1.2. Format wektorowy na wektorowy.....	19
1.3. Format wektorowy na mapę bitową.....	20
1.4. Mapa bitowa do formatu wektorowego.....	20
2. PROGRAMY KONWERTUJĄCE.....	20
2.1. Zamiana PostScript'u na inne standardy.....	20
2.2. Zmiana innych standardów na PostScript.....	20
<b>APLIKACJA EMULATOR.....</b>	<b>22</b>
1. OPIS PROGRAMU.....	22
1.1. Budowa.....	23

1.2. <i>Rozbudowa</i> .....	30
<b>PODSUMOWANIE</b> .....	<b>35</b>
<b>LITERATURA</b> .....	<b>37</b>
<b>STRESZCZENIE</b> .....	<b>38</b>
<b>SPIS TREŚCI</b> .....	<b>39</b>