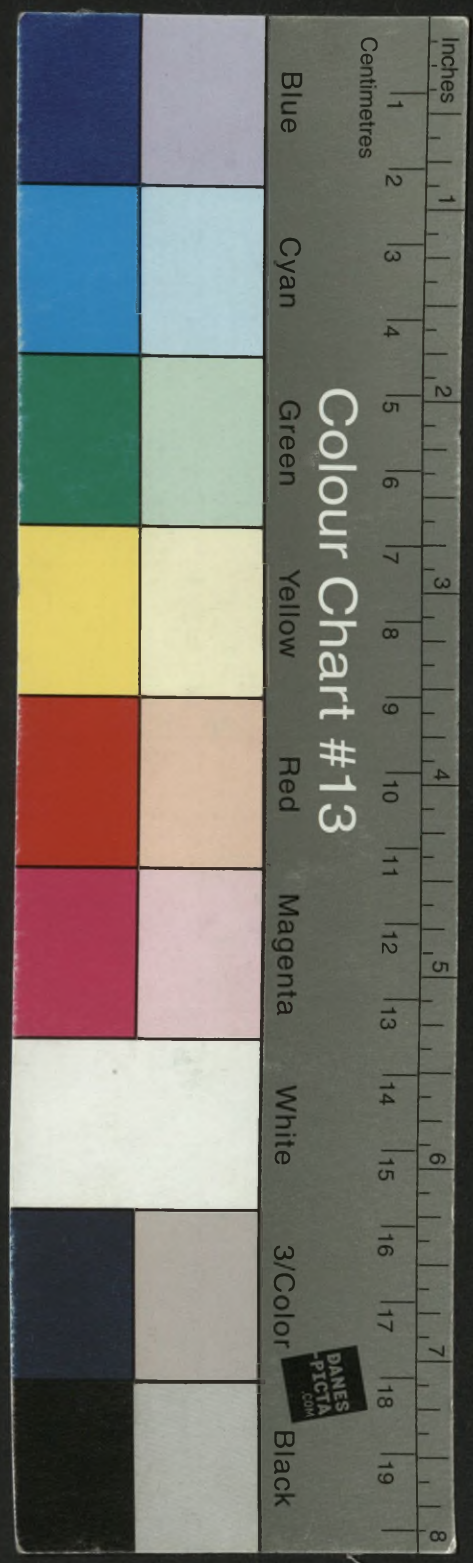
INFORMATYKA W DOWODZENIU

Podręcznik

Część I

PODSTAWY INFORMATYKI

WARSZAWA 1981



AKADEMIA SZTABU GENERALNEGO WP

KATEDRA DOWODZENIA

ASG WP wewn. 3633/81

INFORMATYKA W DOWODZENIU

Podręcznik

Część I

PODSTAWY INFORMATYKI

WARSZAWA 1981

AKADEMIA SZTABU GENERALNEGO WP

KATEDRA DOWODZENIA

ASG WP wewn. 3633/81 Egz.nr ...24

INFORMATYKA W DOWODZENIU

Podręcznik

Część I

PODSTAWY INFORMATYKI

WARSZAWA 1981

Opracował zespół w składzie:

płk mgr inż. Kazimierz ŁĘCKI /rozdział III/

ppłk mgr inż. Jan BEDNAROWSKI /rozdział II/

mjr dr inż. Andrzej BARCZAK /rozdział IV/

mjr dr hab.inż. Piotr SIENKIEWICZ /rozdział I i redakcja/

STR 37

S P I S T R E Ś C I

str.

176

W S T Ę P

Niniejsza praca przeznaczona jest przede wszystkim dla słuchaczy kursów dyplomowych Akademii Sztabu Generalnego WP aczkolwiek może być przydatna również dla słuchaczy Podyplomowego Studium Informatyki i Podyplomowego Studium Zastosowań Informatyki, a także słuchaczy wyższych szkół oficerskich. Napisano ją głównie na podstawie notatek do prowadzonych przez autorów wykładów z przedmiotu "Cybernetyczne podstawy dowodzenia" w latach 1975-1980. Zakres tego przedmiotu ulegał kilkakrotnie istotnym zmianom, wynikającym z rosnącego znaczenia zagadnień cybernetyki i informatyki wśród współczesnych problemów dowodzenia wojskami. Zasadniczym celem przedmiotu było zapoznanie z podstawami cybernetyki ogólnej i wojskowej oraz przygotowanie słuchaczy - przyszłych dowódców i oficerów sztabów - do działania w warunkach nowoczesnych systemów dowodzenia wykorzystujących systemy informatyczne. Osiągnięcie tego celu wymagało stworzenia niezbędnej bazy, której jednym z elementów ma być, w zamierzeniu autorów, niniejsza praca. Niezwykle szeroki zakres problematyki spowodował powstanie opracowania składającego się z dwóch oddzielnych części. Część pierwsza noszącą tytuł: "Podstawy informatyki" obejmuje podstawy automatyzacji systemów dowodzenia /rozdział 1/, elementy architektury współczesnych systemów liczących /rozdział 2/ i podstawy programowania maszyn cyfrowych /rozdział 3/, a także wybrane zagadnienia symulacji komputerowej ze szczególnym uwzględnieniem komputerowych gier wojennych /rozdział 4/. Natomiast część druga pt.: "Informatyczne zabezpieczenie działań bojowych" dotyczy systemów

informatycznych eksploatowanych aktualnie w siłach zbrojnych.

Głównym powodem powstania niniejszego opracowania był brak takiej pomocy dla słuchaczy, która odpowiadałaby programowi nauczania i pozwalałaby na efektywne przygotowanie wykładu i opanowanie podstawowych zagadnień. Należy przy tym podkreślić, że mimo obecnie dużej liczby oryginalnych polskich książek i znakomitych tłumaczeń, a także wielu świetnych skryptów uczelnianych poświęconych zagadnieniom cybernetyki, informatyki i badań systemowych nie ma dotychczas pracy obejmującej te z nich, które winny być omówione w ramach wykładów na temat zastosowań informatyki we współczesnych systemach dowodzenia.

Poszczególne rozdziały kończy wykaz literatury obejmujący te książki, które przede wszystkim są aktualnie dostępne. Ponadto zamieszczono zestaw pytań ułatwiających samodzielne studiowanie przedmiotu.

Mamy nadzieję, że niniejsze opracowanie ułatwi wykładowcom przygotowanie wykładu z informatyki wojskowej, zaś słuchaczom - studiowanie tego pożytecznego i niełatwego przedmiotu.

Autorzy pragną podziękować recenzentom, a szczególnie płk.dr. Janowi NOWAKOWSKIEMU za cenne uwagi, które pozwoliły na uniknięcie wielu usterek w tekście pracy.

Rozdział pierwszy

PODSTAWY AUTOMATYZACJI SYSTEMÓW DOWODZENIA

1. W p r o w a d z e n i e

Rewolucja naukowo-techniczna powoduje określone zmiany jakościowe nie tylko w technologii produkcji, energetyce, narzędziach i przedmiotach pracy, ale także w sferze organizacji i kierowania. To z kolei stanowi źródło coraz częściej pojawiającego się poglądu, że w dobie współczesnej - oprócz pracy, środków rzeczowych i nauki - coraz większego znaczenia nabiera organizacja i kierowanie, jako siła twórcza, harmonizująca, modyfikująca oraz zmieniająca funkcjonowanie pozostałych czynników. O poziomie ekonomicznego rozwoju społeczeństwa decyduje dziś w większym niż kiedykolwiek stopniu efektywność kierowania. Dlatego obok postępu naukowo-technicznego wyróżnia się także postęp organizacyjny, jako podstawowy element rewolucji naukowo-technicznej.

Współczesny postęp naukowo-techniczny i organizacyjny sprzyja szybkiemu rozwojowi ekonomiczno-społecznemu, stanowiąc tym samym jeden z głównych czynników umacniania siły obronnej państwa. Postęp naukowo-techniczny wywiera szczególny wpływ na jakościowy oraz ilościowy rozwój uzbrojenia i wojskowego sprzętu technicznego, co siłą rzeczy prowadzi do zmian w taktyce, sztuce operacyjnej, strategii i organizacji wojsk, jak również stymuluje postęp w systemach dowodzenia. Postęp ten obejmuje zarówno rozwój metod dowodzenia, organizacji i technologii pracy, jak i doskonalenie technicznego wyposażenia systemów dowodzenia na wszystkich szczeblach[x/]. Najwyższą natomiast formą postępu w systemach

---

x/ B.Bidziński, P.Sienkiewicz: Postęp naukowo-techniczny a siły zbrojne. ASG WP 1978.

dowodzenia jest optymalizacja procesów informacyjno-decyzyjnych, co wymaga stosowania odmiennych od tradycyjnego poprawiania i usprawniania wycinkowego, metod doskonalenia organizacji i dowodzenia wojskami.

Analizując różnice pomiędzy strukturą nauki "tradycyjnej" i nauki "nowoczesnej" można sformułować następujące wnioski[x]:

- w nauce "nowoczesnej" rzeczywistość traktowana jest w całości zamiast we fragmentach, co jest rezultatem nastawienia się na wszechstronne zaspokojenie potrzeb społecznych; powoduje to wzrost udziału problemów decyzyjnych /w stosunku do poznawczych/ wśród rozwiązywanych zagadnień naukowych;
- szczególnie preferowana jest w nauce "nowoczesnej" problematyka interdyscyplinarna /w ramach cybernetyki, prakseologii, systemologii/ oraz multidyscyplinarna /w zakresie współdziałania monodyscyplin/;
- w nauce "nowoczesnej" nastąpiło rozszerzenie problematyki abstrakcyjnej /w ramach matematyki i logiki matematycznej/.

We wszystkich tych zmianach widoczne jest dążenie do uogólniania problemów oraz rozszerzenia możliwości twórczych człowieka. Wiąże się z tym szczególne znaczenie rozwiązywania problemów interdyscyplinarnych, tj. tak ogólnych, że otrzymane wyniki mogłyby być wykorzystywane w wielu różnych monodyscyplinach. Ta idea doprowadziła do powstania nauki interdyscyplinarnej - cybernetyki. Jako datę powstania jej przyjmuje się na ogół rok 1948, w którym Norbert Wiener opublikował pracę pt. "Cybernetics or control and communication in the animal and the machine"[xx]. Wiele koncepcji

---

x/ M.Mazur: Cybernetyka i charakter. PIW, Warszawa 1975, s.7-15.

xx/ Istnieje polskie wydanie tej pracy: N.Wiener: "Cybernetyka czyli sterowanie i komunikacja w zwierzęciu i maszynie", PWN, Warszawa 1971.

cybernetycznych pojawiło się jednak znacznie wcześniej, w różnych zresztą okresach rozwoju nauki /np. w pracach Platona, Ampere'a, Trentowskiego, Kartezjusza, de La Mettrie i innych/.

Cybernetyka jest nauką o sterowaniu.

Sterowaniem określa się wszelkie celowe oddziaływanie jednego obiektu na drugi w celu uzyskania określonych zmian przebiegu procesów /osiągnięcia zamierzonych stanów końcowych/.

Sterowanie w obiektach społecznych określać będziemy jako kierowanie. Z kolei, kierowanie w systemach ekonomiczno-gospodarczych nazywać będziemy zarządzaniem, natomiast kierowanie w obiektach wojskowych - dowodzeniem.

Rozwój cybernetyki zrewolucjonizował sposób myślenia w wielu gałęziach nauki i techniki. Język cybernetyki, a w szczególności takie pojęcia jak: sterowanie, informacja, algorytm, system, model, kod, optymalizacja, sprzężenie zwrotne itp. znalazł szerokie zastosowanie. Cybernetyka badając konkretne zjawisko jakim jest sterowanie, wprowadziła metody traktowania rzeczywistości nadające się do stosowania w każdej monodyscyplinie. Towarzyszącym temu wkładem cybernetyki do nauki jest wprowadzenie ogólnej terminologii, umożliwiającej porozumienie między specjalistami z różnych monodyscyplin. Dzięki tym cechom cybernetyka wypełnia lukę między monodyscyplinami konkretnymi, z którymi łączy ją konkretność problematyki, a dyscyplinami abstrakcyjnymi, z którymi łączy ją ogólność problematyki[x/]. Szczególnie mocno należy podkreślić fakt, że konkretny przedmiot cybernetyki - sterowanie - jest z punktu widzenia tej dyscypliny tym samym zjawiskiem bez

---

x/ M.Mazur, wyd.cyt.

względu na obiekt rzeczywisty, w którym ono zachodzi. Sterowanie jest więc procesem, który jest realizowany zarówno w obiektach biologicznych /organizmach/ i społecznych /grupach, zespołach, społeczeństwach/ jak i w obiektach technicznych /automatach/. Cybernetyka dąży zatem do opracowania takich metod i takich modeli /w szczególności modeli matematycznych/, które nadawałyby się do badania /analizy, oceny, syntezy/ sterowania w różnych jego odmianach i w różnych obiektach sterowania.

Jednakże bezpośrednim bodźcem do stworzenia cybernetyki były konkretne zadania wysuwane przez praktykę, a przede wszystkim potrzeby skonstruowania uniwersalnych maszyn liczących, pozwalających na dokonywanie żmudnych, pracochłonnych obliczeń w krótkim, zadowalającym człowieka czasie oraz takich urządzeń automatycznych, które mogłyby wykonywać pewne skomplikowane czynności bez bezpośredniego udziału człowieka /np. przeliczniki artyleryjskie/. Niewątpliwą stymulacyjną rolę w tym zakresie odgrywały i odgrywają nadal wojskowe badania naukowo-techniczne.

Od powstania pierwszej elektronicznej maszyny cyfrowej do współczesnych systemów informatycznych upłynęło niespełna czterdzieści lat. Zaszły w tym okresie nieporównywalne wręcz zmiany w technologii procesów informacyjnych, architekturze i organizacji systemów liczących, metodach programowania i sposobach użytkowania komputerów. Jedna ze zmian polegała właściwie na tym, że pewne koncepcje cybernetyczne rozpoczęły w latach sześćdziesiątych "samodzielny żywot" w postaci nowej dziedziny wiedzy i praktyki zwanej informatyką. W uchwale Rady Ministrów PRL z 12 lutego 1971 r. możemy znaleźć definicję informatyki jako "całokształtu prac nad zbieraniem, przechowywaniem i przetwarzaniem zakodowanej informacji, opracowywaniem i wykorzystaniem w gospodarce narodowej

automatycznego przetwarzania danych oraz zapewnieniem potrzebnych do tego środków technicznych". Zwrócić należy uwagę na to, że informatyka stała się problemem o randze państwowej, dostrzeżono jej rolę jako czynnika, który odtąd wywierać będzie coraz większy wpływ na rozwój społeczno-gospodarczy kraju. Ponadto, przytoczona definicja uwypukla fakt, że głównym przedmiotem informatyki jest automatyzacja działań na informacji, i to automatyzacja kompleksowa, obejmująca wszystkie aspekty działalności związanej z obiegiem informacji w społeczeństwie. Koncentrując uwagę na stronie poznawczej informatyki można przyjąć inną jeszcze definicję:x/

> Informatyka jest dziedziną wiedzy obejmującą dyscypliny nauki i techniki związane ze zbieraniem i przechowywaniem informacji, ich przetwarzaniem oraz sposobami ich reprezentowania, jak również z budową maszyn, urządzeń i systemów służących do powyższych celów.

Informatyka posługuje się dorobkiem tworzonym w takich grupach nauk, jak: formalne, inżynierskie /techniczne/, społeczne i wojskowe. Dodać należy także, że informatyka stała się dzisiaj, po energetyce i motoryzacji, trzecią branżą świata.

Informatyka jest także jednym z podstawowych narzędzi doskonalenia systemów dowodzenia na wszystkich szczeblach organizacyjnych oraz we wszystkich rodzajach wojsk i służb. Wynika to ze szczególnych wymagań współczesnego pola walki dotyczących czasu obiegu informacji w systemach dowodzenia, oceny sytuacji i podejmowanie decyzji. Skrócenie czasu na planowanie i przekazanie zadań bojowych wymaga wykonania wszystkich możliwych prac

---

x/ J.Maroński, M.Muraszkiewicz, Z.Nowicki: Wprowadzenie do informatyki. PWN, Warszawa 1975.

w tym zakresie jeszcze przed otrzymaniem zarządzenia wstępnego i zadania bojowego na podstawie przewidywań. Jest to możliwe przede wszystkim w warunkach efektywnego funkcjonowania zautomatyzowanych systemów dowodzenia, które powinny spełnić wiele wymagań wynikających z dynamiki współczesnych działań bojowych. Nowoczesne zautomatyzowane systemy dowodzenia powinna m.in. charakteryzować nieprzerwana praca przez co najmniej 1-2 doby w zależności od szczebla, zaś czas przerw na przywracanie zdolności technicznych nie powinien przekraczać kilkudziesięciu minut. Zautomatyzowane systemy dowodzenia powinny ponadto zapewniać operatywność i ciągłość dowodzenia zarówno na postoju jak i w ruchu oraz zapewnić zbieranie, przesyłanie, przechowywanie, przetwarzanie i wydawanie informacji o kilku tysiącach charakterystycznych obiektów. Czas przekazywania informacji powinien nie przekraczać wartości od kilku do kilkudziesięciu minut /w zależności od szczebla i kategorii pilności wiadomości/.
Powyższe wymagania taktyczno-techniczne powodują konieczność ciągłego doskonalenia technicznych i programowych środków informatyki i telekomunikacji. Specyficzne wymagania współczesnego dowodzenia stymulują rozwój informatyki wojskowej.

Informatyka wojskowa obejmuje obszar zastosowań informatyki związany z rozwojem procesów zbierania, przechowywania, przetwarzania i reprezentowania informacji operacyjno-taktycznych, jak również z budową i eksploatacją systemów informatycznych zaspokajających potrzeby informacyjne organów dowodzenia i spełniających wymagania współczesnego dowodzenia w zakresie oceny sytuacji i podejmowania decyzji.

Traktując informatykę wojskową w sposób instrumentalny wyróżnia się jako jeden z rodzajów zabezpieczenia działań bojowych informatyczne zabezpieczenie. Podkreślmy, że "głównym celem automaty-

zacji jest zwiększenie operatywności kierowania wojskami, skuteczności użycia broni, sprzętu wojskowego i w ogóle gotowości bojowej wojsk, a także poważne zmniejszenie wysiłku dowódców i oficerów sztabu na prace techniczne"[x].

2. I n f o r m a c j a w d o w o d z e n i u.

Pojęcie informacji stało się pojęciem ogólnonaukowym, przestając być tradycyjnym przedmiotem badań teorii informacji i telekomunikacji. Jest wykorzystywane w wielu bardzo różnych dziedzinach. W ostatnich kilkudziesięciu latach informacja przechodziła wiele interpretacyjnych metamorfoz, z których pierwszą było przejście od pojęcia wiadomości i komunikatu do pojęcia ilości informacji i związania go z prawdopodobieństwem określonego zdarzenia ze zbioru zdarzeń. Samo pojęcie informacji nie doczekało się dotąd powszechnie akceptowanej definicji. Do najpopularniejszych zaś należą: "Informacja jest informacją, a nie materią ani energią" /N.Wiener/ oraz "Informacja jest nazwą treści zaczerpniętej ze świata zewnętrznego, w miarę jak się do niego dostosowujemy i jak przystosowujemy doń swoje zmysły" /N.Wiener/, a ponadto "W cybernetyce nazywa się informacją wszelkie działania fizyczne, któremu towarzyszy działanie psychiczne" /L.Couffignal/[xx] oraz "Informacją są wszelkie wiadomości o procesach i stanach dowolnej natury, które mogą być odbierane przez organy zmysłowe człowieka lub przez przyrodę" /W.Głuszkow/, a także "Informacją nazywamy wielkość abstrakcyjną, która może być przechowywana w pewnych obiektach, przesyłana między pewnymi obiektami, przetwarzana w pewnych obiektach i stosowana do sterowania pewnymi

---

x/ A.Greczko: Siły zbrojne państwa radzieckiego. MON, Warszawa 1975.

xx/ M.Mazur: Jakościowa teoria informacji. WNT, Warszawa 1970.

obiektami, przy czym przez obiekty rozumie się organizmy żywe, urządzenia techniczne oraz systemy tych obiektów" /A.Mazurkiewicz/. Cybernetyczne ujęcie pojęcia informacji wskazuje na nowy aspekt materialnej jedności świata i pozwala na podejście w sposób jednolity do procesów informacyjnych zachodzących w organizmach biologicznych, społecznych i w urządzeniach technicznych[x/].

Uogólniając różne ujęcia zagadnienia natury informacji możemy powiedzieć, że komunikatem nazywamy odpowiednio zakodowaną wiadomość, zawierającą pewną ilość informacji[xx/]. W określeniu tym komunikat traktuje się jako samoistny obiekt fizyczny /tekst pisany, modulowana fala elektromagnetyczna/, a wiadomość jako relację zachodzącą między nadawcą i odbiorcą. Dla celów praktycznych można przyjąć następujące określenia:

Informacje to zbiór faktów, zdarzeń, cech obiektów itp. zawarty w określonej wiadomości, tak ujęty i podany w takiej formie, że pozwala odbiorcy ustosunkować się do zaistniałej sytuacji i podjąć odpowiednie działania umysłowe lub fizyczne.

Dane to zbiór faktów, zdarzeń, cech obiektów itp. przedstawiony w postaci stanowiącej podstawę obliczeń lub manipulacji symbolami, czyli procesu ich przetwarzania.

Podstawowe założenie ilościowej teorii informacji[xxx/] polega na tym, że komunikat zawiera tym więcej informacji, im mniejsze jest prawdopodobieństwo jego wystąpienia. Oznacza to, że ilość

---

x/ Zob. "Mały słownik cybernetyczny", Wiedza Powszechna, Warszawa 1973.

xx/ W.Turski: Propedeutyka informatyki. PWN, Warszawa 1975.

xxx/ Twórcą ilościowej teorii informacji jest C.Shannon, który w latach 1948-1949 ogłosił swoje fundamentalne prace w tej dziedzinie /np. "The Mathematical Theory of Communication", 1949/.

informacji jest tym większa, im większa była niepewność, że określoną wiadomość możemy znaleźć w komunikacie. Otrzymać informację - znaczy dowiedzieć się czegoś, czego nie wiedziało się przedtem, lub dowiedzieć się czegoś więcej o tym, o czym wiedziało się mniej. Z powyższych założeń wynika, że informacja związana jest z nieokreślonością danej sytuacji. Nieokreśloność dowolnej sytuacji może być opisana przy pomocy wielkości zwanej entropia. Ogólnie można powiedzieć, że entropia określa stopień nieokreśloności i nieoczekiwalności stanu. Jeżeli daną sytuację charakteryzuje wielkość X, która może przyjmować wartości $x_1, \ldots, x_n$ z odpowiednim prawdopodobieństwem $p_1, \ldots, p_n$, przy czym $\sum_{i=1}^{n} p_i = 1$, to entropię można określić również jako średnią ilość informacji przypadającą na jedno zdarzenie lub wartość oczekiwaną ilości informacji I w mierzonej wielkości X:

$$H/X/ = E[I/X/] = -\sum_{i=1}^{n} p_i \log p_i$$

Entropia charakteryzuje się następującymi właściwościami:

a/ entropia jest zawsze nieujemna, ponieważ wartości prawdopodobieństw należą do przedziału 0,1 , a więc ich logarytmy są liczbami niedodatnimi;

b/ entropia jest równa zeru tylko w takim krańcowym przypadku, gdy prawdopodobieństwo jednego zdarzenia równa się zero;

c/ entropia osiąga wartość maksymalną w przypadku, gdy prawdopodobieństwa wszystkich zdarzeń są równe.

W ogólnym przypadku ilość informacji wraz ze stopniem zmniejszenia entropii jest określona w drodze doświadczenia lub innego aktu poznawczego.

Jeżeli nieokreśloność znika całkowicie, to uzyskana informacja jest równa entropii: I = H. W przypadku częściowego rozeznania sytuacji uzyskuje się część możliwej informacji, będącą różnicą

między entropią początkową H_o i końcową H_k:

$$I = H_o - H_k$$

Największa ilość informacji uzyskuje się wówczas, gdy nieokreśloność znika, gdy jednocześnie ona była największa /prawdopodobieństwa wszystkich zdarzeń były jednakowe/.

Jako konwencjonalną jednostkę ilości informacji przyjęto alternatywę dwu jednakowo prawdopodobnych i niezależnych symboli, dla których $p_1 = p_2 = 0,5$. Wówczas, posługując się wzorem na przeciętną ilość informacji otrzymamy

$$H = -\sum_{i=1}^{2} p_i \log p_i = \log 2$$

Jeżeli przyjmiemy, że H = 1, to otrzymamy log 2 = 1, co jest spełnione dla logarytmów o podstawie 2. Jednostkę tę nazwano bitem. W przypadku posługiwania się logarytmami naturalnymi jednostki informacji noszą nazwę nit, natomiast w przypadku posługiwania się logarytmami dziesiętnymi jednostką informacji jest dit.

Z cybernetycznego punktu widzenia dowodzenie jest procesem antyentropijnym, gdyż entropia systemu pozbawionego sterowania wzrasta. Zapobiec temu może tylko sterowanie, które w istocie jest "walką z nieuporządkowaniem /nieokreślonością/". Miarą tego nieuporządkowania może być ilość informacji sterującej. Stwierdzić ponadto można, że:

- krytycznym mankamentem dowodzenia jest przeważnie działanie bez dostatecznej ilości istotnych informacji,
- informacje wymagane przez decydentów są im rzeczywiście potrzebne,

- jeśli decydent otrzyma potrzebne informacje, wówczas jego decyzje staną się trafniejsze, a zatem wyższa będzie efektywność działania,
- sprawniejsze informowanie w systemie dowodzenia oznaczać może wyższą efektywność działania.

Obiekty dowodzenia ../walczące wojska/ pod wpływem zakłóceń losowych i innych oddziaływań zmieniają swoje charakterystyki i dążą do różnorodnych sposobów zachowania się, przez co wzrasta stopień ich nieuporządkowania.

Przyjmijmy, że w obiekcie liczba możliwych sposobów zachowania się jest skończona, zatem nie może być uzyskana maksymalna wartość jego nieporządkowania, stąd entropia obiektu $H < H_{max}$. Oddziaływania systemu dowodzenia zmniejszają nieuporządkowanie lecz całkowicie go nie eliminują, czyli $0 < H < H_{max}$. Na początku w obiekcie znajduje się pewna ilość informacji I_o oraz nieuporządkowanie N_o i entropia H_o. W chwili t wprowadza się dodatkową ilość informacji $\Delta I/t/$, dzięki czemu zmniejszą się nieuporządkowanie i entropia do wartości N/t/ i H/t/. Ilość informacji $\Delta I/t/$ określa zależność:

$$\Delta I/t/ = \alpha \ln \frac{N_o}{N/t/}$$

zaś nieuporządkowanie obiektu

$$N/t/ = N_o e^{-\frac{\Delta I/t/}{\alpha}}$$

gdzie α - stała charakteryzująca obiekt.

Likwidacja nieuporządkowania powinna być zatem źródłem efektywności, którą dla obiektu można określić następująco:

$$E/t/ = E_{max}\left[1 - N_o e^{-\frac{\Delta I/t/}{\alpha}}\right]$$

gdzie: E_{max} - maksymalna efektywność idealnie zorganizowanego systemu.

Dowodzenie jako proces informacyjny obniża nieuporządkowanie i entropię w systemie, co stanowi warunek konieczny wzrostu efektywności działania.

Oprócz ilościowego aspektu informacji równie istotne znaczenie mają inne aspekty, a mianowicie:

- aspekt semantyczny związany z treścią, zawartością wiadomości, z efektem psychicznym informacji;
- aspekt pragmatyczny związany z wartością informacji /z jej użytecznością dla działania odbiorcy/, z wpływem danej informacji na sposób zachowania się jej użytkownika.

Oczywiście w dowodzeniu najistotniejszą rolę spełnia ostatni z wymienionych aspektów, czyli w procesie dowodzenia niezbędne jest dysponowanie przez organa dowodzenia informacjami o wymaganej wartości.

Przyjmujemy, że wartość informacji zawartej w danej wiadomości kształtują dwie podstawowe cechy: użyteczność i jakość. Informacja, która niewiedzę decydenta w procesie dowodzenia częściowo lub całkowicie redukuje, jest - z punktu widzenia decydenta - pożądana. Jej ważność i użyteczność dla decydenta zależy:

- od problemu decyzji, gdyż z pewnością mniej cenna jest informacja wówczas, gdy przy każdym stanie rzeczy skutki działań niewiele się między sobą różnią co do wartości, niż gdy są one pod tym względem znacznie zróżnicowane;
- od jakości informacji, ponieważ cenniejsza jest niewątpliwie informacja pełna niż niepełna, dokładna niż niedokładna, aktualna niż nieaktualna.

Zatem o informacji powiemy, że jest użyteczna, gdy zwiększa trafność powziętej decyzji.

Jakość informacji zależy natomiast od stopnia jej aktualności /terminowości/, dokładności /poprawności/ i pełności /kompletności/.

Należy podkreślić, że aktualność i dokładność informacji zależy od takich charakterystyk systemów informatycznych, jak: konfiguracja sieci przesyłania informacji, charakterystyki kanałów, niezawodności łączy, obciążenie sieci, stosowany kod i reguła decyzyjna w odbiorniku itp.

Jedną z pierwszych prób wyrażenia użyteczności informacji było ujęcie A.Charkiewicza:

$$U/I/ = \log_2 p_1 - \log_2 p_0 = \log_2 \frac{p_1}{p_0}$$

gdzie: p_0 - prawdopodobieństwo osiągnięcia celu przed otrzymaniem danej informacji;

p_1 - prawdopodobieństwo osiągnięcia celu po otrzymaniu danej informacji.

Jak widać z powyższego wzoru, użyteczność informacji wyrażana jest, podobnie jak jej ilość, w bitach.

Przykład[x/]

Załóżmy, że prowadząc kolumnę wojskową z punktu A do C, dochodzimy do rozwidlenia trzech dróg d_1, d_2, d_3 w punkcie B. Wiemy, że tylko jedna z tych dróg prowadzi do C, lecz nie wiemy która.

a/ W punkcie B na rozwidleniu napotykamy patrol jadący drogą d_1 w kierunku A. Patrol ten informuje nas, że d_1 nie prowadzi do C. Użyteczność tej informacji wyniesie

$$U/I/ = \log_2 \frac{\frac{1}{2}}{\frac{1}{3}} = \log_2 \frac{3}{2} = 0{,}58 \text{ bita}$$

x/ Zob. L.Kuleszyński: Dowodzenie wojskami a cybernetyka. MON, Warszawa 1967, s. 53-54.

b/ Gdyby patrol poinformował, że do C prowadzi np. droga d_2, to wówczas użyteczność tej informacji wyniosłaby

$$U/I/ = \log_2 \frac{1}{\frac{1}{3}} = \log_2 3 = 1{,}58 \text{ bita}$$

Użyteczność informacji w drugim przypadku byłaby większa o 1 bit.

M.Mazur[x/] posługuje się pojęciem informowania określającym transformację informacji zawartej w oryginałach w informacje zawarte w obrazach. Ze względu na relacje między oryginałami a obrazami wyróżnia się charakterystyczne przypadki informowania:

a/ transinformowanie, czyli informowanie wierne, które jest zapewnione, gdy:

- oryginały są zarazem obrazami /np. dokument nadany i ten sam dokument odebrany/;
- oryginały są takie same, jak obrazy /np. dokument i jego kopia/;
- oryginały są zniekształcone w komunikaty pośrednie, które następnie są odwrotnie zniekształcone w obrazy /np. zaszyfrowanie tekstu i następnie jego odszyfrowanie/.

b/ dezinformowanie, czyli informowanie fałszywe obejmujące takie przypadki, jak:

- zmyślanie /gdy obrazy nie są wynikiem transformacji żadnego oryginału/;
- zatajanie /gdy oryginały nie są transformowane w żaden obraz/;
- przekręcanie /gdy pewne obrazy nie są wynikiem transformacji żadnego oryginału, a pewne oryginały nie są transformowane w żaden obraz/[xx/].

---

x/ Zob. M.Mazur: Jakościowa teoria informacji. WNT, Warszawa 1970.
M.Mazur: Cybernetyka i charakter. PIW, Warszawa 1975.

xx/ W świetle powyższego fałszem jest dezinformowanie, a kłamstwem przedstawienie tego dezinformowania jako transinformowania.

c/ parainformowanie, czyli tzw. domniemywanie obejmujące przypadki:

- paratransinformowanie, czyli domniemywanie trafne /np. zrozumienie aluzji/;
- paradezinformowanie, czyli domniemywanie nietrafne, a w tym: domniemywanie bezpodstawne /np. dopatrzenie się aluzji, której nie było/, domniemywanie niedomyślne /np. niedopatrzenie się aluzji, która była/, domniemywanie opaczne /np. dopatrzenie się aluzji innej niż była/.

d/ pseudoinformowanie, czyli informowanie pozorne, które może być:

- rozwlekłe /w przypadku transformacji jednego oryginału w kilka obrazów/,
- ogólnikowe /w przypadku transmisji kilku oryginałów w jeden obraz/,
- niejasne /w przypadku transformacji jednego oryginału w kilka obrazów, z których jeden jest wynikiem transformacji kilku oryginałów/.

Od projektowanych systemów informacyjnych dla potrzeb dowodzenia, najogólniej, wymaga się, aby zapewniały transinformowanie w systemie dowodzenia. Znajomość natury informacji jest konieczna dla zrozumienia specyfiki procesów dowodzenia oraz roli jaką we współczesnych systemach kierowania spełniają systemy informatyczne.

3. A n a l i z a s y s t e m o w a

Jako jedną z cech współczesnych badań naukowych i rozwojowych wymienia się tzw. "podejście systemowe". Oznacza to w zasadzie przyjęcie paradygmatu[x/] systemowego jako obowiązującego we współczesnych systemach badań i rozwoju. Wyraża on pewien specyficzny

---

x/ Paradygmatem określa się model, wzorzec postępowania naukowego obowiązującego na danym etapie rozwoju nauki.

sposób obserwacji otaczającej rzeczywistości, polegający na dostrzeganiu jej w postaci złożonych zintegrowanych obiektów. Jako podstawowe cechy paradygmatu systemowego przyjmuje się[x/]:

- opis i wyjaśnianie rzeczywistości traktowanej całościowo za pomocą specyficznego języka systemowego;
- traktowanie badanych obiektów jako systemów;
- traktowanie danego systemu jako należącego do systemu większego /tzw. nadsystemu/;
- traktowanie danego systemu jako obiektu złożonego z elementów /tzw. podsystemów/;
- rozpatrywanie rzeczywistych systemów w kategoriach struktury, działania /funkcjonowania/ i rozwoju;
- świadome posługiwanie się modelem /logicznym, matematycznym/ systemu w rozwiązywaniu problemów praktycznych /decyzyjnych/;
- racjonalne optymalizowanie rzeczywistych systemów metodami matematycznymi.

Systemem określamy obiekt złożony, wyróżniony z badanej rzeczywistości, przedstawiany jako całość oraz tworzony przez zbiór elementów /obiektów elementarnych/ i powiązań /relacji/ pomiędzy nimi.

Systemem nazywamy całość, którą tworzą: zbiór elementów M wraz z powiązaniami między nimi R, czyli

$$S = \langle M, R \rangle$$

System określony jest przez:

- zbiór elementów M oraz
- zbiór relacji R pomiędzy tymi elementami, np. $R \subset M \times M$.

---

x/ Zob. P.Sienkiewicz: Teoria efektywności systemów kierowania. T.1 - Wstęp do systemologii. ASG WP 1979.

Zbiór relacji R ma następujące podstawowe własności[x]:
dla dowolnych niepustych zbiorów M' i M'' takich, że $M' \cup M'' = M$ oraz $M' \cap M'' = \emptyset$ istnieją takie elementy $m' \in M'$ i $m'' \in M''$, że prawdziwe jest co najmniej jedno stwierdzenie:

$$\langle m', m'' \rangle \in R \quad \text{albo} \quad \langle m'', m' \rangle \in R.$$

Zapis $\langle m', m'' \rangle \in R$ równoważny jest zapisowi $m' R m''$, który oznacza tyle, co "między elementem m' i elementem m'' występuje relacja, lub elementy m' i m'' są w relacji".

Przykład

Załóżmy, że system tworzą trzy elementy m_1, m_2, m_3; elementy m_1 i m_2 są w relacji R_1, elementy m_2 i m_3 w relacji R_2 oraz elementy m_1 i m_3 w relacji R_3. Wtedy system ten możemy przedstawić w postaci ogólnego modelu:

$$S = \langle M, R \rangle$$

gdzie: $M = \{m_1, m_2, m_3\}$, $R = \{R_1, R_2, R_3\}$, $m_1 R_1 m_2$, $m_2 R_2 m_3$, $m_1 R_3 m_3$.

Stosowanie pojęcia systemu powinno być podporządkowane pewnym regułom, a mianowicie:

a/ ścisłość - system powinien być ściśle określony, aby było wiadomo, co do niego należy, a co nie należy;

b/ niezmienność - określenie systemu powinno być niezmienne w całym toku rozważań;

c/ zupełność - podział systemu na podsystemy powinien być zupełny, czyli że system nie może zawierać elementów nie należących do żadnego z jego podsystemów;

---

x/ Przypomnijmy, że symbol "U" oznacza sumę zbiorów, "∩" - iloczyn zbiorów, "Ø" - zbiór pusty.

d/ rozłączność - podział systemu na podsystemy powinien być rozłączny, czyli że system nie może zawierać elementów należących do kilku podsystemów naraz;

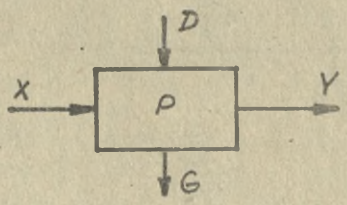
e/ funkcjonalność - system powinien być wyodrębniony z badanej rzeczywistości ze względu na spełniane funkcje, cechy gatunkowe itp.

Przykład

Rozpatrzmy tzw. system ogólny Mesarovića[x/], dla którego określono dwa zbiory: wejścia /bodźce/ X i wyjścia /reakcje/ Y. W ujęciu Mesarovića system jest relacją między wejściami i wyjściami, czyli

$$S \subset X \times Y$$

Załóżmy, że oprócz X i Y dane są następujące obiekty: zbiór dopuszczalnych decyzji D oraz zbiór dopuszczalnych wartości efektów funkcjonowania systemu V.



Ponadto, dane są dwie funkcje:

- funkcja procesu

$$P: X \times D \longrightarrow Y$$

- funkcja efektów

$$G: D \times Y \longrightarrow V$$

---

x/ Np. M.Mesarović: Rozwój matematycznej teorii systemów ogólnych. W: Prakseologia, Warszawa 1973 Nr 2 /46/.

Jeżeli celem działania jest maksymalizacja efektów, zaś funkcja G posiada maksimum, to wtedy można sformułować następujący wniosek:

dla każdego $x \in X$ i $y \in Y$, $/x,y/ \in S$, czyli x i y tworzą system S, wtedy i tylko wtedy, gdy istnieje $d^* \in D$ takie, że dla każdego $d \in D$ spełnione są warunki

$$G/d^*, P/x,d^*// \geqslant G/d, P/x,d//$$

oraz $$y = P/x,d^*/.$$

W ten sposób określono ścisłe warunki istnienia systemu celowego[x/].

Wśród dziedzin wiedzy systemowej coraz większą rolę w badaniach procesów społeczno-gospodarczego rozwoju oraz procesów rozwoju sił zbrojnych odgrywa <u>analiza systemowa.</u>

<u>Analiza systemowa</u> jest metodą /zbiorem technik i procedur/ racjonalnego formułowania i rozwiązywania systemowych problemów decyzyjnych; jest sposobem badania systemów rzeczywistych prowadzącym do wzrostu efektywności dzięki zmianom strukturalnym i funkcjonalnym w systemie.

Na podstawie dotychczasowych doświadczeń można stwierdzić, że jedną z głównych zalet analizy systemowej jest właśnie to, że w oparciu o badanie modeli pozwala ona określić zależności pomiędzy przyczynami i skutkami w procesie funkcjonowania wielu rzeczywistych systemów, a także przewidywać ich zachowanie się pod wpływem różnych czynników.

---

x/ W zasadzie każdy system może być opisany jako układ wejście-wyjście lub też jako system celowy. W ten sposób staje się możliwe pogodzenie wielu sprzecznych koncepcji metodologicznych, takich jak np. behawioryzm Skinnera, czy strukturalizm Chomsky'ego.

Analiza systemowa obejmuje najczęściej takie fazy, jak[x]:

- analiza i sformułowanie problemu /ustalanie celów/,
- analiza funkcji /ustalenie zadań i czynności systemu - przedmiotu analizy/,
- określenie alternatywnych rozwiązań /wariantów decyzji/,
- identyfikacja systemów oddziałujących z systemem analizowanym,
- opracowanie matematycznego modelu systemu,
- zebranie i opracowanie danych ilościowych o działaniu systemu,
- eksperymentowanie z modelem,
- określenie nakładów /kosztów/, efektów systemu,
- rozwiązanie problemów związanych z niepewnością i ryzykiem w działaniu systemu,
- wybór wariantu najkorzystniejszego /optymalnego/.

W procesie analizy systemowej mamy do czynienia z problemami decyzyjnymi powstającymi w trakcie podejmowania decyzji /świadomego wyboru/. Są to zarówno problemy tzw. dobrze zestrukturalizowane, czyli związane ze znanymi dobrze decydentom sytuacjami opisywanymi za pomocą skończonej liczby elementów i ilościowych zależności pomiędzy nimi, jak i problemy "słabo zestruktualizowane" i "niezestruktualizowane". Te ostatnie charakteryzuje niepowtarzalność, bardzo duża liczba elementów, pomiędzy którymi występują również zależności o charakterze jakościowym. Dla problemów konstruowane są matematyczne modele decyzyjne, które stanowią określone uproszczenie rzeczywistej sytuacji.

---

x/ Według RAND Corporation, instytucji badawczo-rozwojowej pracującej na rzecz departamentu obrony USA, gdzie analiza systemowa powstała w latach czterdziestych /jeszcze jako "badania operacji"/ i gdzie została następnie rozwinięta.

Najogólniej matematyczny model decyzyjny tworzą następujące elementy:

- dziedzina modelu, czyli zbiór cech /elementów/ istotnych dla problemu;
- relacje modelu, czyli zbiór relacji /powiązań/ pomiędzy cechami /elementami/ tworzącymi dziedzinę modelu;
- założenia modelu, czyli zbiór postulatów upraszczających wyrażających przyjęte w modelu ograniczenia;
- kryterium modelu, czyli funkcja /funkcje/ wyrażająca cele działania systemu /istotę problemu/, pozwalająca na ocenę wariantów rozwiązania problemów;
- pytanie, określające cel badań i przeznaczenie modelu w aspekcie rzeczywistych potrzeb.

W teorii decyzji[x/] korzysta się z ogólnego modelu sytuacji decyzyjnej, który przedstawia się w następującej postaci:

$$\langle A, S, u \rangle$$

gdzie: $A = \{a_1, a_2, \ldots, a_n\}$ jest skończonym zbiorem alternatyw /możliwych działań/, z którego decydent wybiera jedną z nich zgodnie z własnymi celami.

$S = \{S_1, S_2, \ldots, S_m\}$ jest skończonym zbiorem hipotez o stanie rzeczy determinującym wyniki poszczególnych alternatyw /na ogół decydent nie może przewidzieć z całą pewnością, która z odnośnych hipotez okaże się prawdziwa/.

U jest funkcją rzeczywistą wyników /konsekwencji/, która jest określona na iloczynie kartezjańskim $A \times S$ w ten sposób, że $u/a_i, s_j/ = u_{ij}$ jest wynikiem, który decydent otrzymuje gdy wybiera działanie a_i i hipoteza s_j okaże się prawdziwa.

---

x/ Np. J.Kozielecki: Psychologiczna teoria decyzji. PWN, Warszawa 1975, 1978.

Przykład

Rozpatrzmy charakterystyczne postępowanie decydenta w określonej sytuacji decyzyjnej $\langle A, S, U \rangle$.

a/ Powiemy, że decydent zastosował kryterium maxmin Walda, jeżeli działanie a_r jest optymalne zawsze i tylko, gdy

$$\min_j U_{rj} = \max_i \min_j U_{ij}$$

W myśl powyższego kryterium zaleca się, aby maksymalizować minimalną korzyść możliwą do osiągnięcia za pomocą danego działania.

b/ Powiemy, że decydent zastosował kryterium Hurwicza, jeżeli działanie a_r jest optymalne zawsze i tylko, gdy

$$\alpha \min_j U_{rj} + /1 - \alpha/ \max_j U_{rj} = \max_i \left[\alpha \min_j U_{ij} + /1 - \alpha/ \max_j U_{ij} \right]$$

W myśl tego kryterium zaleca się, aby maksymalizować sumę ważoną najmniejszej i największej korzyści osiągalnej przy podjęciu danego działania. Współczynnik α określa się jako współczynnik pesymizmu /$0 \leq \alpha \leq 1$/, czyli im większe α, w tym większym stopniu decydent liczy się z ewentualnością najgorszą[x/].

Powyższy przykład ilustrował taki model decyzyjny, w którym rozpatruje się wpływ postępowania decydenta na wybór działania najkorzystniejszego. Wpływu tego nie uwzględnia się w popularnych /m.in. dzięki badaniom operacyjnym/ modelach optymalizacyjnych. Ogólnie powiemy, że problem optymalizacji polega na podaniu zbioru X i pewnej funkcji F określonej na tym zbiorze oraz na poszukiwaniu takiego x^*, że

$$x^* \in X$$
$$\bigwedge_{x \in X} \; F/x/ \leq F/x^*/$$

x/ Oprócz kryteriów Walda i Hurwicza istnieje jeszcze kilka innych takich, jak np. kryterium Laplace'a, Savage'a itp.

Problem ten możemy przedstawić w następującej ścisłej postaci matematycznej

dane będą funkcje

$$F : R^n \longrightarrow R^1$$

oraz $$G_i : R^n \longrightarrow R^1, \qquad i = 1, \ldots, m$$

gdzie R^n oznacza n-wymiarową przestrzeń liczb rzeczywistych /n = 1,2,.../. Jeżeli problem sprowadzimy do poszukiwania ekstremum warunkowego funkcji F, to otrzymujemy tzw. <u>zadanie programowania matematycznego.</u>

Zadanie programowania matematycznego polega na znalezieniu n-wymiarowego wektora x należącego do zbioru

$$X_o = \left\{ x : G_i/x/ \leq 0, \quad i = 1, \ldots, m \right\}$$

takiego, że dla każdego $x \in X_o$

$$F/x^*/ \leq F/x/$$

co jest równoważne poszukiwaniu[x/]

$$\min_{x \in X_o} F/x/$$

przy czym: F nazywamy funkcją celu /funkcją kryterialną, funkcją efektywności, jakości itp./, G_i nazywamy funkcją ograniczeń, X_o nazywamy zbiorem rozwiązań dopuszczalnych.

Ze względu na charakter funkcji F i G_i wyróżnia się pewne charakterystyczne przypadki:

a/ jeżeli funkcje F i G_i są funkcjami liniowymi, to otrzymujemy <u>zadanie programowania linicwego:</u>

---

x/ Zadanie to można sprowadzić także do zadania maksymalizacji, gdyż

$$\max F/x/ = -\min \left[-F/x/ \right]$$

b/ jeżeli którakolwiek chociaż z funkcji F i G_i jest funkcją nieliniową /wypukłą, wklęsłą/, to otrzymujemy zadanie programowania nieliniowego;

c/ jeżeli którakolwiek chociaż z funkcji F i G_i jest funkcją losową, to otrzymujemy zadanie programowania stochastycznego.

Ponadto, jeżeli zbiór X_o ma przeliczalną lub skończoną liczbę elementów, to mówimy o problemie dyskretnym, a w przeciwnym wypadku o problemie ciągłym.

Rozwiązanie x^* problemu decyzyjnego nazywamy rozwiązaniem optymalnym, jeżeli dla każdego $x^* \in X_o$

$$F/x^*/ \leqslant F/x/$$

dla zadania minimalizacji funkcji F,

lub $$F/x^*/ \geqslant F/x/$$

dla zadania maksymalizacji F.

Najczęściej spotykane w praktyce problemy optymalizacji dotyczą takich zagadnień, jak np.: optymalizacja przydziału celów środkom walki, optymalizacja rozdziału środków materiałowych, optymalizacja wyboru dróg marszu w sieci komunikacyjnej, optymalizacja terminów i zakresu obsług i remontów urządzeń technicznych itp.[x/]. Do rozwiązywania tych problemów wykorzystywane są matematyczne modele decyzyjne, w których korzysta się z matematycznych metod teorii optymalizacji, algebry i rachunku wariacyjnego, topologii i analizy funkcjonalnej, teorii grafów i teorii gier, teorii prawdopodobieństwa z teorią obsługi masowej itp.

Przykład

Rozpatrzmy tzw. zadanie transportowe z kryterium czasu należące do zadań programowania liniowego. Zadanie to można ogólnie

---

x/ Zob.np. J.Czujkow: Badania operacji w wojsku, MON, Warszawa 1972.
W.Filar: Badania operacyjne a problemy zaopatrywania. MON, Warszawa 1973.

sformułować następująco:

niech danych będzie m magazynów, przy czym w każdym z nich znajduje się odpowiednio: $a_1, \ldots, a_i, \ldots, a_m$ jednostek jednorodnych środków materiałowych oraz n jednostek-odbiorców, których potrzeby wynoszą odpowiednio $b_1, \ldots, b_i, \ldots, b_n$ jednostek tych środków. Spełniony przy tym jest warunek bilansowania potrzeb i możliwości

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$$

oraz dana jest macierz $T = [t_{ij}]_{mxn}$, taka że t_{ij} oznacza czas potrzebny na przewiezienie środków materiałowych ze składu i do jednostki j.

Należy wyznaczyć takie ilości środków materiałowych x_{ij} przewożonych z i do j, które spełnią warunki

$$\sum_{i=1}^{m} x_{ij} = b_j, \qquad \sum_{j=1}^{n} x_{ij} = a_i$$

oraz, które zapewnią zaspokojenie potrzeb materiałowych w najkrótszym czasie.

Rozpatrzmy następujący przykład liczbowy:

dane są trzy magazyny zawierające odpowiednio następujące ilości środków materiałowych: $a_1 = 18$, $a_2 = 10$, $a_3 = 12$.
Z magazynów tych należy dostarczyć środki do czterech jednostek, których potrzeby wynoszą odpowiednio: $b_1 = 11$, $b_2 = 9$, $b_3 = 13$, $b_4 = 7$.

Macierz czasu transportu środków dana jest w postaci tablicy:

T				
i \ j	1	2	3	4
1	1	3	7	12
2	2	8	10	8
3	6	1	4	5

Stosując jedną z metod rozwiązywania zadań programowania uzyskano rozwiązanie optymalne, tj. zapewniające dowóz środków materiałowych w minimalnym czasie, przedstawione w tablicy:

X				
i \ j	1	2	3	4
1	1	9	8	0
2	10	0	0	0
3	0	0	5	7

Uzyskane rezultaty należy interpretować następująco: środki magazynu 1 należy dostarczyć do odbiorcy 1 w ilości 1, do odbiorcy 2 w ilości 9 oraz do odbiorcy 3 w ilości 8; wszystkie środki znajdujące się w magazynie 2 należy dostarczyć do odbiorcy 1, czyli w ilości 10; z magazynu 3 środki należy dowieźć do odbiorcy 3 w ilości 5 i do odbiorcy 4 w ilości 7. W ten sposób potrzeby materiałowe jednostek zostaną zaspokojone w najkrótszym czasie.

Matematyczne modele decyzyjne można podzielić, ze względu na charakter wykorzystywanych w nich informacji, na trzy podstawowe grupy /tabela 2/:

a/ <u>modele deterministyczne</u> /obejmują w tabeli przypadek: DD/, w których dysponuje się dokładnie określonymi danymi;

Tabela 2.

KLASYFIKACJA MODELI MATEMATYCZNYCH DECYZYJNYCH

		Charakter informacji o wojskach własnych		
		DETERMINISTYCZNE	PROBABILISTYCZNE	NIEZNANE
Charakter informacji o wojskach nieprzyjaciela	DETERMINISTYCZNE	DD	DP	DN
	PROBABILISTYCZNE	PD	PP	PN
	NIEZNANE	ND	NP	NN

b/ <u>modele probabilistyczne</u> /obejmują w tabeli przypadki: PD, PP, DP/, w których wykorzystywane informacje określone są w kategoriach probabilistyczno-statystycznych /np. w postaci rozkładów prawdopodobieństwa określonych zdarzeń/;

c/ <u>modele growe</u> /obejmują w tabeli przypadki: ND, NP, PN, DN/, w których występują informacje, o których praktycznie biorąc niewiele wiemy, nie potrafimy ich wartości ściśle określić, ani wyrazić w kategoriach probabilistycznych; możemy jedynie przewidywać pewne zdarzenia, aby nie pomijać ich znaczenia w procesie podejmowania decyzji[x/].

Podsumowując dotychczasowe rozważania powiemy, że ogólnie proces analizy systemowej dla określonej sytuacji decyzyjnej i dla danego systemu rzeczywistego może obejmować następujące etapy:

1 - badanie potrzeby;

2 - blokowe sformułowanie problemu decyzyjnego, a w tym:
- identyfikacja systemu /obiektu analizy/,
- określenie zmiennych decyzyjnych,
- określenie warunków - ograniczeń,
- analiza celów i wybór kryterium optymalności;

3 - budowa matematycznego modelu decyzyjnego;

4 - matematyczne sformułowanie problemu optymalizacji;

5 - wybór metody optymalizacji i opracowanie algorytmów decyzyjnych;

6 - opracowanie programów obliczeniowych;

7 - dokonanie obliczeń na maszynie cyfrowej;

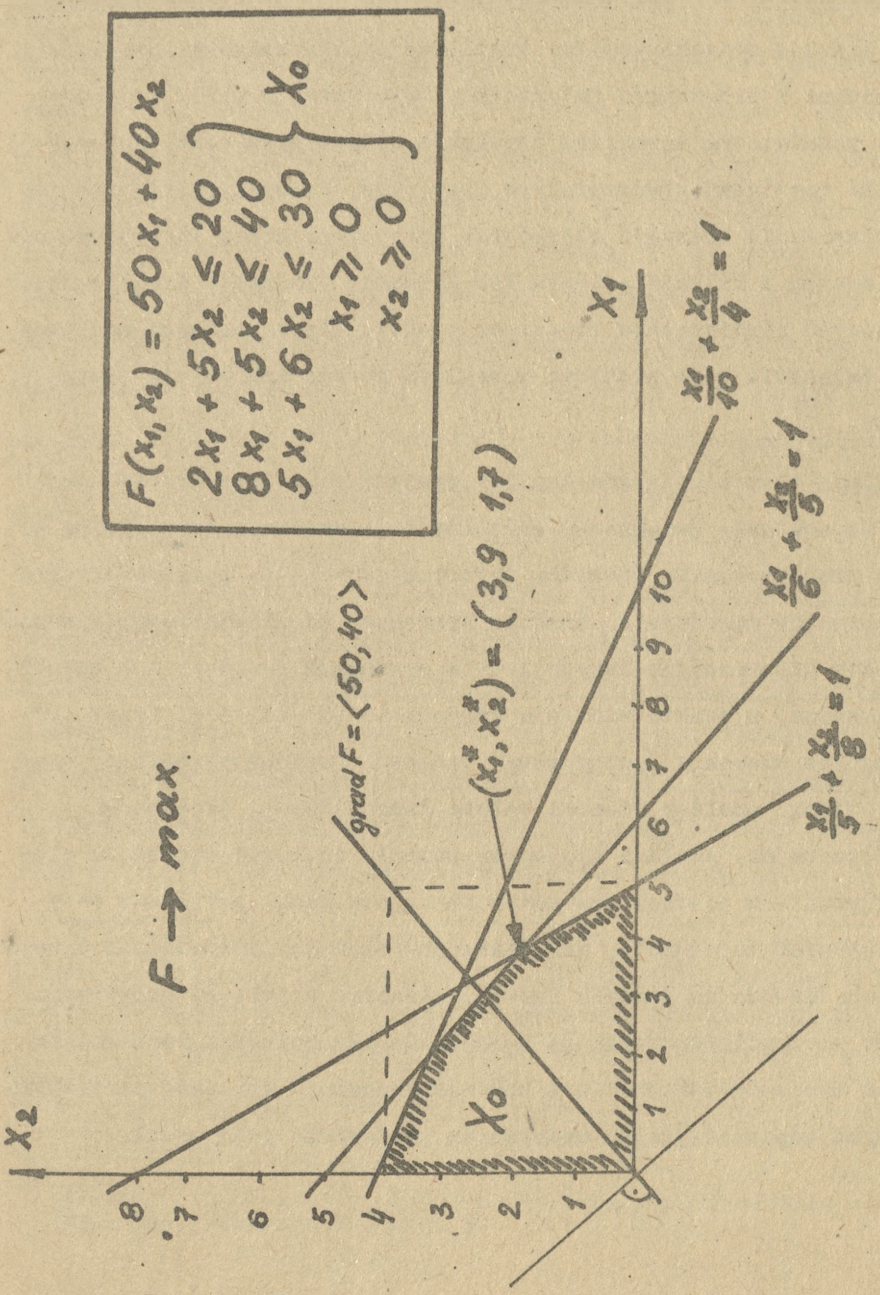
8 - analiza i weryfikacja danych wynikowych;

9 - przedstawienie optymalnego rozwiązania problemu decyzyjnego;

10 - opracowanie decyzji i przekazanie jej wykonawcom.

---

x/ Klasycznym przykładem takich modeli są modele rozpatrywane w teorii gier, czyli tzw. modele konfliktów /gry antagonistyczne/ i modele współdziałania /gry kooperacyjne/.



Rys. 1. Metoda graficzna rozwiązania zadania programowania liniowego.

Przedstawiona ogólna procedura analizy systemowej zastosowanej do rozwiązania "dobrze zestrukturalizowanych" problemów decyzyjnych ukazuje związki analizy systemowej z informatyką. Środki techniczne i programowe informatyki są bowiem, w wielu przypadkach, podstawowym warunkiem uzyskania optymalnych /lub zadowalających/ rozwiązań rzeczywistych problemów decyzyjnych, takich jak np. planowanie operacji wojskowych /dla problemów tych poszukiwany jest wektor o kilkudziesięciu lub kilkuset wymiarach i o liczbie ograniczeń również tego rzędu, co praktycznie przekracza możliwości rozwiązania tego problemu w realnym czasie bez użycia komputerów/.

Inny obszar związków analizy systemowej i informatyki obejmuje zagadnienia związane z automatyzacją systemów dowodzenia, tj. z projektowaniem systemów informatycznych dla potrzeb określonych organów dowodzenia. Analiza systemowa umożliwia rozpatrywanie procesów informacyjno-decyzyjnych w systemach dowodzenia w sposób kompleksowy, z punktu widzenia stopnia realizacji celu przez poszczególne elementy /podsystemy/ badanego /automatyzowanego/ systemu. Obiekt analizy stanowi całość funkcjonalną, problemową i informacyjną. Analizą systemowa pozwala rozłożyć obiekt na elementy proste w postaci elementów funkcjonalnych, problemów decyzyjnych oraz informacji, wzajemnie powiązanych, dotyczących danego systemu. Metoda ta pozwala łatwiej poznawać obiekt automatyzacji i jest szczególnie przydatna w analizie istniejących zautomatyzowanych systemów dowodzenia i budowie koncepcji systemów perspektywicznych odpowiadających wymaganiom przyszłego pola walki.

4. S y s t e m y i n f o r m a t y c z n e

Systemem działania określać będziemy dowolną organizację, czyli zespół ludzi współdziałających dla powtarzalnej realizacji wspólnych celów, wraz z zasobami /środkami, sposobami/ używanymi w działaniu. W każdym systemie działania można wydzielić dwa podstawowe podsystemy: system kierowania i system wykonawczy /roboczy/. Pierwszy realizuje proces kierowania a więc proces o charakterze informacyjnym, drugi - procesy energomaterialne zwane także zasileniowymi. Jeżeli jako system działania rozpatruje się dowolną organizację gospodarczą, to przedmiotem uwagi jest system zarządzania, jeżeli natomiast rozpatruje się dowolną organizację wojskową /pododdział, oddział, związek taktyczny, związek operacyjny itp./, to mamy do czynienia z systemem dowodzenia.

Przyjmijmy, że "dowodzenie wojskami to celowa działalność dowódcy, sztabu i innych organów dowodzenia w zakresie przygotowania działań bojowych i kierowania wysiłków wojsk na pomyślne wykonanie zadania bojowego w toku walki przez uzyskiwanie i studiowanie danych o sytuacji, podejmowanie decyzji stosownie do tej sytuacji oraz przekazywanie zadań wykonawcom"[x]. Wtedy najogólniejsze określenie systemu dowodzenia jest następujące

System dowodzenia to taki system działania, a więc zespół współdziałających ludzi dysponujący określonymi metodami i środkami technicznymi, który realizuje procesy informacyjne niezbędne do osiągnięcia zamierzonych celów działania /walki, bitwy, operacji/.

---

x/ D.Iwanow, W.Sawieljew, P.Szemanski: "Zasady dowodzenia wojskami", MON, Warszawa 1973, s. 30.

Rozszerzeniem powyższej definicji jest definicja, która brzmi następująco:

> System dowodzenia stanowi uporządkowany zgodnie z zasadami sztuki wojennej zbiór dowództw[x] wraz z technicznymi środkami dowodzenia powiązanych pod względem funkcjonalnym, informacyjnym i technicznym, zapewniających wykonanie zadań bojowych /osiągnięcie celów walki, operacji/.

W systemie dowodzenia realizowane są procesy informacyjno-decyzyjne organizujące przebieg procesów roboczych /wykonawczych/. Procesy te decydują o powodzeniu danego działania w określonych warunkach.

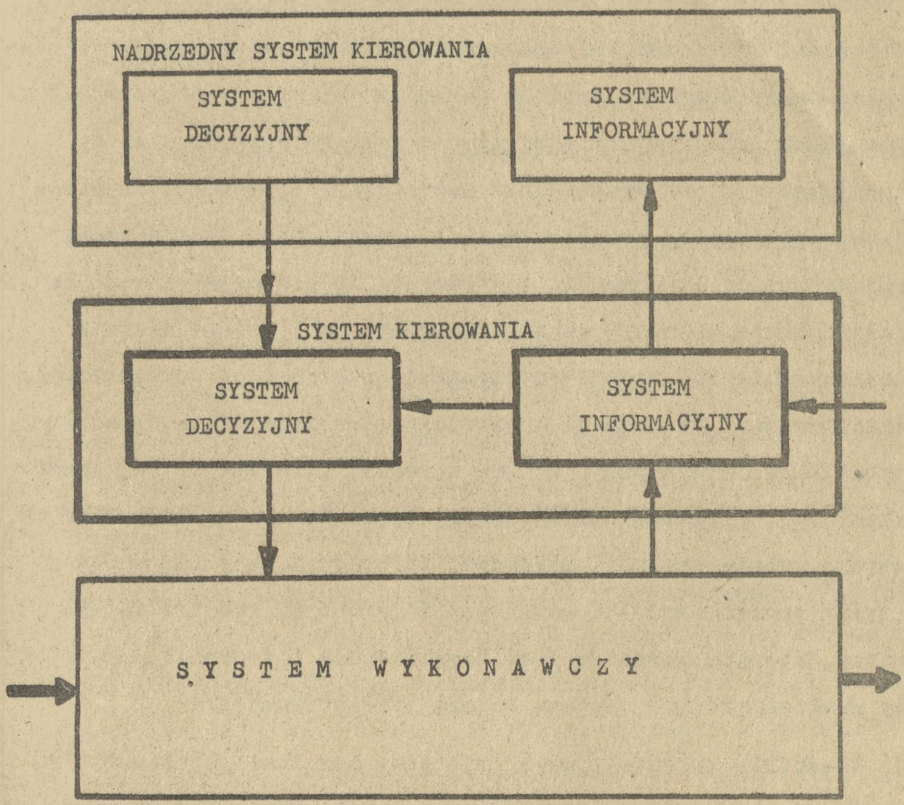
Istotą dowodzenia jest podjęcie przez dowódcę <u>decyzji, czyli dokonanie wyboru celu, sposobów i środków działania</u>[xx], na podstawie m.in. zamiaru wyższego przełożonego i otrzymanego zadania oraz informacji dotyczących aktualnej sytuacji, nieprzyjaciela, wojsk własnych, warunków działania, sąsiadów itp. Właściwa jest zatem koncepcja modelu systemu kierowania /dowodzenia, zarządzania/ tworzonego przez dwa podstawowe podsystemy: <u>system decyzyjny</u> i <u>system informacyjny</u>.

> Systemem decyzyjnym określamy podsystem systemu kierowania /dowodzenia/, którego istotą działania jest podejmowanie decyzji /dokonywanie wyboru celów, sposobów i środków działania/.

---

x/ Przez dowództwo będzie się rozumieć dowódcę i organy dowodzenia, przy pomocy których dowodzi on podległymi wojskami.

xx/ Jest to jedno z kilku ujęć spotykanych np. w prakseologii, ekonomii itd.



Rys. 2. Ogólny model systemu kierowania.

Systemem informacyjnym będziemy nazywali podsystem systemu kierowania /dowodzenia/, którego istotą działania jest realizacja procesów informacyjnych /zbieranie, przechowywanie, przesyłanie, przetwarzanie, rozpowszechnianie, udostępnianie informacji/ zgodnie z potrzebami i wymaganiami użytkowników /decydentów-dowódców, szefów rodzajów wojsk i służb itp./.

Współcześnie systemem informacyjnym będziemy więc nazywali pewien zorganizowany kompleks środków technicznych wraz z obsługującą go kadrą ludzką realizujący zabezpieczenie informacyjne systemu decyzyjnego poprzez wykonywanie następujących procesów informacyjnych: zbieranie, przechowywanie, przesyłanie, przetwarzanie i udostępnianie informacji. W systemach informacyjnych realizowane są zatem takie operacje na informacjach, jak: zmiana miejsca przechowywania informacji /zbieranie, przesyłanie, udostępnianie/, "magazynowanie" informacji w specjalnych urządzeniach "pamiętających" /przechowywanie/, zmiana wartości informacji, jej treści, postaci itp. /przetwarzanie/. Przyjmując, że system informacyjny wywołuje zmianę wartości użytkowej informacji mamy najczęściej na myśli wzrost wartości użytkowej informacji, możliwe są jednak systemy, których zadaniem jest dewaluowanie informacji /np. systemy zakłócające lub systemy celowej dezinformacji/.

Klasyfikacja istniejących systemów informacyjnych może być przeprowadzona z wielu różnorodnych punktów widzenia, na przykład[x]:

1. Z punktu widzenia rodzaju zastosowań - na systemy pośredniczące w przekazywaniu lub przetwarzaniu informacji /np. system

---

x/ J.L.Kulikowski: Teoretyczne podstawy organizacji systemów informacyjnych. Archiwum Automatyki i Telemechaniki, tom XV, zeszyt 4, 1970.

łączności, sieć usługowa ZETO itp./, systemy doradczo-usługowe /np. systemy informacji naukowo-technicznej i ekonomicznej, systemy ewidencyjne,,systemy automatycznego wspomagania projektowania itp./, systemy śledząco-nadzorujące /np. system kontroli ruchu lotniczego, system kontroli przebiegu procesu technologicznego itp./, systemy dyspozycyjno-sterujące oraz systemy informowania kierownictwa /decydentów/.

2. Z punktu widzenia organizacji systemu - na systemy o organizacji konwencjonalnej, w tym systemy o działaniu ciągłym /np. stale działający system obserwacji/, cyklicznym /np. system cyklicznej kontroli obiektów/ lub acyklicznym /np. system automatycznego planowania operacji/, systemy działające metodą grupowania zadań informacyjnych w "bloki" /np. systemy liczące pracujące tzw. metodą wsadową/ i systemy wieloprocesowe, mogące równolegle realizować zadania informacyjne opisane przez niezależne algorytmy i programy /np. wielodostępne systemy liczące oparte na zasadzie tzw. podziału czasu/.

3. Z punktu widzenia synchronizacji pracy systemu z otoczeniem - na systemy pracujące w czasie "własnym", słabo związanym z tempem zmian zachodzących w otoczeniu /np. większość aktualnie działających systemów automatycznego zarządzania i dowodzenia/ i systemy pracujące w czasie "rzeczywistym", w którym tempo przetwarzania informacji jest ściśle narzucone przez tempo zmian zachodzących w otoczeniu /np. systemy sterowania procesami technologicznymi odznaczającymi się dużą zmiennością w czasie wymagającą szybkiej interwencji systemu sterowania/.

4. Z punktu widzenia typu powiązań między częściami składowymi systemu - na systemy o integracji słabej, w których nie są konieczne bezpośrednie fizyczne połączenia między urządzeniami

informacyjnymi i przepływ sygnałów w pewnych fazach jest zastępowany transportem materialnych nośników informacji /połączenia typu "off line" za pośrednictwem kart, taśm perforowanych itp./ i systemy o integracji silnej, w których urządzenia informacyjne są połączone fizycznie, a jedyną formą przenoszenia informacji jest przepływ sygnałów /połączenia typu "on line"/.

5. Z punktu widzenia stopnia standaryzacji części składowych systemu - na systemy niezunifikowane i systemy zunifikowane na poziomie: fizycznych parametrów sygnałów /przede wszystkim zewnętrznych, przenoszących informację między częściami składowymi systemu/, kodów, formatów słów, języków zewnętrznych itp.

6. Z punktu widzenia klasy algorytmów realizowanych przez system - na systemy uniwersalne, w których klasa algorytmów nie jest zdeterminowana i systemy specjalistyczne, w których klasa algorytmów i programów jest z góry określona.

7. Z punktu widzenia autonomii sterowania w systemie - na systemy w pełni autonomiczne, dysponujące własnymi środkami przesyłania i przetwarzania informacji /np. system automatycznego sterowania lotem rakiet stanowiący silnie zintegrowany kompleks środków technicznych wydzielonych wyłącznie do tego celu/ i systemy o ograniczonej autonomii, korzystające w pewnych fazach procesu przetwarzania informacji z usług innych systemów.

8. Ze względu na trwałość struktury systemu - na systemy stacjonarne, w których zmienność struktury nie odgrywa istotnej roli, systemy quasistacjonarne /np. system obserwacji meteorologicznej rozważany z punktu widzenia jego stopniowej ewolucji związanej ze wzrostem liczby punktów obserwacji/ i systemy niestacjonarne, o strukturze szybkozmiennej /np. mobilne polowe zautomatyzowane systemy dowodzenia/.

9. Ze względu na rodzaj techniki przetwarzania informacji w systemie - na systemy cyfrowe lub alfanumeryczne, analogowe i hybrydowe /mieszane/.

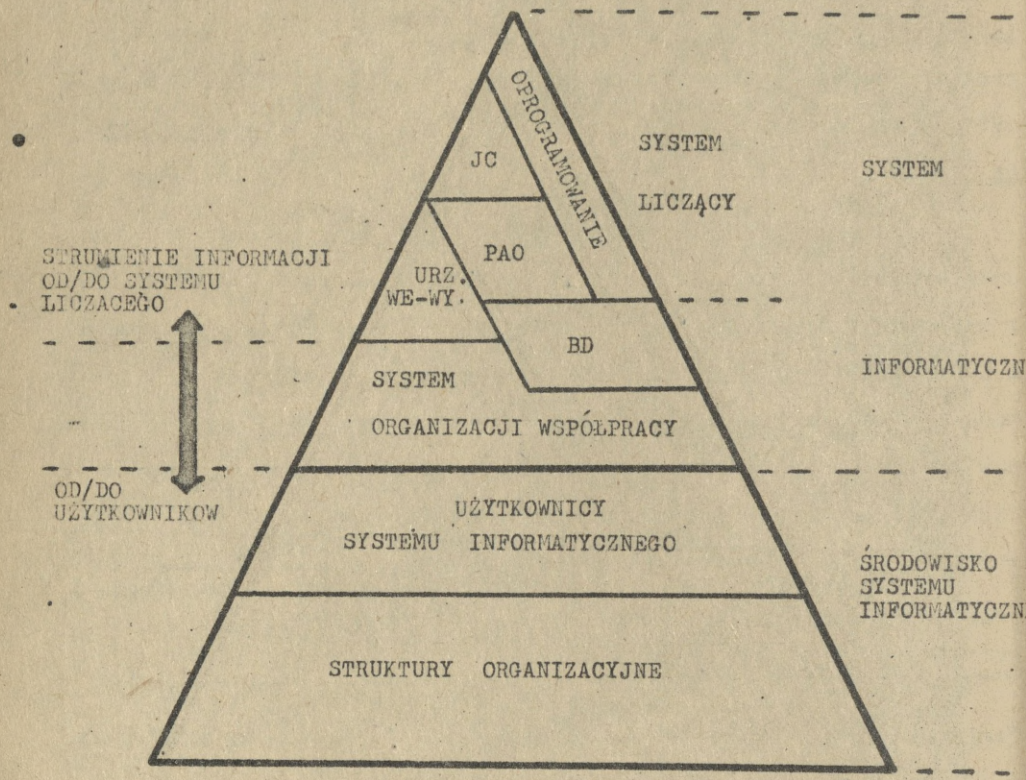
10. Ze względu na stopień automatyzacji podstawowych operacji informacyjnych w systemie - systemy tradycyjne, oparte głównie na pracy ludzkiej, systemy półautomatyczne, w których jedynie część operacji informacyjnych jest zautomatyzowana i systemy automatyczne.

Przytoczona klasyfikacja nie jest oczywiście pełna, gdyż nie uwzględnia zwłaszcza faktu, że najczęściej mamy do czynienia z systemami, w których występują cechy mieszane. Do nich należą z pewnością systemy informacyjne działające w ramach zautomatyzowanych systemów dowodzenia. A zatem, system informatyczny to także system informacyjny, jednak jego działanie opiera się na systemie liczącym, czyli jest wspomagany zestawem komputerowym.

Systemem informatycznym nazywać będziemy taki system informacyjny, w którym procesy informacyjne realizowane są za pomocą technicznych i programowych środków informatyki.

Powyższe określenie systemu informatycznego traktować będziemy, jako określenie w szerokim sensie. Niekiedy spotykane jest określenie w wąskim sensie, w którym system informatyczny traktowany jest jako zestaw programów użytkowych i organizacyjnych opracowanych dla realizacji określonych zadań /funkcji/ dowodzenia /zarządzania/.

Inne spotykane określenie mówi, że "systemem informatycznym nazywamy zespół składający się z trzech nierozłącznie powiązanych ze sobą elementów: systemu cyfrowego /liczącego/, banku danych oraz systemu organizacji współpracy /rys.3 /. Bank danych zwiera



Rys. 3. Uogólniony model struktury systemu informatycznego /JC - jednostka centralna, PAO - pamięć operacyjna, BD - bank danych /.

zbiory danych związanych z funkcjonowaniem danej organizacji. System organizacji współpracy zapewnia właściwe wykorzystanie systemu liczącego oraz efektywny obieg informacji między systemem i jego środowiskiem. Dzięki niemu użytkownicy dostarczają do systemu informatycznego dane wejściowe oraz wykorzystują otrzymane zeń dane wynikowe. W celu usprawnienia procesu komunikacji człowieka-użytkownika z systemem informatycznym dąży się do tego, aby system organizacji współpracy nie był elementem oddzielającym, lecz integrującym użytkowników i system liczący. Jednym z najlepszych rozwiązań w tym zakresie jest wyposażenie użytkowników w ich miejscach pracy w urządzenia końcowe wielodostępnego systemu liczącego /np. monitory ekranowe, drukarki z klawiaturą itp./.

Najbardziej znamienną cechą rozwoju informatyki lat siedemdziesiątych jest powstanie sieci komputerowych, czyli systemów teleinformatycznych[x/] o wielu oddalonych od siebie komputerach obliczeniowych i sterujących przesyłaniem informacji i rozległym systemie łączności między komputerami. Sieci teleinformatyczne to geograficznie rozproszone systemy liczące, w których do celów przekazywania informacji pomiędzy poszczególnymi ich składnikami /komputerami/ stosuje się środki i metody typowe dla telekomunikacji /modemy, koncentratory, synchronizatory, multipleksery, centrale komutacyjne itp./.

Wśród charakterystycznych tendencji rozwojowych systemów informatycznych na uwagę zasługują pewne typy integracji w systemie, a mianowicie:

---

x/ Teleinformatyka to, po prostu, dziedzina telekomunikacji dla komputerów.

1/ integracja funkcjonalna polegająca na tym, że zadania realizowane przez poszczególne systemy wchodzą w skład pewnych kompleksowych procesów kierowania /dowodzenia, zarządzania/, skąd dla systemów składowych wynikają pewne postulaty ujednolicające ich parametry techniczno-eksploatacyjne oraz koordynujące ich współdziałanie;

2/ integracja informacyjna polegająca na wzajemnym udostępnianiu lub wspólnym wykorzystywaniu zbiorów danych /baz danych/, wynikają warunki ujednolicające format zapisu danych, organizację ich przechowywania itp.;

3/ integracja techniczna polegająca na współpracy jednostek przetwarzania danych za pośrednictwem transmisji danych, skąd wynikają warunki dopasowania konfiguracji sprzętowych na poziomie sygnałów, kodów i formatów danych, instrukcji i języków programowania;

4/ integracją organizacyjna polegająca na podporządkowaniu poszczególnych systemów składowych jednemu centrum sterującemu, które określa zarówno ich zadanie perspektywiczne, jak i bieżące, wymagające skoordynowanego działania.

Rozwój wojskowych systemów informatycznych obejmował trzy zasadnicze etapy:

- etap 1 obejmuje rozwiązania autonomicznych zadań informacyjnych /głównie typu ewidencyjno-sprawozdawczego/ w poszczególnych dziedzinach działalności sił zbrojnych, w warunkach różnorodności sprzętu komputerowego, niskiego poziomu oprogramowania oraz wysokich kosztów automatyzacji;
- etap 2 obejmuje próby integracji poszczególnych "dziedzinowych" systemów przetwarzania danych oraz rozwiązywania tzw. zadań

"aktywnych", związanych bezpośrednio z procesem podejmowania decyzji /etap związany jest z rozwojem możliwości komputerów i metodologii programowania/;

- etap 3 obejmuje początki tzw. automatyzacji kompleksowej systemów dowodzenia - ma tu miejsce: integracja poszczególnych podsystemów specjalistycznych, unifikacja sprzętu komputerowego i telekomunikacyjnego, dalszy rozwój metodologii programowania, stosowanie banków danych, tworzenie sieci komputerowych, wykorzystanie łączności satelitarnej, a ponadto - wzrost niezawodności środków technicznych i programowych itp.

Współczesne systemy informatyczne tworzą następujące elementy strukturalne:

1/ struktura przestrzenna systemu, czyli lokalizacja punktów wymiany /pobierania i wydawania/ informacji z otoczeniem, punktów jej gromadzenia, przetwarzania, punktów komutacyjnych, wreszcie dróg przesyłania informacji między tymi punktami.

2/ struktura techniczna systemu tworzona przez węzły, którym przyporządkowane są nazwy i typy urządzeń technicznych wchodzących w skład systemu informatycznego, a skierowanym gałęziom - połączenia informacyjne symbolizujące drogi przepływu sygnałów lub transportu nośników informacji między urządzeniami.

3/ struktura informacyjna systemu, czyli zbiór złożonych relacji informacyjnych /procedur/ realizowanych w danej strukturze technicznej /relacji takich w systemie może być więcej niż jedna: w systemie wielodostępnym każdy użytkownik może np. zainicjować niezależnie od innych użytkowników relację/. Z formalnego punktu widzenia złożona relacja informacyjna /procedura/ może być określona jako spójny układ relacji wiążących pewne zbiory wiadomości.

4/ struktura programowa systemu, czyli zbiór programów nadzorczo-sterujących /systemy operacyjne/ i programów użytkowych niezbędnych dla realizacji zadań informacyjnych przez środki techniczne.

Podsumowując powyższe uwagi na temat współczesnych systemów informatycznych wyróżnimy podstawowe rodzaje systemów wykorzystywanych w siłach zbrojnych[x/].

Automatyczne systemy dowodzenia charakteryzują się tym, że cały cykl kierowania jest realizowany przez środki techniczne bez bezpośredniego udziału człowieka w poszczególnych jego ogniwach. Znajdują one zastosowanie przede wszystkim w procesach kierowania środkami walki, np. w naprowadzaniu środków obrony przeciwlotniczej, w kierowaniu uzbrojeniem okrętów, w naziemnych i pokładowych systemach dowodzenia lotnictwem itp. Do zautomatyzowanych systemów dowodzenia zalicza się systemy, w których część procesów dowodzenia jest wykonywana środkami technicznymi, a część przez człowieka. Udział człowieka i wykorzystanie środków technicznych w tych systemach mogą być różne, jednak w każdym przypadku i sytuacji człowiek spełnia zasadniczą funkcję - podejmuje ostateczną decyzję. Polowe zautomatyzowane systemy dowodzenia są przeznaczone do zapewnienia efektywnego wykorzystania sił i środków bojowych w walce dla pomyślnego wykonania przez wojsko zadań bojowych przy minimalnych stratach osobowych i materiałowych. Podstawowymi elementami polowych zautomatyzowanych systemów dowodzenia są punkty dowodzenia, tyłowe punkty dowodzenia, wysunięte punkty dowodzenia itp., wyposażone w odpowiednie środki łączności i środki

---

x/ A.Horak: Zagadnienia automatyzacji i mechanizacji dowodzenia i zarządzania w wojsku. Wyd. Szt.Gen.WP, Warszawa 1972.

informatyki, a działalność ich podporządkowywana jest jednemu celowi - osiągnięciu zwycięstwa w walce zbrojnej.

5. Projektowanie wojskowych systemów informatycznych

Projektowanie systemów informatycznych obejmuje całokształt przedsięwzięć organizacyjnych i finansowych, technicznych i programowych prowadzących do wykorzystywania efektywnego metod i środków informatyki w systemach kierowania /dowodzenia i zarządzania/. Jest to zatem ciąg operacji zbudowany w oparciu o sekwencję: analiza /sytuacji, w której tkwi problem/, synteza /możliwych rozwiązań/, ocena i decyzja /przyjęcie rozwiązań/, weryfikacja /potwierdzenie słuszności wybranego rozwiązania/. Rezultatem tak rozumianego procesu projektowania jest projekt w postaci dokumentacji projektowej przedstawiającej wybór projektowania /system informatyczny/, który powinien spełniać wymagania określone w zadaniu projektowym. Projekt powinien zawierać opis efektów funkcjonowania systemu projektowanego, funkcje zapewniające uzyskiwanie tych efektów, środki niezbędne do realizacji procesów, warunki i przebieg rozwoju systemu.

Każdy wojskowy system informatyczny, niezależnie od tego, czy jest to system przewidywany /planowany/, opracowywany czy zrealizowany, jest przeznaczony dla konkretnego użytkownika, którym jest np. sztab ogólnowojskowy, szefostwo rodzaju wojsk, instytucja centralna itp. Istotną konsekwencją tego faktu jest brak możliwości określenia jednolitych i niezmiennych szczegółowych algorytmów projektowania systemów informatycznych, które to algorytmy mogłyby być stosowane uniwersalnie. Istnieją jednakże

pewne ogólne zasady projektowania systemów informatycznych, takie jak np.:

a/ w trakcie wszystkich prac nad systemem, a więc podczas opracowywania jego koncepcji, realizacji projektu i wdrażania systemu niezbędna jest ścisła współpraca specjalistów z zespołu projektowego - analityków systemu, projektantów systemów, programistów i inżynierów-informatyków - z użytkownikami systemu;

b/ opracowanie harmonogramów realizowanych prac, prowadzenie bieżącej szczegółowej dokumentacji, ścisła kontrola wykonania poszczególnych etapów prac itd.;

c/ projektowanie systemów zgodnie z ogólnym schematem: opracowanie koncepcji systemu - realizacja systemu;

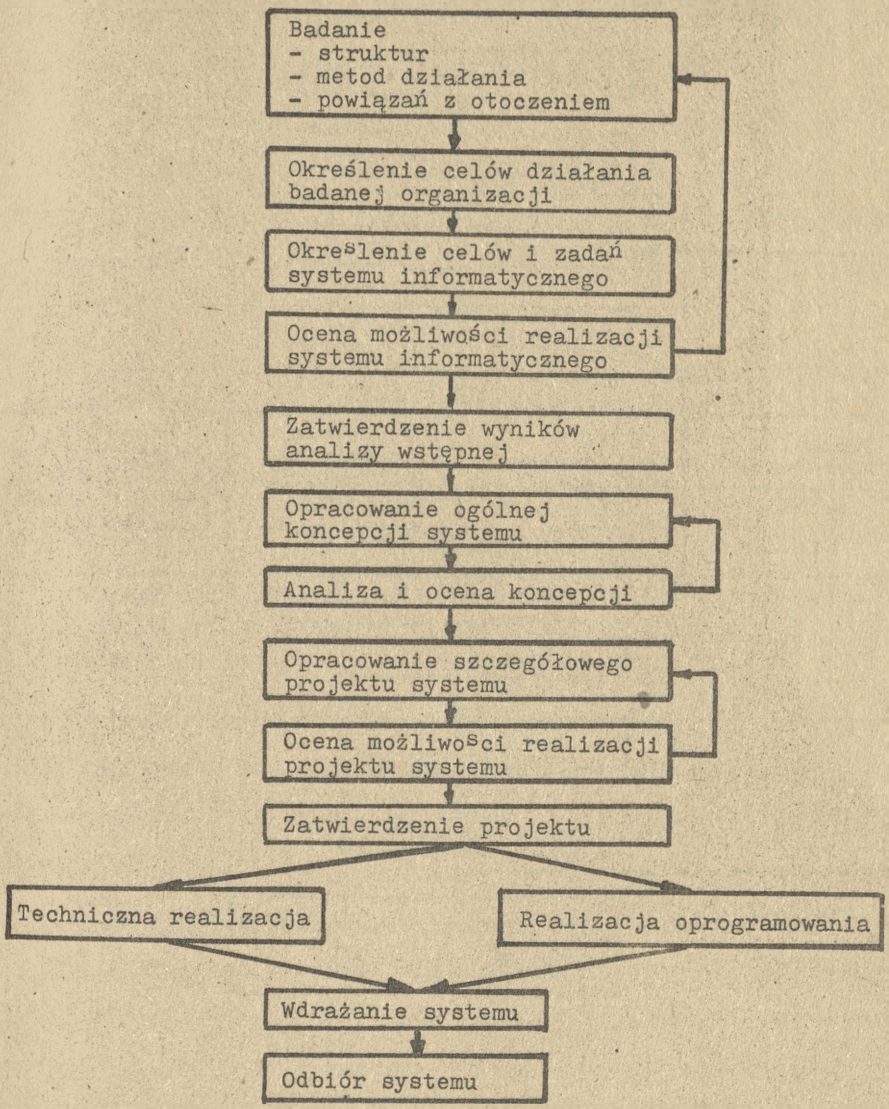
d/ projektowanie systemu informatycznego w taki sposób, by mógł być on rozwijany w miarę ewolucji wykorzystującego go systemu kierowania.

Mówiąc o projektowaniu systemów informatycznych należy wyróżnić dwa przypadki:

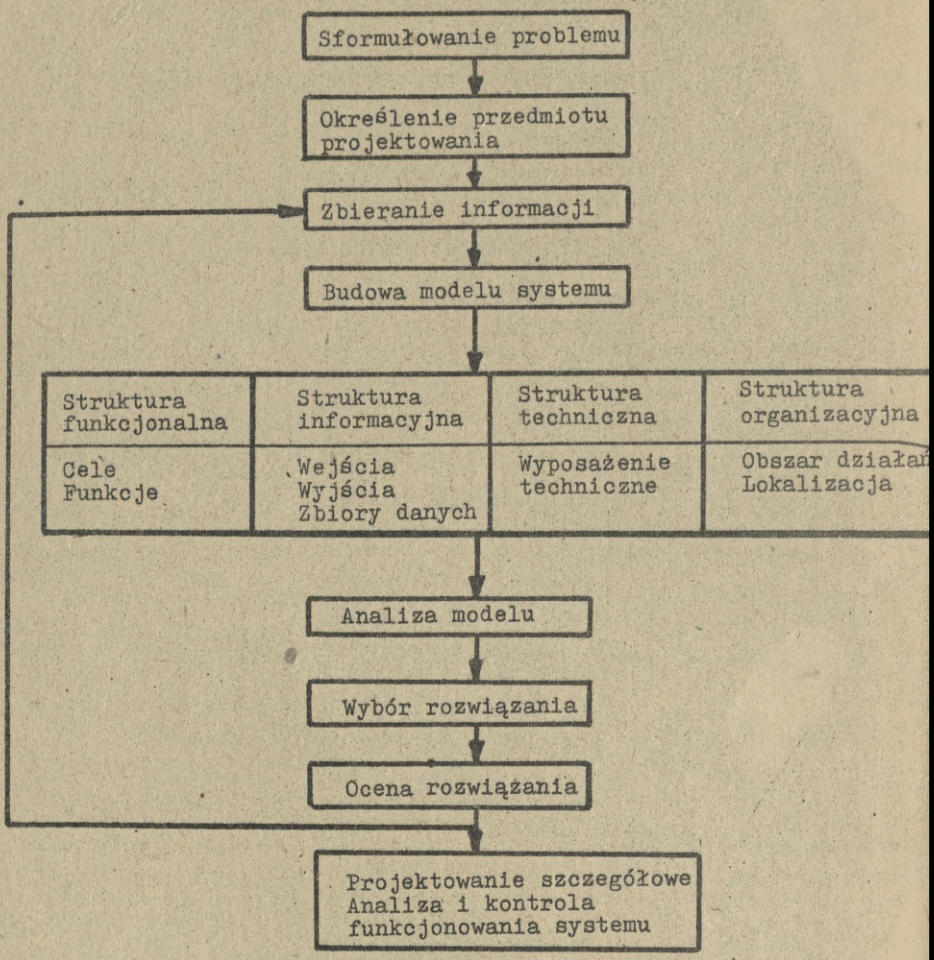
- gdy projektujemy system informatyczny jako kompleks środków technicznych, materiałowych i programowych, realizujących wiele różnych funkcji kierowania w różnych warunkach /w tym zmieniających się natężeń strumieni informacyjnych/;

- gdy projektujemy system informatyczny jako program /zbiór programów/ realizujący określoną funkcję kierowania /np. w warunkach istniejącego już ośrodka obliczeniowego/.

Oczywiście pierwszy z wymienionych przypadków obejmuje znacznie większą ilość złożonych przedsięwzięć organizacyjnych, techniczno-programowych i ekonomicznych. Jednakże zaprojektowanie efektywnego programu użytkowego realizującego złożoną funkcję i korzystającego z dużych zbiorów danych należy również do złożonych przedsięwzięć,



Rys.4. Ogólny schemat procesu projektowania systemu informatycznego.



Rys.5. Ogólny model projektowania systemu.

niewiele mających wspólnego z rutynową pracą programistów. Oba wyróżnione przypadki dotyczą działania twórczego.

W procesie projektowania systemów informatycznych tzw. "wąskie gardło" stanowi oprogramowanie systemu. Jest to wynik "indywidualnego" programowania systemów dostosowanych do potrzeb różnych użytkowników. Wynikła stąd konieczność racjonalizacji programowania, którą osiąga się między innymi poprzez:

- wykorzystywanie w systemach uniwersalnych pakietów zastosowań dostarczonych przez producentów komputerów;
- parametryzację programów realizujących typowe czynności potrzebne dla każdego rodzaju zastosowań /użytkownik jak gdyby programuje poprzez sprowadzanie tzw. parametrów/;
- modularność programów, przejawiającą się w tym, że każdą z logicznych części programu można modyfikować lub wymienić bez wprowadzania zmian do pozostałych części programu.

Z pewnością na sposób postępowania w projektowaniu systemów informatycznych mają wpływ wymagania użytkowników z punktu widzenia możliwości wykorzystania systemu, dotyczące takich cech, jak np.:

a/ rodzaj dostępu do zasobów systemu, który pozwala wyróżnić systemy jednodostępne /praktycznie przeznaczone dla jednego użytkownika lub kilku użytkowników w przypadku ich łącznego zlokalizowania w tym samym miejscu/, w których urządzenia zewnętrzne są podłączone bezpośrednio do kanałów komputera oraz systemy wielodostępne /przeznaczone dla wielu użytkowników zlokalizowanych zazwyczaj w zupełnie różnych miejscach/, które wymagają podłączenia urządzeń zewnętrznych poprzez dodatkowe urządzenia sterujące umożliwiające zwiększenie przepustowości kanałów wejścia/wyjścia komputera;

b/ sposób wprowadzania danych do przetwarzania: wsadowy /gdy dane i programy do ich przetwarzania są wprowadzane łącznie po ich odpowiednim zgrupowaniu i uporządkowaniu/, w systemach takich urządzenia zewnętrzne mogą być jednokierunkowe, albo konwersacyjny /gdy zarówno dane, jak i programy mogą być dostarczane w sposób systematyczny, w miarę jak zostają zrealizowane wcześniejsze zadania/, który wymaga zwrotnych urządzeń zewnętrznych;

c/ sposób przesyłania danych pomiędzy użytkownikiem a komputerem, który może być tradycyjny w systemach lokalnych lub też przy wykorzystaniu techniki telekomunikacyjnych w systemach zdalnych;

d/ charakter strumienia danych, który może być opóźniony, jeśli np. dane zbierane są okresowo, lub nadążny, jeśli dane są przekazywane do przetwarzania niemal natychmiast po ich powstaniu /systemy wykorzystują urządzenia do bezpośredniego i natychmiastowego wprowadzania danych/.

Określenie wyróżnionych cech funkcjonalnych systemu informatycznego pozwala ukierunkować przebieg prac projektowych na właściwe zadanie projektowe oraz wykorzystać właściwe metody i techniki projektowania.

W odniesieniu do wojskowych systemów informatycznych zostały przyjęte określone ustalenia ujęte w instrukcji projektowania i wdrażania. Zgodnie z nimi w zakres przedsięwzięć projektowo-wdrożeniowych w poszczególnych fazach ich realizacji wchodzą: prace przygotowawcze do projektowania, właściwe projektowanie systemu informatycznego, zabezpieczenie organizacyjno-techniczne systemu oraz wdrażanie zaprojektowanego systemu i jego doskonalenie w czasie eksploatacji. Użytkownikiem systemu informatycznego

jest każda korzystająca z usług tego systemu instytucja, organ dowodzenia i zarządzania, placówka naukowo-badawcza, przedsiębiorstwo, jednostka wojskowa i osoba funkcyjna. Wykonawcami przedsięwzięć projektowo-wdrożeniowych, w zależności od etapu prac, są organa /komórki, osoby funkcyjne/ sztabowe użytkownika i informatyki oraz organa /placówki naukowe, zespoły/ wykonawcze informatyki.

Faza prac przygotowawczych do projektowania systemu informatycznego stanowi kompleks przedsięwzięć poprzedzających i warunkujących rozpoczęcie prac projektowych związanych z zamiarem użycia metod i środków informatyki do usprawnienia systemu kierowania. Celem tych prac jest dokonanie analizy, określenie i uzasadnienie potrzeby, kierunku i zakresu zastosowania metod i środków informatyki oraz sformułowanie ogólnych wymagań i założeń, a także wstępne oszacowanie oczekiwanych nakładów i efektów systemu. Prace przygotowawcze obejmują: badanie /analizy/ systemu kierowania i formułowanie zadania projektowego systemu informatycznego. Celem formułowania zadania projektowego jest opracowanie ogólnych wymagań oraz założeń operacyjnych, organizacyjnych, technicznych i technologicznych dotyczących budowy systemu informatycznego w usprawnianym systemie kierowania. Opracowuje je użytkownik, przy współpracy konsultacyjnej organu projektującego, pod kierownictwem instytucji kierującej.

Prace projektowe realizuje się w trzech kolejnych etapach: projektowania koncepcyjnego, wstępnego i technologicznego. W wyniku zakończenia prac w fazie projektowania powinny być rozwiązane problemy założonego usprawnienia systemu kierowania, w postaci projektu systemu informatycznego, odpowiednio udokumentowanego i przygotowanego pod względem organizacyjnym i technicznym

do próbnej eksploatacji. Celem projektowania koncepcyjnego jest określenie i udokumentowanie rozwiązań modelowych systemu informatycznego i organizacji jego działania oraz oszacowanie kosztów i efektów budowy i działania tego systemu. Celem projektowania wstępnego jest określenie i udokumentowanie koncepcji technologicznej realizacji procesu przetwarzania danych oraz takie określenie organizacji systemu i jego wyposażenia technicznego, które mogłoby stanowić podstawę do dalszych prac projektowych i organizacyjno-technicznych związanych z budową systemu. Celem projektowania technologicznego systemu informatycznego jest oprogramowanie systemu oraz opracowanie dokumentacji programowej i eksploatacyjnej, określającej sposób funkcjonowania wszystkich elementów systemu oraz umożliwiającej jego wdrożenie. Projekt technologiczny systemu informatycznego powinien zawierać:

- dokumentację programową,
- dokumentację eksploatacyjną dla organów eksploatujących system,
- dokumentację eksploatacyjną przeznaczoną dla użytkownika.

Dokumentacja programowa każdego programu obejmuje następujące elementy:

- ogólną charakterystykę funkcji programu i zastosowanych rozwiązań technologicznych,
- schemat blokowy programu,
- opis i strukturę obszarów roboczych programu,
- wydruk programu źródłowego,
- wybrane wyniki testowania programu na danych modelowych.

Dokumentacja eksploatacyjna dla organów eksploatujących system powinna zawierać między innymi: szczegółowe schematy operacyjne /podsystemów, jednostek funkcjonalnych, modułów/, schematy operacyjne programów, wzory parametrów i danych sterujących, zasady kontro

dokumentów źródłowych i danych przenoszonych na maszynowe nośniki informacji, instrukcje sporządzania maszynowych nośników informacji instrukcję kontrolowania dokumentów wynikowych itp.

Dokumentacja eksploatacyjna przeznaczona dla użytkownika powinna obejmować:

- organizację i zasady działania systemu,
- obowiązki organów i osób funkcyjnych uczestniczących w procesie eksploatacji systemu,
- wzory dokumentów źródłowych,
- zasady wypełniania, sprawdzania, kompletowania i przekazywania dokumentów źródłowych,
- wzory zestawień wynikowych i opis ich zawartości,
- sposób analizy i kontroli informacji zawartej w zestawieniach,
- zasady postępowania w przypadku zakłóceń w systemie,
- zasady wykorzystywania i korekty dokumentów wynikowych, kontrolnych i wykazów błędów,
- zasady współpracy instytucji i osób uczestniczących w procesie eksploatacji systemu,
- harmonogram eksploatacji.

Wdrażanie systemu informatycznego jest fazą, w której uruchamia się próbnie, a następnie włącza zaprojektowany system informatyczny do działającej struktury organizacyjno-funkcjonalnej oraz informacyjnej i technicznej usprawnianego systemu kierowania. Faza ta ma więc na celu sprawdzenie i zweryfikowanie rozwiązań oraz użytkowe włączenie zaprojektowanego systemu informatycznego do funkcjonującego systemu kierowania.

Eksploatacja próbna jest etapem wdrażania, w którym następuje uruchamianie i próbne włączenie do działającego systemu kierowania wszystkich lub wybranych ogniw organizacyjnych systemu kierowania i elementów systemu informatycznego. W wyniku eksploatacji próbnej otrzymuje się zweryfikowany, poprawiony, skompletowany

i zakwalifikowany do eksploatacji użytkowej system informatyczny oraz określony poziom sprawności działania wszystkich ogniw organizacyjnych systemu kierowania i elementów systemu informatycznego objętych eksploatacją próbną.

W skład pełnej podstawowej dokumentacji merytorycznej systemu informatycznego wchodzą:

1. Analiza systemu kierowania.
2. Zadanie projektowe systemu informatycznego.
3. Projekt koncepcyjny systemu informatycznego.
4. Projekt wstępny systemu informatycznego.
5. Dokumentacja programowa systemu informatycznego.
6. Dokumentacja eksploatacyjna systemu informatycznego.
7. Instrukcja organizacji i użytkowania systemu informatycznego.

6. Banki danych w zautomatyzowanych systemach dowodzenia

Bank danych jest nie tylko atrakcyjną nowoczesną ideą informatyczną, lecz także trwałym rezultatem ewolucji systemów informatycznych. Istota tej idei polega na odejściu od ujęcia projektowego ukierunkowanego na komputer do ujęcia wyrażającego punkt widzenia użytkownika na zbiory niezbędnych danych w procesie kierowania /dowodzenia, zarządzania/. Wyraża się to w postaci pewnych czynników takich, jak:

- uelastycznienie systemu informatycznego z punktu widzenia podatności na zmiany zakresu rzeczowego danych wejściowych, zawartości zbiorów i przekrojów informacji wynikowych,
- szybsza obsługa potrzeb informacyjnych /zmniejszenie czasu odpowiedzi/,

- zmniejszenie stopnia redundancji /nadmiaru, powtarzalności/ danych w zbiorach oraz zwiększenie stopnia integracji informacji /możliwość uzyskiwania złożonych powiązań logicznych między różnorodnymi informacjami/,

- efektywniejsze oprogramowanie systemowe /dzięki wykorzystaniu standardowego oprogramowania banku danych/,

- racjonalizacja technologii przetwarzania m.in. przez zmniejszenie liczby przebiegów sortowania,

- lepsze odwzorowanie przez system informatycznej istoty procesów kierowania, lepsze wspomaganie procesów decyzyjnych /np. dzięki szybkiemu dostępowi do żądanych informacji/,

- zdolność rozbudowy systemu informatycznego itp.

Mówiąc o banku danych będziemy rozumieć pod tym pojęciem bazy danych, system zarządzania bazą danych i języki banku danych.

Baza danych jest to zestaw zbiorów utrzymywanych przez system i wykorzystywanych według potrzeb użytkowników w procesach zakładania i aktualizacji zbiorów danych oraz obsługi zapytań.

System zarządzania bazą dnych jest to zestaw programów zakładania, aktualizacji, modyfikacji, wyszukiwania danych zgodnie z potrzebami użytkowników, funkcjonujący pod nadzorem systemu operacyjnego.

Języki banku danych są językami programowania służącymi do opisu baz danych, operowania na danych oraz wyszukiwania danych /realizacji zapytań użytkowników/.

Opis baz danych, ustalenie oraz opisanie powiązań między danymi stanowią łącznie opis tzw. logicznej struktury danych. Wyraźne wyodrębnienie tego typu struktury /obok fizycznej struktury/ stało się nową jakością w dziedzinie przetwarzania danych, umożliwiającą właściwe odzwierciedlenie potrzeb użytkowników. Wśród podstawowych rodzajów struktur danych wyróżnia się:

- struktury listowe /listy/, czyli struktury sekwencyjne, w których istnieje relacja pomiędzy każdym elementem poprzedzającym i następującym, z wyjątkiem ostatniego i pierwszego elementu; wyróżnia się listy jednokierunkowe /element poprzedzający wskazuje położenie elementu następującego/ i dwukierunkowe /poszczególne elementy wskazują położenie poprzedniego i następnego elementu/;
- struktury pierścieniowe /pierścienie/, które mają podobną strukturę jak listy jednokierunkowe, lecz dodatkowo ostatni element wskazuje położenie pierwszego elementu;
- struktury dendrytowe /drzewa/, które są strukturami hierarchicznymi mającymi elementy nadrzędne i podporządkowane /każdy element, za wyjątkiem najwyższego w hierarchii, jest połączony z jednym elementem nadrzędnym oraz z dowolną liczbą elementów podrzędnych/;
- struktury sieciowe /sieci/, w których każdy element może być połączony z innymi elementami w dowolny sposób.

Najczęściej spotykanymi w praktyce organizowania baz danych są struktury dendrytowe i struktury sieciowe. Mogą w nich występować zarówno powiązania jednokierunkowe, jak i dwukierunkowe.

Określona logiczna struktura danych musi być odwzorowana w pamięci systemu liczącego tworząc tzw. fizyczną strukturę danych. Przy definiowaniu tej struktury należy odpowiedzieć na szereg pytań, decydujących o wyborze konkretnego modelu organizacji zbiorów bazy danych i przyjęciu najefektywniejszych metod dostępu do zbiorów bazy danych:

- czy zestaw informacji przewidzianych do gromadzenia i przechowywania w bazie danych jest stały, czy też będzie w czasie eksploatacji ulegał poważnym zmianom,
- jaki jest wymagany czas dostępu do danych,
- czy wymagane jest wyszukiwanie danych według więcej niż jednego kryterium,
- czy jest ustalona i zamknięta lista możliwych pytań,
- jaka jest aktywność zbioru,
- jaki zakłada się tryb aktualizacji zbiorów,
- jaka jest budowa rekordów /zapisów/ i ich wielkość,
- jaka jest wielkość bazy danych

oraz jakimi urządzeniami pamięciowymi oraz jakim oprogramowaniem systemowym dysponuje się.

Możliwości techniczne decydują o określeniu fizycznej struktury bazy danych, przy czym jednym z podstawowych wymogów jest bezpośredni dostęp do danych. Niezbędne zatem staje się stosowanie pamięci dyskowej lub bębnowej. Ponadto, zakłada się, że dla właściwego funkcjonowania złożonych struktur danych przeciętnej wielkości zbioru potrzebny jest komputer o minimalnej pojemności 64 Ksłów, jako że pamięć operacyjna musi zawierać programy systemu zarządzania bazą danych, systemu operacyjnego, translatory itp.

Bank danych można więc określić jako specyficzny sposób przechowywania i wykorzystywania różnorodnych informacji, powiązanych ze sobą i przeznaczanych dla wielu użytkowników, polegający na:

a/ odejściu od klasycznych zbiorów danych do strukturalnie zmiennych baz danych;

b/ zastosowaniu oprogramowania, zwanego systemem zarządzania bazą danych, w którym opis danych jest oddzielony od procedur i od danych;

c/ stosowaniu opisu danych nie tylko z "punktu widzenia komputera", lecz również użytkownika /w postaci logicznych struktur danych, odzwierciedlających powiązania danych/;

d/ zastosowaniu języka opisu danych oraz języka operowania /manipulacji/ na danych, w tym języka zapytań;

e/ wprowadzeniu specjalnego systemu zabezpieczenia /ochrony/ danych i rekonstrukcji bazy danych.

Zasadniczym celem banku danych w zautomatyzowanych systemach dowodzenia jest:

- wyeliminowanie nadmiaru /powtarzalności/ informacji w zbiorach,
- umożliwienie szybkiego wyszukiwania informacji zgodnie z potrzebami użytkowników,
- wykorzystanie tego samego oprogramowania banku danych do różnych zastosowań /realizacji funkcji dowodzenia/.

Tak więc problematyka baz danych w systemach informatycznych związana jest z organizacją zbiorów danych w urządzeniach pamięciowych systemu liczącego i sposobami ich wykorzystywania w procesie przetwarzania danych.

W urządzeniach pamięciowych systemów liczących bity informacji grupowane są w <u>słowa.</u> przy czym słowo maszynowe jest zespołem binarnych cyfr o ustalonej liczbie bitów. Każda binarna cyfra zapamiętana jest w jednym elementarnym segmencie pamiętającym.

Miejsce na nośniku informacji, które zajmuje słowo, nosi nazwę komórki pamięci. Kolejne komórki pamięci lub ich zespoły /segmenty/, oznaczane są kolejnymi numerami, oznaczającymi adres danej komórki lub segmentu. Dla wybrania odpowiedniej komórki lub segmentu do odczytu lub zapisu, konieczne jest podanie ich adresu. Kontakt z pamięcią składa się z kilku faz:

a/ podstawianie adresu poszukiwanej komórki, wybranie tej komórki przez układ wybierania i projektowanie tej komórki do zapisu lub odczytu;

b/ odczyt lub zapis informacji;

c/ operacje końcowe /np. regeneracja informacji po odczycie/ i sygnalizacja zakończenia operacji kontaktu z pamięcią.

Cza trwania powyższych trzech faz nosi nazwę czasu cyklu pamieci. Wartość jego jest różna dla różnych typów pamięci i np. dla pamięci operacyjnych współczesnych systemów liczących wynosi od 50 nsek do 2 μ sek.

Każde urządzenie pamięciowe charakteryzują dwa podstawowe parametry:

- pojemność pamięci, czyli maksymalna liczba informacji, która może być zapamiętana w pamięci /mierzona jest zwykle liczbą słów/;

- szybkość pamięci, określona maksymalną liczbą kontaktów z pamięcią w jednostce czasu /szybkość pracy w pamięci jest odwrotnością czasu cyklu pracy pamięci/.

Pojemność pamięci decyduje o zastosowaniu komputera, ponieważ im większa pojemność, tym większe zadania mogą być realizowane przez ten komputer. Szybkość pracy pamięci decyduje natomiast o wydajności systemu liczącego.

Najczęściej używanymi pamięciami zewnętrznymi są pamięci, wykorzystujące w charakterze nośnika informacji materiały magnetyczne. W zależności od formy i konsekwencji nośnika, a także od organizacji zapisu i odczytu, wyróżnia się trzy podstawowe rodzaje pamięci magnetycznych:

- pamięci na taśmach magnetycznych,
- pamięci na dyskach magnetycznych,
- pamięci na bębnach magnetycznych.

Działanie pamięci magnetycznych opiera się na tworzeniu miejscowych zmian stanu namagnesowania powierzchni magnetycznej odpowiednio do zapisywanej informacji binarnej. Pamięci magnetyczne realizują odczyt nie niszczący, tzn. że informacja może być wielokrotnie odczytywana bez potrzeby jej regeneracji. Likwidacja starej informacji następuje przy zapisie na jej miejsce nowej informacji.

Dane na taśmach magnetycznych organizowane są w postaci zbiorów, które mogą być pamiętane na jednej lub wielu szpulach taśmy. Zbiory danych zapisywane są na taśmie w postaci ciągu bloków danych. Na taśmie, oprócz bloków z danymi znajdują się bloki specjalne. Pierwszym blokiem zapisywanym na taśmie i służącym do identyfikacji taśmy jest etykieta krążka taśmy. Na etykiecie zawarte są następujące informacje: numer seryjny taśmy, nazwa zbioru, numer kolejny krążka w zbiorze, numer generacji zbioru, okres różności zbioru, data zapisu. Etykieta krążka taśmy zawsze jest sprawdzana przez oprogramowanie operacyjne, które chroni zbiory przed ich przypadkowym zniszczeniem. Bloki danych ułożone są na taśmie szeregowo /seryjnie/. Przesyłanie danych pomiędzy pamięcią taśmową a jednostką centalną realizowane jest całymi blokami danych. Blok danych składa się z rekordów. Pierwsze słowo rekordu zawiera liczbę określającą ilość słów w rekordzie. Znaki

rekordu pogrupowane są w pola. Każdy rekord może posiadać pole w postaci jednego lub więcej znaków, który wyróżnia go w całym zbiorze. Pole takie nosi nazwę klucza rekordu.
Na taśmach magnetycznych możliwe jest przechowywanie dużych zbiorów danych i programów. Dostęp do informacji na taśmach wymaga z reguły znacznych odcinków czasowych, rzędu sekund, a nawet i minut. Zaletą pamięci taśmowych jest natomiast niski koszt nośnika i składowania danych na taśmie w porównaniu z innymi typami nośników.

W procesie przetwarzania danych wykorzystuje się taśmy, na których zapisane są:

a/ zbiory z danymi wejściowymi do przetwarzania, które powstają w okresie wczytywania maszynowych nośników informacji lub powstają w wyniku bezpośredniego przygotowania wyników na taśmie;

b/ zbiory podstawowe, ze stale przechowywanymi danymi, które poddaje się okresowej aktualizacji w procesie przetwarzania;

c/ zbiory robocze, które istnieją tylko w czasie trwania przebiegu;

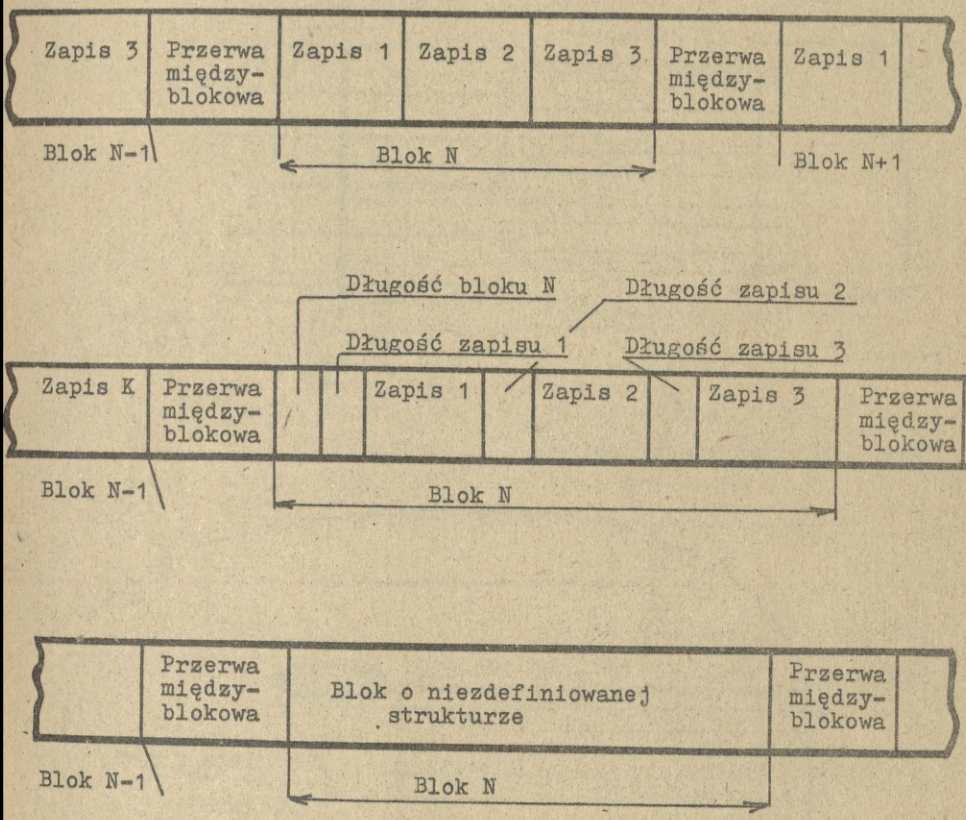
d/ zbiory z wynikami pośrednimi, które powstają na pewnym etapie procesu przetwarzania i stanowią np. zbiory wejściowe do następnego etapu obliczeń.

W pamięciach dyskowych nośnik informacji nanoszony jest po obu stronach dyska. Dyski organizowane są w zespoły zwane pakietamidysków. W pakietach tych dyski ustawione są jeden na drugim na wspólnej osi. Pomiędzy dyskami jest dostęp, który wykorzystywany jest do wprowadzania głowic, umieszczonych na ruchomych ramionach. Pamięci dyskowe wykorzystywane są jako pamięci wymienne i jako pamięci niewymienne /stałe/.

Najczęściej stosowane są dwa typy pakietów dysków:

- pakiety sześciodyskowe o pojemności 8 Mznaków,
- pakiety jedenastodyskowe o pojemności 30 lub 60 Mznaków.

Na każdej powierzchni czynnej znajdują się współśrodkowe ścieżki, na które dokonywany jest zapis danych. Informacja na ścieżkach rozmieszczona jest w formie bloków danych. Długość bloków danych może być stała /np. w systemie ODRA 1300 długość bloku wynosi 512 znaków/ lub zmienne /np. w systemie RIAD/. W projektowaniu zbiorów danych na dyskach stosowane są pojęcia: zbiór, obszar, cylinder, porcja. Cylinder tworzą bloki na pakiecie dysków, które mogą być odczytane lub zapisane bez ruchu głowic. Czas dostępu do dowolnej informacji w obrębie jednego cylindra wynosi średnio 12,5 msek. Obszar tworzą kolejne cylindry pakietu dysków. Zbiór może być zapisany na jednym lub w kilku obszarach. Porcje tworzą bloki, które można zapisać/odczytać za pomocą jednego rozkazu. Rozmiar porcji jest stały dla każdego zbioru i może wynosić: 1,2, 4 lub 8 bloków, Seryjna organizacja zbiorów w pamięciach dyskowych jest bardzo podobna do seryjnej organizacji danych na taśmie magnetycznej, gdyż rekordy w zbiorze seryjnym zapisane są kolejno /jeden za drugim, w jednym obszarze bez przerw, do końca zbioru/. Zbiory sekwencyjne, w odróżnieniu od zbiorów seryjnych, są uporządkowane według klucza i posiadają porcje nadmiarowe. Sekwencyjną organizację zbiorów nazywa się również metodą tablic indeksowych lub metodą sekwencyjno-indeksową.

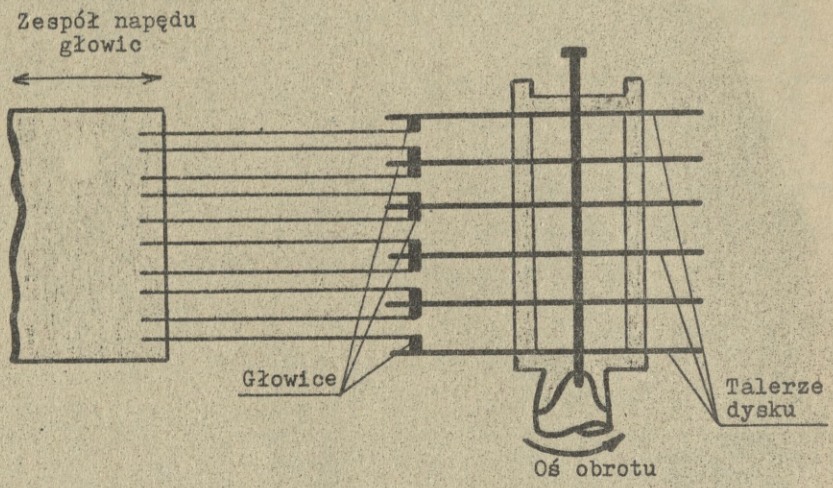


Rys.6.Definicje zapisów na taśmie magnetycznej:

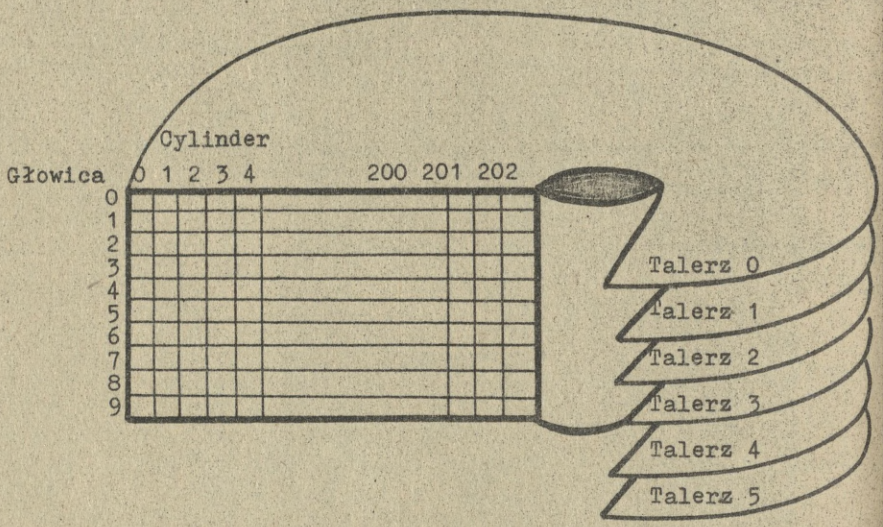
a/ zapis o stałej długości

b/ zapis o zmiennej długości

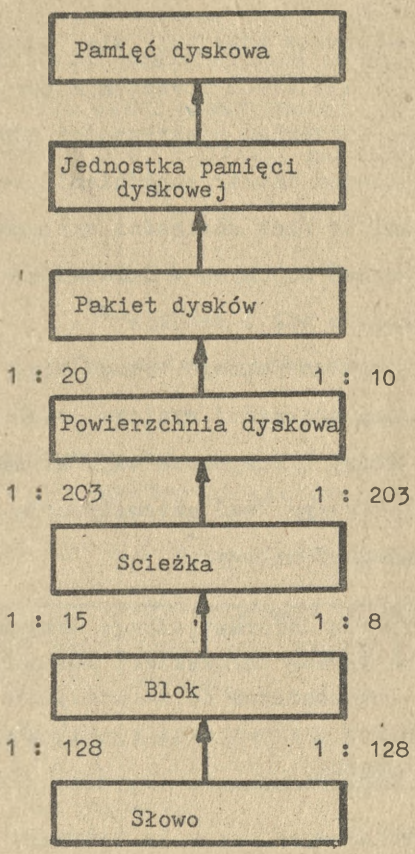
c/ zapis o strukturze niezdefiniowanej.



Rys.7. Schemat konstrukcji pamięci dyskowej



Rys.8. Schemat numeracji cylindrów i głowic pamięci dyskowej



Rys.9. Schemat struktury fizycznej pamięci dyskowej.
/Liczby umieszczone obok strzałek informują o stosunku między elementem podrzędnym a nadrzędnym dla dwóch modeli jednostek pamięci dyskowej EDS - 8 /po prawej stronie/ EDS - 30 /po lewej stronie/, np. ścieżka zawiera 8 bloków dla EDS - 8, 15 bloków dla EDS - 30 /.

7. Ocena efektywności systemów informatycznych

Jako podstawowe założenie przyjmuje się, że analiza i ocena zastosowań informatyki w systemach dowodzenia powinna być podporządkowana nadrzędnym kryteriom efektywności dowodzenia wojskami. Jedno z popularnych ujęć istoty zagadnienia brzmi: "Efektywność dowodzenia wojskami to ogół zdolności systemu dowodzenia do zapewnienia wykonania zadań bojowych w nakazanych terminach oraz przy najmniejszym nakładzie sił i środków"[x/].
Wobec tego od systemu informatycznego wymaga się, aby wpływał na wzrost wymienionych wyżej zdolności systemu dowodzenia.
System informatyczny, który nie spełnia tego wymagania jest, po prostu, systemem nieefektywnym, zaś automatyzacja systemu dowodzenia - przedsięwzięciem niecelowym.

> Efektywnością systemu informatycznego nazywamy jego cechę systemową, która wyraża całokształt zdolności technicznych i programowych - niezbędnych do zaspokojenia potrzeb informacyjnych użytkowników systemu oraz spełnienia ich wymagań ilościowych i jakościowych.

Problem efektywności wojskowych systemów informatycznych znajduje swój wyraz w odpowiedzi na następujące ogólne pytanie: "W jakim stopniu zaspokojenie potrzeb informacyjnych organów dowodzenia i spełnienie ich wymagań przez system informatyczny wpływa na wzrost efektywności dowodzenia?"

Poszukiwanie odpowiedzi na powyższe pytanie jest głównym celem badań w dziedzinie analizy i oceny efektywności systemów

---

x/ "Automatyzacja i mechanizacja systemów kierowania w wojsku", zeszyt 1/38/, wyd. Sztabu Generalnego WP, 1971, s. 15.

informatycznych. Badania te polegają, między innymi, na tworzeniu adekwatnych /w sensie odzwierciedlenia rzeczywistych efektów informatyki/ i logicznie poprawnych wskaźników oceny efektywności oraz na opracowaniu metod pozwalających na ilościowe wyrażenie ich wartości. Wskaźniki oceny efektywności powinny spełniać szereg wymagań, do których można zaliczyć następujące[x/]:

- wskaźników oceny nie może być zbyt wiele,
- wskaźniki powinny rzeczywiście mierzyć analizowane zjawiska /procesy/,
- wskaźniki powinny uwzględniać istotne cechy ocenianego systemu i jego otoczenia,
- wskaźniki powinny krytycznie reagować na zmiany wartości podstawowych parametrów ocenianego systemu,
- wskaźniki powinny być efektywne w sensie statystycznym,
- wskaźniki powinny być zrelatywizowane do podstawowych celów działania systemu nadrzędnego /np. system dowodzenia w wypadku oceny systemu informatycznego/,
- wskaźniki powinny umożliwiać ocenę wielopoziomową /według różnych kryteriów cząstkowych/ oraz powinny nadawać się do konstruowania oceny globalnej,
- wskaźniki powinny umożliwiać zarówno ocenę retrospektywną /tzw. ex post/, jak i ocenę prospektywną /tzw. ex ante/.

Dla współczesnych systemów dowodzenia przyjmuje się następujące kryteria oceny efektywności:

a/ kryterium skuteczności dowodzenia lub trafności podejmowanych decyzji w określonych warunkach działania;

b/ kryterium reaktywności, czyli szybkiego i zdecydowanego reagowania na określone zmiany sytuacji i warunków działania wojsk;

---

x/ Zob. P.Sienkiewicz: Teoria efektywności systemów kierowania. T.2 - Problemy efektywności działania. ASG WP 1979.

c/ kryterium żywotności, czyli zdolności do ciągłego działania organów dowodzenia i szybkiego otwarzania zdolności utraconych w walce;

d/ kryterium gotowości, czyli stałego utrzymywania zdolności systemu dowodzenia na poziomie pozwalającym na sprawne jego funkcjonowanie w warunkach bojowych.

W procesie analizy i oceny efektywności systemów informatycznych dąży się więc do ilościowego wyrażenia stopnia wpływu funkcjonowania tych systemów na wzrost skuteczności, reaktywności, żywotności i gotowości systemów dowodzenia. Ocena natomiast może dotyczyć zarówno okresu przeszłego działania, jak i działania przyszłego.

Rzeczywiste i potencjalne efekty systemów informatycznych podzielić można na trzy rodzaje: efekty bezpośrednie, efekty pośrednie i efekty pochodne. Efekty bezpośrednie są rezultatami, jakie występują "wewnątrz" systemu informatycznego, a więc wiążą się przede wszystkim z poprawą sprawności procesów zbierania, przesyłania, przetwarzania, przechowywania i udostępniania użytkownikom informacji oraz wyrażają np. wzrost szybkości i dokładności obiegu informacji lub obniżkę kosztów realizacji procesów informacyjnych. Efekty pośrednie obejmują te rezultaty funkcjonowania systemów informatycznych, które wyrażają się w poprawie działania organów dowodzenia, a przede wszystkim w zwiększeniu trafności podejmowania decyzji dowódczych, skuteczności planowania operacji, ekonomiczności użycia sił i środków itp. Efekty pochodne obejmują te skutki funkcjonowania systemów informatycznych, które występują zarówno w nadrzędnych /w stosunku do badanego/ systemach dowodzenia /systemach wyższego szczebla/, jak i w otoczeniu badanego systemu dowodzenia /np. w systemach sąsiadów, jednostek współdziałających itp./. Ze względu na możliwość ilościowego wyrażenia powyższe rodzaje efektów można podzielić na efekty wymierne, trudno

wymierne i niewymierne. Większość efektów bezpośrednich to efekty wymierne, zaś efekty pochodne, to głównie efekty niewymierne. Trudnowymiernymi i niewymiernymi efektami systemów informatycznych są np.: zwiększenie operatywności dowodzenia, zwiększenie efektywności wykorzystania potencjału bojowego, skrócenie czasu opracowywania informacji przez organa dowodzenia i zwiększenie ich zakresu, odciążenie kadry dowódczo-sztabowej od prac "rutynowych" na rzecz pracy koncepcyjnej itp.

Z punktu widzenia usprawniania dowodzenia istotnymi efektami są[x/]:

- skrócenie czasu otrzymywania danych wynikowych,
- otrzymywanie jakościowo nowych danych w dowolnych przekrojach za dowolny okres i na żądanie w procesie podejmowania decyzji,
- stworzenie możliwości przeszukiwania obszaru dopuszczalnych rozwiązań i wyboru optymalnego wariantu w dowodzeniu wojskami,
- stworzenie dogodnych warunków do prognozowania gotowości bojowej wojsk,
- zwiększenie sprawności i operatywności funkcjonowania organów dowodzenia,
- zwiększenie oddziaływania na podległy personel w procesie dowodzenia i wykonywania zadań,
- zwiększenie możliwości dokonywania wszechstronnych analiz na podstawie uzyskanych nowych wieloprzekrojowych zestawień danych dotyczących działań bojowych,
- stworzenie możliwości bieżącego informowania dowództwa o realizacji zadań podległych jednostek,
- możliwości automatyzacji kontroli limitów i normatywów oraz uporządkowanie bazy normatywnej,

---

x/ Zob.: "Metoda obliczania kosztów projektowania i oceny efektów systemów informatycznych", wyd. Sztabu Generalnego WP, Warszawa 1978.

- ujednolicenie dokumentów źródłowych,
- wzrost szczegółowości, wiarygodności i dokładności informacji itp.

Załóżmy, że w systemie informatycznym realizowanych jest N rodzajów zadań obliczeniowych, a przeciętna ilość realizacji n-tego zadania w cyklu dowodzenia wynosi λ_n, n = 1,2,..., N. Jeżeli są znane charakterystyki realizacji tych zadań w "tradycyjnym" systemie informacyjnym, tj. takim systemie, w którym nie stosuje się środków informatyki, to jako wskaźnik oceny efektywności systemu informatycznego przyjąć można wartość oczekiwaną pomyślnie zrealizowanych przez system zadań obliczeniowych:

$$N_o = \sum_{n=1}^{N} p_n /T_n/ \lambda_n$$

gdzie: $p_n /T_n/$ - prawdopodobieństwo pomyślnej realizacji n-tego zadania w wymaganym czasie T_n - krótszym niż w systemie "tradycyjnym".

W ten sposób ocena systemu została zrelatywizowana do poziomu, który określają rezultaty funkcjonowania "tradycyjnego" systemu. Przyjmuje się jednakże, że znane są rozkłady prawdopodobieństwa czasów realizacji poszczególnych zadań, zarówno w systemie "tradycyjnym", jak i w systemie informatycznym.

Chcąc wyrazić stopień pomyślnie zrealizowanych zadań przez system informatyczny w cyklu dowodzenia można posłużyć się następującym wskaźnikiem:

$$\eta = \frac{N_o}{\sum_{n=1}^{N} \lambda_n} \cdot 100$$

Przykład

Rozpatruje się trzy rodzaje zadań obliczeniowych, dla których określono wymagania: T_1 = 10 min, T_2 = 20 min, T_3 = 30 min. Zadania te są realizowane w cyklu dowodzenia z częstością: $\lambda_1 = 5$, $\lambda_2 = 3$, $\lambda_3 = 1$. W wyniku analizy procesów informacyjnych określono prawdopodobieństwa pomyślnie zrealizowanych zadań: $p_1/10/ = 0{,}75$, $p_2/20/ = 0{,}85$, $p_3/30/ = 0{,}95$. Bezwzględny wskaźnik efektywności systemu informatycznego ma wartość:

$$N_o = 7{,}25$$

a wskaźnik względny:

$$\eta = 80\%$$

Wynika stąd, że w systemie informatycznym pomyślnie zostanie zrealizowanych przeciętnie 7,25 zadań, czyli system jest efektywny w 80%. Oznacza to, że przeciętnie 80% zadań jest przez system informatyczny realizowanych "lepiej", tj. krócej, niż w systemie "tradycyjnym". Załóżmy, że w wyniku doskonalenia procesów informacyjnych uzyskano poprawę efektywności, np. $p_1/10/ = 1$. Wtedy $N_o = 8{,}5$ oraz $\eta = 94\%$ czyli nastąpił przyrost efektywności o ok. 18%.

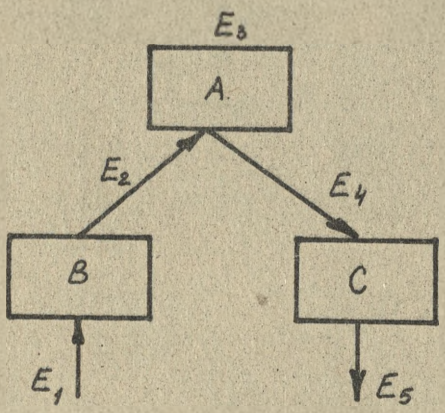
Podstawowymi wskaźnikami oceny efektywności systemów informatycznych wyrażającymi kryterium techniczno-eksploatacyjne są: niezawodność systemu, czyli prawdopodobieństwo znajdowania się systemu w stanie sprawności technicznej przez określony czas oraz gotowość techniczna systemu, czyli średnia część czasu eksploatacji, podczas którego system był zdolny do wykonywania planowanych zadań.

Przykład

Załóżmy, że w modelu systemu informatycznego wyróżniono trzy podstawowe podsystemy /A,B,C/: system wprowadzania infor-

macji /B/, system przetwarzania informacji /A/ i system wyprowadzania informacji /C/. Dana jest niezawodność poszczególnych podsystemów, czyli prawdopodobieństwo poprawnej pracy w okresie 100 g. Wynosi ona odpowiednio: $R_A = 0,9$; $R_B = 0,8$; $R_C = 0,7$. Rozpatruje się pewne charakterystyczne stany funkcjonalne systemu, którym przyporządkowano pewne wagi wyrażające znaczenie uzyskiwanych w nich efektów, a mianowicie:

$E_1 = 0,3$ - wprowadzanie informacji do B,
$E_2 = 0,2$ - przesyłanie informacji z B do A,
$E_3 = 0,3$ - przetwarzanie informacji w A,
$E_4 = 0,1$ - wyprowadzanie informacji z A do C,
$E_5 = 0,1$ - wyprowadzanie informacji z C.



W tabeli przedstawiono metodę analizy efektywności systemu. Kolumna 1 zawiera numery stanów systemów, kolumna 2 - opis stanów /przyjęto nstp. oznaczenia: A - stan sprawności podsystemu przetwarzania informacji, $\bar{A}$ - stan niesprawności itd./, kolumna 3 - postać wyrażenia pozwalającego na określenie prawodpodobieństwa znajdowania się systemu w chwili t w kolejnych stanach, a kolumna 4 - postać wyrażenia na wielkość efektów uzyskanych w poszczególnych stanach.

Nr stanu j	Opis stanu j	$P_j/t/$	E_j
1	2	3	4
1	ABC	$R_A\ R_B\ R_C$	1
2	$AB\bar{C}$	$R_A\ R_B\ /1 - R_C/$	$E_1 + E_2 + E_3$
3	$A\bar{B}C$	$R_A\ /1 - R_B/\ R_C$	$E_3 + E_4 + E_5$
4	$A\bar{B}\bar{C}$	$R_A\ /1 - R_B//1 - R_C/$	E_3
5	$\bar{A}BC$	$/1 - R_A/\ R_B\ R_C$	$E_1 + E_5$
6	$\bar{A}B\bar{C}$	$/1 - R_A/\ R_B\ /1 - R_C/$	E_1
7	$\bar{A}\bar{B}C$	$/1 - R_A//1 - R_B/\ R_C$	E_5
8	$\bar{A}\bar{B}\bar{C}$	$/1 - R_A//1 - R_B//1 - R_C/$	0

Dla wyrażenia przeciętnej efektywności systemu informatycznego przyjęto funkcję

$$E = \sum_{j=1}^{8} P_j/t/\ E_j$$

Po podstawieniu wartości liczbowych otrzymano wynik końcowy: E = 0,79, czyli efektywność systemu wynosi 79%.

W analizie efektywności systemów informatycznych istotną rolę odgrywają wskaźniki ekonomiczne ze względu na wysokie koszty technicznych i programowych środków informatyki oraz wysokie koszty własne systemów. Całkowite koszty własne systemu informatycznego dzieli się na:

- koszty projektowania i wdrażania systemu
- koszty eksploatacji systemu

Koszty projektowania i wdrażania obejmują:

- koszty osobowe,

- koszty techniczne /koszty zastosowania środków informatyki w procesie projektowania i wdrażania systemu, koszty zakupu dodatkowego sprzętu itp./,
- koszty specjalne /koszty szkolenia personelu użytkownika, koszty druku formularzy danych źródłowych, koszty powielenia dokumentacji systemu itp./.

Koszty eksploatacji systemu obejmują:

- koszty tworzenia maszynowych nośników informacji oraz
- koszty eksploatacji zestawu komputerowego.

W skład łącznych kosztów użytkowej eksploatacji systemu informatycznego wchodzą:

- roczny odpis amortyzacji łącznych kosztów projektowania i wdrażania systemu
- ogólny roczny koszt eksploatacji systemu.

Przyjmuje się, że tzw. cząstkowe efekty wymierne systemu informatycznego mogą tworzyć:

a/ efekty w dziedzinie podniesienia efektywności kierowania /np. oszczędności w szkoleniu wojsk, zmniejszenie zużycia środków materiałowych, zmniejszenie zużycia papieru itp./;

b/ efekty w dziedzinie oszczędności sił i środków /np. zmniejszenie stanu zapasów sprzętu, materiałów i części zamiennych, zmniejszenie zatrudnienia, zmniejszenie pracochłonności itp./.

Jeżeli łączne cząstkowe wymierne efekty wynoszą E_W, roczny odpis amortyzacji kosztów projektowania i wdrażania systemu wynosi R, całkowite koszty projektowania i wdrażania systemu wynoszą K i łączne koszty użytkowej eksploatacji systemu wynoszą Q, to do oceny ekonomicznej efektywności systemu informatycznego przyjmuje się:

a/ ekonomiczny efekt netto:

$$O_p = E_W - R$$

b/ wskaźnik ekonomicznej efektywności systemu:

$$W_e = \frac{E_w}{Q}$$

c/ okres zwrotu nakładów:

$$Z_n = \frac{K}{O_p}$$

Przykład

W wyniku analizy efektów i kosztów systemu informatycznego określono następujące wielkości: E_w = 19,5 mln zł, K = 9 mln zł, R = 1,5 mln zł, Q = 5 mln zł.

W wyniku oceny efektywności systemu uzyskano następujące rezultaty:

O_p = 18 mln zł

W_e = 3,9

Z_n = 0,5 roku

Wynika stąd, że efekt roczny wynikający z wdrożenia i eksploatacji systemu wynosi 18 mln zł, zaś efekty są 3,9 raza wyższe od nakładów w skali roku, natomiast zwrot kosztów projektowania i wdrażania systemu może nastąpić w ciągu jednego roku jego eksploatacji. Oznacza to, że eksploatacja systemu informatycznego stwarza realne podstawy uzyskiwania wymiernych efektów.

8. Kierunki rozwoju zautomatyzowanych systemów dowodzenia w armiach państw NATO

W centrum uwagi kierownictwa politycznego i wojskowego USA w końcu lat pięćdziesiątych znalazł się problem stworzenia takiego systemu dowodzenia, który zapewniałby możliwość operatywnego podejmowania decyzji dotyczących strategii państwa w różnych krytycznych sytuacjach na arenie międzynarodowej oraz szybkiego, dokładnego i tajnego przekazywania odpowiednich zarządzeń i roz-

kazów do poszczególnych dowództw sił zbrojnych. "Główną przyczyną podjęcia tego problemu było to, że zbudowane w latach pięćdziesiątych przez poszczególne rodzaje sił zbrojnych własne zautomatyzowane systemy dowodzenia i kierowania wykazywały szereg istotnych braków, które nierzadko dyskwalifikowały przydatność danego systemu w warunkach realnego zagrożenia"[x/]. Podjęcie tego problemu spowodowało realizację kilku programów badawczo-rozwojowych związanych z automatyzacją systemów dowodzenia i kierowania ogniem dla naczelnego kierownictwa polityczno-wojskowego, sił lądowych, sił powietrznych i obrony powietrznej oraz sił morskich.
Należy jednak podkreślić, że pomimo znacznego rozwoju naukowo-technicznego w dziedzinie informatyki, uzyskane rezultaty w automatyzacji systemów dowodzenia są niewspółmierne do zaangażowanych sił i środków, zaś wiele programów badawczo-rozwojowych uległo znacznym zmianom w trakcie ich realizacji. Zatem efektywność systemu badań i rozwoju w dziedzinie automatyzacji systemów dowodzenia budzi poważne zastrzeżenia. Jako jedną z przyczyn tego niezadowalającego stanu wymienia się fakt przeceniania znaczenia nowoczesnej techniki informatycznej w procesie automatyzacji systemów dowodzenia. Powodowało to, że wiele postulatów i wymagań wynikających ze społecznego charakteru procesów dowodzenia i zarządzania nie zostało spełnionych, zaś informatyka stawała się niekiedy źródłem tzw. konfliktów organizacyjnych w systemach dowodzenia. Oprócz wymienionych zjawisk należy pamiętać o znacznym postępie naukowo-technicznym i technologicznym w dziedzinie systemów informatycznych i teleinformatycznych występującym od wielu lat w państwach NATO.

---

x/ Zob. J.Nowicki: Zautomatyzowane systemy dowodzenia i kierowania w armiach zachodnich. MON 1972.

Do ważniejszych zautomatyzowanych systemów dowodzenia połączonych sił zbrojnych NATO na Europejskim Teatrze Wojny można zaliczyć:

- system kierowania siłami i środkami tzw. pogotowania jądrowego SCARS;
- zintegrowany system dowodzenia obroną powietrzną NADGE.

System SCARS przeznaczony do kierowania w okresie pokoju i wojny środkami napadu jądrowego "sił dyżurnych" jest wykorzystywany przez naczelne dowództwo połączonych sił zbrojnych NATO w Europie. Do podstawowych jego zadań należy:

- opracowywanie aktualnych danych o stanie gotowości bojowej własnych środków napadu jądrowego,
- przekazywanie rozkazów wprowadzających wyższe stany gotowości bojowej oraz kierowanie użyciem środków jądrowych,
- sterowanie zaopatrywaniem w amunicję jądrową.

System NADGE /NATO Air Defence Ground Environment/ jest przeznaczony do wykrywania, identyfikacji i śledzenia celów powietrznych oraz naprowadzania na wykryte cele samolotów przechwytujących i rakiet plot typu NIKE i HAWK. System powstał w 1973 r. na bazie narodowych systemów dowodzenia niektórych państw NATO i swym zasięgiem obejmuje 11 państw. Obszar państw NATO /oprócz Islandii i Portugalii/ podzielono organizacyjnie na: strefy, rejony i sektory, w których zorganizowano:

- ośrodki operacyjne strefy, rejonu i sektora,
- ośrodki wykrywania i naprowadzania /CRC/,
- posterunki wykrywania i naprowadzania /CRP/.

Dowodzenie odbywa się z ośrodków operacyjnych rejonów przez ośrodki sektorów. Ośrodek operacyjny sektora kieruje bezpośrednio wszystkimi siłami i środkami znajdującymi się na jego obszarze.

Informacje o położeniu wykrytych przez posterunki radiolokacyjne celów przesyłane są automatycznie do ośrodków operacyjnych sektora, rejonu, strefy oraz do sztabu połączonych sił zbrojnych NATO. Istnieją łącznie 84 posterunki radiolokacyjne, wśród których 37 tzw. CRC posiadają ośrodki przetwarzania danych. Ośrodek wykrywania i naprowadzania CRC /Control and Reporting Center/ jest wyposażony w EMC typu H-3118M firmy HUGHES, pracującą w czasie rzeczywistym.

Ośrodek ten służy do:

- obliczania na podstawie danych z posterunków wykrywania: kursu, prędkości, wysokości celu;
- rozpoznawania przynależności celów za pomocą identyfikatorów "swój-obcy" oraz w oparciu o zawarte w pamięci EMC dane dotyczące planowanych przelotów;
- alarmowanie dowództwa ośrodka w wypadku, gdy samolot przelatujący zmieni ustalone w planie lotów parametry itp.,
- wyboru optymalnego wariantu użycia środków bojowych;
- przekazywania danych o celach do sąsiednich CRC i ośrodka sektora.

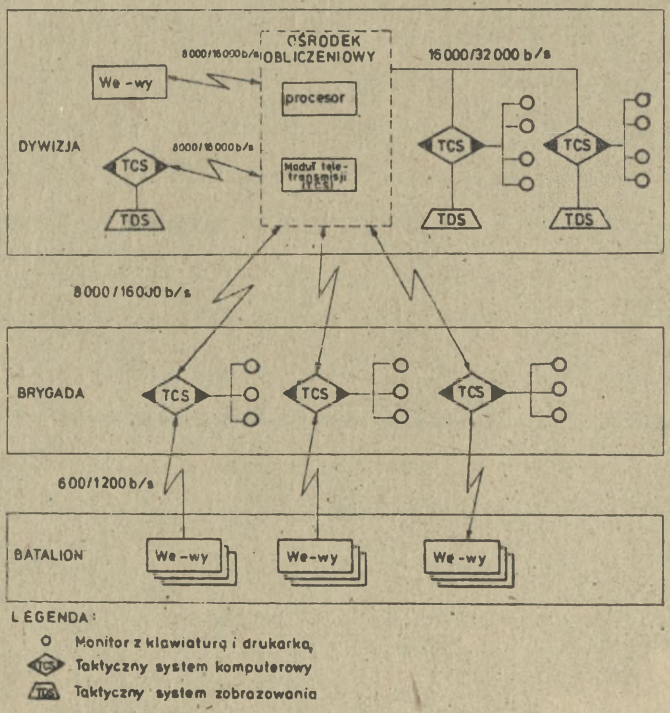
System NADGE jest zorganizowany tak, aby dowództwo każdego szczebla dowodzenia dysponowało w każdej sytuacji, w sposób automatyczny, jednakowym obrazem sytuacji ogólnej i wycinkowej. System może jednocześnie wykryć, śledzić i przechwycić ok. 1100 celów, lecących z prędkości ok. 2 Ma, na wysokości od 50-100 do 30000 m. Do 1981 r. do systemu NADGE zostaną wprowadzone elementy systemu powietrznego AWACS /Airborne Warning and Control System/, którego istota polega m.in. na umieszczeniu na samolocie transportowym Boeing 707 /E-3A/ stacji radiolokacyjnej obserwacji okrężnej wraz z komputerowymi urządzeniami służącymi do analizy, zobrazowania i przekazywania drogą radiową sytuacji powietrznej

oraz naprowadzanie lotnictwa myśliwskiego. Głównym zadaniem systemu jest zapewnienie wczesnego wykrywania i ostrzegania przed napadem powietrznym, szczególnie na bardzo małych i małych wysokościach oraz naprowadzania samolotów myśliwskich na cele powietrzne. Ponadto, samoloty AWACS wyposażone w dodatkowe urządzenia mogą być użyte jako ruchome, zapasowe stanowisko dowodzenia lotnictwem taktycznym i naprowadzania samolotów na cele naziemne i nawodne.

W 1971 r. powstał projekt wyposażenia w sprzęt informatyczny systemów przetwarzania danych dla szczebli taktycznych ARTADS /Army's Tactical Data Systems/. Najważniejszym celem było skoordynowanie poczynań projektowych i wdrożeniowych dla osiągnięcia niezbędnej jednolitości /tzw. kompatybilności/ wyposażenia technicznego zautomatyzowanych systemów dowodzenia. W ramach tego projektu określono szczegółowe wymagania dla:

- systemu kierowania ogniem szczebla taktycznego TACFIRE /Tactical Fire Direction System for the Army in the Field/,
- systemu dowodzenia szczebla taktycznego TOS /Tactical Operations System/,
- systemu dowodzenia i kierowania obroną powietrzną ADCCS MISSILE MINDER /Air Defense Command and Control System/.

Powyższe systemy oparte są na zestawie sprzętu komputerowego TCS /Tactical Computer System/. W 1974 r. Dowództwo Służb Elektronicznych Armii USA /U.S.Army Electronics Command/ podpisało kontrakt z wydziałem Librascope firmy Singer, która w 1977 r. zobowiązała się dostarczyć dwa prototypowe systemy TCS. W typowej konfiguracji system TCS realizuje przesyłanie i przetwarzanie danych w hierarchicznej strukturze dowodzenia. Na szczeblu brygady i dywizji jako urządzenia końcowe wykorzystuje się monitory ekranowe z klawiaturą i drukarki. Integralną częścią systemu są urządzenia



Rys. 10. Taktyczny system informatyczny armii USA w dywizyjnym systemie dowodzenia.

transmisji danych, pracujące z szybkością odpowiednią do wielkości strumieni danych przesyłanych pomiędzy batalionem, brygadą i dywizją. TCS jest modułowym, uniwersalnym systemem przetwarzania, zobrazowania i transmisji danych. Może być eksploatowany zarówno w warunkach polowych, jak i stacjonarnych, bowiem jego konstrukcja pozwala na zamontowanie w wozach dowodzenia M577, lub transporterach S-250 i S-280 oraz w warunkach stacjonarnych np. w schronach. Może być, ponadto, wykorzystywany niezależnie od innych systemów informatycznych, a także może współpracować z systemami podobnego typu oraz ze stacjonarnymi ośrodkami obliczeniowymi.

System TCS składa się z modułów, które tworzą bloki funkcjonalne:

- procesor centralny,
- pamięć o dostępie bezpośrednim,
- monitor ekranowy z klawiaturą alfanumeryczną,
- moduł transmisji danych,
- drukarka wierszowa,
- układy wprowadzania i wyprowadzania danych,
- pamięć zewnętrzna stała,
- układ zasilania.

Procesor - specjalna wersja komputera Rolm 1602 Ruggednova /AN/UYK-19/ - charakteryzuje się dużą szybkością obliczeń, dużymi możliwościami przetwarzania i skutecznymi środkami ochrony danych. Pamięć o dostępie bezpośrednim ma standardową pojemność 65536 słów z możliwością rozbudowy do 256 000 słów oraz może być wykonana z protekcją lub bez protekcji dostępu do zapisanej w niej informacji. Monitor wykonany jest w oparciu o technikę plazmową. Wyposażony jest w ekran o powierzchni 55,2 cm^2. Umożliwia zobrazowanie zarówno tekstów, jak i rysunków /dane taktyczne mogą być wprowadzane z mapy lub szkicu sytuacyjnego umieszczonego przed powierzchnią ekranu/.

Moduł transmisji danych umożliwia współpracę z innymi urządzeniami z szybkością od 600 bitów/sek. do 32 000 bitów/sek., z wykorzystaniem standardowych środków łączności. Umożliwia on przesyłanie danych i komunikację foniczną w 16 kanałach, a także umożliwia pracę na kierunkach radiowych lub w ośmiu sieciach radiowych. Drukarka bezuderzeniowa działa z szybkością 1200 wierszy/min. po 80 znaków w wierszu. Umożliwia także wykreślanie danych graficznych.

Oprogramowanie systemu TCS obejmuje osiem modułów:

- program zarządzający, tzw. egzekutor, który umożliwia sterowanie w czasie rzeczywistym wszystkimi urządzeniami lokalnymi i programami, transmisją danych oraz zdalnymi urządzeniami końcowymi;
- program tworzenia i weryfikacji informacji, który umożliwia operatorowi tworzenie zbiorów danych o określonych strukturach, wyświetla komunikaty błędów itp.;
- program zobrazowania, który umożliwia tworzenie różnych formatów danych wyprowadzanych na ekran monitora;
- program teletransmisji, który umożliwia odbieranie, nadawanie, przechwytywanie i wyświetlanie komunikatów stosownie do procedur transmisyjnych itp.;
- program wprowadzania danych z urządzeń lokalnych i zdalnych;
- program wydruku realizujący wydawanie wiadomości i wyników przetwarzania oraz informacji o stanie systemu;
- programy diagnostyczne i oceny funkcjonowania systemu.

Bibiloteka programów systemu zawiera:

- dyskowy system operacyjny czasu rzeczywistego,
- makroedytor,
- rozbudowany assembler,
- program "ładowania",
- translator języków FORTRANIV i ALGOL,

- tzw. cross-assembler,
- podprogramy specjalistyczne i użytkowe.

W oparciu o moduł monitora ekranowego z klawiaturą i nowoczesne technologie oparte na wykorzystaniu mikroprocesorów wykonano "terminal komputerowy szczebla taktycznego" TCT /Tactical Computer Terminal/. Jest to urządzenie końcowe systemu TCS wyposażone w mikrokomputer o strukturze bajtowej, z zewnętrzną pamięcią o pojemności 65536 bajtów /1 bajt = 8 bitów/ i możliwością współpracy z odległym komputerem /centrum komputerowym/ poprzez dwa kanały teletransmisji. Terminal ten jest przeznaczony dla niższych szczebli dowodzenia wojskami, na których nie są wykorzystywane rozbudowane systemy TCS. Każdy kanał teletransmisji może być wykorzystany bądź do transmisji danych cyfrowych /w ramach przyszłego systemu łączności TRI-TAC/, bądź do komunikacji fonicznej. Spełnia on wysokie wymagania eksploatacyjne w warunkach polowych.

TCS może być wykorzystywany w obecnych i planowanych zautomatyzowanych polowych systemach dowodzenia na szczeblu taktycznym. Wdrażanie systemu TCS w armii USA rozpoczęto w latach siedemdziesiątych, natomiast w latach osiemdziesiątych doskonalone będą jego charakterystyki i właściwości /niezawodność i żywotność urządzeń technicznych, niezawodność oprogramowania, efektywność komunikacji człowiek-maszyna itp./. Uważa się, że dopiero w latach dziewięćdziesiątych mogą nastąpić zasadnicze zmiany techniczne, programowe i strukturalne wynikające z przyszłych osiągnięć naukowo-technicznych.

Wśród podstawowych zintegrowanych systemów dowodzenia i kierowania ogniem istniejących i rozwijanych w ramach systemu USA na szczególną uwagę zasługują takie systemy, jak: SAGE, NADGE, MATRAC, AUTODIN, TACFIRE, TITA. Są to wielokomputerowe sieci

hierarchiczne dowodzenia i kierowania ogniem. Korzysta się w nich z łączności satelitarnej, radiowej naziemnej, z automatycznym kodowaniem i szyfrowaniem. Zastosowano w nich komputery standardowe o średniej i dużej skali integracji. Stosuje się w nich także, na skalę masową, elektroniczne mało- i wielkoformatowe zobrazowanie informacji. Aktualizacja zobrazowanej informacji odbywa się co 10-15 sek. Ponadto masowo wykorzystywane są mikroprocesory i minikomputery pokładowe oraz stosowane są procedury optymalizacji wyboru wariantu użycia środków bojowych.

Należy podkreślić, że o efektywności prowadzonych badań świadczy fakt, że w USA eksploatowane są nowoczesne sieci komputerowe powszechnego użytku /ARPANET, CYBERNET, NASDAQ, ALOHA, TSS IBM, N/140 IBM oraz sieci wykorzystywane w siłach zbrojnych /AUTODIN, ASGE/. Urządzenia komputerowe wykorzystywane w armii spełniają wiele specjalnych wymagań taktyczno-technicznych. Stosowane są ponadto systemy łączności satelitarnej o dużej przepustowości i o wysokiej jakości transmisji. Bardzo zaawansowane są prace nad systemem komutacji cyfrowej. Prowadzi się prace badawczo-wdrożeniowe nad pamięciami masowymi o bardzo krótkim dostępie /np. laboratorium SRI oferuje pamięć typu EBAM o pojemności 1 mln słów z czasem dostępu 30 μ sek, która może być rozbudowana modułowo do kilkudziesięciu mln słów/, nad tzw. maszyną "przepływową" CRAY o szybkości 245 mln operacji/sek. i cyklu 12,5 μ sek./ itp.

Ogólnie - wysokie tempo wdrożeń oraz poziom techniki i technologii produkcji sprzętu elektronicznego i duże nakłady na prace rozwojowe w państwach NATO, a głównie w USA, umożliwiły budowę zautomatyzowanych systemów dowodzenia, które, jak się uważa, np. w zakresie obrony powietrznej zaspokajają aktualne potrzeby organów dowodzenia.

Podobnie sytuacja przedstawia się w dziedzinie automatyzacji systemów informacyjnych Departamentu Obrony USA i instytucji centralnych z nim związanych. Brak natomiast zdecydowanego postępu w dziedzinie automatyzacji systemów dowodzenia sił lądowych zwłaszcza na szczeblu taktycznym. Uważa się, że przynajmniej do połowy lat osiemdziesiątych podstawowe zautomatyzowane systemy dowodzenia USA i innych państw NATO nie ulegną zasadniczym zmianom strukturalnym i technicznym. Stanowić natomiast będą podstawę do dalszego doskonalenia ich własności operacyjno-taktycznych i teochnicznych.

PYTANIA KONTROLNE

1. Omówić najistotniejsze cechy przemian we współczesnej nauce.
2. Omówić przedmiot cybernetyki i jej podstawowe cechy charakterystyczne.
3. Podać określenie sterowania.
4. Omówić istotę sterowania jako procesu.
5. Omówić podstawowe cele i zadania cybernetyki wojskowej.
6. Omówić przedmiot informatyki i jej podstawowe cechy charakterystyczne.
7. Omówić cle i zadania informatyki wojskowej.
8. Omówić podstawowe kierunki zastosowań informatyki w siłach zbrojnych.
9. Podać określenie informacji.
10. Omówić podstawowe założenia ilościowej teorii onformacji.
11. Omówić pojęcie entropii i jej związek z ilością informacji w wiadomości.
12. Podać przykłady wojskowych zastosowań metod ilościowej teorii informacji.
13. Omówić zasady wyznaczania ilości informacji i jej jednostki.
14. Omówić sens określenia dowodzenia jako procesu antyentropijnego.
15. Omówić związek pomiędzy efektywnością działania a ilości informacji kierującej.
16. Omówić istotę semantycznego i pragmatycznego aspektu informacji.
17. Omówić pojęcie wartości informacji.
18. Omówić techniczne determinaty jakości informacji.
19. Podać przykłady zastosowania ujęcia Charkiewicza wartości informacji.
20. Omówić klasyfikację informacji operacyjno-taktycznych.
21. Omówić charakterystyczne przypadki informowania.
22. Omówić istotę ujęcia systemowego.
23. Podać określenie systemu i reguły ścisłego stosowania tego pojęcia.
24. Omówić istotę analizy systemowej jako metody.

25. Omówić elementy ogólnego matematycznego modelu decyzyjnego.

26. Omówić elementy ogólnego modelu sytuacji decyzyjnej.

27. Sformułować problem decyzyjny i omówić jego istotę.

28. Omówić istotę zadania programowania matematycznego.

29. Omówić podstawowe typy zadań programowania matematycznego.

30. Omówić podstawowe typy matematycznych modeli decyzyjnych.

31. Podać charakterystyczne przykłady wojskowych zastosowań matematycznych metod optymalizacji.

32. Omówić możliwości wojskowych zastosowań analizy systemowej.

33. Podać określenie systemu dowodzenia i omówić jego podstawowe elementy.

34. Podać określenie systemu informacyjnego i omówić jego podstawowe elementy.

35. Podać określenie systemu informatycznego i omówić jego istotę.

36. Omówić klasyfikację systemów informatycznych.

37. Omówić istotę podstawowych typów integracji w systemach informatycznych.

38. Omówić podstawowe etapy rozwoju wojskowych systemów informatycznych.

39. Omówić podstawowe elementy strukturalne systemów informatycznych.

40. Omówić podstawowe rodzaje wojskowych systemów informatycznych.

41. Omówić podstawowe typy technicznych środków informatycznych stosowanych w siłach zbrojnych.

42. Dokonać ogólnej charakterystyki procesu projektowania systemów informatycznych.

43. Omówić podstawowe fazy projektowania systemów informatycznych.

44. Omówić podstawowe determinanty projektowania systemów informatycznych.

45. Omówić podstawowe typy dokumentacji projektowej.

46. Omówić podstawowe cechy banku danych w systemach informatycznych.

47. Omówić podstawowe typy struktur danych.

48. Omówić pojęcie efektywności systemu informatycznego.

49. Omówić podstawowe kryteria oceny efektywności systemów informatycznych.

50. Omówić podstawowe rodzaje efektów funkcjonowania systemów informatycznych i metody ich obliczania.

L I T E R A T U R A

1. Bańkowski J., Fiałkowski K — Wprowadzenie do informatyki. PWN, Warszawa 1978.

2. Bazewicz M. — Systemy informatyczne w szkole wyższej. PWN, Warszawa 1980.

3. Budowy A., Raciborska M., Skumiał P — Organizacja przetwarzania danych. WSziP, Warszawa 1976.

4. Czujew J. — Badanie operacji w wojsku. MON, Warszawa 1973.

5. Drużyhin W., Kontorow D. — Idea, algorytm, decyzja. MON, Warszawa 1975.

6. Drużynin W., Kontorov D. — Voprosy voennoj sistiemotiechniki. Voen.izd. MO SSSR, Moskva 1976.

7. Iłczuk J., Jerczyńska M. — Efektywność systemów informatycznych zarządzania. PWE, Warszawa 1979.

8. Kolupa M. /red./ — Programowanie matematyczne i ekonometria w wojsku. MON, Warszawa 1974.

9. Kozielecki J. — Psychologiczna teoria decyzji. PWN, Warszawa 1975.

10. Kuleszyński L. — Dowodzenie wojskami a cybernetyka. MON, Warszawa 1967.

11. Łoszczyłov I. — Pierspiektivy primienienija wycislitielnoj tiechniki v voennom dielie. Voen.izd. MO SSSR, Moskva 1976.

12. Mazur M. — Cybernetyka i charakter. PIW, Warszawa 1975.

13. Mynarski S. — Elementy teorii systemów i cybernetyki. PWN, Warszawa 1979.

14. Nowicki J. — Zautomatyzowane systemy dowodzenia i kierowania w armiach zachodnich. MON, Warszawa 1973.

15. Ryznar Z. — Bank danych w przedsiębiorstwach przemysłowych. PWE, Warszawa 1978.

16. Tiemnikow F., Afonin W., Dmitrijew W. — Podstawy techniki informacyjnej. WNT, Warszawa 1974.

17. Turski W. — Propedeutyka informatyki. PWN, Warszawa 1975.

Rozdział drugi

ARCHITEKTURA SYSTEMÓW LICZĄCYCH

Sprawność systemów działania ściśle wiąże się z jakością procesów informacyjno-decyzyjnych, których rozwiązanie zależy w dużym stopniu od środków stosowanych w ich przetwarzaniu. Środkami tymi są systemy informacyjne wyposażone w sprawne narzędzia i metody. Podstawowym środkiem tego rodzaju są systemy liczące.

S y s t e m l i c z ą c y, jest to zorganizowany zespół sprzętu obliczeniowego wraz z oprogramowaniem, niezbędny dla realizacji zamierzonych celów działania.

Zaprojektowanie systemu liczącego wymaga znajomości sprzętu, oprogramowania, technologii i ekonomiki zastosowania oraz określenia wzajemnych powiązań między tymi dziedzinami.

Dlatego też, a r c h i t e k t u r a s y s t e m ó w l i c z ą c y c h rozumiana jest jako sztuka projektowania, zajmująca się opisem struktury systemów na poziomie funkcjonalnie odrębnych i powiązanych ze sobą bloków oraz ich zachowaniem /działaniem/ w czasie rozwiązywania zadań /obsługi użytkowników/.

1. P o t r z e b y s t o s o w a n i a m a s z y n c y f r o w y c h w c e l o w y m d z i a ł a n i u

Współczesne systemy techniczne i społeczno-gospodarcze rozumiane jako zbiory obiektów wraz z istniejącymi między nimi relacjami charakteryzują się dużą złożonością i ciągłym wzrostem intensywności zachodzących w nich procesów. Z tego względu coraz większego znaczenia nabiera problematyka analizy, syntezy i sterowania złożonymi systemami będąca domeną nauk systemowych.

W każdej z tych sytuacji mamy do czynienia z procesami informacyjnymi /zbieraniem, przetwarzaniem i przesyłaniem informacji/ zachodzącymi w systemach.

P r z e t w a r z a n i e d a n y c h, to przekształcanie treści i postaci danych przez wykonanie operacji celem nadania im postaci korzystnych w działaniu.

Procesy informacyjne realizowane są w podsystemach informacyjnych, których efektywność zależy od ilości i jakości informacji, niezbędnych do celowego działania.

W praktycznym działaniu występują trudności w uzyskiwaniu i tworzeniu informacji. Dlatego nieodzownym jest stosowanie w tym celu środków i urządzeń informatycznych. Z możliwości zastosowania środków i urządzeń informatycznych w realizacji procesów informacyjnych wynikają korzyści w postaci zwiększonej efektywności działania. Wynika to z krótszego czasu przetwarzania większej ilości informacji przy równoczesnym zmniejszeniu nakładu pracy ręcznej /ludzkiej/.

Powszechne stosowanie maszyn cyfrowych do przetwarzania danych wynika z możliwości przedstawienia każdej informacji w postaci ciągów binarnych, łatwości technicznej realizacji układów dwustanowych oraz łatwości sterowania stanami tych układów reprezentującymi pewne informacje. Zatem filozofia stosowania maszyn cyfrowych w przetwarzaniu polega na określeniu zależności między ustalonym przejściem układu z jednego stanu w drugi, a odpowiednim do tego celu jego wysterowaniem.
Właściwe wykorzystanie maszyn cyfrowych wymaga znajomości ich struktury i organizacji oraz umiejętności przedstawiania i przekształcania danych w procesie celowego działania.

2. Podział maszyn cyfrowych ze względu na rozwój technologii i organizacji

Rozwój maszyn cyfrowych i ich zastosowanie prawie we wszystkich dziedzinach działalności ludzkiej /badania naukowe, przetwarzanie danych, sterowanie procesami technologicznymi/ jest w ścisłym związku z rozwojem technologii produkcji sprzętu liczącego, organizacji logicznej maszyn oraz sposobów ich użytkowania.

Moc obliczeniowa, niezawodność i wydajność maszyn cyfrowych jako zasadnicze kryteria ich rozwoju, spowodowały ewolucję w zakresie rozwiązań technologicznych i organizacyjnych maszyn cyfrowych, które ze stosunkowo prostych urządzeń liczących przekształciły się w wysoce sprawne systemy cyfrowe.

Ze względu na technologię dzieli się maszyny cyfrowe /komputery/ na pięć generacji:

- generację zerową, opartą na technologii przekaźnikowej /przedstawicielem tej generacji jest maszyna MARK-1/;
- generację pierwszą, której maszyny zbudowane były na technice lampowej /reprezentantami były maszyny ENIAC, URAL, ZAM-2, UMC-1/;
- generację drugą, opracowaną na technice półprzewodnikowej /przedstawicielami są IBM-7090, MIŃSK-32, ODRA-1304/;
- generację trzecią, zbudowaną na monolitycznych układach scalonych /przykładem tej grupy są maszyny IBM-360, ICL-4, ODRA-1305/;
- generację czwartą, wykorzystującą technologię układów dużej skali integracji.

Podział technologiczny maszyn cyfrowych mimo szerokiego rozpowszechnienia, nie uwzględnia istoty rozwoju maszyn, wskazując jedynie na rozwój technologii sprzętu cyfrowego. We współczesnych

rozwiązaniach maszyn cyfrowych prócz zastosowania nowych technologii ogromnego znaczenia nabierają problemy struktury i organizacji maszyn cyfrowych.

W każdym celowym działaniu, a tym samym i w realizacji procesów informacyjnych zużytkowana jest pewna ilość pracy w określonym czasie. Minimalizacja pracy i skracanie czasu jej wykonania to główne przesłanki rozwiązań strukturalno-organizacyjnych i oprogramowania maszyn cyfrowych. Rozwiązanie problemów oprogramowania podstawowego, języków programowania, szybkości przetwarzania, pojemności pamięci, trybu pracy, integracji przestrzennej i autonomii operacyjnej oraz sposobów wykorzystania maszyn cyfrowych, to tylko niektóre aspekty organizacyjne współczesnych maszyn cyfrowych.

Ewolucja struktury i organizacji maszyn cyfrowych pozwala wyróżnić klasy zwane generacjami strukturalno-organizacyjnymi, takie jak:

1/ generacja zerowa z programem na taśmie papierowej /poza maszyną/, który spełniał zasadniczą funkcję układu sterowania maszyny;

2/ generacja pierwsza, w której centralnym ogniwem w strukturze maszyny był arytmometr, zaś sterowanie realizowane było programem zapisanym w pamięci maszyny;

3/ generacja druga z autonomią urządzeń zewnętrznych w procesie przetwarzania oraz pamięcią operacyjną pełniącą funkcję centralną maszyny;

4/ generacja trzecia z asynchroniczną pracą /pełna autonomia/ arytmometru, pamięci operacyjnej i kanałów we/wy oraz nadzorczą funkcją jednostki centralnego sterowania;

5/ generacja czwarta z możliwością agregowania przestrzennego maszyn /sieci komputerowe/ środkami programowymi;

6/ generacja piąta z wieloma bezpośrednio współpracującymi jednostkami centralnymi /wieloprocesorowość/ zapewniającymi równoczesność przetwarzania wielu zadań /wieloprzetwarzanie/.

3. S t r u k t u r a i o r g a n i z a c j a s y s t e m ó w l i c z ą c y c h

Automatyczne przetwarzanie danych niosących pewne informacje /zadań/ odbywa się w systemach liczących /SL/, składających się z elementów w postaci sprzętu i oprogramowania, wyodrębnionych ze względu na określone między nimi relacje typu rozwiązań technicznych /konstrukcyjnych/ oraz programowych.

Ogół elementów SL ze względu na określone cechy /jakość/ i proporcje /ilość/ tworzy jego s k ł a d. zaś ogół relacji charakteryzujących SL tworzy jego s t r u k t u r ę.

Występujące relacje w SL powodują zróżnicowanie funkcjonalne jego elementów i zespołów, przyporządkowując im określone funkcje /role/.

Zespół relacji /struktura/ SL oraz będący ich konsekwencją podział funkcji /zasada działania/ między poszczególnymi elementami SL tworzy jego o r g a n i z a c j ę.

3.1. B u d o w a s y s t e m u l i c z ą c e g o

Sprzęt współczesnych systemów liczących posiada budowę modułową, polegającą na możliwości zestawienia ich z funkcjonalnych bloków.

S p r z ę t e m o b l i c z e n i o w y m nazywamy zespół urządzeń, takich jak jednostki centralne, pamięci, urządzenia zewnętrzne, zdolnych do przyjmowania, przetwarzania i wydawania informacji.

Poszczególne bloki stanowią odrębne jednostki, z których przez zastosowanie standaryzacji połączeń, możliwe jest tworzenie odpowiednich konfiguracji SL dostosowanych do potrzeb użytkowników.

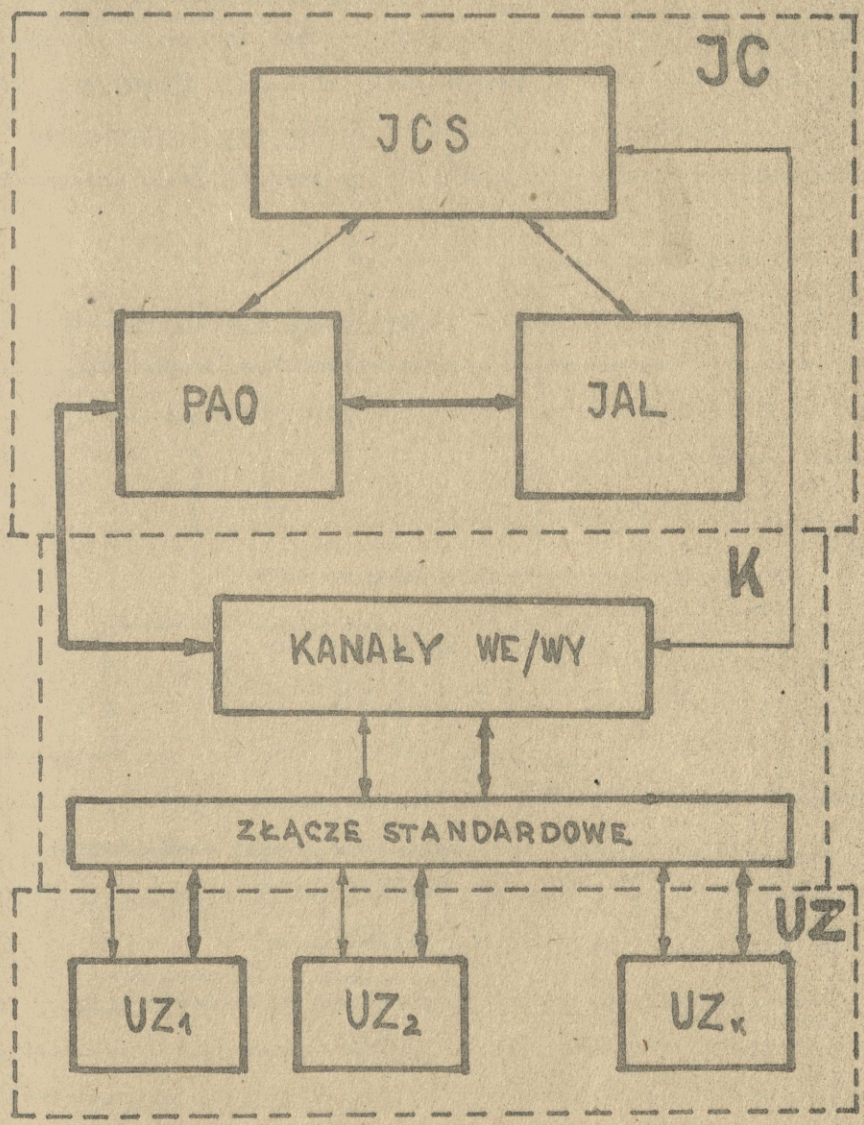
Modułowa budowa pozwala na łatwe rozszerzanie lub ograniczanie możliwości przetwarzania SL. Możliwości te wyrażać się mogą w zwiększaniu pojemności pamięci przez zestawienie odpowiedniej ilości bloków pamięci określonej pojemności np. 8 Ksłów. Podobnie można konstruować systemy wieloprocesorowe i wielomaszynowe przez łączenie większej ilości jednostek centralnych /procesorów/.

Rozwiązania modułowe SL mogą i są już częściowo stosowane na niższych poziomach złożoności zespołów. Na przykład, moduł procesora można zorganizować w reżimie pracy równoległej w zakresie przetwarzania słów określonej długości przez łączenie modułów procesorów przetwarzających słowa 8-bitowe. Innym przykładem może być modułowość sterowania dzięki zastosowaniu mikroprogramowania. Zdolność łatwego dostosowania listy rozkazów do wymagań stawianych przez system pozwala na indywidualne zwiększenie lub zmniejszenie układu sterowania zależnie od potrzeb danego zastosowania SL.

Dla osiągnięcia modułowości organizacja SL musi być starannie zaplanowana, tak aby pożądane konfiguracje były możliwe do zrealizowania. Konfiguracja SL określona jest przez fizyczną lokalizację funkcjonalnych bloków oraz możliwe drogi przesyłania informacji między tymi blokami. Drogami przesyłania informacji jest sieć łączy standardowych i linii transmisyjnych wiążących poszczególne elementy SL.

Ogólnie, w każdym SL /rys. 1/ wyróżnić można następujące moduły:

- jednostkę centralną /JC/;
- kanały we/wy /K/;
- urządzenia zewnętrzne /UZ/.



Rys. 1. Schemat blokowy systemu liczącego

Podstawowym czynnkiem modułowości SL pozwalającym na tworzenie odpowiedniej konfiguracji jest zastosowanie organizacji kanałowej. Polega to na tym, że przesyłanie informacji między JC oraz w JC /pamięć operacyjna/ a pozostałymi elementami SL odbywa się za pośrednictwem kanałów z wykorzystaniem standardowego połączenia.

3.1.1. Jednostka centralna

Głównym elementem SL jest jednostka centralna /JC/. Jednostka centralna, której organizację przedstawia rys. 2 jest zespołem środków technicznych i programowych wykonujących czynności obliczeniowe i sterujące pracą SL.

W skład JC wchodzą:

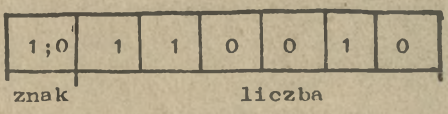
- jednostka arytmetyczno-logiczna /JAL/;
- pamięć operacyjna /PAO/;
- jednostka centralnego sterowania /JCS/.

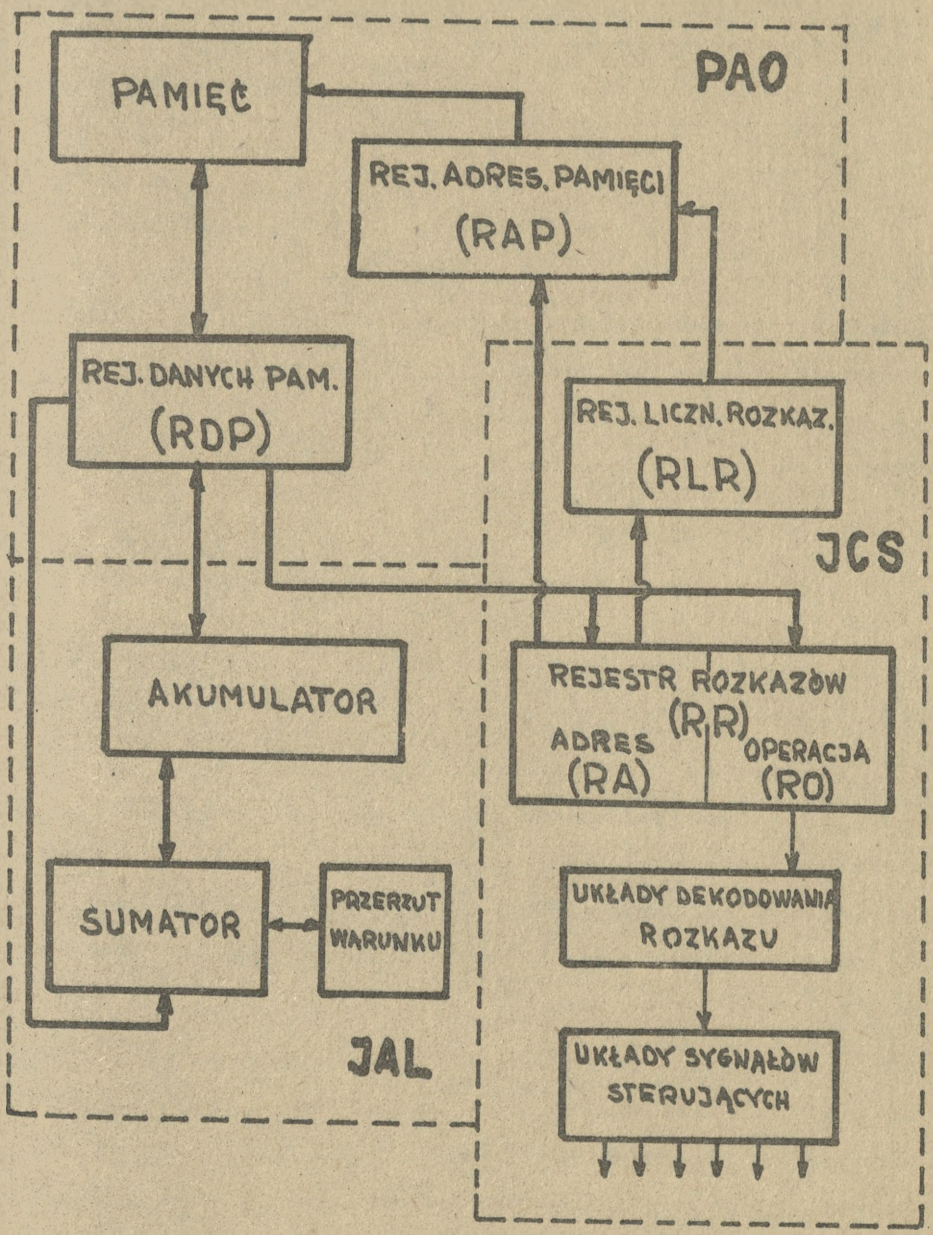
J e d n o s t k a a r y t m e t y c z n o - l o g i c z n a /JAL/ nazywana arytmometrem, wykonuje działania arytmetyczne i logiczne na liczbach systemu dwójkowego znajdujących się w rejestrach połączonych z innymi zespołami oraz między sobą liniami przesyłania. Linie te umożliwiają wymianę danych między rejestrami oraz współpracę JAL z pamięcią i kanałami we/wy.

Liczba w systemie dwójkowym przedstawia skończony ciąg znaków zerojedynkowych np. /110100111/. Przechodzenie z dziesiętnego systemu na system dwójkowy i odwrotnie odbywa się automatycznie w SL.

Liczby w systemie dwójkowym mogą być przedstawione jako stałoprzecinkowe ułamkowe i całkowite

np.

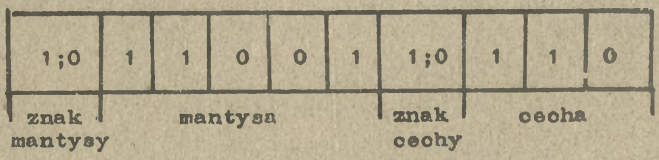




Rys. 2. Schemat jednostki centralnej

oraz zmiennoprzecinkowe w postaci mantysy i cechy /wykładnika/

np.



W większości JAL systemów liczących realizowana jest jedynie operacja arytmetyczna dodawania. Pozostałe operacje arytmetyczne jako pochodne dodawania wykonywane są przez realizację operacji logicznych i operację dodawania. Sterowanie przebiegiem tych operacji odbywa się przez układy logiczne /realizacja sprzętowa/ lub wymaga podprogramów /realizacja programowa/.

W typowej JAL można wyróżnić następujące rejestry:

- rejestr wejść przechowujący wartość pierwszego argumentu;
- rejestr akumulator przechowujący wartość drugiego argumentu i wynik operacji;
- sumator wykonujący operacje arytmetyczno-logiczne.

Z zasady JAL działa w układzie pracy równoległej, tzn. argumenty z rejestrów podawane są do sumatora równocześnie, co powoduje zwiększenie szybkości jego pracy. Pełne wykorzystanie możliwości JAL uzyskuje się przez szybkie dostarczanie nowych argumentów z pamięci operacyjnej, w której są przechowywane, do jego rejestrów.

P a m i ę ć o p e r a c y j n a /PAO/ typowych SL składa się z bloku rdzeni magnetycznych i związanych z nimi układów elektronicznych przeznaczonych do zapisu, przechowywania i odczytu informacji w postaci ciągów zerojedynkowych o określonej długości zwanych słowami lub znakami /bajtami/.

Stosowane konstrukcje wymagają użycia pojedynczego płatu dla każdego bitu słowa. Liczba płatów jest równa liczbie bitów w słowie

/długości słowa/, zaś liczba rdzeni w płacie jest równa liczbie słów, które mogą być przechowywane jednocześnie w PAO.

Liczba miejsc /komórek/ dla jednoczesnego przechowywania słów w pamięci nosi nazwę p o j e m n o ś c i p a m i ę c i.

PAO najczęściej są produkowane w postaci modułów o pojemnościach 1,4,8,16 Ksłów /1 Ksłowo = 1024 słowa/. Większe pamięci o pojemnościach kilkudziesięciu do kilkuset komórek zestawia się z modułów mniejszych, najczęściej o pojemnościach 4 i 8 Kstów.

Elementami składowymi typowej PAO są:

- blok rdzeniowy przechowujący słowa;
- rejestr adresu określający adres komórki słowa;
- dekoder wyznaczający fizyczne miejsce zawartości słów w bloku rdzeniowym;
- rejestr danych pamięci przeznaczonych do zapisu /odczytu zawartości komórek pamięci.

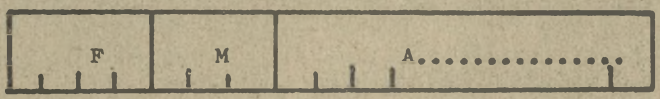
Szybkość pracy JC uwarunkowana jest czasem o y k l u p a m i ę c i czyli czasem niezbędnym na wyszukanie adresu komórki i zapisanie/odczytanie jej zawartości.

Dla skrócenia czasu cyklu pamięci wykorzystywane są pamięci stałe budowane na układach elektronicznych, których czas cyklu jest porównywalny z czasem propagacji sygnałów w JAL. Pamięci stałe posiadają małą pojemność i spełniają rolę wyspecjalizowanych rejestrów, gdzie wymiana informacji między nimi i PAO odbywa się równocześnie z pracą JAL.

Przechowywane w PAO słowa mogą reprezentować argumenty /liczby/ przetwarzane w JAL lub rozkazy programu interpretowane przez układy sterowania.

R o z k a z jest to polecenie na wykonanie określonej operacji arytmetyczno-logicznej i organizacyjnej w określonym miejscu SL.

Struktura rozkazu zazwyczaj ma postać



gdzie:

- F-część operacyjna zawierająca kod funkcji;
- A-część adresowa określająca adres argumentu w PAO, na którym będzie wykonywana operacja;
- M-część modyfikacyjna wskazująca sposób zmiany adresu zawartego w części adresowej słowa.

J e d n o s t k a c e n t r a l n e g o s t e r o w a n i a /JCS/ określa czynności zawarte w słowie rozkazowym oraz rytm i miejsce realizacji w blokach JC.

Realizacja dowolnego rozkazu przebiega w czterech krokach:

1/ odczytanie rozkazu z PAO,

2/ ustalenie adresu skutecznego, modyfikacja,

3/ wykonanie operacji określonej kodem funkcji,

4/ ustalenie adresu następnego rozkazu.

W wyniku realizacji rozkazu wytwarzane są sygnały sterujące, które są przesyłane do poszczególnych układów elektronicznych uczestniczących w wykonywaniu operacji.

W skład JCS wchodzą następujące układy i rejestry:

- rejestr licznika rozkazów określający adres komórki realizowanego rozkazu;
- rejestr rozkazów zawierający aktualnie realizowany rozkaz;
- układ dekodowania części operacyjnej rozkazu;
- układ generatorów sygnałów sterujących /mikrooperacji/ skierowanych do poszczególnych zespołów JC w zależności od realizowanej operacji;

- rejestr adresowy określający adres argumentu na którym realizowana jest operacja;
- układ modyfikacji określający fizyczny adres argumentu w PAO.

Rozwiązania JCS mogą być realizowane na drodze układowej bądź programowej.

W sterowaniu układowym generowanie sygnałów mikrooperacji odbywa się niezależnie dla każdej opeřacji listy rozkazów w autonomicznych układach sterujących. Takie podejście rozwiązań cechuje duża szybkość działania JC lecz równocześnie pociąga duże koszty i sztywność w sensie ewentualnej zmiany listy rozkazów.

Metoda sterowania programowanego polega na tym, że wydziela się zbiór mikrooperacji, które występują w operacjach i z nich zestawia się mikroprogramy realizujące dane rozkazy z listy rozkazów. Ten sposób sterowania zwany mikroprogramowym znalazł szerokie zastosowanie dzięki wprowadzeniu elektronicznych pamięci stałych na układach scalonych.

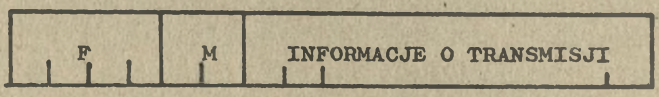
3.1.2. Kanały we/wy

W celu efektywnego wykorzystania sprzętu SL wszystkie operacje związane z przesyłaniem informacji w relacji JC - urządzenia zewnętrzne muszą odbywać się autonomicznie w układach funkcjonalnych zwanych k a n a ł a m i we/wy /K/.

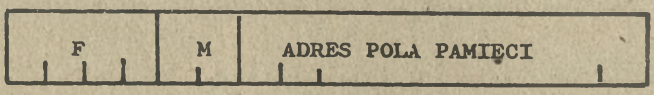
Autonomia pracy JAL i K realizowana jest w technice wstrzymań programowych. Polega to na tym, że jeśli kanały zgłoszą się do współpracy z PAO w czasie kontaktu JAL z PAO, wtedy praca K zostaje wstrzymana i podobnie, jeżeli JAL zgłosi się do współpracy w czasie kontaktu K z PAO, to praca JAL jest wstrzymana do zakończenia cyklu pamięci. Przy równoczesnym zgłoszeniu K i JAL pierwszeństwo w obsłudze mają kanały. Zgłoszenia kanałów realizowane są zgodnie

z zasadą priorytetów. Współpraca /rys. 3/ JC i K odbywa się w tzw. koordynatorze kanałów, czyli układzie sterowania współpracą, zaś współpraca urządzeń zewnętrznych i K odbywa się w układzie sterowania zwanym jednostką sterowania urządzeń, która nadzoruje pracę i adaptuje specyfikę urządzeń zewnętrznych do standardowych zasad współpracy z kanałem za pomocą jednolitego pod względem logicznym, elektrycznym i mechanicznym złącza standardowego.

Praca kanału i związanych z nim urządzeń rozpoczyna się na skutek wykonania rozkazów we/wy. Dla zainicjowania i autonomicznej pracy kanału niezbędne są informacje o transmisji, które mogą być zawarte w części adresowej rozkazów, np.



lub w polu PAO, którego adres zawarty jest w części adresowej rozkazu, np.



Przesyłanie informacji ze względu na ilość i rozmieszczenie można podzielić na:

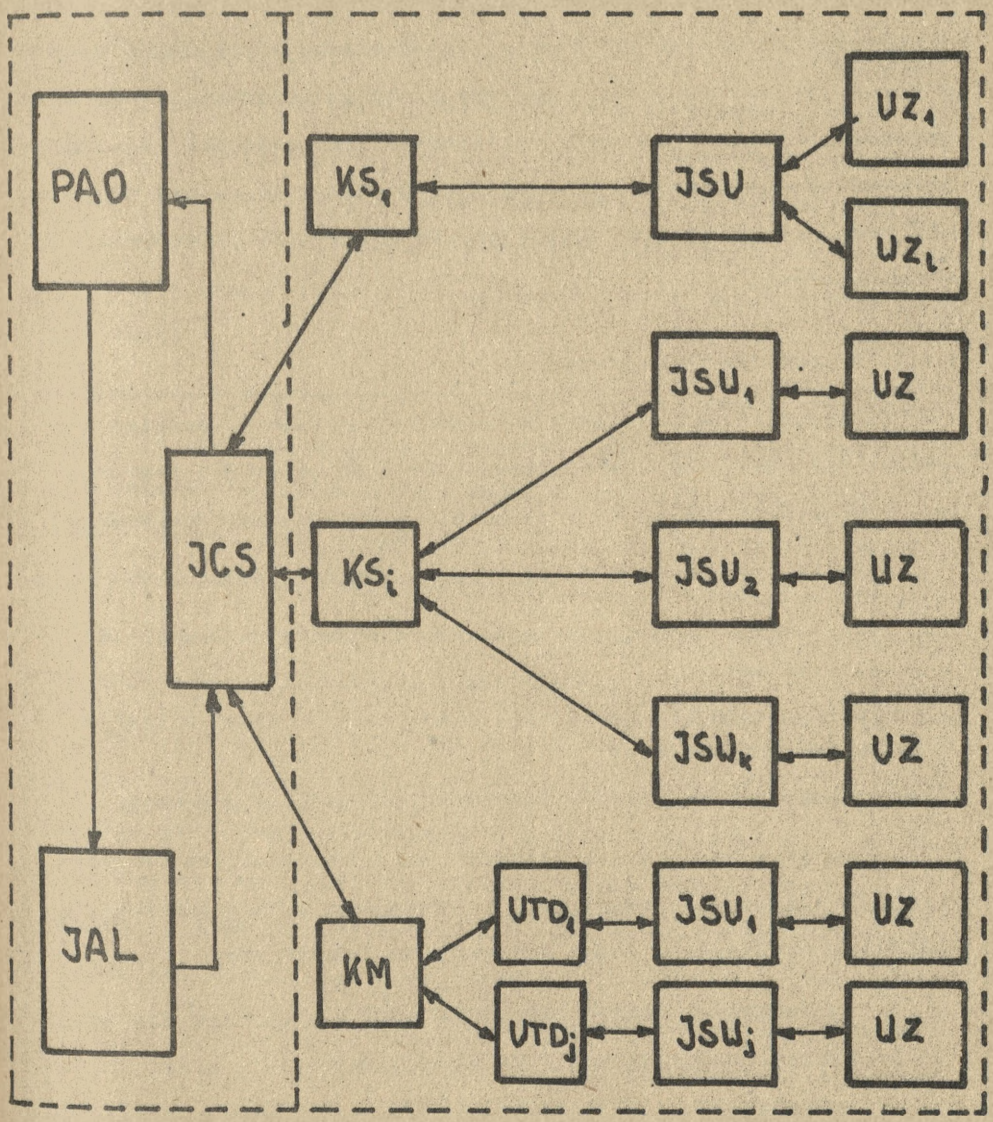
1/ - przesyłanie pojedynczych znaków;

2/ - przesyłanie jednoblokowe;

3/ - przesyłanie wieloblokowe.

Ze względu na różne sposoby współpracy kanałów z PAO i UZ mogą być kanały selektorowe znakowe i autonomiczne, kanały buforowane oraz kanały multipleksorowe.

Do kanałów może być podłączonych kilka urządzeń zewnętrznych, przy czym sterowane jest jedno urządzenie /kanał selektorowy/ lub wiele urządzeń równocześnie /kanał multipleksorowy/.



KS - kanał selektorowy
KM - kanał multipleksorowy
JSU - jednostka sterowania urządzeń
UTD - urządzenie transmisji danych

Rys. 3. Schemat blokowy kanałów we/wy

Urządzenia wolne typu dalekopisy, konsole itp., podłączane są z zasady do kanałów multipleksorowych lub selektorowych nieautonomicznych, a urządzenia średniej szybkości jak czytniki, drukarki do kanałów selektorowych znakowych i autonomicznych, zaś urządzenia szybkie typu pamięci zewnętrzne do kanałów selektorowych autonomicznych.

3.1.3. Urządzenia zewnętrzne

Do urządzeń zewnętrznych zaliczany jest sprzęt umożliwiający wymianę informacji między użytkownikiem a SL przez jej przyjmowanie, trwałe przechowywanie i wydawanie w trakcie przetwarzania.

Ogólnie można wyróżnić trzy grupy urządzeń zewnętrznych:

1/ operatorskie urządzenia zewnętrzne pośredniczące między SL i człowiekiem;

2/ pomiarowo-sterujące urządzenie zewnętrzne do pracy w trybie uwarunkowanym czasowo;

3/ zewnętrzne pamięci masowe rozszerzające pamięci operacyjne.

Grupa pierwsza zawiera wszystkie te urządzenia, które człowiek musi obserwować, obsługiwać i odczytywać z nich informację. Stąd też urządzenia te można podzielić na urządzenia:

- wprowadzania danych /urządzenia we/;
- wyprowadzania danych /urządzenia wy/;
- konwersacyjne /urządzenia we/wy/.

Urządzenia wprowadzania danych spełniają funkcję odczytywania informacji z maszynowych nośników informacji w postaci dziurkowanych kart i taśm papierowych oraz dokumentów magnetycznych, jej przetwarzanie na sygnały elektryczne i przesłanie do JC za pośrednictwem kanałów. Do klasy tych urządzeń zaliczane są wszelkiego

rodzaju czytniki, których konstrukcja zależy od sposobu przedstawienia i odczytu informacji z nośników. W najpowszechniejszym użyciu są czytniki kart i taśm papierowych oraz czytniki dokumentów magnetycznych wykorzystujące magnetyczne i fotoelektryczne zjawiska odczytu. Oprócz układów bezpośredniego odczytu urządzenia te posiadają układy elektryczne i mechaniczne podające i przesuwające nośniki.

Urządzenia wyprowadzania danych realizują funkcje odwrotne do urządzeń wprowadzania, tzn. pobierają dane z JC i po odpowiednim przetworzeniu zapisują na nośniku informacji. Klasa tych urządzeń obejmuje dużą grupę różnorodnych urządzeń takich, jak dziurkarki kart i taśm papierowych, drukarki wierszowe, monitory ekranowe, pisaki graficzne itp., które umożliwiają wyprowadzanie informacji kodowanej, słownej i graficznej.

Urządzenia konwersacyjne są urządzeniami łączącymi cechy opisanych wyżej typów, umożliwiając bezpośredni kontakt człowieka z SL. Zapewniają one wprowadzanie i wyprowadzanie danych w sposób dostosowany do możliwości człowieka. W większości urządzeń konwersacyjnych informacja wprowadzana jest za pośrednictwem klawiatury lub pióra elektronicznego /tzw. pióra świetlnego/, zaś wyprowadzana w postaci wydruku słownego lub graficznego na papierze, bądź obrazu na ekranie. Najczęściej stosowanymi urządzeniami tego typu są maszyny do pisania, dalekopisy, monitory ekranowe z przystawką wprowadzania danych - zwane alfaskopami lub grafoskopami.

W grupie drugiej zawarte są te urządzenia, które pośredniczą między JC i przyrządami pomiarowymi oraz rozsyłają dane wyjściowe do układów sterujących procesem. Urządzenia tej grupy są zwykle elementami systemów nie wymagających stałej obsługi, jak np. systemy automatycznego sterowania. Do urządzeń konwersacyjnych

zaliczyć można tzw. przetworniki analogowo-cyfrowe i cyfrowo-analogowe oraz cyfrowe przyrządy pomiarowe i sterujące.

Pojemność pamięci wewnętrznych PAO jest na ogół ograniczona względami konstrukcyjnymi i kosztowymi. Systemy liczące wymagające w przetwarzaniu dużych zbiorów informacji korzystają z urządzeń zewnętrznych grupy trzeciej zwanych pamięciami masowymi. Funkcją tych urządzeń jest przechowywanie dużych zbiorów informacji i udostępnianie ich w miarę potrzeb procesu przetwarzania. Najczęściej używanymi pamięciami zewnętrznymi SL są pamięci wykorzystujące materiały magnetyczne w charakterze nośnika informacji. Do nich zaliczyć można:

1/ pamięci taśmowe,
2/ pamięci dyskowe,
3/ pamięci bębnowe.

Działanie pamięci magnetycznych jest oparte na tworzeniu miejscowych zmian stanu namagnesowania /polaryzacji/ powierzchni nośnika do postaci ciągów zerojedynkowych. Zmiany stanu polaryzacji wywoływane są za pomocą głowic magnetycznych, które w zależności od prądu płynącego w ich uzwojeniu tworzą pole magnetyczne pod przesuwającym się nośnikiem.

Nośnikiem informacji w p a m i ę c i t a ś m o w e j jest taśma magnetyczna, której pojemność zależy od wielkości krążka i gęstości zapisu informacji w postaci znaków, których bity zapisywane są w poprzecznych rządkach. Znaki te jako informacje grupowane są w pewne bloki zwane zapisami /rekordami/, te zaś w większe jednostki zwane zbiorami, plikami lub kartotekami. Między tymi elementami są wolne pola umożliwiające rozruch i hamowanie taśmy magnetycznej. Dostęp do informacji i jej zapis jest sekwencyjny, stąd pamięć ta jest pamięcią wolną.

Pamięć taśmowa jest wyposażona w mechanizmy napędowe podające i odbierające taśmę oraz utrzymujące jej jednostajny ruch w czasie odczytu/zapisu informacji.

Dla umożliwienia zastosowania większej liczby jednostek taśmy, stosowane są adaptery kcordynujące współpracę wielu jednostek w jednym kanale we/wy.

W p a m i ę c i a c h d y s k o w y c h materiał magnetyczny nanoszony jest dwustronnie na powierzchniach dyskowych. Dyski są organizowane w pakiety cysków osadzone na jednej osi. Informacja zapisywana jest na powierzchniach dysków za pomocą głowic umieszczonych na wspólnym ramieniu, co umożliwia na równoczesne wykorzystanie powierzchni dyskowych pakietu.

Na każdej powierzchni dysku rozmieszczone są współśrodkowe ścieżki, na których jest zapisywana/odczytywana informacja sekwencyjnie bitami. Każda ścieżka podzielona jest na bloki zwane sektorami, zaś odpowiadające ścieżki wszystkich powierzchni pakietu dyskowego tworzą cylindry. Pamięci dyskowe są pamięciami adresowalnymi, zatem dostęp jest prawie swobodny.

W celu zwiększenia pojemności pamięci dyskowych stosuje się większą liczbę jednostek dyskowych podłączonych do jednego kanału za pomocą jednostki sterującej urządzeniami zewnętrznymi.

P a m i ę ć b ę b n o w a jest najstarszą pomocniczą pamięcią zewnętrzną. Nośnikiem jest wirujący walec pokryty materiałem magnetycznym. Na powierzchni rozmieszczone są ścieżki przeznaczone na zapis/odczyt informacji, podzielone na sektory.

Pamięć ta jest pamięcią o dostępie cyklicznym i dużej szybkości działania, co powoduje wykorzystywanie jej jako rozszerzonej pamięci operacyjnej.

Oprócz omówionych pamięci magnetycznych mogą być stosowane pamięci mikrofilmowe, laserowe, cienkowarstwowe i inne. Coraz częściej stosowana jest, zwłaszcza w systemach minikomputerowych, pamięć magnetyczna kasetowa.

3.2. Funkcjonowanie systemów liczących

Działanie SL można traktować jako przepływ informacji między poszczególnymi blokami funkcjonalnymi /rys. 1/ oraz między rejestrami i układami logicznymi w tych blokach.

Podstawą pracy SL jest informacja użytkowa dostarczana z zewnątrz przez urządzenia zewnętrzne i kanały we/wy do pamięci jednostki centralnej w postaci programu i danych.

P r o g r a m e m przyjęto nazywać zbiór rozkazów, w wyniku realizacji których, następuje rozwiązanie określonego zadania.

Program wykonywany jest w JC. W czasie jednego cyklu pracy z pamięci pobierany jest odpowiedni rozkaz programu, następnie rozkaz jest deszyfrowany i następuje wykonanie operacji określonych rozkazem.

W podstawowym cyklu pracy /rys. 2/ realizowanym pod kontrolą sekwencji sterującej wykonywane są następujące czynności:

1. Adres kolejnego rozkazu programu pobranego z PAO do RLR powoduje przesłanie zawartości tego rejestru do rejestru RAP.
2. Zawartość komórki PAO o adresie określonym przez zawartość rejestru RAP jest przesłana do rejestru RDP.
3. Zawartość rejestru RDP jest przesłana do rejestru rozkazów RR, przy czym część operacyjna do RO, część adresowa do RA.
4. Za pomocą układów dekodujących następuje generowanie sygnałów /mikrooperacji/, które wykonują czynności wynikające z typu operacji.

5. Jeśli rozkaz jest rozkazem skoku bezwarunkowego, to zawartość części adresowej rejestru RA zostaje przesłana do RLR i następuje koniec wykonania rozkazu.

5.1. Jeśli rozkaz jest rozkazem skoku warunkowego, to przy spełnieniu warunku, którego kod przechowywany jest w przerzutniku warunku zawartość części adresowej rejestru RA zostaje przesłana do RLR, zaś w przypadku niespełnienia warunku następuje zakończenie wykonywania rozkazu.

5.2. Jeśli w rozkazie występuje argument lub wymaga argumentu, to zawartość części adresowej rejestru RA zostaje przesłana do rejestru RAP. Argument zostaje pobrany z pamięci do rejestru RDP.

6. Zawartość rejestru RDP jest albo przesyłana do akumulatora, albo wraz z zawartością akumulatora bierze udział w operacji wykonywanej w sumatorze; po wykonaniu operacji wynik przesyłany jest do akumulatora i koniec wykonania rozkazu.

7. W wyniku normalnego zakończenia rozkazu następuje zwiększenie zawartości RLR o 1 i rozpoczyna się wykonanie kolejnego rozkazu programu.

W wyniku realizacji poszczególnych rozkazów programu wykonywany jest proces przetwarzania danych, które są sprowadzane do PAO z UZ, zaś uzyskane w rezultacie przetwarzania wyniki przesyłane są do UZ za pośrednictwem kanałów we/wy.

Po zainicjowaniu pracy K wykonywane są w nim operacje zawarte w rejestrze instrukcji kanału zgodnie z kodem operacji rozkazów we/wy, takie jak:

a/ przesyłanie danych,
b/ badanie stanu kanału lub urządzenia,
c/ sterowanie za pośrednictwem przerwań.

Przyczyną przerwań od UZ i K mogą być:

- zakończenie przesyłania danych,

- zaistnienie warunków wymagających interwencji operatora,
- wykrycie błędów w programie,
- wykrycie błędów ochrony,
- pojawienie się zgłoszenia itp.

3.3. Organizacja pracy systemów liczących

W miarę wzrostu możliwości zastosowania i wykorzystania SL w różnych dziedzinach działalności człowieka, rośnie zapotrzebowanie na SL charakteryzujące się dużą mocą obliczeniową, niezawodnością i wydajnością. Kryteria te osiągnąć można, oprócz rozwiązań technologicznych, na drodze odpowiedniej organizacji SL, a w tym i ich organizacji pracy. Od sposobu organizacji pracy SL zależy jego przepustowość.

P r z e p u s t o w o ś c i ą przyjęto określać stosunek liczby zadań do czasu, który jest potrzebny do ich wykonania za pomocą danego SL.

Na przestrzeni historii rozwoju SL wyróżnić można trzy tryby /sposoby/ organizacji pracy, a mianowicie:

1/ tryb tradycyjny,
2/ tryb wsadowy,
3/ tryb krotny.

T r y b t r a d y c y j n y charakteryzuje się tym, że jednostka centralna i zestaw urządzeń zewnętrznych jest w całości angażowany do wykonania zadania jednego użytkownika. W zadaniu czynności wprowadzania, translacji, realizacji programu i wyprowadzania wyników angażują różne elementy SL w sposób sekwencyjny. Powoduje to, że pewne zespoły i urządzenia systemu w trakcie wykonywania czynności przez inne są niewykorzystywane, a tym samym wydajność SL jest niska.

T r y b w s a d o w y jest próbą rozwiązania mającego eliminować główną wadę systemu tradycyjnego, tj. jego niski stopień wykorzystania. Tryb pracy systemu wsadowego polega na wprowadzeniu do SL zestawu zadań zwanych wsadem. Zadania ze wsadu wykonywane są kolejno bez przestojów pod nadzorem programu zarządzającego. Wyniki obliczeń wydawane są po zakończeniu każdego zadania dzięki autonomii pracy kanałów traktowanych jako jednostki /procesory/ komunikacyjne.

Współczesne systemy liczące są systemami organizującymi pracę w trybie krotnym. Cecha ta dotyczy takich pojęć jak: wieloprogramowanie, wieloprzetwarzanie i wielodostęp.

W i e l o p r o g r a m o w a n i e m nazywa się sposób pracy SL, polegający na wykonywaniu wielu programów przez etapową ich realizację w określonej kolejności, uwarunkowanej uzyskaniem maksymalnej przepustowości SL.

W systemach w i e l o p r o g r a m o w y c h wyróżnić można trzy podstawowe poziomy pracy.

Pierwszym, a tym samym najniższym poziomem wieloprogramowości są systemy realizujące równocześnie z innymi operacjami operacje we/wy. Polega to na tym, że program żądający transmisji jest zawieszany w JC i przekazany autonomicznym kanałom we/wy, a do wykonania w JC powoływany jest kolejny program.

Następnym poziomem wieloprogramowania są systemy z cyklicznym przełączaniem programów. Metoda ta polega na jednoczesnym przetwarzaniu pewnej liczby programów przez przyporządkowywanie każdemu z nich kwantu czasu, w którym program jest realizowany przez JC.

Najwyższy poziom stanowią systemy z priorytetami obsługi zadań. Priorytety szeregują wykonywane zadania do pewnych klas ważności zadań. Zadaniom z określonych klas ważności przysługuje odpowiedni czas, kolejność i sposób ich wykonania po zaistnieniu w systemie przerwania. Zadania o wyższym priorytecie rozwiązywane są przed zadaniami o priorytecie niższym.

W i e l o p r z e t w a r z a n i e m nazywamy możliwość równoczesnej realizacji wielu zadań w oddzielnych JC, mających możliwość wymiany informacji poprzez wspólną pamięć lub przez linie transmisji informacji.

W zależności od przestrzennej integracji JC mamy do czynienia z systemami wieloprocesorowymi bądź wielomaszynowymi.

Czynnikami stymulującymi rozwój systemów wieloprzetwarzających jest niezawodność oraz szybkość przetwarzania. Zwiększenie niezawodności uzyskuje się przez wprowadzenie rezerwowych JC, które mogą wykonywać te same zadania co zasadnicze JC, bądź przechodzić do wykonywania zadań w przypadku awarii elementów zasadniczych. Zwiększenie szybkości przetwarzania można zrealizować przez podział zadań i jednoczesne ich rozwiązywanie w poszczególnych procesorach. Problem ten nabiera większej wagi w chwili zbliżania się technologii cyfrowej do nieprzekraczalnego progu szybkości sekwencyjnego przetwarzania informacji, związanego ze skończonym czasem rozchodzenia się sygnałów.

Ze względu na sposób organizacji pracy systemów wieloprzetworzeniowych wyróżnić można systemy hierarchiczne i systemy równoległe.

Hierarchiczność polega na tym, że wśród połączonych JC, jedna z nich pełni funkcję zarządzającą czynnościami pozostałych,

realizując programy systemu operacyjnego, w chwilach wolnych zaś może przetwarzać zadania użytkowników.

Równoległość czyni każdą z JC równoprawną w kierowaniu pracą SL w kolejności przyjmowania funkcji sterowania określonych przez system operacyjny na podstawie analizy stanu SL.

W i e l o d o s t ę p n o ś ć SL jest cechą polegającą na tym, że wielu użytkowników może jednocześnie wykorzystywać sprzęt liczący, przy czym każdy z nich jest przekonany, że jest jedynym właścicielem w czasie użytkowania SL.

W systemach wielodostępnych każdy z użytkowników dysponuje własnym urządzeniem we/wy, zwanym terminalem /urządzeniem końcowym, abonenckim/ służącym do komunikacji z SL.

Systemy wielodostępne poza dużą przepustowością cechuje możliwość pracy konwersacyjnej i przetwarzania uwarunkowanego czasowo /na bieżąco/. Eksploatacja wielodostępnych systemów jest bardziej elastyczna, niezawodna i ekonomiczniejsza niż kilku SL wykorzystywanych przez oddzielnych użytkowników o łącznej mocy obliczeniowej równej mocy systemu wielodostępnego.

Podstawą realizacji systemów krotnych jest:

- podział pamięci,
- podział czasu,
- podział zadań,
- system przerwań,

które omówione zostaną w dalszej części opracowania.

3.4. Organizacja systemów liczących

Szybkość i jakość działania SL zależy w równej mierze od jego rozwiązań technologicznych /sprzętowych elementów i układów/ oraz od organizacji wymiany /przesyłania/ i przetwarzania informacji w układach, jak i między tymi układami. Odpowiednia organizacja i wykorzystanie elementów SL pozwala na określenie dróg przesyłania informacji i efektywną pracę SL.

Wąskim gardłem pracy SL jest często długi czas dostępu i przesyłania informacji między urządzeniami zewnętrznymi a JC oraz między PAO a JAL. Przyczyną tego bywa niedopasowanie ich struktury do struktury samego procesu przetwarzania informacji. Eliminowania tego zjawiska można dokonywać na drodze organizacji /podziału/ i adresowania pamięci, organizacji wymiany informacji, organizacji sterowania /podział czasu, system przerwań/ itp.

Organizację pamięci ze względu na różnorodną pojemność i czas dostępu tworzy się w sposób hierarchiczny. Na najwyższym poziomie hierarchii stoją pamięci najszybsze o małej pojemności, zaś na najniższym - pamięci o dużych pojemnościach i długim czasie dostępu. Pamięci poziomu pierwszego uczestniczą bezpośrednio w pracy JC i nazywane są pamięciami operacyjnymi, pamięci poziomu drugiego uczestniczą w pracy pośrednio i nazywane są pamięciami pomocniczymi /zewnętrznymi/. Każda z pamięci zbudowana jest na odpowiedniej technologii i posiada określoną strukturę organizacyjną.

Najprostszą strukturę organizacyjną ma pamięć adresowa. Każda komórka pamięci ma przypisany stały numer /adres/. Określenie adresu komórki można zrealizować przez adresację bezpośrednią lub pośrednią.

Adresowanie bezpośrednie /obecnie nie stosowane/ polega na umieszczeniu w części adresowej słowa rozkazowego całego adresu komórki pamięci. Pociąga to za sobą dysponowanie długimi słowami rozkazowymi, co powoduje znaczne straty gospodarki pamięcią, gdyż nie wszystkie rozkazy wykorzystują efektywnie całe słowo.

Adresowanie pośrednie pozwala na zmianę adresu, zawartego w części adresowej rozkazu przed jego wykonaniem, przez stosowanie rejestrów /modyfikatorów/. Sposoby adresowania pośredniego polegają na tym, że:

1. W części adresowej zapisany jest adres rejestru, którego zawartość określa adres fizyczny PAO.
2. W słowie rozkazowym jest część adresu, zaś pozostała część w rejestrze, gdzie adres PAO tworzony jest przez zestawienie tych części.
3. W części słowa rozkazowego jest adres względny /logiczny/, który po dodaniu zawartości odpowiedniego rejestru modyfikacji określa bezwzględny /fizyczny/ adres PAO.

Ograniczona pojemność PAO powoduje konieczność umieszczania informacji w pamięci zewnętrznej i sprowadzanie jej w miarę potrzeby do PAO. Wymaga to dokonywania podziału informacji tak, aby nie przekroczyć przydzielonego obszaru PAO. Technika takiego podziału wiąże się z koncepcją pamięci wirtualnej. Idea pamięci wirtualnej polega na tym, że cała pamięć jest ponumerowana na odpowiednie bloki zwane stronami. Pojemności stron mogą być stałe lub zmienne. Komórki stron są adresowane liczbami naturalnymi. Adres komórki pamięci jest określony numerem strony i numerem komórki na danej stronie i nosi nazwę adresu wirtualnego.

Przechodzenie z adresu wirtualnego na adres fizyczny może być sekwencyjne w pamięciach adresowalnych, bądź równoległe w pamięciach asocjacyjnych /skojarzeniowych/ zwanych pamięciami adresowa-

nymi zawartością. Zastosowanie pamięci asocjacyjnej przyspiesza proces pracy SL.

Podstawowe korzyści stosowania pamięci wirtualnych to: możliwość dynamicznego podziału pamięci, ograniczenie strat /luk/ pamięci oraz łatwiejsza ochrona informacji w pamięci.

Pamięci współczesnych systemów liczących mają strukturę blokową, co zwiększa efektywność jednoczesnego korzystania z pamięci w trakcie realizowania różnych czynności w JC. Wymaga to jednak stosowania odpowiednich metod organizacji pracy, podziału i ochrony pamięci.

Z hierarchicznej struktury pamięci wynika, że pewne typy pamięci są szybsze, a inne wolniejsze. Wolniejsze pamięci wpływają na hamowanie pamięci szybkich. Wiemy już, że szybkość pracy JC uwarunkowana jest czasem cyklu pracy pamięci, czyli pamięć wpływa hamująco na szybkość JAL. Dla łagodzenia powyższych czynników stosuje się techniki nakładania i "przeplatania" pamięci oraz ustala się priorytety dla programów i urządzeń.

Wyróżnić można dwa typy ustalania priorytetów:

1. Ustalanie programowe. W metodzie tej przerwania elementarne zapisywane są do rejestru przerwań i następnie dzielone na klasy. Każdej klasie przyporządkowany jest bit rejestru klas i bit rejestru maski. Programowo dostępny jest rejestr maski określający dopuszczalność klasy przerwań. Priorytety następnych przerwań ustalane są programem nadzorczym i mogą być modyfikowane rejestrem maski.

2. Ustalanie układowe. Podobnie jak poprzednio przerwania dzieli się na klasy. Każdej klasie przypisany jest na stałe priorytet. Przerwania o niższym priorytecie czekają na zakończenie zadań o wyższym priorytecie.

Podstawą funkcjonowania systemów krotnych oprócz podziału pamięci jest podział czasu. Podział czasu polega na zastąpieniu czasu ciągłego JC, ciągiem odcinków czasowych, o dowolnych długościach, z których każdy może być przydzielony dla wykonania innego zadania. Początki odcinków czasowych wyznaczone są przez przerwania zegarowe. Zegar kontroluje wykonanie zadania porównując czas faktyczny z czasem przeznaczonym na jego wykonanie.

Techniki nakładania i przeplatania prowadzą do dzielonej organizacji pamięci, a przez to do zwielokrotnienia szybkości jednostki centralnej i całego SL. Wymaga to jednak odpowiedniej organizacji sterowania, czyli stosowania koordynatorów /przełączników/ pamięci. Uzyskuje się to przez kanałową organizację wejścia/wyjścia oraz system przerwań.

Kanałowa technika umożliwia równoczesną pracę UZ i JC. Taki typ pracy zapewniają autonomiczne jednostki sterowania oraz zastosowana technika przerwań programowych.

Przerwanie polega na wytworzeniu, przy zaistnieniu zdarzenia, odpowiedniego sygnału umożliwiającego rozpoznanie przyczyny zjawiska, a następnie jego realizację. Polega ono na zawieszeniu przez JC aktualnie wykonywanego programu i przejście do realizacji programu obsługi przerwania. Po wykonaniu podprogramu obsługi przerwania SL wraca do realizacji przerwanego programu od momentu, w którym nastąpiło przerwanie.

Określenie możliwości obsługi przerwania związane jest z priorytetami przerwań. W SL realizowanych jest wiele programów oraz pracuje równolegle wiele urządzeń, którym konieczne jest przyznanie pewnej wagi. Realizuje się to przez rejestr zegara, w którym zapisany jest czas realizacji zadania. Po upływie czasu następuje przerwanie i przejście do nowego zadania.

4. Oprogramowanie podstawowe systemów liczących

Do wykonywania zadań w SL, oprócz sprzętu, konieczny jest program działania, który byłby realizowany automatycznie przez sprzęt. Program ten musi przewidywać bardzo drobiazgowo olbrzymią liczbę sytuacji, występujących w czasie przetwarzania informacji i przyjęto nazywać go oprogramowaniem podstawowym.

> Oprogramowaniem podstawowym określamy zbiór programów zapewniających funkcjonowanie systemu liczącego.

W oprogramowaniu podstawowym SL wyróżnia się następujące programy:

- system operacyjny - zbiór programów zarządzania i obsługi,
- translatory języków programowania,
- środki uruchamiania programów,
- programy redagowania,
- standardowe programy usługowe,
- standardowe pakiety programów użytkowych,
- programy diagnostyczne.

System operacyjny /SO/ stanowiący nieodłączną część sprzętu obliczeniowego ma na celu zapewnienie właściwego funkcjonowania SL. Powinien on być dostosowany do różnych warunków i sposobów pracy oraz zastosowań. Uzyskuje się to przez modułową strukturę programów zarządzających i obsługujących, które wchodzą w skład SO.

Funkcjami programów zarządzających SO są:

- przyjmowanie i wydawanie zadań,
- przydzielanie i zwalnianie urządzeń,

- podział i ochrona pamięci oraz wymiana informacji między pamięciami,
- podział czasu JC między użytkowników,
- reagowanie na zmiany sprzętowe,
- komunikacja z operatorem o stanie SL,
- udostępnianie zbiorów systemowych,

zaś programów obsługujących:

- sterowanie przebiegiem operacji we/wy,
- sterowanie transmisją informacji między urządzeniami,
- interpretacja i generowanie informacji.

Przechodzenie od programu użytkownika do SO odbywa się na skutek przerwań. Po zakończeniu obsługi przerwania następuje przejście do programu użytkowego zgodnie z algorytmem podziału czasu między użytkowników.

T r a n s l a t o r y j ę z y k ó w programowania są programami tłumaczącymi napisane w językach źródłowych programy użytkowe na programy wynikowe zapisane w języku wewnętrznym SL. Ze względu na rodzaj języków programowania wyróżnia się następujące typy translatorów:

1/ assemblery - programy tłumaczące dla języków symbolicznych,

2/ kompilatory - programy dla języków algorytmicznych,

3/ interpretatory - programy dla języków konwersacyjnych.

Przy ocenie translatorów należy uwzględniać takie parametry jak:

- konfiguracja SL,
- czas tłumaczenia programu,
- poprawność programu.

S t a n d a r d o w e p r o g r a m y usługowe i pakiety programów użytkowych służą do wykonywania często występujących w programach użytkownika czynności /np. sortowanie/ oraz jako podprogramy programów użytkowych /np. metoda transportowa w programowaniu liniowym/.

P r o g r a m y d i a g n o s t y c z n e stanowią grupę środków umożliwiających sprawdzenie poprawnej pracy SL przez wykrywanie błędów i lokalizację uszkodzeń systemu.

Znajomość podstawowych zasad funkcjonowania oraz architektury SL winno przyczynić się do zwiększenia efektywności stosowania metod i środków informatyki w doskonaleniu systemów dowodzenia.

5. A r c h i t e k t u r a r o d z i n y s y s t e m ó w l i c z ą c y c h "O D R A - 1 3 0 0"

Systemy liczące "ODRA-1300" opracowane zostały przez Wrocławskie Zakłady Elektroniczne ELWRO przy współpracy z angielską firmą ICL.

Architektura tych systemów liczących, to jest ich struktura funkcjonalna i logiczna oraz organizacja i współdziałanie środków technicznych i programowych są wspólne. Jednostki centralne, urządzenia zewnętrzne i oprogramowanie są kompatabilne, co umożliwiło współpracę firmy ICL i ELWRO w zakresie rozwijania technologii i oprogramowania komputerów tej rodziny.

W skład rodziny "ODRA-1300" wchodzą trzy komputery o zróżnicowanej mocy obliczeniowej i wydajności, przeznaczone do obliczeń numerycznych, przetwarzania danych i sterowania procesami technologicznymi.

Rozważania nad architekturą SL ODRA-1300 skupione zostaną na tych elementach, które są wspólne dla rodziny SL, a mianowicie na strukturze funkcjonalnej, organizacji i oprogramowaniu, bez wnikania w szczegóły technicznych rozwiązań.

5.1. Struktura systemów liczących ODRA-1300

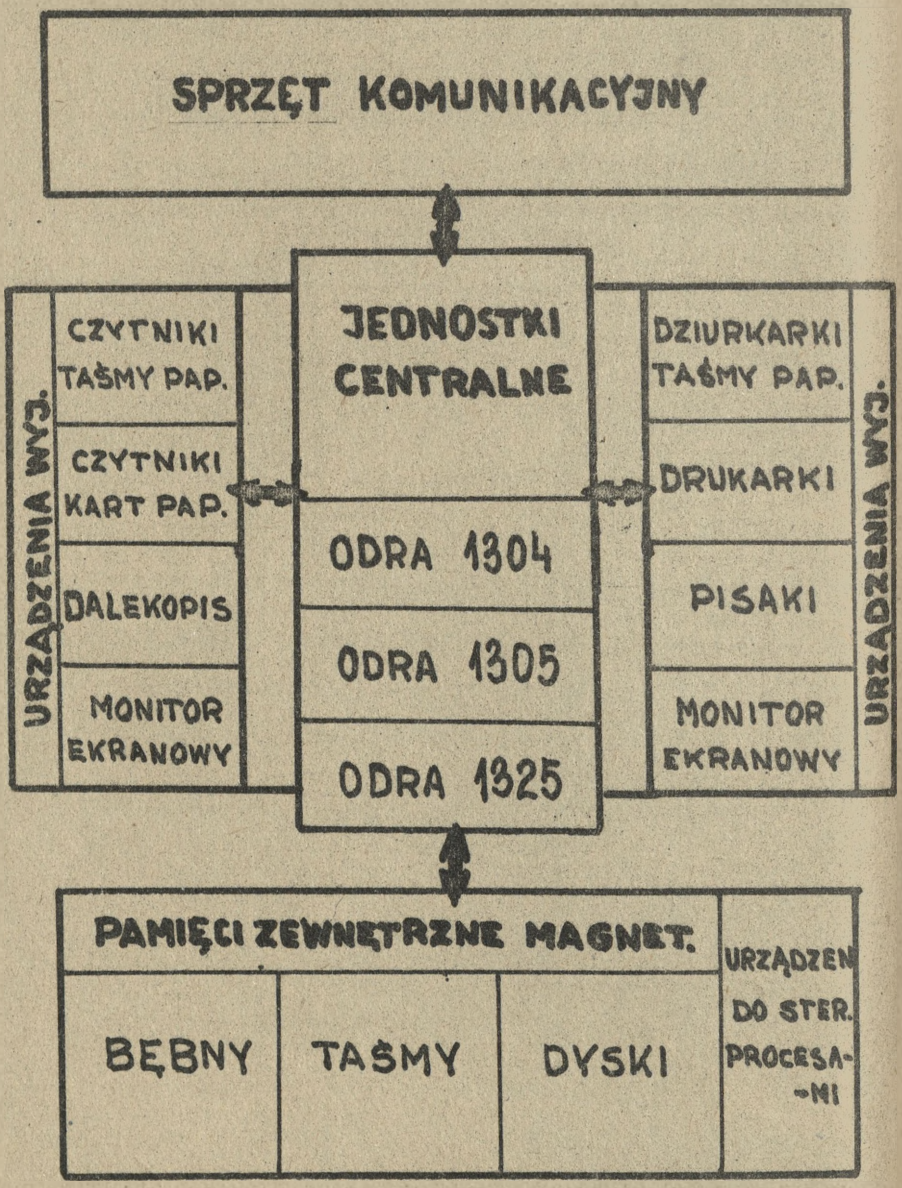
System liczący ODRA-1300 posiada budowę modułową /rys. 4/, co zapewnia tworzenie odpowiednich konfiguracji /zestawów/ urządzeń w zależności od potrzeb i wymagań. Z takiego rozwiązania wynika możliwość wymiany urządzeń zewnętrznych między systemami rodziny komputerów oraz elastyczność w doborze odpowiedniego dla użytkownika zestawu urządzeń.

W skład rodziny ODRA-1300 wchodzą trzy komputery:

- komputer ODRA-1304, oparty na technologii półprzewodnikowej i organizacji prac z podziałem czasu, przeznaczony do obliczeń naukowo-technicznych i przetwarzania danych w stosunkowo niewielkich systemach informacyjnych;
- komputer ODRA-1305, zbudowany na układach średniej skali integracji /scalonych/ i asynchronicznej organizacji pracy urządzeń zewnętrznych i pamięci operacyjnej, wykorzystywany w głównej mierze do przetwarzania danych w systemach informacyjnych;
- komputer ODRA-1325, zbudowany na technice układów scalonych i odpowiedniej organizacji logicznej pozwalającej wykorzystanie maszyny dla potrzeb sterowania procesami technologicznymi.

W systemie ODRA-1300 wyróżnić można następujące zespoły funkcjonalne:

a/ jednostkę centralną składającą się z pamięci operacyjnej, centralnej jednostki przetwarzania i kanałów;



Rys. 4. System liczący ODRA-1300

b/ urządzenia zewnętrzne, w skład których wchodzą jednostki sterujące i mechanizmy urządzeń we/wy.

Dzięki zastosowaniu standardowego układu /złącza/ sprzęgającego możliwe jest wyposażanie wszystkich komputerów rodziny w tego samego typu urządzenia zewnętrzne o dowolnej ściśle określonej z góry ich liczności.

Centralna jednostka przetwarzania /CJP/ steruje wykonaniem rozkazów programu przechowywanego w pamięci operacyjnej, rejestruje i sygnalizuje sekwencję wykonywanych operacji, sygnalizuje pojawienie się błędów, zapoczątkowuje działanie urządzeń we/wy oraz steruje przebiegiem wprowadzania i wyprowadzania informacji.

CJP wyposażona jest w specjalne układy elektroniczne takie jak:

1. Pamięć stała /mikroprogramów/ o pojemności 512 słów 44-bitowych, zawierająca logikę wszystkich operacji wykonywanych przez komputer w postaci ciągów mikrorozkazów tzw. mikroprogramów. Realizacja operacji polega na tym, że każdy rozkaz jest realizowany przy pomocy mikrorozkazów pamiętanych w pamięci stałej. Zawartość pamięci ukierunkowana jest jedynie na odczyt i jest niedostępna dla programisty.

2. Centralna jednostka arytmetyczno-logiczna składająca się z sumatora równoległego, rejestrów 24-bitowych zwanych akumulatorami oraz 1-bitowych rejestrów stanu realizacji programu /program normalny, priorytetowy, sterujący/, warunku i nadmiaru arytmometru, realizująca wszystkie operacje arytmetyczne i logiczne, zgodnie ze sterowaniem przekazanym przez mikrorozkazy zawarte w pamięci stałej. Arytmetyka realizowana jest w systemie dwójkowym /binarnym/.

3. Sterowanie, w skład którego wchodzi rejestr licznika rozkazów, rejestr rozkazów, układy dekodujące oraz układy wykonania i realizacji sekwencji sterującej. Zadaniem sterowania jest odczytywanie kolejnych rozkazów programu zapisanego w pamięci operacyjnej i wytworzenie na podstawie treści rozkazu sygnałów sterujących powodujących wykonanie odpowiednich czynności obliczeniowych i organizacyjnych w komputerze. Wykonanie dowolnego rozkazu w maszynie odbywa się w czterech etapach:

- odczytanie rozkazu z pamięci operacyjnej według stanu licznika rozkazów,
- określenie adresu skutecznego /bezwzględnego/ argumentu - modyfikacja adresu,
- wykonanie operacji rozkazu,
- ustalenie adresu następnego rozkazu.

P a m i ę ć o p e r a c y j n a /PAO/ służy do przechowywania danych /argumentów/ i rozkazów aktualnie przetwarzanych programów w CJP. Zbudowana jest na rdzeniach ferromagnetycznych i posiada strukturę modułową /blokową/ o pojemności 16 lub 32 Kseów /1 Kseowo = 1024 sł./ każdy. Słowo maszynowe zawiera 24 bity informacyjne, jeden bit kontrolny /parzystości/ i może przedstawiać liczbę, cztery znaki alfanumeryczne, wielkość logiczną lub rozkaz programu. Liczba bloków 32 Kseów może wynosić od 1 do 8 w zależności od typu komputera i stanowić pojemność PAO od 32÷256 Kseów. Dla zapewnienia współpracy komputera z poszczególnymi blokami zastosowany został koordynator pamięci.

Podział pamięci na bloki /moduły/ umożliwia zwiększenie pojemności oraz stosowanie metody przelotu adresów, polegający na umieszczaniu kolejnych adresów rozkazów programu lub programów /w pracy wieloprogramowej/ w różnych blokach, co zapewnia korzysta-

nie z pamięci przez różne urządzenia zapisu/odczytu/, a tym samym powoduje zmniejszenie użytkowego czasu cyklu pracy pamięci i skrócenie czasu przetwarzania.

Pamięć operacyjna może współpracować z dowolnym urządzeniem zewnętrznym i centralną jednostką przetwarzania.

K a n a ł y służą do zapewnienia współpracy urządzeń zewnętrznych z pamięcią operacyjną w celu wymiany informacji. Dla zapewnienia jednoczesnej pracy arytmometru i kanałów przesyłania informacji z PAO zastosowano układ sterujący zwany koordynatorem kanałów /KK/, który wstrzymuje kontakty arytmometru z pamięcią operacyjną na czas przesłania jednego słowa z PAO do kanału i odwrotnie.

W zależności od typu zastosowanego sterowania kanału systemy liczące ODRA-1300 mogą być wyposażone w następujące typy kanałów:

1. Kanały znakowe, przeznaczone dla organizacji przesyłania danych między PAO, a wolnymi urządzeniami zewnętrznymi typu urządzeń taśmy papierowej, kart papierowych, drukarek wierszowych, monitorów ekranowych. W kanałach tych znaki 6-bitowe przesyłane są szeregowo bez grupowania w bloki. Ilość kanałów znakowych jest różna i przeważnie wynosi od 12 do 18 kanałów.
2. Kanały buforowane służą do organizacji przesyłania danych pomiędzy PAO, a szybkimi urządzeniami zewnętrznymi typu pamięć dyskowa, bębnowa lub taśmowa w postaci 24-bitowych słów. Kanały te zawierają dodatkowo bufor 4-znakowych rejestrów 6-bitowych, co powoduje czterokrotne zmniejszenie czasu kontaktowania się z PAO. Wśród kanałów buforowanych mogą być kanały autonomiczne i nieautonomiczne. Kanały autonomiczne wyposażone są dodatkowo w rejestry adresowe i licznikowe, zatem aktualizacja słów sterujących nie odbywa się w PAO. Liczba kanałów autonomicznych wynosi standardowo 4 lub 8.

3. Kanały multipleksorowe przeznaczone do organizacji do 63 jednoczesnych przesyłań danych pomiędzy wolnymi urządzeniami zewnętrznymi /urządzenia abonenckie lub teleprzetwarzania/ a pamięcią operacyjną maszyny. Wyposażone są w specjalne rejestry, których zadaniem jest pamiętanie numeru obsługiwanego użytkownika /podkanału/. Współpraca z urządzeniami zewnętrznymi odbywa się w oparciu o zasadę podziału czasu.

4. Kanały priorytetowe /przemysłowe/, których przeznaczeniem jest organizacja współpracy maszyny z urządzeniami sterującymi procesem zachodzącym w czasie rzeczywistym. Kanał taki wyposażony jest w dodatkowe układy logiczne umożliwiające szybką obsługę przerwania i posiada najwyższy priorytet zgłoszeń.

Każdy z kanałów pracuje przez złącze standardowe /interface/. Proces przesyłania danych rozpoczyna się od zgłoszenia urządzenia we/wy w złączu standardowym, gdy urządzenie jest gotowe do natychmiastowej transmisji danych. Transmisja może odbywać się w trybie przesyłania normalnego /jednoblokowego/, przesyłania długiego /wieloblokowego/ i przesyłania specjalnego. Przy zgłoszeniu się kanału do koordynatora kanałów następuje jego zapis w rejestrze zgłoszeń, gdzie na podstawie tej informacji w układzie priorytetowym wypracowany jest sygnał, który służy do określenia adresu słowa sterującego w PAO i przygotowania urządzeń do rozpoczęcia transmisji pomiędzy PAO i odpowiednim kanałem.

U r z ą d z e n i a zewnętrzne podłączone do jednostki centralnej przez złącze standardowe przeznaczone są do wprowadzania danych i programów do PAO oraz wyprowadzania wyników i programów z PAO przy pomocy maszynowych nośników informacji. Informacja wprowadzana do jednostki centralnej jest dekodowana w urządzeniu

zewnętrznym do postaci 6-bitowych ciągów, zaś wyprowadzana z jednostki centralnej w postaci 6-bitowych ciągów jest kodowana w urządzeniu zewnętrznym przez niezależny układ sterowania zwany jednostką sterowania. Pozwala to na wprowadzanie i wyprowadzanie danych bez zajmowania czasu pracy jednostki centralnej, a tym samym na równoczesne przetwarzanie i transmisję danych.

Zestaw urządzeń zewnętrznych współpracujących z systemem liczącym ODRA-1300 jest następujący:

1. Czytnik taśmy papierowej przeznaczony do wprowadzania informacji z taśmy dziurkowanej 5,6,7,8-ścieżkowej z szybkością czytania 1000 znaków/sek. Przetwarzanie informacji zapisanej w kodzie 7-bitowym jest automatyczne /6-bitów informacyjnych + 1 bit parzystości/, w pozostałych kodach występuje dodatkowe tłumaczenie programowe.

2. Dziurkarka taśmy papierowej służy do wyprowadzania informacji na taśmę papierową 5,6,7,8-ścieżkową z szybkością 100 zn/s i gęstości perforacji 10 zn/cal. Wprowadzanie informacji z 6-bitowego kodu wewnętrznego na 7-ścieżkowy jest automatyczne, w pozostałych przypadkach wymaga dodatkowych programów.

3. Czytnik kart papierowych, do wprowadzania informacji z kart perforowanych zawierających po 80 kolumn i 12 wierszy ponumerowanych od 10,11,0,1, ..., 9, z szybkością czytania 1000 kart/min. Czytanie informacji zapisanej w 64-znakowym kodzie ICT odbywa się kolumnami i jest automatycznie przetwarzane na 6-bitowy kod maszynowy.

4. Drukarka wierszowa służy do wyprowadzania informacji alfanumerycznych z szybkością 1300 linii/min. o długości linii równej 120 znaków i gęstości 6,8 linii/cal oraz 10 zn/cal. Drukarka

posiada 64 różne znaki automatycznie kodowane według zmodyfikowanego kodu ICT.

5. Pamięć taśmowa przeznaczona jest do przechowywania dużych zbiorów informacji udostępnianych jednostce centralnej z szybkością około 128 000 zn/sek. Posiada budowę modułową, co pozwala na podłączenie do 12 taśm do jednej jednostki sterującej /adaptera/. Informacja na taśmie magnetycznej zapisana jest na 9 ścieżkach /8 ścieżek informacyjnych + 1 ścieżka nieparzystości/. Pojemność jednej taśmy /szpuli/ o długości około 730 m wynosi 10 milionów znaków, czas przewijania jednej szpuli trwa 2,5 min. Pamięć taśmowa jest pamięcią o sekwencyjnym dostępie do informacji.

6. Pamięć bębnowa podobnie jak pamięć taśmowa ma strukturę modułową i pozwala na podłączenie do 4 bębnów do jednostki sterującej. Pojemność bębna wynosi 64 Ksłów 24-bitowych umieszczonych na 128 ścieżkach po 512 słów. Szybkość przesyłania informacji 12 000 słów/sek., gęstość zapisu 16 bitów/mm.

7. Pamięć dyskowa na dyskach wymiennych o 10 powierzchniach dyskowych tworzących pakiet ma również strukturę modułową, która pozwala na dołączenie do 8 pakietów do jednostki sterującej pamięcią. Na powierzchniach dysków znajduje się po 203 ścieżki /3 ścieżki organizacyjne/, z których każda podzielona jest na 8 sektorów o pojemności 128 słów 24 bitowych każdy. Pojemność pakietu wynosi 2000 Ksłów, a prędkość przesyłania 208 Kzn/sek. Pamięć dyskowa podobnie jak pamięć bębnowa jest pamięcią adresowalną /ścieżka, głowica, sektor/.

8. Multipleksor spełniający funkcję systemu komunikacyjnego, który umożliwia poprawną, zdalną i jednoczesną łączność z centralną jednostką przetwarzania do 63 wolnych urządzeń zewnętrznych i urządzeń transmisji danych.

PODSTAWOWE DANE TECHNICZNE SL "ODRA-1300"

PARAMETRY	MIANA	ODRA-1304	ODRA-1305	ODRA-1325
1	2	3	4	5
I. JEDNOSTKA CENTRALNA				
1. Arytmetyka	-	dwójkowa uzup.	dwójkowa uzup.	dwójkowa uzup.
2. Długość słowa	bit			
długie		48	48	48
krótkie		24	24	24
3. Reprezentacja liczb				
stałoprzec.krótka		24	24	24
-"- długa		48	48	48
zmiennoprzec.				
mantysa		38	38	38
cecha		9	9	9
4. Długość znaku	bit	6	6	6
5. Czasy podstawowych operacji	s			
pobranie liczby		20	1,2	2,2
dodanie stprzec.		26	1,6	2,6
mnożenie -"-		96	9	12
dzielenie -"-		200	14	18
dodawanie zmprzec.		250	10	9
mnożenie -"-		770	22	16
dzielenie -"-		880	34	25

1	2	3	4	5
6. Szybkość działania	oper/s			
obliczeń numerycz.		15 000	280 000	270 000
przetwarzania danych		45 000	400 000	300 000
7. Wieloprogramowość	ilość	4	16	2÷8
8. Wieloprocesorowość	ilość	0	2	0
9. Pojemność PAO	Ksłowa	32	32,64,96,128	8,16,32;;128
10. Cykl PAO	s	6	256 1	1
11. Czas dostępu	s	3	0,4	0,4
II. KANAŁY WE/WY				
1. Kanały	ilość			
znakowe selektorowe		10	18	12
buforowane autonomicz,		1	8	2
-"- nieautonomicz,		2	0	2
przemysłowe		0	1	2
multipleksorowe		1	2	1
2. Szybkość przesyłania	zn/s			
kanał bufor.autonom.		450 000	500 000	-
-"- nieautonom.		140 000	-	500 000
III. URZĄDZENIA ZEWNĘTRZNE				
1. Monitor	zn/s	10	10	10
2. Czytnik taśmy	-"-	1000	1000	1000

1	2	3	4	5
3. Czytnik kart	kart/min	1000	1000	1000
4. Perforator taśmy	zn/s	100	100	100
5. Drukarka wierszowa	lin/min	1300	13000	1300
6. Jednostki taśmy magnet.	ilość	12	6	6
7. Szybkość zap/odcz. PT	Kzn/s	43	128	128
8. Jednostki bębnowe	ilość	4	4	4
9. Szybkość zap/odcz. PB	Kzn/s	51	130	130
10. Pojemność znaków PB	mln	1 x 4	2,6 x 4	2,6 x 4
11. Pojemność zn. PD	mln	8 x 6	8 x 6	

9. Inne urządzenia takie jak, monitory ekranowe, pisaki X-Y, czytniki dokumentów, przetworniki do sterowania procesami technologicznymi, które spełniają standardowe zasady współpracy z kanałami systemu liczącego ODRA-1300 lub ICL-1900.

5.2. Organizacja systemu liczącego ODRA-1300

System liczący ODRA-1300 zbudowany jest z wielu układów elektronicznych /bloków funkcjonalnych/ i urządzeń zewnętrznych spełniających różne funkcje przetwarzania danych. Dla zapewnienia dużej efektywności /mocy obliczeniowej, wydajności i niezawodności/ systemu liczącego, nieodzownym jest opracowanie jego organizacji logicznej, a między innymi określenie formatów informacji, listy rozkazów, systemu podziału czasu, podziału pamięci, systemu przerwań, ochrony pamięci itp.

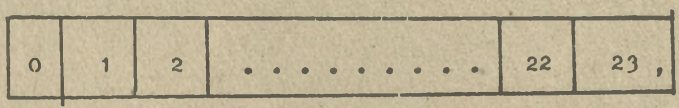
Podstawową organizacją maszyny ODRA-1300 jest organizacja słowowa. Słowo maszynowe jest stałej długości, składa się z 24 bitów informacyjnych i 1 bitu technicznego /parzystości/ i określa wielkość liczbową, modyfikator lub rozkaz.

Słowo jako wielkość liczbowa może przedstawiać:

a/ liczbę stałoprzecinkową.

W zależności od położenia przecinka liczby stałoprzecinkowe dzielą się na:

- liczby całkowite, gdy przecinek umieszczony jest po ostatnim bicie słowa



bit znaku bity znaczące przecinek

i zakresie z przedziału $/-2^{23}; 2^{23} - 1/$;

- liczby ułamkowe, gdy przecinek umieszczony jest po bicie znaku /ułamek właściwy/ lub po dowolnej pozycji słowa

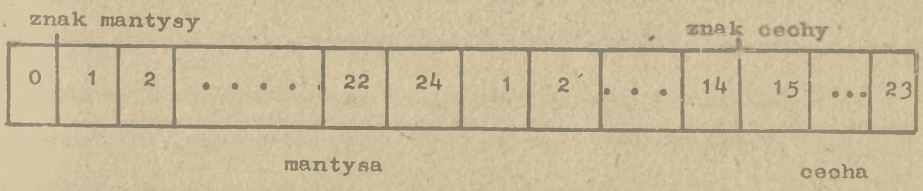
/ułamek niewłaściwy/ i zakresie ułamka właściwego z przedziału /-1; 1 - 2^{-23}/.

b/ liczbę stałoprzecinkową długą.

Liczba stałoprzecinkowa długa jest podwójnej długości i tworzy słowo długie 47-bitowe /znak drugiego słowa nie jest wykorzystywany/ o zakresie z przedziału /-2^{46}; 2^{46} - 1/ dla liczb całkowitych i /-1; 1 - 2^{-46}/ dla liczb ułamkowych.

c/ liczbę zmiennoprzecinkową.

W celu zwiększenia zakresu liczb wprowadzono liczby zmiennoprzecinkowe przedstawione w postaci mantysy i cechy /wykładnika/

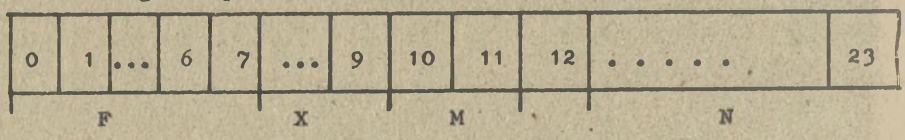


o zakresie /-1; -1/2/ lub /1; 1/2/ dla mantysy ora /-256; 256/ dla cechy.

W pewnych przypadkach słowo może być traktowane jako modyfikator i używane do modyfikacji rozkazu przed jego wykonaniem. Pierwotny adres pobranego rozkazu ma 12-bitów i można zaadresować tylko 4096 komórek pamięci operacyjnej. W celu rozszerzenia adresu do 18 bitów umożliwiających adresowanie do 256 Kkomórek, stosuje się modyfikatory czyli trzy pierwsze rejestry z ośmiu akumulatorów stosowanych w systemie ODRA-1300, w których pamiętane są pełne 18-bitowe adresy. Modyfikacja w tym przypadku polega na rozszerzeniu adresu pobranego do wykonania rozkazu przez dodanie do 12-bitowego adresu tego rozkazu zawartości jednego z trzech modyfikatorów.

Systemy liczące ODRA-1300 wykonują działania arytmetyczne i logiczne na argumentach 24 lub 48-bitowych oraz na znakach

6-bitowych. Słowo maszynowe traktowane jako rozkaz ma długość 24 bitów i zajmuje jedną komórkę pamięci operacyjnej. Format słowa rozkazowego ma postać



gdzie:

F - typ operacji /kod operacji/ z listy 128 oper.
X - adres jednego z ośmiu akumulatorów
M - adres jednego z trzech modyfikatorów
N - adres pierwotny komórki pamięci operacyjnej.

W rodzinie maszyn systemu liczącego ODRA-1300 istnieje kilka typów rozkazów, które różnią się między sobą strukturą i sposobem działania. Najogólniej wszystkie rozkazy dzielą się na dwie grupy:

a/ rozkazy normalne składające się z rozkazów arytmetycznych i przesuwania;

b/ rozkazy skokowe /sterujące/, w skład których wchodzą rozkazy skokowe zwykłe i rozkazy skokowe relatywne.

Zbiór wszystkich rozkazów dla danego systemu tworzy listę rozkazów. Liczność rozkazów na liście rozkazów w rodzinie maszyn ODRA-1300 określa kod 7-bitowy części operacyjnej i nie przekracza 128 rozkazów.

Jednolita lista rozkazów i jednakowe formaty informacji dla systemu ODRA-1300 zapewniają zgodność programową. W maszynach tego systemu rozkazy realizowane są technicznie przez układy elektroniczne lub mikroprogramy oraz programowo przez specjalne procedury przechowywane w systemie operacyjnym.

Praca w systemie liczącym realizowana jest w trybie wieloprogramowym lub na zasadzie podziału czasu centralnej jednostki

przetwarzania. Oznacza to, że przez cały czas pracy maszyny przechowywanych jest w pamięci operacyjnej do 16 programów, które są pseudorównocześnie bądź równocześnie wykonywane w zależności czy konfiguracja maszyny jest jedno czy wieloprocesorowa. Przekazywanie jednostki centralnej z jednego programu na drugi odbywa się według pewnych reguł powodujących podział czasu jednostki centralnej.

W celu organizowania podziału czasu pracy jednostki centralnej pomiędzy programami użytkowymi oraz dla jednoczesnej pracy urządzeń zewnętrznych w pamięci operacyjnej systemu liczącego stale rezyduje program sterujący zwany EXECUTIVE. Program ten stanowi integralną część sprzętu systemu i zajmuje pojemność pamięci operacyjnej równą 8000 komórek.

Programy użytkowe zajmują dowolne pole pamięci i są przechowywane w dowolnej kolejności, z tym że łączna długość wszystkich programów nie może przekroczyć pojemności zainstalowanej pamięci operacyjnej. Przechowywane w pamięci programy użytkowe są adresowane relatywnie /względnie/, a w czasie realizacji programu adresy względne są automatycznie zamieniane na bezwzględne. Pozwala to na przenoszenie w zależności od potrzeb programów z jednego do drugiego pola pamięci zapewniając racjonalną gospodarkę pamięcią.

Każdy program użytkowy zajmuje dla siebie określony obszar pamięci i kontakt w czasie realizacji rozkazów programu może odbywać się jedynie pomiędzy komórkami tego obszaru bez wchodzenia do obszarów pozostałych programów, co powodowałoby niszczenie informacji innych programów. Dlatego też zachodzi konieczność zabezpieczenia/ochrony/pamięci.

Ochrona pamięci oraz zamiana adresów względnych na bezwzględne realizowana jest przez układy elektroniczne /rejestr dolny i górny/,

w których przechowywane są graniczne bezwzględne adresy aktualnie realizowanego programu. Jeżeli w programie wystąpi adres, który nie zawiera się w przedziale adresów granicznych następuje przerwanie realizacji programu i przejście do następnego programu.

Przydziałem czasu pracy jednostki centralnej dla poszczególnych programów zajmuje się system przerwań. W systemie ODRA-1300 wyróżnia się trzy podstawowe reżimy pracy jednostki centralnej:

- praca programu użytkowego normalnego;
- praca programu sterującego EXECUTIVE;
- praca programu priorytetowego.

Przerwania programowe odbywają się według ustalonego systemu priorytetowego, w którym przyjęto, że program normalny ma mniejszy priorytet, zaś priorytetowy najwyższy. Wszystkie przerwania można podzielić na:

a/ przerwania wewnętrzne /dobrowolne/, zainicjowane pojawieniem się w programie użytkowym rozkazu nie mającego realizacji układowej, a np. programową /ekstrakody/,

b/ przerwania zewnętrzne /wymuszone/, zainicjowane przez system priorytetowy.

5.3. Oprogramowanie systemu liczącego ODRA-1300

Wraz z rozwojem techniki i organizacji systemów liczących ich możliwości zastosowania i wykorzystania w działalności ludzkiej są ogromne. Z drugiej strony, ich duża moc obliczeniowa, wydajność i niezawodność oraz różnorakie zastosowanie spowodowały, że obsługa ich stawała się trudna, zaś przygotowanie programów bardzo złożone. Z tych też względów oprócz nowych rozwiązań technologicznych zaczęto rozwijać oprogramowanie systemów liczących.

W skład oprogramowania systemu ODRA-1300 nierozerwalnie związanego z techniką komputerową wchodzi oprogramowanie podstawowe i specjalistyczne.

Oprogramowanie podstawowe można podzielić na:

1. Translatory języków programowania, przeznaczone są do tłumaczenia programów zapisanych w wyższego rzędu/niż język wewnętrzny maszyny/językach programowania. Tłumaczenie realizowane jest przez kompilację, konsolidację /łączenie podprogramów/ i wprowadzanie. Językami programowania są języki: PLAN - język symboliczny, FORTRAN i ALGOL - języki algorytmiczne dla celów numerycznych, COBOL - język do zastosowań gospodarczych, SIMON i CSL - języki symulacyjne, JEAN - język konwersacyjny.
2. Programy techniczne składające się z testów i zadań kontrolnych, służące do badania poprawności pracy systemu liczącego.
3. Programy organizacyjne zapewniające wstępne przygotowanie i przetwarzanie danych.
4. Systemy operacyjne rozumiane jako programy sterujące i zarządzające pracą systemu liczącego w trybie wieloprogramowania i wielodostępu, realizacji ekstrakodów, komunikacji z operatorem i realizacji wielu funkcji operatorskich, automatycznej kontroli poprawności działania systemu itp. Podstawowymi systemami operacyjnymi są systemy EXECUTIVE i GEORGE.
5. Programy biblioteczne realizujące standardowe funkcje i procedury.

Oprogramowanie specjalistyczne umożliwia stosowanie jednostek centralnych do wyspecjalizowanych zadań i optymalne ich realizowanie dzięki istniejącym, gotowym procedurom i składa się z programów matematyczno-technicznych, programów przetwarzania i zarządzania danymi i zbiorami danych.

6. Architektura komputerów Jednolitego Systemu "RIAD"

W wyniku prowadzonych w krajach RWPG intensywnych prac powstała rodzina komputerów charakteryzujących się jednolitą architekturą. Rodzina maszyn składa się z modeli maszyn typu R-10, R-20, R-21, R-22, R-30, R-32, R-40, R-50, R-60. Wszystkie modele z wyjątkiem R-10 i R-21 charakteryzują się jednolitością struktury, organizacji i programowania.

Opisując architekturę maszyn cyfrowych Jednolitego Systemu skupiona została uwaga na tych elementach, które są wspólne dla całej rodziny bez szczegółowego rozważania konkretnych rozwiązań.

6.1. Struktura Jednolitego Systemu "RIAD"

Podobnie jak SL rodziny ODRA, również SL Jednolitego Systemu /JS/ mają strukturę modułową /rys. 5/. To zapewnia elastyczność ich zastosowania i wykorzystania.

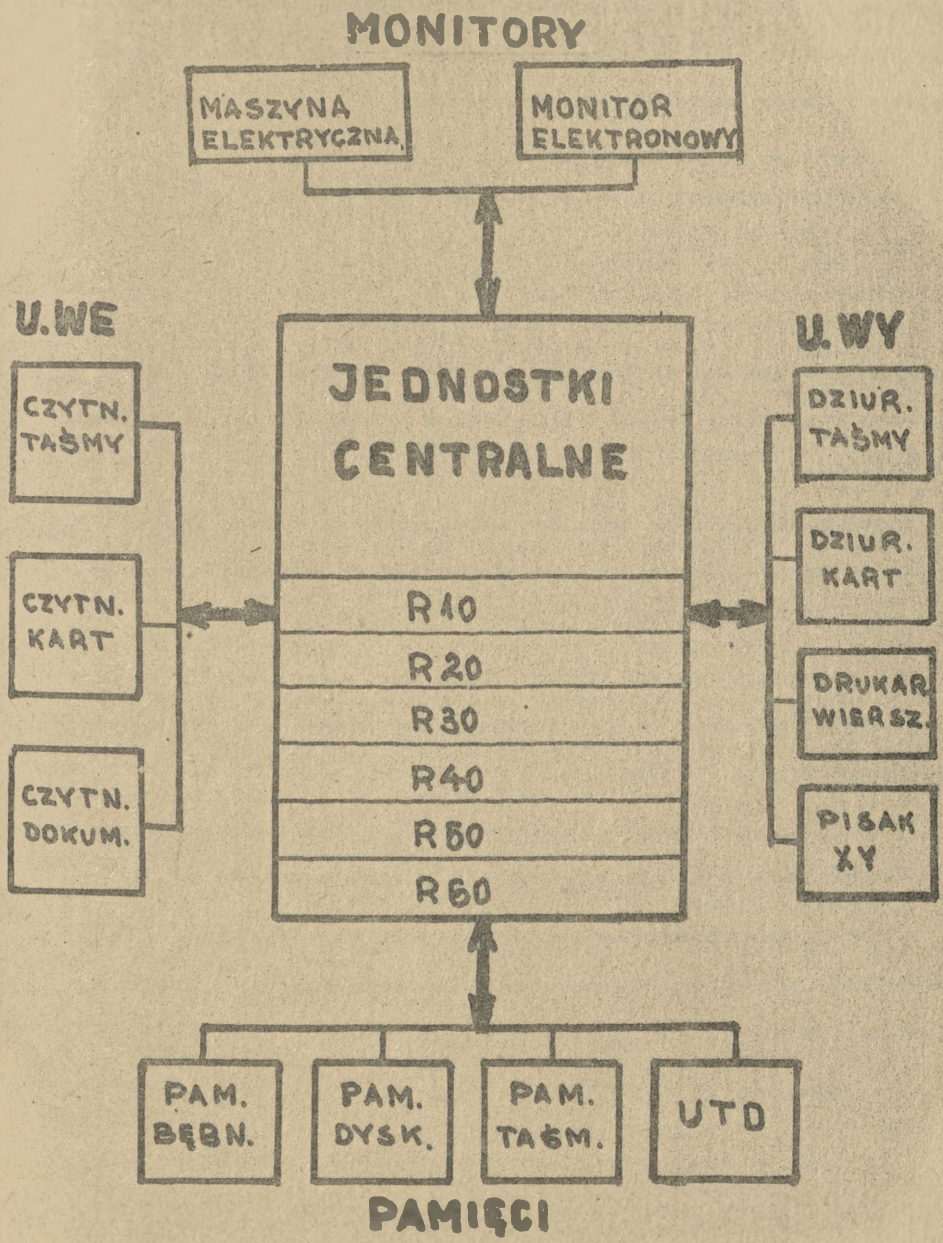
Modułami maszyn JS są następujące bloki funkcjonalne:

a/ procesor,
b/ pamięć operacyjna,
c/ kanały,
d/ urządzenia zewnętrzne.

Procesor wraz z pamięcią operacyjną tworzą jednostkę centralną JS RIAD.

Procesor maszyn JS zawiera rejestry i układy elektroniczne służące do adresowania PAO, realizacji operacji arytmetyczno-logicznych, inicjowania komunikacji PAO z UZ oraz kontroli poprawności działania systemu liczącego.

W skład procesora wchodzi JAL /sumator słowowy i bajtowy/, pamięć stała /16 rejestrów ogólnych, 4 rejestry zmiennoprzecinkowe



Rys. 5. Struktura JS "RIAD"

i rejestry robocze/ oraz układy sterowania. W procesorze mogą być przetwarzane liczby binarne, dziesiętne i dane alfanumeryczne. Wykonywanymi w procesorze operacjami są:

- arytmetyczne operacje stałoprzecinkowe,
- arytmetyczne operacje zmiennoprzecinkowe,
- logiczne operacje,
- dziesiętne operacje.

Formatami liczb w operacjach arytmetycznych są liczby stałoprzecinkowe krótkie /2 bajty/, stałoprzecinkowe długie /4 bajty/, zmiennoprzecinkowe krótkie /4 bajty/ i zmiennoprzecinkowe długie /8 bajtów/.

Operacje logiczne mogą być wykonywane na danych stałej długości /4 bajty/ oraz na polach danych alfanumerycznych zmiennej długości /do 256 bajtów/.

W operacjach dziesiętnych liczby mogą występować w formacie spakowanym /w jednym bajcie 2 cyfry dziesiętne/ lub rozpakowanym /w jednym bajcie 1 cyfra/.

Pamięć operacyjna jest pamięcią ferrytową o konstrukcji blokowej. Procesor i kanały posiadają niezależny dostęp do PAO.

Podstawową jednostką informacji jest bajt /8 bitów/. Większe jednostki informacji stanowią wielokrotność bajtu. Jednostkami tymi mogą być:

- półsłowo /2 bajty/;
- słowo /4 bajty/;
- podwójne /długie/ słowo /8 bajtów/.

Każdy bajt pamięci jest ponumerowany kolejnymi liczbami całkowitymi dodatnimi. Numery te są adresami pamięci. Adresem większej jednostki informacji /słowa, półsłowa, podwójnego słowa/ jest adres pierwszego z lewej strony bajtu tej jednostki.

Pojemność PAO jest zmienna i waha się w granicach 64÷1024 Kbajt.

Pracą komputera JS RIAD steruje program, którego rozkazy podobnie jak dane przechowywane są w pamięci operacyjnej. Każdy rozkaz zawiera kod operacji oraz adresy danych bądź dane biorące udział w operacji. Kod operacji zajmuje zawsze jeden bajt, co pozwala na zakodowanie 2^8 rozkazów wchodzących w skład listy rozkazów.

Wszystkie rozkazy JS RIAD można podzielić na pięć podstawowych typów, a mianowicie:

a/ rozkazy rejestr-rejestr /RR/ o długości 2-ch bajtów,
b/ rozkazy rejestr-pamięć /RX/ o długości 4-ch bajtów,
c/ rozkazy rejestr-pamięć bez indeksacji /RS/ - 4 bajty,
d/ rozkazy pamięć-argument bezpośredni /SI/ - 4 bajty,
e/ rozkazy pamięć-pamięć /SS/ o długości 6-u bajtów.

Podstawowym urządzeniem służącym do przesyłania informacji w systemie jest k a n a ł, który zapewnia równoległą pracę JC i urządzeń zewnętrznych.

System liczący jest wyposażony w dwa typy kanałów:

1. Kanały selektorowe w ilości 2÷6, umożliwiające szybkie przesyłanie danych z jednego z 256 urządzeń we/wy, które mogą być dołączone do jednego kanału.

2. Kanał multipleksorowy, współpracujący ze 128 urządzeniami we/wy poprzez urządzenie miltipleksorowe.

Kanały wyposażone we wspólną pamięć roboczą, niedostępną programiście, zawierającą rejestry sterujące operacjami we/wy.

U r z ą d z e n i a z e w n ę t r z n e w zależności od rodzaju wykonywanych czynności, można podzielić na:

1. Urządzenia wprowadzania danych z taśm i kart dziurkowanych, z dokumentów źródłowych i mikrofilmów,

2. Urządzenia wyprowadzania danych alfanumerycznych na maszynowe nośniki informacji, dokumenty wynikowe na drukarkach i monitorach oraz danych graficznych na pisakach i monitorach,

3. Pamięci zewnętrzne dyskowe, taśmowe i bębnowe,

4. Urządzenia transmisji danych.

Oprogramowanie podstawowe JS RIAD zostało podzielone na trzy podstawowe grupy:

1. Pakiety systemów sterujących dla pracy jednoprocesorowej, wieloprocesorowej i wielomaszynowej, systemów sterujących pracą uwarunkowaną czasowo i z podziałem czasu oraz systemów sterujących procesami zautomatyzowanymi.

2. Zestawy programów obsługi technicznej, w skład których wchodzą testy uruchomieniowe oraz testy kontrolne i diagnostyczne.

3. Systemy operacyjne typu OS, MOS i DOS, zapewniające optymalne możliwości wykorzystania zasobów komputera.

L I T E R A T U R A

1. Boris Beizer, Organizacja systemów komputerowych, PWN - 1979 r.

2. Jerzy Bromirski, Ryszard Pawęska, Podstawy informatyki cz. III skrypt, PW - 1976 r.

3. Donald Eadir, Nowoczesne maszyny i systemy cyfrowe, WNT 1975 r.

4. Zbigniew Gachowski, Projektowanie systemów informatycznych zarządzania, WNT 1974 r.

5. Janusz Hawryluk, Maszyna cyfrowa narzędzie człowieka współczesnego, WNT 1976 r.

6. Pod redakcją Z.Huzera, Organizacja maszyn cyfrowych, skrypt, PW 1973 r.

7. Romuald Jagielski, Komputery Jednolitego Systemu, PWE 1980 r.

8. Thenasis Kamburelis, Architektura logiczna maszyny cyfrowej ODRA 1305, MERA-ELWRO 1979 r.

9. Zbigniew Kieszkowski, Elementy informatyki. Technika metody zastosowania, PWN 1976 r.

10. Witold Komorowski, Ryszard Pawęska, Organizacja maszyn cyfrowych cz. I, skrypt, PW 1974 r.

11. Materiały szkoleniowe, ICT, wstęp do serii 1900, CODKK 1969 r.

12. Praca zbiorowa, Podstawy elektronicznej techniki obliczeniowej, skrypt PP 1978 r.

Rozdział trzeci

PODSTAWY ALGORYTMIZACJI I PROGRAMOWANIA MASZYN CYFROWYCH

1. Charakterystyka zadań operacyjno-taktycznych z punktu widzenia rozwiązywania ich za pomocą komputerów

Mając na względzie rozwiązywanie zadań operacyjno-taktycznych za pomocą komputerów, należy zdać sobie sprawę z właściwości tych zadań.

Na ogół zadania operacyjno-taktyczne charakteryzują się koniecznością wykonywania dużej ilości operacji matematyczno-logicznych. Tylko niewielka ilość zadań tego typu może być rozwiązywana wyłącznie za pomocą operacji matematycznych lub logicznych.

Przy rozwiązywaniu większości zadań operacyjno-taktycznych należy uwzględniać nie tylko wskaźniki ilościowe, ale także jakościowe /poziom wyszkolenia żołnierzy, jakość uzbrojenia, sprawność dowodzenia itp./. Wskaźniki takie uwzględnia się w obliczeniach przez stosowanie odpowiednich współczynników wagowych.

Zadania operacyjno-taktyczne wymagają często rozwiązań wielowariantowych, co wynika nie tylko z różnych wariantów działań własnych i przeciwnika, ale także z tego faktu, że informacje o przeciwniku są na ogół niepełne, nie zawsze wiarygodne, a nawet sprzeczne. Stąd też zazwyczaj zadań tych nie można rozwiązywać za pomocą metod analizy matematycznej, lecz należy stosować metody badań operacyjnych, a w szczególności teorię gier strategicznych.

Zadania operacyjno-taktyczne posiadają z reguły po kilka - spośród dopuszczalnych - rozwiązań, w związku z czym należy poszukiwać rozwiązań optymalnych lub przynajmniej suboptymalnych

ze względu na pewne, uprzednio określone kryteria efektywności.

Wśród zadań operacyjno-taktycznych można wyróżnić trzy podstawowe grupy:

- zadania obliczeniowe;
- -"- informacyjne
- -"- specjalne.

1.1. Zadania obliczeniowe

Zadania obliczeniowe dotyczą wszelkich kalkulacji niezbędnych w podejmowaniu decyzji, planowaniu działań bojowych i kierowaniu walką. Są to zadania tego typu jak: obliczanie stosunku sił stron walczących, określenie ilości sił i środków niezbędnych do wykonania określonego zadania oraz podział tych sił i środków, wszelkiego rodzaju obliczenia i kalkulacje, dotyczące planowania przegrupowania wojsk, użycia WRiA oraz sił i środków OPL, wszelkiego rodzaju zabezpieczenia działań bojowych itp.

Z punktu widzenia maszynowego przetwarzania danych zadania obliczeniowe charakteryzują się na ogół niewielkimi ilościami danych wejściowych i wyjściowych o charakterze liczbowym oraz dużą ilością złożonych operacji matematycznych.

1.2. Zadania informacyjne

Zadania informacyjne polegają na gromadzeniu, ciągłym aktualizowaniu, opracowywaniu, wyszukiwaniu i wydawaniu informacji operacyjno-taktycznych, niezbędnych w dowodzeniu do takich procesów jak: analiza otrzymanego zadania, ocena położenia, ocena stanu oraz możliwości sił i środków, podejmowanie decyzji, koordynowanie i kontrolowanie działalności wojsk itp. Dla przykładu mogą to być zadania związane z oceną składu, ugrupowania i działania wojsk własnych i nieprzyjaciela, z sytuacją skażeń promieniotwórczych

i chemicznych w rejonie działań, z sytuacją hydrologiczno-meteorologiczną, tyłową /stany zapasów amunicji, umundurowania, żywności, mps/ itp.

Z reguły do rozwiązywania zadań informacyjnych za pomocą komputera potrzebne są bardzo duże zbiory danych wejściowych, które w toku przetwarzania na dane wyjściowe podlegają na ogół prostym operacjom obliczeniowym.

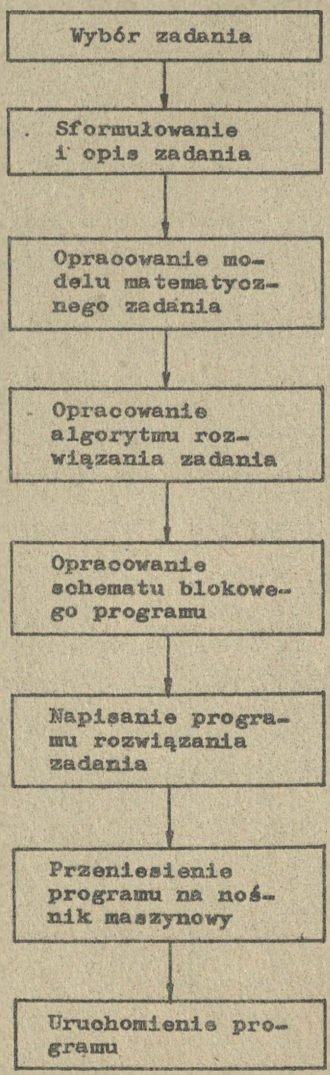
1.3. Zadania specjalne

Zadania specjalne związane są z kierowaniem środkami walki i urządzeniami specjalistycznymi. Zadania tego typu mogą np. dotyczyć: określania danych początkowych do prowadzenia ognia środkami rakietowymi i artyleryjskimi, obliczeń bombardierskich, obliczeń nawigacyjnych /lotniczych lub morskich/, kierowania ruchem lotniskowym itp.

Zadania specjalne rozwiązywane są zwykle przy pomocy uniwersalnych lub specjalizowanych przelicznków, które najczęściej stanowią element składowy danego zestawu, obiektu lub urządzenia i działają wg stałego programu.

2. Procedura rozwiązywania zadań operacyjno-taktycznych za pomocą komputerów

W procesie związanym z przygotowaniem dowolnego, a więc i taktyczno-operacyjnego zadania do rozwiązania za pomocą komputera można wyróżnić następujące, kolejno po sobie realizowane, etapy /rys. 1/.


Wybór zadania
Sformułowanie i opis zadania
Opracowanie modelu matematycznego zadania
Opracowanie algorytmu rozwiązania zadania
Opracowanie schematu blokowego programu
Napisanie programu rozwiązania zadania
Przeniesienie programu na nośnik maszynowy
Uruchomienie programu

Rys. 1.

Zakres prac poszczególnych etapów zależy od treści i złożoności zadań. W konkretnych zadaniach niektóre z wymienionych etapów mogą nie wystąpić. Np. zadania proste nie będą wymagać opracowania modelu matematycznego, czy też pełnego opisu i szczegółowego algorytmu. Dla szeregu zadań z dziedziny analizy matematycznej, statystyki, badań operacyjnych i wielu innych dziedzin, algorytmy rozwiązania są znane i wystarczy je tylko dobrać bądź odpowiednio przystosować do rozwiązania komputerowego.

2.1. Wybór zadania

Decydując się na wykorzystanie komputerów do rozwiązywania zadań operacyjno-taktycznych, należy uzmysłowić sobie fakt, że nie zachodzi potrzeba i nie opłaca się rozwiązywać wszelkich zadań tego typu w sposób zautomatyzowany. Dlatego też przy wyborze zadań operacyjno-taktycznych do rozwiązywania komputerowego należy kierować się następującymi względami:

- rangą zadania w procesie dowodzenia;
- czasem rozwiązywania zadania;
- złożonością i dokładnością obliczeń;
- powtarzalnością użytkowania zadania;
- możliwością praktycznej realizacji zadania na dostępnych komputerach.

2.1.1. Ranga zadania

Do rozwiązywania za pomocą komputerów powinno dobierać się przede wszystkim te spośród zadań operacyjno-taktycznych, które wynikają z faktycznych potrzeb organów dowodzenia w zakresie analizy sytuacji bojowej, przygotowania danych do decyzji, planowania i organizacji walki oraz kierowania działaniami bojowymi.

Zautomatyzowanie rozwiązywania tych zadań powinno skrócić cykl dowodzenia wojskami, zmniejszyć czas reakcji systemu dowodzenia na istotne zjawiska zachodzące na polu walki i zapewnić wyższą jakość wyników w porównaniu z tradycyjnymi sposobami rozwiązania.

2.1.2. Czas rozwiązywania zadania

Czas rozwiązania zadania operacyjno-taktycznego za pomocą komputera powinien być krótszy od czasu rozwiązywania metodami tradycyjnymi. W szczególnych przypadkach warunek ten może być potraktowany drugorzędnie, np. jeśli użycie komputera uwalnia oficerów sztabu od żmudnych oraz pracochłonnych obliczeń na rzecz pracy organizacyjno-koncepcyjnej lub zapewnia uzyskanie znacznie lepszych rozwiązań. W żadnym jednak przypadku czas rozwiązywania zadania za pomocą komputera nie może przekroczyć czasu wynikającego z potrzeb procesu dowodzenia, albowiem nawet najcenniejsze pod względem operacyjno-taktycznym wyniki stają się bezsensowne, jeśli nie zostaną przekazane decydentowi w odpowiednim czasie.

2.1.3. Złożoność i dokładność obliczeń

Rozwiązywanie zadań za pomocą komputerów jest uzasadnione również ich złożonością oraz koniecznością zastosowania dużej dokładności obliczeń. Szereg zadań operacyjno-taktycznych ma właśnie taki charakter. Wymagają one bowiem uwzględnienia bardzo wielu czynników, dotyczących zarówno wojsk własnych jak i przeciwnika, różnych jego prawdopodobnych wariantów działań, warunków terenowych, meteorologicznych i wielu innych.

Np. zaplanowanie przegrupowania wojsk wymaga uwzględnienia takich czynników, jak: sposób przegrupowania; rodzaj i stan ilościowy wojsk, sprzętu bojowego i środków transportowych; lokalizacja rejonów wyjściowych, pośrednich i końcowych oraz punktów

przejścia i linii wyrównania; rodzaje odpoczynków, ich miejsca oraz czasy trwania jak również wiele innych czynników.

Najczęściej problemy operacyjno-taktyczne, wynikające podczas różnych sytuacji bojowych, sprowadzają się do gier o niepełnej informacji, a to wymaga przeanalizowania dużej liczby wariantów rozwiązań, aby stworzyć podstawę do wyboru strategii optymalnej.

2.1.4. Powtarzalność użytkowania zadania

Różne zadania operacyjno-taktyczne stosowane są z różną częstotliwością. Niektóre z nich rozwiązywane są tylko w okresie poprzedzającym działania bojowe, jak np. planowanie przegrupowania wojsk. Szereg zadań, zwłaszcza podczas prowadzenia działań bojowych, użytkowanych jest z dużą powtarzalnością. Należą tu np. takie zadania jak: obliczanie stosunku sił, analiza sytuacji powietrznej, ocena skażeń promieniotwórczych oraz chemicznych itp.

Z punktu widzenia powtarzalności najbardziej celowym jest rozwiązywanie za pomocą komputerów tych zadań, które wykorzystywane są wielokrotnie, nawet gdyby były proste w realizacji. Należy jednak zdawać sobie sprawę z tego, że nawet niepowtarzalne zadania należy rozwiązywać w sposób zautomatyzowany, jeżeli przemawia za tym ich ranga operacyjna. Taki przykład stanowić mogą zadania związane z mobilizacyjnym i operacyjnym rozwinięciem wojsk.

2.1.5. Możliwość praktycznej realizacji zadania na dostępnych komputerach

Współczesne pole walki nastręcza tak dalece złożonych problemów, że nie jest możliwe rozwiązanie ich na czas przy użyciu dostępnych komputerów. W takich przypadkach należy pomijać wiele czynników drugorzędnych i rezygnować z dokładnych metod rozwiązań pod warunkiem jednak, że uzyskane wyniki zadowolą decydenta.

Z drugiej strony istnieją zadania, które są ważne pod względem operacyjno-taktycznym, ale z uwagi na wymagany bardzo ograniczony czas rozwiązywania, nie mogą być realizowane przy użyciu dostępnych komputerów oraz środków zbierania źródłowych danych jak również środków transmisji tych danych i wyników ich przetwarzania. Są to z reguły zadania wymagające przetwarzania w czasie rzeczywistym, a to z kolei możliwe jest tylko w przypadku posiadania odpowiedniego systemu wyposażonego w nowoczesne, bardzo szybkie i niezawodne komputery, współpracujące z nowoczesnymi środkami transmisji danych, z nowoczesnymi stacjami radiolokacyjnymi bądź innymi automatycznymi nadajnikami danych, urządzeniami automatycznego zobrazowania sytuacji itp.

2.2. Sformułowanie i opis zadania

Sformułowania i opisu zadania, rozwiązywanego za pomocą komputera, dokonuje jego użytkownik. W przypadku więc zadań operacyjno-taktycznych rola ta spoczywa na specjalistach organów dowodzenia, ponieważ tylko oni znają potrzeby tych organów i są merytorycznie przygotowani do prawidłowego sformułowania problemu oraz określenia związanych z nim założeń i ograniczeń. W przypadkach szczególnych, gdy w grę wchodzą zadania złożone pod względem matematyczno-logicznym, przy ich formułowaniu i opisie powinni współuczestniczyć specjaliści z zakresu algorytmizacji i programowania.

Zadanie należy formułować i opisywać w sposób krótki, jasny i jednoznaczny. W sformułowaniu zadania podaje się jego nazwę i cel.

2.2.1. Nazwa zadania

Nazwa zadania powinna odzwierciedlać jego istotę i określać jego użytkownika. Może mieć ona np. następujące brzmienie: "Planowanie przegrupowania związku taktycznego własnym transportem" lub "Obliczenie nasycenia i stosunku sił na szczeblach taktycznych i operacyjnych".

2.2.2. Cel zadania

Cel zadania powinien ukierunkowywać jego opracowanie i określać, co chcemy uzyskać poprzez jego rozwiązanie. Np. zadanie "Planowanie przegrupowania związku taktycznego własnym transportem" może realizować następujące cele: usprawnienie i podwyższenie dokładności obliczeń, określenie czasu potrzebnego na przegrupowanie, optymalizacja wykorzystania przepustowości dróg marszu, usprawnienie przekazywania zadań dla wykonawców itp.

W opisie zadania powinny znaleźć się założenia i ograniczenia dotyczące jego rozwiązania, określenie danych wejściowych oraz danych wyjściowych, czyli wyników rozwiązania, określenie kryteriów efektywności rozwiązania jak również inne dodatkowe wyjaśnienia.

2.2.3. Założenia i ograniczenia

Założenia i ograniczenia, dotyczące rozwiązania zadania, wynikają z obowiązujących norm operacyjno-taktycznych i technicznych, możliwości środków bojowych, dopuszczalnych kosztów, obowiązującego czasu rozwiązania zadania, właściwości technicznych komputera, na którym zadanie będzie rozwiązywane itp. Założenia i ograniczenia mogą wywrzeć bezpośredni wpływ na przyjęcie metody rozwiązania zadania.

W zadaniu "Planowanie przegrupowania związku taktycznego własnym transportem" występują założenia i ograniczenia, wynikające z normatywów regulaminowych i doraźnie przyjętych, a dotyczące parametrów ugrupowania masrszowego, marszu, marszrut, rodzajów oraz terminów i czasów trwania odpoczynków i wielu innych zagadnień. Wystąpią tu również ograniczenia wynikające z zakresu, dokładności i wariantów obliczeń, czasu na przygotowanie danych wejściowych, czasu rozwiązywania zadania na komputerze i przekazywania wyników użytkownikom itp.

2.2.4. Określenie danych wejściowych do rozwiązania zadania

Używając terminu dane wejściowe, mamy na uwadze wprowadzane do komputera informacje, na których wykonuje on odpowiedni ciąg operacji arytmetycznych, logicznych i organizacyjnych w celu uzyskania wyników zadania.

Wśród danych wejściowych wyodrębnia się dane stałe oraz dane zmienne.

Stałymi nazywamy te dane wejściowe, które w ciągu dłuższego czasu nie zmieniają swoich wartości i mogą być wielokrotnie używane do rozwiązywania tego samego zadania. W przypadku zadań operacyjno-taktycznych taki charakter mają dane dotyczące organizacji i stanu etatowego pododdziałów, oddziałów i związków taktycznych, wszelkiego rodzaju normy operacyjno-taktyczne i techniczne, parametry uzbrojenia itp.

Np. w zadaniu "Planowanie przegrupowania związku taktycznego własnym transportem" do danych stałych zaliczymy parametry wozów bojowych i pojazdów mechanicznych wpływające na szybkość i długość kolumn, odległości między pododdziałami itp.

Dane stałe przechowuje się w pamięci zewnętrznej lub wprowadza do komputera wraz z programem przed uruchamianiem zadania.

Zmiennymi nazywamy takie dane wejściowe, które przy każdorazowym rozwiązywaniu tego samego zadania przyjmują różne wartości. Mogą to być np. dane z rozpoznania dotyczące położenia wojsk przeciwnika, o stanie ukompletowania i zaopatrzenia wojsk własnych, o sytuacji hydrologiczno-meteorologicznej itp.

W rozpatrywanym przykładzie, dotyczącym przegrupowania związku taktycznego, do danych zmiennych zaliczymy nazwy kolumn, klasy dróg, miejsca rejonów wyjściowych, pośrednich i końcowych, parametry charakteryzujące warunki atmosferyczne itp.

Dane zmienne wprowadza się do maszyny bezpośrednio przed uruchomieniem zadania.

2.2.5. Określenie danych wyjściowych /wyników zadania/

Treść, postać i forma danych wyjściowych powinny być określone przez użytkownika, odpowiadać jego potrzebom i spełniać jego wymagania. Jeżeli z wyników zadania korzysta kilku użytkowników do różnych celów, to pożądane jest, ze względu na przejrzystość oraz łatwość w posługiwaniu się, aby poszczególni użytkownicy otrzymywali tylko te dane, którymi są zainteresowani.

W przykładzie, dotyczącym przegrupowania związku taktycznego, wyniki wyprowadza się w postaci wydruku zawierającego tabele marszu dla poszczególnych kolumn marszowych. W tabelach tych znajdują się dane niezbędne do organizacji przegrupowania, takie np. jak numery dróg, po których poruszać się będą poszczególne kolumny, czas opuszczenia rejonu wyjściowego, czas osiągnięcia charakterystycznych punktów na drogach przemarszu i rejonu końcowego, prędkość marszu na określonych odcinkach dróg, miejsca i czasy trwania odpoczynków itp.

Z reguły przy określaniu danych wyjściowych użytkownik ustala także ich dokładność.

2.2.6. Wyjaśnienia dodatkowe

W wyjaśnieniach dodatkowych opisu zadania podaje się w różnej postaci /opis, tabela, wykres, schemat/ wszelkie informacje, które mogą być istotne w realizacji kolejnych etapów przygotowania zadania do rozwiązania za pomocą komputera.

Etap sformułowania i opisu zadania może w praktyce być bardzo owocny, gdyż zmusza on użytkownika do dokładnego i ścisłego opisu problemu i jego dogłębnego zrozumienia, co w dużym stopniu ułatwia drogi rozwiązania problemu. W szczególnych przypadkach może już na tym etapie okazać się, że problem bardziej racjonalnie może być rozwiązany bez użycia komputera.

2.3. Opracowanie modelu matematycznego zadania

Po sformułowaniu i opisaniu zadania można przystąpić do opracowania jego modelu matematycznego. Jest to jeden z najtrudniejszych etapów przygotowania zadania do rozwiązania za pomocą komputera, gdyż wymaga często zastosowania złożonego aparatu matematycznego, ma charakter twórczy i może być bardzo czasochłonny.

Dla zadań operacyjno-taktycznych o charakterze informacyjno-ewidencyjnym, typowych np. dla organów tyłowych, opracowanie modelu matematycznego nie jest procesem złożonym, ponieważ związki między wielkościami występującymi w tych zadaniach można wyrazić za pomocą arytmetyki, algebry liniowej lub statystyki.

Sprawa bardziej komplikuje się, gdy w grę wchodzą zadania, w których występuje zagadnienie optymalizacji.

Traktując rzecz ogólnie, istota budowy modelu matematycznego takich zadań przedstawia się następująco. Wynikające z treści zadania założenia, warunki i ograniczenia zawężają zbiór możliwych jego rozwiązań. Oznaczmy ten zbiór literą D. Na elementach zbioru D określona jest pewna funkcja $F(X)$, zwana funkcją-kryterium i przypisująca każdemu rozwiązaniu $X = (x_1, x_2, \ldots, x_n)$ określoną liczbę rzeczywistą, którą będziemy traktować jako efektywność rozwiązania. W zależności od treści zadania jego optymalizacja może polegać na minimalizacji lub maksymalizacji funkcji-kryterium. W przypadku, kiedy maksymalizujemy $F(X)$, rozwiązanie X_1 jest lepsze od rozwiązania X_2, jeśli $F(X_1) > F(X_2)$. Jeśli natomiast minimalizujemy $F(X)$, to rozwiązanie $F(X_1)$ jest lepsze od rozwiązania $F(X_2)$, jeśli $F(X_1) < F(X_2)$. Wśród wszystkich dopuszczalnych rozwiązań $X \in D$ może znaleźć się rozwiązanie X_0 nie gorsze od każdego innego, które nazywać będziemy rozwiązaniem optymalnym. Rozwiązanie to w przypadku maksymalizacji spełni warunek: $F(X_0) \geq F(X)$, zaś minimalizacji warunek: $F(X_0) \leq F(X)$.

Jak wynika z rozważania, może okazać się, że zadanie posiada szereg optymalnych rozwiązań. W takich przypadkach należy zastosować dodatkowe kryteria, które spowodują ograniczenie liczby rozwiązań optymalnych do jednego lub tylko kilku.

Przy modelowaniu zadań optymalizacyjnych najtrudniejszą sprawą jest określenie funkcji-kryterium oraz ułożenie na podstawie założeń, warunków i ograniczeń odpowiednich układów równań, nierówności i innych warunków logicznych, określających jednoznacznie zbiór dopuszczalnych rozwiązań.

Modelowanie matematyczne zadań operacyjno-taktycznych, szczególnie z zakresu podejmowania decyzji, jest procesem bardzo złożonym. Brak jest bowiem tradycji w stosowaniu aparatu matema-

tycznego w rozwiązywaniu zadań tego typu. Każde niemal zadanie operacyjno-taktyczne charakteryzuje się pewną specyfiką wymagającą stosowania w rozwiązaniu kilku metod lub opracowania metody całkiem nowej. Dlatego też modele matematyczne bardziej złożonych zadań operacyjno-taktycznych powinny być opracowywane przez zespoły złożone ze specjalistów znających doskonale problem tkwiący w treści zadania, jak również współczesne metody matematyczne.

Na marginesie, aby zdać sobie sprawę ze złożoności modelowania matematycznego, warto wspomnieć o typach modeli decyzyjnych. Otóż ze względu na charakter parametrów problemu występującego w zadaniu, rozróżniamy dwie grupy modeli matematycznych: deterministyczne oraz indeterministyczne. Wśród indeterministycznych wyróżnimy modele: probabilistyczne, statystyczne i strategiczne.

Model deterministyczny ma miejsce wówczas, gdy wszystkie parametry problemu mają stałe i znane wartości. Do rozwiązywania modeli tego typu stosowany jest szeroko rachunek różniczkowy pozwalający znaleźć ekstrema funkcji jednej bądź wielu zmiennych. W szczególnych przypadkach do rozwiązania modeli deterministycznych stosuje się również programowanie liniowe lub nieliniowe.

Model probabilistyczny charakteryzuje się tym, że występuje w nim co najmniej jeden parametr problemu, który jest zmienną losową o znanym rozkładzie prawdopodobieństwa. Narzędzie do rozwiązywania takich modeli stanowi rachunek prawdopodobieństwa wraz z kombinatoryką.

Model statystyczny wyróżnia ten fakt, że co najmniej jeden parametr problemu jest zmienną losową o nieznanym rozkładzie prawdopodobieństwa. Do rozwiązywania modeli tego typu znajduje zastosowanie rachunek prawdopodobieństwa, statystyka matematyczna oraz metody symulacyjne, których sens sprowadza się do wielo-

krotnego odtwarzania procesu za pomocą komputera. W szczególnych przypadkach do rozwiązywania modeli statystycznych może znaleźć zastosowanie również teoria masowej obsługi.

Model strategiczny występuje wtedy, gdy co najmniej jeden z parametrów problemu zależy od decyzji przeciwnika. Modele tej klasy rozwiązywane są przy pomocy teorii gier, szczególnie gier macierzowych oraz gier o niepełnej informacji.

W rozwiązywaniu modeli matematycznych wykorzystywane jest również programowanie dynamiczne, stosowane nie tylko przy rozpatrywaniu procesów dynamicznych, ale także statycznych. Do metod programowania dynamicznego zalicza się analizę funkcjonalną, procesy stochastyczne, rachunek wariacyjny i inne.

2.4. Opracowanie algorytmu rozwiązania zadania

Komputer rozwiązuje zadania w sposób automatyczny wg programu umieszczonego w jego pamięci i opracowanego przez człowieka. Programować natomiast można tylko takie zadania, dla których istnieją algorytmy rozwiązania.

Podstawę do algorytmizacji prostych zadań stanowi ich sformułowanie i opis, a w przypadkach bardziej złożonych także model matematyczny. Algorytmizacją zadań zajmują się analitycy /problemiści/.

2.4.1. Pojęcie algorytmu

Aczkolwiek pojęcie algorytmu jest od dawna znane w matematyce /algorytm Euklidesa znajdowania wspólnego podzielnika dwóch liczb naturalnych - III/IV wiek p.n.e., algorytmy Al-Chorezmiego działań arytmetycznych w systemie dziesiętnym - IX wiek/ jako jedno z pojęć podstawowych, do tej pory nie zostało

ono ściśle zdefiniowane. Do naszych celów wystarczy intuicyjne pojęcie algorytmu, które można sformułować następująco.

> Algorytm jest to ścisły opis metody postępowania, całkowicie i jednoznacznie określający, co należy robić w każdej fazie rozwiązywania zadania i w każdej sytuacji, jaka może przy tym powstać.

Po pojawieniu się komputerów i w miarę ich bardzo szybkiego rozwoju oraz szerokiego zastosowania do rozwiązywania problemów z różnych dziedzin, zrodziła się pilna potrzeba stworzenia teorii algorytmów, która obecnie stanowi już odrębny dział matematyki.

Teoria algorytmów zajmuje się sposobami ich formułowania, właściwościami, dobieraniem do różnych problemów, porównywaniem ich skuteczności itp.

Użytkowników komputerów interesuje nie tylko znalezienie algorytmów rozstrzygających rozwiązywalność danych problemów, ale algorytmów nadających się do realizacji za pomocą komputerów. Algorytmy takie muszą mieć strukturę przystosowaną do możliwości komputerów i dlatego też możemy je nazwać algorytmami maszynowymi.

2.4.2. Cechy algorytmów

Algorytmy charakteryzują się pewnymi cechami, do których przede wszystkim należy zaliczyć: dyskretność, deterministyczność i masowość.

Dyskretność oznacza, że algorytm jest to proces stopniowego budowania pewnych wielkości w skończonej liczbie kroków. Pierwszy krok tego procesu realizowany jest na podstawie pewnych zadanych wielkości, zwanych danymi. W każdym kroku powstaje nowy system wielkości zwany wynikami tego kroku.

Deterministyczność oznacza, że wyniki pierwszego kroku zależą jednoznacznie od danych i tylko od nich, a wyniki każdego kroku następnego zależą jednoznacznie od wyników kroku poprzedniego.

Masowość oznacza, że dane algorytmu można wybierać ze zbioru, który jest potencjalnie nieskończony /inaczej: algorytm podaje rozwiązanie nie dla jednego zadania, lecz dla wszystkich zadań jednego typu/.

2.4.3. Sposoby wyrażania algorytmów

Algorytmy można wyrazić w postaci opisowej, analitycznej, schematu blokowego, wykreślnej lub operatorowej.

Opracowując algorytmy należy dobierać taką ich postać, która byłaby jak najprostsza, zwięzła i przejrzysta. Algorytmy zadań, w tym również operacyjno-taktycznych, najwygodniej jest przedstawiać w postaci schematów blokowych. Algorytmy prostych zadań można również łatwo wyrazić w postaci opisowej lub analitycznej. Z tych też względów zajmiemy się omówieniem tylko tych trzech postaci.

2.4.2.1. Wyrażanie algorytmu w postaci opisowej

W tym przypadku sposób realizacji poszczególnych kroków algorytmu przedstawiony jest w postaci opisu. W taki sposób zapoznawano nas w szkole podstawowej z algorytmami dodawania, odejmowania, mnożenia i dzielenia słupkowego. Dla przykładu rozpatrzymy opisowy algorytm zamiany liczb całkowitych z systemu dziesiętnego na dwójkowy.

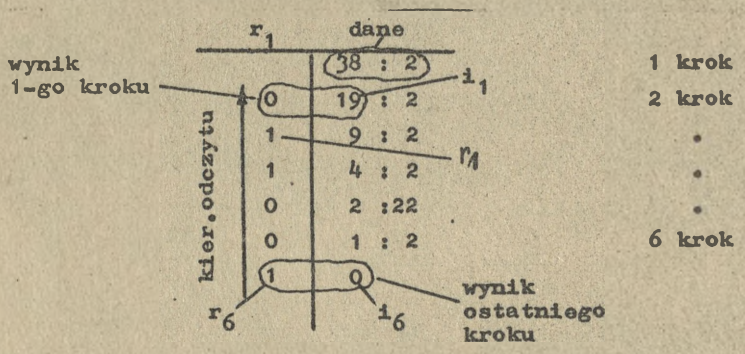
1 krok. Podziel daną liczbę dziesiętną przez 2; w wyniku otrzymasz pierwszy iloraz cząstkowy i_1 oraz resztę z dzielenia r_1.

2 krok. Podziel otrzymany iloraz i_1 przez 2; w wyniku otrzymasz drugi iloraz cząstkowy i_2 oraz resztę z dzielenia r_2.

Krok uogólniony. Kontynuuj ten proces aż do n-tego kroku, po którym n-ty iloraz cząstkowy i_n osiągnie wartość równą zeru.

Wynik działania algorytmu daje ciąg kolejnych reszt odczytywanych od końca.

Przykład. Wyraź liczbę 38 za pomocą systemu dwójkowego.



38 = 100 110

Jak widać, algorytm zamiany liczby 38 na system dwójkowy został zrealizowany w 6-ciu krokach.

2.4.2.2. Wyrażanie algorytmu w postaci analitycznej

W tym przypadku algorytmy wyrażane są za pomocą wzorów. Postać ta wygodna jest przy rozwiązywaniu zadań o charakterze numerycznym. Taki charakter mają np. zadania z zakresu bombardowania, strzelań artylerii naziemnej i przeciwlotniczej, strzelań rakietowych itp.

Przykład. Posiadamy możliwość oddania do celu n strzałów. Obliczyć prawdopodobieństwo trafienia celu co najmniej jednym strzałem, jeżeli znamy prawdopodobieństwo trafienia celu jednym strzałem.

Algorytm rozwiązania tego zadania stanowi znany z rachunku prawdopodobieństwa wzór:

$$p_{1,n} = 1 - (1 - p)^n$$

gdzie: p - prawdopodobieństwo trafienia celu jednym strzałem;

n - ilość strzałów oddanych do celu;

$p_{1,n}$ - prawdopodobieństwo trafienia celu co najmniej jednym strzałem przy oddaniu n strzałów.

2.4.2.3. Wyrażanie algorytmu w postaci schematu blokowego

Algorytm w postaci schematu blokowego, zwanego też schematem czynnościowym, stanowi graficzne przedstawienie rodzajów operacji i kolejności ich wykonywania w toku rozwiązywania zadania. Poszczególne etapy czynności związanych z rozwiązywaniem zadania przedstawia się za pomocą figur geometrycznych, zwanych blokami. We wnętrzu bloków definiuje się elementarne lub złożone operacje, jakie należy wykonywać na danych wejściowych, wynikach pośrednich lub końcowych. Operacje te mogą być wyrażone opisowo, za pomocą wzorów matematycznych lub przy użyciu symboli teorii, której dotyczy rozwiązywane zadanie /wzory matematyczne, warunki logiczne itp./. Bloki łączy się między sobą liniami zaopatrzonymi w strzałki, dzięki czemu powstaje schemat struktury algorytmu obrazujący jego wszystkie połączenia, rozgałęzienia oraz cykle.

Na schematach blokowych algorytmów zadań, przewidywanych do rozwiązania za pomocą komputera, uwzględnia się nie tylko operacje matematyczne i logiczne, ale także organizacyjne. W ten sposób schemat blokowy umożliwia bardzo szczegółowe opisa-

nie tych wszystkich czynności elementarnych, jakie komputer ma wykonać podczas automatycznego rozwiązywania zadania. Tak opracowane schematy blokowe zawierają pełną informację, niezbędną do zaprogramowania zadania w odpowiednim języku programowania, przy czym programista nie musi znać problemu, którego zadanie dotyczy, aby napisać prawidłowy program jego rozwiązania. Dlatego też schematy blokowe algorytmów nazywane są również schematami blokowymi programów, a sam proces algorytmizacji określany jest mianem programowania blokowego lub graficznego.

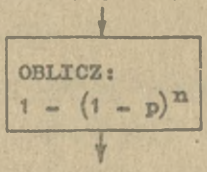
Programowanie blokowe stanowi swego rodzaju język graficzny, który z jednej strony w komunikatywnej formie przedstawia istotę problemu i sposób jego rozwiązania, a z drugiej - sposób zaprogramowania rozwiązania tego problemu w języku zrozumiałym dla komputera. Programowanie blokowe ułatwia również kontrolę przyjętych założeń, pozwala prześledzić funkcjonowanie przyjętej metody rozwiązania i szybko zorientować się w organizacji programu. Dlatego też prawie dla każdego zadania, przewidzianego do rozwiązania za pomocą komputera, opracowuje się schemat blokowy programu. Schemat taki sporządzany jest na podstawie algorytmu wyrażonego w innej postaci bądź stanowi jedyną postać algorytmu rozwiązania zadania.

2.5. Opracowanie schematu blokowego programu

2.5.1. Symbole graficzne stosowane w programowaniu blokowym

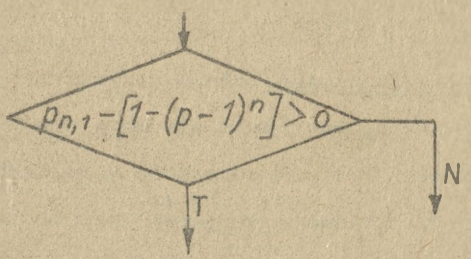
Zestaw symboli graficznych, stosowanych w programowaniu blokowym, został znormalizowany wg PN-72 E-01226 - Przetwarzanie danych symbole graficzne. Zgodnie z przytoczoną normą poszczególnym blokom nadaje się następujące nazwy, kształty i znaczenia.

Blok obliczeniowy przedstawiany jest w postaci prostokąta z jednym wejściem i jednym wyjściem. Wewnątrz bloku definiuje się rodzaj operacji lub grupy operacji oraz określa nazwy zmiennych użytych w obliczeniach /rys. 2/.



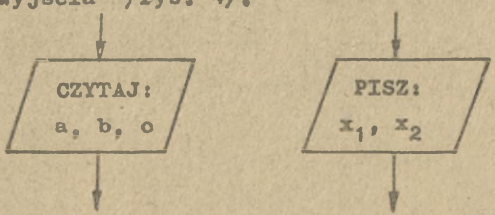
Rys. 2

Blok decyzyjny ma postać rombu z jednym wejściem oraz dwoma wyjściami. Określa on, jaką gałąź programu należy realizować w przypadku spełnienia /T/ lub niespełnienia /N/ pewnego warunku, zdefiniowanego wewnątrz bloku /rys. 3/.



Rys. 3.

Blok wejścia/wyjścia przedstawiany jest w postaci równoległoboku z jednym wejściem oraz jednym wyjściem i oznacza operację wprowadzania /czytania/ danych do maszyny lub wyprowadzania /pisania/ wyników. Wewnątrz bloku określa się rodzaj czynności oraz nazwy zmiennych zgodnie z kolejnością ich występowania w operacji wejścia/wyjścia /rys. 4/.



Rys. 4.

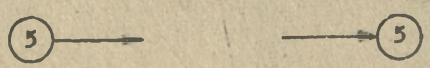
Za pomocą opisanych powyżej trzech bloków można przedstawić przebieg przetwarzania danych podczas rozwiązywania dowolnego zadania. Dlatego te bloki traktujemy jako podstawowe. Blok obliczeniowy oraz blok wejścia/wyjścia nazywać będziemy blokami wykonawczymi.

Przystąpimy z kolei do omówienia bloków pomocniczych, odgrywających w strukturze schematów rolę drugorzędną.

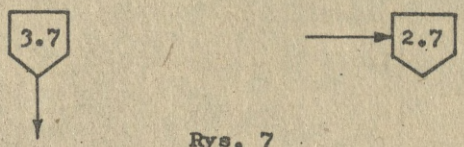
Blok graniczny posiada kształt owalu i oznacza początek lub koniec schematu blokowego bądź przerwanie działań /rys. 5/.

Rys. 5.

Bloki łącznikowe /łączniki/ służą do łączenia odrębnych części schematu blokowego. Rozróżnia się przy tym bloki łącznikowe wewnątrzstronicowe /rys. 6/, oznaczane w ppstaci koła i stosowane do łączenia części schematu występujących na tej samej stronie, oraz bloki łącznikowe międzystronicowe /rys. 7/, oznaczane pięciokątem i służące do łączenia odrębnych części schematu blokowego, występujących na oddzielnych stronach. Odpowiadające sobie pary bloków łącznikowych oznaczone są umownymi symbolami, najczęściej liczbami naturalnymi. W przypadku łączników międzystronicowych stosuje się symbole dwucyfrowe, w których pierwsza cyfra oznacza nr strony, druga - numer bloku łącznikowego.

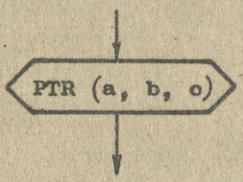


Rys. 6.



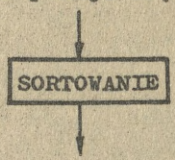
Rys. 7

Blok podprogramu przedstawiany jest za pomocą spłaszczonego sześciokąta i oznacza zmianę w przebiegu programu z powodu przejścia do realizacji podprogramu. We wnętrzu bloku umieszcza się nazwę podprogramu oraz jego ewentualne parametry /rys. 8/.



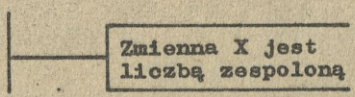
Rys. 8.

Blok fragmentu wyrażary jest figurą o kształcie prostokąta i stosowany do oznaczania fragmentu programu zdefiniowanego oddzielnie. Wewnątrz bloku podaje się nazwę fragmentu /rys. 9/.



Rys. 9.

Blok komentarza nie wyraża żadnej operacji, lecz zawiera tylko wyjaśnienia dla użytkownika schematu, ułatwiające mu zrozumienie poszczególnych elementów schematu bądź identyfikację zmiennych i parametrów rozwiązywanego zadania /rys. 10/.



Rys. 10.

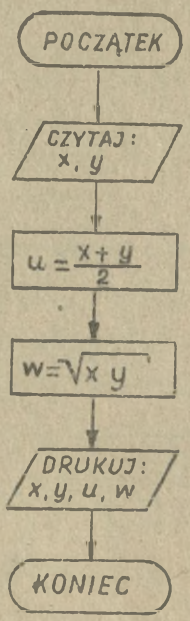
2.5.2. Typy schematów blokowych /programów/

Schematy blokowe, a tym samym programy, mogą posiadać strukturę liniową lub cykliczną.

Liniowymi nazywamy takie schematy blokowe /programy/, w których nie występują ciągi czynności powtarzalnych. Wśród schematów liniowych wyróżnić można schematy rozgałęzione i nie-rozgałęzione. W schematach rozgałęzionych, oprócz bloków wykonawczych i pomocniczych, występuje przynajmniej jeden blok decyzyjny. W schematach liniowych nie-rozgałęzionych występują tylko bloki wykonawcze i pomocnicze.

Przykład /schemat liniowy nie-rozgałęziony/

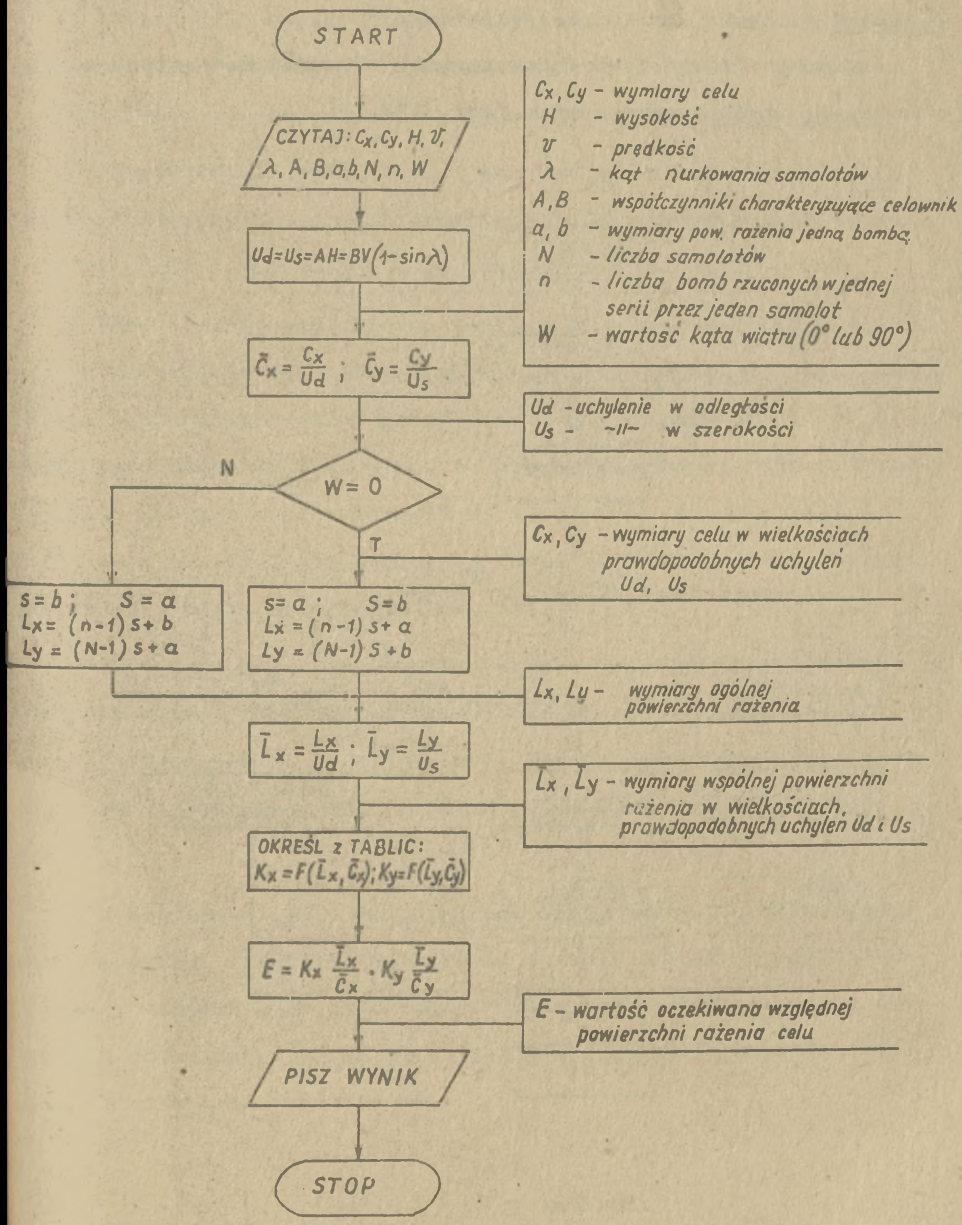
Obliczyć średnią arytmetyczną "u" oraz średnią geometryczną "w" liczb x,y. Dane znajdują się na karcie perforowanej, a wyniki należy wyprowadzić w postaci wydruku /rys. 11/.



Rys. 11.

Przykład /schemat liniowy rozgałęziony/

Obliczyć wartość oczekiwaną względnej powierzchni rażenia celu bombami odłamkowo-burzącymi przez grupę samolotów wykonujących atak z lotu nurkowego /rys. 12/.

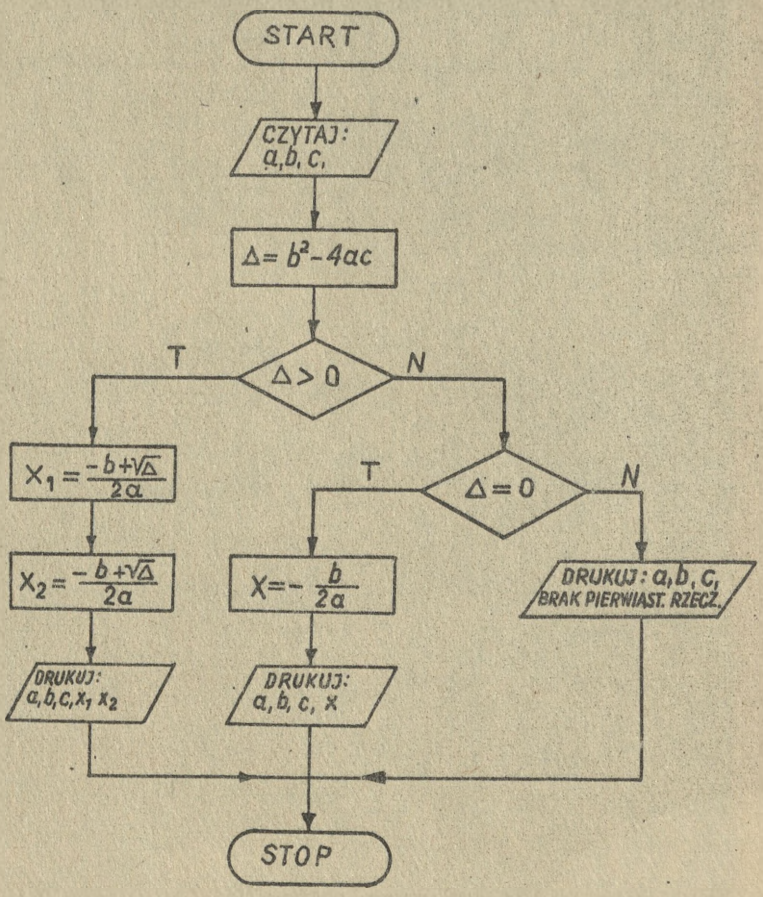


Rys. 12.

Bardzo pouczający przykład budowania rozgałęzionych schematów blokowych stanowi algorytm rozwiązywania równania kwadratowego.

Przykład /schemat liniowy rozgałęziony/

Zaprogramować blokowe rozwiązanie równania kwadratowego w zakresie liczb rzeczywistych /rys. 13/.



Ry. 13.

Cyklicznymi, w przeciwieństwie do liniowych, nazywamy schematy blokowe /programy/, w których występuje pewna ilość czynności powtarzalnych.

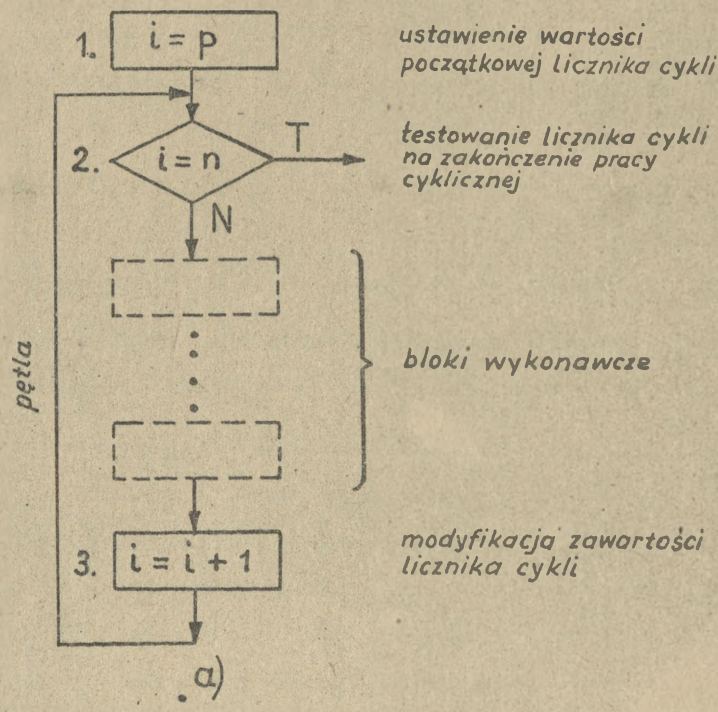
Wśród cyklicznych wyróżnia się schematy ze znaną liczbą n powtórzeń oraz schematy z nieznaną liczbą powtórzeń. Te drugie, w odróżnieniu od pierwszych, nazywamy schematami iteracyjnymi.

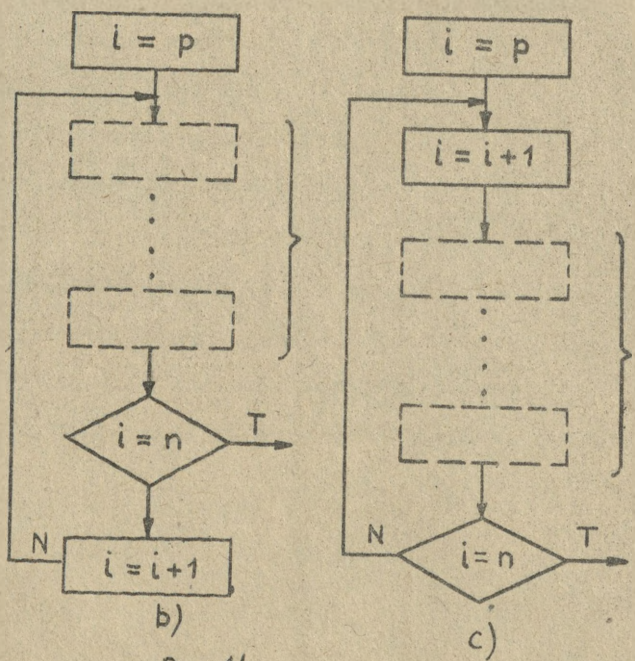
Przetwarzanie cykliczne, zwane też przetwarzaniem w pętli, wymaga dodatkowych operacji organizacyjnych, które określają parametry cyklu i zapewniają jego realizację.

Przy organizowaniu pracy cyklicznej o znanej liczbie powtórzeń n przeznacza się jedną komórkę pamięci operacyjnej na licznik cykli. Przed rozpoczęciem pracy cyklicznej ustawia się wartość początkową licznika p /i = p/. Po wykonaniu każdego cyklu wartość licznika poddaje się testowaniu na zakończenie pracy w cyklu. Jeśli cykl ma być kontynuowany, zawartość licznika cykli podlega modyfikacji przez dodanie jedynki.

Do tak realizowanego cyklu wymagane są trzy dodatkowe operacje organizacyjne, którym na schemacie blokowym odpowiadają trzy dodatkowe bloki /1,2,3 na rys. 14a/. Organizacja cyklu o znanej liczbie powtórzeń może odbywać się kilkoma sposobami /rys. 14 a,b,c/.

W przypadku cyklu iteracyjnego ilość powtórzeń określonego ciągu operacji zależy od spełnienia pewnego warunku, wynikającego z treści zadania bądź organizacji przetwarzania. Dlatego też dla cyklu iteracyjnego, z wyjątkiem pewnych szczególnych przypadków, licznika cykli nie zakłada się.

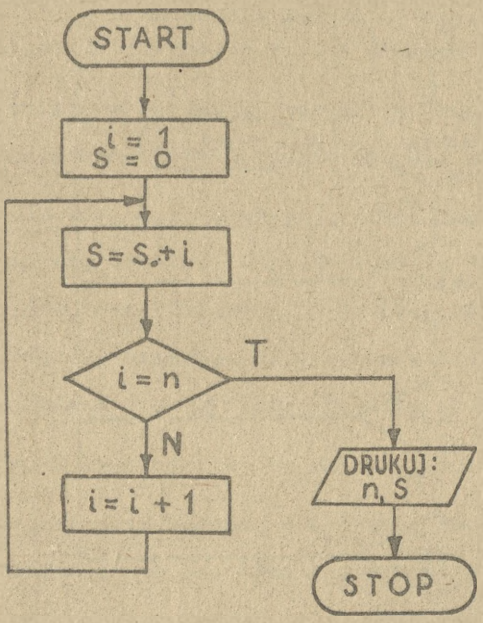

1.
i = p
ustawienie wartości początkowej licznika cykli
2.
i = n
T
N
testowanie licznika cykli na zakończenie pracy cyklicznej
pętla
bloki wykonawcze
3.
i = i + 1
modyfikacja zawartości licznika cykli
a)


i = p
i = n
T
N
i = i+1
b)
i = p
i = i+1
N
i = n
T
c)

Rys. 14.

Przykład /cykl o znanej liczbie powtórzeń/

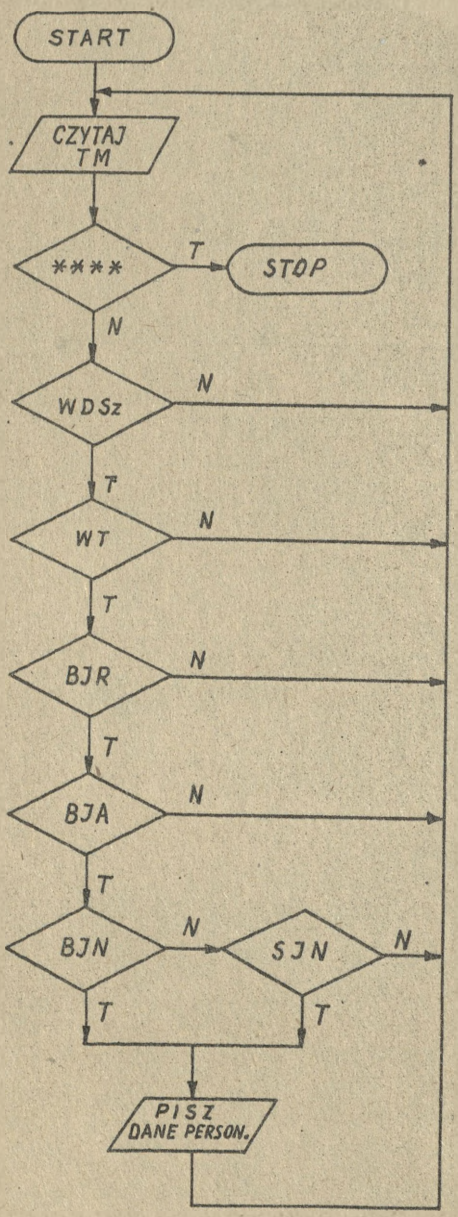
Narysować schemat blokowy programu zliczającego sumę n pierwszych kolejnych liczb naturalnych nieparzystych. Wynik wyprowadzić w postaci wydruku /rys. 15/.



Rys. 15.

Przykład /cykl iteracyjny/

Na taśmie magnetycznej zapisany jest zbiór danych personalnych wszystkich oficerów WP. Zaprogramować blokowo wydrukowanie listy zawierającej stopień, nazwisko, imię, rok urodzenia oraz miejsce służby tych oficerów, którzy ukończyli wyższe studia dowódczo-sztabowe i techniczne oraz znają biegle język rosyjski i angielski, jak również biegle lub słabo język niemiecki. Dane personalne każdego oficera zapisane są na oddzielnym rekordzie. Koniec zbioru oznaczony jest rekordem zawierającym symbol złożony z czterech gwiazdek /rys. 16/.


START
CZYTAJ TM
****
T
STOP
N
WDSz
N
T
WT
N
T
BJR
N
T
BJA
N
T
BJN
N
SJN
N
T
T
PISZ DANE PERSON.

Rys. 16.

Oznaczenia:

WDSz - wyższe dowódczo-sztabowe
WT - wyższe techniczne
BJR - biegła znajomość języka rosyjskiego
BJA - biegła znajomość języka angielskiego
BJN - biegła znajomość języka niemieckiego
SJN - słaba znajomość języka niemieckiego

Jak widać ze schematu, warunkiem zakończenia pracy cyklicznej jest odczytanie rekordu zawierającego znacznik końca zbioru,

W wielu zadaniach występują takie problemy, których rozwiązanie wymaga zastosowania w programowaniu cykli złożonych - dwu- lub więcej niż dwustopniowych. W takim przypadku każdy cykl posiada niezależną organizację. Zasadę organizacji cyklu dwustopniowego ilustruje schemat blokowy przedstawiony na rys. 17.

Przykład /cykl dwustopniowy ze znanymi liczbami powtórzeń/

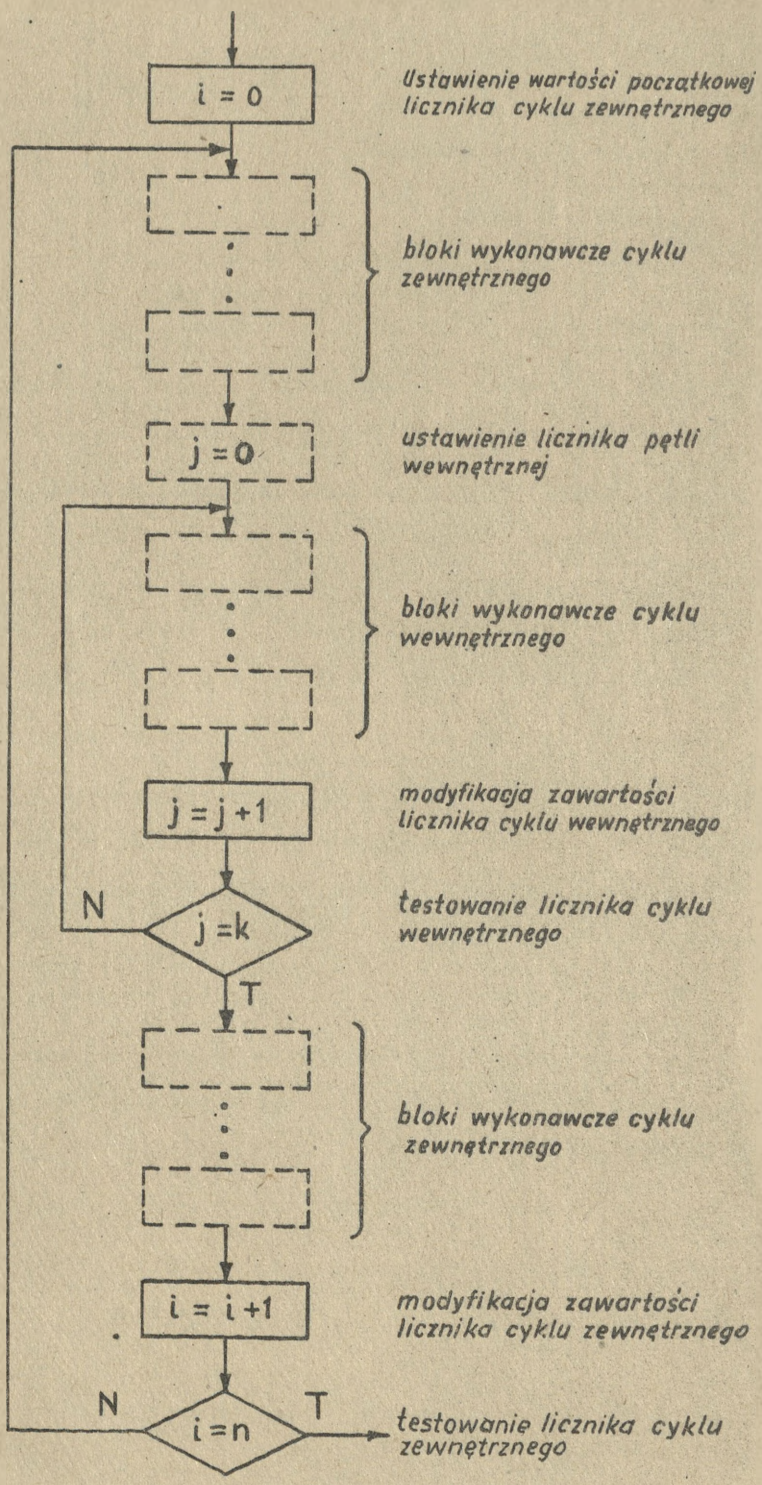
Dany jest ciąg złożony z n liczb. Narysować schemat blokowy programu zliczającego sumę dziesiątych potęg tych liczb /rys.18/. Przykład jest na tyle prosty, a schemat komunikatywny, że nie wymagane są bliższe wyjaśnienia.

Przykład /cykl trzystopniowy ze znanymi liczbami powtórzeń/

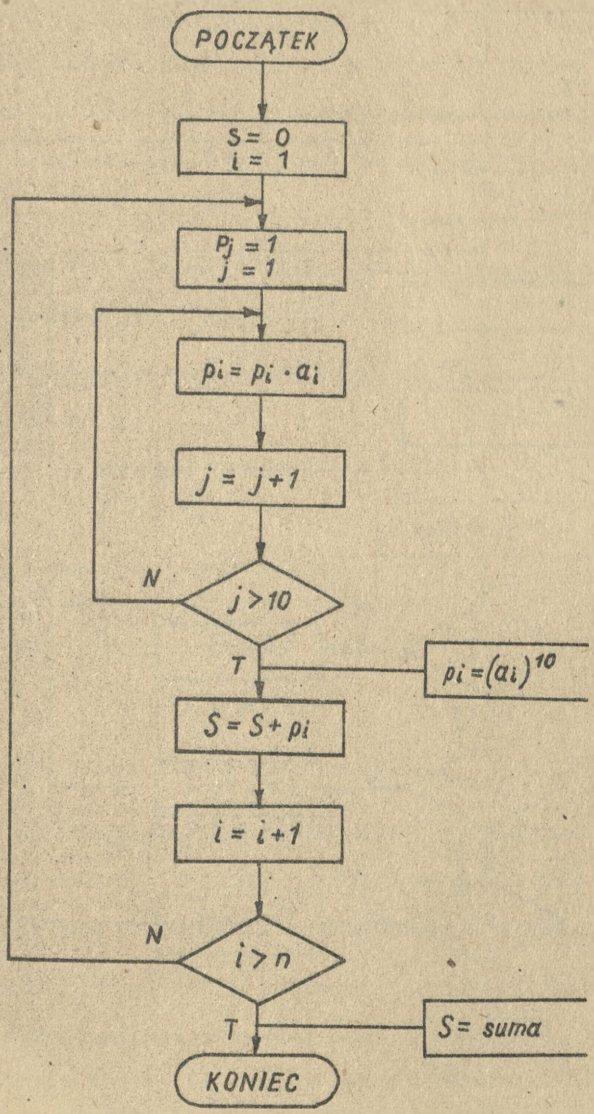
Narysować schemat blokowy programu na pomnożenie macierzy $A = \{a_{i,j}\}$ o elementach rzeczywistych przez macierz $B = \{b_{j,i}\}$ o elementach rzeczywistych, gdzie:

$$i = 1, \ldots, m; \qquad j = 1, \ldots, n$$

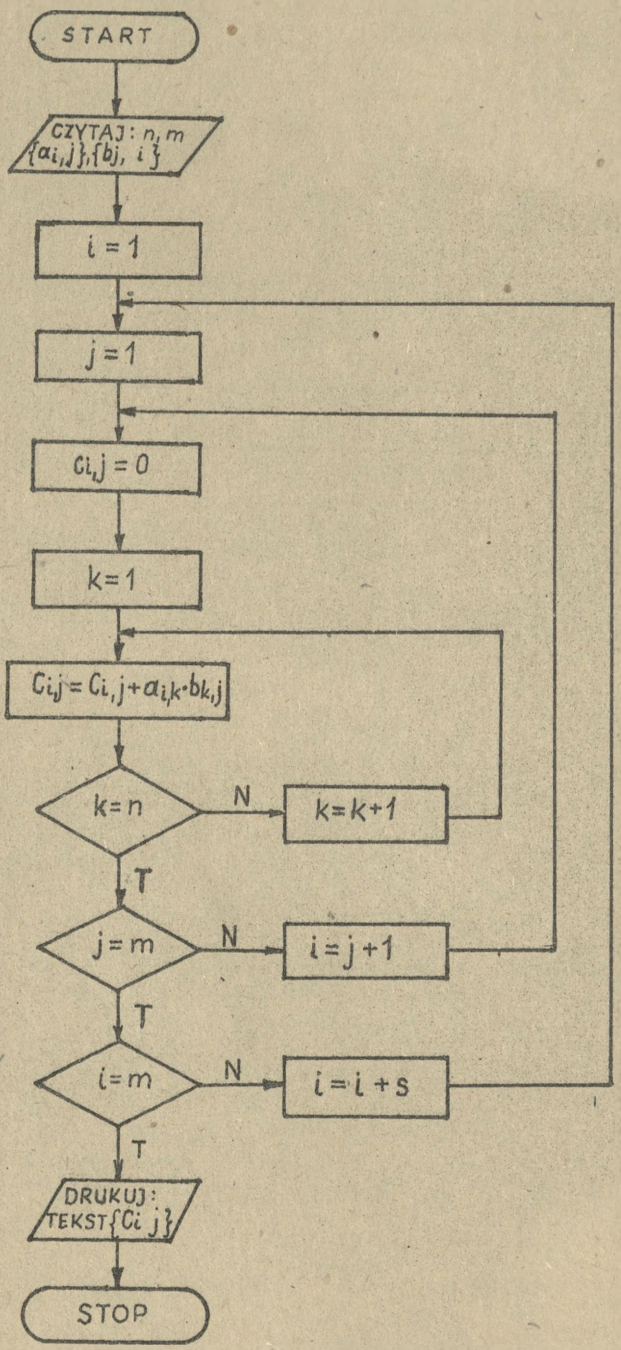
Wartość n, m oraz elementy macierzy wczytać z karty perforowanej. Macierz wynikową $C = \{c_{i,j}\}$ wyprowadzić w postaci wydruku /rys. 19/.


i = 0
Ustawienie wartości początkowej licznika cyklu zewnętrznego
bloki wykonawcze cyklu zewnętrznego
j = 0
ustawienie licznika pętli wewnętrznej
bloki wykonawcze cyklu wewnętrznego
j = j + 1
modyfikacja zawartości licznika cyklu wewnętrznego
N
j = k
testowanie licznika cyklu wewnętrznego
T
bloki wykonawcze cyklu zewnętrznego
i = i + 1
modyfikacja zawartości licznika cyklu zewnętrznego
N
T
i = n
testowanie licznika cyklu zewnętrznego

Rys. 17


POCZĄTEK
S= 0
i = 1
Pj = 1
j = 1
pi = pi · ai
j = j+1
N
j > 10
T
pi = (ai)10
S = S + pi
i = i+1
N
i > n
T
S = suma
KONIEC

Rys. 18.


START
CZYTAJ: n, m
{ai,j}, {bj, i}
i = 1
j = 1
Ci,j = 0
k = 1
Ci,j = Ci,j + ai,k · bk,j
k = n
N
k = k + 1
T
j = m
N
i = j + 1
T
i = m
N
i = i + s
T
DRUKUJ:
TEKST {Ci,j}
STOP

Rys. 19.

Przykład /cykl dwustopniowy z nieznaną liczbą cykli - oba cykle iteracyjne/

Obliczyć przybliżoną wartość pierwiastka kwadratowego dla liczb rzeczywistych. Dane wejściowe zawarte są w zbiorze kart perforowanych, zakończonym kartą z napisem KZ. Każda karta zawiera trzy dane - liczbę pierwiastkowaną, wstępne oszacowanie jej pierwiastka oraz dokładność. Wyniki należy wyprowadzić w postaci wydruku.

Do rozwiązania zadania posłużymy się najpowszechniej stosowaną metodą iteracyjną Newtona-Raphsona.

Aby obliczyć pierwiastek kwadratowy tą metodą, posługujemy się wielokrotnie wzorem:

$$Y_{i+1} = \frac{1}{2}\left(\frac{a}{y_i} + y_i\right)$$

gdzie: a - liczba pierwiastkowana,
y_i - i-te przybliżenia $\sqrt{a}$,

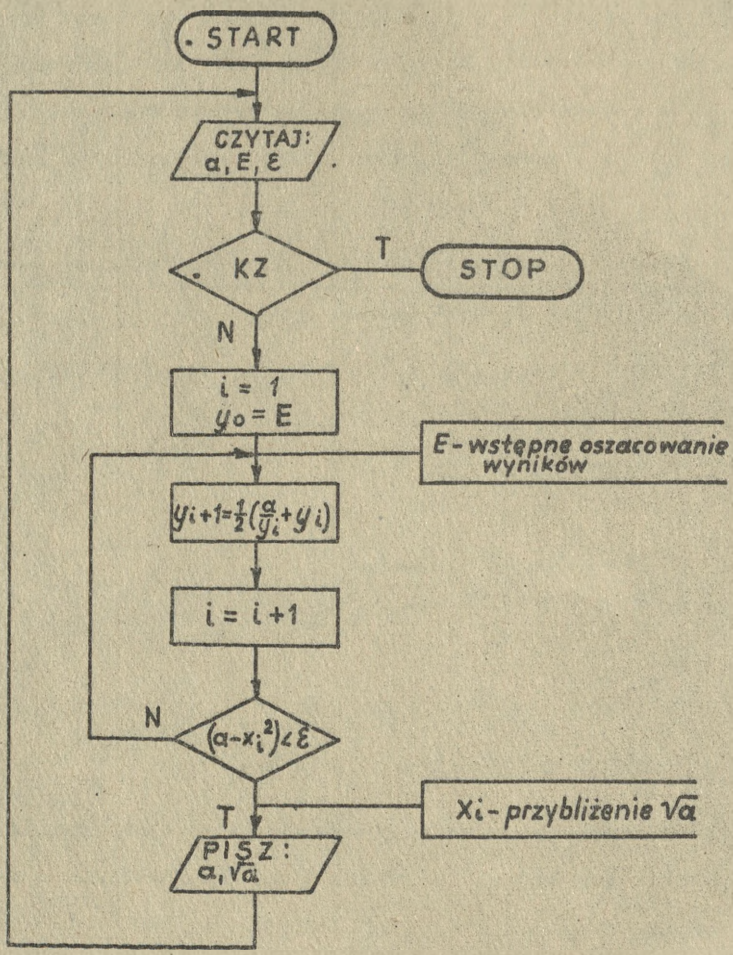
oraz warunkiem

$$|a - y_i^2| < \varepsilon$$

gdzie: ε - dowolnie mała liczba dodatnia.

Zatem do rozwiązania zadania wystarczy przyjąć dowolną wartość początkową J_o oraz dostatecznie małą wartość dodatnią liczby , zależnie od zakładanej dokładności wyniku.

Jak widać z rys. 20, w przytoczonym przykładzie występuje licznik w cyklu wewnętrznym. Rola tego licznika nie polega jednak na zliczaniu wykonanej liczby cykli, lecz na generowaniu indeksu kolejnego przybliżenia wyniku. Ilość powtórzeń w cyklu wewnętrznym zależy od tego, jakie zostaną przyjęte: oszacowanie wstępne oraz dokładność wyniku .


START
CZYTAJ:
a, E, ε
KZ
T
STOP
N
i = 1
y₀ = E
E - wstępne oszacowanie wyników
yᵢ₊₁ = ½(a/yᵢ + yᵢ)
i = i + 1
N
(a - xᵢ²) < ε
T
Xi - przybliżenie √a
PISZ:
a, √a

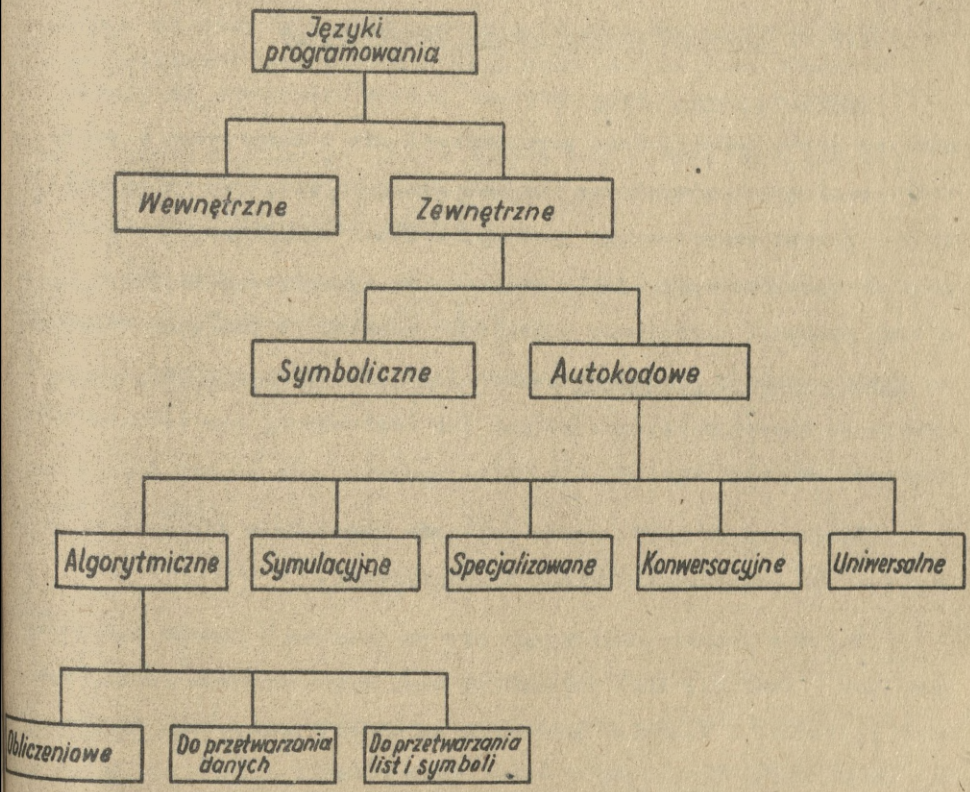
Rys. 20

2.6. Napisanie programu rozwiązania zadania

Schemat blokowy stanowi dla programisty podstawę do pisania programu rozwiązania zadania w języku zrozumiałym dla komputera, czyli w odpowiednim języku programowania.

2.6.1. Klasyfikacja języków programowania

Do tej pory pojawiła się bardzo duża ilość różnorodnych języków programowania, dlatego też warto zapoznać się z ich klasyfikacją /rys. 21/ oraz charakterystyką tych spośród nich, które zyskały dużą rangę w skali światowej i są powszechnie stosowane.



Rys. 21.

Języki programowania charakteryzują się pewnymi ustalonymi regułami gramatycznymi i posiadają swój własny słownik. Niektóre z nich odbiegają bardzo od języka naturalnego, inne zbliżone są do języka matematycznego, a jeszcze inne korzystają ze zwrotów i wyrażeń stosowanych powszechnie w języku naturalnym. W zdecydowanej większości języki programowania opracowane zostały w mutacji angielskiej i w takiej są rozpowszechnione także poza sferą tego języka, w tym również w Polsce.

Rozwój języków programowania zdążał i nadal zdąża do coraz większego ułatwienia programowania zadań i całkowitego uniezależnienia pracy programisty od struktury komputerów.

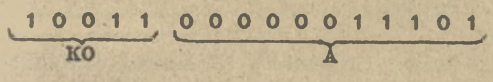
2.6.2. Charakterystyka języka wewnętrznego

Językiem wewnętrznym nazywamy system kodowania, za pomocą którego można bezpośrednio porozumiewać się z komputerem i opisywać zachodzący w nim proces automatycznego przetwarzania danych. Twórcą języka wewnętrznego jest konstruktor komputera, który ustala wykaz operacji, jakie komputer ma wykonywać, i każdej z tych operacji przypisuje odpowiedni rozkaz. W ten sposób powstaje lista rozkazów komputera zawierająca nie tylko rozkazy do wykonywania operacji arytmetycznych lub logicznych, ale również operacji organizacyjnych.

Zgodnie z przyjętą listą rozkazów konstruuje się do ich wykonywania odpowiednie układy elektroniczne.

Zarówno rozkazy, składające się na program w języku wewnętrznym, jak i wszelkie inne informacje wewnątrz komputera /dane, wyniki pośrednie i końcowe/ posiadają strukturę numeryczną - binarną, czyli zero-jedynkową.

Pojedynczy rozkaz, w przypadku np. słowa złożonego z 16-tu bitów, może mieć następującą postać:



KO - kod operacji
A - adres argumentu

Rozkaz taki może np. oznaczać: "Do zawartości akumulatora dodaj zawartość komórki PAO o adresie 29.

Języki wewnętrzne charakteryzują następujące właściwości:

- każdy komputer lub rodzina komputerów posiada swój własny język wewnętrzny;
- język wewnętrzny zrozumiały jest dla układów elektronicznych, a nie jest przystosowany do potrzeb człowieka;
- pisanie programu w języku wewnętrznym jest bardzo uciążliwe i pracochłonne oraz narażone na popełnianie błędów, trudnych do wykrycia;
- programista piszący program w języku wewnętrznym musi rezerwować dla niego jak również dla danych, wyników pośrednich i końcowych, odpowiedni obszar PAO;
- program napisany w języku wewnętrznym, po wprowadzeniu go do PAO, może być natychmiast wykonywany;
- języki wewnętrzne zapewniają bardzo ekonomiczne wykorzystanie PAO;
- za pomocą języka wewnętrznego można programować zadania z dowolnych dziedzin.

Trzy ostatnie właściwości stanowią zaletę języków wewnętrznych.

W językach wewnętrznych programowano zadania dla pierwszych komputerów, jakie pojawiły się. Z kolei, aby ułatwić zapamiętywanie kodów operacji i adresów oraz nadać programom bardziej przejrzystą postać, zarówno kody operacji jak i adresy wyrażono w sys-

temie ósemkowym. Tłumaczenia programu z systemu ósemkowego na dwójkowy dokonywał automatycznie komputer, co w realizacji technicznej jest procesem bardzo prostym.

Przytoczony uprzednio rozkaz w systemie binarnym przyjąłby w systemie ósemkowym postać następującą:

$$\underbrace{2\ 3}_{\text{KO}}\ \underbrace{0\ 0\ 3\ 5}_{\text{A}}$$

Przykład /program w języku wewnętrznym komputera ODRA-1003 na uporządkowanie 8 liczb metodą przestawiania/

```
:401   10000:
72.
-1.
12.
-24.
0.
-53.
-24.
3.
:032   00007   10011   20
:032   00006     +     30
:032   00000     +     10
:546   10014   10026   20
:137   10000     +     20
:437   10001     3     11
:146   10025     +     00
:137   10000     +     11
:247   00000     +     70
:137   10001     +     11
:207   10000     +     11
:177   00000     +     70
:207   10001     +     11
:646   10013   10013   10
:060   00000     +     30
:402   00000     +     20
:546   10012     +     30
:032   00007     +     40
:032   00000     +     10
:137   10000     +     11
:746   01010     +     00
:746   01002     +     00
:646   10037     +     10
:546   10033     +     40
:726   00000   10010   00
:726   00000:
```

Rys. 22.

Obecnie język wewnętrzny stosuje się nadal do pisania programów specjalnych, zwanych translatorami, które wchodzą w skład oprogramowania wszystkich komputerów.

2.6.3. Charakterystyka języków zewnętrznych

Językiem zewnętrznym nazywamy każdy język programowania różny od języka wewnętrznego. Języki zewnętrzne mają charakter pośredni między językami wewnętrznym a matematycznym czy naturalnym. Ich cechą charakterystyczną jest to, że programy napisane w tych językach wymagają translacji, czyli przetłumaczenia na język wewnętrzny. Translacja realizowana jest przez komputer w sposób automatyczny za pomocą specjalnego programu, nazywanego translatorem.

Program w języku zewnętrznym nazywa się programem źródłowym, natomiast program przetłumaczony na język wewnętrzny nosi nazwę programu wynikowego.

Jak widać ze schematu klasyfikacji języków programowania /rys. 21/, wśród języków zewnętrznych wyróżnia się języki symboliczne oraz autokodowe.

2.6.3.1. Charakterystyka języków symbolicznych

Języki symboliczne charakteryzują się tym, że przy pisaniu rozkazów zarówno kodom operacji jak i adresom nadaje się nazwy mnemotechniczne ułatwiające programiście ich zapamiętywanie i nadające programom bardziej przejrzystą postać.

I tak, dla kodów operacji język symboliczny przewiduje symbole będące skrótami nazw tych operacji, np.:

ADD	-	dodaj	
MUL	-	mnóż	/skrót od MULTIPLY/
DIV	-	dziel	/skrót od DIVIDE/
STO	-	zapamiętaj	/skrót od STORE/

Części adresowej nazwę nadaje programista, czyniąc to na ogół w taki sposób, aby ta nazwa symbolizowała zawartość danej komórki PAO. Jeśli np. w komórce PAO zawarta jest wartość zmiennej X, to komórce tej nadaje się również nazwę X. Postępując analogicznie można nadawać części adresowej np. takie nazwy jak: CIEZAR, ILOŚC, WIELKOSC, DELTA itp.

Dla języków symbolicznych charakterystyczne są następujące właściwości:

- języki symboliczne są ukierunkowane maszynowo /zorientowane na maszynę/, tzn. że dany komputer bądź rodzina komputerów ma własny język symboliczny i programy napisane w tym języku nie mogą być uruchamiane na komputerach innych typów;
- program napisany w języku symbolicznym wymaga przetłumaczenia na język wewnętrzny komputera; tłumaczenie to odbywa się za pomocą translatora zwanego assemblerem;
- program w języku symbolicznym jest dłuższy od programu w języku wewnętrznym, ponieważ zawiera dodatkowo deklaracje określające obszary PAO wydzielone dla programu /obszar roboczy, obszar dla stałych itp./ oraz instrukcje sterujące, które kierują tłumaczeniem programu z języka symbolicznego na wewnętrzny;
- język symboliczny umożliwia programowanie zadań z dowolnych dziedzin.

Przykładem języków symbolicznych stosowanych w Polsce są np.:

PLAN - dla komputerów serii ODRA-1300 oraz ICL-1900;

SPS-1 i SPS-2 - dla komputera IBM-1401;

PIES - dla komputera ZAM-41;

ASSK - dla minikomputera K-202.

Programowanie w języku symbolicznym jest znacznie prostsze niż w języku wewnętrznym, tym niemniej jest także procesem dość zawiłym, jak to widać z przytoczonego przykładu:

Przykład /program w języku symbolicznym PLAN na wyszukiwanie numerów pozycji liczb równych liczbie zadanej Z/.

```
# STEER              LIST, OBJEKT, CONSOLIDATE
# PROGRAM            SZUK
# PERIPHERAL         TRO, LPO
# LOWER
                     L, N, Z, DRUK (2)
# LOWER
  KON1               0/4, 0, 8, 0/1
  KON2               0/4, 0, 4, 0, 1
  KON3               2/0, 0, 5, 0/DRUK.3
 # PROGRAM
 # ENTRY             0
       LDN    0      #41
       STO    0      DRUK
       STOZ          L
       PERI   0      KON1
       LDN    6      0
       LDN    7      0
       LSX    2      '4/N.0'
       CDB    6      0(2)
       BCHX   2      *-1
       STO    7      N
 S2    PERI          KON2
       LDN    0      1
       ADS    0      L
       SBX    1      Z
       BNZ    1      S1
       LDX    7      L
       LDN    6      0
       LDX    2      '4/DRUK + 1.0'
       CBD    6      0(2)
       BCHX   2      *-1
       PERI   0      KON3
 S1    LDX    0      N
       SBX    0      1
       BNZ    0      S2
# END
# FINISH
  ****
```

Rys. 23.

Języki symboliczne nadal stosowane są do takich celów, jak:

- pisanie translatorów języków wyższego rzędu;

- pisanie programów dla komputerów o ograniczonej pojemności PAO, nie wystarczającej do zapisu translatora wyższego rzędu;
- zapewnienie efektywnego wykorzystania właściwości konstrukcyjnych komputerów, czego nie da się osiągnąć za pomocą języków wyższego rzędu.

2.6.3.2. Charakterystyka języków autokodowych

Języki autokodowe[x], zwane też proceduralnymi lub językami wyższego rzędu, są uniwersalne w tym sensie, że programy napisane w tych językach mogą być realizowane przez dowolny komputer. Przystosowanie języków autokodowych do komputerów różnych typów osiąga się za pomocą odpowiednich translatorów. Aczkolwiek za pomocą dowolnego języka autokodowego można by zaprogramować dowolne zadanie, to jednak różne języki autokodowe preferowane są do różnych zastosowań i w tym sensie są one językami specjalistycznymi /zorientowanymi na problem/.

Programy napisane w języku autokodowym wymagają translacji na język wewnętrzny.

Rozkazy pisane w językach autokodowych nazywane są instrukcjami. Poszczególne instrukcje w trakcie translacji tłumaczone są na odpowiadające im rozkazy lub grupy rozkazów języka wewnętrznego.

Wśród języków autokodowych wyróżnia się języki: algorytmiczne, symulacyjne, specjalizowane, konwersacyjne oraz uniwersalne /rys. 21/.

Języki algorytmiczne wywodzą swą nazwę stąd, że ich struktura jest bardzo zbliżona do struktury algorytmów. Do tej grupy zaliczamy języki obliczeniowe, do przetwarzania danych oraz do przetwarzania symboli.

---

x/ Określenie "języki autokodowe" wywodzi się stąd, że programy napisane w tych językach mogą być przez komputery automatycznie tłumaczone /kodowane/ na język wewnętrzny.

Języki obliczeniowe, zwane też językami zorientowanymi matematycznie lub naukowo, są w swej notacji bardzo zbliżone do symboliki matematycznej. Najbardziej typowymi i rozpowszechnionymi przedstawicielami tej grupy są języki: ALGOL, FORTRAN, PASCAL.

Języki do przetwarzania danych, zwane też językami ekonomicznymi bądź językami dla potrzeb zarządzania, są zewnętrznie podobne do języka naturalnego. Spośród istniejących języków do przetwarzania danych najbardziej rozpowszechniony jest język COBOL, natomiast mniej - język RPG.

Języki do przetwarzania list i symboli charakteryzują się tym, że ich podstawowymi elementami danych nie są liczby, lecz ciągi znaków alfanumerycznych. Języki te umożliwiają łatwe dokonywania na takich ciągach operacji tego rodzaju, jak scalanie, rozdzielanie, wyszukiwanie itp. Znajdują one zastosowanie do pisania translatorów, automatycznego tłumaczenia jednego języka na inny, rozpoznawania obrazów, symulacji ludzkiego poznania itp. Do przedstawicieli tej grupy należą takie języki, jak: EOL, LISP, SNOBOL.

Języki symulacyjne, zwane też językami modelowania, przeznaczone są do budowania cyfrowych modeli układów technicznych, biologicznych i innych oraz do przeprowadzania z tymi modelami eksperymentów metodą symulacji ich działania. Do tej grupy zaliczane są języki: CSL, GPSS, SIMULA, SIMSCRIPT.

Języki specjalizowane przeznaczone są do programowania zadań specjalizowanych, jak np. pisanie kompilatorów, obliczenia konstrukcyjne, sterowanie procesami technologicznymi itp. W skład tej grupy wchodzą języki: STRESS, NET /obliczenia konstrukcyjne/, PROFILDATA /sterowanie obrabiarek/ i inne.

Języki konwersacyjne umożliwiają prowadzenie bezpośredniego dialogu w relacji człowiek-komputer za pośrednictwem odpowiednich urządzeń zewnętrznych, takich jak: dalekopis, elektryczna maszyna do pisania, monitor ekranowy lub grafoskop. Genezą takich języków była idea wykorzystania komputerów do rozwiązywania krótkich , niepowtarzalnych zadań, których uruchomienie w trybie tradycyjnym byłoby nieekonomiczne, niewygodne i zbyt czasochłonne. Najbardziej rozpowszechnione są dwa języki konwersacyjne: BASIC oraz JEAN.

Języki uniwersalne charakteryzują się tym, iż nadają się do programowania zadań z różnych dziedzin. Twórcy wielu języków programowania reklamowali je jako uniwersalne. W rzeczywistości jednak były to zapowiedzi bez pokrycia. Spośród współczesnych języków programowania jedynie język PL/1 spełnia warunki języka uniwersalnego. Łączy on bowiem właściwości i możliwości takich podstawowych języków jak: ALGOL, FORTRAN i COBOL.

2.6.3.3. Charakterystyka języka ALGOL 60

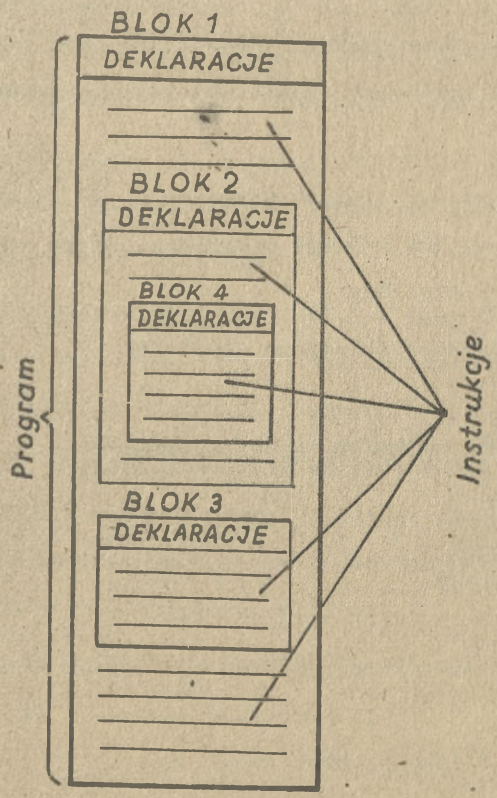
Język ALGOL /ALGOrithmic Language/ jest twprem specjalistów amerykańskich i zachodnioeuropejskich. Liczba 60 oznacza rok /1960/ opublikowania tej wersji języka.

Język ALGOL 60 przeznaczony jest do obliczeń numerycznych /naukowo-technicznych/. Stąd też jego notacja jest zbliżona do notacji matematycznej. Programy pisane w języku ALGOL mogą być uruchamiane na dowolnym komputerze, wyposażonym w translator tego języka.

Obecnie w świecie pisze się tylko kilka procent programów w języku ALGOL 60. Jednak posiada on dodatkowe znaczenie z tego względu, że - jako język teoretycznie dopracowany i ściśle zdefiniowany - wykorzystywany jest w czasopismach fachowych z zakresu

metod numerycznych i programowania do publikacji algorytmów rozwiązywania typowych zadań matematycznych. Na bazie języka ALGOL 60 powstały inne języki programowania, a wśród nich ALGOL 68 oraz PASCAL.

Język ALGOL 60 posiada strukturę blokową, przy czym struktura ta jest hierarchiczna /rys. 24/. Jak widać z rysunku, przykładowy program składa się z czterech bloków, przy czym blok 1 obejmuje cały program i zawiera bloki 2 i 3, które znajdują się na jednym poziomie hierarchii. Blok 4 zawarty jest w bloku 2 i w stosunku do bloków 2 i 3 reprezentuje niższy poziom hierarchii.



Rys. 24.

Każdy blok stanowi autonomiczną jednostkę programu, która jest w dużym stopniu niezależna od pozostałych bloków. W skład każdego bloku wchodzą deklaracje oraz instrukcje.

Deklaracje, umieszczone na początku bloku, mają znaczenie lokalne i zawierają informacje dla kompilatora, niezbędne do przetłumaczenia programu z języka FORTRAN na język wewnętrzny oraz rozmieszczenia go w PAO. Są to informacje dotyczące nazw zmiennych użytych w bloku, wielkości PAO na przechowanie tych zmiennych itp. Przy tym, w danym bloku można użyć tylko tych wielkości, które zostały zadeklarowane w tymże bloku lub w odpowiadającym mu bloku o wyższej hierarchii.

Instrukcje, jako odpowiedniki rozkazów lub grup rozkazów, stanowią dla komputera wskazówki, jakie operacje ma wykonywać i na jakich danych.

Rys. 25 przedstawia tabulogram programu w języku ALGOL-60 dla tego samego zadania, dla którego program w języku symbolicznym PLAN pokazany został na rys. 23.

Przykład /program w języku autokodowym ALGOL 60/

```
'begin'
   'Comment' SZUK - program szukający numerów
        pozycji liczb równych liczbie zadanej Z;
   'integer' L,N;
   'real' A, Z;
   select input /3/;
   select output /2/;
   N:= read;
   Z:= read;
   'for' L:= 1 'step' '1' until 'N' do'
   'begin'
       A:= read;
       'if' A = Z 'then' print /L, 4, 0/;
       newline /1/;
   'end'
'end'
```

Rys. 25.

Jak widać, program tego samego zadania w języku ALGOL 60 jest znacznie krótszy, prostszy i bardziej komunikatywny w porównaniu z programem napisanym w języku PLAN.

2.6.3.4. Charakterystyka języka FORTRAN

Język FORTRAN /FORmula TRANslator/ opracowany został przez amerykańską firmę IBM. Jest to język doskonale przystosowany do obliczeń numerycznych, o notacji bardzo zbliżonej do matematycznej, łatwy do przyswojenia bez żadnego przygotowania wstępnego i dlatego też, jako język autokodowy, najbardziej rozpowszechniony w świecie. W przeciwieństwie do języka ALGOL 60, nie powstał na bazie teoretycznych opracowań, lecz praktycznych doświadczeń. Z tego względu nie jest on ściśle zdefiniowany, co umożliwia powstawanie coraz to doskonalszych jego wersji.

Programy napisane w języku FORTRAN mogą być uruchamiane na dowolnym komputerze, w którego oprogramowaniu znajduje się translator tego języka.

Język FORTRAN posiada strukturę segmentową, przy czym w programie napisanym w tym języku mogą wystąpić cztery rodzaje segmentów: segment główny /MASTER/, segment podprogramu /SUBROUTINE/, segment funkcji /FUNCTION/ oraz segment bloku danych /BLOCK DATA/.

Rys. 26 ilustruje przykład tabulogramu programu w języku FORTRAN dla rozpatrywanego już uprzednio zadania. Program ten zbudowany jest tylko z segmentu głównego.

Język FORTRAN stał się bazą do opracowania języka symulacyjnego CSL oraz konwersacyjnego BASIC.

W dalszej części niniejszego skryptu podane są zasady programowania w języku FORTRAN. Język ten obecnie w Polsce jest stosowany najpowszechniej, w tym również szeroko do programowania zadań operacyjno-taktycznych.

```
      PROGRAM (SZUK)
      INPUT  1 = CRØ
      OUTPUT 2 = LPØ
      END
C SZUK - PROGRAM SZUKAJĄCY NUMERÓW POZYCJI
C LICZB RÓWNYCH LICZBIE ZADANEJ
      MASTER SZUK
      READ (1,1) N,Z
    1 FORMAT (I1Ø, E1Ø.4)
      DO 2L = 1,N
      READ (1,3) A
    3 FORMAT (E1Ø.4)
      IF (A.EQ.Z) WRITE (2,4) L
    4 FORMAT (5X, I1Ø)
    2 CONTINUE
      STOP
      END
      FINISH
```

Rys. 26.

2.6.3.5. Charakterystyka języka COBOL

Język COBOL /COmmon Business Oriented Language/ opracowany został na zamówienie rządu USA. Stosowany jest on na całym świecie w automatyzacji zarządzania do programowania zadań w dziedzinie handlu, gospodarki materiałowej, zapasów, płac itp.

Język COBOL charakteryzuje się rozbudowanymi mechanizmami manipulowania dużymi zbiorami danych, bogatym asortymentem różnorodnych instrukcji edytorskich oraz notacją zbliżoną do języka naturalnego.

Podobnie, jak w przypadku uprzednio scharakteryzowanych języków, programy napisane w języku COBOL można uruchomić na dowolnym komputerze, w którego oprogramowaniu znajduje się translator tego języka.

Na strukturę programu w języku COBOL składają się cztery podstawowe części, zwane rozdziałami, a mianowicie: rozdział identyfikacji, rozdział struktury, rozdział danych oraz rozdział procedury.

Rozdział identyfikacji /IDENTIFICATION SECTION/ określa dane dotyczące programu, takie jak: nazwę programu, jego numer i wersję, datę realizacji, nazwisko autora itp.

Rozdział struktury /ENVIRONMENT SECTION/ przeznaczony jest na zapis danych dla kompilatora w komputerze, na którym program będzie tłumaczony na język wewnętrzny i realizowany. Są to takie dane jak: typ komputera, ilość i rodzaj urządzeń zewnętrznych, pojemność PAO itp.

Rozdział danych /DATA SECTION/ zawiera opisy zbiorów danych. W rozdziale tym podaje się nazwy zbiorów danych, ich rozmiary, strukturę itp.

Rozdział procedury /PROCEDURE SECTION/ służy do zapisu instrukcji programu, zgodnie z którymi komputer rozwiązuje zadanie.

Jak dalece program napisany w języku COBOL przypomina język naturalny /angielski/ świadczy chociażby przytoczona poniżej instrukcja:

MULTIPLAY VELOCITY BY TIME GIVING DISTANCE;

W instrukcji tej słowa podkreślone są słowami języka COBOL, natomiast słowa pozostałe są nazwami zmiennych nadawanymi przez programistę. Biorąc pod uwagę, że poszczególne słowa w przytoczonej instrukcji mają sens:

MULTIPLAY	-	pomnóż
BY	-	przez
GIVING	-	dając
VELOCITY	-	prędkość
TIME	-	czas
DISTANCE	-	odległość,

widzimy, że instrukcja ta poleca komputerowi:

POMNÓŻ PRĘDKOŚĆ PRZEZ CZAS PODAJĄC ODLEGŁOŚĆ

Ponieważ programista zmiennym może nadawać dowolne nazwy, przestrzegając przy tym tylko odpowiednich reguł, przeto instrukcję tę można sformułować następująco:

MULTIPLAY PREDKOSC BY CZAS GIVING ODLEGŁOSC

Jak łatwo zauważyć, zarówno w językach COBOL, FORTRAN, ALGOL, jak również w innych językach programowania, opracowanych w wersji angielskiej, przy nadawaniu zmiennym polskich nazw nie można używać takich liter alfabetu polskiego, jak: ą, ę, ó, ś, ż itp., lecz należy zastąpić je odpowiednio literami: a,e,o,s,z itp.

Rys. 27 ilustruje część tabulogramu programu w języku COBOL. Zadanie polega na przepisywaniu danych z kart perforowanych na taśmę magnetyczną przy jednoczesnym uzupełnianiu ich dodatkowymi informacjami.

```
IDENTIFICATION DIVISION.
PROGRAM-ID. WYDR 9Ø.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. ICL 19Ø4.
OBJECT-COMPUTER. ICL 19Ø4.
   MEMORY SIZE 4ØØØ WORDS.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
   SELECT KARTY ASSIGN TO CARD-READER Ø.
   SELECT TASMA-P ASSIGN TO PAPER-READER Ø.
   SELECT TABUL ASSIGN PRINTER Ø.
   SELECT TASMA-M ASSIGN TO 1 TAPES.
DATA DIVISION.
FILE SECTION.
FD KARTY
Ø1 R-KARTY.
   Ø2 K-N PIC 9(2).
   Ø2 S-W PIC 9(8).
      .   .
      .   .
      .   .
PAR-12.
   MOVE SPACES TO A
   MOVE "I" TO A1 A2 A3 A4 A5 A6 A7 A8 A9.
   WRITE A
```

Rys. 27.

2.6.3.6. Charakterystyka języka CSL

Język CSL /Control and Simulation Language/ przeznaczony jest do modelowania cyfrowego i badania układów dynamicznych, tak deterministycznych jak i statystycznych. Stąd też jest on szeroko stosowany do programowania zadań z dziedziny dynamiki obiektów, masowej obsługi, gospodarki materiałowej itp.

Język CSL powstał na bazie języka FORTRAN i jest do niego bardzo podobny. Tłumaczenie programu z języka CSL na język wewnętrzny komputera odbywa się w dwóch etapach. Początkowo dokonywany jest przekład na język FORTRAN, co realizuje specjalny program, a dopiero z kolei następuje normalna kompilacja z FORTRAN-u na język wewnętrzny.

Przy programowaniu w języku CSL można stosować wstawki w języku FORTRAN oraz szeroko wykorzystywać bogate podprogramy biblioteczne tego języka.

Ze względu na charakter zadań, do programowania których jest przeznaczony, język CSL ma bogato rozbudowany repertuar operacji na zbiorach i dysponuje generatorem typowych rozkładów prawdopodobieństwa, umożliwiających generację rozkładów zgodnie z danymi empirycznymi.

Fragment tabulogramu programu napisanego w języku CSL, a dotyczącego zliczania czasów przestojów spowodowanych brakiem surowców, przedstawiony został na rys. 28.

```
      MASTER MGZN
      CLASS TIME SUROWIEC. 4(2) SET BRAKI SUROWCA , SUROWCE
      FLOAT PRAWDOPODOBIENSTWO
      ARRAY NORMA (4), ROZKLAD (2,11), PRAWDOPODOBIENSTWO (4)
      ENTRY 37
      START
      I = 1
      READ (1,1) (NORMA (I), I = 1,4)
      CHECK (3) I
C
C     SPRAWDZENIE WARTOSCI ZMIENNEJ
C
    1 FORMAT (4(I3, 2x)
      I = 1
      J = 1
      READ (1,2) ((ROZKLAD (I,J), J = 1, 11))
    2 FORMAT (I3, 2x, I3)
   37 POST MORTEM SPRAWOZDANIE
                .       .
                .       .
                .       .
   84 FORMAT (21H CZAS PRZESTOJU RAZEM, I6)
      EXIT
      END
```

Rys. 28.

2.6.3.7. Charakterystyka języka PL/1

Język PL/I (Programming Language I) opracowany został przez specjalistów amerykańskiej firmy IBM i ma charakter języka uniwersalnego. Potrzeba stworzenia takiego języka wyniknęła z tego faktu, że w podejmowaniu decyzji i rozwiązywaniu złożonych zadań tkwią problemy z różnych dziedzin. Stosownie do programowania takich problemów różnych języków programowania powoduje komplikację tego procesu, wymaga stosowania różnych translatorów i prowadzi do nieefektywnego wykorzystywania komputerów.

Język PL/I charakteryzuje wiele zalet. Mogą z niego korzystać zarówno początkujący jak i zaawansowani programiści. Jest on wygodny w programowaniu problemów naukowo-technicznych i administracyjnych, nadaje się do programowania przetwarzania symboli, sterowania procesami w czasie rzeczywistym, budowy i badania modeli

ymulacyjnych itp. Przy tworzeniu języka PL/I, wzorowano się na ajlepszych cechach takich języków, jak: ALGOL, FORTRAN i COBOL. ąd też w budowie tego języka wykorzystano strukturę blokową LGOL-u, opis struktury hierarchicznej z COBOL-u i opisy formatów FORTRAN-u. Strukturę języka przystosowano również do najnowszych iągnięć konstrukcji, organizacji i użytkowania współczesnych mputerów, takich jak: modułowość, systemy przerywań, wieloprogramowość, wieloprzetwarzanie, wielodostęp, wieloprocesorowość, tonomiczność kanałów, współpraca z pamięciami masowymi itp.

Cenną zaletą języka PL/I jest jego modularna budowa, dzięki órej do programowania zadań z różnych dziedzin można posłużyć ię różnymi podzbiorami języka. Dlatego też programista nie musi ać całego języka PL/I, a tylko pewne jego moduły, przydatne mu programowania zadań z danych dziedzin.

Na rys. 28a podano przykład programu napisanego w języku L/I, a dotyczącego porządkowania zbioru liczb w kolejności enącej.

```
A: PROCEDURE OPTIONS (MAIN)
    DECLARE (I,J,N) FIXET DECIMAL, (Z,A (1ØØ)) FLOAT;
    GET LIST (N,A);
    POR: DO I = 1 TO N-1 BY1;
         DO J = I + 1 TO N BY1;
         IF A(I) > A(J) THEN BEGIN
                  Z = A(I); A(I) = A(J); A(J) = Z;
    END POR;
    PUT LIST (A);
END A;
```

Rys. 28a.

Prace nad dalszym rozwojem i doskonaleniem języka PL/I ą kontynuowane. Można spodziewać się, że w miarę wprowadzania do ksploatacji komputerów Jednolitego Syststemu RIAD, wzorowanych na omputerach firmy IBM, język ten rozpowszechni się również w krach RWPG.

2.6.3.8. Charakterystyka języka BASIC

Język BASIC /Beginner's All-purpose Symbolic Instruction Kode/, opracowany w USA przez profesorów wyższej uczelni technicznej - Dartmonth College na bazie języka FORTRAN, jest obecnie najpopularniejszym i najbardziej uniwersalnym językiem konwersacyjnym. Jest on używany do programowania zadań naukowo-technicznych, organizacyjnych i ekonomicznych, przewidywanych do uruchomienia zarówno na wielkich systemach komputerowych, pracujących z podziałem czasu, jak i na minikomputerach, a nawet mikrokomputerach. Stosowany jest również w systemach sterowania procesami technologicznymi. Z założenia miał to być język przeznaczony dla początkowego nauczania programowania komputerów. I w rzeczywistości jest on bardzo prosty i na tyle łatwy do przyswojenia, że można go opanować po kilku godzinach zajęć.

Najistotniejszą cechą języka BASIC jest to, że umożliwia on prowadzenie dialogu w relacji użytkownik-komputer. Dialog ten może być prowadzony za pośrednictwem dalekopisu lub monitora ekranowego, wyposażonego w klawiaturę. Użytkownik, posługując się klawiaturą urządzenia, przekazuje komputerowi instrukcje, które jednocześnie zapisywane są na taśmie papierowej dalekopisu lub wyświetlane na ekranie monitora. Instrukcje, w trakcie wprowadzania do komputera, podlegają natychmiastowemu tłumaczeniu na język wewnętrzny, co realizowane jest przez specjalny translator zwany interpreterem, i z kolei podlegają wykonaniu. Informacje przekazywane użytkownikowi przez komputer, ściślej interpreter, zapisywane są na taśmie dalekopisu lub na ekranie monitora. Użytkownik, w trakcie prowadzenia dialogu z maszyną, ma możliwość wykrywania i natychmiastowego usuwania błędów w programie.

Niezależnie od konwersacyjnego trybu współpracy z komputerem można wprowadzić z klawiatury do PAO komputera cały program i spowodować jego uruchomienie.

Przykład programu w języku BASIC, dotyczącego sumowania 10 liczb, przedstawiono na rys. 29.

```
1Ø REM PROGRAM SUMUJACY 1Ø LICZB
2Ø REM UMIESZCZONYCH W DEKLARACJI DATA
3Ø S=Ø
4Ø FOR L=1 TO 1Ø
5Ø READ X
6Ø S=S+X
7Ø NEXT L
8Ø PRINT "SUMA LICZB = " S
9Ø DATA 2,5,7,11,15,19,21,32,45,9Ø
9ØØ END
```

Rys. 29.

2.6.3.9. Charakterystyka języka JEAN

Język konwersacyjny JEAN opracowany został przez specjalistów angielskiej firmy ICL. Jest on bardzo prosty, łatwy do opanowania i przeznaczony do rozwiązywania zadań matematycznych oraz logicznych. Charakteryzuje się ponadto notacją bardzo zbliżoną do matematycznej oraz walorami dydaktycznymi, wynikającymi z prostoty jego struktury jak również logicznej spójności. Język JEAN umożliwia współpracę użytkownika z komputerem za pośrednictwem dalekopisu bądź elektrycznej maszyny do pisania. Współpraca ta może przebiegać w trybie natychmiastowym lub w trybie obliczeń programowych. W przypadku pierwszym instrukcja przekazana komputerowi wykonywana jest natychmiast, natomiast w przypadku drugim komputer zapamiętuje ciąg instrukcji, które zostają wykonane na polecenie użytkownika.

Język JEAN opracowany został w trzech wersjach systemowych: w wersji swobodnej, w wersji z systemem operacyjnym MINIMOP oraz w wersji z systemem operacyjnym GEORGE 3, przy czym każda z wymie-

nionych wersji dostępna jest w trzech językach: angielskim, francuskim oraz niemieckim.

Na rys. 30 przytoczony jest przykład programu w języku JEAN na uporządkowanie zbioru liczb w kolejności rosnącej.

```
- 1.1 DEMAND N
- 1.2 DEMAND A(I) FOR I = 1(1) N
- 1.3 I = 1
- 1.4 TO STEP 1.91 IF A(I)  = A(I + 1)
- 1.5 Z = (A(I)
- 1.6 A(I) = A(I + 1)
- 1.7 A(I + 1) = Z
- 1.8 I = I - 1
- 1.9 TO STEP 1.4 IF IO
- 1.91 I = I + 1
- 1.92 TO STEP 1.4 IF I  N
- 1.93 TYPE A(I) FOR I = 1(1) N
- DO PART 1
```

Rys. 30.

2.6.4. Składniki programu

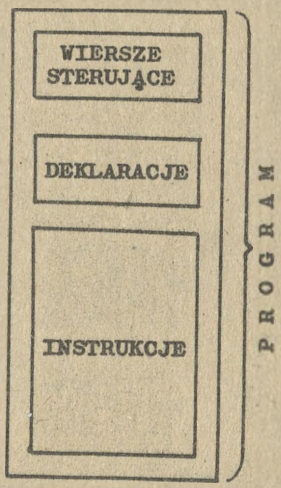
Bez względu na to, w jakim języku zewnętrznym opracowany jest program, zawsze występują w nim dwa podstawowe składniki: definicje obiektów, które w nim występują, zwane deklaracjami, oraz definicje operacji, które wykonywane są na danych, zwane instrukcjami.

Wyróżnia się przy tym następujące rodzaje instrukcji: arytmetyczne i logiczne, warunkowe, wejścia/wyjścia oraz organizacyjne /wywołujące podprogramy, zapewniające przetwarzanie cykliczne, zawieszające bądź zatrzymujące działanie programu itp./.

W wielu językach programowania nieodłączną część programu stanowią tzw. wiersze sterujące, które zawierają informacje dla translatora, niezbędne w procesie tłumaczenia programu źródłowego na wynikowy. Są to informacje określające rodzaj listowania przebiegu translacji /wyprowadzania informacji o tym przebiegu/,

zawierające dane o programie /np. nazwę/, o urządzeniach zewnętrznych, używanych przez program, o poziomie wyszukiwania błędów, o priorytetach programów, uruchamianych w systemie wieloprogramowym /wielodostępnym/ itp.

Ogólna struktura programu zilustrowana została na rys. 31.



Rys. 31.

2.6.5. Formularze programowe

Programiści piszą programy zadań odręcznie na odpowiednich formularzach, przystosowanych do poszczególnych języków programowania. Formularze takie podzielone są na ponumerowane kolejno kolumny i wiersze. Na każdy znak tekstu programu przewidziane jest w wierszu oddzielne miejsce. Przykład formularza do pisania programów podany jest na rys. 32. Przytoczony na nim program napisany został w języku FORTRAN.

FORTRAN

TYTUŁ: Wykr. | LICZBA ARKUSZY: 3 | NR ARKUSZA: 1 | ROZDZIAŁ:

OPRACOWAŁ: SOŁKIEWICZ | SPRAWDZIŁ: SALWOWSKI | DATA: 21.02.1977

ETYKIETA	OPERACJA	ARG	CZĘŚĆ ADRESOWA
	LIST		
	PROGRAM(WYKR)		
	INPUT 1=CRØ		
	OUTPUT 2=LPØ		
	TRACE 2		
	END		
	MASTER PA		
	INTEGER N,I,J,P		
	REAL D(13),V1(1Ø),V2(1Ø),PK,P1(1Ø,13,1Ø),H,PNW		
	READ(1,1ØØ)N,H,PK,PNW		
1Ø	FORMAT(I1Ø,3FØ.Ø)		
	READ(1,11Ø)(V2(I),I=1,1Ø)		
11Ø	FORMAT(15FØ.Ø)		
	READ(1,11Ø)(V1(P),P=1,1Ø)		
	READ(1,11Ø)(D(J),J=1,13)		
	DO 12Ø I=1,1Ø		
	DO 12Ø J=1,13		
	DO 12Ø P=1,1Ø		
12Ø	P1(I,J,P)=(12-D(J))*(1Ø*EXP(1+(V2(I)/V1(P)))**2))*N*PK*PNW)/M		

Rys. 32

Inf-pr 2

FORTRAN

TYTUŁ: WYKR. | LICZBA ARKUSZY: | NR ARKUSZA: 2 | ROZDZIAŁ:

OPRACOWAŁ: SOŁKIEWICZ | SPRAWDZIŁ: SALWOWSKI | DATA: 21.02.1977

ETYKIETA	OPERACJA	AKM	CZĘŚĆ ADRESOWA
	WRITE(2,130)N,H,PK,PNW		
130	FORMAT(1H1,/1/49H PRAWDOPODOBIENSTWO WYKRYCIA OP NA RUBIEŻY ZOP N=,		
1	I3,5H H=,F5.1,6H PK=,F6.4,7H PNW=,F6.4)		
	DO 155 I=1,10		
	WRITE(2,140)V2(I)		
140	FORMAT(4H V2=,F7.1/1/)		
	WRITE(2,150)(V1(A),P=1,10)		
150	FORMAT(4H V1=,10F7.1/1/)		
	DO 155 J=1,13		
155	WRITE(2,160)(P1(I,J,P),P=1,10)		
160	FORMAT(4H ,10F7.1)		
	STOP		
	END		
	FINISH		

20.02.80. WKP/XII-270

2.7. Przeniesienie programu na nośnik maszynowy

Po napisaniu programu na odpowiednim formularzu i sprawdzeniu jego poprawności następuje przeniesienie treści programu na nośnik maszynowy. Nośnik taki stanowią najczęściej karty perforowane lub taśma perforowana. Perforacja programu realizowana jest za pomocą dalekopisu, dziurkarki taśmy papierowej lub dziurkarki kart maszynowych. Czynność tę wykonują operatorzy wymienionych maszyn.

2.8. Uruchomienie programu

Etap uruchomienia programu obejmuje trzy fazy: translację programu źródłowego na wynikowy, jego eksploatację próbną oraz opracowanie dokumentacji programu użytkowego.

2.8.1. Translacja programu

Po przeniesieniu programu na nośnik maszynowy wczytuje się go do PAO komputera i poddaje przetłumaczeniu z języka zewnętrznego na język wewnętrzny, co - jak już mieliśmy okazję stwierdzić - realizowane jest za pomocą specjalnego programu, zwanego translatorem. W zależności od tego, z jakimi językami mamy do czynienia, translatorom nadaje się odpowiednie nazwy.

> Assemblerem nazywamy translator tłumaczący program z języka symbolicznego na wewnętrzny.

Translacja za pomocą assemblera przebiega w stosunku "jeden-jeden" lub "jeden-wiele", co oznacza, że jeden rozkaz w języku symbolicznym tłumaczony jest na jeden rozkaz lub wiele rozkazów w języku wewnętrznym.

Kompilatorem nazywamy translator tłumaczący program z języka wyższego rzędu na język wewnętrzny.

Kompilacja charakteryzuje się tym, że przebiega w stosunku "wiele-jeden" lub "wiele-wiele". Kompilatory wykorzystuje się do tłumaczenia programów napisanych w takich językach, jak: ALGOL, FORTRAN, COBOL, PL/1 itp.

Interpreterem nazywamy translator, który tłumaczy na język wewnętrzny kolejne fragmenty programu, poddawane natychmiastowemu wykonaniu.

Praca interpretera przeplata się więc z wykonywaniem programu. Interpretacji poddawane są np. programy napisane w językach konwersacyjnych.

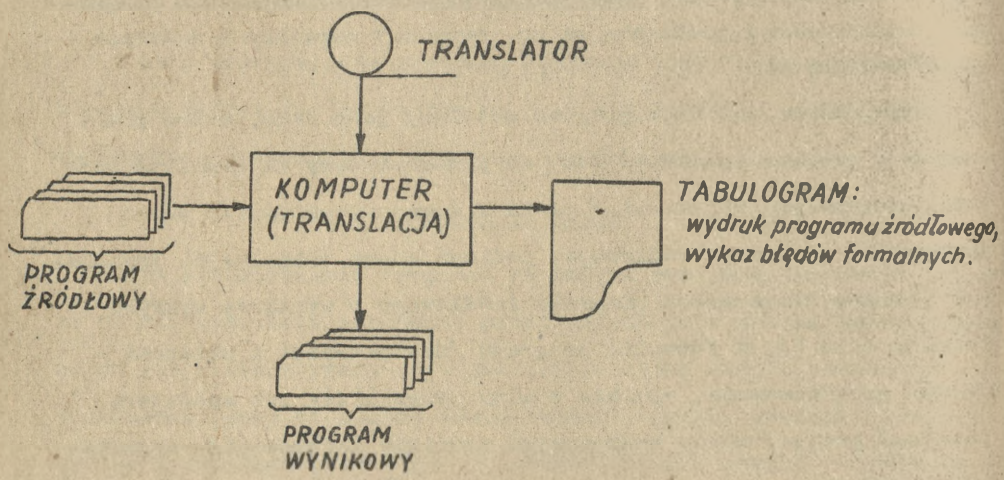
Translator traktuje program źródłowy jako dane, które przetwarza w program wynikowy. Jest on przechowywany na dowolnym nośniku maszynowym, najczęściej na TM i wprowadzamy do PAO przed wczytaniem programu źródłowego, podlegającego translacji. Translator podczas tłumaczenia programu źródłowego w wynikowy wykrywa jednocześnie błędy formalne programu /niezgodności z zasadami języka programowania, wynikłe z winy programisty lub operatora urządzeń przygotowania maszynowych nośników informacji/, sygnalizując ich obecność, charakter i miejsce w programie. Informacje na ten temat wyprowadzane są w postaci tabulogramu, który oprócz tego zawierać może treść programu źródłowego oraz inne dane dotyczące translacji. Jeśli translator wykryje błąd formalny, zostaje zawieszone tłumaczenie programu źródłowego na wynikowy, ale proces translacji przebiega dalej w celu wykrycia dalszych, ewentualnych błędów formalnych.

W przypadku wykrycia błędów formalnych należy wnieść poprawki do programu źródłowego i ponownie dokonać jego translacji. Należy

przy tym zaznaczyć, że translator nie wykrywa błędów logicznych programu.

Poprawny program wynikowy może być realizowany natychmiast przez komputer bądź wyprowadzony na dowolnym nośniku maszynowym do dyspozycji użytkownika.

Translacja programu realizowana jest przez operatora komputera przy ścisłej współpracy z autorem programu. Jej przykładową organizację ilustruje rys. 33.



Rys. 33.

2.8.2. Eksploatacja próbna programu

Eksploatacja próbna programu, prowadzona przez operatora komputera, programistę oraz użytkownika, obejmuje testowanie programu oraz sporządzenie dokumentacji programu.

Testowanie programu polega na sprawdzeniu prawidłowości działania programu na danych próbnych. Jest to - szczególnie w przypadkach bardziej złożonych i skomplikowanych programów - bardzo ważny, żmudny i pracochłonny etap. Zasadniczym przedsięwzięciem testowania jest wykrycie błędów wykonania programu, na które składają się błędy w programie oraz błędy tkwiące w danych. We współczesnych komputerach proces ten jest w znacznym stopniu zautomatyzowany. W tym celu stosuje się specjalne programy pomocnicze, a mianowicie - programy śledzące oraz programy monitujące.

Programy śledzące są to programy, które wykrywają błędy wykonania, natomiast programy monitujące powodują wyprowadzanie odpowiednich komunikatów o wykrytych błędach, ich charakterze oraz lokalizacji w programie.

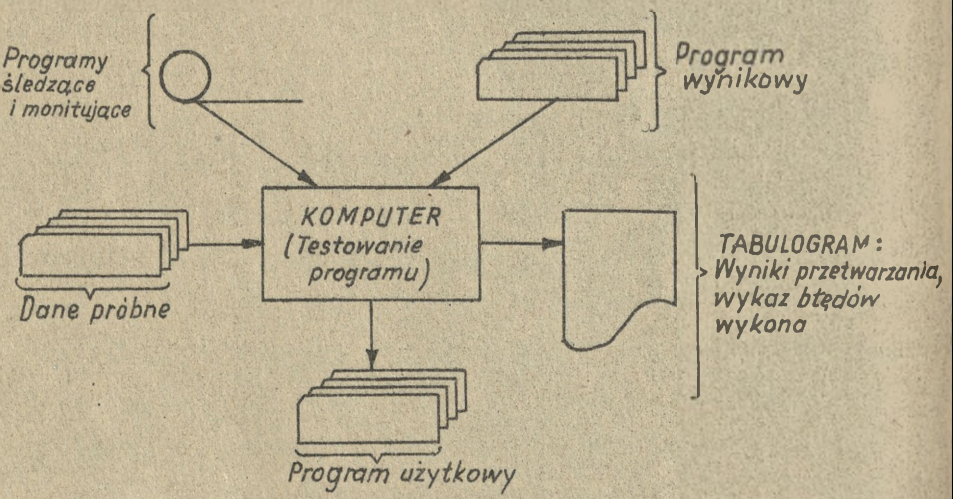
W zależności od żądania programisty, co uwarunkowane jest złożonością uruchamianych programów, programy śledzące mogą realizować niższy lub wyższy poziom śledzenia, z czym wiąże się automatycznie mniejszy lub większy czas przebiegu programu oraz zajęcie w tym celu mniejszego lub większego obszaru PAO. Programy śledzące stwarzają możliwość wydrukowania zawartości wszystkich - istotnych dla testowanego programu - rejestrów w miarę wykonywania kolejnych rozkazów. Dzięki temu programista może prześledzić krok po kroku działanie testowanego programu, co pozwala mu na dokonanie oceny tego przebiegu oraz wyjaśnienie przyczyn występujących błędów. Bardziej uproszczone wersje programów testujących pozwalają śledzić jedynie rozkazy skokowe, co w wielu przypadkach jest dla programisty wystarczające do wyciągnięcia właściwych wniosków dotyczących stanu testowanego programu.

Sprawdzanie programów wynikowych poprzez testowanie, a szczególnie programów zadań złożonych i skomplikowanych, powinno być

wszechstronne i przebiegać zgodnie z zasadą, że najpierw sprawdza się normalne sytuacje, następnie sytuacje rzadziej spotykane i wreszcie sytuacje wyjątkowe, związane np. z niekompetentnymi lub niepoprawnymi danymi. Wskaźnikami błędów w programie wynikowym lub innych przyczyn nieprawidłowego przetwarzania, oprócz informacji wyprowadzonych przez programy monitujące, mogą być niepoprawne wyniki zadań lub zatrzymania komputera z podaniem ich przyczyn.

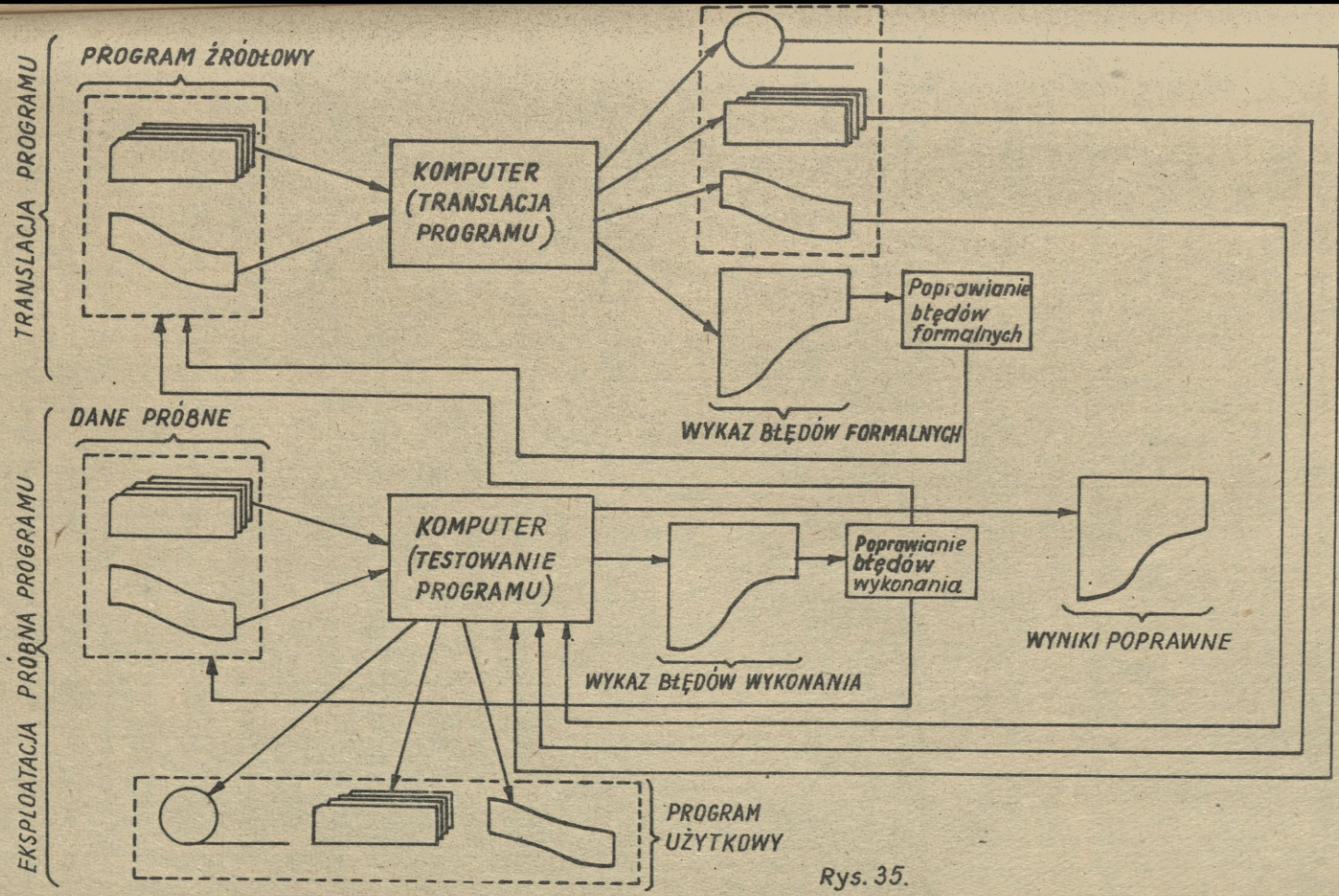
Program wynikowy, po usunięciu zeń błędów wykonania, staje się programem użytkowym.

Przykładową organizację eksploatacji próbnej programu przedstawia rys. 34.



Rys. 34.

Dokładniejszą ilustrację przebiegu uruchamiania programu przedstawia rys. 35. Uwzględniono na nim różne rodzaje maszynowych


TRANSLACJA PROGRAMU
PROGRAM ŹRÓDŁOWY
KOMPUTER (TRANSLACJA PROGRAMU)
Poprawianie błędów formalnych
WYKAZ BŁĘDÓW FORMALNYCH
EKSPLOATACJA PRÓBNA PROGRAMU
DANE PRÓBNE
KOMPUTER (TESTOWANIE PROGRAMU)
Poprawianie błędów wykonania
WYKAZ BŁĘDÓW WYKONANIA
WYNIKI POPRAWNE
PROGRAM UŻYTKOWY

Rys. 35.

nośników informacji, na jakie mogą być naniesione dane próbne, programy - źródłowy, wynikowy i użytkowy, jak również zabiegi związane z poprawianiem błędów formalnych i błędów wykonania.

2.8.3. Opracowanie dokumentacji programu użytkowego

Przedsięwzięciem końcowym procesu uruchamiania programu jest wykonanie jego dokumentacji technologiczno-eksploatacyjnej, w skład której wchodzi dokumentacja programowa oraz dokumentacja instrukcyjna.

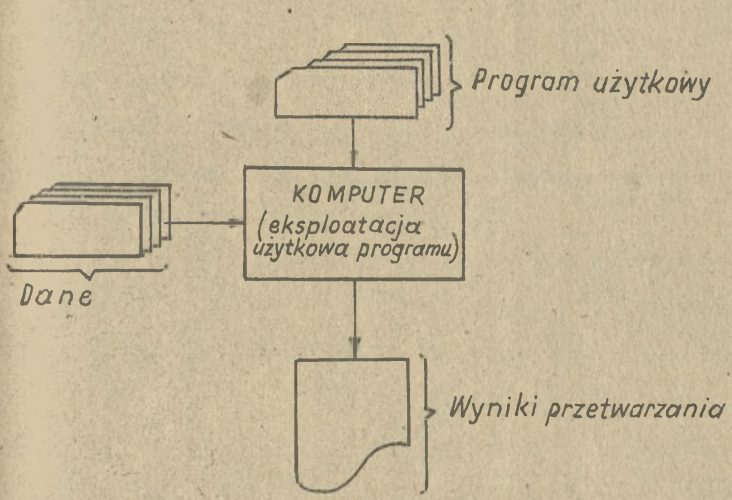
Dokumentacja programowa zawiera informacje niezbędne w operowaniu programem użytkowym, takie jak:

- nazwa programu, jego przeznaczenie i ogólna charakterystyka;
- algorytm zadania;
- schemat blokowy programu;
- wydruk programu źródłowego;
- maszynowy nośnik informacji z programem użytkowym;
- wzory danych wejściowych /źródłowych/;
- wzory danych wyjściowych /wyników/;
- konkretny przykład rozwiązania zadania;
- instrukcja operatorska.

Dokumentacja instrukcyjna opisuje sposoby realizacji czynności związanych z eksploatacją programu użytkowego i wykonywanych wyłącznie przez człowieka. Chodzi tu o zasady:

- wypełniania formularzy dokumentów źródłowych;
- kontroli poprawności danych wejściowych;
- poprawiania błędów;
- przenoszenia danych wejściowych z dokumentów źródłowych na maszynowe nośniki informacji;
- interpretacji oraz wykorzystania wyników itp.

Program użytkowy, wraz z jego dokumentacją, zostaje przekazany do biblioteki programów ośrodka obliczeniowego i w miarę potrzeby może być wykorzystywany przez użytkownika. Przykładową organizację eksploatacji użytkowej programu przedstawia rys. 36.



Rys. 36.

Pytania

1. Jakie wyróżniamy grupy zadań operacyjno-taktycznych? Scharakteryzować je z punktu widzenia przetwarzania za pomocą komputerów.
2. Wymienić etapy przygotowania zadania do rozwiązania za pomocą komputera.
3. Jakimi względami należy kierować się przy wyborze zadania operacyjno-taktycznego do rozwiązania komputerowego?

4. Omówić zasady formułowania i opisu zadania rozwiązywanego za pomocą komputera.
5. Scharakteryzować etap opracowywania modelu matematycznego zadania.
6. Scharakteryzować rodzaje modeli matematycznych.
7. Objaśnić pojęcie algorytmu.
8. Omówić cechy algorytmów.
9. Wymienić i scharakteryzować sposoby wyrażania algorytmów.
10. Omówić symbole graficzne stosowane w programowaniu blokowym.
11. Wymienić i scharakteryzować typy schematów blokowych /programów/.
12. Omówić klasyfikację języków programowania.
13. Scharakteryzować język wewnętrzny.
14. Scharakteryzować język symboliczny. Przykłady takich języków.
15. Scharakteryzować języki autokodowe.
16. Scharakteryzować język ALGOL-60.
17. Scharakteryzować język FORTRAN.
18. Scharakteryzować język COBOL.
19. Scharakteryzować język CSL.
20. Scharakteryzować język PL/1.
21. Scharakteryzować język BASIC.
22. Scharakteryzować język JEAN.
23. Omówić translację programu.
24. Wyjaśnić pojęcia: program źródłowy, program wynikowy, program użytkowy.
25. Wyjaśnić pojęcia: assembler, kompilator, interpreter.
26. Na czym polega testowanie programu.
27. Wyjaśnić pojęcia: błędy formalne programu, błędy wykonania programu.
28. Jaką rolę spełniają programy śledzące i programy monitujące.
29. Scharakteryzować eksploatację użytkową programu.
30. Omówić dokumentację programu użytkowego.

3. Zasady programowania w języku FORTRAN

W praktyce stosowane są dwie wersje języka FORTRAN: wersja podstawowa FORTRAN II oraz wersja standardowa FORTRAN IV. FORTRAN IV, powstały w wyniku modernizacji FORTRANu II pod wpływem języka ALGOL, posiada w porównaniu z FORTRANem IV rozszerzone możliwości.

Nas interesuje odmiana języka FORTRAN IV, przystosowana do angielskich komputerów serii LCL-1900, a tym samym komputerów serii ODRA-1300, znana pod nazwą FORTRAN-1900.

3.1. Podstawowe elementy języka FORTRAN

Przyjęto zasadę, że rzeczywiste kodowanie programu w języku FORTRAN wyrażane jest za pomocą dużych liter, natomiast małe litery stosowane są do symbolicznego zapisu pewnych wielkości.

3.1.1. Znaki używane w języku FORTRAN

Do zapisu słów języka FORTRAN używa się:

- liter alfabetu łacińskiego od A do Z /tylko dużych/
- cyfr dziesiętnych od 0 do 9
- znaków specjalnych:

+	znak liczby, symbol dodawania /plus/
-	znak liczby, symbol odejmowania /minus/
*	symbol mnożenia
/	symbol dzielenia
.	kropka
,	przecinek
=	symbol podstawiania
(	lewy nawias
)	prawy nawias
b, ␣	symbole spacji /odstępu/

W tak zwanej stałej TEXT oraz w tak zwanej H specyfikacji FORMATu dopuszczalne są znaki dodatkowe:

Celem uniknięcia pomyłek cyfra zero wyrażana jest symbolem ϕ.

3.1.2. Dyrektywy, instrukcje i komentarze

Dyrektywami /deklaracjami/ nazywamy informacje przeznaczone dla komputera i dotyczące struktury programu napisanego w języku zewnętrznym, identyfikacji typów danych i rezerwacji dla nich miejsca w PAO, organizacji odczytu danych oraz wydruku wyników itp.

Dyrektywy wykorzystywane są tylko w trakcie kompilacji programu, natomiast pomijane są podczas jego wykonywania.

Instrukcjami nazywamy informacje określające operacje, jakie komputer ma wykonywać w trakcie realizacji programu.

Komentarzami nazywamy informacje uzupełniające treść programu i przeznaczone wyłącznie dla programisty, operatora maszyny lub użytkownika programu.

Komentarze nie mają żadnego wpływu ani na przebieg kompilacji, ani też wykonania programu.

3.1.3. Ogólna struktura programu

Program w języku FORTRAN posiada strukturę segmentową i może składać się z jednego lub kilku segmentów. Każdy segment jest autonomiczny i może być oddzielnie pisany oraz kompilowany, niezależnie od segmentów pozostałych.

Język FORTRAN przewiduje cztery rodzaje segmentów, a mianowicie: MASTER, SUBROUTINE, FUNCTION oraz BLOCK DATA.

Segment MASTER jest systemem głównym, sterującym wykonaniem całego programu.

Segment SUBROUTINE przeznaczony jest do umieszczania podprogramu.

Segment FUNCTION stanowi jeden ze sposobów wyrażania wielkości funkcyjnych, definiowanych przez programistę.

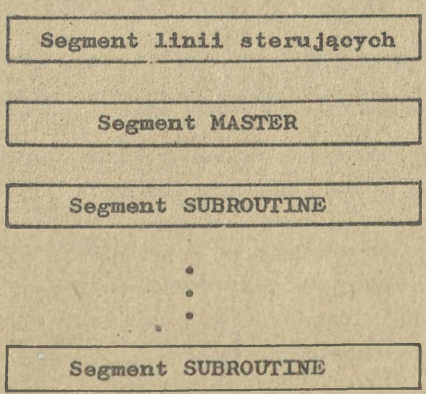
Segment BLOCK DATA wykorzystywany jest do nadawania wartości początkowych pewnym zmiennym.

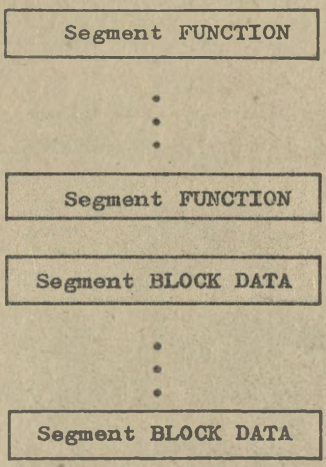
Jako oddzielny segment programu można potraktować także wiersze sterujące kompilatora, które zawierają informacje niezbędne do wytworzenia programu wynikowego.

Segment MASTER oraz segment wierszy sterujących muszą wystąpić w każdym programie. Natomiast pozostałe segmenty, zwane pomocniczymi, stosowane są w miarę potrzeby, przy czym w programie może wystąpić po kilka segmentów pomocniczych każdego rodzaju.

Bliższe zapoznanie się ze strukturą poszczególnych segmentów omówione zostanie w dalszej części opisu języka FORTRAN.

Struktura programu w języku FORTRAN może więc mieć następującą postać /rys. 37/.





Rys. 37.

3.1.4. Nazwy

Nazwę w języku FORTRAN stanowi ciąg złożony z liter i cyfr.

Nazwy stosowane są do identyfikacji programu, segmentów, zmiennych, tablic lub funkcji.

Przy tworzeniu nazw należy przestrzegać następujących zasad:

- pierwszym znakiem nazwy musi być litera, przy czym znak ten może mieć specjalne znaczenie /może określać typ wartości/;
- długość nazwy w języku FORTRAN nie może przekroczyć 32 znaków;
- spacje wewnątrz nazwy są ignorowane;
- nazwa wewnątrz jednego segmentu może mieć tylko jedno znaczenie;
- ta sama nazwa może być użyta w różnych segmentach i nie musi odnosić się do tej samej wielkości.

Przykłady nazw:

ILOSC IbLbObSbC	nazwy równoważne /b - oznacza spację/
T-34	nazwa niewłaściwa, bo zawiera znak specjalny, jakim jest myślnik
3M	nazwa niewłaściwa, bo zaczyna się od cyfry, a nie litery
SUMA 1	

3.1.5. Arkusz programowy

Arkusz programowy /rys. 32/ podzielony jest na wiersze oraz 80 ponumerowanych kolumn. Każda kolumna w wierszu przeznaczona jest na jeden znak. Przeznaczenie poszczególnych kolumn jest następujące:

kolumny 7 - 72	- zapis dyrektyw i instrukcji
kolumny 73 - 80	- notatki programisty /np. numeracja kart perforowanych/
kolumny 1 - 5	- zapis etykiety
kolumna 6	- znak kontynuacji poprzedniego wiersza

Każda dyrektywa lub instrukcja muszą być pisane z nowego wiersza, przy czym mogą one zająć część wiersza, cały wiersz bądź więcej niż cały wiersz. Jeżeli zapis dyrektywy lub instrukcji zajmuje więcej niż jeden wiersz, wówczas każdy następny wiersz zapisu należy poprzedzić symbolem kontynuacji wiersza, umieszczonym w 6-tej kolumnie. Symbolem tym może być dowolny z dopuszczalnych znaków alfanumerycznych, z wyjątkiem spacji lub zera.

Etykietą nazywamy numer nadawany niektórym dyrektywom lub instrukcjom /np. przy organizowaniu pracy cyklicznej, odczytu, wydruku/.

Etykietyzowanie wymaga przestrzegania następujących zasad:

- etykietą może być dowolna liczba całkowita z przedziału 1 - 99999;
- etykieta musi być zapisana w pierwszym wierszu dyrektywy lub instrukcji;
- przy etykietyzowaniu programu może być stosowana dowolna kolejność;
- zapis etykiety może rozpoczynać się z dowolnej kolumny 1-5;
- do etykiety można odwoływać się tylko w tym segmencie, w którym ona występuje;
- w jednym segmencie nie może być jednakowych etykiet.

Komentarze mogą być zapisywane w kolumnach 2-72. Każdy wiersz komentarza musi być poprzedzony literą C, umieszczoną w pierwszej kolumnie.

3.2. Dane stałe, zmienne proste i zmienne indeksowane

Język FORTRAN przewiduje możliwość użycia danych w postaci stałych, zmiennych prostych oraz zmiennych indeksowanych.

3.2.1. Opis danych stałych

Stałe reprezentowane są w wyrażeniach języka FORTRAN przez swoją wartość, której nie mogą zmieniać w czasie wykonywania programu. Rozróżnia się przy tym następujące typy danych stałych:

- INTEGER /integer numbers - liczby całkowite/;
- REAL /real numbers - liczby rzeczywiste/;
- DOUBLE PRECISION /double precision numbers - liczby podwójnej dokładności/;
- COMPLEX /complex numbers - liczby zespolone/;
- LOGICAL /logical constans - stałe logiczne/;
- TEXT /Hollerith constans - stałe Holleritha, czyli stałe tekstowe/.

Stałe typu INTEGER są to liczby całkowite, wykorzystywane do organizacji obliczeń oraz jako wskaźniki elementów tablic.

Stałych typu INTEGER w zasadzie nie używa się do obliczeń, ponieważ wynik działania /np. dzielenia/ na liczbach całkowitych może nie być liczbą całkowitą. Liczby całkowite charakteryzują się tym, że:

- nie zawierają kropki dziesiętnej ani litery E;
- mogą być poprzedzone znakiem plus lub minus, przy czym znak plus może być pominięty;
- ich zakres wynosi od -8388607 do +8388607;
- zajmują w PAO jednostkę pojemności, czyli dwie komórki.

Przykłady: -10, 407, 0, +999

Stałe typu REAL są to liczby rzeczywiste używane do obliczeń /wyrażania danych i wyników/.

Istnieją dwie postacie liczb rzeczywistych:

- postać z kropką dziesiętną;
- postać wykładnicza

Postać z kropką dziesiętną przedstawiana jest jako ciąg cyfr, zawierający kropkę pozycyjną i rozpoczynający się cyfrą, znakiem plus lub minus, bądź kropką pozycyjną.

Wprowadzając oznaczenia:

n - ciąg cyfr części całkowitej
m - ciąg cyfr części ułamkowej,

stosowane postacie liczb rzeczywistych z kropką dziesiętną można przedstawić następująco:

± n.	np. 35.
± n.m	np. -105.03
± .m	np. -.28 /-0,28/

Postać wykładnicza złożona jest z mantysy, która jest liczbą rzeczywistą z kropką dziesiętną lub liczbą całkowitą, z litery E i z następującej po niej cechy, która jest liczbą całkowitą.

Przyjmując dodatkowe oznaczenia:

E - oznaczenie potęgi o podstawie 10
S - cecha /wykładnik potęgowy/,

można stosowane postacie wykładnicze liczb rzeczywistych przedstawić następująco:

± n.E ± S	np. -56.E-2/$-56 \cdot 10^{-2} = -0,56$/
± n.m E ± S	np. 2.5 E 3 /$2,5 \cdot 10^{3} = 2500$/
± .m E ± S	np. -.37 E + 2 /$-0,37 \cdot 10^{2} = -37$
± n E ± S	np. 320 E - 4 /$320 \cdot 10^{-4} = 0,032$/

W przypadku danych wejściowych dopuszcza się ponadto postać z pominięciem litery E.

Przykłady: 25.3 - 2 /$25,3 \cdot 10^{-2} = 0,253$/
152 - 3 /$152 \cdot 10^{-3} = 0,152$/

Zakres liczb typu REAL wynosi od $-5,6 \cdot 10^{76}$ do $5,6 \cdot 10^{76}$, dokładność - 11 cyfr znaczących. Zajmują one w PAO jedną jednostkę.

Stałe typu DOUBLE PRECISION są liczbami wykorzystywanymi do prowadzenia szczególnie dokładnych obliczeń.

Zajmują one w PAO 2 jednostki i posiadają następujące postacie:

± n.D ± S	np. 32.D - 1/$32 \cdot 10^{-1} = 3,2$/
± n.m D ± S	np. 251.027 D+2/$251,027 \cdot 10^{2} = 25102,7$/
± .m D ± S	np. .26D2 /$0,26 \cdot 10^{2} = 26$/
± nD ± S	np. -356 D - 4 /$-356 \cdot 10^{-4} = 0,0356$/

gdzie: D - oznaczenie potęgi o podstawie 10
S - wykładnik potęgi

Zakres liczb typu DOUBLE PRECISION jest taki sam jak dla liczb typu REAL.

Stałe typu COMPLEX są to liczby zespolone.

Zapisywane są one w języku FORTRAN jako uporządkowana para liczb typu REAL, oddzielonych przecinkiem i ujętych w nawiasach zwykłych. Pierwsza liczba takiej pary oznacza część rzeczywistą, a druga - część urojoną liczby zespolonej. Ich postać jest zatem następująca: (A, B).

Przykłady: (5.2,2.12) oznacza 5,2 + 2,12 i
(-5.,.2) oznacza -5 + 0,2 i

Liczby typu COMPLEX zajmują w PAO 2 jednostki.

Stałe typu LOGICAL są to stałe używane w wyrażeniach logicznych.

Istnieją tylko dwie takie stałe: .TRUE. /prawda/ oraz .FALSE. /fałsz/.

Stałe typu TEXT. zwane też stałymi Holleritha, oznaczają zapis tekstowy o określonej liczbie znaków. Przyjmują one następującą postać:

n H s

gdzie: n - liczba znaków /łącznie ze spacjami/
s - ciąg znaków dopuszczalnych w języku FORTRAN-1900.

Przykłady: 5 H ILOSC
9 H I␣L␣O␣S␣C
16 H ZADANIE ␣ PIERWSZE
10 H CZOLG ␣ T-34

3.2.2. Opis danych zmiennych

Dane, które w trakcie wykonywania obliczeń zmieniają swoje wartości, nazywane są zmiennymi. Rozróżnia się przy tym zmienne proste oraz zmienne indeksowane /zmienne ze wskaźnikami/.

3.2.2.1. Zmienne proste

Zmienne proste są to pojedyncze dane, przechowywane w komórkach PAO, którym nadaje się nazwy /nazwy zmiennych/ na ogół tak tworzone, aby swym sensem odpowiadały przechowywanym /reprezentowanym/ danym.

Język FORTRAN przewiduje 5 typów zmiennych prostych: INTEGER, REAL, DOUBLE PRECISION, COMPLEX, LOGICAL. Wyszczególnione powyżej słowa wykorzystywane są do deklarowania typów zmiennych i noszą nazwę dyrektyw /deklaracji/ typu. Dyrektywa typu ma postać:

"dyrektywa typu" lista zmiennych prostych, oddzielonych przecinkami

Przykłady:

INTEGER WIELKOSC - zmienna o nazwie WIELKOSC jest liczbą całkowitą

INTEGER ALFA, W2, STOPIEN

REAL MASA, ILOSC - zmienne o nazwach MASA, ILOSC są liczbami rzeczywistymi

DOUBLE PRECISION WAGA,M25 - zmienne WAGA, M25 są liczbami o podwójnej dokładności

COMPLEX PIERWIASTEK 1, PIERWIASTEK 2 - deklaracja zmiennych zespolonych

LOGICAL TEZA, B1 - deklaracja zmiennych logicznych

Zmienne proste typu INTEGER lub REAL można deklarować również za pomocą pierwszej litery ich nazw. Jeżeli nazwy rozpoczynają się od liter I,J,K,L,M,N, to kompilator reprezentowanym przez nie zmiennym przypisuje typ INTEGER. Natomiast, gdy nazwy

zmiennych rozpoczynają się od pozostałych liter alfabetu /A,B,C, D,E,F,G,H,O,P,Q,R,S,T,U,W,V,X,Y,Z/, wówczas reprezentują one zmienne typu REAL.

Przykłady:

BILANS, ZAPAS . - nazwy zmiennych typu REAL
LICZNIK 1, INDEKS - nazwy zmiennych typu INTEGER

3.2.2.2. Tablice, zmienne indeksowane

W języku FORTRAN grupę zmiennych tego samego typu można połączyć pod wspólną nazwą w jedną tablicę, przechowywaną w wydzielonym obszarze PAO. Zmienne, stanowiące elementy tablic, uporządkowane są za pomocą indeksów i stąd też nazywa się je zmiennymi indeksowanymi.

Zmienne indeksowane, a więc i cała tablica, mogą być następujących typów: INTEGER, REAL, DOUBLE PRECISION, COMPLEX, LOGICAL. Nadawanie typu tablicom odbywa się za pomocą dyrektyw typu lub pierwszej litery nazwy tablicy /analogicznie jak to ma miejsce w przypadku zmiennych prostych/.

W FORTRANIE można stosować tablice jednowymiarowe i wielowymiarowe, przy czym w FORTRANIE-1900 dopuszczalne są tablice do 32 wymiarów. Wymiary tablic oraz zakres wartości, które mogą przyjmować indeksy elementów tablic, określa się za pomocą dyrektyw: DIMENSION, COMMON lub typu, użytych zgodnie ze schematem:

"dyrektywa" A(i), B(j), ..., X(y)

gdzie: A,B, ..., X - nazwy tablic
i,j, ..., y - lista wskaźników określających wymiary i wielkości tablic

Przykład:

```
INTEGER WEKTOR
DIMENSION WEKTOR (100)
```

Pierwsza dyrektywa informuje komputer, że tablica o nazwie WEKTOR posiada elementy typu INTEGER, natomiast druga określa, że jest to tablica jednowymiarowa i stuelementowa /zajmuje w PAO 100 jednostek/.

Przykład:

```
REAL     A1, K1, MACIERZ
COMMON   A1(5,10,3), K1(50), MACIERZ (15,2)
```

Obie dyrektywy deklarują trzy tablice typu REAL, z których pierwsza jest 3-wymiarowa i 150-elementowa, druga - jednowymiarowa i 50-elementowa, trzecia - dwuwymiarowa i 30-elementowa.

Przykład:

```
DOUBLE PRECISION OMEGA (10, 15)
```

Przytoczona dyrektywa deklaruje jednocześnie typ tablicy, jej nazwę, wymiarowość i wielkość.

Przykład:

```
INTEGER TABLICA
DIMENSION TABLICA (5,10)
```

zapisy równoważne

```
INTEGER TABLICA
COMMON  TABLICA (5,10)
INTEGER TABLICA (5,10)
```

Odwoływanie się do całej tablicy jest możliwe poprzez podanie jej nazwy, natomiast odwoływanie się do dowolnego elementu tablicy wymaga podania nazwy tablicy oraz odpowiedniej listy indeksów umieszczonej w nawiasach. Indeksy te muszą być liczbami typu INTEGER.

Przykłady:

WEKTOR (3) - trzeci element tablicy jednowymiarowej o nazwie WEKTOR

MACIERZ 2,3 - element występujący na przecięciu drugiego wiersza i trzeciej kolumny tablicy o nazwie MACIERZ

Indeksem może być stała, zmienna lub wynik wyrażenia arytmetycznego, zawierającego dodawanie, odejmowanie lub mnożenie. Dopuszczalne postacie indeksów przytacza tablica nr 1.

Tablica nr 1

Postać ogólna	Przykłady
V	M
C	4
V + C	K + 5
V - C	L - 3
C * V	2 * N
C*V + C'	3*I + 1
C*V - C'	5*I - 2

V - dowolna zmienna prosta

C,C' - liczby całkowite bez znaku

Przykład:

BETA (3*K, N - 1)

Jeżeli: K = 1, N = 5, to chodzi o element BETA (3,4).

3.3. Wyrażenia

Wyrażenie FORTRANu stanowi układ stałych i zmiennych połączonych znakami operacji.

Rozróżnia się przy tym wyrażenia arytmetyczne oraz wyrażenia logiczne.

3.3.1. Wyrażenia arytmetyczne

Wyrażeniem arytmetycznym nazywamy odpowiednio skonstruowany ciąg operatorów arytmetycznych oraz elementów typu: INTEGER, REAL, DOUBLE PRECISION, COMPLEX.

Do budowy wyrażeń arytmetycznych mogą być użyte następujące operatory:

- dodawania +
- odejmowania -
- mnożenia *
- dzielenia /
- potęgowania **

Natomiast elementami wyrażeń arytmetycznych mogą być:

- stała;
- zmienna prosta;
- zmienna indeksowana;
- funkcja;
- wyrażenie arytmetyczne ujęte w nawiasy okrągłe.

Wyrażenie arytmetyczne ujęte w nawiasach traktowane jest jako pojedynczy element bardziej złożonego wyrażenia.

Należy zdać sobie również sprawę z tego, że pojedynczy element /stała, zmienna/ jest także wyrażeniem arytmetycznym.

Przy budowaniu wyrażeń arytmetycznych należy przestrzegać następujących reguł:

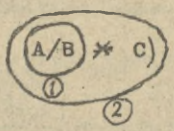
- między elementami wyrażenia muszą występować operatory arytmetyczne;
- dwa operatory arytmetyczne nie mogą występować kolejno po sobie;
- w celu określenia kolejności wykonywania operacji w wyrażeniu arytmetycznym mogą być użyte nawiasy, przy czym reguły ich stosowania są takie same, jak w tradycyjnym zapisie algebraicznym.

Realizacja obliczania wartości wyrażenia arytmetycznego rozpoczyna się od najbardziej wewnętrznej pary nawiasów i kontynuowana jest na zewnątrz, natomiast w jednej parze nawiasów, lub w wyrażeniu jako całości, obliczenia realizowane są w następującej kolejności:

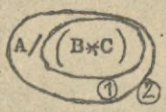
1/ funkcje i wyrażenia w nawiasach
2/ potęgowanie
3/ mnożenie i dzielenie
4/ dodawanie i odejmowanie

Wewnątrz wymienionych grup przyjmuje się kolejność wykonywania operacji od lewej ku prawej..

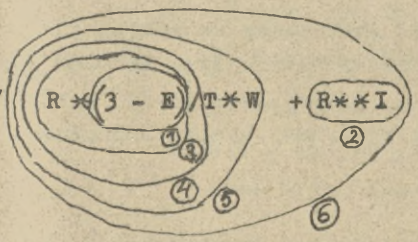
Przykłady /kolejność obliczeń oznaczona jest kolejnymi liczbami naturalnymi/

1/ 

$\left(\frac{A}{B} \cdot C\right)$

2/ 

$\left(\frac{A}{BC}\right)$

3/ 

$\left(R \cdot \frac{3-E}{T} \cdot W + R^{I}\right)$

W przykładzie trzecim, równorzędne operacje 3 , 4 i 5 wykonywane są w kolejności z lewa na prawo.

Typ wyrażenia arytmetycznego zależy od typu użytych w nim elementów i operatorów. Odpowiednie zależności podane są w tabelach nr 2 i nr 3.

Tabela nr 2

Operatory: + - * /		Typ elementu drugiego			
		INTEGER	REAL	COMPLEX	DOUBLE PRECISION
Typ elementu pierwszego	INTEGER	INTEGER	REAL	COMPLEX	DOUBLE PRECISION
	REAL	REAL	REAL	COMPLEX	DOUBLE PRECISION
	COMPLEX	COMPLEX	COMPLEX	COMPLEX	N
	DOUBLE PRECISION	DOUBLE PRECISION	DOUBLE PRECISION	N	DOUBLE PRECISION

Tabela nr 3

Operator **		Wykładnik			
		INTEGER	REAL	COMPLEX	DOUBLE PRECISION
Podstawa potęgi	INTEGER	INTEGER	REAL	N	DOUBLE PRECISION
	REAL	REAL	REAL	N	DOUBLE PRECISION
	COMPLEX	COMPLEX	N	N	N
	DOUBLE PRECISION	DOUBLE PRECISION	DOUBLE PRECISION	N	DOUBLE PRECISION

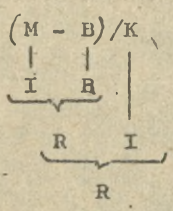
Objaśnienia:

1/ N oznacza kombinację nieodzowną;

2/ podstawa 0 nie może być podnoszona do potęgi zerowej;

3/ element ujemny nie może być podnoszony do potęgi REAL bądź DOUBLE PRECISION. Wynik wyrażenia typu INTEGER jest najbliższą liczbą całkowitą, której bezwzględna wartość nie przekracza wartości bezwzględnej wyniku poprawnego.

Przykład:



typ użytych zmiennych określają ich nazwy

3.3.2. Wyrażenia logiczne

Wyrażeniem logicznym nazywamy odpowiednio skonstruowany ciąg operatorów logicznych i elementów typu LOGICAL.

Zarówo elementy logiczne jak i wyrażenia logiczne mogą przyjmować tylko jedną z dwóch wartości:

.TRUE. /prawda/ lub .FALSE. /falsz/

kropki obowiązkowe

Elementami wyrażeń logicznych mogą być:

- stała logiczna;
- zmienna logiczna;
- funkcja logiczna;
- para wyrażeń arytmetycznych oddzielonych operatorami relacji;
- wyrażenie logiczne ujęte w nawiasy okrągłe.

Język FORTRAN przewiduje trzy operatory logiczne:

.NOT. /negacja - nie/
.AND. /koniunkcja - i/
.OR. /alternatywa - lub/

Przy czym: .NOT.X jest .FALSE., gdy X jest .TRUE.

.NOT.X jest .TRUE., gdy X jest .FALSE.

X.AND.Y jest .TRUE., gdy X i Y są .TRUE.,

w pozostałych przypadkach X.AND.Y jest .FALSE.

X.OR.Y jest .FALSE., gdy X i Y są .FALSE., w pozostałych przypadkach X.OR.Y jest .TRUE.

W wyrażeniach logicznych mogą być użyte relacje zawierające następujące operatory:

.LT. - mniejszy niż /Less Than/
.LE. - mniejszy lub równy /Less than or Equal to/
.EQ. - równy /EQual to/
.NE. - nie równy /Not Equal to/
.GT. - większy /Graeter Than/
.GE. - większy lub równy /Graeter than or Equal to/

Relacje przyjmują następującą postać:

$$e_1 \; r \; e_2.$$

gdzie: e_1, e_2 - wyrażenie arytmetyczne
r - operator relacji

Relacja przyjmuje wartość .TRUE. wtedy, gdy wartości wyrażeń e_1 i e_2 połączone operatorem relacji, spełniają ją. Jeśli natomiast

relacja nie jest spełniona, wówczas przyjmuje ona wartość .FALSE.

Przykłady relacji:

A.GE.1ØØ..AND. A.LT.2ØØ	/100 ⩽ a ⩽ 200/
A.GT.B. OR.C.LE.D	/a ⩾ b lub c ⩽ d/
B**2.GT.4*A*C	/$b^2 > 4ac$/

Przy konstruowaniu wyrażeń logicznych muszą być przestrzegane następujące zasady:

- między elementami logicznymi wyrażenia muszą wystąpić operatory logiczne;
- obok siebie mogą wystąpić dwa operatory logiczne tylko wtedy, gdy pierwszym jest .AND. lub .OR., a drugim .NOT.;
- między wyrażeniami arytmetycznymi muszą wystąpić operatory relacji;
- w celu określenia kolejności wykonywania operacji, w wyrażeniu logicznym mogą być użyte nawiasy.

Podobnie jak w przypadku wyrażeń arytmetycznych, określenie wartości wyrażenia logicznego rozpoczyna się od najbardziej wewnętrznej pary nawiasów i kontynuuje na zewnątrz, natomiast w jednej parze nawiasów lub w wyrażeniu jako całości kolejność działań jest następująca:

1/ relacje, funkcje i wyrażenia logiczne w nawiasach
2/ operacja .NOT.
3/ operacja .AND.
4/ operacja .OR.

Wewnątrz wymienionych grup operacje o tym samym priorytecie wykonywane są w kolejności od lewej ku prawej.

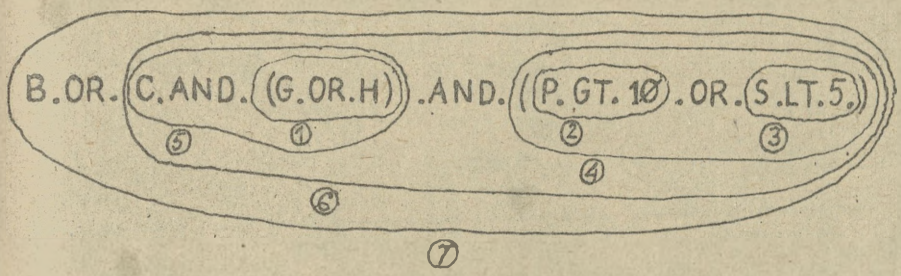
Przykład:

Napisać w języku FORTRAN następujące wyrażenie logiczne:

b lub c i (g lub h) i (p 10 lub s 5)

i określić, w jakiej kolejności będzie ono realizowane.

Rozwiązanie:



3.4. Funkcje standardowe

W języku FORTRAN istnieją trzy sposoby wyrażania wielkości funkcyjnych, a mianowicie:

- za pomocą funkcji standardowych;
- za pomocą funkcji lokalnych;
- za pomocą segmentu FUNCTION

W tym miejscu zajmiemy się jedynie funkcjami standardowymi, czyli typowymi funkcjami matematycznymi przechowywanymi w PAO wraz z kompilatorem FORTRANu i dołączonymi do programu wynikowego.

Funkcje standardowe posiadają następującą postać:

"nazwa funkcji" (lista argumentów)

W tabeli nr 4 zestawione zostały funkcje standardowe najczęściej używane w programowaniu.

Tabela nr 4

Nazwa funkcji	Opis funkcji	Liczba argumentów	Typ argumentów	Typ funkcji
1	2	3	4	5
ABS IABS DABC	Wartość bezwzględna (a)	1	REAL INTEGER D.P.	REAL INTEGER D.P.
AMAXO AMAX1 MAXO MAX1 DMAX1	Wybór największej wartości Max $(a_1, a_2, \ldots, a_n)$	2	INTEGER REAL INTEGER REAL D.P.	REAL REAL INTEGER INTEGER D.P.
AMINO AMIN1 MINO MIN1 DMIN1	Wybór najmniejszej wartości Min $(a_1, a_2, \ldots, a_n)$	2	INTEGER REAL INTEGER REAL D.P.	REAL REAL INTEGER INTEGER D.P.
FLOAT	Zamiana liczby całkowitej na rzeczywistą	1	INTER	REAL
IFIX	Zamiana liczby rzeczywistej na całkowitą	1	REAL	INTEGER
EXP DEXP CEXP	Funkcja wykładnicza e^x	1	REAL D.P. COMPLEX	REAL D.P COMPLEX
EXP10	Funkcja wykładnicza przy podstawie 10	1	REAL	REAL
ALOG DLOG CLOG	Logarytm naturalny $\log_e(a)$	1	REAL D.P. COMPLEX	REAL D.P. COMPLEX
ALOG10 DLOG10	Logarytm dziesiętny $\log_{10}(a)$	1	REAL D.P.	REAL D.P.
ALOG2	Logarytm przy podstawie 2 $\log_2(a)$	1	REAL	REAL
SIN DSIN CSIN	Funkcja sinus $\sin(a)$	1	REAL D.P. COMPLEX	REAL D.P. COMPLEX
COS DCOS CCOS	Funkcja cosinus $\cos(a)$		REAL D.P. COMPLEX	REAL D.P. COMPLEX
TAN	Funkcja tangens $tg(a)$	1	REAL	REAL

Tabela nr 4 /od/

1	2	3	4	5
COT	Funkcja cotangens ctg (a)	1	REAL	REAL
SINH	F.sinus hiperboliczny sinh (a)	1	REAL	REAL
COSH	F.cosinus hiperboliczny cosh (a)	1	REAL	REAL
TANH	F.tangens hiperboliczny tgh (a)	1	REAL	REAL
COTH	F.cotangens hiperboliczny otgh (a)	1	REAL	REAL
SQRT DSQRT CSQRT	Pierwiastek kwadratowy $(a)^{\frac{1}{2}}$	1	REAL D.P. COMPLEX	REAL D.P. COMPLEX
ASIN	Arcus sinus arc sin (a)	1	REAL	REAL
ACOS	Arcus cosinus arc cos (a)	1	REAL	REAL
ATAN	Arcus tangens arc tg (a)	1	REAL D.P.	REAL D.P.
CABS	Moduł argumentu zespolonego Moduł |a|	1	COMPLEX	REAL

Uwaga: Argumenty funkcji trygonometrycznych muszą być podstawiane w radianach.

Aby wywołać w programie funkcję standardową należy podać nazwę funkcji i w nawiasie argument /argumenty/, którym może być dowolne wyrażenie arytmetyczne, w tym wyrażenie zawierające również wywołanie funkcji.

Przykłady:

1/ SQRT (2.Ø*A)

2/ 4*SQRT (ALOG (G))

$$\frac{(\sqrt{2a})}{(4\sqrt{\log_e g})}$$

3/ R* SIN (GAMA) $(r \sin\gamma)$

Funkcje standardowe można wywoływać w dowolnym segmencie /oprócz segmentu BLOCK DATA/.

3.5. Instrukcje podstawiania

Instrukcja podstawiania jest podstawową instrukcją w procesie obliczeniowym. Powoduje ona zmianę wartości zmiennej prostej lub zmiennej indeksowanej. W języku FORTRAN stosowane są dwa typy instrukcji podstawiania:

- instrukcja podstawiania arytmetycznego;
- instrukcja podstawiania logicznego.

Pierwsza z nich ma następującą postać:

"nazwa zmiennej lub ciąg nazw" = wyrażenie arytmetyczne

Symbol = nie oznacza znaku równości, lecz posiada sens "ma być zastąpione przez".

Działanie instrukcji:

1/ w trakcie wykonywania instrukcji następuje obliczenie wartości wyrażenia arytmetycznego i podstawienie jej w miejsce poprzedniej wartości zmiennej /zmiennych/,

2/ wartości wyrażenia arytmetycznego występującego po prawej stronie nadaje się taki typ, jaki posiada zmienna występująca po lewej stronie; pewne typy wyników wyrażeń arytmetycznych są niedopuszczalne /tabela nr 5/.

Tabela nr 5

TYP WYRARZ. / TYP ZMIENNEJ	INTEGER	REAL	D.P.	COMPLEX	LOGICAL
INTEGER	T	T	T	N	N
REAL	T	T	T	N	N
D.P.	T	T	T	N	N
COMPLEX	N	N	N	T	N
LOGICAL	N	N	N	N	T

T - tak, N - nie

Uwagi:

1/ po lewej stronie znaku podstawiania nie może pojawić się wyrażenie arytmetyczne,

2/ po prawej stronie może wystąpić ta sama zmienna, która występuje po stronie lewej.

Przykłady:

```
I = 25
LICZNIK = LICZNIK + 1
WARTOSC = ILOSC*CENA
A(I,K) = B(I,K) + C(I)
DELTA = B**2 - 4*A*C
A,B,C = ø  /równoważne trzem instrukcjom: A = 0, B = 0 oraz C = 0/
A,B = SQRT (A) + B
X1 = (-B - SQRT (DELTA))/(2*A)
```

Postać instrukcji podstawienia logicznego przedstawia się następująco:

"nazwa zmiennej lub ciąg nazw" = wyrażenie logiczne

Działanie instrukcji:

W trakcie wykonywania instrukcji zmienna /zmienne/ występująca po lewej stronie uzyskuje jedną z dwóch wartości .TRUE. lub .FALSE., w zależności od wyniku wyrażenia logicznego.

Przykłady:

1/ W = B**2. GT. 4*A*C
.TRUE. .TRUE.

2/ GAMA = ALFA. AND. BETA
.FALSE. .TRUE. .FALSE.

3.6. Funkcje lokalne

Oprócz funkcji standardowych, programista - pisząc program - może korzystać z funkcji definiowanych przez siebie, zwanych funkcjami lokalnymi. Nazwa "funkcje lokalne" pochodzi stąd, iż mogą one być użyte tylko w tym segmencie, w którym zostały zdefiniowane, przy czym zdefiniowanie to musi poprzedzać wywołanie funkcji.

Funkcje lokalne wyliczają tylko jedną wartość i w zasadzie stanowią skróconą formę zapisu instrukcji podstawiania dla wyrażenia arytmetycznego lub logicznego.

Do funkcji lokalnej można odwoływać się kilkakrotnie. Nie ma ona tylko charakteru rekursywnego, tzn., że nie może odwoływać się do samej siebie.

Postać funkcji lokalnej:

"nazwa funkcji" (lista parametrów formalnych) = wyrażenie arytmetyczne lub logiczne

Nazwa funkcji nadawana jest zgodnie z zasadami tworzenia nazw. Jeśli pierwsza litera nazwy nie określa typu funkcji, to typ

ten musi być zadeklarowany za pomocą dyrektywy typu w segmencie zawierającym funkcję lokalną.

Parametry formalne są to zmienne proste, za pomocą których definiowana jest funkcja lokalna poprzez odpowiednie wyrażenie arytmetyczne lub logiczne. Funkcja lokalna musi posiadać przynajmniej jeden parametr formalny. W definicji funkcji lokalnej mogą wystąpić również zmienne nie wyszczególnione na liście parametrów formalnych pod warunkiem, że będą one znane w momencie obliczania wartości funkcji. W składzie definicji funkcji lokalnej mogą wystąpić inne, zdefiniowane uprzednio, funkcje lokalne jak również funkcje standardowe.

Wywołanie funkcji ma następującą postać:

$$\text{"nazwa funkcji"}(a_1, a_2, \ldots a_n)$$

gdzie: a_1, a_2, ..., a_n - parametry aktualne, którymi mogą być dowolne wyrażenia.

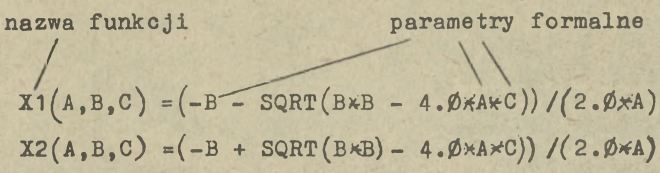
Działanie funkcji lokalnej

W momencie napotkania w wyrażeniu nazwy funkcji lokalnej, z podanymi w nawiasach parametrami aktualnymi, następuje podstawienie do definicji funkcji parametrów aktualnych w miejsce parametrów formalnych, obliczenie wartości funkcji i wstawienie tej wartości w odpowiednie miejsce obliczanego wyrażenia.

Między parametrami aktualnymi a formalnymi musi wystąpić zgodność co do ich kolejności, liczby i typu.

Przykład:

Jeżeli w pewnych wyrażeniach zachodzi potrzeba wyliczania pierwiastków równań kwadratowych, to obliczanie tych pierwiastków można zorganizować poprzez dwie funkcje lokalne.



Wywołanie tych funkcji w trakcie obliczania wyrażenia arytmetycznego nastąpi poprzez podanie nazwy funkcji wraz z parametrami aktualnymi.

Przykład

```
W1 = SIN(ALFA) + X1(2.5, 4.Ø, - .52)
W2 = COS(BETA) - X2(S, 3.5 B, D + E/F)
```

3.7. Instrukcje sterujące

Instrukcje programu realizowane są podczas jego wykonywania na ogół w kolejności ich zapisania. Kiedy jednak zachodzi konieczność zmiany kolejności wykonywania instrukcji, chwilowego wstrzymania realizacji programu lub zakończenia jego przebiegu, wówczas należy użyć odpowiedniej instrukcji sterującej.

Wśród instrukcji sterujących języka FORTRAN rozróżniamy:

- instrukcje skokowe,
- instrukcje warunkowe,
- instrukcję powtórzeń

oraz instrukcje powodujące wstrzymanie działania programu, jego kontynuację lub zakończenie.

3.7.1. Instrukcje skokowe /GO TO .../

Do instrukcji skokowych zaliczamy:

- instrukcję skoku bezwarunkowego GO TO;
- instrukcję skoku warunkowego GO TO;
- instrukcję skoku warunkowego GO TO określanego instrukcją ASSIGN.

3.7.1.1. Instrukcja skoku bezwarunkowego GO TO

Postać instrukcji

```
GO TO k
```

gdzie: GO TO oznacza "skocz do"

k - oznacza etykietę instrukcji tego samego segmentu programu, w którym występuje instrukcja GO TO K

Działanie instrukcji

W trakcie wykonywania instrukcji realizacja programu zostaje przeniesiona do instrukcji o etykiecie K.

Przykład /fragmentu programu/

```
      .     .
      .     .
      .     .
    B = B + 10*C
    GO TO 5 ------------------+
instrukcje  .     .           |
pominięte   .     .           |
            .     .           |
  5 A = B**2 <----------------+
      .     .
      .     .
      .     .
```

skok do instrukcji o etykiecie 5

Bezwarunkowa instrukcja GO TO pojawia się zwykle po grupie instrukcji realizowanych przez komputer w wyniku jakiejś innej instrukcji sterującej.

3.7.1.2. Instrukcja skoku warunkowego GO TO

Postać instrukcji

```
GO TO (k1, k2, ..., kn), i
```

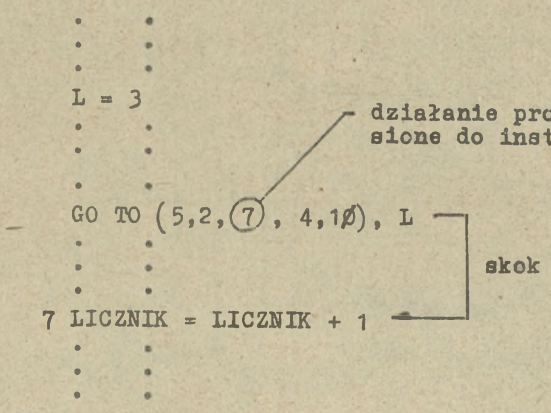
gdzie: $k_1, k_2, \ldots, k_n$ - lista etykiet instrukcji tego samego segmentu, w którym występuje GO TO ...

i - zmienna prosta typu INTEGER

Działanie instrukcji

Podczas wykonywania instrukcji realizacja programu zostaje przeniesiona do instrukcji o etykiecie, której miejsce wśród etykiet $k_1, k_2, \ldots, k_n$ określa wartość zmiennej "i" w taki sposób, że jeśli np. wartość ta wynosi 3, to jako następna będzie realizowana instrukcja o etykiecie występującej na trzecim miejscu listy etykiet.

Przykład



Gdyby zawartość zmiennej "i" była większa od liczby elementów zawartych na liście etykiet, wówczas sposób wykonania instrukcji skoku warunkowego byłby nieokreślony.

Instrukcja skoku warunkowego stosowana jest wtedy, gdy programista chce zmienić kolejność wykonywania programu w zależności od wartości określonej zmiennej.

3.7.1.3. Instrukcja skoku warunkowego GO TO określonego instrukcja ASSIGN

Postać instrukcji

GO TO i, $(k_1, k_2, \ldots, k_n)$

gdzie: i, k_1, k_2, ..., k_n - jak w instrukcji poprzedniej.

Działanie instrukcji

Instrukcja przenosi realizację programu do miejsca uzależnionego od wykonywanej wcześniej instrukcji ASSIGN, przypisującej zmiennej "i" wartość będącą etykietą instrukcji, do której wykonywany jest skok.

Postać instrukcji ASSIGN

```
ASSIGN k TO i
```

gdzie: ASSIGN... TO - przypisz do

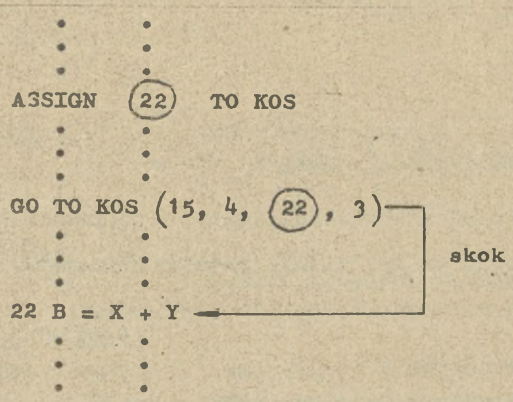
k - etykieta instrukcji tego samego segmentu, w którym występują ASSIGN oraz GO TO, i ..., i do której ma nastąpić skok

i - zmienna typu INTEGER

Działanie instrukcji ASSIGN

Instrukcja przyporządkowuje etykietę "k" zmiennej "i" /ale nie nadaje jej takiej wartości/.

Przykład



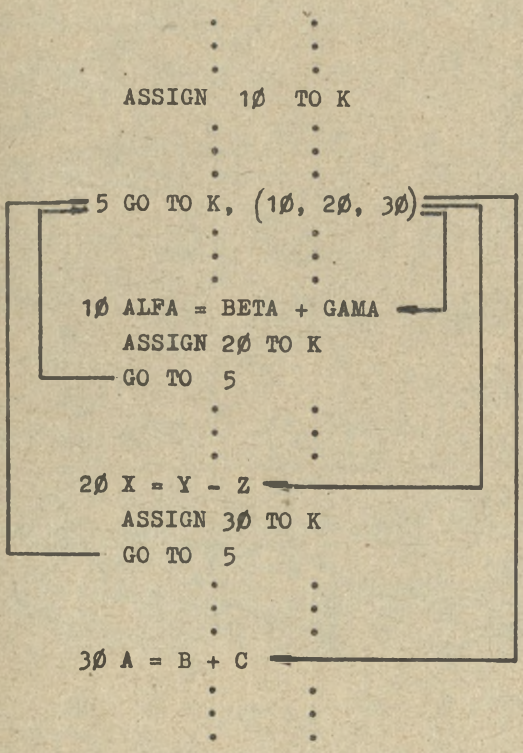
Uwagi:

1/ w obu instrukcjach /ASSIGN i GO TO i, .../ musi wystąpić zmienna o tej samej nazwie /KOS/;

2/ etykieta zawarta w instrukcji ASSIGN musi być jedną z etykiet występujących na liście etykiet instrukcji GO TO i, ...;

3/ do pewnej instrukcji GO TO i, ... może odnosić się kilka instrukcji ASSIGN.

Przykład



3.7.2. Instrukcje warunkowe /IF.../

Instrukcje warunkowe umożliwiają zmianę kolejności wykonywania programu w zależności od spełnienia pewnych warunków. Język FORTRAN przewiduje dwa rodzaje instrukcji warunkowej:

- instrukcję warunkową arytmetyczną /IF arytmetyczne/;
- instrukcję warunkową logiczną /IF logiczne/.

3.7.2.1. Instrukcja warunkowa arytmetyczna

Postać instrukcji

IF (wyrażenie arytmetyczne) k_1, k_2, k_3

gdzie: IF - jeśli

k_1, k_2, k_3 - etykiety instrukcji tego samego segmentu, w którym występuje instrukcja IF

Wynik wyrażenia arytmetycznego może być typu: INTEGER, REAL lub DOUBLE PRECISION.

Działanie instrukcji

W trakcie wykonywania instrukcji realizacja programu zostaje przeniesiona do instrukcji o etykiecie k_1, jeśli wartość wyrażenia arytmetycznego jest ujemna lub do instrukcji o etykiecie k_2 - gdy jest równa zeru, bądź do instrukcji o etykiecie k_3 - gdy jest dodatnia.

Przykład

```
      .          .
      .          .
15 X1 = (-B - SQRT(DELTA))/(2*A)
      IF (DELTA) 15, 2Ø, 1Ø
      .          .
      .          .
      .          .
2Ø X = -B/(2*A)
      .          .
      .          .
      .          .
1Ø X2 = (-B + SQRT(DELTA))/(2*A)
      .          .
```

3.7.2.2. Instrukcja warunkowa logiczna

Postać instrukcji

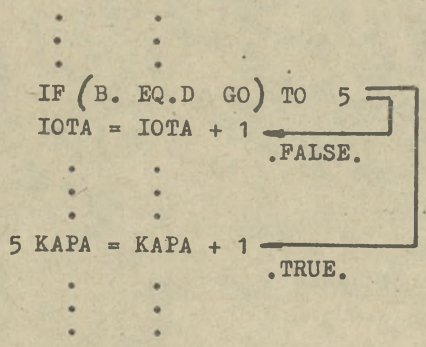
IF (wyrażenie logiczne) s

gdzie: s - jakakolwiek instrukcja z wyjątkiem instrukcji DO lub innej instrukcji IF logiczne

Działanie instrukcji

Jeśli wartością wyrażenia logicznego jest .TRUE., wówczas instrukcja "S" jest wykonywana, gdy natomiast wartością tego wyrażenia jest .FALSE., wtedy instrukcja "S" jest ignorowana i działanie programu przenosi się do następnej instrukcji po IF.

Przykład



3.7.3. Instrukcja powtórzeń /DO.../

Instrukcja powtórzeń DO zapewnia wielokrotne wykonywanie w programie określonej grupy instrukcji, czyli realizację pracy cyklicznej /w pętli/.

Postać instrukcji

DO n I = m_1, m_2, m_3

gdzie: DO - ma sens "wykonaj w pętli"

n - etykieta ostatniej instrukcji w pętli

i - zmienna kontrolna /typ INTEGER/

m_1 - wartość początkowa zmiennej kontrolnej /$m_1 > 0$ i typu INTEGER/

m_2 - wartość końcowa zmiennej kontrolnej /$m_2 > 0$ i typu INTEGER/

m_3 - parametr kroku /$m_3 > 0$ i typu INTEGER/

Działanie instrukcji

Instrukcja powoduje powtarzalne wykonywanie następującego po niej ciągu instrukcji aż do instrukcji o etykiecie "n" włącznie. Powtarzanie to odbywa się wg następującego schematu:

1° Zmiennej kontrolnej "i" zostaje przypisana wartość początkowa "m_1", która musi spełniać warunek:

$$m_1 \leqslant m_2;$$

jeśli $m_1 = m_2$, ciąg instrukcji będzie wykonywany tylko jeden raz;

2° Następuje wykonanie wszystkich instrukcji z zakresu pętli, tzn. od instrukcji bezpośredniej po DO do instrukcji o etykiecie "n";

3° Jeśli działanie dotrze do ostatniej instrukcji z zakresu DO, wówczas zmienna kontrolna jest powiększona o wartość parametru kroku, czyli o "m_3";

4° Jeśli nowa wartość zmiennej kontrolnej "i" jest niewiększa od wartości końcowej zmiennej kontrolnej "m_2", wtedy powtarzane są czynności od punktu 2; w przeciwnym razie praca w pętli zostaje zakończona i wykonywana jest następna instrukcja po ostatniej instrukcji z zakresu DO.

Uwagi:

1° Ostatnią instrukcją z zakresu DO nie mogą być instrukcje: GO TO, RETURN, STOP, PAUSE, DO, IF arytmetyczne bądź IF logiczne, zawierające dowolną z powyższych instrukcji. Jeśli jednak któraś z zastrzeżonych instrukcji musi być użyta jako ostatnia instrukcja w zakresie DO, to po niej musi wystąpić instrukcja CONTINUE.

2° Wyjście z zakresu instrukcji DO jest dopuszczalne i zmienna kontrolna "i" zachowuje swoją wartość.

3° Jeśli w instrukcji DO nie podaje się "m_3", oznacza to, że $m_3 = 1$.

4° Podanie $m_3 = 0$ wywołałoby pracę cykliczną, nie kończącą się.

Przykłady:

1/ Zaprogramować obliczenie wartości sumy:

$$\sum_{n=1}^{100} \frac{na^3 + b^2}{2c}$$

gdzie: a,b,c - stałe współczynniki.

Rozwiązanie:

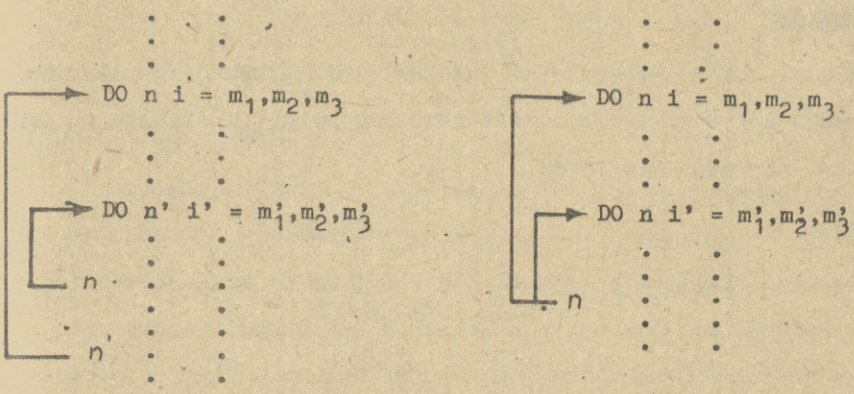
```
      .  .
      .  .
      .  .
      S = Ø
      DO 1Ø K = 1, 1ØØ, 1
1Ø    S = S+(N*A**3 + B**2)/(2*C)
      .  .
      .  .
      .  .
```

2/ Zaprogramować obliczanie sumy 100 kolejnych liczb naturalnych nieparzystych

Rozwiązanie:

```
      .  .
      .  .
      .  .
      SUMA = Ø
      LICZBA = 1
      DO 12 K = 1, 1ØØ, 1
      SUMA = SUMA + LICZBA
      LICZBA = LICZBA + 2
      .  .
      .  .
      .  .
```

Przy organizacji cykli złożonych należy przestrzegać zasady, że wewnątrz jednej pętli może wystąpić inna, ale musi być całkowicie zawarta w pętli zewnętrznej, przy czym może być wspólna tylko instrukcja końcowa dla obu pętli. Ilustrują to schematy a i b na rys. 38. Wzajemne przenikanie się pętli jest niedopuszczalne.



a/ b/

Rys. 38.

Język FORTRAN pozwala budować pętle wielostopniowe, o ile tylko nie wynikną ograniczenia ze strony pojemności PAO.

Przykład /na pętlę dwustopniową/

Utworzyć elementy tablicy X jako sumy odpowiednich elementów dwuwymiarowych, stuelementowych tablic Y i Z

```
      .  .
      .  .
      .  .
   DIMENSION  X(1Ø, 1Ø), Y(1Ø, 1Ø), Z(1Ø, 1Ø)
   DO 15 I = 1, 1Ø
   DO 15 I = 1, 1Ø
15 C(I,J) = A(I,J) + B(I,J)
      .  .
      .  .
      .  .
```

3.7.4. Instrukcje CONTINUE, PAUSE, STOP

Instrukcja CONTINUE jest instrukcją pustą, co oznacza, że nie wykonuje żadnej operacji, lecz poleca maszynie: "nic nie rób, przejdź dalej".

Postać instrukcji

```
CONTINUE
```

Zastosowanie

Instrukcja stosowana jest jako ostatnia w zakresie instrukcji DO wówczas, gdy w razie jej braku ostatnią byłaby któraś ze zbioru instrukcji niedopuszczalnych.

Przykład

Napisać fragment programu, który będzie przeglądał tablicę jednowymiarową, stuelementową o nazwie A tak długo, dopóki nie znajdzie elementu o wartości 25.5. Po znalezieniu takiego elementu działanie programu należy przenieść do instrukcji o etykiecie 50. Jeśli tablica nie zawiera elementu 25.5, należy kontynuować program dalej.

```
      .  .
      .  .
      .  .
      DO 1Ø I = 1, 1ØØ
      IF(25.5 - A(I)) 1Ø, 5Ø, 1Ø
1Ø    CONTINUE
            .  .
            .  .
5Ø          .  .
            .  .
            .  .
```

Instrukcja PAUSE może przyjąć jedną z dwóch postaci:

PAUSE	PAUSE n

gdzie: n - ciąg znaków od 1 do 5 cyfr ósemkowych.

Działanie instrukcji

Instrukcja powoduje chwilowe wstrzymanie realizacji programu w celu wykonania pewnych czynności manualnych, takich jak np. założenie nowego krążka taśmy. Instrukcja PAUSE umożliwia restart programu, czyli kontynuację jego realizacji. Ciąg znaków ósemkowych stosowany jest w celu oznaczenia w programie różnych instrukcji PAUSE.

Instrukcja STOP posiada również dwie postacie:

STOP	STOP n	n - jw.

Działanie instrukcji

Instrukcja powoduje zakończenie przebiegu programu i wymazanie go z PAO. Nie stwarza więc możliwości restartu.

3.8. Instrukcje wejścia/wyjścia

Dane między PAO, a urządzeniami zewnętrznymi, przekazywane są w postaci rekordów. W przypadku nośników perforowanych, pojedynczym rekordem jest ciąg znaków znajdujących się na jednej karcie perforowanej lub ciąg znaków na odcinku taśmy perforowanej, zakończony znakiem nowego wiersza. W przypadku tabulogramu drukarki rekord stanowi jeden wiersz wydruku.

Rekordy dzielą się na pola, czyli części przeznaczone na stałą, zmienną prostą, zmienną indeksowaną lub ciąg znaków tekstu. Rekordy wprowadzane są do PAO za pomocą instrukcji READ, a wyprowadzane - za pomocą instrukcji WRITE.

Podczas programowania odczytu danych wejściowych bądź wyprowadzania wyników należy określić postać zewnętrzną rekordów. W tym celu służy dyrektywa FORMAT.

Dane wejściowe mogą być wprowadzane do PAO za pomocą różnych urządzeń wejściowych. Podobnie wyniki z PAO można wyprowadzać za pomocą różnych urządzeń wyjściowych. Celem identyfikacji urządzeń zewnętrznych, wykorzystywanych przez dany program, programista nadaje im odpowiednie numery, umieszczane w instrukcjach READ i WRITE oraz w odpowiednich wierszach sterujących.

3.8.1. Instrukcja READ

Postać instrukcji

```
READ (u, f) k
```

gdzie: READ- oznacza CZYTAJ
u - numer programowy urządzenia wejściowego
f - numer etykiety dyrektywy FORMAT
k - lista wejścia

Lista wejścia k ma następującą postać:

$$k = e_1, e_2, \ldots, e_i, \ldots, e_n$$

gdzie: e_i - nazwa zmiennej prostej, tablicy, zmiennej indeksowanej lub DO implikowane.

Działanie instrukcji

Instrukcja powoduje wprowadzenie do PAO nowego rekordu z urządzenia o numerze "u" i nadanie wprowadzonym danym nazw zgodnie z listą wejścia. W pewnych przypadkach instrukcja READ może pominąć część danych zawartych w rekordzie zewnętrznym. Również w szczególnych przypadkach pojedyncza instrukcja READ może wczytać do PAO dane z dwu lub kilku rekordów.

3.8.2. Instrukcja WRITE

Postać instrukcji

```
WRITE (u, f) k
```

gdzie: WRITE - oznacza PISZ
u - numer programowy urządzenia wyjściowego
f - numer etykiety dyrektywy FORMAT
k - lista wyjścia

$$k = e_1, e_2, \ldots, e_i, \ldots e_n$$

gdzie: e_i - nazwa zmiennej prostej, tablicy, zmiennej indeksowanej lub DO implikowane.

Działanie instrukcji

. Instrukcja powoduje wyprowadzenie z PAO wartości poszczególnych elementów listy wyjścia na urządzenie wyjściowe o numerze "u".

3.8.3. Dyrektywa FORMAT

Dyrektywa FORMAT określa zewnętrzną postać rekordów wprowadzanych do PAO bądź wyprowadzanych z niej. Posiada ona następującą postać:

FORMAT (s)

gdzie: s - specyfikacja FORMATu, stanowiąca ciąg opisu pól, przecinków i ukośnych kresek.

Przykład

FORMAT(///3ØX, 15H KAT b WYCHYLENIA = , E14.7,5x)

3.8.3.1. Opisy pól

W języku FORTRAN stosowane są najczęściej następujące opisy pól:

I w
F w.d
E w.d
D w.d
w H $h_1 h_2 \ldots h_n$
w X

gdzie: I,F,E,H,X - kody konwersji
w - szerokość pola w rekordzie zewnętrznym
d - ilość cyfr w części ułamkowej liczby
$h_1 h_2 \ldots h_n$ - znaki ze zbioru znaków dostępnych w języku FORTRAN

3.8.3.2. Kody konwersji

Kod konwersji I używany jest do przesyłania wartości typu INTEGER.

W przypadku instrukcji READ opis pola Iw powoduje zamianę "w" znaków zewnętrznego rekordu na liczbę typu INTEGER, którą przyporządkowuje odpowiedniemu elementowi listy wejścia.

Konstruując opis pola, należy mieć na uwadze, że:

- postać zewnętrzna liczby może być ze znakiem lub bez znaku;
- liczby bez znaku traktowane są jako dodatnie;
- spacje przed pierwszą cyfrą są ignorowane, ale muszą być uwzględniane przy określaniu szerokości pola "w";
- spacje wewnątrz i na końcu pola liczby traktowane są jako zera;
- pole zawierające tylko spacje jest traktowane jako zero.

Przykład

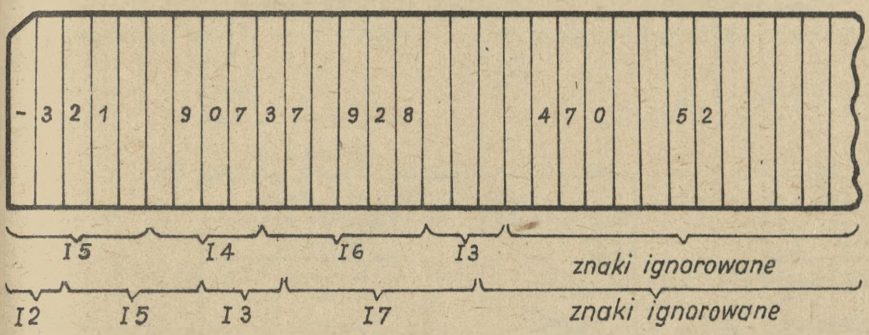
```
      .   .
      .   .
      .   .
   READ (1, 15) M2, K3, L52, J
      .   .
      .   .
      .   .
15 FORMAT (I5, I4, I6, I3)
      .   .
      .   .
      .   .
2Ø FORMAT (I2, I5, I3, I7)
      .   .
      .   .
      .   .
```

W przykładzie powyższym instrukcja READ poleca: czytaj z urządzenia zewnętrznego o numerze programowym 1, wg FORMATu o etykiecie 15, cztery liczby, z których pierwszy nadaj nazwę M2, drugiej - K3, trzeciej - L52, czwartej - J.

Z FORMATu o etykiecie 15 wynika, że liczba M2 zajmuje w rekordzie zewnętrznym 5 znaków, liczba K3 - 4 znaki, liczba L52 - 6 znaków, a liczba J - 3 znaki.

Jeśli zatem na karcie perforowanej będzie rozmieszczony ciąg znaków zgodnie z rys. 39, to do PAO zostaną wczytane następujące liczby:

M2 = -3210
K3 = +907
L52 = +370928
J = 0



Rys. 39.

Gdyby natomiast użyto w programie instrukcji: READ (1, 2Ø) M2, K3, L52, J, wówczas z tej samej karty perforowanej do PAO wczytane zostałyby następujące liczby:

M2 = -3
K3 = +21009
L52 = +073
J = +7092800

Podczas odczytu danych mogą zajść następujące przypadki:

1° Jeśli lista wejścia zostanie całkowicie wyczerpana, a specyfikacja FORMATu do końca zinterpretowana, wówczas pozostałe znaki w rekordzie zostaną zignorowane /nie zostaną wczytane do PAO/.

2° Jeśli dane w rekordzie skończą się, a lista wejścia nie zostanie całkowicie wyczerpana i specyfikacja FORMATu nie zostanie do końca zinterpretowana, wówczas w ramach tej samej instrukcji READ rozpocznie się odczyt danych z następnego rekordu /z następnej karty perforowanej/.

3° Jeśli specyfikacja FORMATu zostanie całkowicie zinterpretowana, a lista wejścia nie zostanie jeszcze wyczerpana, wówczas odczyt dalszych danych rozpocznie się wg tej samej specyfikacji od jej początku.

Powyższe trzy uwagi dotyczą także pozostałych kodów konwersji.

Przykłady /tabela nr 6/

Tabela nr 6

Opis pola zewnętrznego	Postać zewnętrzna liczby	Postać wewnętrzna liczby /w PAO/
I5	bb230	+230
I6	bbbb-5	-5
I4	92b7	+9207
I3	bbb	0

Jeśli w dyrektywie FORMAT wystąpi ta sama wielokrotna specyfikacja, wtedy można skorzystać z opisu pola o następującej postaci:

r I w

gdzie: r - ilość powtórzeń.

Przykład

```
46 FORMAT  I5, I5, I5
46 FORMAT  3I5
```

} zapisy równoważne

W przypadku użycia instrukcji WRITE opis pola Iw powoduje wyprowadzenie wartości odpowiedniego elementu listy wyjścia jako liczby typu INTEGER, zawierającej "w" pozycji w wewnętrznym zapisie, z uwzględnieniem znaku liczby ujemnej. Liczba nie wypełniająca całkowicie pola poprzedzona jest spacjami. Jeśli ilość znaków wyprowadzonej liczby przekracza ilość znaków przewidywanych specyfikacją, wówczas liczba poprzedzana jest znakiem gwiazdki i wyprowadzana ze wszystkimi pozycjami /może przy tym nastąpić naruszenie zaplanowanej postaci wyników/.

Przykłady /tabela nr 7/

Tabela nr 7

Opis pola zewnętrznego	Postać wewnętrzna liczby /w PAO/	Postać zewnętrzna liczby
I5	+3542	b3542
I3	+3542	*3542
I6	-23	bbb -23

Kod konwersji F używany jest do przesyłania wartości typu REAL /lub jednej ze składowych typu COMPLEX/.

Przy użyciu instrukcji READ opis pola Fw.d powoduje zamianię "w" znaków rekordu zewnętrznego na liczbę typu REAL z kropką dziesiętną. Liczba ta zostaje przyporządkowana odpowiedniemu elementowi listy wyjścia. Jeśli pole zewnętrzne zawiera kropkę dziesiętną, to będzie ona honorowana, natomiast kropka określana przez "d" zostanie pominięta. O ile liczba wprowadzana jest w postaci wykładniczej, wtedy jej cecha mpże przyjąć jedną z następujących postaci: ±d, Ed, E±d, gdzie d oznacza stałą całkowitą nie większą od 76.

Przy tym:

- spacje wewnątrz mantysy są traktowane jako zera;
- spacje wewnątrz cechy są ignorowane.

Przykłady /tabela nr 8/

Tabela nr 8

Opis pola zewnętrznego	Postać zewnętrzna liczby	Postać wewnętrzna liczby /w PAO/	U w a g i
F7.3	1234567	+1234.567	
E6.2	987E+3	+9870	
E9.5	bb3527E-1	+0.003527	
E8.2	-1.234E2	-123.4	zignorowanie kropki w opisie pola
E7.3	b-137-3	-.000137	

Przy użyciu instrukcji WRITE opis pola Fw.d powoduje wyprowadzenie wartości odpowiedniego elementu listy w postaci REAL z kropką dziesiętną, z zaokrągleniem do "d" miejsc po kropce i z uwzględnieniem znaku liczby ujemnej. Liczba nie wypełniająca pola poprzedzana jest spacjami. Jeśli liczba nie ma części całkowitej, to na jej początku zostanie dopisane zero. Gdy ilość wyprowadzanych znaków przekracza szerokość pola, wtedy liczba wyprowadzana jest ze wszystkimi znakami i zostaje poprzedzona znakiem gwiazdki.

Przykłady /tabela nr 9/

Tabela nr 9

Opis pola zewnętrznego	Postać wewnętrzna liczby	Postać zewnętrzna liczby	U w a g i
F10.4	+2534.3870	b2534.3870	
F9.2	-351.226	bb-351.23	zaokrąglenie
F8.3	+19.4	bb19.400	
F5.2	-3451.2387	-3451.24	

Kod konwersji E używany jest do przesyłania wartości typu REAL /lub jednej ze składowych typu COMPLEX/.

W przypadku instrukcji READ opis pola Ew.d wywołuje takie same efekty, jak kod konwersji F.

Natomiast podczas użycia instrukcji WRITE opis pola Ew.d powoduje wyprowadzenie wartości odpowiedniego elementu listy wyjścia w postaci wykładniczej z mantysą z przedziału $0.1 \leq m \leq 1$ zaokrągloną do "d" cyfr po przecinku i z odpowiednią cechą postaci Ebd_1d_2 lub $E-d_1d_2$, gdzie d_1d_2 są cyframi dziesiętnymi.

Przykłady /tabela nr 10/

Tabela nr 10

Opis pola zewnętrznego	Postać wewnętrzna liczby	Postać zewnętrzna liczby
E14.5	+12345678	bbb0.12346Eb08
E10.2	-.03	bb-0.3E-01
E11.4	+.000132	b0.1320E-03
E8.3	-12453	-0.125Eb05

Kod konwersji D służy do przesyłania wartości typu DOUBLE PRECISION i działa identycznie jak kod konwersji E /litera E w opisie pola jest zastąpiona literą D/.

Kod konwersji H określa sposób przesyłania ciągów znaków dopuszczalnych w języku FORTRAN.

Przy użyciu instrukcji READ, odnoszącej się do FORMATU zawierającego w swej specyfikacji opis pola $wHh_1h_2 \ldots h_n$, nastąpi wprowadzenie do PAO "w" znaków z rekordu zewnętrznego i zapamiętanie ich w specyfikacji FORMATU.

W przypadku użycia instrukcji WRITE opis pola $wHh_1h_2 \ldots h_n$ wywoła wyprowadzenie na zewnętrz "w" znaków $h_1 h_2 \ldots h_n$ jako części rekordu zewnętrznego.

Przykłady

1/

```
      .  .
      .  .
      .  .
    WRITE (1, 15)                    bez listy wejścia
      .  .
      .  .
      .  .
15 FORMAT (17Hbb WYNIKI b OBLICZEN)
      .  .
      .  .
      .  .
```

W rezultacie działania przytoczonej instrukcji WRITE nastąpi wydrukowanie od początku linii tekstu WYNIKI OBLICZEN, poprzedzonego dwiema spacjami.

2/
```
      .   .
      .   .
      .   .
    READ (1, 15)
15  FORMAT (4H DANE)
      .   .
      .   .
      .   .
```

Jeśli przytoczony fragment programu będzie znajdował się w PAO, a aktualnie wczytywany rekord będzie zawierał tekst WYNIKI, to instrukcja READ (1,15) spowoduje umieszczenie w specyfikacji FORMATu o etykiecie 15 napisu WYNIKI w miejsce napisu DANE. Użycie z kolei instrukcji WRITE (2, 15) spowodowałoby wydrukowanie tekstu WYNIKI.

Kod konwersji X znajduje zastosowanie do opisu pól złożonych z samych spacji.

W przypadku stosowania instrukcji READ opis pola w X powoduje zignorowanie "w" kolejnych znaków rekordu wprowadzanego.

Przykład

```
      .   .
      .   .
      .   .
    READ (1, 15) K, L, M
15  FORMAT (5x, I6, 1Øx, I3, 5x, I4)
      .   .
      .   .
      .   .
```

Podany fragment programu spowoduje wczytanie do PAO wartości trzech zmiennych typu INTEGER, którym zostaną nadane nazwy X,Y,Z, z tym, że zostaną pominięte w odczycie następujące znaki wprowadzanego rekordu:

od 1 do 5 /włącznie/

od 12 do 21 /włącznie/
od 25 do 29 /włącznie/

Przy użyciu instrukcji WRITE opis pola wX wywoła wyprowadzenie "w" spacji.

Przykład

```
      .  .
      .  .
      .  .
      WRITE (2, 3Ø)
   30 FORMAT (1ØX, 6H WYNIKI, 3X, OBLICZEN)
      .  .
      .  .
      .  .
```

Przytoczony fragment programu, o ile urządzeniem nr 2 będzie drukarka, spowoduje wydruk tekstu WYNIKI OBLICZEN, który poprzedzony zostanie od początku wiersza dziesięcioma spacjami. Oba wyrazy tekstu rozdzielone będą trzema spacjami.

Posługując się zatem opisem pola wX, można wyprowadzoną treść rozmieszczać dowolnie w wierszu.

3.8.3.3. Ograniczniki pól i rekordów

Opisy pól w specyfikacji FORMATu mogą być oddzielone przecinkiem, ukośną kreską bądź ciągiem ukośnych kresek. Przecinek oznacza koniec pola, a ukośna kreska - koniec pola i koniec rekordu. Ukośne kreski mogą występować między opisami pól, mogą poprzedzać ciąg opisów pól lub kończyć go.

Ciąg "n" ukośnych kresek między polami powoduje:

- na wejściu - zignorowanie /n-1/ wprowadzanych rekordów i zainicjowanie przesłania nowego rekordu;
- na wyjściu - wyprowadzenie /n-1/ rekordów pustych i zainicjowanie wyprowadzenia nowego rekordu.

Ciąg "n" ukośnych kresek umieszczonych na początku lub końcu specyfikacji FORMATu wywołuje:

- na wejściu - zignorowanie "n" rekordów;
- na wyjściu - wyprowadzenie "n" rekordów pustych

Przykłady

1/

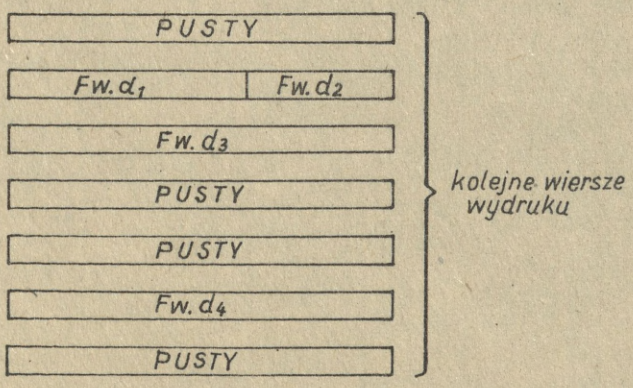
```
      .  .
      .  .
      .  .
   READ (1, 5) A, B, C, D, E
5  FORMAT (Fw.d1, Fw.d2, Fw.d3////Fw.dn, Fw.d5)
      .  .
      .  .
      .  .
```

Przytoczony fragment programu spowoduje odczyt trzech rekordów, zawierających liczby typu REAL, wyrażone w kodzie konwersji F. Po odczytaniu dwóch pierwszych rekordów trzy rekordy zostaną pominięte /4 ukośne kreski/, a z kolei nastąpi odczyt rekordu trzeciego.

2/

```
      .  .
      .  .
      .  .
   WRITE (1, 1Ø) P, R, S, T
1Ø FORMAT  /Fw.d1, Fw.d2/Fw.d3///Fw.d4
      .  .
      .  .
      .  .
```

Wynik działania przytoczonego fragmentu programu ilustruje rys. 40.



Rys. 40.

Za pomocą ukośnych kresek, umieszczanych w specyfikacji FORMATu, można więc regulować odstępy między wierszami wydruku.

Opisy pola wX oraz ukośne kreski w specyfikacji FORMATu pozwalają programiście projektować różne postacie wyprowadzanych wyników /tabele, zestawienia itp./.

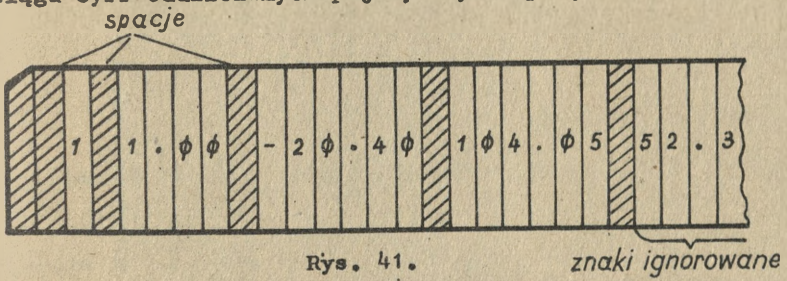
3.8.4. Formaty swobodne

Formaty swobodne dotyczą tylko trzech z omówionych dotychczas opisów pól: Iw, Fw.d oraz Ew.d i tylko instrukcji READ.

Jeśli "w" oraz "d" w przytoczonych opisach są zerami, wówczas ogranicznikiem pola na rekordach wejściowych jest spacja lub koniec rekordu. W takim przypadku spacje występujące przed polem numerycznym są ignorowane, a spacje wewnątrz liczby są niedopuszczalne.

Przykład

Rys. 41 przedstawia rozmieszczenie na karcie perforowanej ciągu cyfr oddzielanych pojedynczymi spacjami.



Rys. 41.

```
      .  .
      .  .
      .  .
   READ (1, 25) L, A, B, C
15 FORMAT (IØ, 3 FØ.Ø)
      .  .
      .  .
      .  .
```

W wyniku działania instrukcji READ nastąpi przeniesienie z karty perforowanej do PAO następujących danych:

L = 1
A = 1.00
B = 20.40
C = +104.C5

3.8.5. Sterowanie wydrukiem

Pierwszy znak rekordu wydruku spełnia funkcję sterującą zgodnie z tabelą nr 11.

Tabela nr 11

Znak sterujący	Działanie
spacja	przesunięcie o jeden wiersz
Ø	przesunięcie o dwa wiersze
1	przesunięcie do początku nowej strony
+	brak przesunięcia
2 do 7	przesunięcie do ścieżki od 2 do 7

Przykład

```
15 FORMAT ( 1H1, I5, 1Øx, F 12.2 )
```

Opis pola 1H1 spowoduje wydruk wyników od nowej strony tabulogramu.

3.8.6. DO implikowane

DO implikowane /ukryte/ jest instrukcją stosowaną przy odczycie lub zapisie całych tablic bądź ich elementów.

Postać DO implikowanego:

$$e_1, e_2, \ldots, e_n, \quad i = m_1, m_2, m_3$$

gdzie: $e_1, e_2, \ldots, e_n$ - zmienna prosta, zmienna indeksowana, tablica

i - zmienna kontrolna

m_1 - wartość początkowa zmiennej kontrolnej

m_2 - wartość końcowa zmiennej kontrolnej

m_3 - parametr kroku /może być pominięty, gdy $m_3 = 1$/

$$m_1, m_2, m_3 \geq 0; \quad m_2 \geq 1$$

Przykład

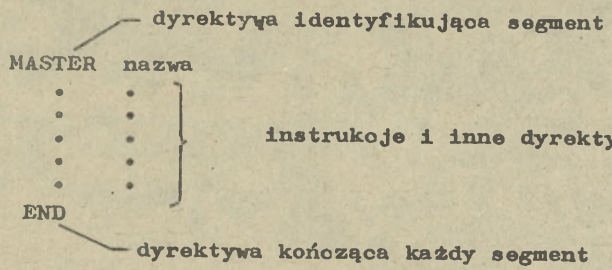
```
      .  .
      .  .
      .  .
   READ (1, 15) A(1), A(2), ..., A(1ØØ)
15 FORMAT (1ØØ FØ.Ø)
      .  .
      .  .
      .  .
```

Przytoczony fragment programu, powodujący odczyt stuelementowej tablicy jednowymiarowej, można zastąpić prostszym zapisem za pomocą DO implikowanego:

```
      .  .
      .  .
      .  .
   READ (1, 15) (A(I), I = 1, 1ØØ, 1)
15 FORMAT (1ØØ FØ.Ø)
      .  .
      .  .
      .  .
```

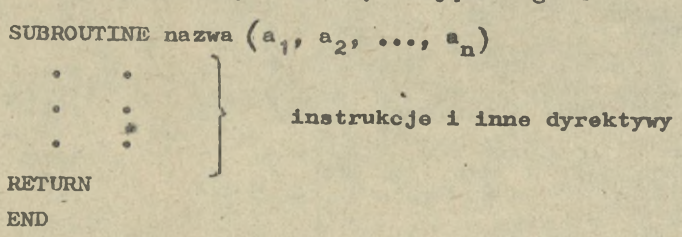
3.9. Struktura segmentów

Segment MASTER, jak już mieliśmy okazję stwierdzić, jest głównym segmentem programu, sterującym jego wykonaniem. Musi on wystąpić w każdym programie, przy czym program może składać się tylko z segmentu MASTER. Segmentowi MASTER, podobnie jak i innym segmentom /oprócz segmentu BLOCK DATA/, nadawana jest przez programistę dowolna nazwa, zgodnie z zasadami tworzenia nazw. Posiada on następującą postać:



Segment MASTER realizuje obliczenia, wprowadzenie danych, wyprowadzanie wyników oraz wywoływanie segmentów SUBROUTINE lub FUNCTION.

Segment SUBROUTINE, czyli segment podprogramu, stosowany jest w programie w miarę potrzeby. Może więc w nim w ogóle nie wystąpić, ale program może także zawierać kilka takich segmentów. Segment SUBROUTINE może posiadać strukturę z parametrami lub bez parametrów.



gdzie: $a_1, a_2, \ldots, a_n$ - parametry formalne

Nazwa segmentu może być dowolna i nadawana jest przez programistę zgodnie z obowiązującymi zasadami.

Segment SUBROUTINE może być wywoływany w dowolnym innym segmencie, z wyjątkiem segmentu BLOCK DATA. Wywoływanie takie odbywa się za pomocą instrukcji CALL.

Postać instrukcji CALL

CALL S $(a_1, a_2, \ldots, a_n)$

gdzie: CALL ma sens "skocz do"
S - nazwa wywoływanego segmentu SUBROUTINE
$a_1, a_2, \ldots, a_n$ - parametry aktualne

Postać instrukcji:

```
CALL S
```

używana jest wtedy, gdy podprogram nie zawiera parametrów formalnych.

Działanie instrukcji

Instrukcja powoduje skok do segmentu SUBROUTINE o podanej nazwie i wykonanie jego kolejnych instrukcji aż do napotkania instrukcji RETURN, która powoduje powrót do segmentu wywołującego podprogram.

Postać instrukcji RETURN

```
RETURN
```

gdzie: RETURN - ma sens "wróć"

Działanie instrukcji

Instrukcja RETURN, występująca także w segmencie FUNCTION, powoduje - po wykonaniu segmentu wywoływanego - powrót do segmentu wywołującego, przy czym z segmentu SUBROUTINE powrót ten ma miejsce do instrukcji następnej po CALL, natomiast powrót z segmentu FUNCTION ma miejsce do wyrażenia wywołującego ten segment.

W podprogramie może być użytych kilka instrukcji RETURN.

Segment FUNCTION stosowany jest w programie wówczas, gdy zachodzi potrzeba częstego używania pewnej funkcji. Posiada on następującą postać:

dyrektywa identyfikująca segment

typ FUNCTION nazwa $(a_1, a_2, \ldots, a_n)$

. .
. . } instrukcje i inne dyrektywy
. .

RETURN
END

gdzie: $a_1, a_2, \ldots, a_n$ - parametry formalne

Typu funkcji /INTEGER, REAL itp./ nie podaje się, jeśli jest on określony przez pierwszą literę nazwy funkcji.

Segment FUNCTION musi posiadać przynajmniej jeden parametr formalny, a poza tym nazwie segmentu trzeba przynajmniej raz nadać wartość w treści segmentu. W programie może wystąpić kilka segmentów FUNCTION.

Segment FUNCTION może być wywoływany w dowolnym innym segmencie /oprócz segmentu BLOCK DATA/, przy czym wywołanie to ma następującą postać:

"nazwa segmentu funkcji" $(a_1, a_2, \ldots, a_n)$

gdzie: $a_1, a_2, \ldots, a_n$ - parametry aktualne.

Powrót do segmentu wywołującego ma miejsce po napotkaniu w segmencie FUNCTION instrukcji RETURN. Powrót następuje do instrukcji zawierającej wywołanie segmentu FUNCTION.

Podobnie jak w segmencie SUBROUTINE, również w segmencie FUNCTION instrukcja RETURN może być użyta wielokrotnie.

Każdy segment programu w języku FORTRAN jest autonomiczny i dlatego zmienne występujące w danym segmencie mają znaczenie i wartość określoną tylko dla tego segmentu i nie mogą być zamieniane na zewnątrz segmentu.

Przekazywanie wartości między segmentami wywołującymi i wywoływanymi może być organizowane dwoma sposobami, a mianowicie:

- poprzez rezerwację /podczas kompilacji programu/ wspólnego obszaru pamięci, dostępnego dla wymienionych segmentów;
- za pośrednictwem parametrów.

Wspólny obszar pamięci rezerwowany jest za pomocą dyrektywy COMMON.

Dyrektywa COMMON wyodrębnia pewne obszary PAO jako wspólne bloki, które udostępnia dla dwu lub kilku segmentów programu. Dzięki temu wartości uzyskane w jednym segmencie mogą być użyte w innym. Wspólnym blokom programista nadaje nazwy, które w dyrektywie COMMON zawarte są między ukośnymi kreskami. Jeden wspólny blok może nie mieć nazwy. Jest on nazywany blokiem pustym i oznaczany w dyrektywie dwiema ukośnymi kreskami //. Jeśli taki blok występuje w dyrektywie jako pierwszy, to kreski ukośne można pominąć.

Postać dyrektywy

COMMON/X1/a_1/X2/a_2 .../X_n/a_n

gdzie: X1,X2,...,X_n - nazwy wspólnych bloków
a_1,a_2,...,a_n - lista nazw zmiennych lub tablic zawartych we wspólnych blokach

Przykłady:

1/ COMMON // ALFA, BETA
COMMON ALFA, BETA
} zapisy równoważne

Obie dyrektywy rezerwują wspólny blok pusty do przechowywania zmiennych ALFA i BETA.

2/ Jeżeli w segmencie MASTER użyjemy dyrektywy:

COMMON WYROZNIK,

a w segmencie SUBROUTINE dyrektywy:

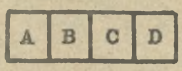
COMMON DELTA,

to w tym samym miejscu PAO - pustym bloku, będzie przechowywana ta sama wielkość, która w różnych segmentach jest różnie nazwana, /w jednym - WYROZNIK, w drugim - COMMON/.

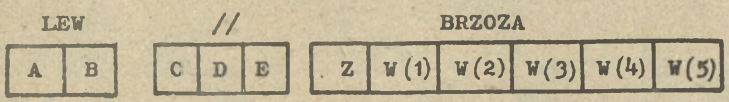
3/ COMMON/AKACJA/A,B,C,D

Z przytoczonego przykładu wynika, że we wspólnym bloku o nazwie AKACJA przechowywane są 4 zmienne o nazwach: A,B,C i D. Zmienne te rozmieszczone są w wymienionym bloku w takiej kolejności, w jakiej występują w dyrektywie:

AKACJA



4/ COMMON/LEW/A,B//C,D,E/BRZOZA Z, W(5)



W bloku BRZOZA przechowywana jest zmienna prosta Z oraz tablica jednowymiarowa, pięcioelementowa o nazwie W.

Jeśli np. w segmencie MASTER wystąpi dyrektywa:

COMMON//ALFA(2), B,GAMA(3)/ZEFIR/D,EWA(2,2)/ALA/F,G(2),

a w segmencie SUBROUTINE dyrektywa:

COMMON//T(5), X/ALA/ZOFIA(3),

natomiast w segmencie FUNCTION dyrektywa:

COMMON//SOWA(6)/ZEFIR/WIKTORIA(5),

to w obszarze pamięci wspólnej zostaną utworzone trzy bloki, a w nich wystąpią następujące przyporządkowania:

BLOK //

S.MASTER	S.SUBROUTINE	S.FUNCTION
ALFA 1	T 1	SOWA 1
ALFA 2	T 2	SOWA 2
B	T 3	SOWA 3
GAMA 1	T 4	SOWA 4
GAMA 2	T 5	SOWA 5
GAMA 3	X	SOWA 6

BLOK ZEFIR

S.MASTER	S.FUNCTION
D	WIKTORIA 1
EWA 1,1	WIKTORIA 2
EWA 2,1	WITKORIA 3
EWA 1,2	WIKTORIA 4
EWA 2,2	WIKTORIA 5

BLOK ALA

S.MASTER	S.SUBROUTINE
F	ZOFIA 1
G 1	ZOFIA 2
G 2	ZOFIA 3

Wielkości INTEGER, REAL i LOGICAL zajmują we wspólnym bloku po jednej jednostce PAO, natomiast wielkości typu DOUBLE PRECISION i COMPLEX - po dwie jednostki.

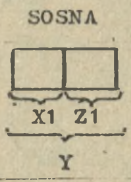
Jeśli w systemie MASTER wystąpią:

```
REAL X1, Z1
COMMON /SOSNA/ X1, X2,
```

a w segmencie pomocniczym:

```
COMPLEX Y
COMPLEX /SOSNA/ Y,
```

to części rzeczywistej zmiennej zespolonej Y zostanie przyporządkowana zmienna rzeczywista X1, zaś części urojonej zmiennej zespolonej Y - zmienna rzeczywista Z1.



W celu wyjaśnienia zasad użycia dyrektywy COMMON posłużymy się przykładem.

Przykład

Napisać program wczytania trzech danych typu REAL A, B, C, przemnożenia ich przez siebie i wydrukowania rezultatu pod nazwą WYNIK.

Program złożony tylko z segmentu MASTER mógłby mieć następującą postać:

```
      MASTER P1
      READ (1,5) A,B,C
    5 FORMAT (3FØ.Ø)
      WYNIK = A*B*C
      WRITE( 2,1Ø) WYNIK
   1Ø FORMAT (1Ø X,' F8.2)
      STOP
      END
```

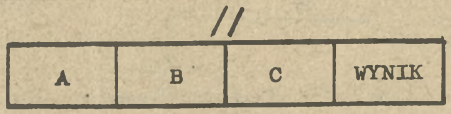
To samo zadanie można by zaprogramować przy użyciu trzech segmentów SUBROUTINE oraz segmentu MASTER, sterującego wykonaniem programu.

```
      MASTER P1
      COMMON A,B,C, ILOCZYN
      CALL CZYTANIE
      CALL LICZENIE
      CALL DRUKOWANIE
      STOP
      END

      SUBROUTINE CZYTANIE
      COMMON A,B,C, WYNIK
      READ (1,5) A,B,C
    5 FORMAT (3FØ.Ø)
      RETURN
      END
```

```
      SUBROUTINE LICZENIE
      COMMON A,B,C, WYNIK
      WYNIK = A*B*C
      RETURN
      END

      SUBROUTINE DRUKOWANIE
      COMMON A,B,C, WYNIK
      WRITE (2, 1Ø) WYNIK
   1Ø FORMAT (1ØX, F8.2)
      RETURN
      END
```

Pozornie wydaje się, że program byłby poprawny bez dyrektyw COMMON. Tak jednak nie jest, ponieważ wszystkie segmenty SUBROUTINE są niezależne i te same zmienne: A,B,C, WYNIK dla różnych podprogramów otrzymałyby różne miejsca w PAO i współpraca między segmentami byłaby niemożliwa.

Pozornie może się również wydawać, że w segmencie SUBROUTINE DRUKOWANIE dyrektywa COMMON zawiera zbędne nazwy A,B,C, skoro te zmienne nie są drukowane. Trzeba jednak wziąć pod uwagę, że zmienne we wspólnym bloku rozmieszczone są następująco:



Gdybyśmy zatem w segmencie SUBROUTINE DRUKOWANIE użyli dyrektywy COMMON WYNIK, to powstałby jednak błąd, ponieważ w segmencie tym zmiennej WYNIK odpowiadałaby wartość zmiennej A bloku pustego i taka zostałaby wydrukowana.

Organizację przekazywania wartości między segmentami za pośrednictwem parametrów wyjaśnimy również na przykładzie.

Przykład

Napisać program analizujący trójmiany kwadratowe ax^2+bx+c i wydający o nich następujące informacje:

- jeśli a = 0 i b = 0, należy zmiennej I nadać wartość równą zeru,
- jeśli a = 0, b ≠ 0, należy zmiennej I nadać wartość równą 1,
- jeśli a ≠ 0 oraz:

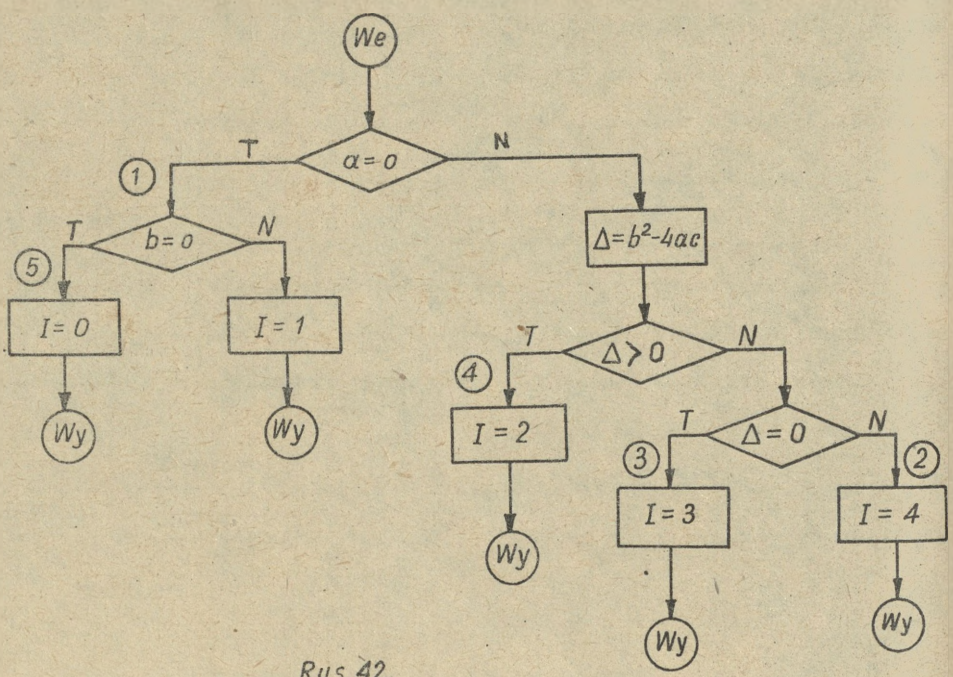
$\Delta >$ 0, należy zmiennej I nadać wartość równą 2

$\Delta =$ 0, należy zmiennej I nadać wartość równą 3

$\Delta <$ 0, należy zmiennej I nadać wartość równą 4

Rozwiązanie zadania zaprogramujemy za pomocą podprogramu o nazwie ANALIZA TK. W segmencie MASTER uwidocznimy tylko te fragmenty, które mają istotny wpływ na współpracę między segmentami.

Schemat blokowy działania podprogramu przyjmie postać /rys. 42/:



Rys. 42.

```
      MASTER ADAM
         .  .              parametry aktualne
         .  .                  /  /  \  \
         .  .
      CALL ANALIZA TK (A1, B1, C1 W)
      WRITE (1,5) A1, B1, C1, W
    5 FORMAT  . . .
         .  .
         .  .
         .  .
      END
                                  parametry formalne
                                    /  /  \  \
      SUBROUTINE ANALIZA TK  A, B, C, I
      IF(A.EQ.Ø) GO TO 1
      DELTA = B*B -4.Ø*A*C
      IF DELTA 2,3,4
    1 IF (B.EQ.Ø) GO TO 5
      I = 1
      RETURN
    2 I = 4
      RETURN
    3 I = 3                  wielokrotne użycie
      RETURN                 instrukcji RETURN
    4 I = 2
      RETURN
    5 I = 0
      RETURN
      END
```

Ponieważ w systemie MASTER"W"jest parametrem aktualnym,odpowiadającym formalnemu parametrowi I, który z kolei odpowiada wynikowi, więc wartość wyniku uzyskaną w segmencie SUBROUTINE można wykorzystać w segmencie wywołującym MASTER, używając zmiennej"W".

Segment BLOCK DATA

Przy omawianiu tego segmentu warto dokonać zestawienia możliwych sposobów wprowadzania danych do programu. Otóż sposoby te są następujące:

- za pomocą instrukcji READ;

- za pomocą instrukcji podstawiania;
- za pomocą dyrektywy DATA, użytej w segmencie różnym niż segment BLOCK DATA;
- za pomocą segmentu BLOCK DATA.

Dwa pierwsze sposoby są nam już znane.

Dyrektywa DATA używana jest do nadawania wartości początkowych zmiennym prostym, zmiennym indeksowanym i tablicom.

Postać instrukcji

DATA $R_1/d_1/$, $R_2/d_2/$, ..., $R_n/d_n/$

gdzie: $R_1, R_2, \ldots, R_n$ - lista zmiennych prostych, zmiennych indeksowanych lub tablic

$d_1, d_2, \ldots, d_n$ - lista wartości początkowych dla zmiennych prostych, zmiennych indeksowanych, tablic

Wartości początkowe $d_1, d_2, \ldots, d_n$ mogą przyjmować jedną z dwóch postaci: j lub r*j, gdzie "j" jest dowolną stałą, natomiast "r" jest stałą całkowitą bez znaku. Symbol "r*j" zastępuje r-krotne powtórzenie j.

Przykład

```
1/ DATA A, B(1), C / 2*1.Ø, 3.5/
```

lista zmiennych — lista wartości początkowych

Dyrektywa nada następujące wartości początkowe zmiennym:

```
A = 1.0
B(1) = 1.0
C = 3.5
```

```
2/   .  .
     .  .
     .  .
     DIMENSION A(5, 2)
     DATA A B / 5*-1.Ø, 5*1.Ø, .25/
       .  .
       .  .
       .  .
```

Zmiennym indeksowanym tablicy dwuwymiarowej, dziesięcioelemen-
towej o nazwie A oraz zmiennej B dyrektywa DATA nada następujące
wartości początkowe:

```
A (1,1) = -1
A (2,1) = -1
A (3,1) = -1
A (4,1) = -1
A (5,1) = -1
A (1,2) = 1
A (2,2) = 1
A (3,2) = 1
A (4,2) = 1
A (5,2) = 1
B = 0.25
```

Segment BLOCK DATA używany jest do nadawania wartości począt-
kowych zmiennym i tablicom należącym do wspólnego bloku /oprócz
bloku pustego/.

W segmencie BLOCK DATA mogą występować tylko dyrektywy.
Ma on postać:

```
BLOCK DATA
  .  .
  .  .   } inne dyrektywy
  .  .
END
```

Przykład

```
BLOCK DATA
INTEGER A,B,C
COMPLEX D,E
COMMON/ALFA/A,B,C/BETA/D,E
DATA A,B,C/1,1Ø,25/, D,E/2*(1.5, -2.5)
```

W języku FORTRAN obowiązuje pewna kolejność, w jakiej mogą
wystąpić dyrektywy i instrukcje w segmentach. Biorąc pod uwagę

tylko te dyrektywy, z którymi zapoznaliśmy się, jest ona następująca:

```
SUBROUTINE lub FUNCTION
Deklaracje typu: DIMENSION
   -"-       COMMON
   -"-       DATA
Definicje funkcji lokalnych
Treść segmentów
END

BLOCK DATA
Deklaracje typu: DIMENSION
   -"-       COMMON
   -"-       DATA
END
```

3.10. Kompilatory i wiersze sterujące

Tłumaczenie programu źródłowego na program wynikowy odbywa się w dwóch etapach. Pierwszy z nich nazywany jest kompilacją, drugi - konsolidacją.

W trakcie kompilacji program źródłowy z dowolnego języka algorytmicznego tłumaczony jest na tzw. postać półskompilowaną, która jest wspólna dla wszystkich języków. Kompilacja realizowana jest za pomocą specjalnego programu, zwanego kompilatorem. Każdy język algorytmiczny posiada swój kompilator lub nawet kilka kompilatorów.

W trakcie konsolidacji następuje tłumaczenie programu z postaci półskompilowanej na postać wynikową, czyli postać wyrażoną w języku wewnętrznym danego komputera. Etap konsolidacji realizuje specjalny program, zwany konsolidatorem, który jest niezależny od języka zewnętrznego. Dlatego też półskompilowane segmenty programu,

przetłumaczone z różnych języków zewnętrznych przez ich kompliatory, można na etapie konsolidacji razem łączyć.

3.10.1. Kompilatory

Język FORTRAN wyposażony jest w kilka kompilatorów. Przy pracy w systemie z taśmą magnetyczną stosowany jest kompilator noszący nazwę # XFAM. W systemie z dyskami magnetycznymi znajdują zastosowanie dwa kompilatory # XFAT oraz # XFAE, natomiast w systemie z taśmą perforowaną - kompilator # XFAP.

Rozpatrzymy minimalne wymagania kompilatora # XFAM:

- zajętość PAO - 12032 słowa
- jedno urządzenie listujące, którym może być drukarka wierszowa lub dziurkarka taśmy
- jedno urządzenie wejściowe w postaci czytnika kart perforowanych lub czytnika taśmy perforowanej
- dwie taśmy magnetyczne, z których jedną przeznacza się na kompilator, konsolidator oraz grupę programów o nazwie SRF7, wykorzystywaną przez kompilator, a drugą - na przechowywanie segmentów w postaci półskompilowanej.

Grupa SRF7 zawiera programy: funkcji standardowych, czytania, pisania oraz ujawniania błędów.

3.10.2. Wiersze sterujące

Wiersze sterujące dostarczają kompilatorowi informacji niezbędnych do wytworzenia programu wynikowego. Rozróżniamy następujące wiersze sterujące:

- wiersze sterujące wstępne;
- wiersze sterujące opisu programu;
- wiersze sterujące międzysegmentowe;
- ogranicznik FINISH

Wiersze sterujące wstępne i opisu programu łączy się w jeden segment i umieszcza na początku programu. Wiersze sterujące zapisywane są na arkuszu programowym w kolumnach od 7 do 72.

3.10.2.1. Wiersze sterujące wstępne

Wiersze sterujące wstępne określają stopień szczegółowości listowania programu źródłowego, wyprowadzanego podczas jego kompilacji na drukarkę wierszową /LP/ lub dziurkarkę taśmy /TP/. Możliwe są przy tym trzy przypadki:

- listowanie pełne;
- listowanie krótkie;
- pominięcie listowania.

Listowanie pełne może być zrealizowane tylko na drukarce wierszowej i w przypadku użycia wiersza sterującego:

LIST

List pełny zawiera takie informacje dotyczące kompilowanego programu, jak:

- nazwę kompilatora;
- datę i dokładny czas kompilacji (h/min/s);
- wiersze sterujące;
- dyrektywy i instrukcje poszczególnych segmentów programu;
- komunikaty o błędach formalnych.

Na końcu każdego segmentu podawane są:

- długość segmentu /ilości wygenerowanych instrukcji w programie wynikowym/;
- informacja, czy segment zawiera błędy formalne.

Listowanie krótkie obejmuje tylko nazwy segmentów programu i ich wymiary. Wynik listowania wyprowadzany jest na drukarkę wierszową, jeśli programista użyje wiersz:

SHORT LIST

Taki sam efekt będzie miał miejsce, gdy programista nie użyje żadnego wiersza sterującego.

Natomiast użycie wiersza:

```
NO LIST
```

spowoduje, że listowanie będzie pominięte i podczas kompilacji nie zostanie wyprowadzona żadna informacja dotycząca programu źródłowego.

3.10.2.2. Wiersze sterujące opisu programu

Wiersze sterujące opisu programu zawierają informacje, które dotyczą programu wynikowego. Są to informacje tego rodzaju, jak:

[redacted]

MT - stacja taśmy magnetycznej
ED - stacja pamięci dyskowej

Numer logiczny urządzenia wejściowego jest liczbą całkowitą z przedziału 0 ÷ 15, przy czym numer logiczny 7 zapewnia szybszą transmisję informacji między urządzeniem wejściowym a PAO.

Do sterowania wyjściem stosowany jest wiersz:

OUTPUT k = z

gdzie: k i z mają taki sam sens, jak w przypadku wejścia, tylko dotyczą urządzenia wyjściowego.

Urządzenia wyjściowe koduje się za pomocą skrótów:

TP - dziurkarka taśmy
LP - drukarka wierszowa
MT - stacja taśmy magnetycznej
ED - stacja pamięci dyskowej

Przykłady:

INPUT 1 = CRØ
INPUT 2 = TR 7
OUTPUT 1 = LPØ
OUTPUT 3 = TP7

Podczas testowania i wykonywania programu używane są programy śledzące i monitujące, czyli wykrywające błędy wykonania i wyprowadzające o nich odpowiednie komunikaty. Można przy tym stosować 3 poziomy śledzenia:

- poziom 0;
- poziom 1;
- poziom 2.

Poziom 0 stosowany jest w przypadku programów starannie przygotowanych i dobrze sprawdzonych. Zapewniany jest on poprzez użycie w programie wierszy:

TRACE Ø lub NO TRACE

Poziom 1 stosowany jest najczęściej i zapewniany za pomocą wiersza:

```
TRACE 1
```

Ten sam efekt można uzyskać nie stosując żadnego wiersza sterującego poziomu śledzenia.

Poziom 2 stosowany jest podczas wstępnego uruchamiania programów i zapewnia go użycie wierszy sterujących:

```
TRACE 2 lub TRACE
```

Należy zdać sobie sprawę z tego, że w miarę wzrostu poziomu śledzenia rośnie również czas kompilacji i przebiegu programu, jak również objętość obszaru PAO zajęta przez program.

3.10.2.3. Wiersze sterujące międzysegmentowe

Podstawowym wierszem sterującym międzysegmentowym jest wiersz:

```
READ FROM (z)
```

gdzie: z - nazwa urządzenia wejścia /TR, CR, MT, ED/.

Wiersz ten stosowany jest wtedy, gdy poszczególne segmenty programu zakodowane są na różnych nośnikach informacji. W takich przypadkach wyboru pierwszego nośnika, z którego rozpoczyna się odczyt programu, dokonuje operator. Natomiast napotkanie w programie wiersza READ FROM spowoduje, że następna część programu odczytywana będzie z urządzenia określonego przez ten wiersz.

Ogranicznik FINISH jest ostatnim wierszem sterującym, umieszczonym przed segmentem BLOCK DATA. Jego zadaniem jest sygnalizacja o końcu programu. Ma on postać:

```
FINISH
```

Przykład

Segment wierszy sterujących

```
LIST                - wydruk pełnego listu
PROGRAM (BETA)-     program wynikowy ma nazwę BETA
INPUT 2 = TRØ -     nr programowy urządzenia 1 określony jest
                    jako czytnik kart 0
OUTPUT 1 = LPØ -    nr programowy urządzenia 2 określony jest
                    jako drukarka wierszowa 0
TRACE 2             - do programu wynikowego należy włączyć
                    poziom śledzenia 2
END
Segmenty
FINISH
```

Przykład kompletnego programu zawierającego niezbędne wiersze sterujące pokazany jest na rys. 32.

3.11. Komunikaty błędów

Automatyczne wykrywanie błędów programu odbywa się w dwóch etapach:

- podczas kompilacji programu źródłowego
- podczas testowania i wykonywania programu.

3.11.1. Wykrywanie błędów w czasie kompilacji

W trakcie kompilacji programu źródłowego wykrywane są błędy formalne, czyli niezgodności z zasadami programowania w języku FORTRAN. Błędy formalne wynikają z winy programisty bądź operatora urządzeń przygotowania maszynowych nośników informacji /np. opuszczenie nawiasu, brak etykiety, opuszczenie lub użycie niewłaściwej litery itp./.

Po wykryciu błędu formalnego kompilator wstrzymuje operację programu półskompilowanego, ale kontynuuje proces wykrywania dalszych błędów w pozostałej części programu źródłowego. Wyprowadza przy tym komunikat błędu na urządzeniu wyjściowym, zadeklarowanym w pierwszym wierszu sterującym OUTPUT, lub na urządzeniu wyjściowym

zadeklarowanym w dowolnym wierszu sterującym OUTPUT, o ile w ślad za tym wierszem wystąpi ujęte w nawiasach słowo MONITOR.

Przykłady:

```
1/ PROGRAM (ODRA)
   INPUT 8 = TRØ
   OUTPUT 2 = LPØ
   OUTPUT 1 = TPØ
   END
```

komunikat błędu zostanie wyprowadzony na drukarkę wierszową

```
2/ PROGRAM (NYSA)
   INPUT 8 = TRØ
   OUTPUT 2 = TPØ
   OUTPUT 1, (MONITOR) = LPØ
   END
```

program zostanie wyprowadzony na drukarkę wierszową

Postać wyprowadzonego przez kompilator komunikatu o wykrytym błędzie formalnym jest następująca:

ERROR n IN LAST STATEMENT, LINE p, CHAR q

gdzie: n - numer błędu na liście błędów wykrywanych podczas kompilacji
p - numer wiersza z błędem, liczony od 0
q - numer błędnego znaku w wierszu, liczony od 1.

Należy przy tym pamiętać, że błąd może nie wystąpić dokładnie na pozycji określonej przez p i q, ale nieco wcześniej. Poza tym błąd wykazany podczas kompilacji może być konsekwencją błędu wykazanego wcześniej i wystarczy usunąć tylko błąd poprzedni.

W przypadku błędu związanego z etykietą wyprowadzany jest komunikat:

ERROR n - LABEL m

gdzie: n - numer błędu na liście błędów /15 lub 16/
m - numer pominiętej etykiety

Na zakończenie kompilacji podawany jest komunikat podający sumaryczną ilość błędów formalnych w programie źródłowym:

END OF COMPILATION - n ERRORS

gdzie: n - liczba błędów formalnych.

Jeżeli program źródłowy nie zawiera błędów formalnych, kompilator wyprowadza na końcu komunikat:

END OF COMPILATION - NO ERRORS

Poniżej /tabela nr 12/ podany jest fragment listy błędów wykrywanych podacz kompilacji. Kompletne listy podawane są w rozszerzonych opisach języka FORTRAN.

Tabela nr 12

Nr błędu	Interpretacja błędu
1	źle umieszczone nawiasy
2	brak nawiasów
3	opuszczony jeden ze znaków /)
4	błędna postać wyrażenia
5	niedopuszczalna wartość stałej
...	
7	tablica niezadeklarowana lub zadeklarowana dwukrotnie
...	
10	niekompletna instrukcja
...	
14	nieprawidłowa nazwa
15	brak etykiety w instrukcji DO

3.11.2. Wykrywanie błędów w czasie testowania i wykonywania programu

W trakcie testowania i wykonywania programu po wykryciu błędu wykonania działanie programu zostaje wstrzymane, a na urządzenie wyjściowe wyprowadzany jest komunikat błędu.

W przypadku poziomu śledzenia 0, komunikat błędu przyjmuje postać:

```
EXEXUTION ERROR n PROGRAM TERMINATED
```

gdzie: n - numer błędu na liście błędów wykonania.

Poza numerem błędu wykonania, którego charakterystyka podana jest na liście błędów wykonania, wyprowadzony komunikat nie podaje żadnej dodatkowej informacji ułatwiającej lokalizację wykrytego błędu.

Przy zastosowaniu poziomu śledzenia 1 i po wykryciu błędu wykonania pojawia się komunikat:

```
EXECUTION ERROR n PROGRAM TERMINATED
```

natomiast tuż za nim nagłówek "ślad segmentów", posiadający postać:

```
SEGMENT TRACE
```

Pod wymienionym nagłówkiem podana jest lista zawierająca informację o ostatnich 25 wejściach i wyjściach segmentowych programu. Na liście tej wejście do segmentu podane jest poprzez jego nazwę /obciętą do 8 znaków/, natomiast wyjście - za pomocą wyrazu RETURN.

Poziom śledzenia 2 zapewnia wyprowadzenie komunikatu błędu zawierającego najbardziej bogatą treść dotyczącą lokalizacji wykrytego błędu wykonania. Pojawia się, jak poprzednio, komunikat:

```
EXECUTION ERROR n PROGRAM TERMINATED
```

a za nim nagłówek "ślad instrukcji" przyjmujący postać:

STATEMENT TRACE

Pod tym nagłówkiem wypisywanych jest 100 lub mniej wierszy informacji o 100 lub mniej instrukcjach wykonanych przed wystąpieniem błędu. Każdy wiersz dotyczy jednej instrukcji i zawiera następującą informację:

- numer kolejnego wiersza w zakresie od -100 do -1;
- etykietę instrukcji lub spację w razie braku etykiety;
- skrót wykonywanej instrukcji;
- wartość liczbową lub logiczną wykonywanej instrukcji lub spację, jeśli instrukcja nie ma wartości.

Stosowane przy tym skróty instrukcji przyjmują postać podaną w tabeli nr 13.

Tabela nr 13

Instrukcja	Skrót	Wynik wykonania instrukcji
Podstawienie arytmetyczne	ARTH	Wartość lewej strony
Podstawienie logiczne	LOGC	Wartość lewej strony
IF arytmetyczne	IF	Wartość wyrażenia w nawiasach
IF logiczne	LIF	Wartość wyrażenia logicznego
GO TO bezwarunkowe	GO	brak
GO TO warunkowe	CGO	brak
GO TO z ASSIGN	AGO	brak
DO	DO	Wartość początkowa zm.kontr.
READ	READ	brak
WRITE	WR TE	brak
PAUSE	PAUS	
STOP	STOP	
CALL	CALL	
RETURN	RETN	
CONTINUE	CONT	
ASSIGN	ASGN	

Po "śladzie instrukcji" wyprowadzony jest "ślad segmentów"

SEGMENT TRACE

o ostatnich 25 wejściach i wyjściach segmentowych, zawierający taką samą treść jak w przypadku poziomu śledzenia 1.

Tabela nr 14 zawiera fragment listy błędów wykrywanych podczas testowania lub wykonania programu.

Tabela nr 14

Nr błędu	Interpretacja błędu
...	
10	Błąd w strukturze dyrektywy FORMAT, np. nierozpoznany znak
11	Zmiennym typu I, DP lub L przyporządkowano kod konw. E
12	Zmiennym typu R,D,P,C lub L przyporządkowano kod konw. I
...	
16	Zerowa lub nieokreślona szerokość pola w kodach konw. A, L. H. X
...	
105	Podczas czytania kart wystąpił symbol końca danych
106	Podczas czytania taśmy wystąpił symbol końca danych
...	
109	Błąd przesyłania z czytnika kart
...	
509	Zadeklarowano więcej niż 8 numerów programowych urządzeń
...	
512	Numer logiczny urządzenia jest spoza przedziału 0-15
...	
...	

Pytania

1. Wymienić stosowane wersje języka FORTRAN.
2. Jakich znaków używa się w języku FORTRAN.
3. Co to jest dyrektywa?
4. Co to jest instrukcja?
5. Co to jest komentarz?
6. Co to jest etykieta?
7. Co stanowi nazwę?
8. Z jakich segmentów może składać się program napisany w języku FORTRAN?
9. Wymienić typy danych stałych.
10. Omówić stałe typy INTEGER.
11. Omówić stałe typy REAL.
12. Omówić stałe typy DOUBLE PRECISION.
13. Omówić stałe typy COMPLEX.
14. Omówić stałe typy LOGICAL.
15. Jakie dane nazywamy zmiennymi? Scharakteryzować rodzaje zmiennych i ich typy.
16. Co nazywamy wyrażeniem arytmetycznym?
17. Objaśnić zasady budowania wyrażeń arytmetycznych.
18. Co nazywamy wyrażeniem logicznym?
19. Objaśnić zasady budowania wyrażeń logicznych.
20. Wymienić i objaśnić operatory logiczne używane w języku FORTRAN.
21. Wymienić operatory relacji. Jaką postać przyjmuje relacja
22. Objaśnić sposób użycia funkcji standardowych.
23. Omówić instrukcje podstawiania.
24. Omówić zasady budowy i użycia funkcji lokalnych.
25. Omówić instrukcję skoku bezwarunkowego.
26. Omówić instrukcję skoku warunkowego.

27. Omówić instrukcję skoku warunkowego,określonego instrukcją ASSIGN.
28. Omówić instrukcję warunkową arytmetyczną.
29. Omówić instrukcję warunkową logiczną.
30. Omówić instrukcję powtórzeń.
31. Omówić instrukcje: CONTINUE, PAUSE, STOP.
32. Omówić postać i działanie instrukcji READ.
33. Omówić postać i działanie instrukcji WRITE.
34. Omówić dyrektywę FORMAT.
35. Wymienić rodzaje opisów pól.
36. Omówić kod konwersji I.
37. Omówić kod konwersji F.
38. Omówić kod konwersji E.
39. Omówić kod konwersji H.
40. Omówić kod konwersji X.
41. Omówić ograniczniki pól i rekordów.
42. Na czym polega stosowanie formatów swobodnych?
43. Objaśnić DO implikowane.
44. Omówić strukturę segmentu MASTER.
45. Omówić strukturę segmentu SUBROUTINE.
46. Omówić strukturę segmentu FUNCTION.
47. Omówić dyrektywę COMMON.
48. Omówić instrukcję DATA.
49. Omówić strukturę segmentu BLOCK DATA.
50. Wymienić rodzaje wierszy sterujących.
51. Omówić wiersze sterujące wstępne.
52. Omówić wiersze sterujące opisu programu.
53. Omówić wiersz sterujący READ FROM.
54. Jaką rolę spełnia wiersz sterujący FINISH?

55. Omówić automatyczne wykrywanie błędów podczas kompilacji.

56. Omówić automatyczne wykrywanie błędów podczas testowania i uruchamiania programów.

L I T E R A T U R A

1. A.Horak, J.Greń: Usprawnianie procesów dowodzenia i zarządzania w wojsku z zastosowaniem metod i środków informatyki. MON, Warszawa 1976.

2. M.Pasternak: Ogólne wiadomości z zakresu algorytmizacji i programowania. MON, Warszawa 1976.

3. Cz.Gozdecki: Wybrane zagadnienia algorytmizacji zadań operacyjno-taktycznych. MON, Warszawa 1971.

4. K.Jaroń: Zasady algorytmizacji zadań wojskowych. ASG WP, Warszawa 1971.

5. Praca zbiorowa: Automatyczne przetwarzanie informacji. PWE, Warszawa 1976.

6. L.A.Sieniawski: Problemy i metody programowania. Politechnika Wrocławska, Wrocław 1980.

7. J.Hawryluk: Maszyna cyfrowa narzędzie człowieka współczesnego. WNT, Warszawa 1976.

8. Praca zbiorowa: O maszynach cyfrowych. PWE, Warszawa 1968.

9. Peter Fisher George F.Swindle: Systemy programowania maszyny cyfrowej. WNT, Warszawa 1971.

10. J.Bańkowski i inni: Programowanie w języku FORTRAN. PWN, Warszawa 1978.

Rozdział czwarty :

KOMPUTEROWE GRY WOJENNE

Rozwój nauki i techniki powoduje, że na istotę i treść aktualnych zjawisk składa się coraz większa liczba czynników i coraz bardziej skomplikowane stają się współzależności między nimi. W wielu dziedzinach racjonalnej działalności ludzkiej coraz trudniejsze stają się warunki w jakich trzeba realizować zamierzone cele. Ten stan rzeczywistości stawia większość decydentów w obliczu bardzo złożonych sytuacji problemowych, których jakość rozstrzygnięcia pociąga za sobą nieraz daleko idące następstwa o charakterze społecznym, politycznym, gospodarczym, naukowym, technicznym, wojskowym, itp.[7],[8],[12]. Dlatego też właściwa realizacja procesów kierowania /zarządzania, dowodzenia/ w wielu obszarach ludzkiej działalności nabiera aktualnie szczególnego znaczenia. Na decydentów wszystkich szczebli w różnych systemach kierowania, nakłada się stale rosnące wymagania, dotyczące umiejętności dokonywania bieżącej analizy i oceny sytuacji, aktywności i odwagi w procesach decyzyjnych, zdolności przewidywania następstw podejmowanych decyzji, wysokiej operatywności podczas podejmowania decyzji.

Mając na względzie obserwowany aktualnie proces przyspieszonego starzenia się wiedzy, szczególnego znaczenia dla efektywnej działalności decydentów nabiera bieżące doskonalenie i pogłębianie przez kadrę kierowniczą niezbędnego wykształcenia specjalistycznego. Nie zawsze bowiem można oczekiwać właściwych i przyszłościowo uzasadnionych decyzji, podejmowanych w oparciu o wiedzę z przed lat kilku czy kilkunastu.

W zakresie doskonalenia kadr kierowniczych, coraz wyraźniej wysuwa się na plan pierwszy, istotny problem doboru metod i form przekazywania odpowiedniej wiedzy o kierowaniu zespołami ludzi oraz kształtowanie intelektualnych cech aparatu kierowniczego, stanowiących o sprawności tego kierowania. Wypracowanie nowych i doskonalenie istniejących metod kierowania wielkimi systemami, to jedno z podstawowych zadań, jakie stoi przed matematykami, psychologami, socjologami i dowódcami różnych szczebli.

Jedną z ważniejszych współcześnie metod aktywnego uczenia i doskonalenia kadry kierowniczej jest metoda gier kierowniczych, zwanych również decyzyjnymi, a w obszarze zastosowań wojskowymi grami kierowniczymi.

Grą kierowniczą nazywać będziemy model określonego systemu działania, w którym odwzorowano procesy robocze i informacyjno-decyzyjne, jak również procesy współdziałania decydentów przy realizacji celu systemu [4], [7], [8], [9].

Celem gier kierowniczych jest doskonalenie kadry kierowniczej w rozwiązywaniu złożonych problemów decyzyjnych w sytuacjach współpracy, współzawodnictwa i w sytuacjach konfliktowych, poszukiwanie najkorzystniejszych rozwiązań konkretnych problemów powstających w działalności systemów wojskowych, ocena kwalifikacji kadry oraz dobór kandydatów na kierownicze stanowiska służbowe [4], [7], [10], [13].

Szczególnym rodzajem wojskowych gier kierowniczych są gry wojenne. Specyfika ich polega na tym, że przedmiotem gier wojennych są tylko i wyłącznie systemy działań bojowych.

1/ Grę kierowniczą można interpretować jako symulację procesów realizowanych w określonym systemie działania, prowadzoną w celach szkoleniowych lub badawczych [2], [4], [6], [9].

Gry kierownicze, a w szczególności gry wojenne znane są od dawna jako sposób sprawdzania wiedzy i umiejętności jej wykorzystania w praktyce. Najprostszą formą prowadzenia gier wojennych są wszelkiego typu ćwiczenia prowadzone na mapach i stołach plastycznych w oparciu o zasady sztuki wojennej.

Podstawowymi elementami gry wojennej są: model określonego systemu działań bojowych,[2/] poznanie sposobu funkcjonowania, którego zarówno w obszarze realizacji procesów walki, jak i informacyjno-decyzyjnych stanowić może przedmiot gry, uczestnicy gry, reguły, scenariusz i procedury gry.

Takie gry wojenne, w których do odwzorowania procesów realizowanych w systemie działań bojowych opracowano model rozwiązywany /uaktywniany/ przy pomocy techniki komputerowej, nazywać będziemy komputerowymi grami wojennymi [1], [5], [6], [10], [13].

W przeszłości gry kierownicze wykorzystywane były przede wszystkim w procesie nauczania kadr, natomiast współcześnie dzięki pojawieniu się techniki komputerowej, zaistniała możliwość wykorzystania ich do badania różnych aspektów funkcjonowania złożonych systemów działania.

Badania systemów aktywnych, tylko i wyłącznie za pomocą komputerów, bez twórczego udziału człowieka, jak pokazały doświadczenia ostatnich dziesięcioleci, nie mogą dać pełnego obrazu ich funkcjonowania. Przyczyną takiego stanu rzeczy jest brak możliwości opracowania formalnie poprawnego modelu dowolnego systemu działania, w którym adekwatnie do rzeczywistości odwzorowanoby działalność człowieka i wpływ tych czynników niemierzalnych, które mają istotny wpływ na efektywność funkcjonowania

---

2/ W systemie działań bojowych /związek operacyjny, związek taktyczny, oddział .../ wyróżniamy dwa podstawowe podsystemy: system dowodzenia i system walki [8]. System działań bojowych obejmować może zarówno wojska własne, jak i przeciwnika.

systemu. Wydaje się, że stosowanie metody gier kierowniczych, której istotą jest udział w procesie ich realizacji człowieka, traktowanego jako aktywny element modelowanego systemu działania, ułatwi proces badania złożonych systemów aktywnych.

Przedmiotem dalszych rozważań będą przede wszystkim komputerowe gry wojenne, projektowane dla potrzeb doskonalenia decydentów systemów działań bojowych wojsk lądowych.

1. Charakterystyka podstawowych elementów komputerowej gry wojennej i opis wybranych procedur jej projektowania

Proces projektowania komputerowych gier wojennych jest przedsięwzięciem trudnym i czasochłonnym. Jak wykazują doświadczenia wielu zespołów projektowych czas projektowania i próbnej eksploatacji wielu gier komputerowych wynosił często kilka lat [5], [10], [13].

Przed przystąpieniem do projektowania komputerowej gry wojennej należy jasno i precyzyjnie sformułować cel gry. Jeżeli gra mieć będzie charakter szkoleniowy, należy określić system działań bojowych będący przedmiotem gry oraz sprecyzować jacy decydenci w jakiej kolejności i w jakim zakresie będą szkoleni. Jeśli natomiast celem gry jest poznanie sposobu funkcjonowania systemu, należy w miarę dokładnie określić obszar jego badania.

Na etapie formułowania celu gry, szczególna rola do spełnienia przypada użytkownikom i decydentom systemu rzeczywistego, którzy z racji wykonywanych zadań i sprawowanych funkcji kierowniczych powinni najlepiej znać specyfikę funkcjonowania systemu będącego przedmiotem gry.

Przeprowadzenie analizy funkcjonowania systemu działań bojowych, będącego przedmiotem gry, determinuje realizację dalszych prac projektowych, w wyniku których należy opracować i określić między innymi :

- model systemu działań bojowych;
- sposób realizacji /przebiegu gry/;
- uczestników gry;
- reguły prowadzenia gry;
- scenariusz gry;
- role uczestników gry;
- informacyjne zabezpieczenie procesu realizacji gry;
- materiały szkoleniowe dla uczestników gry,

oraz przeprowadzić próbną eksploatację gry komputerowej.

1.1. Model systemu działań boojowych

Opracowanie modelu wybranego systemu działań bojowych wojsk własnych i /lub/ przeciwnika, sposób funkcjonowania których stanowi przedmiot gry, jest zadaniem najbardziej złożonym i czasochłonnym.

W procesie konstrukcji modelu można dążyć do odwzorowania w różnym zakresie i z różnym stopniem dokładności procesów realizowanych zarówno w systemie dowodzenia /podsystemy: informacyjny i decyzyjny/, jak również w systemie walki /oddziały, pododdziały/ wybranego systemu działań bojowych.

W większości opracowanych komputerowych grach wojennych, w modelu systemu mającego często postać modelu matematycznego, odwzorowano przede wszystkim procesy walki, w mniejszym zakresie procesy informacyjne i decyzyjne, których odwzorowania przeniesiono w obszar funkcjonowania uczestników gry. Dlatego też można je interpretować jako określony typ ćwiczeń dowódczo-sztabowych bez udziału wojsk [1], [5], [6], [10], [13].

Jeżeli przedmiotem gry będzie system działań bojowych typu np. pododdział czołgów, to model takiego systemu powinien odwzoro-

wywać między innymi proces niszczenia z dokładnością do pojedynczego czołgu. Inaczej wyglądać będzie sytuacja gdy przedmiotem gry będzie system działań bojowych typu związek operacyjny. Tak szczegółowe odwzorowanie procesów niszczenia nie jest konieczne i niecelowe ze względu na między innymi zbyt długi czas realizacji gry komputerowej, jak również ze względu na pewne trudności w formułowaniu globalnych charakterystyk funkcjonowania systemu działań bojowych w oparciu o charakterystyki jego procesów elementarnych. Jest to zagadnienie, które w bardzo uproszczonej postaci można sprowadzić do pytania - W jaki sposób odwzorować proces niszczenia, realizowany np. przez oddział czołgów bez odwzorowania procesów niszczenia realizowanych przez pojedynczy czołg?

Funkcjonowanie systemów działań bojowych w przewidywanych warunkach pola walki mieć będzie zwykle bardzo złożony charakter. Opracowanie modelu uwzględniającego szeroki zakres zmian warunków i sytuacji w jakich funkcjonować może system, jest zagadnieniem złożonym, z tego też względu projektuje się gry komputerowe w których model odwzorowuje funkcjonowanie systemu w pewnym zakresie warunków i sytuacji pola walki /np. działanie systemu w czasie przełamywania obrony przeciwpancernej, forsowania z marszu przeszkody wodnej, rozwijania wojsk i wejścia ich do walki lub bitwy/.

W modelu systemu odwzorowuje się przede wszystkim te procesy i elementy systemu rzeczywistego, które decydują o zakresie "wkomponowanego" w model realizmu pola walki. Do najważniejszych z nich należy zaliczyć jednostki wojsk własnych i przeciwnika oraz realizowane przez nie procesy obejmujące między innymi ruch i manewr sił i środków, wykrycie i niszczenie celów, szacowanie strat oraz teren przewidywanych działań [1], [3], [5], [6], [10], [13].

Najistotniejszym elementem modelowanego systemu są jednostki wojsk własnych i przeciwnika. Odwzorowanie ich powinno obejmować między innymi:

- stan ilościowy ludzi i sprzętu, wybranych typów uzbrojenia;
- planowane kierunki manewru;
- planowane rodzaje działań.

Zbiór informacji o wojskach własnych i przeciwnika zapisywać będziemy odpowiednio w macierzach $[WW]_{NxP}$, $[WP]_{MxP}$, natomiast stan j-tej jednostki w chwili t będziemy zapisywać w postaci następującego wektora:

$$J^j(t) = \langle WP^j(t),\ S^j(t),\ D^j(t) \rangle$$

gdzie:

$j = 1,2,\ldots,J$ - numery jednostek;

WP^j t - wskaźnik przynależności j-tej jednostki w chwili t;

$S^j(t) = \langle SO^j(t),\ SW^{j,1}(t),\ldots,SW^{j,s}(t),\ldots,SW^{j,S_j}(t) \rangle$ - stan ludzi $/SO^j(t)/$ i uzbrojenia $/SW^{j,s}(t)/$ j-tej jednostki w chwili t;

$S = 1,2,\ldots,s,\ldots\ S_j$ - numery środków walki poszczególnych rodzajów;

$D^j(t) = \langle d^j(t), T^j, \langle x_k^j, y_r^j \rangle, \ldots \langle x_s^j, y_z^j \rangle, PZ^j, PO^j \rangle$ - rodzaj realizowanych działań $/d^j(t)/$, przewidywany czas realizacji przyjętego rodzaju działań $/T^j/$, planowany kierunek działania /przemieszczania się jednostki/ $\langle x_k^j, y_r^j \rangle, \ldots$ - współrzędne kwadratów terenu wyznaczające planowane kierunki działania jednostki[3], PZ^j - potencjał bojowy j-tej jednostki, przy którym następuje załamanie się działań zaczepnych - przejście do obrony $/PO^j$ - załamanie się obrony - przejście do wycofania/[4].

Każdy środek walki opisujemy za pomocą następującego wektora parametrów :

3/ Jeżeli określona jednostka zamierza prowadzić działania obronne to podane numery kwadratów terenu oznaczać mogą między innymi kierunek ewentualnego wycofania jednostki.

4/ Taki stan sił i środków, który nie rokuje powodzenia dla prowadzonego aktualnie rodzaju działań bojowych.

$$SW^{j,s}(t) = \left\langle SI^{j,s}(t), P^{j,s}(t), \Lambda_a^{j,s}(t), ZA^{j,s}(t), \Lambda_p^{j,s}(t), ZP^{j,s}(t) \right\rangle$$

gdzie:

$SI^{j,s}(t)$ - ilość środków walki s-tego typu j-tej jednostki w chwili t ;

$P^{j,s}(t) = \left\langle P_{1,1}^{j,s}(t), \ldots, P_{r,1}^{j,s}(t), \ldots, P_{R,1}^{j,s}(t) \right\rangle$ - wektor prawdopodobieństw zniszczenia jednym pociskiem w chwili t jednego środka walki wojsk przeciwnika;
Np. $P_{r,1}^{j,s}(t)$ - pradowpodobieństwo zniszczenia jednym pociskiem w chwili t r-tego środka walki przez s-ty środek walki j-tej jednostki wojsk własnych;

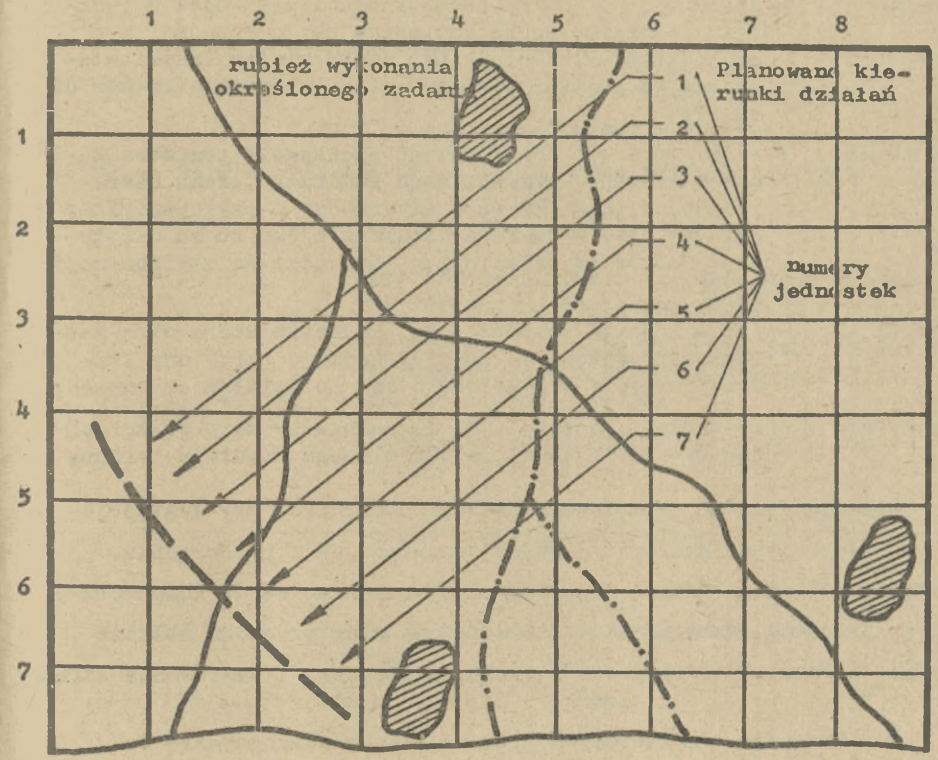
$\Lambda_a^{j,s}(t)$ - intensywność prowadzenia ognia przez środek walki s-tego typu j-tej jednostki w chwili t;

$ZA^{j,s}(t)$ - zapas amunicji s-tego środka walki j-tej jednostki w chwili t;

$\Lambda_p^{j,s}(t)$ - intensywność zużycia paliwa przez środek walki s-tego typu j-tej jednostki w chwili t;

$ZP^{j,s}(t)$ - zapas paliwa s-tego środka walki j-tej jednostki w chwili t.

Informacje o planowanym kierunku działania jednostek wojsk własnych i przeciwnika opracowuje się przed rozpoczęciem gry komputerowej w trzech etapach. Etap pierwszy obejmuje wrysowanie informacji o planowanych zadaniach i kierunkach działania jednostek na mapy ćwiczących /grających/ uczestników gry. W etapie drugim informacje te przenosi się na specjalnie przygotowaną oleatę w której przewidywany teren działań bojowych podzielono na kwadraty z podaniem ich numeracji. Tak przygotowany dokument stanowi swoistego rodzaju element scenariusza gry komputerowej, którego jedną z możliwych postaci przedstawia rys. 1. W trzecim etapie informacje o planowanych kierunkach działania jednostek /numery kolejnych kwadratów przez które planuje przejść określona jednostka/ wpisuje się do macierzy $[WW]_{NxP}$ i $[WP]_{MxP}$ w oparciu o które przygotowuje się dane wejściowe do symulacyjnych programów komputerowych.



Rys. 1. Graficzna ilustracja informacji o planowanych kierunkach działania jednostek, przygotowana dla potrzeb komputerowej gry wojennej.

Odwzorowanie możliwych rodzajów działania jednostek stanowi w procesie modelowania element najważniejszy. Dla potrzeb komputerowych gier przyjmuje się często między innymi takie umowne rodzaje /sposoby/ działań, jak:

- marsz - M;
- natarcie - N;
- obrona - O;
- wycofanie - W.

Wymienione rodzaje działań stanowią swoisty element reguł gry. Odwzorowane w programach symulacyjnych w postaci określonych procedur i instrukcji mogą mieć następującą interpretację:

- marsz - instrukcja ta ma na celu przesunięcie wojsk z jednego kwadratu terenu do innego bez styczności z przeciwnikiem. Powinna być użyta między innymi wtedy gdy rozpoczyna się wprowadzanie drugich rzutów do walki;

- natarcie - siły które otrzymają rozkaz atakowania przesuwają się w kierunku wyznaczonego kwadratu terenu niezależnie od tego czy są w styczności z przeciwnikiem, czy też nie. Jeżeli na drodze swojego ruchu napotykają jednostkę przeciwnika, dochodzi do konfrontacji sił;

- obrona - instrukcja ta powoduje, że jednostka która otrzymała rozkaz przejścia do obrony, zajmuje określony kwadrat terenu i przygotowuje się do działań obronnych;

- wycofanie - instrukcja ta powoduje, że zaangażowana w walkę jednostka wycofuje się do określonego kwadratu terenu.

Rodzaje działań określone są przez uczestników gry występujących zarówno w roli decydentów wojsk własnych jak i przeciwnika. Następnie w postaci danych wejściowych wprowadzane są do komputera i przez programy symulacyjne, tłumaczone na ciąg procedur obliczeniowych, odwzorowujących procesy ruchu, niszczenia i szacowania strat.

W większości komputerowych gier wojennych odwzorowanie terenu obejmuje opis właściwości fizycznych oraz analizę terenu z punktu widzenia jego wpływu na tempo i rodzaj prowadzonych działań bojowych.

Informacje o terenie przygotowuje się przed rozpoczęciem gry wojennej i wprowadza do komputera jako dane stałe.

W celu przygotowania informacji o terenie, w postaci umożliwiającej wprowadzenie jej i przetwarzanie na komputerze, teren przewidywanych działań bojowych "dzieli" się na kwadraty, których rozmiar powinien odpowiadać normom operacyjno-taktycznym rozmieszczenia wojsk w terenie dla jednostek tego szczebla organizacyjnego, który przyjmujemy za podstawowy w danej grze. 5/

5/ Np. dla jednostek typu pułk zmechanizowany wymiary kwadratów przyjmuje się (10 x 10) km.

Każdy kwadrat terenu opisywany jest zbiorem parametrów, które charakteryzują jego właściwości fizyczne i operacyjno-taktyczne. Zbiór informacji o terenie zapisywać możemy w postaci macierzy $[TER]_{IxK}$, której poszczególne elementy mogą mieć następującą interpretację:

$$TER(i,k) = \begin{cases} \text{- typ terenu z punktu widzenia jego wpływu na tempo poruszania się wojsk } (k = 1); \\ \text{- typ terenu z punktu widzenia jego wpływu na skuteczność i formę prowadzenia działań obronnych } (k = 2); \\ \text{- ważniejsze właściwości fizyczne i polityczne } (k = 3); \\ \text{- numery jednostek aktualnie znajdujących się w i-tym kwadracie terenu } (k = 4); \end{cases}$$

gdzie:

$k = 1,2,...K$ - numery parametrów opisujących właściwości terenu;

$i = 1,2,...I$ - numery kwadratów terenu.

Teren z punktu widzenia wpływu na tempo poruszania się wojsk podzielić można między innymi na następujące typy:

- doskonały - dominujące warunki terenowe pozwalają na swobodne poruszanie się wozów bojowych we wszystkich kierunkach z prędkościami uwarunkowanymi jedynie właściwościami poruszających się jednostek;

- dobry - przeważające warunki terenowe umożliwiają swobodne poruszanie się wozów bojowych w dowolnym kierunku z prędkością nieco mniejszą od maksymalnej;

- umiarkowany - dominujące warunki terenowe umiarkowanie hamują ruch lub ograniczają kierunki poruszania się wojsk. Przyczyną tych ograniczeń mogą być bagna, jeziora lub kanały, lokalne zalesienia i płytkie doliny oraz strome zbocza;

- trudny - przeważające warunki terenowe poważnie hamują poruszanie się na przełaj pojazdów gąsienicowych. Ograniczenia mogą być spowodowane przez zalesione wzgórza, gęste lasy, jeziora, bagna itp.;

- bardzo trudny- dominujące warunki terenowe pozwalają na prędkośćnieco poniżej wskaźników słusznych dla warunków trudnych.

Przedstawiony podział terenu ma charakter umowny i dla potrzeb konkretnej gry komputerowej może mieć inną postać.

Dla każdego z wymienionych typów terenu, wprowadza się współczynnik $\alpha \in [0,1]$, który pozwala na określenie prędkości poruszania się wojsk, właściwej dla danego typu terenu.

Charakterystyka terenu z punktu widzenia jego wpływu na skuteczność prowadzenia działań obronnych, powinna odzwierciedlać w jakim stopniu właściwości fizyczne i obronne terenu ułatwiają lub utrudniają prowadzenie działań obronnych.

Tak sformułowane informacje o terenie przyszłych działań bojowych stanowią zbiór informacji wejściowych do programów komputerowych,w wyniku realizacji których oblicza się między innymi tempo działań bojowych, czas osiągania określonych rubieży, stopień strat wojsk własnych i przeciwnika.

Informacje o terenie przechowywane są w pamięci komputera przez cały czas realizacji gry i na bieżąco uaktualniane.

Przedstawiony w ogólnym zarysie model systemu działań bojowych będącego przedmiotem gry, stanowi podstawę do opracowania algorytmu i symulacyjnych programów komputerowych.

1.1.1. Ogólny opis algorytmu realizacji wybranych procesów funkcjonowania systemu działań bojowych

W prezentowanym algorytmie odwzorowano procesy ruchu wojsk, niszczenia i szacowania strat dla jednostek będących w dyspozycji obu grających stron. Wymienione procesy odwzorowano z uwzględnieniem właściwości terenu, rodzaju i sposobów prowadzonych działań bojowych, a także decyzji podejmowanych przez uczestników gry.

Algorytm, którego ogólny schemat blokowy przedstawia rys. 2. funkcjonuje w następujący sposób.

Dla wybranych chwil czasu, wyznaczonych metodą kolejnych zdarzeń[3], dla każdej jednostki wojsk własnych i przeciwnika oblicza się czas osiągania kolejnych kwadratów terenu, uwzględniając rodzaje prowadzonych działań, potencjały bojowe oraz właściwości terenu. Z obliczonych czasów wybiera się czas minimalny, który dodany do bieżącego czasu systemowego wyznacza chwilę kolejnej zmiany stanu systemu i pozwoli także uaktualnić położenie jednostek.

Kolejne procedury algorytmu polegają na sprawdzeniu dla każdego kwadratu terenu, jakie jednostki w nich się aktualnie znajdują.

Jeżeli w danym kwadracie terenu znajdują się jednostki wojsk własnych i przeciwnika, dochodzi do konfrontacji sił właściwej dla rodzajów działań, jakie zaplanowali dla swoich jednostek uczestnicy gry. Dla wymienionych jednostek, w oparciu o informacje o terenie, rodzaju prowadzonych działań i aktualnym stosunku potencjałów bojowych, przy pomocy specjalnego zbioru wskaźników /prawdopodobieństwo zniszczenia jednym strzałem przez r-ty środek walki i-tej jednostki, s-tego środka walki j-tej jednostki, intensywność prowadzenia ognia, tempo prowadzenia działań bojowych dla różnych wartości stosunku potencjałów bojowych jednostek/

i funkcji oceny potencjałów bojowych, oblicza się tempo prowadzonych działań bojowych oraz poniesione straty [4], [5], [13].

Opracowanie wspomnianych wskaźników i funkcji oceny potencjałów bojowych jednostek, stanowi najistotniejszy element procedury projektowania komputerowych gier wojennych i sam w sobie kolejny problem badawczy.

Przykład zależności tempa prowadzenia określonego rodzaju działań bojowych od stosunku potencjałów bojowych jednostek dla ustalonego typu terenu przedstawia rys. 3 .

Po obliczeniu strat, uaktualnia się stan ilościowy sprzętu i uzbrojenia oraz sprawdza czy osiągnął on poziom załamania właściwy dla danego rodzaju działań.

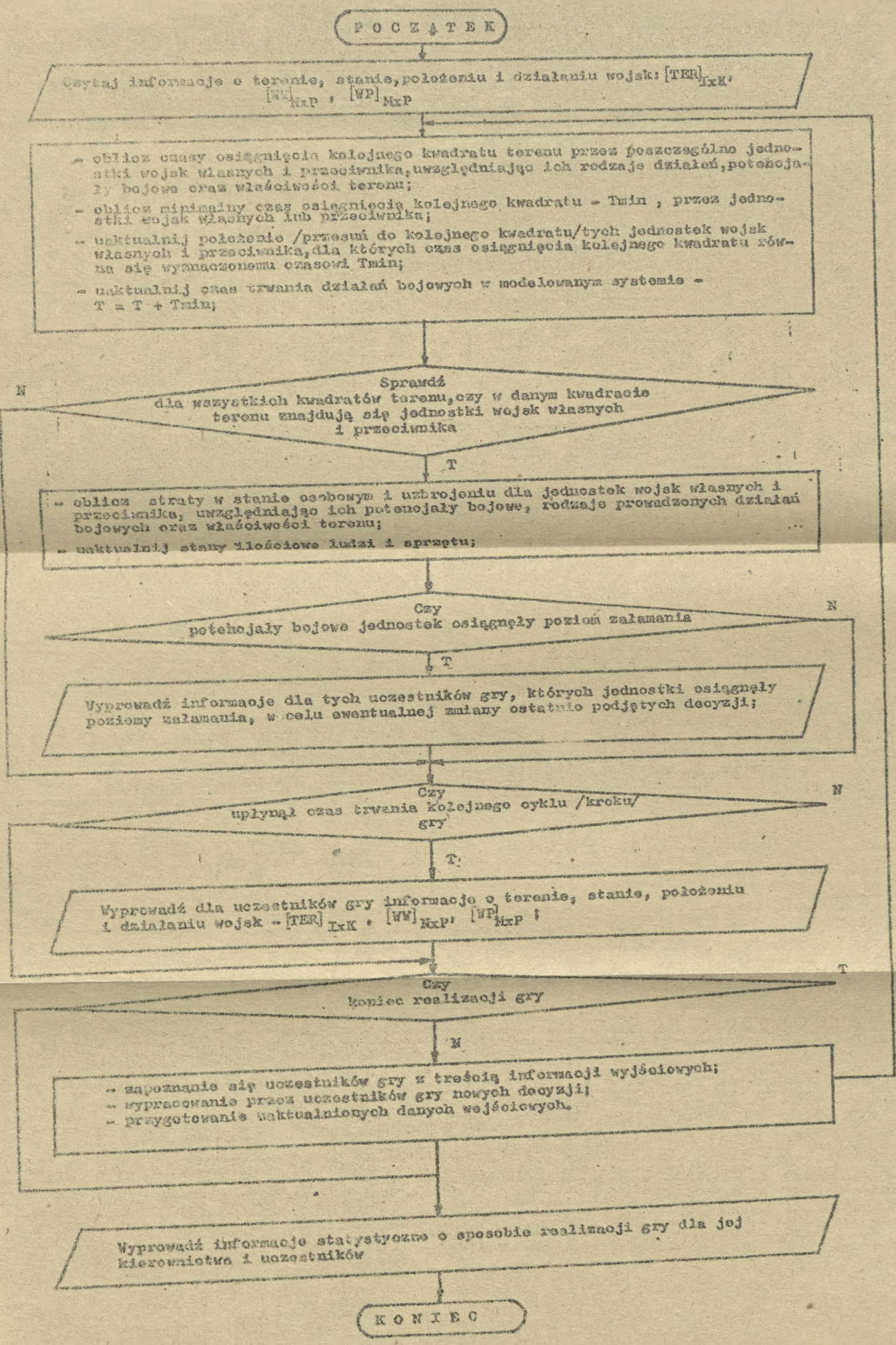
Dla jednostek, które osiągnęły poziom załamania, z komputera wyprowadzane są informacje, które wybranym uczestnikom gry pozwolą na wypracowanie /skorygowanie/ decyzji właściwych dla zaistniałej sytuacji.

Informacje wynikowe /o stanie ilościowym uzbrojenia i aktualnym położeniu jednostek/ wyprowadzane są z komputera dla określonych wartości czasu systemowego, które odpowiadają wybranym wartościom czasu funkcjonowania systemu rzeczywistego.

Proces obliczeń kończy się gdy czas systemowy osiągnie zadaną wartość.

1.2. Scenariusz gry

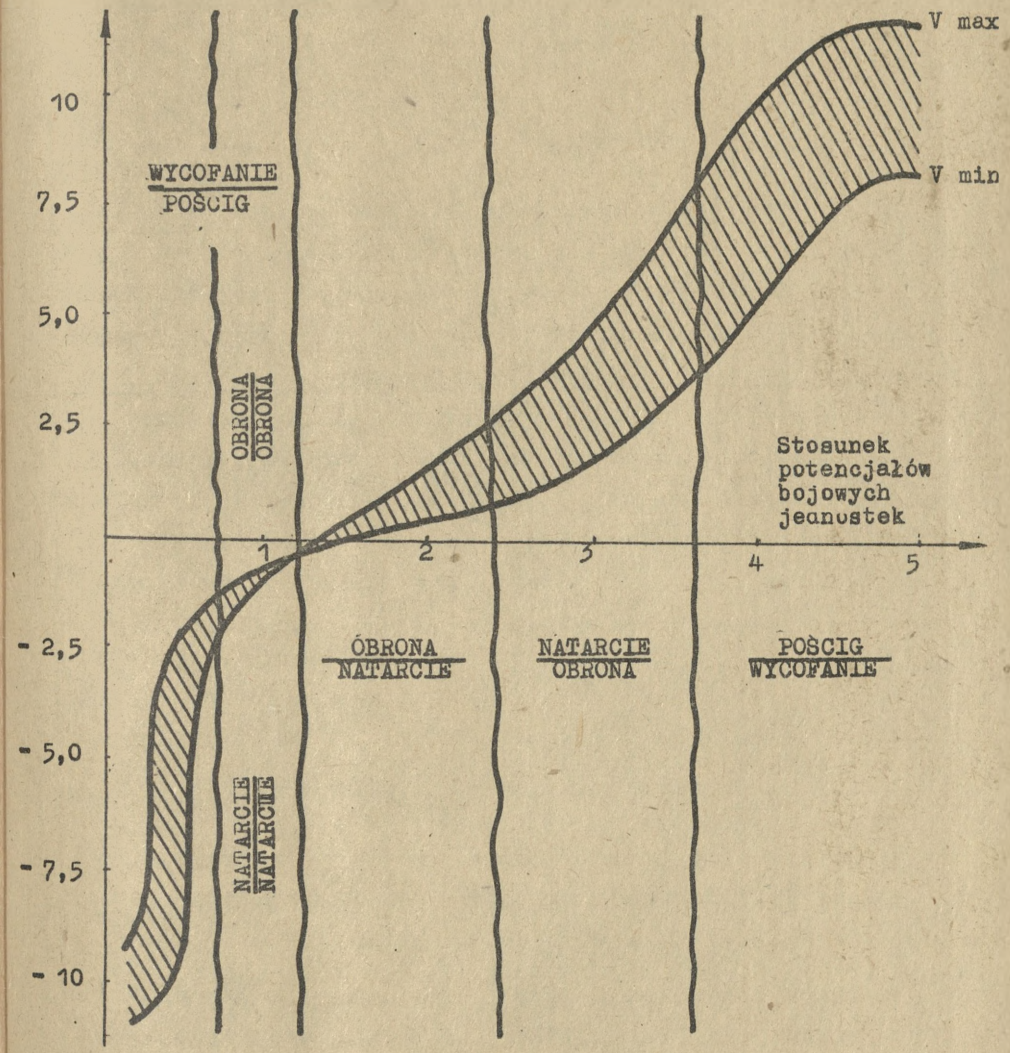
Wyniki analizy funkcjonowania systemu z uwzględnieniem przewidywanych warunków pola walki i pełna identyfikacja możliwych sytuacji w obszarze funkcjonowania podsystemów walki i dowodzenia, pozwolą na sprecyzowanie szeroko rozumianego opisu wariantów działań systemu. Na bazie tak rozumianego opisu stanowiącego swoistego rodzaju repertuar zachowań /historię/ systemu, opracowujemy scenariusz gry, który można interpretować jako jeden z epizodów wspomnianego



Rys. 2. Ogólny algorytm realizacji wybranych procesów funkcjonowania systemu działań bojowych.

Tempo prowadzenia określonego rodzaju działań bojowych [km/godz]

Typ terenu = const



Rys. 3. Przebieg zmian tempa prowadzenia określonego rodzaju działań bojowych

opisu. Scenariusz gry jest tym elementem, który jej uczestnikom determinuje sytuację początkową, precyzuje ogólne tło działania systemu i ukierunkowuje ich na poznanie tylko wybranych obszarów i aspektów funkcjonowania systemu.

1.3. Role uczestników gry

Istotą gier kierowniczych jest to, że w procesie ich realizacji uczestniczy człowiek jako aktywny element modelowanego systemu. Stwarza to wyjątkowe możliwości w zakresie wykorzystania potencjału intelektualnego uczestników gry i stymulowania ich działania w kierunku mobilizacji ich uwagi i wiedzy na najbardziej twórczy udział w grze. Mając na względzie te możliwości, przy precyzowaniu ról dla uczestników gry należy dążyć do wkomponowania w nie elementów zainteresowania najbardziej efektywnym przebiegiem i rezultatami gry, przy jednoczesnym zapewnieniu satysfakcji z uczestnictwa w niej. Role uczestników gry wynikają w określonym stopniu z zadań, obowiązków i kompetencji jakie spełniają oni w warunkach funkcjonowania systemu rzeczywistego [4], [7], [8], [12].

1.4. Informacyjne zabezpieczenie przebiegu gry

W procesie funkcjonowania komputerowej gry wojennej ważną rolę spełnia właściwy obieg informacji i stosowane do potrzeb uczestników gry jej informacyjne zabezpieczenie.

Analiza i identyfikacja procesów informacyjnych /treść, forma, operacje wykonywane na informacjach, czasy wykonywania tych operacji/ realizowanych w systemie rzeczywistym, pozwolą na sprecyzowanie między innymi informacji wejściowych do opracowanego modelu oraz informacji wyjściowych uzyskiwanych w wyniku rozwiązania modelu.

Informacje wejściowe do modelu odwzorowującego działanie podsystemu walki, mają interpretację informacji decyzyjnych opracowywanych przez uczestników gry. Natomiast informacje wejściowe dostarczone uczestnikom gry w wyniku realizacji programów komputerowych opracowanych na bazie przyjętego modelu, można interpretować jako informacje sytuacyjne /stan, położenie, działanie i ugrupowanie wojsk/.

Zabezpieczenie informacyjne procesu realizacji gry komputerowej sprowadza się do sprecyzowania formy, treści oraz dróg i czasu przekazywania informacji z uwzględnieniem potrzeb uczestników gry /element podsystemu dowodzenia/ oraz wymagań związanych z funkcjonowaniem i realizacją modelu systemu /element podsystemu walki/.

Informacyjne zabezpieczenie komputerowej gry wojennej związane jest ściśle z konstrukcją modelu badanego systemu. Dlatego też treść, forma i sposób przetwarzania informacji powinny być jasno i precyzyjnie określone.

1.5. Reguły gry

Analiza i odwzorowanie szeroko rozumianych procesów informacyjnych realizowanych w systemie rzeczywistym warunkują określenie przepływu informacji w trakcie realizacji gry i w określonym stopniu opracowanie modelu systemu. Natomiast analiza procesów decyzyjnych i związanych z nimi informacji pozwoli na opracowanie reguł gry.

Reguły gry odzwierciedlają stosunki między ludźmi, a bardziej dokładnie między elementami systemu w modelowanych /rozgrywanych/ sytuacjach i są pewnego rodzaju ograniczeniami w ramach których uczestnicy gry mogą przejawiać indywidualne możliwości. Reguły gry określają prawa i obowiązki uczestników gry, ustalają zakresy podejmowanych przez nich decyzji oraz bardzo często określają sposoby oceny tychże wyników.

Dokładność opracowania formalnych reguł gry nie powinna wykluczać swobody indywidualnego postępowania uczestników gry. Oprócz tego reguły gry powinny stymulować rozwój twórczego myślenia, wybór najbardziej efektywnych strategii, itp. Ograniczenia nałożone na reguły gry nie powinny być zbyt ścisłe, ale jednocześnie przy konstrukcji komputerowej gry wojennej nie należy pozostawiać pełnej swobody uczestnikom gry /w systemie rzeczywistym są to ograniczenia o charakterze norm i zarządzeń/.

Opracowując w oparciu o zbiór możliwych i dopuszczalnych zachowań decydentów systemu rzeczywistego, reguły komputerowych gier wojennych należy dążyć do tego, by sytuacje decyzyjne w jakich znajdują się uczestnicy gry nie byłyby uproszczone do tego stopnia by można rozwiązywać je bez głębszej ich analizy.

1.6. Uczestnicy gry

W procesie realizacji gry komputerowej rolę najważniejszą spełniają uczestnicy gry. Od ich decyzji zależą bowiem przebieg i wyniki gry. Dlatego też pełna identyfikacja potencjalnych uczestników /odbiorców/ gry ze wskazaniem ich miejsca i roli w strukturze organizacyjnej i funkcjonalnej systemu wraz z oceną ich zawodowych i intelektualnych możliwości, w zasadniczym stopniu decyduje o efektywności prac projektowych, szczególnie w zakresie realizacji programów komputerowych i sposobu przebiegu gry.

Dokonując doboru uczestników gry należy w szczególności zwrócić uwagę na ich przygotowanie zawodowe i stopień zainteresowania rezultatami gry. Należy również uwzględnić to, by funkcje jakie będą wykonywać w procesie realizacji gry odpowiadały w maksymalnym stopniu ich zawodowym nawykom. Bowiem tylko w takich przypadkach uczestnicy gry czuć będą się najlepiej, a ich faktyczne zachowanie odpowiadać będzie ich zachowaniu w systemie rzeczywistym.

1.7. Sposób przebiegu gry

Przed rozpoczęciem komputerowej gry wojennej, informacje o terenie, wojskach własnych i przeciwnika wprowadzane są do komputera jako informacje wejściowe do programów symulacyjnych. Informacje te stanowią odbicie scenariusza gry, który zawiera między innymi opis terenu, jednostek wojsk własnych i przeciwnika oraz ogólną charakterystykę sytuacji wraz z zadaniami postawionymi przez nadrzędne systemy dowodzenia.

Proces realizacji komputerowej gry wojennej obejmuje zwykle kilka cykli /kroków/, których czas trwania w zależności od podstawowego szczebla jednostek odwzorowanych w modelowanym systemie, może mieć różną wartość.[6/]

Podobnie jak w tradacyjnych grach wojennych, w grach komputerowych, w każdym okresie funkcjonowania modelowanego systemu wyróżnić można następujące etapy:

- zbieranie i formułowanie informacji o sposobie funkcjonowania modelowanego systemu,
- analiza położenia, planowanie działań i wypracowanie przez uczestników gry decyzji,
- realizacja podjętych decyzji.

Na etapie zbierania i formułowania informacji, uczestnikom gry dostarczane są z komputera na stosowne urządzenia końcowe /dalekopis, monitor ekranowy, drukarka/ informacje o stanie, położeniu, działaniu i ugrupowaniu jednostek, wypracowane w oparciu o programy komputerowe imitujące procesy realizowane w modelowanym systemie działań bojowych. Informacje te stanowią dla uczestników gry podstawę do oceny położenia i wypracowania decyzji w kolejnej fazie gry.

6/ Dla jednostek szczebla oddziału czas trwania jednego cyklu gry odpowiada zwykle (4 - 6) godzinom funkcjonowania systemu rzeczywistego.

W oparciu o informacje uzyskiwane z komputera oraz przy pełnej znajomości reguł, scenariusza gry i sposobu funkcjonowania modelu systemu, uczestnicy gry wypracowują stosowne decyzje. W procesie wypracowywania przez uczestników gry decyzji uwzględnia się także sformułowane przez kierownictwo gry. oceny o poszczególnych uczestnikach gry, jak również o stopniu osiągania celu modelowanego systemu.

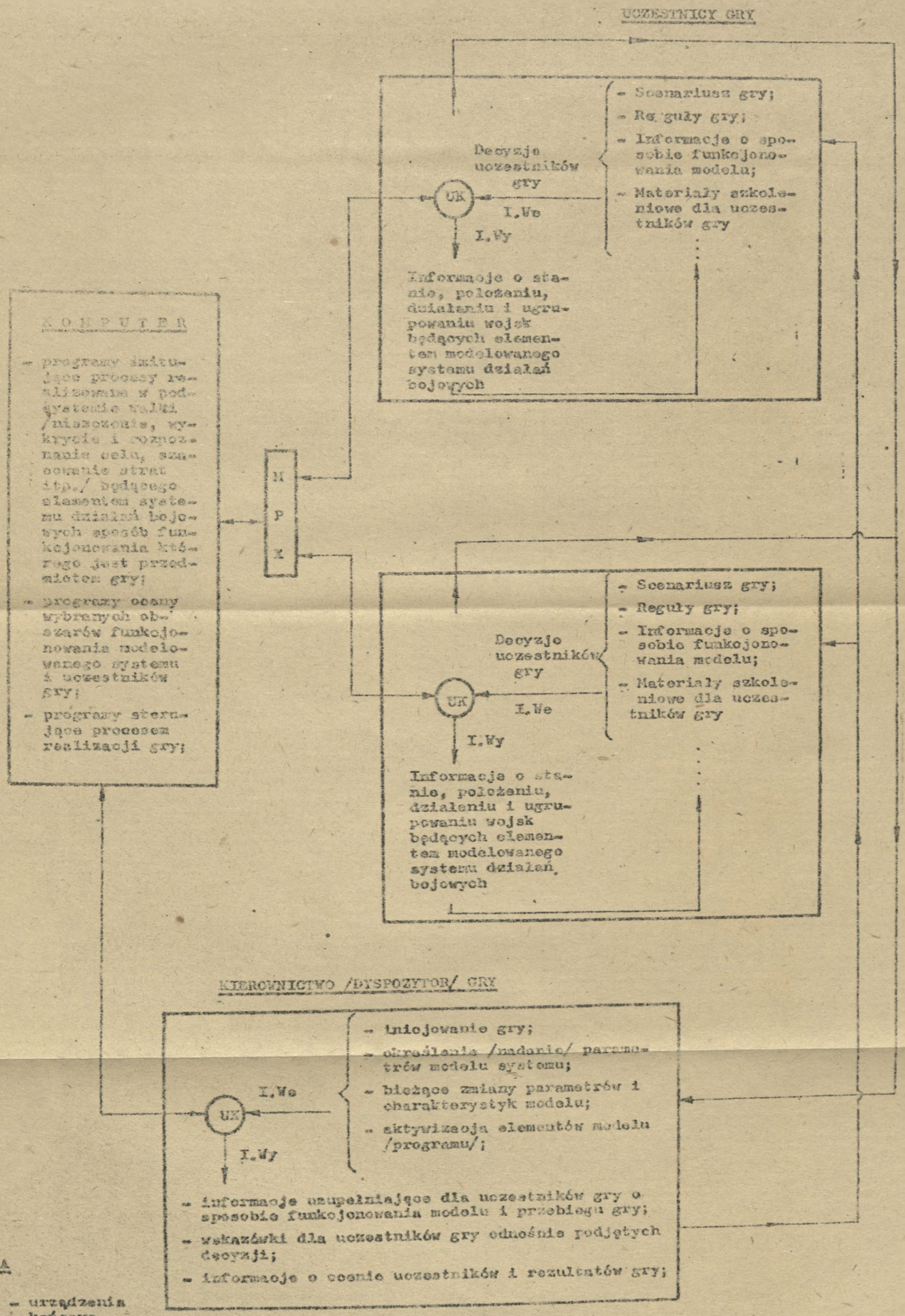
Podjęte przez uczestników gry decyzje są podstawą do przygotowania danych wejściowych do symulacyjnych programów komputerowych, których realizacja odpowiada trzeciemu z wymienionych etapów funkcjonowania modelowanego systemu - rozpoczyna się kolejny cykl gry, kolejny okres "komputerowej walki".

Po zakończeniu gry komputerowej kierownictwo gry dokonuje oceny jej uczestników.

Ogólny schemat funkcjonowania komputerowej gry wojennej wraz z opisem wybranych jej elementów przedstawiono na rys. 4.

1.8. Kierownictwo gry

W procesie projektowania komputerowych gier wojennych szczególną uwagę należy zwrócić na odwzorowanie wpływu otoczenia na działanie modelowanego systemu działań bojowych. Jednym z elementów szeroko rozumianego otoczenia jest podsystem dowodzenia nadrzędnego systemu działań bojowych. Odwzorowanie jego wpływu na działanie systemu będącego przedmiotem gry w większości projektowanych gier wojennych sprowadza się do opracowania procedur oceny jakości. rezultatów podejmowanych przez uczestników gry decyzji i sposobów dochodzenia do ich wypracowania, jak również oceny kwalifikacji uczestników gry i stopnia osiągania celów gry modelowanego systemu działań bojowych. W komputerowych grach wojennych wspomniane procedury oceny realizowane są często przez powołaną



UWAGA

UK - urządzenia końcowe /dalekopis, drukarka, monitor, czytnik; itp./

I.We - informacje wejściowe;

I.Wy - informacje wyjściowe;

MPX - multiplekser.

Rys. 4. Ogólny schemat funkcjonowania komputerowej gry wojennej

specjalnie grupę ludzi zwaną kierownictwem gry. Kieronictwo gry spełnia zasadniczą dwie funkcje. Z jednej strony stanowi element odwzorowujący wpływ podsystemu dowodzenia nadrzędnego systemu działań bojowych /procedury oceny uczestników gry i sposobu jej realizacji/ z drugiej strony steruje przebiegiem gry w aspekcie organizacyjno-programowym [4], [6], [7], [9], [12].

W komputerowych grach wojennych procedury oceny uczestników gry mogą być realizowane również przy pomocy specjalnych programów komputerowych. Ocena sposobów wypracowania decyzji i osiąganych przez uczestników gry rezultatów winna uwzględniać umiejętność przewidywania możliwych sposobów działania systemu w wyniku podjętych decyzji, jak również sposób rozumowania i uzasadniania gry i stopnia osiągania celu modelowanego systemu winny uwzględniać bardzo często znaczną liczbę czynników z których większość ma charakter losowy. Stwarza to dodatkowe trudności w konstrukcji i programowej realizacji formalnie poprawnych i odpowiadających celowi gry procedur oceny. Z tego też względu procedury oceny w grach komputerowych tylko w ograniczonym zakresie realizowane są programowo, natomiast zasadnicza ich część realizowana jest przez kierownictwo gry. Należy zauważyć, że nie istnieje aktualnie, ogólnie poprawna i użyteczna metody oceny uczestników gier kierowniczych. Dla większości projektowanych komputerowych gier procedury oceny opracowywano wyłącznie dla potrzeb danej gry [5], [13]. Procedury i kryteria oceny opracowywane są w oparciu o wyniki nadań poligonowych, oceny ekspertów i wyniki studiów operacyjno-taktycznych ze współczesnych i przyszłych wojen [5], [13].

1.9. Materiały szkoleniowe dla uczestników gry

Jednym z końcowych zadań projektowania komputerowych gier wojennych jest przygotowanie dokumentacji i materiałów szkoleniowych dla uczestników gry oraz przeprowadzenie je próbnej eksploatacji. Materiały szkoleniowe zawierać mogą między innymi opis celu, reguł i scenariusza gry, jak również opis funkcjonowania modelu systemu

i sposobu przygotowania danych wejściowych oraz korzystania z informacji powstałych w wyniku realizacji programów komputerowych. Znajomość przez uczestników gry materiałów szkoleniowych w znacznym stopniu decyduje o wysokiej efektywności realizacji celu gry.

Pozytywne wyniki próbnej eksploatacji komputerowej gry wojennej warunkują realizację kolejnej fazy prac związanej z przygotowaniem i realizacją gry. W ramach tego etapu prac należy dokonać wyboru uczestników i kierownictwa gry oraz zorganizować i przeprowadzić szkolenie wszystkich potencjalnych uczestników gry.

Projektowanie gry komputerowej należy zakończyć poszukiwaniem maksymalnego obszaru jej zastosowań.

Proces projektowania gier komputerowych odpowiednio skraca się dzięki zastosowaniu odpowiednich języków modelowania, pozwalających opisywać właściwości złożonego systemu przy pomocy pojęć i symboli używanych w systemie rzeczywistym.

2. Wykorzystanie komputerowych gier wojennych

Komputerowe gry wojenne stanowiące specyficzne "laboratorium" współczesnego pola walki mogą być wykorzystywane w sferze dydaktycznej, naukowo-badawczej i praktycznej działalności dowództw i sztabów.

W sferze dydaktycznej komputerowe gry wojenne spełniać mogą między innymi takie funkcje, jak:

- prezentacja koncepcji, zasad, metod i sposobów dowodzenia wojskami;
- doskonalenie nawyków dowodzenia wojskami oraz opanowanie metod rozwiązywania zadań w złożonych sytuacjach operacyjno-taktycznych;

- aktywizacja i intensyfikacja procesu nauczania.

W obszarze zastosowań naukowo-badawczych gry komputerowe spełniać mogą funkcję poznawczą, weryfikacyjną i formalizacyjną.

Poznawcza funkcja komputerowych gier wojennych ujawnia się z całą siłą w procesie projektowania gry, a w szczególności w trakcie opracowywania modelu wybranego systemu działań bojowych. Nigdy bowiem stan wiedzy o modelowanym systemie działań bojowych nie jest taki, by zespół projektowy znał w pełni całą specyfikę funkcjonowania systemu. Dlatego też proces projektowania jest ciągłym udokładnianiem i formułowaniem hipotez, koncepcji, teorii w obszarze funkcjonowania systemu. Poznawcze funkcje gier komputerowych przez wiele zespołów projektowych oceniano często wyżej niż rezultaty wykorzystania gier.

Weryfikacyjna funkcja komputerowych gier wojennych ujawnia się szczególnie w procesie przebiegu gry, kiedy to ścierają się różne koncepcje i hipotezy wkomponowane w "mechanizm" gry, jak również prezentowane przez jej uczestników.

Wyniki projektowania komputerowej gry wojennej oraz uzyskane przez jej uczestników spostrzeżenia podczas wielokrotnego wykorzystania gry, umożliwiają często formalny opis wybranych obszarów funkcjonowania systemu działań bojowych, jak również pozwalają na formalizację postępowania decydentów przy rozwiązywaniu niektórych zadań w złożonych sytuacjach operacyjno-taktycznych.

W praktycznej działalności dowództw i sztabów komputerowe gry wojenne spełniać mogą swoją funkcję jeżeli stanowić będą integralny element oprogramowania użytkowego polowego informatycznego systemu dowodzenia. Bowiem tylko wtedy w warunkach dostatecznej ilości czasu mogą być wykorzystane do uzasadniania zamiaru i planu działań oraz oceny decyzji podejmowanych w trakcie dynamiki działań bojowych. Przykłady takiego wykorzystania gier wojennych chociaż tradycyjnych, dostarcza nam historia minionych wojen.

Komputerowe gry wojenne mają wiele zalet. Główną ich zaletą jest stworzenie uczestnikom gry możliwości zapoznania się ze specyfiką procesu podejmowania decyzji, co ze względu na ryzyko zawarte w decyzjach nie jest łatwe w praktyce dowódczo-sztabowej.

Uczestnicy i kierownictwo komputerowych gier wojennych muszą jednak zdawać sobie sprawę z tego, że model symulacyjny działań bojowych nie pomaga w procesie szkolenia /sfera dydaktyczna/ w lepszym przedstawieniu lub przewidywaniu przyszłej rzeczywistości wojennej. Stwarza ona jedynie obiektywną sytuację, jako podstawę procesu dowodzenia. Sytuacje takie mogą nawet w przyszłości rzeczywiście nastąpić, jednak nie wolno ich uważać za istniejącą rzeczywistość.

Mając na względzie potrzebę ciągłego doskonalenia systemów dowodzenia, komputerowe gry wojenne należy uznać za jedną z bardziej efektywnych, przyszłościowo uzasadnionych metod badawczych.

LITERATURA

1. BARCZAK A., Komputerowe Gry Wojenne, Zeszyty Naukowe ASG WP, 2/5/78.
2. BARTON R., Wprowadzenie do symulacji i gier. Warszawa: WNT 1967.
3. EVANS G., i inni., Symulacja na maszynach cyfrowych, Warszawa: WNT 1973.
4. GRAHAM R.G., GRAY C.F.: Business games handbook. American Mangament Association, Inc., 1969.
5. HARRIS K., WEGNER L., Tactical Airpower in NATO Contingencies: A Joint Air - Battle /Ground - Battle Model TALLY/TOTEM, Rand 1974.
6. JAGER M., Wykorzystanie komputerów w grze wojennej. WPZ.6/1978.
7. JEMJELJANOW W., i inni., Mietod diełowych igr. Moskwa 1976.
8. KOZŁOWA O., RAZU W., Diełowyje igry i ich rol w powyszeni kwalifikacii kadrow, izd. "Znanije", Moskwa 1978.
9. Materiały 4-go sieminara socjalisticzieskich stran po probliemie imitacionnych igr. Warszawa 1977.
10. SCHINZER D., Gry wojenne w szkoleniu taktycznym sił lądowych Stanów Zjednoczonych. WPZ., 7,8/1978.
11. SIENKIEWICZ P., Koncepcja modelowania systemu dowodzenia. Zeszyty Naukowe ASG WP nr 2/3/77.

12. SKIBIŃSKI J., Gry kierownicze w wojsku. Warszawa: MON 1973.

13. WILSON A.: The bomb and the computer, Barrie and Rockliff the Cresset Press London, 1968.

PYTANIA KONTROLNE

1. Podać określenie komputerowej gry wojennej.
2. Scharakteryzować podstawowe elementy komputerowej gry wojennej.
3. Omówić podstawowe rodzaje informacji wykorzystywanych do modelowania systemu działań bojowych wojsk lądowych.
4. Omówić algorytm realizacji wybranych procesów funkcjonowania systemu działań bojowych.
5. Omówić sposób funkcjonowania komputerowej gry wojennej.
6. Podać wady i zalety komputerowych gier wojennych.

Druk. ASG WP OXV-8904, zam.nr 1638 z dn. 13.08.1981r.