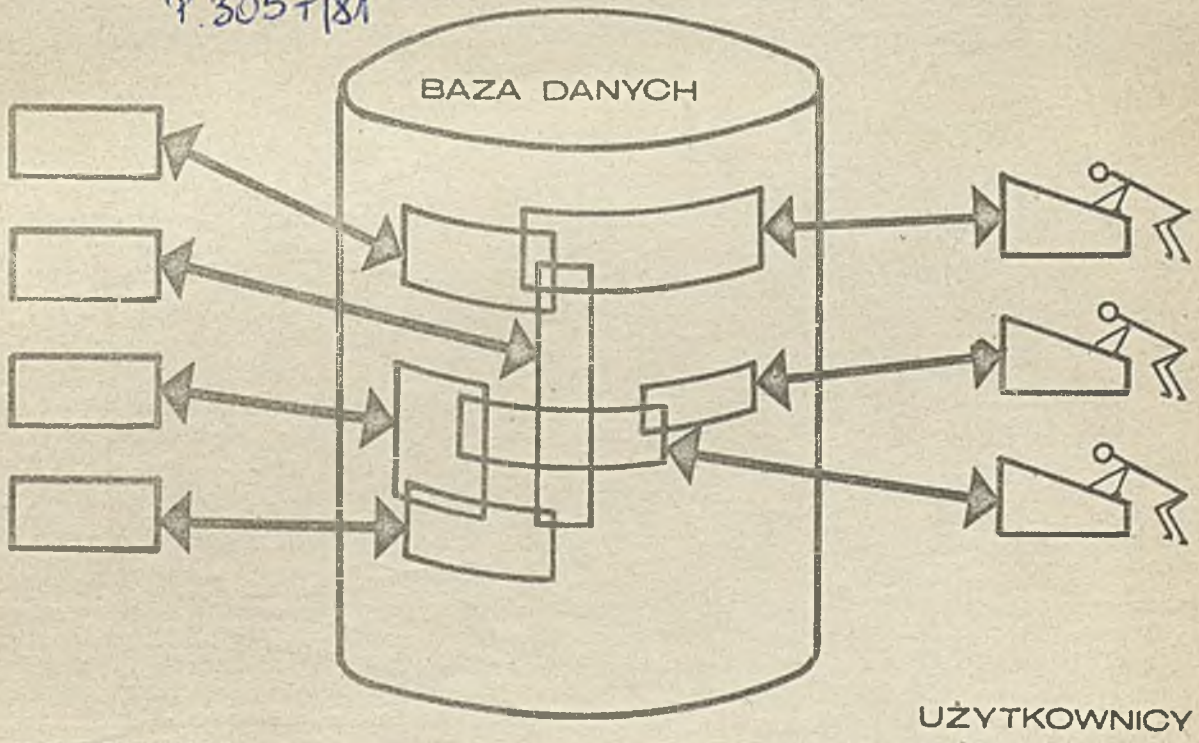
biuletyn informacyjny

1 '81

MERA IMM

OBIEKTOWE SYSTEMY KOMPUTEROWE



Zjednoczenie Przemysłu Automatyki i Aparatury Pomiarowej „MERA"
Instytut Maszyn Matematycznych „MERA IMM" Branżowy Ośrodek INTE

Rysunek na okładce zaczerpnięty z:
Date C.J.: An Introduction to Database Systems, Addison-Wesley 1975

Druk okładki IMM zam. 43/81 nakł. 820 egz.

Biuletyn Informacyjny OBIEKTOWE SYSTEMY KOMPUTEROWE

Rok XIX № 1 1981

Spis treści

Содержание

Contents

D W U M I E S I Ę C Z N I K

Wydaje:
CENTRUM NAUKOWO-PRODUKCYJNE TECHNIK KOMPUTEROWYCH I POMIARÓW
I N S T Y T U T M A S Z Y N M A T E M A T Y C Z N Y C H
Branżowy Ośrodek Informacji Naukowej Technicznej i Ekonomicznej

Adres redakcji: ul. Krzywickiego 34, 02-078 Warszawa
tel. 28-37-29 lub 21-84-41 w. 244

mgr inż. Włodzimierz Mardal
Instytut Maszyn Matematycznych

Systemy zarządzania bazą danych

Geneza powstania

Systemy zarządzania bazą danych zrodziły się na gruncie postępującej złożoności systemów informatycznych i rosnących trudności z prowadzeniem dużych zbiorów danych o tradycyjnej organizacji plikowej. Przypomnijmy, że ta do dziś stosowana jeszcze metoda organizowania danych polega na tworzeniu wielu kartotek obejmujących zapisy (rekordy) dotyczące zwykle obiektów jednego typu, przy czym ich budowa i rozkład w pamięci są podyktowane specyfiką zastosowania tj. profilem funkcjonalnym programów, które z tej kartoteki mają korzystać. W miarę projektowania nowych, nieprzewidzianych uprzednio, procesów przetwarzania tworzy się nowe kartoteki, zwykle pokrywające się częściowo z już istniejącymi. Rozbudowywany w ten sposób system informatyczny osiąga często znaczną liczbę plików o dużym stopniu redundancji danych. Powtórzenie tych samych, a zmiennych w czasie, danych prowadzi do sprzeczności między kartotekami, co z kolei powoduje poważne błędy w wynikach przetwarzania.

Taka technika organizowania danych była w dużym stopniu zdeterminowana możliwościami technicznymi wczesnych maszyn cyfrowych do przetwarzania danych. Wady tej techniki, a jednocześnie opanowanie nowych urządzeń pamięci masowych (pamięci dyskowe) spowodowały utworzenie nowej organizacji złożonych kolekcji danych. Początkowo zakładano, że organizacja ta powinna:

- umożliwić zintegrowanie danych tworzących dotychczas zestaw plików,
- ograniczyć lub wyeliminować zupełnie redundancję danych,
- zapewnić niezależność programów od danych, tj. taką właściwość, że rozszerzanie zawartości bazy oraz zmiany jej zapisu fizycznego nie naruszają funkcjonowania wcześniej opracowanych systemów.

Trzeba od razu powiedzieć, że te postulaty (może z wyjątkiem pierwszego) nie zostały dotychczas w pełni zrealizowane, mimo że na rynku komputerowym mamy już sporą liczbę systemów zarządzania bazą danych, a badania i prace implementacyjne w tej dziedzinie trwają już od ponad 15 lat. W trakcie tych prac sformułowano dalsze wymagania, jakie spełniać powinna nowa organizacja danych i związany z nią system programowy umożliwiający jej utworzenie oraz operowanie danymi, m.in. powstał problem ochrony danych przed zniszczeniem i niepowołanym dostępem, metod kontroli wewnętrznej zgodności i prawdziwości danych a także wiele innych.

Pojęcie bazy danych nie zostało dotychczas dobrze zdefiniowane. Określenia przytaczane w literaturze są zbyt ogólnikowe i nieprecyzyjne aby pozwalały odróżniać w praktyce co jest, a co nie jest bazą danych, np. w pracy "Data Base Management Systems" [1] napisano, że "System zarządzania bazą danych jest takim systemem programowym, który czyni pewną zbiorowość danych dostępną określonej zbiorowości użytkowników", zaś w pracy C.J. Date [2] pisze, że "baza danych są to zasoby danych niezbędne dla funkcjonowania systemu informatycznego w skali instytucji". Może najpełniej sens bazy danych i system zarządzania bazą oddano w "Auerbach Computer Technology Report" [4], gdzie bazę danych i system nią zarządzający określono następująco: "jest to pewna kolekcja danych przechowywana w pamięci o dostępie bezpośrednim, która ma następujące cechy:

- jest dostępna równocześnie dla wielu programów,
- jej elementami są zapisy (rekordy) logiczne,
- umożliwia dodawanie, usuwanie, pobieranie i modyfikację zarówno pojedynczych rekordów jak i grup rekordów pozostających w określonych związkach logicznych ze sobą".

Brak dobrej definicji bazy danych przyczynił się do dewaluacji tego pojęcia. Dziś mianem bazy danych określa się często niepoprawnie nawet pojedynczy plik danych najprostszego systemu informatycznego, zaś miano systemu zarządzania bazą danych nadaje się programom obsługi zwykłej, plikowej organizacji danych.

Struktura logiczna baz danych (model danych)

Pojęcie struktury logicznej bazy

Problem struktury logicznej złożonego kompleksu danych jest w istocie problemem opisu otaczającej nas rzeczywistości w kategoriach informatycznych. Zazwyczaj interesuje nas dość wąski wycinek tej rzeczywistości tj. taki obszar, który ma być domeną działania projektowanego systemu informatycznego. Ale nawet w takim wąskim wycinku znajdziemy wszystkie elementy ilustrujące pojęcie struktury logicznej nawet najbardziej złożonych baz danych. Są to mianowicie:

- różnorodne obiekty fizyczne (osoby, maszyny, materiały, książki itp.), które charakteryzują się właściwymi sobie zestawami cech (np. personalia, parametry itd.),
- związki między obiektami tego samego lub różnych typów (związki rodzinne, technologiczne, własności, autorstwa itd.),
- zdarzenia, zmieniające zarówno cechy poszczególnych obiektów jak i powiązania między nimi praz powodujące pojawienie się lub zanik jakiegoś obiektu w obszarze objętym bazą.

Struktura logiczna bazy danych powinna umożliwiać dokładny i jednoznaczny zapis co najmniej dwóch pierwszych kategorii elementów. Trzecia kategoria nie musi być uwzględniana, gdyż można ją potraktować jako czynnik wpływający na ewolucję zawartości bazy obejmującej tylko bieżący stan obiektów i powiązań między nimi. Dzięki temu rozmiary bazy nie rosną nazbyt szybko, co miałoby miejsce gdyby w bazie danych rejestrowano wszelkie zmiany zachodzące w opisywanej rzeczywistości. W konsekwencji jednak komplikują się wszelkie takie procesy przetwarzania, które wymagają "historycznego" spojrzenia na rozwój tej rzeczywistości.

Tak więc przez strukturę logiczną bazy danych należy rozumieć w zasadzie pewien zbiór reguł opisywania obiektów i ich właściwości oraz powiązań logicznych między nimi.

Określonej strukturze logicznej bazy odpowiada określony zestaw operacji, które mogą być wykonywane na elementach bazy danych. Służą one zarówno do zmiany zawartości bazy tj. wpisywania danych, ich zmiany i usuwania jak i do wyszukiwania i pobierania zestawów danych niezbędnych dla określonych zadań przetwarzania.

W teorii i praktyce systemów zarządzania bazą danych rozróżnia się dwie główne kategorie logicznych struktur bazy danych: strukturę sieciową i relacyjną. Ze względów historycznych wyodrębnia się też tzw. strukturę hierarchiczną, którą jednak można potraktować jako szczególny rodzaj struktury sieciowej.

Właściwości wymienionych dwóch kategorii struktur bazy można najlepiej zauważyć rozpatrując jakiś prosty przykład bazy. Rozważmy więc bazę danych, która opisuje zakłady badawcze pewnego instytutu, jego tematy prac, pracowników i pomieszczenia. Każdemu z wymienionych elementów można przyporządkować pewien zestaw cech oraz wskazać powiązania z innymi elementami bazy. Wiadomo więc, że:

- zakłady opracowują jeden lub kilka tematów,
- zakład zajmuje jedno lub kilka pomieszczeń,
- w jednym pomieszczeniu pracuje jeden lub wielu pracowników, ale tylko z jednego zakładu
- zakład zatrudnia wielu pracowników.

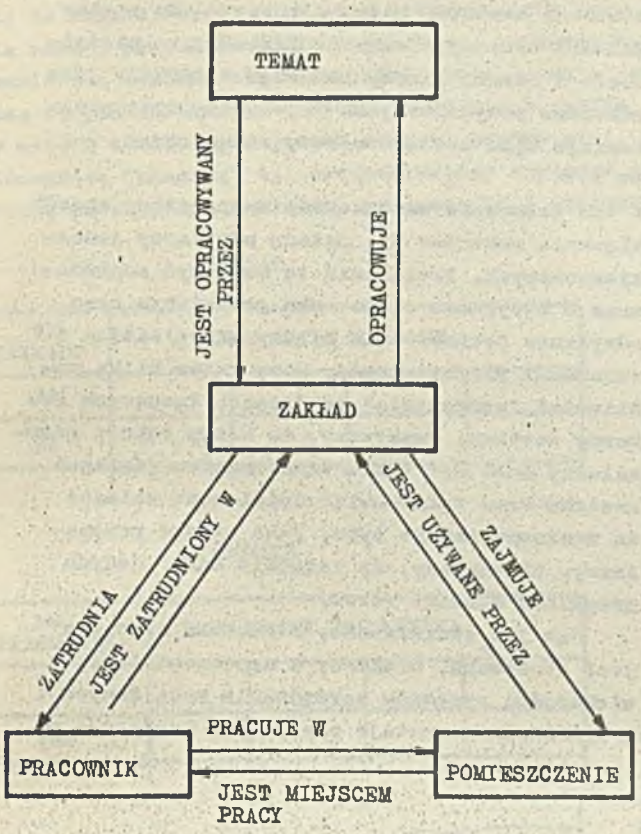
Można by tu wskazać jeszcze inne powiązania, np. powiązanie między tematami i pracownikami, stosunek podwładny - przełożony między pracownikami itd.

Rzecz prosta, obrany zespół powiązań w dużym stopniu determinuje właściwości użytkowe bazy. Podobnie rzecz się ma z wyborem cech opisujących poszczególne obiekty bazy. Dotykamy tu niezmiernie istotnego problemu projektowania bazy, tj. ustalenia struktury i zawartości bazy, stosownie do potrzeb konkretnego systemu informatycznego. Problem ten jest dużo bardziej złożony przy stosowaniu technologii baz danych niż przy organizacji plikowej danych..

Złożona zawartość bazy przykładowej składa się z powiązanych ze sobą rekordów ZAKŁAD, TEMAT, PRACOWNIK i POMIESZCZENIE. Schematycznie możemy tę bazę przedstawić jak na rysunku 1. Taki "kształt" bazy wykorzystamy teraz dla zilustrowania sieciowej i relacyjnej struktury bazy danych

Sieciowa struktura bazy

Koncepcja sieciowej struktury bazy ma już dość długą historię lecz dokładnie zdefiniowano ją dopiero na przełomie lat sześćdziesiątych i siedemdziesiątych. Był to wynik prac specjalnej grupy



Rys.1. Zależności między obiektami przykładowej bazy danych.

roboczej (Data Base Task Group) powołanej przez Komitet GODASYL (Conference on Data System Languages) utworzony na podstawie porozumienia kilkunastu instytucji przemysłowych, naukowych i rządowych USA [3]. Od tego czasu niemal wszystkie nowe implementacje systemów zarządzania bazą danych opierają się na strukturze sieciowej lub jej uproszczonych wersjach. Jej istota zasadza się na tym, że każdy związek między dwoma typami obiektów (np. zakład ⟶ temat) jest reprezentowany przez zestaw (w oryginalnej terminologii - "set") rekordów logicznych, który zawiera jeden rekord pełniący rolę właściciela (owner) zestawu i szereg rekordów przynależnych do zestawu (member). Zestaw ten jest budowany w ten sposób, że tworzy jak gdyby oko sieci. Każdy rekord przynależny może być z kolei właścicielem zestawu innego typu. Rekord należy tu rozumieć jako opis pewnego obiektu lub też po prostu pewną wyodrębnioną porcję danych. Przy układaniu bazy w kształt sieci obowiązują pewne ograniczenia:

- dany typ rekordu nie może partycypować w tym samym typie zestawu jako właściciel i rekord przynależny,
- konkretny rekord może należeć tylko do jednego zestawu rekordów danego typu.

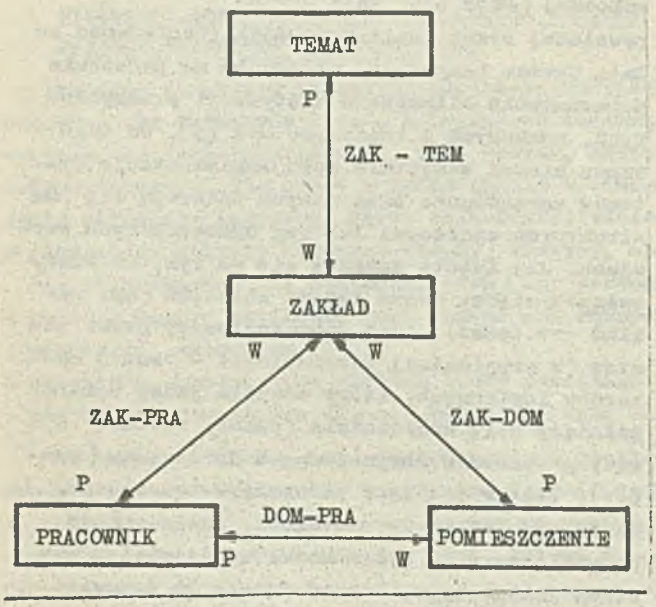
Ograniczenia te dla naszej bazy oznaczają, że nie można w prosty sposób przedstawić takiej sytuacji gdy jeden temat jest opracowywany przez kilka zakładów lub gdy w jednym pomieszczeniu znajdują się pracownicy różnych zakładów. Inaczej mówiąc, sieciowy model danych nie daje możliwości prostego zapisu związków typu M:N między rekordami dwóch różnych typów.

Z ograniczeń modelu sieciowego wynika także, że utrudnione byłoby uwzględnienie w naszej bazie związku przełożony-podwładny gdyż rekord PRACOWNIK musiałby wystąpić w odpowiednim zestawie w podwójnej roli, tj. jako rekord właściciel oraz jako rekord przynależny, co jest niedozwolone. (Zastrzegam się, że w tym miejscu opieram się na opracowaniach DBTG z lat 1971 i 1973. Być może późniejsze opracowania zniosły podane wyżej ograniczenia modelu sieciowego).

W konwencji sieciowej nasza przykładowa baza danych może zawierać następujące zestawy rekordów:

- zestaw "ZAK-TEM" zawierający rekord ZAKŁAD jako właściciela oraz szereg rekordów przynależnych TEMAT,
- zestaw "ZAK-PRA", w którym właścicielem jest rekord ZAKŁAD a przynależne są rekordy PRACOWNIK,
- zestaw "ZAK-POM", z rekordem ZAKŁAD jako właścicielem i rekordami POMIESZCZENIE jako przynależnymi,
- zestaw "POM-PRA" obejmujący rekord POMIESZCZENIE jako właściciela oraz rekordy PRACOWNIK jako przynależne.

Budowa zestawu rekordów umożliwia "ruch dwukierunkowy" tj. znając właściciela możemy bez trudu znaleźć wszystkie jego rekordy przynależne i odwrotnie - dla każdego rekordu przynależnego do danego zestawu można łatwo znaleźć jego właściciela oraz inne rekordy przynależne do zestawu. Zatem każdy zestaw reprezentuje związek wzajemny, choć poszczególne typy rekordów nie są równouprawnione. Schemat naszej przykładowej bazy można teraz przedstawić następująco:

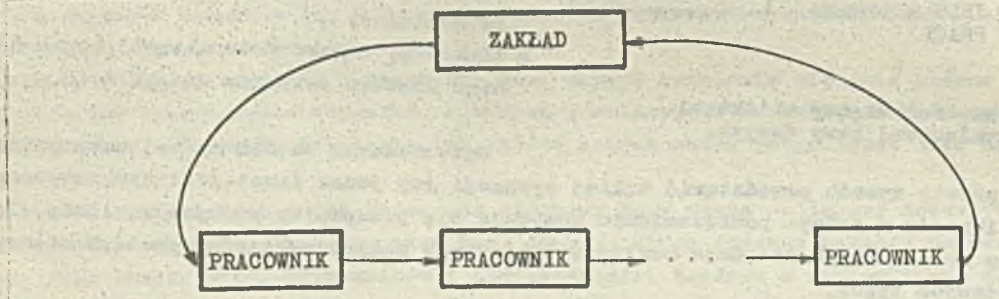


Rys.2. Przykładowa baza danych w konwencji sieciowej (literą W oznaczono rekord właściciela, P - rekord przynależny)

Przyjrzyjmy się teraz bliżej zasadom konstrukcji zestawu. Weźmy dla przykładu zestaw ZAK-PRA złożony z rekordu ZAKŁAD jako właściciela i pewnej liczby rekordów PRACOWNIK jako rekordów przynależnych. Pojedynczy egzemplarz takiego typu zestawu możemy zilustrować jak na rys.3.

W tym schemacie zastosowano najprostszy sposób wiązania rekordów tj. metodę odsyłaczy jednokierunkowych. Powiązania te mogą być rozbudowane o odsyłacze o kierunku przeciwnym oraz odsyłacze bezpośrednie między właścicielem i rekordami przynależnymi. Mamy zatem kilka możliwości "wędrowania" po drogach łączących rekordy zestawu. Zważywszy, że każdy rekord przynależny może być także właścicielem jakiegoś zestawu oraz rekord-właściciel może należeć do zestawów innego typu, jako rekord przynależny, zrozumiemy, że istotnie baza danych przybiera kształt sieci.

Jak już stwierdzono, struktura sieciowa jest stosowana, niekiedy z uproszczeniami, w większości systemów zarządzania bazą danych dostępnych obecnie na rynku programistycznym. Nadal właściwie pozostaje ona jedyną koncepcją w pełni zdefiniowaną.



Rys.3. Przykład pojedynczego egzemplarza zestawu rekordów

Relacyjna struktura bazy

Pomysł zastosowania matematycznego pojęcia relacji jako podstawy modelu logicznego bazy danych poddał jako pierwszy w r. 1970 E.F. Codd [5]. Mimo że temu modelowi przypisuje się wydatną wyższość nad modelem sieciowym, to jednak do tej pory nie zastosowano go w żadnym systemie, który miałby uznane znaczenie na rynku oprogramowania. Wiele natomiast systemów relacyjnych jest opracowywanych na zasadzie eksperymentów badawczych. Jak się wydaje, badań wymagają głównie pewne problemy implementacyjne, ale można też sądzić, że nie jest wcale oczywista łatwość posługiwania się samym modelem relacyjnym przy użytkowaniu bazy danych. Nie wdając się w wyjaśnienia teoretycznych podstaw modelu relacyjnego stwierdzimy, że prowadzi on do "tablicowego" widzenia bazy danych. Każdy element bazy tj. wyodrębniona w postaci rekordu porcja danych, jak również każdy związek między obiektami stanowi wiersz tablicy, obejmującej pewną liczbę kolumn reprezentujących elementarne dane, przy czym zarówno kolumny, jak i całe tablice mają swoje nazwy. Obowiązuje przy tym zasada, że żadna tablica nie zawiera dwóch identycznych wierszy. Taka struktura danych odróżnia się w sposób istotny od struktury sieciowej. O ile bowiem w strukturze sieciowej inaczej zapisuje się wartość jakiejś cechy obiektu (pole w rekordzie) a inaczej związek między obiektami (zestaw rekordów), o tyle w strukturze relacyjnej obydwa te typy informacji zapisywane są identycznie - w tablicy o odpowiednim układzie kolumn. Związek między jakimiś

obiektami zapisuje się w ten sposób, że w jednym wierszu, w odpowiednich kolumnach tablicy służącej do zapisu faktów tego typu umieszcza się dane jednoznacznie identyfikujące "wiązane" obiekty. Dla naszej przykładowej bazy założymy, że każdy typ obiektu (temat, zakład, pracownia, pomieszczenie) ma odpowiednią numerację, która pozwala jednoznacznie identyfikować poszczególne obiekty. Bazę tę przedstawimy w postaci 6 tablic: 4 tablic opisujących cechy poszczególnych obiektów bazy oraz 2 tablic opisujących związki między nimi, przy czym pierwsza z nich opisuje relację dwuelementową (binarną) tj. związek między zakładem i tematem zaś druga relację trójelementową zbudowaną na zbiorach zakładów, pomieszczeń i pracowników.

ZAKŁAD

NR ZAKŁADU	INNE CECHY ZAKŁADU			
Z1				
Z2				

TEMAT

NR TEMATU	INNE CECHY TEMATU			
T1				
T2				

PRACOWNIK

NR PRACOWNIKA	INNE CECHY PRACOWNIKA			
PR1				
PR2				

POMIESZCZENIE

NR POMIESZ-CZENIA	INNE CECHY POMIESZCZENIA			
P1				
P2				

ZAK-TEM

NR ZAKŁADU	NR TEMATU
Z1	T5
Z2	T3
Z1	T7
Z2	T7

ZAK-POM-PRA

NR ZAKŁADU	NR POMIESZ-CZENIA	NR PRACOWNIKA
Z1	P7	PR4
Z1	P7	PR8

Rys.4. Przykład bazy w konwencji relacyjnej

Już na tym prostym przykładzie można zauważyć pewne specyficzne właściwości relacyjnego podejścia do bazy danych. Ujednolicenie formy zapisu cech obiektów i ich powiązań z innymi obiektami powoduje wzrost liczby typów rekordów logicznych. Ponadto rośnie liczba egzemplarzy tych rekordów. Zauważmy bowiem, że jeżeli jakiś obiekt ma jakąś cechę wielowartościową jak np. imię pracownika lub datę awansu to dla zapisu wszystkich jej wartości trzeba wypełnić szereg wierszy tablicy, różniących się tylko wartością w kolumnie odpowiadającej danej wielowartościowej. Z tego można wyciągnąć wniosek, że model relacyjny prowadzi do większej redundancji danych w porównaniu z modelem sieciowym. Tak istotnie jest jeśli chodzi o poziom logiczny dostępu do danych. Nie musi to jednak oznaczać fizycznego wielokrotnego zapisu tych samych danych. Należy sądzić, że właśnie tego typu problemy, tj. problemy fizycznej implementacji bazy o "kształcie" relacyjnym, są nadal w stadium badań.

Model relacyjny ma wielu gorących zwolenników lecz większość specjalistów wypowiada się w tej sprawie dość ostrożnie. Można powiedzieć, że dominuje przekonanie, iż na obecnym etapie rozwoju maszyn cyfrowych i oprogramowania obydwie te koncepcje będą miały swoje obszary zastosowań, gdyż żadna z nich nie dominuje nad drugą we wszystkich charakterystykach bazy danych.
Do zalet koncepcji relacyjnej zaliczyć niewątpliwie należy:

- łatwość zapisu relacji złożonych (m:n), które w modelu sieciowym zapisuje się z pewnymi kłopotami,
- wysoki poziom języka manipulacji danymi sprzyjający niezależności struktury bazy od jej zastosowań.

Struktura sieciowa natomiast daje m.in. możliwość wyeliminowania redundancji danych oraz wpływania użytkownika na efektywność dostępu do danych.

Obydwa typy struktur logicznych stanowią obecnie swoistą klasykę teorii struktur danych o najwyższym stopniu złożoności. Nie oznacza to wcale, że istniejące systemy zarządzania bazą danych stosują się ściśle do tych wzorcowych rozwiązań. Większość z nich stosuje struktury stanowiące uproszczenie struktur klasycznych, głównie sieciowej. Jest także wiele systemów o charakterze wyspecjalizowanym, które posługują się strukturami logicznymi baz ściśle przystosowanymi do konkretnych sfer zastosowań. Rozwój struktur danych jak również systemów zarządzania bazą danych podlega bowiem tym samym prawom, które dotyczą rozwoju wszelkich narzędzi programowych. Rozwijają się one na zasadzie przeplatających się dwóch nurtów tj. implementacji wielu systemów wyspecjalizowanych oraz coraz ogólniejszych systemów uniwersalnych, bazujących na doświadczeniu z rozwiązywania sytuacji szczególnych. W tym sensie zapewne obecne klasyczne struktury baz danych są podsumowaniem naszej dotychczasowej wiedzy o danych. W przyszłości możemy się spodziewać takich nowych typów danych, które nie dadzą się zawrzeć w znanych nam dzisiaj konstrukcjach logicznych.

Budowa systemów zarządzania bazą danych

Najogólniej rzecz biorąc, system zarządzania bazą danych jest to zespół środków programowych umożliwiający tworzenie i użytkowanie bazy danych o określonej strukturze. W większości wypadków stanowią one rozszerzenie ogólnie stosowanych języków programowania w formie dodatkowych zdań, procedur i makrorozkazów. Takie rozwiązanie zapewnia bezpośredni "styk" między bazą a programami przetwarzającymi. Bywają jednak także inne rozwiązania, w których zarówno dla tworzenia bazy jak i dla jej wykorzystania konstruuje się języki specjalne. Całość środków programowych systemu zarządzania bazą danych dzieli się na dwie grupy:

- środki tworzenia struktury bazy (data description language - DDL), służące do opisu konstrukcji całej bazy oraz określenia zakresów jej dostępności dla poszczególnych grup użytkowników,
- środki manipulowania danymi (data manipulation language - DML), służące do wprowadzania danych do bazy, ich zmiany, usuwania i odczytywania zawartości bazy.

Środki konstruowania bazy(DDL) stanowią główne narzędzie pracy zespołu zwanego administratorem bazy, który wypełnia wiele funkcji w trakcie tworzenia i eksploatacji bazy. Ogólnie czynności tego zespołu można ująć następująco:

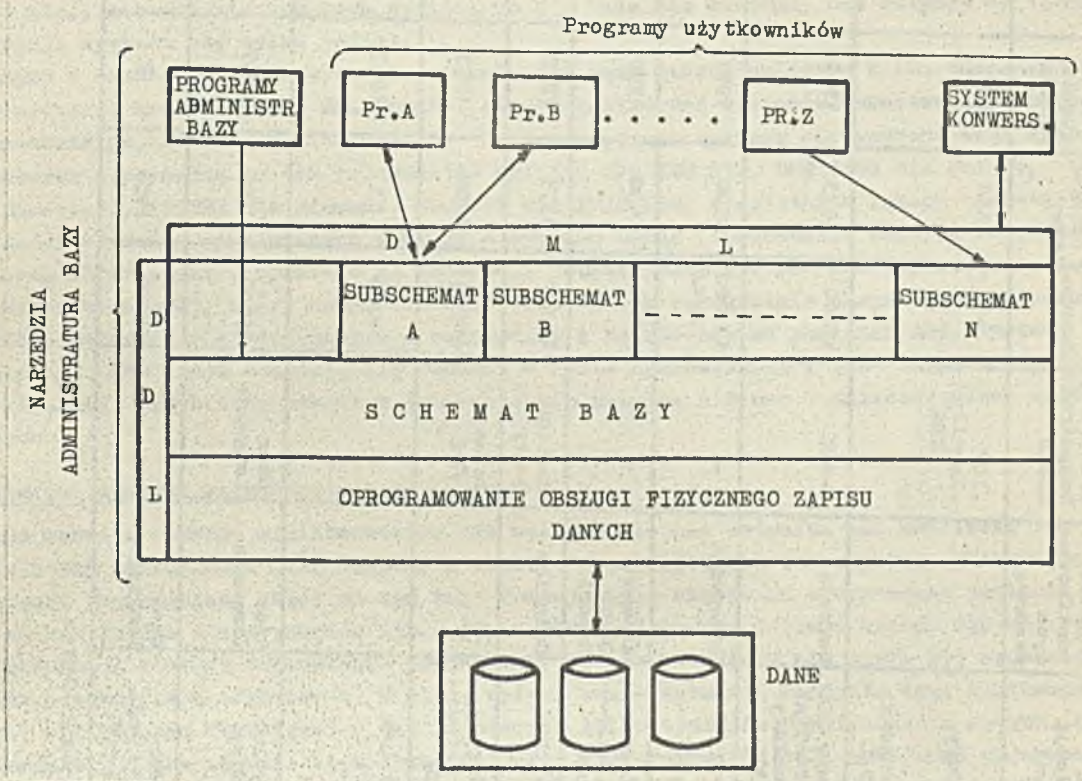
- sporządzenie i wprowadzenie do systemu opisu struktury bazy i jej zawartości (schematu bazy) oraz subschematów określających zakres i sposób widzenia bazy dla poszczególnych grup użytkowników,
- sporządzenie opisu i realizacja fizycznego zapisu bazy danych najlepiej przystosowanego do sposobu użytkowania bazy, a także modyfikacja zapisu fizycznego w razie zmian metod eksploatacji,
- kontrola stanu bazy tj. stosowanie specjalnych procedur (programów) weryfikacji poprawności danych zawartych w bazie,
- opracowanie i stosowanie procedur ochrony bazy przed zniszczeniem i niepożądanym dostępem,
- współpraca z użytkownikami bazy w celu zapewnienia poprawności ich współpracy z bazą.

Drugą grupę środków składających się na system zarządzania bazą danych stanowią narzędzia do programowania wszelkich działań na zawartości bazy bez naruszania jej struktury.

Systemy zarządzania bazą danych wyposaża się zwykle w oprogramowanie dodatkowe, które ułatwia ich wdrożenie i obniża koszty eksploatacji. W skład tego oprogramowania mogą wchodzić:

- pakiet zapytań formułowanych ad hoc i realizowanych wsadowo,
- system bezpośredniego, konwersacyjnego dostępu do bazy dla użytkowników nie będących programistami,
- pakiet programów wykonujących niektóre czynności administratora bazy.

Główne elementy systemu i jego powiązania zewnętrzne można przedstawić schematycznie następująco:



Rys.5. Struktura systemu zarządzania bazą danych i jego powiązania z użytkownikami

System zarządzania bazą danych można w dużym stopniu określić, definiując jego języki DDL i DML. Ich możliwości determinują zarazem możliwości wewnętrznych mechanizmów systemu. W językach tych przede wszystkim odbijają się możliwości konstrukcji kształtu bazy i jej elementów oraz możliwości wykorzystywania bazy. Na ocenę wartości konkretnego systemu ma także wpływ jego dodatkowe wyposażenie programowe, a także zakres i jakość świadczeń dostawcy oraz cena systemu. W sumie tworzy to dużą liczbę czynników, co powoduje, że wybór systemu dla konkretnego zastosowania spośród wielu ofert nie należy do zadań łatwych.

Krótki przegląd systemów zarządzania bazą danych dla m.o. IBM 360/370

Liczby wszystkich opracowanych na świecie systemów dla tych maszyn nie można ustalić na podstawie dostępnych materiałów. Trudno też znaleźć dane na temat liczby instalacji systemów zarządzania bazą danych na maszynach tego typu. Można jedynie podać na podstawie pracy Bakera z r. 1979 [6], że ogólną liczbę instalacji systemów różnych typów na różnych maszynach szacuje się na ok. 6000, z czego ok. 5000 przypada na 7 najbardziej rozpowszechnionych systemów. Główne charakterystyki tych systemów i dwóch dalszych zestawiono w tab. 1. Dane o systemach pochodzą z różnych źródeł i w wielu wypadkach nie są zbyt pewne, gdyż w literaturze technicznej spotyka się informacje wysoce nieprecyzyjne, a niekiedy nawet sprzeczne. Większość danych pochodzi z publikacji, które ukazały się w 1978 r.

Ceny zakupu (dzierżawy) starano się podać dla takiej wersji systemu, którą opracowano dla m.o. IBM 360/370 z systemem operacyjnym OS, przy czym dla wszystkich systemów poza ADABAS cena nie obejmuje dodatkowych pakietów i opcji. W praktyce nabywca musi zakupić również dodatkowe wyposażenie programowe.

TAB.1. SYSTEMY ZARZĄDZANIA BAZĄ DANYCH DLA MASZYN IBM 360/370 /OS/

Lp.	Nazwa systemu	Dostawca	Struktura logiczna bazy	Narzędzie opisu i konstrukcji bazy	Język programowania manipulacji danymi	Pakiet zapytań realizowanych wsadowo	Pakiet dostępu bezpośredniego	Cena x/		Rok rozpoczęcia DKSTR.	Liczba instalacji xx/
								Zakupu	Dzierżawy		
1	IMS-2	IBM	hierarchiczna	DL/I /procedury COBOL, PL/1 makrorozkazy BAL/	DL/I procedury COBOL, PL/1 Makrorozkazy BAL	GIS	.IQF .Inquiry/ /IMS	-	646	1968	1500
2	TOTAL	CINCOM SYSTEMS	sieciowa uproszczona	własny DDL	procedury COBOL, PL/I, FORTRAN makrorozkazy BAL	Socrates	T-ASK	41000	1025	1969	2000
3	ADABAS	SOFTWARE AG	zestaw plików z powiązaniami pośrednimi	własny	własny	Adascript	-	132000 xxx/	2500	1971	200
4	SYSTEM 2000	MRI SYSTEMS	hierarchiczna	własny	rozszerzone języki uniwersalne	Access Queue Report	-	35000	1430	1970	700
5	IIMS	CULLINANE CORP	sieciowa	Codasyl DDL	Codasyl DML	Culprint	On-Line Query	45000	1575	1973	400
6	DATACOM/DB	APPLIED DATA RESEARCH	zestaw plików	własny	procedury języków uniwersalnych	-	Datacom/ /DQ	40000	1200	1973	120
7	RAMIS II	MATHEMATICA PRODUCTS GROUP	hierarchiczna	własny lub procedury języków uniwersalnych	własny lub procedury	?	?	34000	?	1967	200
8	INQUIRE	INFODATA SYSTEMS	zestaw plików z powiązaniami pośrednimi	własny lub procedury języków uniwersalnych	własny lub procedury	User Language	User Language	65000	500	1969	100
9	MODEL 204	COMPUTER CORPORATION of AMERICA	zestaw plików z powiązaniami bezpośrednimi	procedury języków uniwersalnych	procedury języków uniwersalnych	User Language	?	36000	1450	?	?

x cena minimalna, dotyczy samego systemu i nie obejmuje programów dodatkowych
xx dane z r. 1978 obejmują użytkowników systemu na różnych maszynach
xxx cena systemu wraz z podstawowymi pakietami

Liczba instalacji obejmuje implementacje różnych wersji systemów na różnych maszynach. Ocenia się, że liczba ta rośnie w tempie 45 % rocznie [6]. Jeśli ta tendencja się utrzyma to w r. 1984 liczba instalacji systemów wyniesie ok. 50000. W literaturze brak jest właściwie danych o kosztach opracowania systemów tego typu. Jedynie dla systemu TOTAL podano, że jego opracowanie kosztowało ok. 20 mln., a koszt wiersza kodu wyniósł 70 $, ale nie wiadomo, czy dotyczy to łącznie wszystkich wersji systemu czy tylko jednej.

Na tle danych o zachodnim rynku systemów zarządzania bazą danych należy w kilku zdaniach scharakteryzować sytuację krajową w tej dziedzinie. Liczbę zastosowań systemów importowanych należy, jak sądzę, szacować na kilka bądź kilkanaście. Są to zazwyczaj systemy sprowadzane wraz z odpowiednim komputerem i pochodzą od ich producentów np. IMS dla IBM 370, DMS 1100 dla Univac, IDS dla maszyny Honeywell, PRISMA dla Siemens, SOCRATE dla IRIS itd. W ostatnich latach rozpoczęto dystrybucję dwóch systemów opracowanych w kraju - systemu RODAN (opracowanie Centrum Projektowania i Zastosowań Informatyki), opartego na koncepcji CODASYL oraz systemu SAD (Opracowanie Instytutu Maszyn Matematycznych), który można określić jako system zarządzania danymi zorganizowanymi w postaci zbioru rekordów niejednorodnych z pośrednimi i bezpośrednimi powiązaniami. Systemy te opracowane dla m.c. IBM i R32 znajdują się obecnie w fazie opanowywania i prób eksploatacyjnych u wielu użytkowników. Przyszłość pokaże w jakim stopniu przyjmą się one w zastosowaniach systemów informatycznych.

Minikomputerowe systemy zarządzania bazą danych

W pierwszym okresie rozwoju minikomputerów nie występowała ani potrzeba ani możliwość ich wyposażenia w systemy zarządzania bazą danych, a nawet w oprogramowanie dla zwykłej, plikowej organizacji danych. Przeważająca część maszyn tego rodzaju była stosowana w systemach kontroli i sterowania, automatyzacji eksperymentów itp., gdzie nie występowały większe zbiory danych. Ponadto stan techniki i względy ekonomiczne powodowały, że minikomputery nie mogły być wyposażone w duże pamięci operacyjne i zewnętrzne. W miarę upływu czasu sytuacja zmieniła się. Minikomputery stawały się atrakcyjnym rozwiązaniem dla niektórych typów systemów przetwarzania danych. Dostawcy minikomputerów, jak również niezależni wytwórcy oprogramowania jęli wzbogacać oprogramowanie zestawów minikomputerowych. Dziś mamy do dyspozycji już nie tylko oprogramowanie organizacji plikowej danych dla większości minikomputerów, ale dla wielu z nich całkiem rozbudowany systemy zarządzania bazą danych. Zakres funkcjonalny systemów jest bardzo zróżnicowany w zależności od parametrów minikomputera, np. najbardziej rozbudowany system dla minikomputerów PDP11-DBMS11 wymaga zestawu PDP11/70 i zajmuje 128k słów pamięci operacyjnej.

Rozwój minikomputerowych systemów zarządzania bazą danych ma pewną specyfikę w porównaniu z systemami dla maszyn dużych. Te ostatnie, zorientowane głównie na przetwarzanie danych, są budowane w ten sposób aby dane pobierane z bazy mogły być przetwarzane za pomocą istniejących języków programowania. Z tego też powodu język manipulacji danymi realizuje się tak, aby uzyskać dostęp do danych z programu napisanego w dowolnym języku programowania. Dla tzw. użytkowników końcowych bazy danych opracowuje się specjalne pakiety, które pozwalają uzyskiwać dane bazy bezpośrednio i łatwo, lecz przy małym stopniu przetworzenia. Inaczej rzecz się ma nieco, jak się wydaje, z systemami dla minikomputerów. Tu baza danych jest traktowaną częściej jako komputerowo prowadzona ewidencja niż jako podstawa systemu przetwarzania danych. Najlepszym przykładem takiego podejścia jest system MUMPS-11, który zawiera nawet specjalny system operacyjny i język zorientowany na utworzenie i prowadzenie bazy o określonej strukturze logicznej. Dla prowadzenia bazy typu ewidencyjnego minikomputery mają pewną przewagę nad maszynami dużymi. Główne zastosowania minikomputerów tj. systemy sterowania i pomiarów ukształtowały bowiem tak organizację i oprogramowanie podstawowe tych maszyn aby zapewnić wysoką sprawność wymiany danych ze światem zewnętrznym, a więc z urządzeniami zewnętrznymi minikomputera. Właściwość ta jest bardzo przydatna przy prowadzeniu bazy z bezpośrednim dostępem. Współpraca z bazą danych ulokowaną na dysku polega przecież w głównej mierze na wymianie danych między procesorem a dyskiem oraz między procesorem a terminalami użytkowników. W tabl. 2 zestawiono główne systemy zarządzania bazą danych zrealizowane na minikomputerach PDP11. Ich charakterystyki funkcjonalne są bardzo zróżnicowane, co wyraża się również w cenie systemów. Stosunkowo niewielką liczbę ich instalacji można wytłumaczyć po części tym, że są to produkty dość nowe a zastosowania minikomputerów w systemach ewidencyjnych mają niedługą historię.

TAB.2. SYSTEMY ZARZĄDZANIA BAZĄ DANYCH DLA MASZYN PDP 11

Lp.	Nazwa systemu	Dostawca	Struktura logiczna bazy	Język opisu i konstrukcji bazy	Język programowania manipulacji danych	Wymagany system operacyjny	Wymagany model maszyny	Wymagana pamięć operacyjna	Cena $		Liczba instalacji
									Zakupu	Dzierżawy	
1	DBMS11	DIGITAL EQUIPMENT CORPORATION	sieciowa	CODASYL DDL	CODASYL DML	IAS	11/70	128KS	16500	-	60
2	TOTAL	CINCOM SYSTEMS	sieciowa uproszczona	własny	procedury wywoływane przez CALL	RSX-11M RSX-11D IAS RSTSIE	11/34--11/70	15-18 KB	18500	500	?
3	ADABAS/M	SOFTWARE AG	zestaw plików z powiązaniami pośrednimi	własny specjalny	procedury wywoływane zdaniem CALL	RSX-11 lub IAS	11/35-11/70	?	40000	-	?
4	DATA BOSS/2	FLORIDA COMPUTERS	hierarchiczna	BASIC-PLUS	BASIC-PLUS	RSTS/E	11/35--11/70	28KS	14000	?	75
5	PRODUCT-3	ELS SYSTEMS ENGINEERING	zestaw plików z powiązaniami	procedury wywoływane przez CALL z dowolnego języka	procedury wywoływane przez CALL z dowolnego języka	RSX-11/D, RSX-11/M, RSTS/E UNIX	?	32KB	7500	3500	190
6	ADMINS/11	ADMINS INC	zestaw plików z powiązaniami	własny	własny	RSX-11D	?	?	50000	ROCZNIE 20000	50
7	MUMPS/11	DIGITAL EQUIPMENT	hierarchiczna	BASIC--MUMPS	BASIC-MUMPS	MUMPS	11/10-11/70	16KS	8250	-	?

Literatura

[1] Data Base Management Systems.
QED Information Sciences Inc. Wellesley 1978

[2] Date C.J.: An Introduction to Database Systems.
Addison - Wesley 1975

[3] CODASYL Data Base Task Group Report, April 1971

[4] Auerbach Computer Technology Reports. Data Base Management Systems.
Auerbach Publishers 1977-78

[5] Codd E.F.: A Relational Model of Data for Large Shared Data Banks.
Communications of the ACM 1970 nr 6

[6] Baker G.: Database Software - a wide Choice. Data Processing 1979 nr 4

[7] Zornes J.A. Jr.: Data base management systems on minicomputers. Data Base 1977 nr 1

[8] Boylan D.T.: Minicomputer DBMS. Computer Decisions 1977 nr 1

[9] ACM Computing Surveys. Special Issue. Data Base Management Systems 1976 nr 1.

mgr inż. Zbigniew Poznański
Instytut Technologii Elektronowej

SIMULA-67 - uniwersalny język programowania. Cz.3

W poprzednich dwóch częściach opisu języka SIMULA-67 przedstawiono jego podstawowe pojęcia tj. pojęcie klasy, prefiksowania klas i bloków, wielkości wirtualne, a także zaprezentowano systemową klasę SIMSET, która umożliwia w tym języku operacje na zbiorach implementowanych za pomocą struktur listowych.

Język SIMULA-67 jest wyposażony również w silny aparat do operacji na znakach i tekstach, mechanizmy umożliwiające dokonywanie operacji wejścia/wyjścia (klasa BASICIO) oraz wiele zdefiniowanych w nim generatorów liczb pseudolosowych. Problemy te szczegółowo omówiono poniżej.

Operacje na znakach i tekstach

Szerokie możliwości SIMULI-67 w zakresie operacji na znakach i tekstach nabierają szczególnego znaczenia w takich zastosowaniach jak: edycja dokumentów, przetwarzanie danych handlowych, danych dla potrzeb gospodarki materiałowej i magazynowej, wyszukiwanie informacji, banki danych, itp. Język ten można tu wykorzystywać nie tylko do tworzenia programów użytkowych ale także do opisu i symulacji systemów związanych z powyższymi zagadnieniami.

Operacje na znakach

Znaki stanowią wartości zmiennych typu character. Znakiem może być litera (A÷Z), cyfra (0÷9) lub symbol (np.=,+,:) . W zależności od implementacji znaki mają następującą postać, np. reprezentacja litery A:

'A' - (EMC CYBER 72),
"A" - (EMC IRIS 80). *)

Zmienne i tablice typu character deklarowane są w następujący sposób:

```
character  znak,symbol,cyfra;
character array  dokument (1:10);
```

Wartości zmiennym tego typu nadawane są przez zwykłe instrukcje podstawienia, np.

```
cyfra := "9";
symbol := znak;
```

Każdej wartości zmiennej typu character (znakowi) odpowiada liczba naturalna. Odwzorowanie to jest charakterystyczne dla określonej implementacji, np.

```
"=" - 126          "1" - 241
"A" - 193           :
 :                  .
"Z" - 233          "9" - 249 .
```

W SIMULI-67 zdefiniowano dwie procedury funkcyjne char i rank , które realizują wspomniane odwzorowanie, jak również odwzorowanie odwrotne. Niech C będzie zmienną typu character, CN - liczbą naturalną odpowiadającą wartości zmiennej C.

- character procedure char (CN) ; integer CN; wyznacza wartość zmiennej C odpowiadającą liczbie naturalnej CN. Gdy liczbie CN nie odpowiada żaden znak - sygnalizowany jest błąd.
- integer procedure rank(C); character C; wyznacza liczbę naturalną CN odpowiadającą wartości zmiennej C.

---

*) W pracy przyjęto taką postać znaku jak w EMC IRIS 80.

Prawdziwe zatem zawsze są związki:

C = char (CN)
CN = rank (C) .

Na przykład rank ("A") = 193, char (233) = "Z".

Wspomniane procedury umożliwiają porównywanie wartości zmiennych typu character (znaków) z wykorzystaniem operatorów relacji. Na przykład zmienna boolowska bool

bool := "I" > "B"

przyjmuje wartość true ponieważ rank ("I") > rank ("B") .
Widać stąd, że porównywanie znaków sprowadza się do porównywania odpowiadających im wartości liczbowych.

W Simuli zdefiniowano ponadto dwie procedury boolowskie:

- boolean procedure digit (C) ; character C; ,
  która przyjmuje wartość true, jeśli wartością zmiennej C jest cyfra tzn. "0" ÷ "9" .
- boolean procedure letter (C) ; character C; ,
  która przyjmuje wartość true jeśli wartością zmiennej C jest litera tzn. "A" ÷ "Z" .

Rozważmy następujący przykład. Niech będzie dany tekst o długości 30 znaków. Należy wczytać go znak po znaku oraz wydrukować liczbę znaków "A" oraz liczbę cyfr występujących w tym tekście. Instrukcją wczytującą wartości zmiennych typu character jest procedura inchar. Instrukcją wyprowadzającą wartości zmiennych typu całkowitego jest wywołanie procedury outint(a,b), gdzie a - wyprowadzana wartość, b - wielkość pola.
Mamy zatem:

```
begin
  integer i,liczbacyfr,liczbaliterA;
  character C;
  for i:=1 step 1 until 30 do
  begin
    C:=inchar;
    if C = "A" then liczbaliterA:=liczbaliterA + 1;
    if digit(C) then liczbacyfr:=liczbacyfr + 1;
  end;
  outint (liczbaliterA,6);
  outint (liczbacyfr,6);
end;
```

Operacje na tekstach

Zmienne tekstowe

Wartością zmiennej typu text jest łańcuch znaków w postaci, np.
'System sterowania procesem wytwarzania' .

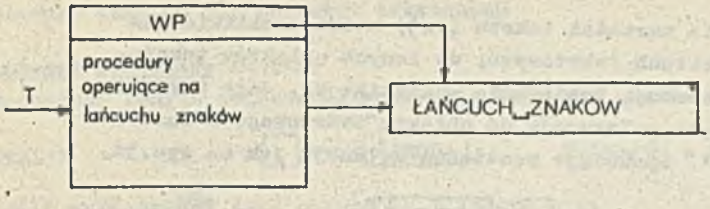
Zmienne tekstowe deklaruje się w następujący sposób:

```
text  wiersz,kolumna,tytuł;
text array  lista(1:100);  .
```

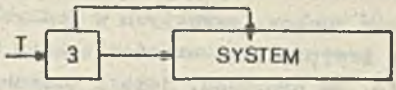
Aby umożliwić operacje na tekstach wykorzystano pojęcie obiektu należącego do pewnej fikcyjnej klasy, w której zdefiniowano operacje na łańcuchach znaków. Wartością zmiennej tekstowej jest referencja do tego obiektu. Obiekt ten zawiera:

- łańcuch znaków (wartość obiektu tekstowego),
- wskaźnik pozycji (WP) dostępnego znaku w łańcuchu znaków obiektu,
- procedury realizujące operacje na łańcuchu znaków obiektu.

Reprezentację obiektu tekstowego T ilustrują na przykład rys. 31 i 32.



Rys.31



Rys. 32

Początkową wartością zmiennej tekstowej jest notext . Zmienna ta wskazuje wtedy na obiekt tekstowy zawierający pusty ciąg znakowy.
Należy podkreślić, że typy character i text nie są kompatybilne, tzn. text T; character C;.
T:= "A"; C := 'A'; jest błędne.

Tworzenie obiektów tekstowych

Obiekty tekstowe mogą być tworzone za pomocą jednej z dwóch systemowo zdefiniowanych procedur, tj. blanks i newtext*).

text procedure blanks(n); integer n;

Procedura ta tworzy obiekt tekstowy o długości n ($n \geqslant 0$) znaków, przy czym początkową wartością tekstową obiektu jest n spacji, np.

T :- blanks (6) ; T — 1 — ␣␣␣␣␣␣ ,

gdzie T jest zmienną typu text .

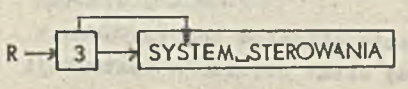
text procedure newtext(T); value T; text T;

Procedura ta tworzy obiekt tekstowy z wartością tekstową identyczną do wartości tekstowej parametru aktualnego oraz daje referencję do tego obiektu, np.

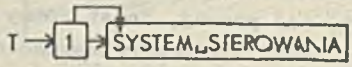
T :- newtext ('SYSTEM') ; T → 1 → SYSTEM
R :- newtext (T) ; R → 1 → SYSTEM

gdzie R,T - zmienne typu text.

Podczas tworzenia obiektu tekstowego wartość wskaźnika WP wynosi zawsze 1. Oznacza to, że jest wtedy dostępny pierwszy znak ciągu znakowego danego obiektu. Na przykład niech zmienna tekstowa R będzie referencją do obiektu tekstowego, którego WP = 3 (rys. 33).



Rys. 33



Rys. 34

Wtedy instrukcja T :- newtext(R), gdzie T jest jest zmienną tekstową, powoduje utworzenie nowego obiektu tekstowego z wartością identyczną do wartości obiektu tekstowego R, przy czym wskaźnik WP obiektu T jest równy 1 (rys. 34).
W powyższych przykładach wykorzystywana była instrukcja podstawienia referencyjnego (:-). Możliwe to jest dlatego, że zmienne typu text podobnie jak zmienne referencyjne stanowią referencje do obiektów (ich nazwy). Działanie instrukcji przypisania referencji tekstu ilustruje kolejny przykład:

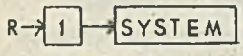
```
text R,T;
R :- newtext ('SYSTEM') ;
T :- R ;
```

R → 1 → SYSTEM
R → 1 → SYSTEM
T → 1

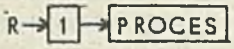
Widać stąd, że w wyniku wykonania instrukcji przypisania referencyjnego kopiowana jest wartość wskaźnika WP obiektu tekstowego znajdującego się z prawej strony tej instrukcji.

*) W implementacji na EMC IRIS-80 procedura generowania obiektu tekstowego nosi nazwę newtext, na EMC CDC6000-text. W innych implementacjach można spotkać - copy.

W Simuli 67 istnieje także instrukcja przypisania wartości tekstu (:=), która umożliwia kopiowanie łańcuchów znaków, zawartych w jednych obiektach tekstowych, do innych obiektów tekstowych. Instrukcje przypisania wartości tekstu nie powodują kopiowania wskaźnika WP. Jego wartość nie ulega zmianie. Na przykład, jeżeli zmienna R jest referencją do obiektu tekstowego z wartością 'SYSTEM' (rys. 35) to instrukcja R := 'PROCES' spowoduje powstanie sytuacji jak na rys.36.



Rys.35



Rys.36

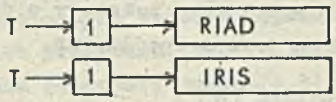
Jeśli ponadto zmienna tekstowa T jest referencją do obiektu tekstowego z wartością 'SYSTEM', to instrukcja R := T spowoduje powstanie sytuacji jak na rys. 35.

Widać stąd, że instrukcja przypisania wartości tekstu umożliwia kopiowanie tekstów, przy czym obowiązują tu pewne zasady, które podano poniżej.

Niech długość łańcucha znaków w obiekcie tekstowym z lewej strony instrukcji przypisania będzie równa L, a z prawej - P, Niech ponadto T będzie zmienną typu text. Efekt działania instrukcji przypisania wartości tekstu zależy od relacji między L i P.

Jeśli L=P, to cały łańcuch znaków z prawej strony instrukcji przypisania jest wpisany w miejsce tekstu będącego wartością obiektu tekstowego z lewej strony tej instrukcji, np. (rys. 37)

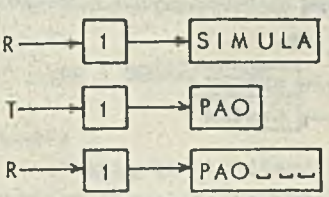
```
T :- newtext ('RIAD');
T := 'IRIS';
```



Rys.37

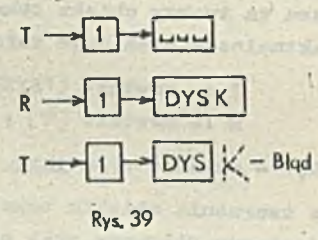
Jeśli L > P, to łańcuch znaków z prawej strony instrukcji przypisania jest wpisany w miejsce pierwszych P znaków tekstu z lewej strony. Pozostałych L - P znaków tekstu z lewej strony zostaje wypełnionych spacjami, np. (rys. 38).

```
R :- newtext ('SIMULA');
T :- newtext ('PAO');
R := T ;
```



Rys. 38

```
T :- blanks (3);
R :- newtext ('DYSK');
T := R;
```



Rys. 39

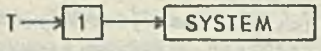
Jeśli L < P to sygnalizowany jest błąd (np. rys. 39).

Instrukcja przypisania wartości tekstowych jest również niepoprawna, jeżeli wartością jej lewej strony jest notext, zaś prawej, tekst różny od notext.

Z powyższych rozważań wynika równoważność dwóch sposobów generowania obiektów tekstowych z daną wartością, np. 'SYSTEM'. Niech T będzie zmienną typu text. Wtedy efekt działania instrukcji

```
T :- newtext ('SYSTEM');      oraz
T :- blanks (6);     T := 'SYSTEM' ;
```

jest ten sam i powoduje powstanie sytuacji jak na rys. 40.



Rys.40.

Procedury operacji na obiektach tekstowych

Do operacji na łańcuchach znaków, stanowiących wartości obiektów tekstowych, zdefiniowano liczne procedury, które kolejno omówimy. Dostęp do nich możliwy jest pod warunkiem wykorzystania zasady "kropkowanego" dostępu oraz zmiennej tekstowej jako referencji do danego obiektu tekstowego, tzn.

< proste wyrażenie tekstowe > . < nazwa procedury > .

Procedury lokalne dla obiektów tekstowych

- integer procedure length;

Procedura length oblicza długość pola łańcucha znaków np.

text T T → [1] → [SIMULA] T.length = 6.

Jeśli wartością T jest notext to T.length = 0.

- integer procedure pos;

Procedura pos wyznacza wartość wskaźnika pozycji (WP) znaku w łańcuchu znaków, np.

text T T → [3] → [SYSTEM OPERACYJNY] T.pos = 3.

Jeśli wartością T jest notext to T.pos = 1.
Możliwe jest także T.pos = T.length+1.

- procedure setpos(n); integer n;

Procedura setpos ustawia wartość wskaźnika WP na pozycji n w łańcuchu znaków. Jeśli $n \notin [1,length]$, wtedy n:=length + 1 *)

np. a) text T T → [2] → [SIMULA]

Instrukcja T.setpos (4) powoduje:

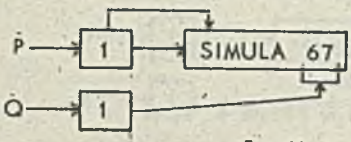
T → [4] → [SIMULA] .

b)
```
begin
  text R,T;
  R.setpos (5); $ R.pos=1, ponieważ wartością R jest notext $
  R :- blanks (10);
  R.setpos (8) ; $ R.pos=8 $
end;
```

- text procedure sub (i,n); integer i,n;

Wywołanie procedury T.sub (i,n) wyznacza podtekst o długości n tekstu w obiekcie tekstowym T, zaczynając od i-tej pozycji. Jeśli $i \leq 0$ lub $n < 0$ lub $i+n > T.length+1$, to sygnalizowany jest błąd. Zawsze WP podtekstu jest równe 1. Na przykład (rys. 41):

```
begin
  text P,Q;
  P :- newtext ('SIMULA 67') ;
  Q :- P.sub(7,2);
end
```



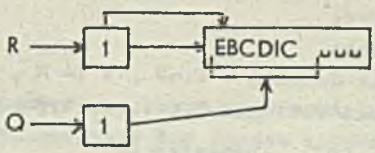
Rys.41

Należy dodać, że jeśli n=0 wtedy wynikiem procedury jest notext.

- text procedure strip;

Wyrażenie T.strip jest równoważne wyrażeniu T.sub(1,n), gdzie n jest najmniejszą liczbą całkowitą, taką, że pozostałe znaki tekstu obiektu tekstowego T (jeśli istnieją) są spacjami. A zatem wynikiem działania procedury strip jest podtekst tekstu T z usuniętymi spacjami końcowymi, np. (rys. 42):

```
begin
text R,Q;
R :- newtext ('EBCDIC␣␣␣');
Q :- R.strip;
end;
```



Rys.42

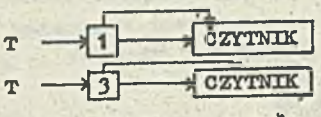
- boolean procedure more;

Procedura more przyjmuje wartość true, jeśli w obiekcie tekstowym T jego WP $\in [1,length]$.

*) Uwaga: WP $\in [1,length + 1]$

W przeciwnym razie T.more = false, np.

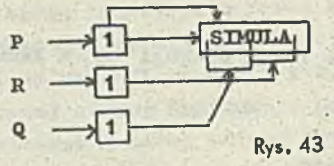
```
begin
text T;
T :- newtext ('CZYTNIK');
T.setpos(3);
$ T.more = true, ponieważ T.length=7 i WP=T.pos < 7 $ .
end;
```



- text procedure main;

Wyrażenie T.main daje referencję do obiektu tekstowego, który stanowi bądź zawiera wartość tekstową obiektu tekstowego T, np. (rys. 43):

```
begin
text P,R,Q;
P :- newtext ('SIMULA');
R :- P.sub (4,3);
Q :- P.sub(1,3);
$ P.main, Q.main, R.main są
referencjami do obiektu teks-
towego P z wartością tekstową 'SIMULA' $ .
end;
```



Rys. 43

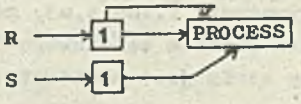
- character procedure getchar;

Jeśli T.more = true, wartością wyrażenia T.getchar jest wskazywany, przez wskaźnik pozycji WP obiektu T, znak, W przeciwnym razie sygnalizowany jest błąd. Wartość wskaźnika WP zwiększana jest o 1, np.

```
begin
  text A,B; character C,D;
  $ A.pos=1, B.more=false $
  A :- newtext ('PROCEDURY NA TEKSTACH');
  $ A.pos.=1 $
  C := A.getchar;
  D := A.getchar;
  $ A.pos=3, C = "P",D = "R" $
  B :- A;
  $ B.pos=3 $
  end;
```

Uwaga: Niech R,S,T będą zmiennymi typu text, C,D - zmiennymi typu character. Niech ponadto:

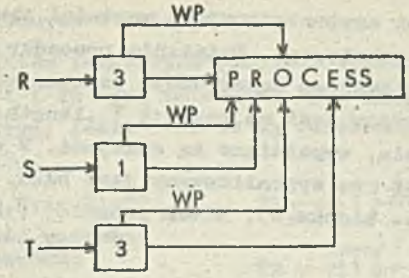
```
R :- newtext ('PROCESS');
S :- R;
C := R.getchar;
D := R.getchar;
$ Wtedy wynik działań : T:-R , T :- S , nie jest jednakowy, gdyż w pierwszym wypadku
  T.pos=3, zaś w drugim T.pos.=1. Sytuację, gdy T:-R ilustruje rys. 44 $
```



- procedure putchar (C); character C;

Jeśli T.more = true, instrukcja T.putchar (C) umieszcza znak C w miejscu łańcucha znaków wskazywanym przez wskaźnik pozycji WP w obiekcie tekstowym T oraz zwiększa wartość WP o 1. W przeciwnym razie sygnalizowany jest błąd, np.

```
begin
  text P;
  P :- newtext ('ODRA') ;
  P.putchar ("M");
```



Rys. 44

```
        P.putchar ("E");
    end;
```

W wyniku obliczenia powyższego bloku wartość tekstowa obiektu tekstowego P wynosi 'MERA'.

Procedury edycyjne i de-edycyjne

W Simuli 67 zdefiniowano procedury do przekształcania wartości arytmetycznych do postaci tekstowej i odwrotnie.

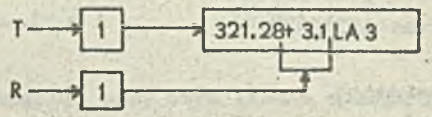
a) Procedury de-edycyjne (konwersji wartości tekstowej do wartości arytmetycznej).
Niech T będzie referencją do obiektu tekstowego. Procedury de-edycyjne:

- lokalizują najdłuższy element numeryczny (łańcuch znaków interpretowany jako liczba np. "259") zawarty w T i zawierający pierwszy znak wartości tekstowej obiektu T ; jeżeli nie można znaleźć takiego łańcucha znaków to sygnalizowany jest błąd,
- odnaleziony element numeryczny interpretują jako liczbę,
- ustawiają wskaźnik pozycji WP obiektu T na pozycji o 1 większej od pozycji ostatniego znaku elementu numerycznego.

- integer procedure getint;

  Procedura getint lokalizuje podtekst zawierający element numeryczny całkowity, który staje się wartością procedury, np. (rys. 45):

```
    begin
    text T,R;
    T :- newtext ('321.28+3.1LA3') ;
    R :- T.sub (7,4);
    end;
```



Rys. 45

  Wtedy T.getint=321, R.getint=3 i odpowiednio T.pos=4, R.pos=3.

- real procedure getreal;

  Procedura getreal lokalizuje podtekst zawierający element numeryczny całkowity lub rzeczywisty (z kropką dziesiętną lub wykładniczy), który staje się wartością procedury. Nawiązując do ostatniego przykładu (rys.45) mamy: T.getreal=321.28, R.getreal = 3.1 i odpowiednio T.pos = 7, R.pos =5.

- integer procedure getfrac;

  Procedura getfrac lokalizuje podtekst zawierający cyfry, spacje, kropki dziesiętne, a następnie ignoruje spacje i kropki dziesiętne. Otrzymana w ten sposób liczba całkowita staje się wartością procedury, np.

```
  begin
    text T;
    T :- newtext ('32␣␣15.4␣8␣+7A') ; § T.getfrac=321548,T.pos=12 §
  end;
```

b) Procedury edycyjne (konwersji wartości arytmetycznej do wartości tekstowej).
Niech T będzie referencją do obiektu tekstowego. Działanie procedur edycyjnych polega na wypełnieniu pola tekstowego obiektu T elementem numerycznym poczynając od prawej strony tego pola. Wskaźnik pozycji znaku WP ustawiany jest na pozycji T.length+1. Poprzedzające element numeryczny, nie wypełnione pozycje pola, wypełniane są spacjami. W elementach dodatnich pomija się znak "+". Jeżeli T == notext to sygnalizowany jest błąd.
Niech będzie dany obiekt tekstowy T :- blanks(8). Niech ponadto T:='␣␣␣␣␣111'.

- procedure putint (i) ; integer i;

  Procedura putint wypełnia pole tekstowe obiektu T liczbą całkowitą i, np.

  T.putint (10) T ='␣␣␣␣␣␣10'
  T.putint (121.7) T ='␣␣␣␣␣122' .

- procedure putfix (a,b); real a; integer b;

  Procedura putfix wypełnia pole tekstowe obiektu T liczbą rzeczywistą a z b cyframi po kropce dziesiętnej. Jeśli b < 0, to sygnalizowany jest błąd, np.

  T.putfix (125.38,2) T = '␣␣125.38'
  T.putfix (-3.78,1) T = '␣␣␣␣-3.8' .

- procedure putreal(a,b); real a; integer b;

  Procedura putreal wypełnia pole tekstowe obiektu T liczbą rzeczywistą w postaci wykładniczej z b cyframi znaczącymi. Jeśli b < 0, wtedy sygnalizowany jest błąd, np.

  T.putreal (-25.32,1) T = '␣␣-3E+01'
  T.putreal (0.01583,3) T = '1.58E-02' .

- procedure putfrac(a,b); integer a,b;

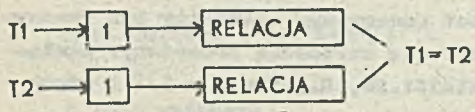
  Procedura putfrac wypełnia pole tekstowe obiektu T liczbą rzeczywistą z b miejscami po kropce dziesiętnej, otrzymaną z liczby całkowitej a . Każde 3 cyfry liczby, licząc od kropki dziesiętnej, oddzielone są spacją, np.

  T :- blanks(15);
  T .putfrac (12345678,4) T = '␣␣␣␣ 234.567 8'
  T .putfrac(12345678,0) T = '␣␣␣␣␣12 345 678' .

Relacje na tekstach

a) Operatory relacji dla wartości tekstowych

Wartości tekstowe T1 i T2 mogą być porównywane za pomocą operatorów: <,<=,=,>,>=,/= .
Dwie wartości tekstowe są równe tzn. T1=T2, jeśli są tym samym łańcuchem znaków (rys. 46) , w przeciwnym razie są one różne (rys. 47).



Rys. 46

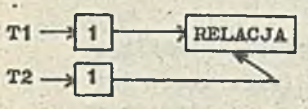
Rys. 47

Wartość tekstowa T1 jest mniejsza od wartości tekstowej T2, jeżeli obie są różne i zachodzi jeden z następujących warunków:

- T1 jest pusta, a T2 jest dowolnym tekstem, np. notext < 'relacja',
- T2 jest równe T1 z następującym po nim jednym lub większą liczbą znaków, np.
  'relacja' < 'relacjatext' ,
- jeśli pierwsze i-1 znaków w T1 i T2 są równe, a i-ty znak T1 jest mniejszy od i-tego znaku T2, np.
  'symulacja' < 'system' , gdyż rank("m") < rank ("s").

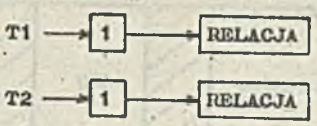
b) Operatory relacji dla zmiennych tekstowych

Zmienne referencyjne do obiektów tekstowych (zmienne tekstowe) mogą być porównywane za pomocą operatorów : == i =/=. Niech T1 i T2 będą zmiennymi typu text.
Relacja T1 == T2 daje wartość true, jeśli T1 i T2 są referencjami do tego samego obiektu tekstowego lub do notext, np. (rys. 48):



T1==T2 oraz T1=T2
(równość wartości tekstowych)

Rys.48



T1=/=T2, ale T1=T2
(równość wartości tekstowych)

Rys. 49

Relacja T1 =/= T2 daje wartość true, jeśli T1 i T2 nie wskazują na ten sam obiekt tekstowy, nawet gdy wartości tekstowe obiektów, na które wskazują są identyczne, (np. rys. 49):

Rozważmy prosty przykład:

```
begin
   text P,Q,R; boolean A,B,C;
   P:- newtext ('relacja');
   Q:- newtext ('relacja');
   R:-P;
   A:= P=/=Q;  B:=P=Q;   C:= not R=/=P and R=P;
end;
```

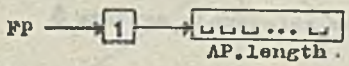
Wartości wszystkich zmiennych boolowskich A,B,C są true.

Przesyłanie parametrów typu text w procedurach i klasach

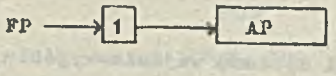
W punktach "Procedury" i "Klasy" (oz.I) poz. [6] podano zasady przesyłania parametrów w procedurach i klasach, przy czym wskazano na inny charakter przesyłania przez wartość parametrów tekstowych.
Przesyłanie parametru tekstowego przez wartość odbywa się w dwóch krokach:

- FP :- blanks (AP.length);
- FP := AP;

gdzie FP - parametr formalny, AP - parametr aktualny.
W pierwszym kroku utworzony zostaje obiekt zawierający spacje, przy czym liczba spacji jest równa liczbie znaków w łańcuchu znaków parametru aktualnego, tzn.



W drugim kroku obiektowi tekstowemu FP nadana zostaje wartość tekstowa parametru aktualnego, tzn.



Rozważmy przykład:

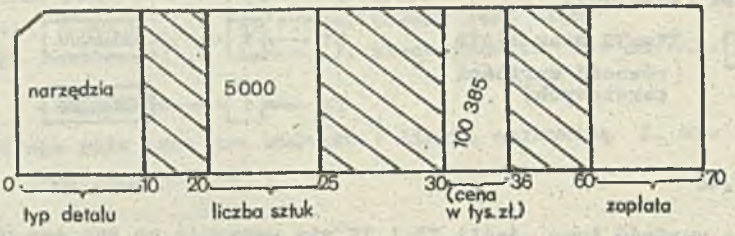
```
procedure A (t); value t; text t;
outtext (t);
```

Wywołanie procedury A ('Simula') spowoduje:

- t → 1 → ␣␣␣␣␣␣ t:-blanks (6); AP.length=6
- t → 1 → Simula t:='Simula';
- wypisanie wartości tekstowej obiektu t, tzn. Simula.

Widać stąd, że przesłanie parametru tekstowego przez wartość powoduje skopiowanie wartości parametru aktualnego. W wyniku tego unika się efektów ubocznych polegających na zmianie, wskutek wywołania procedury, wartości parametru aktualnego.
Należy tu podkreślić, że normalnym sposobem przesyłania parametrów tekstowych w klasach i procedurach jest przesyłanie przez referencję.

Rozważmy teraz bardziej złożony przykład, w którym wykorzystamy kilka przedstawionych wyżej procedur operacji na obiektach tekstowych. Dany jest dokument, będący elementem pewnego banku danych dotyczących gospodarki materiałami i narzędziami, w postaci:



Uwaga: W polach zakreskowanych mogą być przechowywane inne informacje

Należy wydobyć informacje niezbędne do określenia zapłaty za narzędzia oraz obliczyć i wpisać do dokumentu wielkość tej zapłaty. Realizuje to następujący program:

```
begin
  text kwit,typ,suma,cena,zapłata;
  integer płaca,ogółem;
  kwit :- blanks (80);
  kwit :- intext (80) ;   $ wczytanie dokumentu $
  typ :- kwit.sub (1,10) ;
  $ zmienna  typ  wskazuje na podtekst dokumentu dotyczący typu detalu $
  suma :- kwit.sub (20,5);
  $ zmienna suma wskazuje na podtekst dokumentu dotyczący liczby sztuk detali $
  cena :- kwit.sub (30,6);
  $ zmienna cena wskazuje na podtekst dokumentu dotyczący ceny detali w tysiącach zł $
  zapłata :- kwit.sub (60,10);
  $ zmienna zapłata wskazuje na podtekst dokumentu dotyczący zapłaty za detale. W to
    miejsce należy wpisać wielkość zapłaty $
  if typ = 'narzędzia' then
  begin
    płaca := suma.getint * cena.getfrac;
    $ obliczenie wielkości zapłaty $
      zapłata.putfrac (płaca,2);
    $ wpisanie zapłaty w pole określone referencją zapłata $
    end ;
  end; $ programu $
```

Klasa "Basicio"

Podstawowym pojęciem koncepcji organizacji systemu wejścia-wyjścia jest zbiór (file). Zbiór jest sekwencją wartości tekstowych, z których każda nazywana jest rekordem, np.

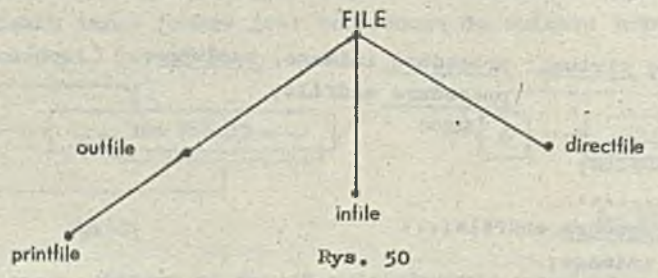
- pakiet kart perforowanych, gdzie rekordem jest tekst jednej karty
- wyjście z drukarki, gdzie rekordem jest drukowana linia, itp.

Koncepcja zbioru obejmuje także zbiory znajdujące się w pamięciach zewnętrznych tzn. dyskowych, bębnowych, taśmowych. W tym wypadku informacja, która ma być przeniesiona do pamięci zewnętrznej staje się wartością obiektu tekstowego, po uprzednim przekształceniu jej do postaci znakowej (character). Długość pola obiektu tekstowego zwykle odpowiada wielkości rekordu. W wypadku odczytu z pamięci zewnętrznej informacja numeryczna znajdująca się w obiekcie tekstowym musi być przekształcona z postaci znakowej do wartości numerycznej. Operacje tego typu, jak również operacje wczytywania i wypisywania rekordów wymagają zdefiniowania wielu procedur. Procedury te zdefiniowane są w klasach systemowych, które reprezentują zbiory zewnętrzne, dostępne dla użytkownika Simuli 67.

Koncepcję organizacji systemu wprowadzania i wyprowadzania informacji zdefiniowano w systemowej klasie BASICIO*). Zwróćmy uwagę na to, że prefiksowanie bloku programu klasą BASICIO umożliwia korzystanie ze wszystkich procedur w niej zadeklarowanych.
Ogólna deklaracja klasy BASICIO ma postać:

```
class BASICIO(LINELENGTH); integer LINELENGTH;
begin
  ref(infile)SYSIN;
  ref (printfile) SYSOUT;
  ref(infile)procedure sysin;
  sysin :- SYSIN;
  ref(printfile)procedure sysout;
  sysout:- SYSOUT;
  class FILE;
  FILE class infile..;
  FILE class outfile..;
  FILE class directfile..;
  outfile class printfile..;
  SYSIN:- new infile('SYSIN');
  SYSOUT:- new printfile ('SYSOUT');
  SYSIN.open(blanks(80));
  SYSOUT.open(blanks(LINELENGTH));
  inner;
  SYSIN.close;
  SYSOUT.close;
end; § BASICIO §
```

Klasa BASICIO jest niedostępna dla programisty. Zmienne referencyjne SYSIN i SYSOUT reprezentują odpowiednio standardową jednostkę wejścia zorientowaną na karty perforowane i standardową jednostkę wyjścia zorientowaną na drukarkę. Wartości tych zmiennych można uzyskać wykorzystując procedury sysin i sysout. Z kwalifikacji zmiennych SYSIN i SYSOUT wynika, że umożliwiają one dostęp odległy do wszystkich atrybutów klasy infile, printfile i outfile. Kolejno omówimy wszystkie klasy zdefiniowane w klasie BASICIO. Hierarchię tych klas ilustruje graf na rys. 50. Dokładne treści procedur zdefiniowanych w tych klasach podano w Dodatku 1.



Rys. 50

Klasa FILE

```
Definicja: class FILE(NAME); value NAME; text NAME;
           virtual: procedure open, close;
           begin  text image;  boolean OPEN;
             procedure open...;
             procedure close...;
             procedure setpos(i); ...;
             integer procedure pos..;
             boolean procedure more..;
             integer procedure length..;
           end;
```

*) Wszystkie nazwy pisane dużymi literami są niedostępne bezpośrednio dla programisty

Klasa FILE jest niedostępna dla programisty. Każdy obiekt tej klasy posiada atrybut NAME. Atrybut ten identyfikuje zewnętrzny zbiór, który przez określony mechanizm, zdefiniowany w implementacji, powiązany jest z danym obiektem klasy FILE.

Zmienna tekstowa image jest referencją do obiektu tekstowego odpowiadającego przetwarzanemu rekordowi. Wprowadzanie i wyprowadzanie rekordów z wykorzystaniem obiektu tekstowego image możliwe jest tylko wtedy, gdy zmienna boolowska OPEN przyjmuje wartość true (zbiór jest "otwarty"). Procedury open i close odpowiednio "otwierają i "zamykają" operacje na zbiorze. Są one atrybutami wirtualnymi, a więc mogą być definiowane ponownie na wewnętrznych poziomach prefiksowania (w podklasach klasy FILE).

Procedura open nadaje zmiennej OPEN wartość true oraz tworzy referencję 'image' do obiektu tekstowego. Długość tekstu będącego wartością zmiennej image jest określona parametrem aktualnym procedury open. Procedura close nadaje zmiennej OPEN wartość false, a obiektowi tekstowemu image - wartość tekstową równą notext. Procedura ta likwiduje połączenie ze zbiorem zewnętrznym.

Początkowo zbiór jest zawsze "zamknięty", tzn. OPEN=false. Powyższa uwaga nie dotyczy zbiorów, do których referencję dają zmienne SYSIN i SYSOUT. Zbiory te są "otwierane" systemowo (zob. pkt. "Algorytm klasy BASICIO") . Chcąc zatem przetworzyć np. zbiór kart należałoby napisać ciąg instrukcji

```
text T;
T :- new FILE('karty') ; $ utworzenie zbioru $
T.open(blanks(80)) ;  $ otwarcie zbioru $
< przetwarzanie informacji (inner) >
T.close;    $ zamknięcie zbioru $
```

Pozostałe procedury klasy FILE dotyczą operacji na rekordach, a właściwie na reprezentujących je obiektach tekstowych (image) oraz wykorzystują procedury o tych samych nazwach omówione w pkt. "Operacje na znakach i tekstach". Obiekt każdej klasy prefiksowanej klasą FILE reprezentuje zatem określony zbiór. W klasie BASICIO zdefiniowano cztery takie zbiory:

- infile - reprezentujący zbiór sekwencyjny, w którym dostępne są operacje wejścia,
- outfile - reprezentujący zbiór sekwencyjny, w którym dostępne są operacje wyjścia,
- directfile - reprezentujący zbiór, w którym dostępne są operacje wejścia i wyjścia (w szczególności przy współpracy z pamięciami zewnętrznymi),
- printfile - reprezentujący zbiór wyjściowy zorientowany na drukarkę.

Klasa infile

Definicja:

```
FILE class infile; virtual: procedure inimage, boolean
                                procedure endfile;
        begin
          boolean ENDFILE;
          ..................
          boolean procedure endfile;..;
          procedure inimage;
          procedure open..;
          procedure close..;
          character procedure inchar..;
          boolean procedure lastitem..;
          integer procedure inint..;
          real procedure inreal..;
          text procedure intext..;
        end;
```

- procedure open;

  Procedura open działa podobnie do procedury open w klasie FILE, a ponadto umieszcza wskaźnik pozycji znaku WP w obiekcie image na pozycji length + 1. Każde wywołanie procedury wejścia spowoduje wczytanie pierwszego rekordu ze zbioru wejściowego.

- procedure close;

Procedura close działa analogicznie do procedury close w klasie FILE, a ponadto nadaje zmiennej ENDFILE wartość true.

- procedure inimage;

Procedura inimage nadaje obiektowi image (obiektowi tekstowemu, do którego referencję daje zmienna image) wartość tekstową rekordu zbioru zewnętrznego. Jeżeli wielkość pola obiektu tekstowego image jest mniejsza od długości tekstu rekordu, to sygnalizowany jest błąd. W przeciwnym razie pozostała część pola obiektu image wypełniona zostaje spacjami. Wskaźnik pozycji znaku WP w obiekcie image ustawiony jest na 1. Zwróćmy uwagę na to, że ponowne wywołanie procedury inimage powoduje nadanie obiektowi image nowej wartości tekstowej wczytywanego rekordu.

- boolean procedure endfile;

Procedura endfile daje dostęp do zmiennej ENDFILE. Przyjmuje ona wartość true:
- jeśli napotkano koniec zbioru zewnętrznego (ENDFILE:=true)
- przed "otwarciem" zbioru infile,
- po zamknięciu zbioru infile.

- boolen procedure lastitem;

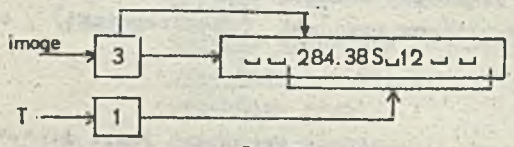
Procedura lastitem przyjmuje wartość false, jeśli zbiór zewnętrzny zawiera w chwili wywołania co najmniej 1 znak nie będący spacją. Jeśli w zbiorze zewnętrznym znajdują się same spacje to procedura przyjmuje wartość true. Należy tu zaznaczyć, że gdy image.more jest false to wywoływana jest procedura inimage.

- character procedure inchar;

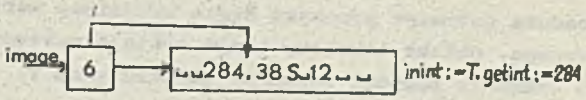
Procedura inchar daje dostęp do aktualnie dostępnego znaku tekstu image. Jeśli image.more = = false, to automatycznie wywoływana jest procedura inimage i wartością inchar staje się pierwszy znak nowej wartości tekstowej obiektu image. Np. C := inchar;

- integer procedure inint;

Procedura inint powoduje wprowadzenie liczby całkowitej ze zbioru zewnętrznego, np.U:=inint; Rozważmy bliżej działanie tej procedury (Dodatek 1). Procedura inint, analizując wartość tekstową obiektu image od aktualnej pozycji wskaźnika WP, pomija wszystkie spacje do pierwszego napotkanego znaku, a następnie przez zmienną tekstową T daje referencję do pozostałego podtekstu obiektu image (image jest referencją do obiektu tekstowego odpowiadającego przetwarzanemu rekordowi) (rys.51).



Rys. 51



Rys.52

W przypadku braku znaku różnego od spacji automatycznie wywoływana jest procedura inimage. Następnie de-edycyjna procedura T.getint lokalizuje podtekst obiektu T z najdłuższym elementem numerycznym całkowitym (w naszym wypadku - 284), który staje się wartością procedury inint (rys. 52).
Wskaźnik pozycji znaku WP obiektu image ustawiony zostaje na pierwszym znaku, który nie jest częścią de-edytowanego elementu numerycznego. Po wczytaniu w analogiczny sposób liczb 38 i 12 (rys. 52) , każda próba odczytania nowej liczby spowoduje wywołanie procedury inimage (ze względu na spacje w końcowej części pola tekstowego obiektu image), która wprowadzi wartość tekstową nowego rekordu w polo tekstowe obiektu image.

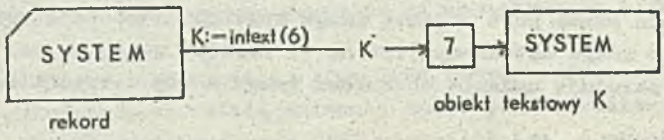
- real procedure inreal;

Procedura inreal powoduje wprowadzenie liczby rzeczywistej ze zbioru zewnętrznego, np. A:=inreal; Zasada działania tej procedury jest analogiczna do działania procedury inint,

z tą różnicą, że wykorzystuje ona procedurę de-edycyjną getreal.

- text procedure intext(m); integer m;

Procedura intext daje referencję do nowego obiektu tekstowego zawierającego m kolejnych znaków zbioru zewnętrznego, np. K:-intext(6), (rys. 53). Jeśli w zbiorze zewnętrznym nie ma m znaków to sygnalizowany jest błąd. Należy dodać, że danych m znaków może znajdować się w różnych kolejnych rekordach.



Rys. 53

Klasa outfile

```
Definicja: FILE class outfile; virtual: procedure outimage;
           begin
             procedure open..;
             procedure close..;
             procedure outimage;...;
             procedure outchar (C); character C;..;
             procedure outint (i,w); integer i,w;...;
             procedure outfix (r,n,w); real r; integer n,w;...;
             procedure outreal (r,n,w) ; real r; integer n,w;...;
             procedure outtext (T); text T;...;
             ..............
           end; $ outfile $
```

- procedure open;

Procedura open działa analogicznie do procedury open zdefiniowanej w klasie FILE.

- procedure close;

Procedura close działa analogicznie do procedury close w klasie FILE, a ponadto, gdy wskaźnik pozycji znaku WP obiektu image jest różny od 1, wywołuje procedurę outimage.

- procedure outimage;

Procedura outimage przesyła kopię aktualnej wartości tekstowej obiektu image do zbioru zewnętrznego. Obiekt tekstowy image zostaje następnie wypełniony spacjami (image:=notext) , a jego wskaźnik pozycji znaku WP ustawiony na 1.

- procedure outchar (C);

Procedura outchar powoduje wyprowadzenie wartości zmiennej typu character, np. jeśli A:="+" to instrukcja outchar (A) spowoduje wyprowadzenie do zbioru zewnętrznego znaku dodawania. Jeżeli image.more jest false, to wywoływana jest procedura outimage.

- procedure outtext (T); value T; text T;

Procedura outtext (T) powoduje wyprowadzenie tekstu T do obiektu image poczynając od aktualnie dostępnego znaku. Jeśli część image, która została do wypełnienia, jest za krótka, wtedy następuje wywołanie procedury outimage, zaś tekst T zostaje wpisany do image poczynając od pierwszej pozycji.

- procedure outint (i,w);

Procedura outint powoduje wyprowadzenie liczby całkowitej i, przy czym w jest wielkością pola wyjściowego, np.

```
outint (12,2)          -12
outint (-12,8)         -␣␣␣␣␣-12
outint (-3282.2,8)     -␣␣␣␣-3282
```

- procedure outfix (r,n,w) ;

Procedura outfix powoduje wyprowadzenie liczby rzeczywistej r , przy czym n jest wielkością pola części dziesiętnej, w - wielkością pola wyjściowego, np.

outfix (12,0,6) -␣␣␣12

outfix (61.24,3,8) -␣␣61.240 .

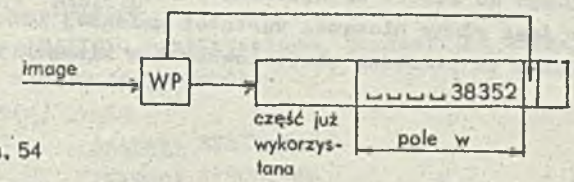
- procedure outreal(r,n,w);

Procedura outreal powoduje wyprowadzenie liczby rzeczywistej r , w postaci wykładniczej, z częścią ułamkową o polu n-1, gdzie w jest wielkością pola wyjściowego. Wielkość części wykładniczej jest zdefiniowana dla określonej implementacji i zależy od długości słowa w EMC, np.

outreal (1213.257,5,10) $-1.2133_{10}+03$

outreal (-0.00576,5,14) $-␣␣␣-5.7600_{10}-03$.

Wyprowadzanie wartości zmiennych za pomocą procedur outchar, outint, outfix i outreal polega na wypełnieniu pola tekstowego obiektu image odpowiednimi elementami numerycznymi, otrzymanymi w wyniku przekształcenia wartości numerycznych tych zmiennych do postaci tekstowej przy pomocy procedur odpowiednio putchar, putint, putfix i putreal (rys. 54).



Rys. 54

Jeżeli pozostałe pole tekstowe obiektu image jest zbyt małe, aby wypełnić je łańcuchem numerycznym, to wywoływana jest automatycznie procedura outimage, która umożliwia wyprowadzanie do zbioru zewnętrznego aktualnej wartości tekstowej obiektu image. Obiekt image zostaje wypełniony spacjami. Teraz dany łańcuch numeryczny można już wprowadzić do obiektu image począwszy od pierwszej pozycji jego pola (wtedy WP=1) itd. Bliżej ilustrują to przykłady w końcowej części tego punktu.

Klasa directfile

```
Definicja: FILE class directfile;
                virtual: boolean procedure endfile, procedure locate,inimage, outimage;
                begin
                  integer LOC;
                  integer procedure location;...;
                  procedure locate;...;
                  procedure open;...;
                  procedure close;...;
                  boolean procedure endfile;...;
                  procedure inimage;...;
                  procedure outimage;...;
                  character procedure inchar;...;
                  boolean procedure lastitem;...;
                  integer procedure inint;...;
                  real procedure inreal;...;
                  text procedure intext;...;
                  procedure outchar;...;
                  procedure outint;...;
                  procedure outfix;...;
                  procedure outreal;...;
                  procedure outtext;...;
                end; $ directfile $
```

- procedure location;
  Procedura location daje dostęp do aktualnej wartości zmiennej LOC, której wartość jest kolejnym numerem rekordu zewnętrznego.
- procedure locate;
  Procedura locate nadaje nową wartość zmiennej LOC,
- procedure open;
  Procedura open działa analogicznie do procedury open zdefiniowanej w klasie FILE, a ponadto daje dostęp do pierwszego rekordu zbioru.
- procedure endfile;
  Procedura endfile przyjmuje wartość true, jeśli aktualna wartość zmiennej LOC nie identyfikuje rekordu w zbiorze zewnętrznym.
- procedure inimage;
  Procedura inimage działa analogicznie do procedury inimage w klasie infile, a ponadto przesyła do obiektu tekstowego image kopię rekordu zewnętrznego, zidentyfikowanego przez zmienną LOC (jeśli taki istnieje). Wartość zmiennej LOC zwiększana jest wtedy o 1 wg procedury locate.
- procedure outimage;
  Procedura outimage jest analogiczna do procedury outimage w klasie outfile, a ponadto przesyła kopię wartości tekstowej obiektu image do zbioru zewnętrznego, poprzez dodanie do tego zbioru rekordu, którego kolejny numer jest równy bieżącej wartości zmiennej LOC. Pozostałe procedury klasy directfile są analogiczne do procedur zdefiniowanych w klasach infile i outfile.

Klasa printfile

```
Definicja:  outfile class printfile;
            begin
              integer LINES PER PAGE, SPACING, LINE;
              procedure open;...;
              procedure close;...;
              procedure outimage;...;
              procedure lines per page;...;
              integer procedure line;...;
              procedure spacing;...;
              procedure eject;...;
              procedure page;...;
              ..........
            end; $ printfile $
```

Klasa printfile orientuje klasę outfile na wyprowadzanie informacji za pomocą drukarki. Stąd obiekt tekstowy image reprezentuje tu linię na drukowanej stronie.

- procedure open;
  Procedura open działa analogicznie do procedury open w klasie FILE, a ponadto ustawia aktualnie drukowaną linię jako pierwszą linię strony.
- procedury close;
  Procedura close działa analogicznie do procedury close w klasie FILE a ponadto, jeżeli pos aktualnego obiektu tekstowego image (image.pos) jest różny od 1, wyprowadza bieżącą wartość tekstową tego obiektu (wywołując procedurę outimage) do zbioru zewnętrznego i nadaje odpowiednie wartości zmiennym LINE, SPACING i LINES PER PAGE.
- procedure outimage;
  Procedura outimage działa analogicznie do procedury outimage w klasie outfile, a dodatkowo uaktualnia wartość zmiennej LINE.
- integer procedure line;
  Procedura line jest procedurą funkcyjną, a jej wartością jest aktualna wartość zmiennej LINE wskazującej numer linii, w której nastąpió ma wyprowadzenie informacji.
- procedure lines per page (n);
  Procedura lines per page, nadając wartość n zmiennej LINES PER PAGE określa maksymalną liczbę

linii n, które można wydrukować na jednej stronie, włącznie z liniami zawierającymi same spacje. Jeżeli w programie procedura ta nie jest wywoływana, to wartość zmiennej LINES PER PAGE przyjmowana jest zgodnie z określoną implementacją.

- procedure spacing (n);

Procedura spacing nadaje zmiennej SPACING wartość n oraz powoduje wydruk n linii pustych (zawierających same spacje). W chwili początkowej SPACING=1. Jeśli n > LINES PER PAGE lub n < 0, to sygnalizowany jest błąd.

- procedure eject(n);

Procedura eject powoduje przejście do linii o numerze n. Można tu wyróżnić następujące sytuacje:

n ⩽ 0 - sygnalizowany jest błąd,
n > LINES PER PAGE - jest równoważne eject(1),
n ⩽ LINE - przejście do linii o numerze n na następnej stronie,
n > LINE - przejście do linii o numerze n na bieżącej stronie.

- procedure page;

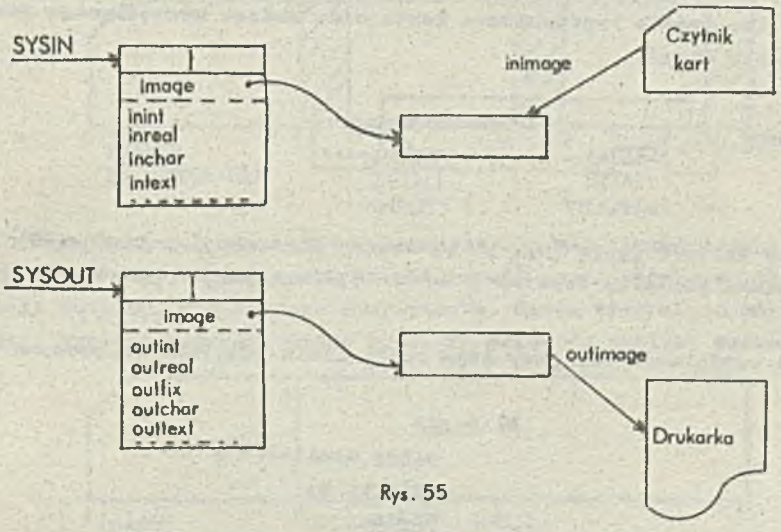
Procedura page powoduje przejście do nowej strony.

Algorytm klasy BASICIO

Alforytm klasy BASICIO przedstawiono, prezentując definicję tej klasy. Każdy program użytkownika działa w następujący sposób:

```
BASICIO(n) begin
             inspect SYSIN do
             inspect SYSOUT do
             < program użytkownika >
           end;
```

Z chwilą rozpoczęcia wykonywania programu użytkownika wykonuje się algorytm klasy profiksującej blok, tzn. BASICIO. System automatycznie (zgodnie z algorytmem BASICIO) generuje dwa zbiory: infile- do wprowadzania kart perforowanych, wskazywany przez zmienną SYSIN (SYSIN:-new infile ('SYSIN');) oraz printfile - do wyprowadzania informacji na drukarkę, wskazywany przez zmienną SYSOUT (SYSOUT:-new printfile ('SYSOUT');) (rys. 55) . Następnie zbiory te zostają "otwarte" wywołaniem procedury open, tzn. SYSIN.open(blanks(80)), SYSOUT.open(LINELENGTH). W wyniku tego w każdym z obiektów klasy infile i printfile utworzony zostaje obiekt tekstowy, do którego re-



Rys. 55

ferencję daje zmienna image. Początkową wartością tekstową tych obiektów jest notext. Od tej chwili rozpoczyna się wykonywanie programu użytkownika. Zmiana wartości tekstowych obiektów image następuje z chwilą wprowadzania lub wyprowadzania informacji. Najogólniej mówiąc, obiekty tekstowe image stanowią "bufory", w których tworzony jest rekord wejściowy bądź wyjściowy. Zasady tworzenia tych rekordów zilustrowano przykładami w końcu punktu. Po zakończeniu programu użytkownika zbiory infile i printfile zostają "zamknięte" przez wywołanie procedury close.

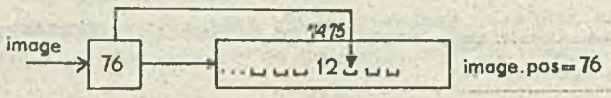
Należy podkreślić, że opisana powyżej procedura, poza programem użytkownika, wykonuje się systemowo. System zatem, w wyniku prefiksowania bloku klasą BASICIO oraz zastosowania mechanizmu połączenia (inspect) zapewnia użytkownikowi bezpośredni dostęp do wszystkich procedur zdefiniowanych w klasach infile, outfile, directfile oraz printfile.

Przykłady

- Dany jest program:

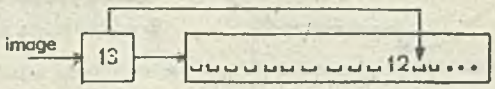
```
begin
  integer x,y;
  y:=x:=12;
  outimage;
  outint (x,75);
  outint(y,12);
end;
```

Przeanalizujmy jego działanie.
Wywołanie procedury outimage spowodowało wypełnienie obiektu tekstowego image spacjami, tzn. image:=notext. Instrukcja outint (x,75) powoduje powstanie sytuacji jak na rys. 56.



Rys. 56

Wartość tekstowa obiektu image nie może być jeszcze wyprowadzona do zbioru zewnętrznego, ponieważ może to odbyć się tylko za pośrednictwem procedury outimage. Następna instrukcja outint(y,12) powoduje sprawdzenie warunku WAR:= pos+w-1 > length (length=80). Ponieważ w przykładzie naszym WAR:=87 > 80 := true, więc automatycznie wywoływana jest procedura outimage (patrz Dodatek 1 procedure FIELD). W wyniku tego wyprowadzona zostaje kopia wartości tekstowej obiektu image do zbioru zewnętrznego (rys. 56), a pole tekstowe tego obiektu wypełniane jest spacjami. Następnie kontynuując wykonywanie instrukcji outint(y,12), po zadziałaniu procedury edycyjnej putint, obiektowi image zostaje nadana wartość tekstowa '␣␣␣␣␣␣␣␣␣␣12' (rys. 57). Może być ona wyprowadzona na zewnątrz dopiero po zakończeniu programu. Ostatnią instrukcją klasy BASICIO jest wywołanie procedury close, która w wypadku klasy printfile wywołuje procedurę outimage. W ten sposób wyprowadzona zostaje do zbioru zewnętrznego pozostała część wartości tekstowej obiektu image.



Rys. 57

Jeżeli zmienna WAR przyjęłaby wartość false (np. po pierwszej instrukcji outint(x,10)) , wtedy obiekt tekstowy image wypełniony zostałby tekstem '␣␣␣␣␣␣␣␣12' zaczynając od aktualnej pozycji wskaźnika WP itd.

- Dany jest program na kartach perforowanych, przy czym każda linia programu rozpoczyna się od pierwszej kolumny (a).

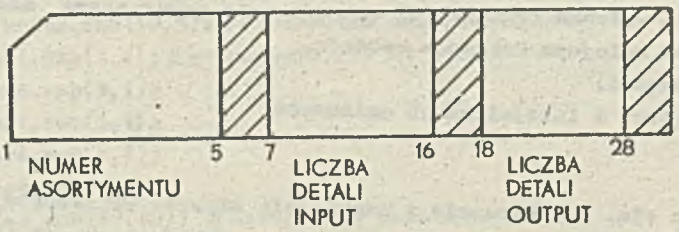
a)
```
begin
class monitor(x);
integer x;
begin
procedure A ;
begin null;
end;
end;
new monitor(3);
end;
```

b)
```
begin
  class monitor(x);
  integer x;
  begin
    procedure A;
    begin null;
    end;
  end;
  new monitor(3);
end;
```

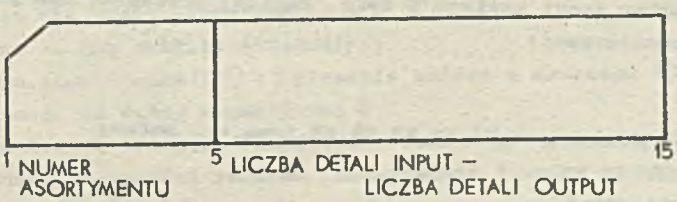
Należy napisać program, który wczytuje program (a), a wyprowadza na drukarkę ten sam program lecz w postaci (b). Mamy więc:

```
begin
  integer A,B;
  text T,U,W,V;
  A:=1;
  T :- sysin.image;
  U :- sysout.image;
  $ utworzono obiekty tekstowe image $
  V :- T.sub(1,7);
  $ V daje referencję do podtekstu obiektu T identyfikującego słowo begin $
  W :- T.sub(1,4);
  $ W daje referencję do podtekstu obiektu T identyfikującego słowo end $
  inimage; $ wczytanie rekordu $
  while not endfile do
  begin
    if V = 'begin␣␣' then B:=A + 3
    else if W = 'end;' then A:=A-3;
    $ jeśli napotkano begin lub end to wartości wskaźników kolumny A i B ulegają odpowiednio zwiększeniu lub zmniejszeniu o 3 $
   U.sub(A,81-A):=T.sub(1,81-A);
   $ część pola wyjściowego obiektu image wypełniona zmodyfikowaną wartością tekstową
     podtekstu wejściowego obiektu image $
   outimage; $ wyprowadzenie wartości tekstowej wyjściowego obiektu image,
               image:=notext $
   if B/= 0 then begin A:=B; B:=0; end;
   $ jeśli napotkano begin to B /= 0 i bieżący wskaźnik kolumny wynosi A, tzn. zostaje
     przesunięty w prawo $
   inimage; $ wczytanie w pole wejściowego obiektu image nowego rekordu $
  end;
end; $ programu $
```

- Dany jest zbiór transakcji na kartach perforowanych, np. w postaci



Stanowi on pewien bank danych, w których rejestruje się liczbę detali, które wpłynęły do magazynu z procesu produkcyjnego (liczba detali input) oraz liczbę detali, które opuściły magazyn (liczba detali output), dla każdego asortymentu. Każda transakcja znajduje się na oddzielnej karcie. Należy napisać program, który przesyła zbiór do pamięci taśmowej w postaci rekordów

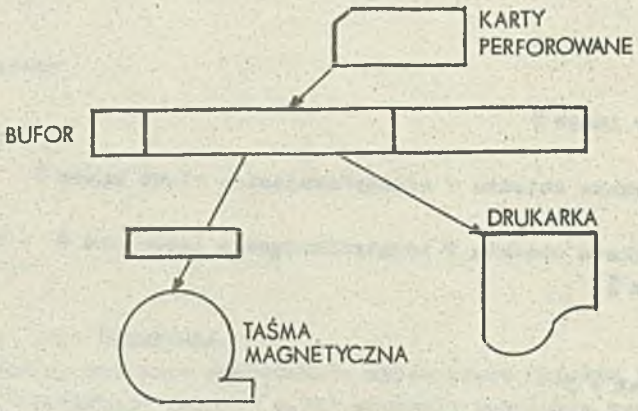


Numer asortymentu jest liczbą 4-cyfrową $a_1a_2a_3a_4$, zaś piąta cyfra a_5 jest cyfrą kontrolną. Jeżeli piąta cyfra a_5 spełnia zależność

$$a_5 = \text{ostatnia cyfra liczby } (7*(a_1+a_4)+3*a_3+a_2)$$

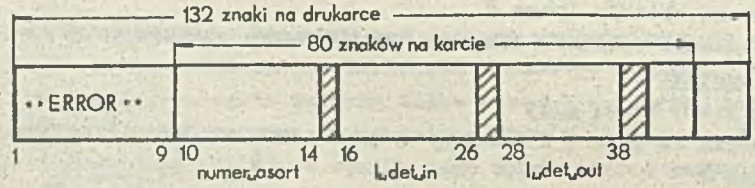
oraz pola: liczba detali input i liczba detali output wypełnione są tylko cyframi bądź

cyframi poprzedzonymi spacjami, wtedy rekord taki może być przesłany do pamięci taśmowej.W przeciwnym razie na drukarkę wyprowadzana jest informacja o błędzie. Program nasz wykorzystuje trzy zbiory: infile dla kart, outfile dla pamięci taśmowej i printfile dla drukarki. Wygodne będzie tu użycie wspólnego bufora dla zbiorów infile i printfile (rys.58).



Rys. 58

W buforze tym znajdować się będą rekordy wpływające z zewnątrz (z procesu) oraz rekordy wyprowadzane na drukarkę, informujące o błędzie. Sposób organizacji bufora ilustruje rys. 59.



Rys. 59

W celu łatwej analizy tekstu znajdującego się na kartach celowe jest podzielić pole karty na 3 pola o nazwach: numer_asort, l_det_in (liczba detali input) i l_det_out (liczba detali output). Analogicznie, dla sprawdzenia warunku poprawności rekordu pole numer_asort można podzielić na pola K1,..,K5, odpowiadające kolejnym kolumnom rekordu.
Program składa się więc z trzech części:
- konstrukcja bufora, otwarcie zbiorów i inicjalizacja zmiennych,
- przetwarzanie rekordów,
- zamknięcie zbiorów.

Bufor skonstruowany jest zgodnie z rys. 59. Otwarcie i zamknięcie zbiorów wykorzystuje procedury open i close. Przetwarzanie zaś rekordów wymaga zdefiniowania dwóch funkcyjnych (boolowskich) procedur pomocniczych. Pierwsza z nich o nazwie cyfra z parametrem typu text sprawdza czy dany tekst wypełniony jest cyframi bądź cyframi poprzedzonymi spacjami. Druga - o nazwie rekord_poprawny - sprawdza, czy wszystkie pola tzn. numer_asort, l_det_in i l_det_out, spełniają zadane warunki oraz czy pole numer_asort zawiera 5 cyfr. Poniżej przedstawiono pełny tekst programu wraz ze szczegółowymi komentarzami:

```
begin
  $ deklaracja zmiennych $
    text numer_asort,l_det_in,l_det_out,nr,K1,K2,K3,K4,K5,tsum,ts, bufor;
    integer suma, stan_magazynu;
  $ deklaracja referencji do zbiorów $
    ref(printfile)drukarka;
    ref(outfile)tasma_magnetyczna;
    ref(infile)proces;
  $ deklaracja procedury  cyfra $
```

```
boolean procedure cyfra (tekst); text tekst;
begin
  character c;
  tekst.setpos(1); $ iniojalizacja wskaźnika WP $
  c := ' ';
  while c = ' ' and tekst.more do
  c := tekst.getchar;
$ warunkiem wyjścia z powyższej pętli jest napotkanie pierwszego znaku różnego od spacji
  bądź przekroczenie przez WP długości tekstu $
  while digit(c) and tekst.more do
  c := tekst.getchar;
$ sprawdzenie czy kolejne znaki tekstu są cyframi. Pętla ta wykonuje się do chwili napotka-
  nia pierwszego znaku nie będącego cyfrą lub do chwili, gdy WP > tekst.length $
  cyfra := digit(c); $ Jeśli c jest cyfrą to wartością procedury jest true $
end; $ cyfra $
boolean procedure rekord_poprawny;
begin
  boolean pięć_cyfr;
  pięć_cyfr := digit(K1.getchar); $ K1 jest referencją do pierwszej kolumny pola numer_asort.
  Jeśli zawartość tej kolumny jest cyfrą to pole numer_asort zawiera 5 cyfr $
  rekord_poprawny := if pięć_cyfr then cyfra(numer_asort) and
                     cyfra (l_det_in) and cyfra(l_det_out)
                     else false;
end; $ rekord poprawny $
$ utworzenie bufora $
bufor :- blanks(132);
bufor.sub(1,9):='** ERROR **';$ wypełnienie bufora tekstem zgodnie z rys. 59 $
$ utworzenie zbioru dla kart $
proces :- new infile('KARTY') ; $ poprzez nazwę KARTY następuje przyporządkowanie zbioru do
urządzenia zewnętrznego $
proces.open(bufor.sub(10,80)) ; $ otwarcie zbioru o długości 80, ponieważ tyle kolumn zawiera
karta $
numer_asort :- bufor.sub(10,5); $ utworzenie pola numer_asort $
K1 :- numer_asort.sub(1,1) ; $ utworzenie pól K1 ÷ K5 $
K2 :- numer_asort.sub(2,1);
K3 :- numer_asort.sub(3,1) ;
K4 :- numer_asort.sub(4,1);
K5 :- numer_asort.sub(5,1) ;
l_det_in :- bufor.sub(16,10); $ utworzenie pola l_det_in $
l_det_out :- bufor.sub(28,10); $ utworzenie pola l_det_out $
$ utworzenie zbioru dla drukarki $
drukarka :- new printfile ('DRUKARKA');
drukarka.open(bufor); $ otwarcie zbioru o długości pełnego bufora $
$ utworzenie zbioru dla taśmy magnetycznej $
taśma_magnetyczna :- new outfile ('TASMA') ;
taśma_magnetyczna.open(blanks(15)) ; $ otwarcie zbioru o długości 15, gdyż taka jest długość
rekordu przesyłanego na taśmę magnetyczną $
$ utworzenie pomocniczych obiektów tekstowych: tsum - do wprowadzenia wyniku dodawania przy
obliczeniu warunku poprawności rekordu (maksymalnie 3 cyfry wyniku) oraz ts wskazującego na
ostatnią cyfrę tego wyniku $
tsum :- blanks(3);
ts :- tsum.sub(3,1) ;
$ początek programu przetwarzania rekordów $
inspect taśma_magnetyczna do
```

```
begin
  proces.inimage; $ wprowadzenie rekordu do bufora na pola 10-80 $
  while not proces.endfile do
  begin
  $ sprawdzenie poprawności karty $
    if reford_poprawny then
    begin
    $ sprawdzenie warunku poprawności pola numer_asort $
      suma:=7 *  (K1.getint+K4.gecint) +3*K3.getint+K2.getint;
    $ wprowadzenie wartości sumy do obiektu  tsum $
      tsum.putint(suma);
    $ sprawdzenie zgodności ostatniej cyfry , na którą wskazuje  ts $
      if ts /= K5  then drukarka.outimage else
      begin
      $wyprowadzenie na taśmę magnetyczną stanu magazynu$
        stan_magazynu := l_det_in.getint - l_det_out.getint;
        outtext(numer_asort);
        outint(stan_magazynu,10);
        outimage;
      $ procedury outtext i outint wywoływane są w bloku kwalifikowanym klasą outfile
        (inspect taśma_magnetyczna do) stąd wyprowadzenie informacji następuje na taśmę
        magnetyczną $
      end;
    end else drukarka.outimage; $ rekord błędny $
    proces.inimage;
  end; $ while $
end; $ inspect $
proces.close; $ zamknięcie zbiorów $
taśma_magnetyczna. close;
drukarka.close;
end; $ programu $
```

Generatory liczb pseudolosowych

W Simuli 67 zdefiniowano wiele generatorów liczb pseudolosowych pomocnych przy symulacji procesów stochastycznych oraz rozwiązywaniu problemów metodami Monte Carlo. Procedury generowania rozkładów prawdopodobieństwa oparte są na generatorze rozkładu równomiernego na przedziale $[0,1]$.

Procedura generowania liczb pseudolosowych z rozkładu równomiernego na przedziale $[0,1]$ jest rekurencyjną procedurą funkcyjną

$$U_n = F(U_{n-1}) \ ,$$

gdzie U_i - i-ta wygenerowana liczba pseudolosowa. Na przykład w implementacji na komputerze IRIS 80 procedura ta ma postać:

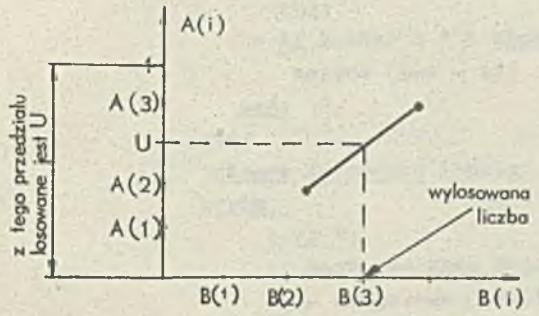
$$U_{n+1} = \lambda \mod 2^{32} \ ,$$

gdzie $\lambda = 162571342215$ (oktalnie). Okres takiego generatora wynosi 2^{30}.

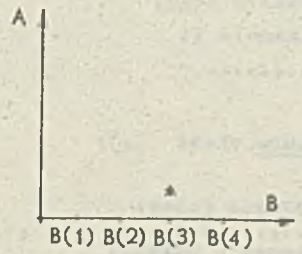
Podamy teraz ogólny opis wszystkich procedur generatorów rozkładów prawdopodobieństwa, występujących w Simuli 67. Dokładne treści tych procedur znajdują się w Dodatku 2.

- boolean procedure draw(a,U); name U; real a; integer U;
  Procedura draw daje wartość true z prawdopodobieństwem a i wartość false z prawdopodobieństwem 1-a. Jeśli a > 1, to wartością procedury jest true, natomiast gdy a < 0 - false.
  Np. if draw(0.5,U) then a:=3;
  Uwaga: Parametr procedury U jest przesyłany przez nazwę, stąd wywołanie procedury np. draw (0.5,3) jest błędne. Należy wcześniej nadać wartość zmiennej U, a następnie wywołać procedurę, tzn. U:=3; draw(0.5,U); Powyższa uwaga dotyczy także wszystkich pozostałych generatorów.

- integer procedure randint(A,B,U); name U; integer A,B,U;
  Procedura randint losuje liczbę całkowitą z przedziału [A,B] zgodnie z rozkładem równomiernym, np. a:=b+randint(1,5,U); Jeżeli A> B, wtedy sygnalizowany jest błąd.
- real procedure uniform(A,B,U); name U; real A,B; integer U;
  Procedura uniform losuje liczbę rzeczywistą z przedziału [A,B] zgodnie z rozkładem równomiernym, np. if b < uniform(3.1,4.2,U) then c:=randint (1,5,U); Jeśli A≥B, wtedy sygnalizowany jest błąd.
- real procedure normal (A,B,U); name U; real A,B; integer U;
  Procedura normal losuje liczbę zgodnie z rozkładem normalnym o średniej A i odchyleniu standardowym B, np. if draw(0.1,U) then bool := normal(10.,2.,U) > 5;
- real procedure negexp(A,U); name U; real A; integer U;
  Procedura negexp losuje liczbę zgodnie z ujemnym rozkładem wykładniczym ze średnią 1/A($f(x) = Ae^{-Ax}$), np. hold(negexp(0.5,U));
- integer procedure poisson(A,U); name U; real A; integer U;
  Procedura poisson losuje liczbę całkowitą k zgodnie z rozkładem Poissona z parametrem A, tzn. $P(X=k) = \frac{A^k}{k!} e^{-A}$. Np. kolejka:=kolejka + poisson (3.,U);
- real procedure erlang(A,B,U); name U; real A,B; integer U;
  Procedura erlang losuje liczbę zgodnie z rozkładem Erlanga ze średnią 1/A i odchyleniem standardowym $1/(A\sqrt{B})$, tzn. np. activate kolejka delay erlang(0.1,0.3,U);
- integer procedure discrete(A,U); name U; array A; integer U;
  Procedura discrete losuje liczbę całkowitą zgodnie z dystrybuantą rozkładu prawdopodobieństwa, zawartą w jednowymiarowej tablicy A, tzn. najmniejszą liczbę i, dla której U < A(i).
- real procedure linear(A,B,U); name U; array A,B; integer U;
  Procedura linear losuje liczbę rzeczywistą zgodnie z dystrybuantą F wykorzystując liniową interpolację funkcji F danej dwiema tablicami A i B, w których elementy mogą przyjmować dowolne wartości (A(i)= F(B(i)),) (rys. 60). Tablice A i B muszą być jednowymiarowe oraz posiadać tę samą wielkość. Ponadto muszą być spełnione warunki:
  - pierwszy i ostatni element tablicy A musi być równy odpowiednio 0 i 1 (przedział dla dystrybuanty),
  - $\forall(i>j)(A(i) \geq A(j) \wedge B(i) \geq B(j))$ (dystrybuanta jest funkcją rosnącą).



Rys. 60



Rys. 61

- integer procedure histd(A,U); name U; array A; integer U;
  Procedura histd losuje liczbę całkowitą z przedziału $[n_1,n_2]$, gdzie n_1 i n_2 są najmniejszą i największą wartością wskaźnika jednowymiarowej tablicy A. Tablica A jest histogramem obserwacji zmiennej całkowitej.
  Procedura ta jest bardziej czasochłonna od procedury discrete, gdzie dana jest funkcja dystrybuanty. Jest ona jednak pożyteczna wtedy, gdy operuje na histogramach uaktualnianych podczas przebiegu programu.
- procedure histo(A,B,C,D); array A,B; real C,D;
  Procedura histo tworzy histogram. Jest on zdefiniowany przez tablicę A i B obserwacji zmiennej C z wagą D. Element tablicy A, $A(n_1+i)$ jest zwiększany o D (najczęściej D=1), gdzie i jest najmniejszą liczbą całkowitą taką, że $C \leq B(n_1+i)$, a n_1 - najmniejszą wartością wskaźnika tablicy A i B. Ostatni element tablicy A odpowiada tym obserwacjom, które są większe od wartości wszystkich elementów tablicy B. Wynika stąd, że liczba elementów tablicy A musi

być o 1 większa od liczby elementów tablicy B. Na przykład, jeśli w trakcie tworzenia histogramu (rys.61) otrzymamy C=2.5, to wykonana zostanie instrukcja A(3):=A(3)+1 (graficznie można to zrealizować przez dodanie gwiazdki), gdyż liczba i=3 jest najmniejszą liczbą całkowitą taką, że $C \leq B(i)$.

Podane wyżej systemowe procedury generowania liczb pseudolosowych dotyczą najbardziej znanych rozkładów prawdopodobieństwa. Należy jednak pamiętać, że często w rzeczywistych procesach mamy do czynienia z innymi rozkładami. Należy wtedy we własnym zakresie zbudować generator liczb pseudolosowych o zadanym rozkładzie, wykorzystując na przykład metodę odwracania dystrybuanty. Jeżeli dysponujemy dystrybuantą empiryczną bądź histogramem, wygodnie jest zastosować jedną z procedur linear, histd lub discrete.

W następnej części (ostatniej) przedstawiony będzie mechanizm quasi-jednoczesności oraz klasa SIMULATION kierunkująca język Simula 67 na zagadnienia symulacyjne.

Dodatek 1. Klasa BASICIO (opis)

```
class BASICIO (LINELENGTH); integer LINELENGTH;
begin ref (infile) SYSIN;
      ref (printfile) SYSOUT;
      ref (infile) procedure sysin;
       sysin: - SYSIN;
      ref (printfile) procedure sysout;
       sysout: - SYSOUT;
      class FILE (NAME,..); value NAME; text NAME;
      virtual: procedure open, close;
      begin
         text image;
         boolean OPEN;
           procedure open (T,.....); text T;....
           begin
                if OPEN then BLAD;
                OPEN:= true;
                image:= T;
                ........
           end;
           procedure close  (.);
           begin
                OPEN:= false;
                ..............
                image:= notext;
           end;
           procedure setpos (i); integer i; image. setpos (i);
           integer procedure pos; pos:= image. pos;
           boolean procedure more; more:= image.more;
           integer procedure length; length:= image.length;
        end; $ FILE $
        FILE class infile; virtual: procedure inimage; boolean
                           procedure endfile;
        begin
          procedure open ....;
          begin .........
                ENDFILE:= false;
                image:= notext;
                setpos (length + 1);
```

```
end; § open §
procedure close;....;
 begin ......;
     ENDFILE:= true;
 end;
boolean ENDFILE;
boolean procedure endfile; endfile:= ENDFILE;
procedure inimage;
begin
     if ENDFILE then BLÅD;
     ........
     setpos(1);
end;
character procedure inchar;
begin
     if not  more then
     begin
          inimage;
          if ENDFILE then BLÅD;
     end;
     inchar:= image.getchar;
end;
boolean procedure lastitem;
begin
L: if ENFFILE then lastitem:= true else
   Begin
M:    if not  more then
      begin
          inimage;
          goto L;
      end;
      if inchar = " " then goto M else
       setpos (pos - 1);
   end;
end;
integer procedure inint;
begin
     text T;
     if lastitem then BLÅD;
     T: - image.sub (pos,length-pos+1);
     inint:= T.getint;
     setpos(pos + T.pos - 1);
end;
real procedure inreal ......;
text procedure intext(w); integer w;
begin
     text T; integer m;
     T:-blanks (w);
     for m:=1 step 1 until w do
      T.putchar (inchar);
     intext:-T;
end;
.......
ENDFILE:=true;....
```

```
end; $ infile $
FILE class outfile; virtual: procedure outimage;
begin
   procedure open .....;
   begin ....; setpos(1); end;
   procedure close ...;
   begin if pos/=1 then outimage; end;
   procedure outimage;
       begin
            if not OPEN then BLAD;
            image:= notext;
            setpos (1);
       end;
   procedure  outchar (c); character c;
   begin
       if not more then outimage;
       image.putchar (c);
   end;
   text procedure FIELD (w); integer w;
   begin
       if w < = 0∨w > length then BLAD;
       if pos +w-1 > length then outimage;
       FIELD:- image.sub (pos,w);
       setpos (pos + w);
   end;
   procedure outint (i,w); integer i,w;
   FIELD (w).putint (i);
   procedure outfix (r,n,w); real r; integer n,w;
   FIELD (w). putfix (r,n);
   procedure outreal (r,n,w); real r; integer n,w;
   FIELD (w).putreal (r,n);
   procedure outtext (T); value T; text T;
      FIELD (T.length) :=T;
end $ outfile $
FILE class directfile; virtual: boolean procedure
                       endfile; procedure locate, inimage,
                       outimage;
begin
   integer LOC;
   integer procedure location ; location:= LOC;,
   procedure locate (i); integer i ;
   begin
        if not  OPEN then BLAD;
        .....
        LOC:= i;
   end;
   procedure open .....;
   begin
        .......
        setpos (1); locate (1);
   end;
   procedure close;......;
   boolean procedure endfile ...;
   procedure inimage;
   begin
```

```
            ......
            locate (LOC + 1);
            setpos (1)
        end;
        procedure outimage;
        begin ....
            locate (LOC + 1);
            image:= notext;
             setopos (1);
        end;
        character procedure inchar ;.....;
        boolean procedure lastitem;.....;
        integer procedure inint;.....;
        real procedure inreal;.....;
        text procedure intext;.....;
        procedure outchar;.....;
        text procedure FIELD;...;
        procedure outint;....;
        procedure outfix;....;
        procedure outreal;....;
        procedure outtext;....;
    end; $ directfile $
outfile class printfile;
begin
integer LINES PER PAGE, LINE, SPACING;
integer procedure line; line:= LINE;
procedure lines per page (n); integer n;
    LINES PER PAGE:=n;
procedure spacing (n);  integer n;
          SPACING:=n;
procedure eject (n) integer n;
begin
     if not OPEN then BLAD;
     if n > LINES PER PAGE then n:=1;
     .......
     LINE:=n;
end;
procedure PAGE;.....;
procedure open.....;
begin ......;
 setpos (1);  eject (1);
end;
procedure close;
begin,...;
 if pos/=1 then outimage;
   SPACING:=1;
   eject (LINES PER PAGE);
   LINES PER PAGE:=....;
   LINE := ϕ ;
 end;
 procedure outimage;
 begin
       if not OPEN ∨ image == notext  then BLAD;
       if LINE > LINES PER PAGE then eject (1);
```

```
        LINE:=LINE + SPACING;
         image := notext;
         setpos (1);
    end;
    LINES PER PAGE := .....;
    SPACING:=1;
end; $ PRINTFILE $ ;
 $ program główny klasy BASICO $
 SYSIN :- new infile ('SYSIN');
 SYSOUT:- new printfile ('SYSOUT');
 SYSIN.open (blanks (80));
 SYSOUT.open (blanks (LINELENGTH));
 inner;
 SYSIN.close;
 SYSOUT.close;
end; $ BASICIO $
```

Dodatek 2. Generatory liczb pseudolosowych (EMC IRIS 80)

Niech $\varrho(U)$ będzie liczbą wylosowaną zgodnie z rozkładem równomiernym na przedziale $[0,1]$.

```
• boolean procedure draw (A,U); name U; real A; integer U;
  draw := ϱ(U) < A;
• integer procedure randint (A,B,U); name U; integer A,B,U;
  begin
    if A>B then BŁĄD;
    randint := A + entier(ϱ(U) * (B-A+1));
  end;
• real procedure uniform (A,B,U); name U; real A,B; integer U;
  uniform:=A + ϱ(U) * (B-A);
• real procedure negexp (A,U); name U; real A; integer U;
  negexp := - logn( ϱ(U)) / A;
• real procedure normal (A,B,U); name U; real A,B; integer U;
  normal:= A + B *sqrt (-logn( ϱ (U)) * cos (π * ϱ (U)) *  sqrt(2));
• integer procedure poisson(A,U); name U; real A; integer U;
  if A < 20 then
  begin real work,max; integer N;
    work:=1; max := exp (-A);
    for work:=work*ϱ(U) while work > max do
    N:=N+1;
    poisson:=N;
  end else poisson:=entier(normal(A,sqrt(A), U) +0.5);
• real procedure erlang (A,B,U); name U; real A,B; integer U;
  begin real work; integer C,N;
    C:=entier(B); work:=1;
    for N:=1 step 1 until C do
    work:=work *ϱ(U);
    erlang:=(-logn(work)-(B-C) * long( ϱ (U))) / (A*B);
  end;
```

W pozostałych procedurach przyjęto następujące oznaczenia:

A.inf - najmniejsza wartość wskaźnika tablicy A,
A.sup - największa " " "
A.wym - wymiar tablicy A.

```
• integer procedure discrete(A,U); name U; array A; integer U;
  begin integer i; real val;
    if A.wym /= 1 then BLAD;
    val := ρ(U);
    for i:=A.inf step 1 until A.sup do
    if val < A(i) then goto L;
  L: discrete := i;
  end;
• real procedure linear(A,B,U); name U; array A,B; integer U;
  begin real y; integer i;
    y:=ρ(U); if A.wym /= 1 or B.wym /= 1 then BLAD;
    for i:=A.inf step 1 until A.sup do
    if y <= A(i) then goto L;
    BLAD;
  L: linear:= B(i-1)+(B(i)-B(i-1)) * (y-A(i-1)) / (A(i)-A(i-1));
  end;
• integer procedure histd (A,U); name U; array A; integer U;
  begin real max; integer i;
    max:=0;
    for i:=A.inf step 1 until A.sup do
    if max < A(i) then max:=A(i);
    if max <= 0 then BLAD;
  L: i:=A.inf + ρ(U) * (A.sup-A.inf);
     if max*ρ(U) > A(i) then goto L;
    histd := i;
  end;
• procedure histo (A,B,C,D); array A,B; real C,D;
  begin integer i;
    if A.wym /= 1 or B.wym /= 1 then BLAD;
    for i:=A.inf step 1 until A.sup do
    if C < B(i) then goto L;
    BLAD;
  L: A(i):=A(i) + D;
  end;
```

Literatura

[1] Birtwistle G. i in.: Notes on the Simula Language. Publication No. S-7 NCC Forskningsveien B, Oslo 1969

[2] Dahl O.I. i in.: Common Base Language. Simula Information, Publication No. S-22 NCC Forskningsveien 1 B Oslo 1970

[3] Birtwistle G. i in.: Simula Begin. Auerbach Publishers Inc. Philadelphia, Pa, 1973

[4] Simula sous SIRIS7/SIRIS8 manuel d'utilisation. Tome 2 CII, 1973

[5] Oktaba H., Ratajczak W.: Simula 67 . Warszawa: WNT 1980

[6] Poznański Z.: Simula-67 - Uniwersalny język programowania. Biuletyn Informacyjny Obiektowe Systemy Komputerowe 1979 nr 3.

mgr Janina Oleda
Instytut Maszyn Matematycznych

Komputeryj w kontroli jakości

Wstęp

Niniejsze opracowanie jest próbą takiego przeanalizowania pełnego procesu powstawania wyrobów, które pozwoliłoby określić możliwości wykorzystywania komputerów w statystycznej kontroli jakości (SKJ) wyrobów produkcji masowej.

Opracowanie to wraz z opracowaniem Teresy Friling [3] mogłoby stać się podstawą do sformowania założeń dla pakietu programów wspomagających SKJ. W opracowaniu zaprezentowano więc najbardziej powszechne poglądy na określenie jakości, na czynniki decydujące o tejże jakości oraz na kryteria, według których ocenia się jakość. Przegląd taki ma na celu zapewnienie jednoznacznego rozumienia dalszych sformułowań.

Jakość wyrobu w dużej mierze decyduje o wynikach ekonomicznych producenta. Dlatego też w działalności przedsiębiorstw problem jakości stanowi jedno z podstawowych zadań. Obecnie na świecie przeważa pogląd, że wysiłki przedsiębiorstwa powinny koncentrować się na zagadnieniu utrzymania wysokiego poziomu jakości produkowanych wyrobów. W naszej rzeczywistości gospodarczej nie jest to jednak jeszcze powszechna praktyka. Potwierdzeniem tego są wyniki z badań ankietowych w 4 dużych zakładach przemysłu maszynowego przeprowadzonych w 1977 r. przez Zakład Badań Społecznych Instytutu Organizacji Przemysłu Maszynowego. Stwierdzono, że producenci nadal większy nacisk kładą na ilość niż na jakość produkcji. Kontroluje się jedynie wyrób gotowy oceniając, czy jest dobry czy zły. Nie ma to jednak wpływu na jakość osiąganą w poszczególnych fazach technologicznych, inaczej mówiąc nie wpływa to na proces technologiczny. Łączy się z tym niezmiernie częste stosowanie przez producentów kart odstępstw od ustalonych w dokumentacji technicznej wymagań, co pozwala na obniżanie jakości wyrobów [2]. Sytuacja taka powoduje, że w strategii działania, nie tyle nawet przedsiębiorstw, a bardziej zjednoczeń i resortów obserwuje się wysiłki skierowane jedynie na poprawę jakości wyrobu. Natomiast bardziej pożądana byłaby sytuacja,kiedy produkowany wyrób od początku produkcji w przedsiębiorstwie osiągałby wysoki poziom jakości, a wysiłki przedsiębiorstwa skierowane byłyby na działanie zapobiegające obniżeniu tej jakości. Taka praktyka stosowana jest w przedsiębiorstwach wysoko uprzemysłowionych państw, w których główny nacisk kładzie się na zapobieganie sytuacjom, które mogą doprowadzić do obniżenia jakości wyrobów [2] .

Potwierdzeniem słuszności takiej praktyki są renomowane firmy światowe, które dzięki wysokiej jakości produkowanych wyrobów utrzymują się na światowych rynkach zbytu.

Odpowiedzialność za jakość wyrobów spoczywa zawsze na ich producencie. Dla kierownictwa przedsiębiorstwa jakość wyrobu powinna być problemem o wyraźnie ekonomiczno-handlowym charakterze. Rys.1 [8] pokazuje straty przedsiębiorstwa, wynikające z produkcji wyrobów złej jakości.

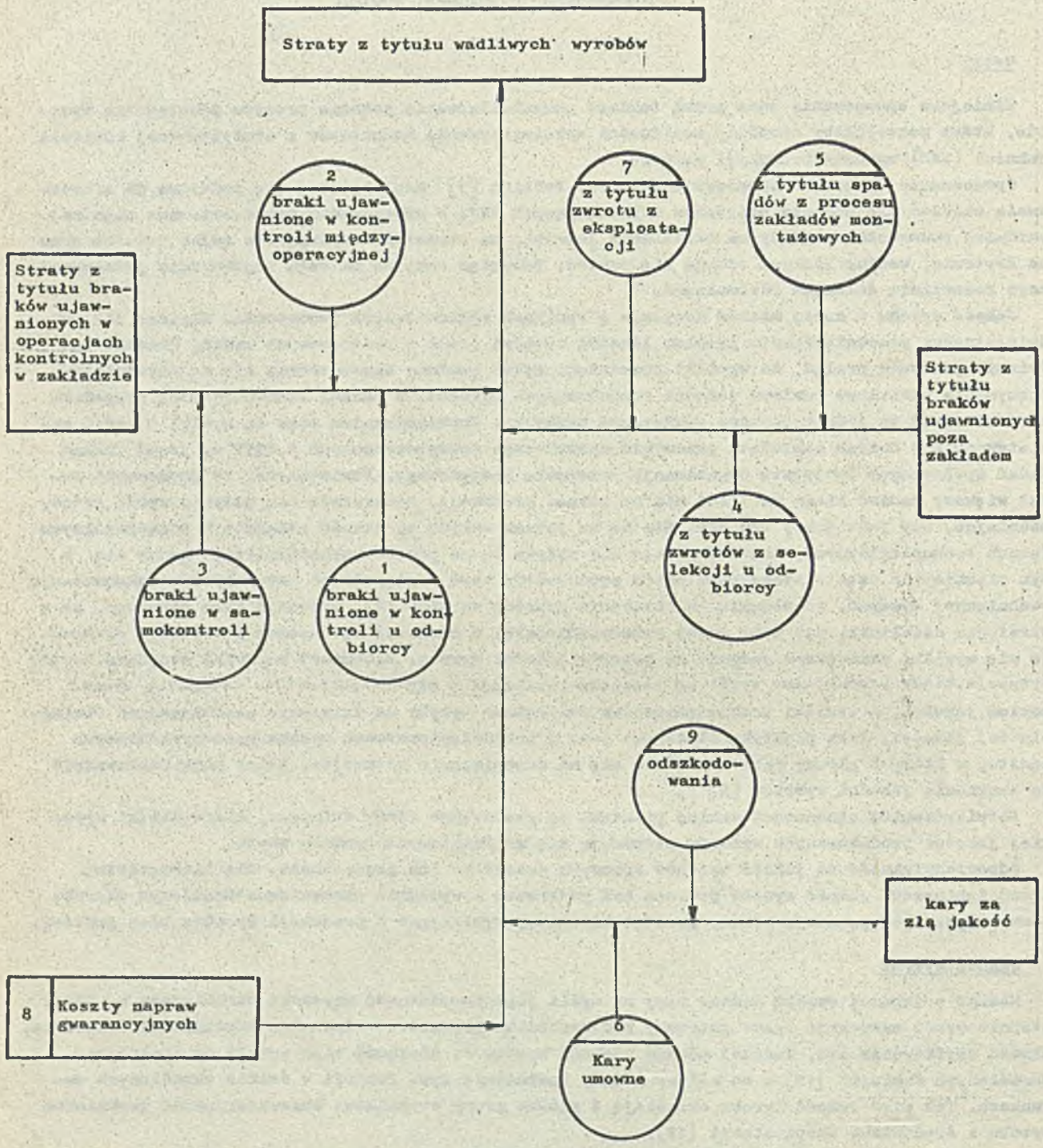
Jakość wyrobu

Mówiąc o jakości wyrobu zawsze mamy na myśli jego przydatność użytkową określającą w jakim stopniu wyrób zaspokaja nasze potrzeby pod względem sprawności działania, dokładności działania, okresu użytkowania itd. Inaczej mówiąc "jakość wyrobu to zdolność tego wyrobu do spełniania określonych funkcji" [12] i co należy dodać, spełnianie tych funkcji w ściśle określonych warunkach. Tak więc jakość wyrobu określają 2 główne grupy czynników: charakterystyka techniczna wyrobu i środowisko eksploatacji [12].

Pierwsza grupa obejmuje zbiór takich cech technicznych, które zachowują zawsze te same wartości bez względu na miejsce dokonywania oceny tych wartości.

Drugą grupę czynników decydujących o jakości wyrobu charakteryzuje całokształt warunków, w jakich wyrób mógłby być wykorzystany.

Przyjęto więc, że wśród czynników określających wyrób, dalej nazywanych parametrami wyrobu, można wyodrębnić następujące grupy [5], [12], [11]:



Rys.1. Układ strukturalny strat powstających z tytułu wykonywania wadliwych wyrobów [8]

- parametry użyteczności.
- parametry niezawodnościowe
- parametry decydujące o gatunku wyrobu
- parametry określające estetykę wyrobu.

Poniżej zostaną omówione pokrótce wymienione grupy parametrów.

Użyteczność wyrobu określona jest stopniem zaspokajania potrzeb danego użytkownika, co oznacza, że w zależności od środowiska, w którym wyrób będzie eksploatowany może mieć on różne charakterystyki techniczne. L. Wasilewski [12] nazywa to różnicowaniem jakości typu i wyodrębnia 3 kierunki różnicowania określające jakość typu:

- generację, jeśli daje się wyodrębnić całkowicie inne rozwiązania oparte na odmiennych koncepcjach.
- odmianę, dającą różnicowanie wyrobów w zależności od środowiska, w którym wyrób będzie eksploatowany,
- klasę wyrobu, mieszczącą się w każdej odmianie i charakteryzującą się różnym poziomem niektórych cech użytkowych.

Niezawodność wyrobu to "prawdopodobieństwo, że w danych warunkach produkt (wyrób) będzie spełniał bez uszkodzeń określoną funkcję" [5]. W jakości wyrobu parametr ten ma niezwykle istotne znaczenie, a ponadto często decyduje o zdrowiu i życiu ludzkim.

Znaczenie zagadnień niezawodności jest tak ogromne, że opracowana została już osobna teoria stanowiąca wyodrębnioną dyscyplinę naukową, wykładaną już jako samodzielny przedmiot studiów w wielu wyższych uczelniach, m.in. w Polsce [7].

Niezawodność oceniana jest różnorodnymi wskaźnikami, na przykład można tu wymienić czasy międzyawaryjne strumienia uszkodzeń, czy czasy napraw. Zestaw takich podstawowych parametrów odnoszących się do sprzętu komputerowego i zależności zachodzące między tymi parametrami scharakteryzowano w opracowaniu [1].

Gatunek wyrobu zależy od rodzaju odchyleń od standardu. Wszelkie odstępstwa od wzorca oznaczają, że już w momencie wyjścia wyrobu z produkcji nie odpowiada on założonym wymaganiom. Tak więc można uznać, że parametry grupy trzeciej są w zasadzie pochodnymi względem dwu pierwszych grup i oznaczają, w jakim stopniu wykonanie wyrobu zgodne jest z wzorcem. Wobec tego w dalszym ciągu niniejszego opracowania parametry tej grupy nie będą szczegółowiej rozważane.

Podobnie w zasadzie pomijane będą parametry grupy czwartej - estetyki wyrobu, które są parametrami uzupełniającymi, nie wpływającymi na techniczną użyteczność wyrobu. Należy jednak wspomnieć, że właśnie w tej grupie parametrów powinny znaleźć wyraz te zalecenia ergonomii, które mówią, że aby jego eksploatacja nie była utrudniona - wyrób musi być dostosowany do pewnych cech człowieka. Estetyka wyrobu odgrywa niemałą rolę zwłaszcza w wypadku wyrobów przeznaczonych do użytkowania masowego - indywidualnego i może wówczas decydować o popycie. Aby więc wybrane parametry tej grupy móc kontrolować w sposób jednoznaczny, są one zazwyczaj wyrażane w odpowiedniej formie ścisłej, stając się tym samym parametrami grupy pierwszej.

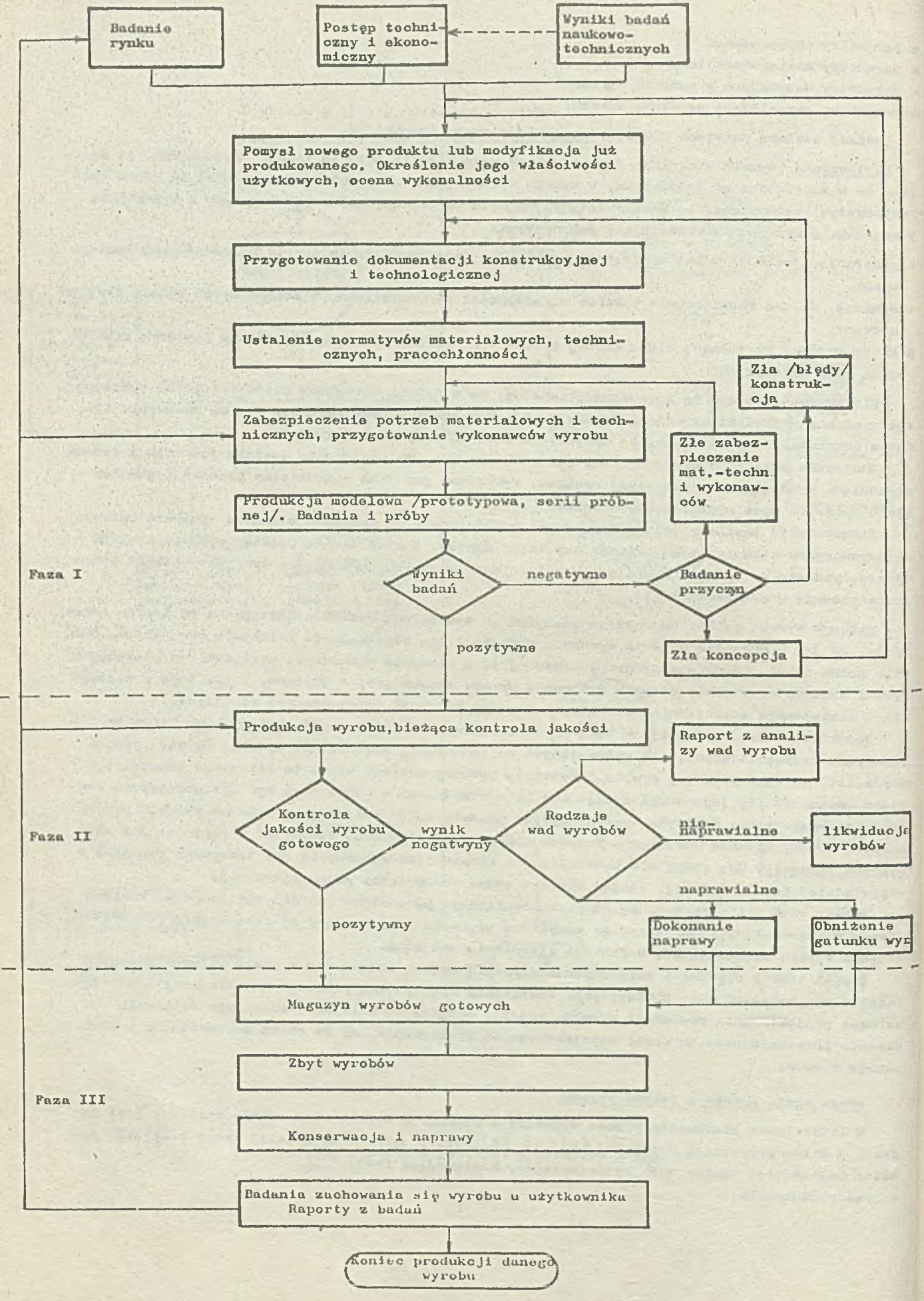
Należy zdać sobie sprawę, że fizyczna realizacja parametrów jakości występuje w różnych fazach wytwarzania wyrobu, oraz, że wszystkie czynności zmierzające do osiągnięcia jakościowo dobrego wyrobu zawierają się w funkcji sterowania jakością.

Zespół takich czynności związanych z jakością wyrobu składa się z wielu różnych rodzajów działań wzajemnie ze sobą powiązanych. Powiązania między nimi pokazano w niniejszej pracy jako schemat projektowania produkcji wyrobu (rys.2). Określa on kolejność ludzkiego działania od momentu powstania pomysłu czyli zapotrzebowania społecznego, aż do czasu zaprzestania produkcji danego wyrobu.

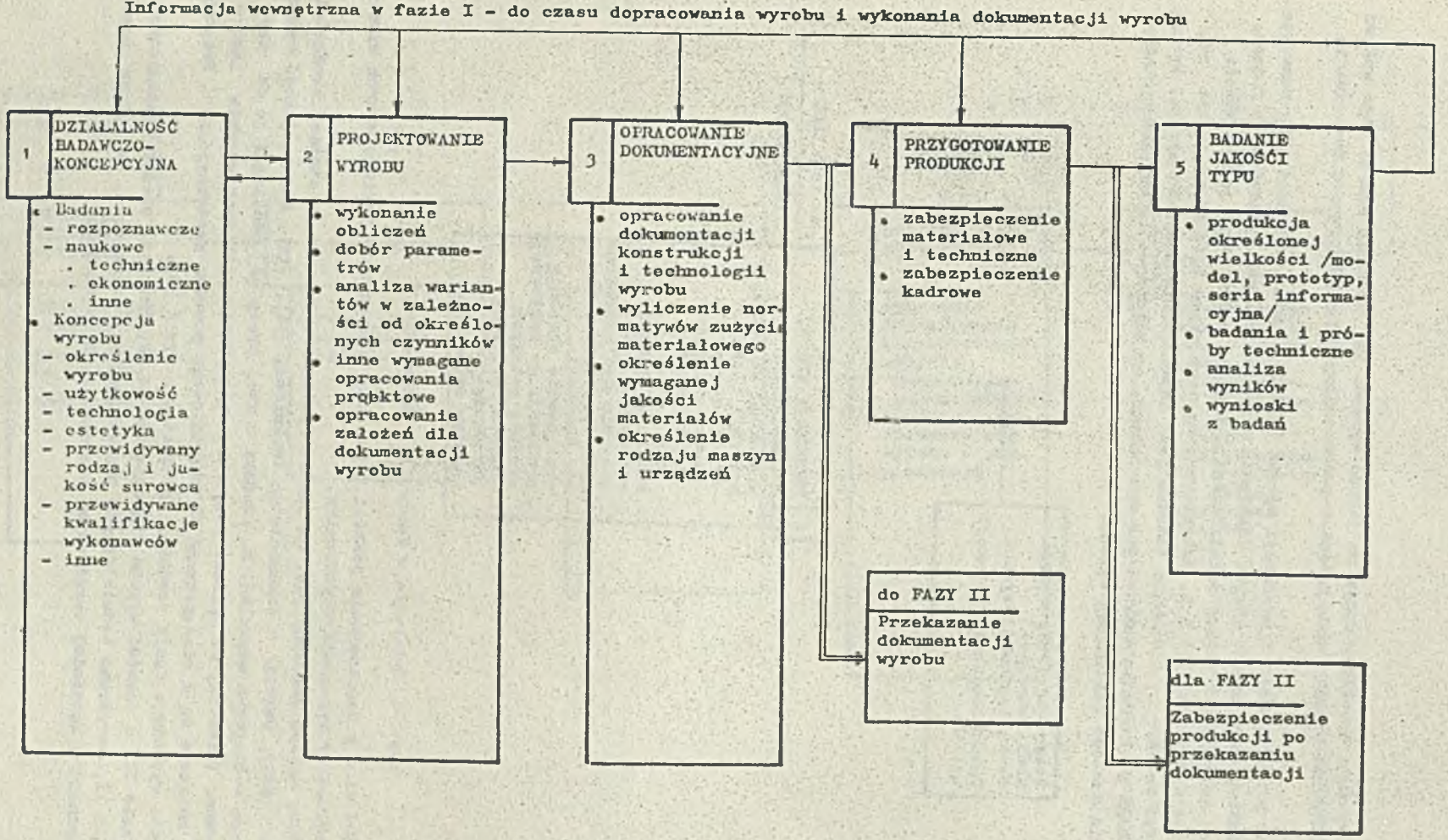
Droga życia wyrobu a jakość wyrobu

W literaturze przedmiotu proces związany z produkcją wyrobu powszechnie dzielony jest na fazy. Z kilku propozycji podziału [4],[5], [8], [11], [12] w niniejszej pracy przyjęto podział najbardziej ogólny [11] wyodrębniający następujące fazy:

- przedprodukcyjną,



Rys.2. Schemat projektowania i produkcji wyrobu

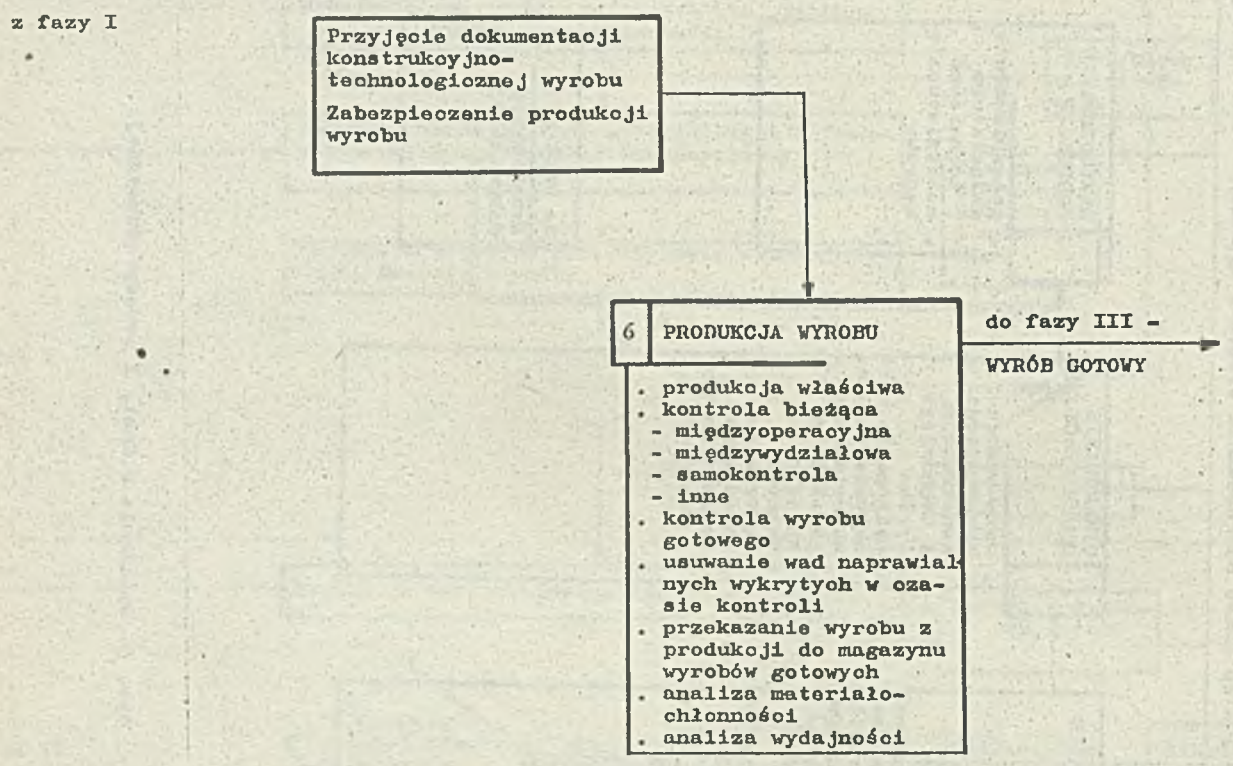


Rys. 3. Działania w fazie I przedprodukcyjnej

- produkcyjną,
- poprodukcyjną.

Każda z tych faz wywiera określony wpływ na jakość wyrobu. Aby określić zakres tego wpływu konieczne jest prześledzenie etapów działalności w poszczególnych fazach procesu powstawania wyrobu.

Analizując schemat projektowania i produkcji wyrobu (rys.2) można stwierdzić, jak przedstawia się kolejność i zakres określonych etapów działania w poszczególnych wymienionych fazach. Aby dokładniej pokazać kolejność i zasięg działalności związanej z wyrobem i jego jakością opracowano prezentowane dalej rys. 3,4 i 5. Zobrazowano na nich etapy działań wzajemnie się warunkujących. Zauważyć należy, że niezwykle istotne jest dopracowanie projektu wyrobu. Wymaga to czasami wielokrotnego powtarzania zakresu już wykonanego, co zdecydowanie zwiększa prawdopodobieństwo powstania dobrego jakościowo wyrobu.

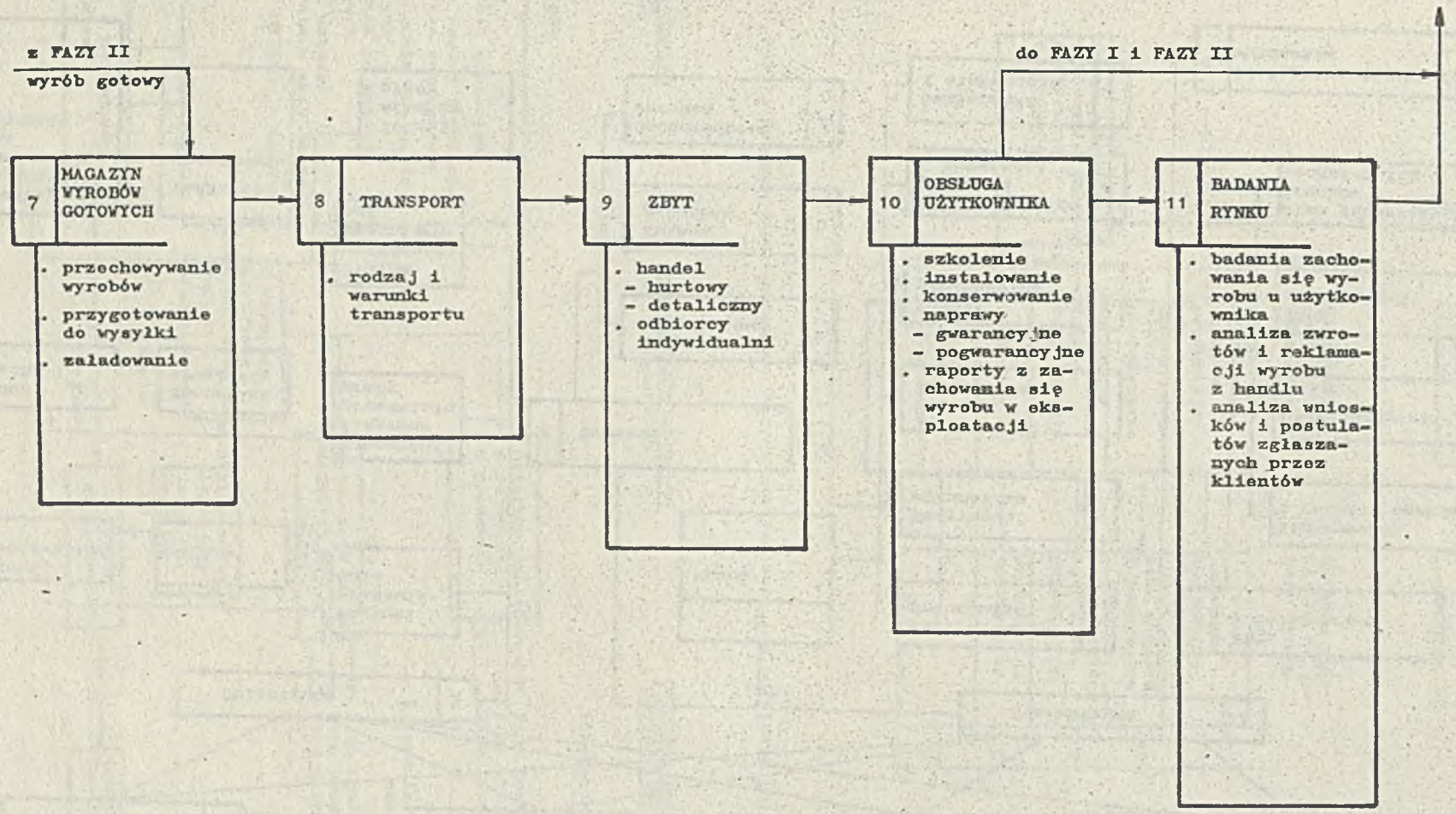


Rys. 4. Działania w fazie II - produkcyjnej

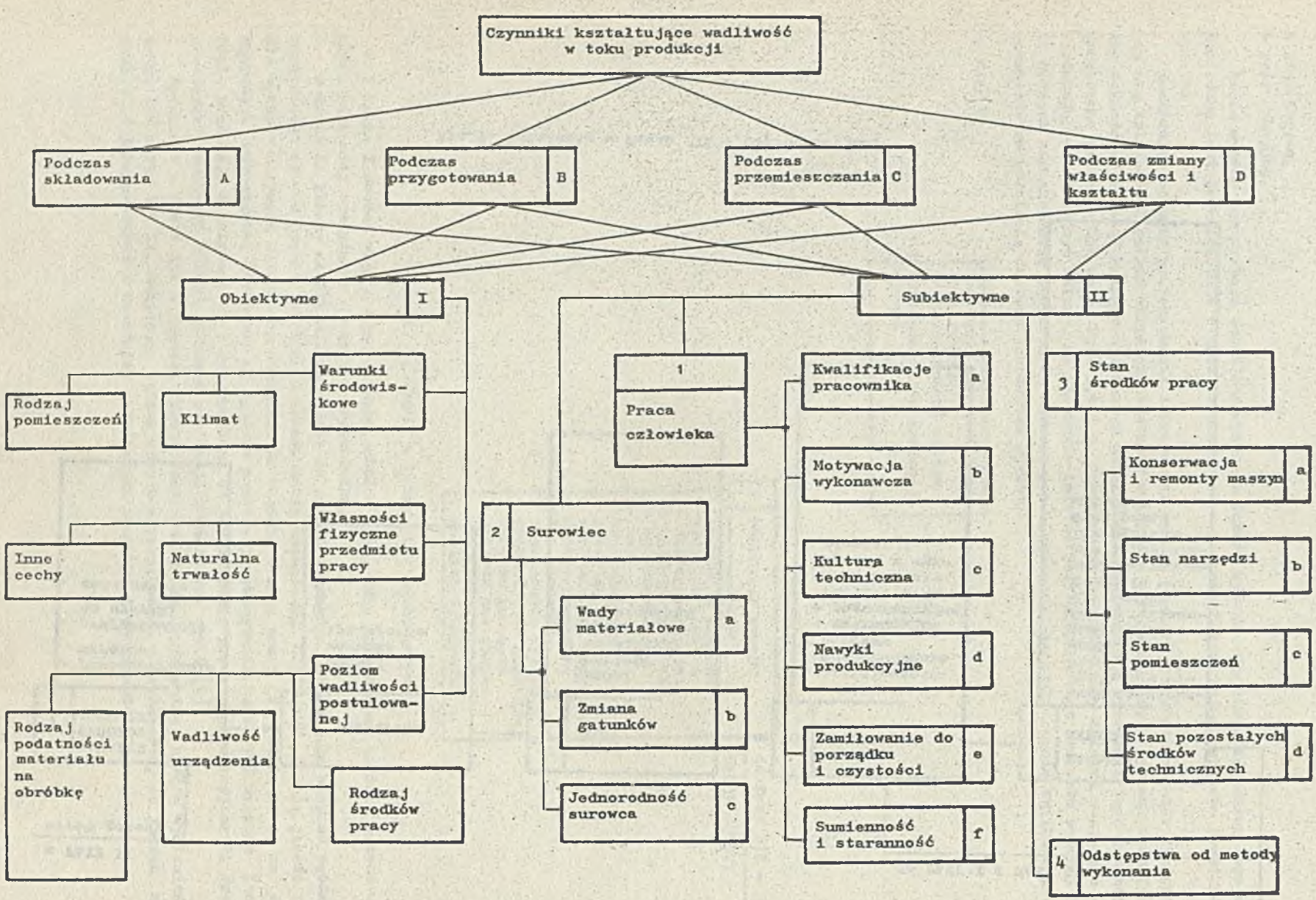
W fazie I chodzi więc o dopracowanie jakości typu, tzn. opracowanie takiego wzorca wyrobu, który najlepiej spełniałby wymagania użytkownika.

W fazie II główny nacisk położony jest na jakość wykonania. Oznacza to stopień zgodności konkretnego wyrobu (serii, partii) z dokumentacją techniczną [11], [12]. Na złą jakość wykonania wyrobów wpływają różnorodne czynniki. M. Okoński w swej pracy [8] podzielił je na zewnętrzne i wewnętrzne. Twierdzi, że "główne przyczyny wykonywania wadliwych wyrobów leżą w fazie produkcji i związane są z odstępstwami od prawidłowego procesu produkcyjnego". Przyczyny powodujące wadliwość wyrobów w fazie produkcji przedstawia rys.6 [8].

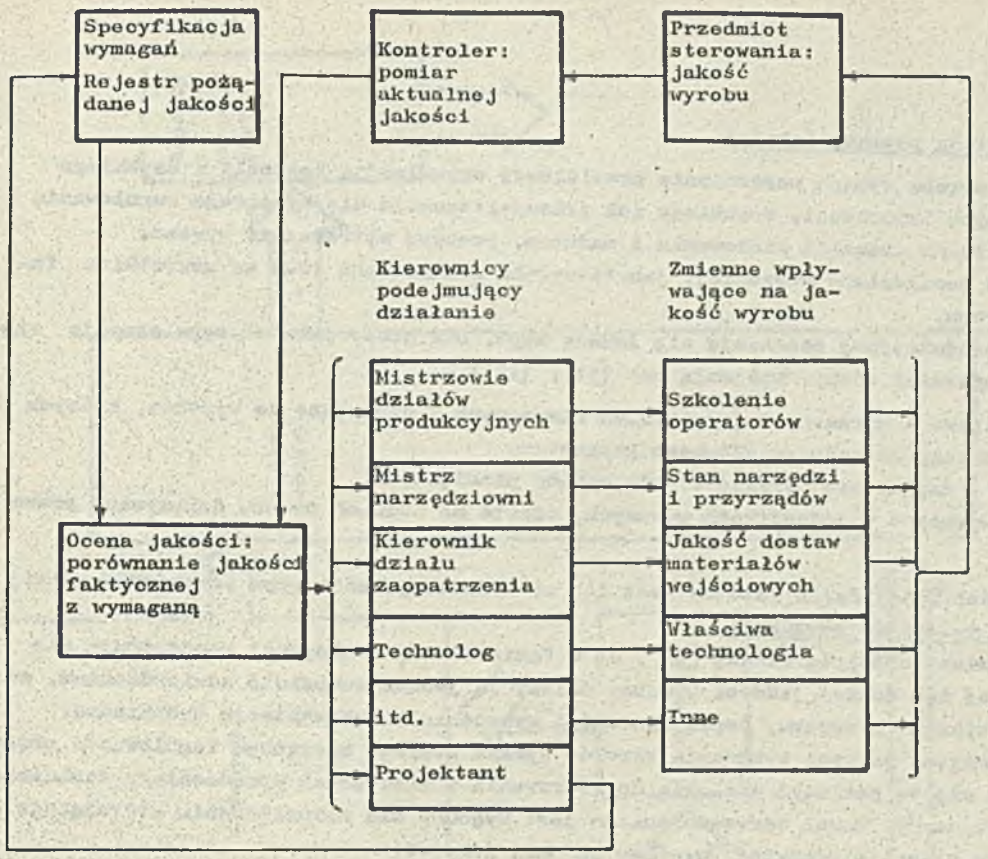
Natomiast w fazie III o jakości wyrobu trafiającego do użytkownika, oprócz wymienionych czynników z faz I i II, decydować będzie sposób magazynowania i transportu, co jednak można traktować jako zgodność z wcześniej określonymi wzorcami tych czynności.



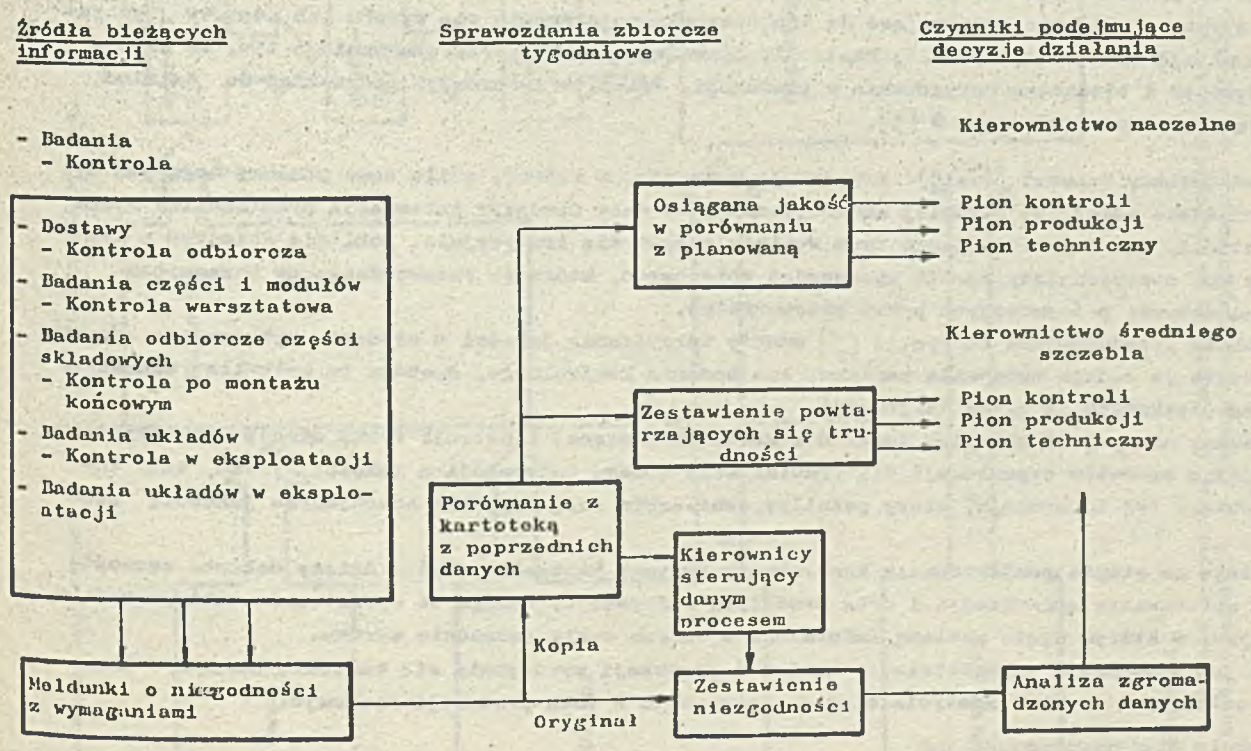
Rys.5. Działania w fazie III - poprodukcyjnej



Rys.6. Przyczyny wewnętrzne powstawania wadliwych wyrobów



Rys.7. Schemat serwomechanizmu sterowania jakością (na podstawie [4])



Rys.8.Schemat przepływu informacji potrzebnych do działania korygującego (na podstawie [5])

Kontrola jako metoda poprawy jakości

Troska o jakość wyrobu wymaga zapewnienia prawidłowej organizacji kontroli i szybkiego przekazywania zebranych informacji, szybkiego ich przeanalizowania dla bieżącego regulowania przez personel wszystkich szczebli kierowania i nadzoru, procesu wytwarzania wyrobu.

Zgodnie z tym co powiedziano wcześniej, jakość wyrobu realizowana jest we wszystkich fazach powstawania wyrobu.

W fazie I przedprodukcyjnej realizuje się jakość typu. Dla oceny jakości typu stosuje się często syntetyczne mierniki oceny. Obejmują one [11], [12]:

- oceny jednoparametrowe - wyrażane w jednostkach fizycznych i stosowane do wyrobów, których wartość użytkowa zależy od jednego głównego parametru,
- oceny ekonomiczne - dające się wyrazić w jednostkach pieniężnych,
- oceny punktowe - wyrażane w jednostkach umownych, oparte na ocenach wyrobu dokonywane przez ekspertów.

W kontroli jakości w tej fazie istotne jest to, że badania jakości typu przeprowadza się przed dopuszczeniem wyrobu do produkcji.

Przyjmując natomiast omówioną zasadę [8] , że w fazie II - produkcyjnej koncentrują się główne przyczyny złej lub dobrej jakości wyrobu, należy ją jednak uzupełnić stwierdzeniem, że dotyczy to jakości wykonania wyrobu, czyli zgodności wykonania z dokumentacją techniczną.

Zapewnienie należytej jakości wykonania wyrobów wymaga stałego bieżącego regulowania procesu produkcji. Można się tu posłużyć analogią do sterowania w niektórych urządzeniach technicznych przez serwomechanizmy. Model serwomechanizmu jest wygodny dla zilustrowania sterującego znaczenia kontroli w procesie produkcji wyrobów żądanej jakości.

Rys. 7 [5] przedstawia schemat serwomechanizmu, w którym rolę sterującą procesem produkcyjnym pełnią ludzie odbierający informacje z systemu kontroli jakości i reagujący na tę informację podejmowaniem odpowiednich decyzji, dotyczących dalszego przebiegu procesu produkcyjnego. Rysunek ten pokazuje jakie czynniki, czy inaczej mówiąc działania, wpływają na dobrą lub złą jakość wyrobu. Działania zmierzające do zapobiegania pogarszaniu się wyrobu lub poprawy jego jakości wymagają bieżącej sprawowanej kontroli, bieżącego szybkiego informowania o tym, co dzieje się z wyrobem i bieżącego korygowania w produkcji. Przepływ informacji potrzebnej do działań korygujących przedstawia rys. 8 [5].

Przedstawiony schemat (rys.8) prezentuje sytuację, w której, o ile same pomiary mogą już być przeprowadzane mniej lub bardziej automatycznie, to cały przepływ informacji oraz analiza wyników kontroli, jak i przetwarzanie tych wyników odbywa się tradycyjnie. Dobitnie świadczy o tym sztywny np. dwutygodniowy termin sprawozdań zbiorczych, które to sprawozdania są zasadniczą podstawą decyzji podejmowanych przez kierownictwo.

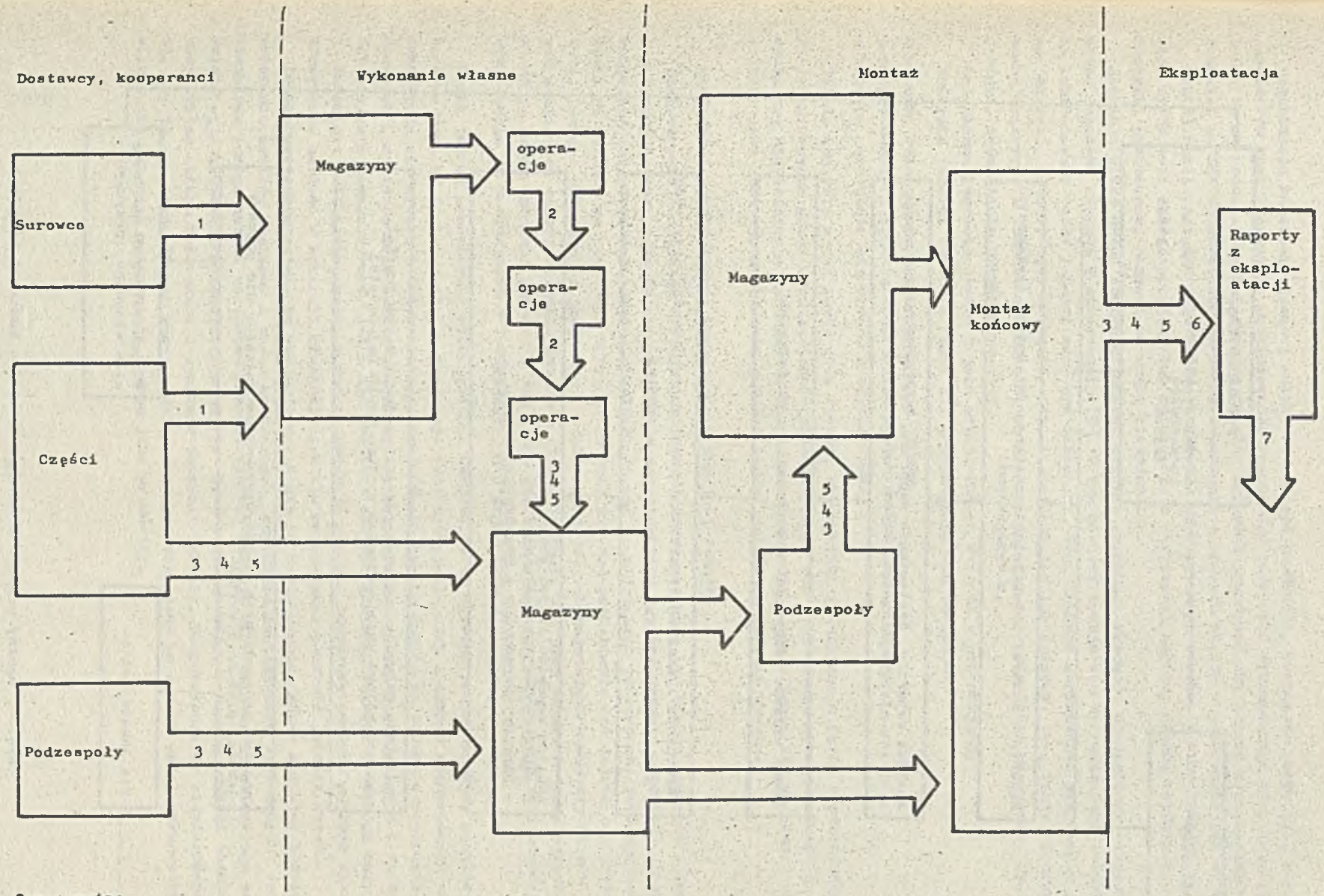
Podobnie przedstawiono na rys. 9 [5] punkty sprawdzania jakości w procesie wytwarzania odnoszą się jedynie do samego wykonania pomiaru, lub badania kontrolnego. Brakuje tu natomiast wskazania na obieg uzyskanych tą drogą informacji.

Pokazany na rys. 10 przebieg działań w kontroli bieżącej ilustruje tylko zasadę merytoryczną nie podając sposobów organizacji dla prawidłowego obiegu informacji o jakości wyrobu, tzn. takiego obiegu tej informacji, który pełniłby skutecznie rolę czynnika sterującego jakością produkcji.

Dlatego na etapie projektowania konkretnego systemu kontroli jakości należy dokonać szczegółowego opracowania organizacji i dróg przepływu informacji. Wymaga to opracowania takiego modelu przepływu, w którym ujęte zostaną informacje z całego cyklu tworzenia wyrobu.

Jak już wspomniano parokrotnie, w kontroli produkcji wyodrębnia się kontrolę bieżącą i kontrolę ostateczną wyrobu. Kontrola bieżąca dokonywana w toku produkcji obejmuje:

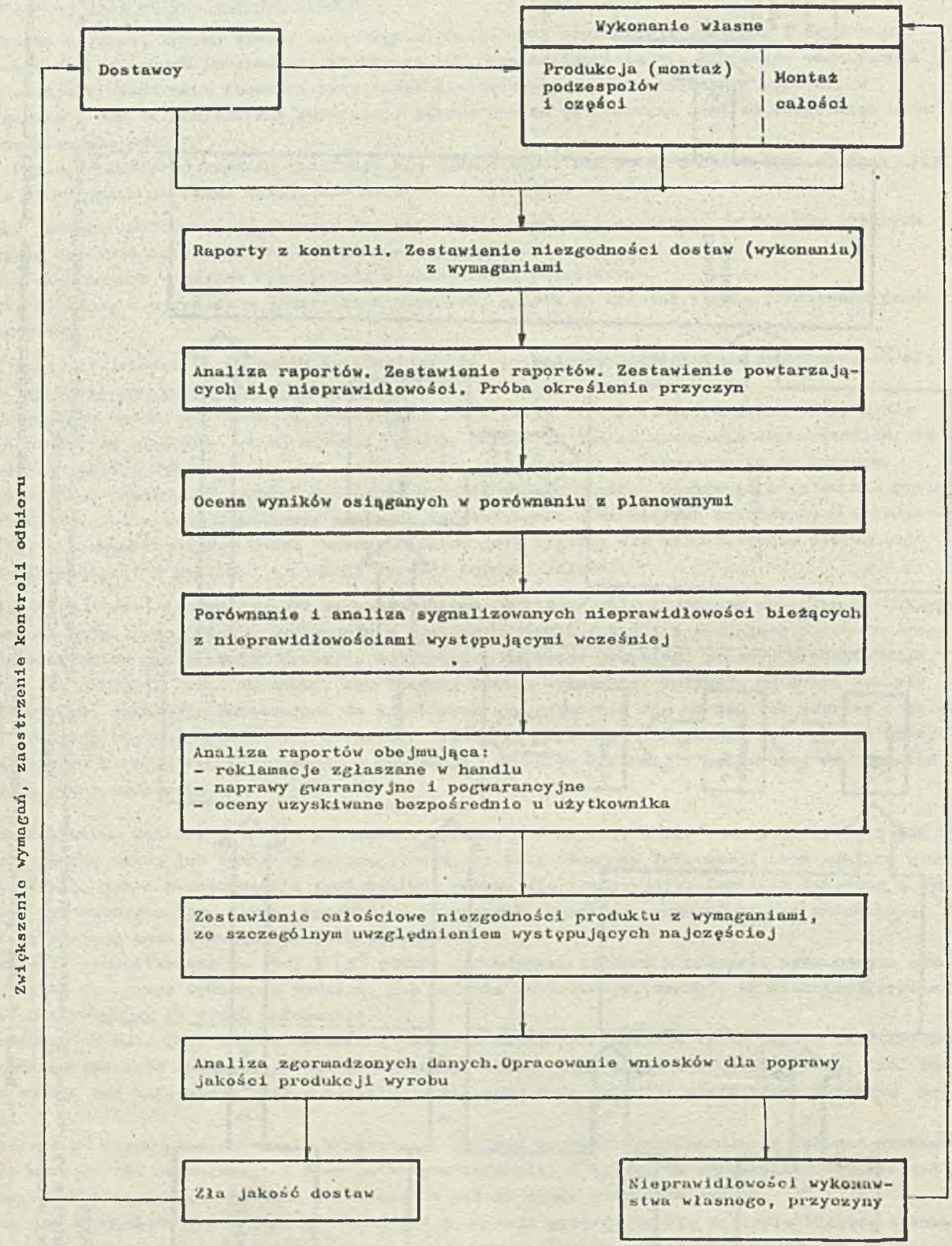
- kontrolę międzyoperacyjną lub
- kontrolę międzywydziałową oraz
- samokontrolę.



Oznaczenia:

1 - kontrola wyrywkowa, 2 - kontrola międzyoperacyjna, 3 - kontrola wymiarów, 4 - badanie cech funkcjonalnych, 5 - badania wyrywkowe bieżącej produkcji, 6 - kontrola przygotowania do wysyłki, 7 - raporty z eksploatacji

Rys.9. Schemat przebiegu procesu produkcyjnego ze wskazaniem punktów sprawdzania jakości i niezawodności [5]



Rys. 10. Ogólny schemat przebiegu działań w kontroli bieżącej

Organizacja kontroli bieżącej zależy od rodzaju produkcji, rozmiarów produkcji, rozkładu cech parametrów kontrolowanych, przygotowania zawodowego personelu wykonawczego itp. Z założenia kontrola ta ma pozwalać na korygowanie działań mogących powodować złą jakość produkowanych wyrobów. Pozwalać ma ona równocześnie na szybkie usuwanie wykrytych błędów w poszczególnych wyrobach.

Takiej możliwości nie daje kontrola końcowa produkowanego wyrobu. W konsekwencji tej ostatniej wyrób może zostać przyjęty lub odrzucony, a możliwości naprawy błędów wykrytych w wyniku kontroli ostatecznej czasami są wręcz nieopłacalne.

Istnieje wiele metod kontroli jakości wyrobów produkcji masowej. Szczegółowe omówienie tych metod dokonane zostało w opracowaniu [3]. W tym miejscu wydaje się wystarczające stwierdzenie, że kontrolowane cechy jakości mogą być mierzalne, tzn. zbadane odpowiednim przyrządem pomiarowym, np. długość, ciężar, kąt, napięcie, itp. i niemierzalne lub opisowe, tzn. takie, których nie można zmierzyć, a jedynie stwierdzić, że zaistniało (lub nie) określone zdarzenie, np. świecenie żarówki, zadziałanie przekaźnika, itp.

Ostateczna kontrola wyrobu i jego odbiór kończą fazę II - produkcyjną.

Kontrola jakości wyrobu w fazie III - poprodukcyjnej obejmuje:

- warunki składowania i przechowywania wyrobu,
- przygotowanie i przetransportowanie wyrobu do odbiorcy,
- badanie zachowania się wyrobu u użytkownika.

Istotą kontroli w tej fazie jest troska, aby wysiłki poniesione w fazie produkcyjnej nie zostały zniweczone przez nieodpowiednie przechowywanie lub uszkodzenie wyrobu wskutek złego załadunku czy niewłaściwego transportu. Można zatem powiedzieć, że w fazie III w głównej mierze prowadzona jest kontrola zachowania jakości typu i jakości wykonania wyrobu.

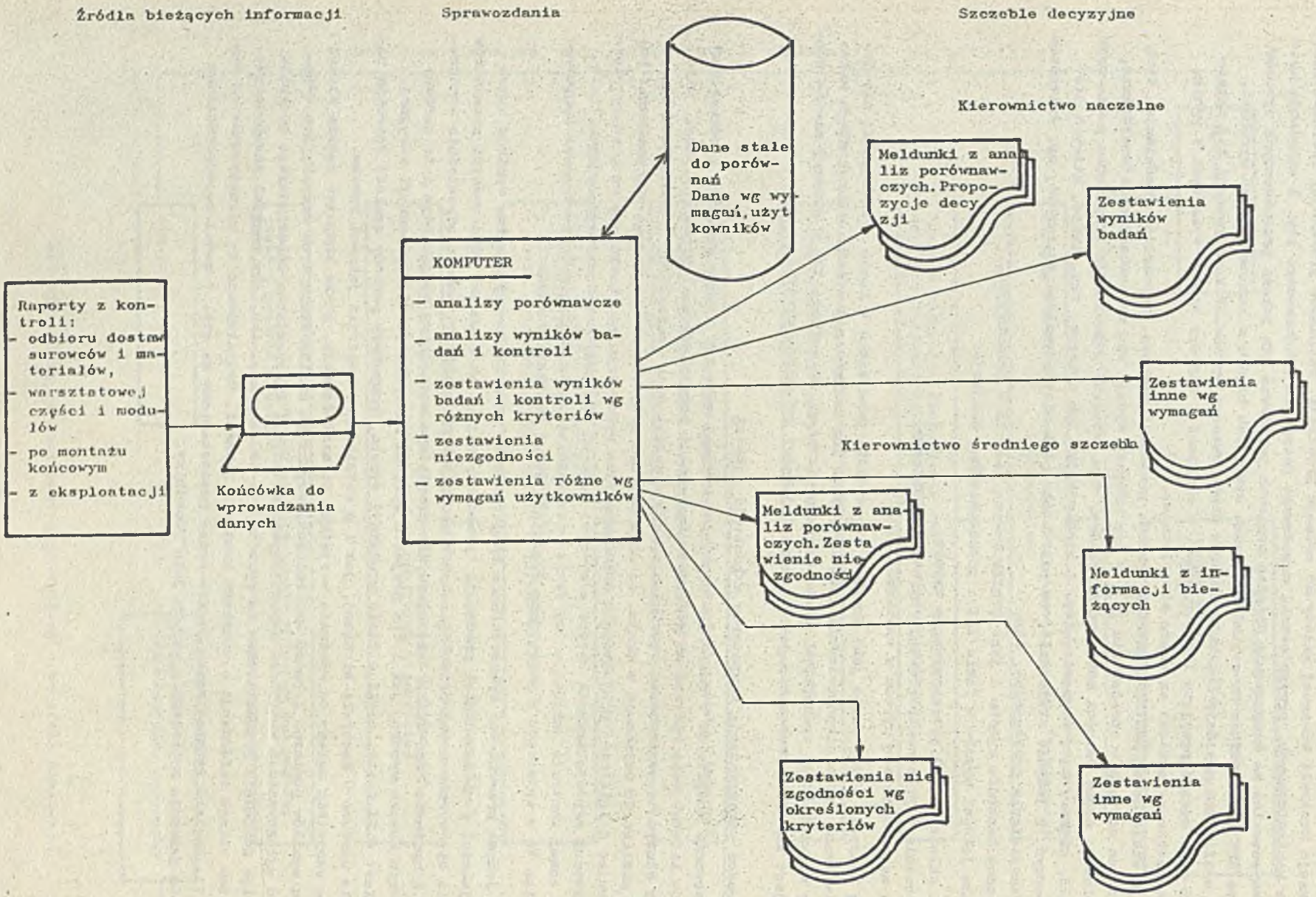
Możliwości wykorzystania komputerów w kontroli jakości

Zastosowanie komputerów w fazie I - przedprodukcyjnej może być bardzo szerokie i różnorodne i w sposób istotny może wpływać na podniesienie jakości przygotowywanej produkcji. Mogą więc komputery służyć do gromadzenia wszelkich danych o materiałach (surowcach, podzespołach itp.), które są możliwe do uzyskania w kraju, co może być dla projektanta nie dającą się przecenić pomocą. Również działalność projektowa i dokumentacyjna wykorzystująca komputery ma o wiele szersze możliwości zaprojektowania wyrobu o wyższej jakości niż przy metodach tradycyjnych. Natomiast do samej kontroli jakości w tej fazie nie widać na razie konkretnych zastosowań komputerów. Wynika to z opisanych w poprzednim rozdziale zadań kontroli w fazie I.

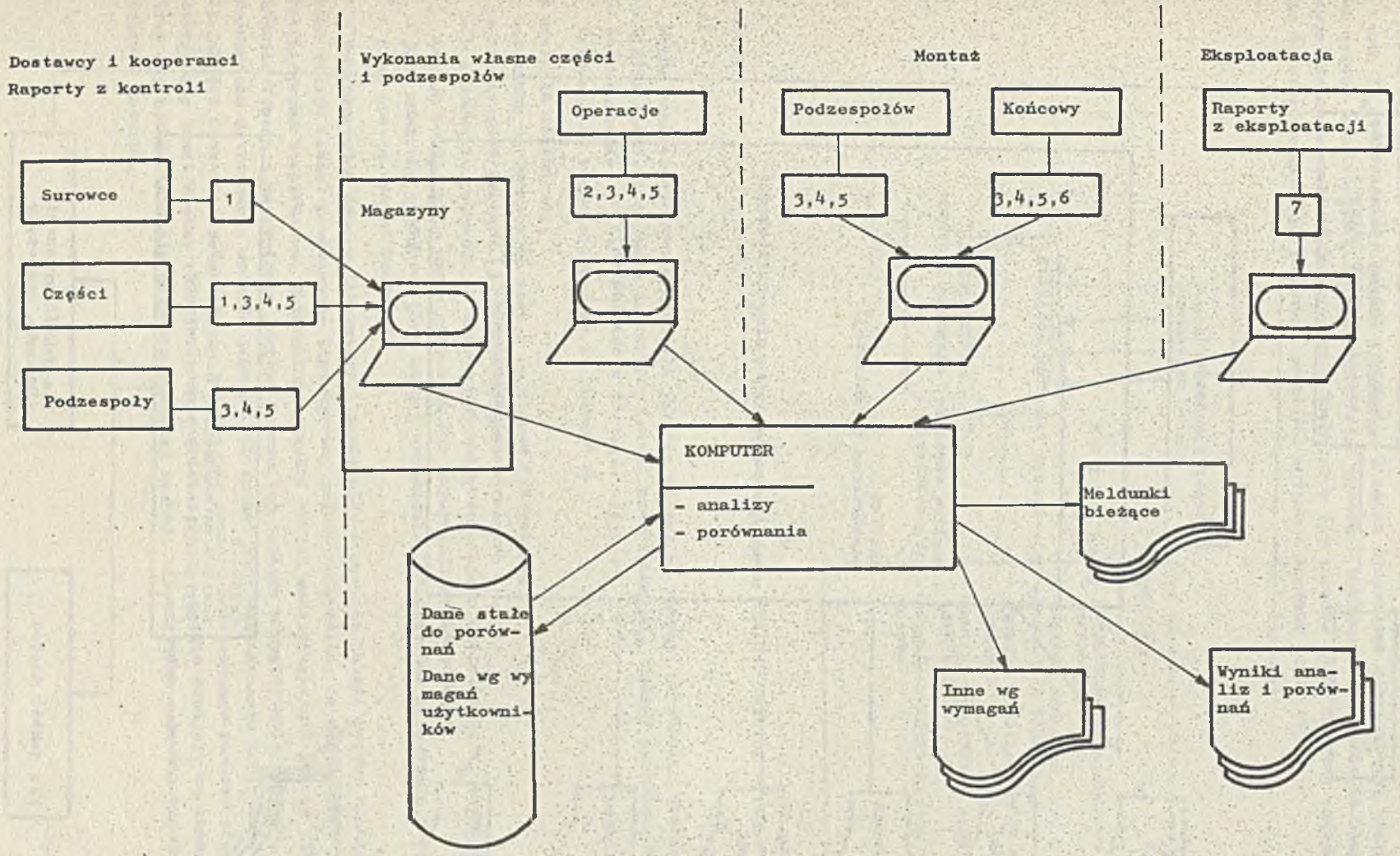
Można jednak przewidywać wykorzystanie komputerów do wszechstronnej analizy wyników badań zarówno jakości i niezawodności produkcji "przedseryjnej", jak i innych różnorodnych parametrów zachowania się wyrobu z tej produkcji. Wykonywanie takich analiz za pomocą odpowiednio oprogramowanych komputerów może okazać się niezwykle przydatnym narzędziem projektowym i to zarówno dla projektu samego wyrobu, jak i dla projektu systemu kontroli jakości produkcji seryjnej.

Natomiast w II fazie, czyli w fazie produkcji wyrobu, komputery powinny znaleźć szerokie zastosowanie zarówno w kontroli bieżącej, jak i w ostatecznej kontroli jakości wyrobu.

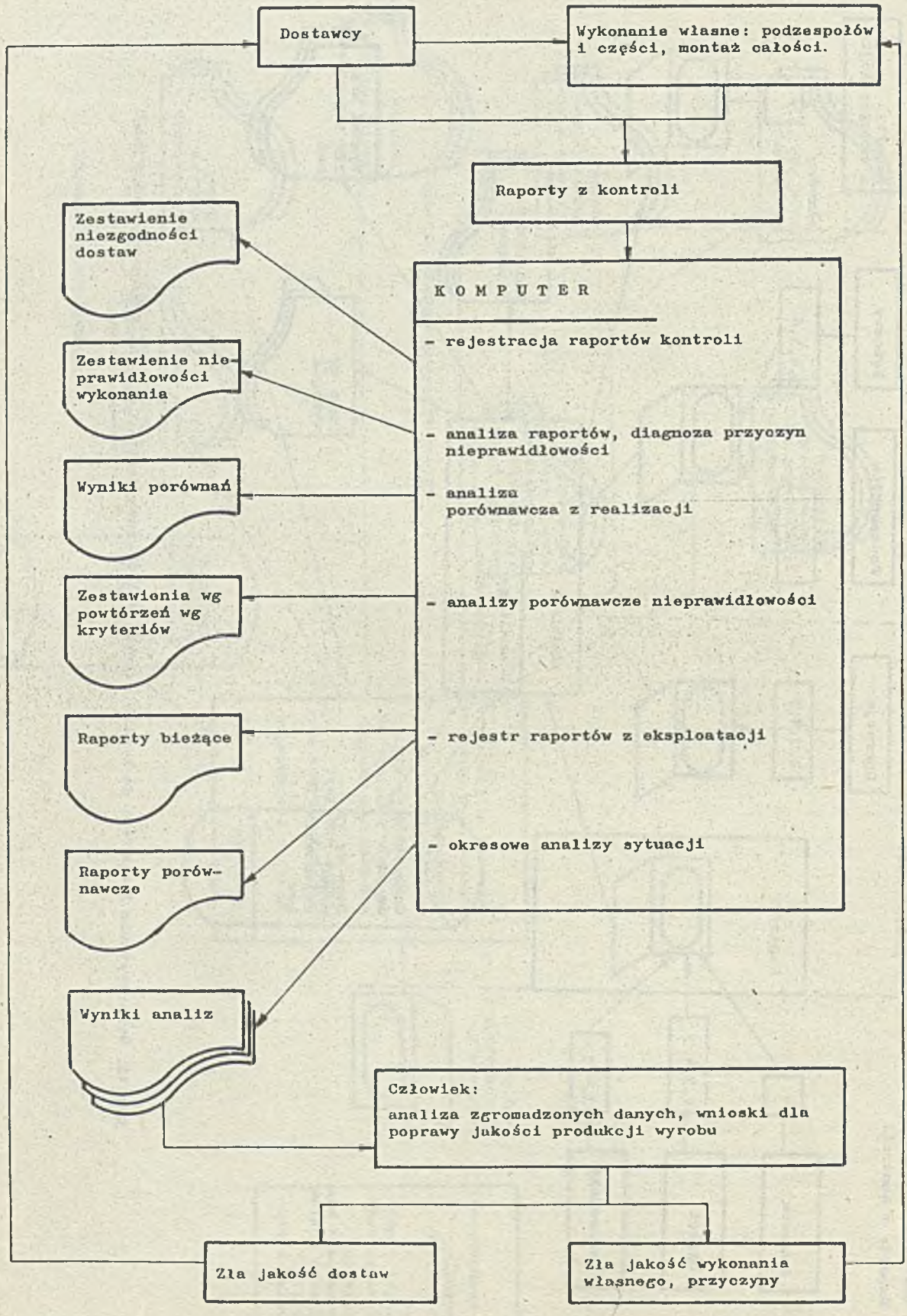
Przede wszystkim należy tu wymienić wszystkie te zastosowania, gdzie komputer będzie służył zbieraniu wyników pomiarów i badań kontrolnych, szybkiemu analizowaniu tych danych; ich odpowiedniemu agregowaniu i wstępnemu przygotowywaniu decyzji kierownictwa odpowiedniego szczebla. Należy więc uzupełnić przedstawione na rys. 8, 9 i 10 schematy, tak, aby powstał skomputeryzowany system obiegu informacji o wynikach kontroli jakości. Uzupełnienia te zilustrowano na rys. 11, 12, 13. Dopiero wówczas można będzie uznać przedstawiony na rys. 7 model serwomechanizmu sterowania jakością wytwarzania wyrobu jako prawdziwy.



Rys. 11. Schemat przepływu informacji potrzebnych do działania korygującego z wykorzystaniem komputera

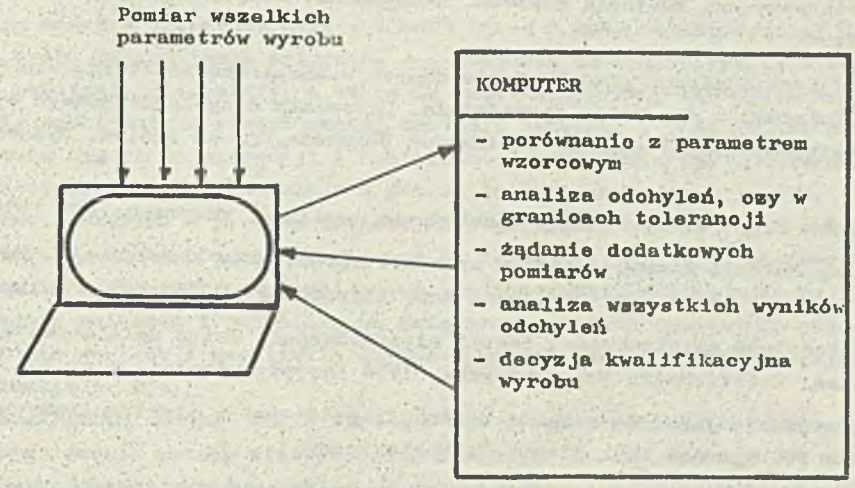


Rys. 12. Wykorzystanie komputera do kontroli jakości i niezawodności w całości procesu produkcyjnego



Rys.13. Wykorzystanie komputera do analiz z bieżącej kontroli jakości

Można też widzieć jeszcze inną rolę komputerów w kontroli jakości. Jak powiedziano wcześniej, w fazie II realizuje się jakość wykonania. A zatem dysponując dokumentacją techniczną i prototypem wyrobu przebadanym w I fazie realizacji wyrobu - porównuje się zgodność konkretnego wyrobu z odpowiednimi kryteriami wzorcowymi - komputer w tym wypadku byłby również zaangażowany bezpośrednio do sterowania samym badaniem zgodności wyrobu z wzorcem (rys. 14).



Rys. 14. Komputer w systemie badania jakości wyrobu

Wybór konkretnej metody kontroli jakości zależy od rodzaju produkcji, wielkości produkcji, ustalenia liczby i rodzaju parametrów (cech) kontrolowanych. Badane parametry muszą być określone jednoznacznie.

Wspomaganie komputerowe może tu polegać na:

- analizie ilościowej,
- badaniach porównawczych modelu ze stanem faktycznym,

automatyzacji procesu [5] przejawiającej się:

- w sterowaniu procesem, w którym przyrządy wskazujące, rejestrujące, liczące i sterujące dopomagają regulowaniu przez człowieka lub zapewniają samoregulację,
- jako kontrolę produktu, przy której urządzenia automatyczne sprawdzają i segregują produkt oraz podają wyniki kontroli.

Wykorzystując dane z literatury i pamiętając o tym, że jakość wyrobu realizuje się we wszystkich trzech fazach powstawania tego wyrobu, należy podkreślić, że zagadnienie jakości musi być rozwiązywane kompleksowo. Zagadnienie jakości zakład musi uwzględnić już na etapie planowania produkcji. Troska o zapewnienie jakości wymaga też takiego przygotowania zakładu, aby m.in. zostało zapewnione dotrzymanie wzorców jakości. Natomiast sens zastosowania komputerów do systemu kontroli jakości ujawnia się dopiero po osiągnięciu pewnego podstawowego poziomu organizacyjnego, tzn. wtedy, gdy nawet drobne odchylenia od wzorców będą się silnie odbijały na końcowej ocenie wyrobu, oraz gdy będzie istniała potrzeba szybkiej reakcji odpowiednich szczebli kierowania na wszelkie wykryte usterki.

Literatura

[1] Bonkowicz-Sittauer S., Konopka T.: Propozycja pakietu programów dla oceny eksploatacyjnej niezawodności sprzętu komputerowego.

[2] Dziewulak A.: Wybrane problemy kontroli jakości produkcji. Raport z badań sondażowych Zakładu Badań Społecznych Instytutu Organizacji Przemysłu Maszynowego. Warszawa 1978

[3] Friling T.: Statystyczna kontrola jakości. Rozpoznanie metod. Warszawa:IMM 1979. Archiwum Opracowań nr 72

[4] Janiszewski J.: Elektroniczne przetwarzanie danych w kompleksowym sterowaniu jakością oraz wykorzystanie informacji dla celów zarządzania. Materiały z ogólnokrajowego sympozjum naukowego Jakość-Producent-Użytkownik. Seminarium Branżowe, grupa ogólna. Bydgoszcz: TNOIK 1972

[5] Juran J.M., Gryna F.M. jr: Jakość - projektowanie, analiza. Warszawa: WNT 1974

[6] Kiliński A.: Zagadnienia niezawodności w procesie opracowania konstrukcji. Poradnik 1. Warszawa: 1971. JOPM, Ośrodek Doskonalenia Kadr Kierowniczych Przemysłu Maszynowego.

[7] Migdalski J.: Podstawy strukturalnej teorii niezawodności. Wstęp do niezawodności systemów ogólnych. Kielce: Politechnika Świętokrzyska. 1978 Skrypty uczelniane nr 57

[8] Okoński M.: Sterowanie jakością w fazie produkcji podzespołów elektronicznych biernych. Praca doktorska PW. Wydział Mechaniczny Technologiczny

[9] Praca zbiorowa - System badań niezawodności i trwałości maszyn włókienniczych w warunkach przemysłowych z zastosowaniem ETO. Łódź: Centr. OBR Maszyn Włókienniczych POLMATEX-CENARO, 1977

[10] Tomaszewski A.: - Sterowanie jakością produkcji maszyn i urządzeń. WNT, Warszawa, 1977

[11] Wasilowski L.: Metody statystyczne kontroli jakości w procesach. TNOIK 1978. Biblioteczka poradników jakości. Materiały szkoleniowe, zeszyt 10. Bydgoszcz

[12] Wasilewski L.: Podstawowe pojęcia kompleksowego systemu sterowania jakością. Bydgoszcz: TNOIK 1977. Biblioteczka poradników jakości. Materiały szkoleniowe, zeszyt 1.

[13] Teresa Wolańska: Statystyczna kontrola jakości. Uwagi ogólne. Biuletyn Informacyjny OSK 1979 nr 5/6.

mgr Teresa Wolańska
Instytut Maszyn Matematycznych

Statystyczna kontrola jakości. Uwagi ogólne

WSTĘP

Artykuł ten ma za zadanie przedstawienie ogólnych zasad i metod statystycznej kontroli jakości produkcji; jest próbą uzasadnienia celowości zastosowania maszyn cyfrowych w tej dziedzinie. Powstał głównie na podstawie literatury, a także na podstawie rozmów przeprowadzanych w ośrodkach zajmujących się problematyką statystycznej kontroli jakości[x/]. Szczególnie prowadzone rozmowy pozwoliły upewnić się co do celowości i konieczności stworzenia oprogramowania wypomagającego statystyczną kontrolę jakości produkcji w przedsiębiorstwach przemysłowych.

Wszystkie czynności związane z przeprowadzaniem statystycznej kontroli jakości wymagają wielu często żmudnych obliczeń, które obecnie wykonywane są "ręcznie", np. przy stosowaniu tej samej metody kontroli zmieniają się tylko dane wejściowe. Zastosowanie maszyny cyfrowej znacznie przyspieszyłoby wykonywanie obliczeń i pozwoliło na zwiększenie liczby wykonanych pomiarów, a więc zwiększyłoby aktualność wyników i umożliwiło szybsze podejmowanie decyzji korygujących proces produkcyjny i eliminację braków.

Równocześnie jednak należy zdawać sobie sprawę z małej popularności lub niedoceniania roli, jaką w przemyśle mogą odegrać metody statystycznej kontroli jakości. W praktyce można się też spotkać ze zbyt małą troską o jakość w zakładach produkcyjnych. Na podstawie pracy [6] można wyciągnąć wniosek, że w wielu zakładach większy nacisk kładzie się na ilość, a nie na jakość, że realizuje się plany wg zasady: "ilość za wszelką cenę".

Ponadto często zdarza się, jak to wynikało z wielu rozmów, że wykorzystywanie w przedsiębiorstwach metody kontroli jakości mają podstawy wyłącznie "intuicyjne" i często są w sprzeczności z zasadami wnioskowania statystycznego.

W praktyce, jak to opisywano m.in. w Instytucie Badań Systemowych PAN, statystyczna kontrola w czasie produkcji wygląda przeważnie tak, że na danym stanowisku pracy wisi tabelka z podanymi licznościami partii i odpowiadającymi jej licznościami próbek oraz dopuszczalną i dyskwalifikującą liczbę sztuk wadliwych. Taką procedurę prowadzi się od szeregu lat i użytkownikowi wydaje się ona być dostateczna i nie wymagająca żadnych zmian.

Z przeprowadzonych rozmów wynika, że w zasadzie trudno jest dociec, jakimi kryteriami posługują się służby kontroli jakości przy wyborze odpowiedniej metody statystycznej. Na ogół są to nie zawsze najszczęśliwiej wybrane plany z normy PN-74/N-03021, dotyczącej kontroli odbiorczej według oceny alternatywnej. Nie stosuje się w takich wypadkach badań porównawczych różnych planów, a możnaby to zrobić wykorzystując różne kryteria, takie jak np. przebieg krzywej OC lub oczekiwana liczba kontrolowanych sztuk. W praktyce nie stosuje się planów badania cechy mierzalnej, które umożliwiają uzyskanie więcej informacji na temat przebiegającego procesu produkcyjnego. Jak wynikało z rozmów, w dwóch zakładach produkcyjnych trudności w stosowaniu tych planów polegają na braku pracowników, którzy mogliby dokonywać stosowanych pomiarów. Z tych samych źródeł wiadomo również, że metody statystycznej kontroli jakości stosowane są często tylko w celach sprawozdawczych i z takich czy innych powodów nie są wykorzystywane do poprawy produkcji.

Nie należy przypuszczać, aby zastosowanie komputerów do SKJ stanowić mogło radykalne rozwiązanie wszystkich problemów, z których parę opisano powyżej. Należy jednak mieć nadzieję, że dzięki temu można będzie spopularyzować niektóre z nich, a inne, takie jak plany odbiorów ciągłych podać w formie umożliwiającej prostę stosowanie.

---

x/ Były to następujące ośrodki: Instytut Organizacji Przemysłu Maszynowego, Ośrodek Badania i Kontroli Jakości Wyrobów Przemysłu Maszynowego, Instytut Badań Systemowych PAN, Centralny Ośrodek Badawczy Normalizacji, Zakłady Radiowe im. M.Kasprzaka, Zakłady Wytwórcze Urządzeń Telefonicznych ZWUT.

Pojęcie jakości i jej ocena

W zasadzie nie istnieje jedna, obowiązująca definicja pojęcia "jakość". Każdy autor korzysta z takiej, która najlepiej pasuje do omawianego przez niego zagadnienia. Wasilewski [5] jako "jakość produktu" rozumie "stopień spełnienia przez wyrób funkcji wymaganych przez użytkownika". Stopień ten jest zmienną stochastyczną zależną od:

- środowiska eksploatacji,
- charakterystyki technicznej wyrobu, czyli cech wynikających z konstrukcji i wykonania.

Ostatnio proponuje się, w celu uniknięcia wieloznaczności, słowo "stopień" zastąpić słowem "prawdopodobieństwo". Jakość wyrobu dzieli się zwykle na dwie składowe: jakość typu i jakość wykonania.

Jakość typu określa wynikające z dokumentacji prawdopodobieństwo zaspokojenia przez wyrób określonych potrzeb odbiorców. Jakość typu zależy w zasadzie od czynności sfery pozaprodukcyjnej, tj. sfery przedprodukcyjnej (projektowanie i przygotowanie procesu produkcji) oraz poprodukcyjnej (użytkowanie).

Jakość wykonania określa stopień zgodności określonej partii bądź serii produkcyjnej z wymaganiami dokumentacji. Jakość wykonania zależy zasadniczo od czynności sfery produkcyjnej, z tym, że przyczyny zakłócające proces technologiczny są w głównej mierze skutkami określonych działań procesu przygotowania produkcji [5].

W celu kontrolowania jakości wytwarzanego produktu, należy przyjąć określony sposób "pomiaru" tej jakości. Można to zrobić, zgodnie z przytoczoną na początku definicją, obliczając wartość prawdopodobieństwa. Można też skorzystać z innej definicji, w myśl której jakość jest to "zespół cech wyrobu decydujących o spełnieniu przezeń określonych funkcji" [5].

Aby móc stwierdzić, czy poziom jakości, często oceniany na podstawie wielu kryteriów i cech jest odpowiedni oraz, aby móc porównywać ze sobą jakość wielu wyrobów opracowano wiele metod oceny jakości. Często wiążą się one z konkretnym rodzajem produkcji, a także z przyjętą definicją jakości. Większość istniejących metod oceny jakości związanych z drugą definicją, polega na ocenianiu cech wyrobu przez grupy ekspertów. Metody te noszą nazwę "metod ekspertów" [5].

Otrzymany takimi metodami wynik podawany jest w jednostkach umownych w stosunku do oceny jakości wyrobu przyjętego za wzorzec. Utrudnia to operowanie miernikiem jakości w powiązaniu z innymi danymi ekonomicznymi mającymi swój rzeczowy lub wartościowy wyraz. Ponieważ o jakości środków produkcji decydują między innymi także efekty osiągane przez użytkownika w wyniku zainstalowania ocenianego urządzenia produkcyjnego oraz nakłady związane z jego użytkowaniem, rolę miernika jakości może spełniać stosunek wydajności rzeczywistej urządzenia (wyrażającej wartość pracy wykonanej w jednostce czasu) do wartości nakładów przypadających na jednostkę czasu [2].

Ten rodzaj oceny jest dogodny z punktu widzenia badań porównawczych poziomów jakości urządzeń spełniających podobne funkcje.

Statystyczne badanie wyrobów

Współczesną technikę charakteryzuje masowe wytwarzanie wielu wyrobów i wzmożone zainteresowanie ich jakością. Zarówno w produkcji, jak i poprzedzających ją pracach doświadczalnych wnioskuje się o właściwościach (cechach) wyrobów na podstawie odpowiednich badań. Przedmiotem badania może być jedna sztuka albo też określony zbiór wyrobów uznanych a priori za jednakowe, np. partia wyrobów jednego typu wykonanych w danej wytwórni. Przed przeprowadzeniem badań zakłada się, że wszystkie sztuki są badane w taki sam sposób, oraz że istnieje niezmierzony rozrzut wartości badanej cechy dla poszczególnych sztuk w jednym zbiorze wyrobów.

Uzyskanie dokładnych informacji o rozkładzie wartości danej cechy w rozpatrywanym zbiorze wyrobów wymaga zbadania wszystkich sztuk należących do tego zbioru, tzn. badania stuprocentowego, co w praktyce bywa trudne, a nieraz wręcz niemożliwe do zrealizowania. Dlatego zazwyczaj nie bada się całego zbioru wyrobów, a tylko jego część, nazywaną próbką reprezentacyjną [1]. Na podstawie wyników badania próbki reprezentacyjnej wyciąga się wnioski dotyczące określonych cech całego zbioru. Badania takiej próbki zamiast całego reprezentowanego przez nią zbioru, nazywa się badaniem statystycznym. W celu lepszej reprezentatywności próbki poszczególne sztuki pobiera się ze zbioru w sposób losowy. Otrzymaną w ten sposób próbkę nazywa się próbką losową [1]. Jej liczność jest na ogół dużo mniejsza niż liczność zbioru, z którego została wylosowana.

Technika losowania powinna gwarantować każdemu elementowi zbioru jednakowe prawdopodobieństwo dostania się do próbki. Próbka, której elementy spełniają ten warunek nazywana jest próbką losową prostą. W celu uzyskania takiej próbki stosuje się losowanie ze zwracaniem, które w praktyce, jeżeli liczność próbki n jest znacznie mniejsza niż liczność zbioru N (tzn. $n < 0.1 \times N$), zastępuje się mniej kłopotliwym losowaniem bez zwracania, uważając otrzymaną próbkę losową za próbkę losową prostą [1].

Całokształt badania statystycznego ustala się w jednym dokumencie, nazywanym programem badań (planem, procedurą, algorytmem). Zawiera on:

- określenie badanego zbioru wyrobów,
- przepis badania, określający liczność próbki losowej i sposób jej pobrania, określenie cech wyrobów oraz sposobów, w jaki mają być badane,
- przepis analizy wyników badań, określający wskaźniki charakteryzujące rozważany zbiór wyrobów ze względu na badane cechy oraz wnioskowania o wartościach tych wskaźników na podstawie wyników badania poszczególnych sztuk w próbce losowej.

Wynik badania próbki losowej wskutek swojej losowności, może różnić się od wyniku ewentualnego badania całego zbioru wyrobów. Z tego względu należy zachować ostrożność przy formułowaniu wniosków z przeprowadzonych badań statystycznych.

Przy okazji warto nadmienić, że mylny jest pogląd o stuprocentowej pewności wykrycia wszystkich braków przy stuprocentowej kontroli. Na podstawie wielu doświadczeń udowodniono, że nawet parokrotnie przeprowadzona stuprocentowa kontrola tej samej partii towaru nie była w stanie wykryć wszystkich braków [4]. Związane jest to naturalnie z uciążliwością takiej kontroli i zmęczeniem kontrolerów, czyli zawodnością człowieka.

ZASADY WNIOSKOWANIA STATYSTYCZNEGO

Metody statystycznej kontroli jakości można dzielić na podstawie różnych kryteriów. Zasadniczego podziału dokonuje się na metody kontroli bieżącej i odbiorczej. Pierwsze z nich dotyczą fazy trwania produkcji, drugie stosuje się po zakończeniu tej fazy.

Inne kryterium dotyczy rodzaju kontrolowanej cechy, która może być mierzalna lub niemierzalna.

Zarówno dla cech mierzalnych, jak i dla cech niemierzalnych można mówić o wartościach danej cechy dla poszczególnych sztuk wyrobów i o zbiorze możliwych wartości tej cechy. Cechę pojętą w tym ostatnim sensie, będzie się dalej oznaczać przez X, a jej konkretne wartości przez x. W wypadku cechy niemierzalnej, zbiór możliwych jej wartości jest dwuelementowym zbiorem liczb $\{0,1\}$, w wypadku cechy mierzalnej skokowej - przeliczalnym zbiorem liczb $\{x_1, x_2, \ldots, x_m \ldots\}$, a w wypadku cechy mierzalnej ciągłej - określonym przedziałem liczbowym $[a,b]$.

Dla oceny danej produkcji należy dokonać pomiarów odpowiednich parametrów konkretnych wyrobów, które znalazły się w wylosowanych próbkach oraz przeprowadzić wnioskowanie zgodnie z zasadami wnioskowania statystycznego. Stawiana hipoteza dotyczy zagadnienia, czy przedstawiona do kontroli partia została wytworzona "dobrze", to jest zgodnie z pewnymi przyjętymi kryteriami, czy "źle", to jest nie zgodnie. Aby ocenić tę zgodność, należy badać określone parametry, zwane cechami wylosowanych przedmiotów. Ponieważ wartości cechy zależą od elementów, które w losowy sposób znalazły się w próbie, są one również losowe. Można więc tak rozumianą cechę traktować jako zmienną losową, oznaczaną dalej X. Rozkład cechy X oznaczony P_θ nie jest znany i zależny od pewnego parametru rozkładu $\theta = (\theta_1, \ldots, \theta_m)$, którego wartość należy wyestymować na podstawie wyników badania wylosowanej próby. Każde przypuszczenie dotyczące nieznanego rozkładu zmiennej losowej nazywamy hipotezą statystyczną. Hipoteza statystyczna określająca jedynie wartości nieznanych parametrów liczbowych zmiennej losowej nosi nazwę hipotezy parametrycznej i może mieć postać np.: $H_o(\theta = \theta_o)$. Metody służące do sprawdzania, czy też, jak się często mówi, do zweryfikowania wysuniętej hipotezy H_o, noszą nazwę testów statystycznych. Test służący do zweryfikowania hipotezy parametrycznej nazywany jest testem parametrycznym. Jeżeli wysuwa się tylko jedną hipotezę i jedynym celem testu statystycznego jest zweryfikowanie, czy ta hipoteza nie jest fałszywa i nie bada się jednocześnie innych hipotez, to taki test nazywany jest testem istotności.

W wypadku statystycznej kontroli jakości mamy do dyspozycji pewną n-elementową próbkę $(x_1, \ldots, x_n)$, gdzie x_i reprezentuje i-ty pomiar cechy.

Niech weryfikowana hipoteza ma postać $H_o(\theta = \theta_o)$. Rozpatrzmy statystykę[x] $Z_n = \varphi(X_1,\ldots,X_n)$ taką, że jeżeli hipoteza H_o jest prawdziwa, to zaobserwowana wartość statystyki Z_n

$$z_n = \varphi(x_1,\ldots,x_n)$$

powinna z prawdopodobieństwem $1-\alpha$ zawierać się w przedziale (z', z''), czyli:

$$P(z' < Z_n < z'' \mid \theta = \theta_o) = 1-\alpha \qquad (1)$$

gdzie: z' i z'' - wartości statystyki Z_n wybierane z tablic rozkładów zmiennych losowych,
$1-\alpha$ - tzw. poziom ufności testu.

Wyrażenie po lewej stronie (1) oznacza prawdopodobieństwo tego, że w wypadku słuszności hipotezy H_o, to znaczy, gdy rzeczywiście $\theta = \theta_o$, wartość statystyki Z_n będzie większa od z' i mniejsza od z''. α, tak zwany poziom istotności testu jest pewnym krytycznym prawdopodobieństwem, którego wartość przyjmuje się na ogół albo 0.05 albo 0.01.

Przy takich założeniach stosuje się poniższy test.

Jeżeli zaobserwowana wartość $z_n = \varphi(x_1,\ldots,x_n)$ spełnia nierówność $z_n \leq z'$ lub $z_n \geq z''$, to hipotezę $H_o(\theta = \theta_o)$ odrzuca się. Jeżeli zaś $z' < z_n < z''$, to mówi się, że próba nie doprowadza do odrzucenia hipotezy H_o. Wtedy poziom istotności testu α wyrażony wzorem

$$P(Z_n \leq z' \mid \theta = \theta_o) + P(Z_n \geq z'' \mid \theta = \theta_o) = \alpha \qquad (2)$$

jest prawdopodobieństwem odrzucenia hipotezy prawdziwej. Jest to tak zwany błąd I rodzaju.

Można też popełnić inny błąd, mianowicie przyjąć hipotezę H_o wtedy, gdy w rzeczywistości jest ona nieprawdziwa. Jest to błąd II rodzaju. Oznaczając prawdopodobieństwo popełnienia takiego błędu przez β zachodzi:

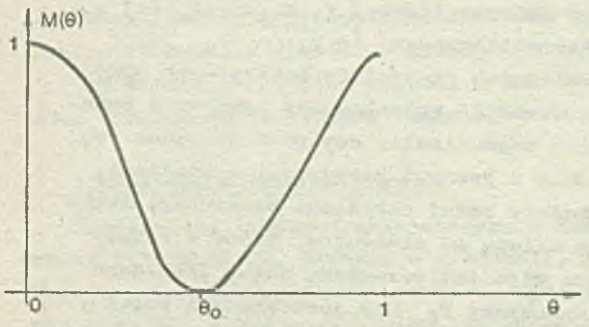
$$P(Z_n \leq z' \mid \theta \neq \theta_o) + P(Z_n \geq z'' \mid \theta \neq \theta_o) = \beta \qquad (3)$$

Powstawanie błędów pierwszego i drugiego rodzaju spowodowane jest przez to, że wszelkie podejmowane na podstawie przeprowadzonego testu statystycznego decyzje podejmuje się z pewnym, różnym od jedności, prawdopodobieństwem. Nawet wtedy, gdy hipoteza H_o jest prawdziwa, z prawdopodobieństwem α (wzór(2)) może zajść sytuacja odrzucenia hipotezy na podstawie wyników badania wylosowanej próby.

Aby zdobyć więcej informacji o teście, niż na to pozwala poziom istotności α, konstruuje się funkcję mocy testu (rys.1). Jest to prawdopodobieństwo odrzucenia badanej hipotezy $H_o(\theta = \theta_o)$ traktowane jako funkcja argumentu θ.

$$M(\theta) = P(\text{odrzucenia } H(\theta = \theta_o) \mid \theta) \qquad (4)$$

Funkcja mocy testu powinna gwarantować duże prawdopodobieństwo odrzucenia hipotezy fałszywej (najlepiej jak najbliższe jedynce) oraz małe prawdopodobieństwo odrzucenia hipotezy prawdziwej (najlepiej jak najbliższe zera). Innymi słowy, test jest "dobry", gdy ma mały poziom istotności α i często prowadzi do odrzucenia hipotezy $H_o(\theta = \theta_o)$, gdy $\theta \neq \theta_o$.



Rys. 1. Przebieg funkcji mocy testu $H_o(\theta = \theta_o)$, $0 < \theta_o \leq 1$

W ścisłym związku z pojęciem mocy testu pozostaje funkcja operacyjno-charakterystyczna, oznaczana dalej przez OC i określana wzorem:

$$OC(\theta) = 1 - M(\theta) \qquad (5)$$

Jest to więc prawdopodobieństwo przyjęcia hipotezy wyrażone jako funkcja argumentu θ.

ODBIORCZA KONTROLA JAKOŚCI

Jednym z ważnych rodzajów czynności wykonywanych w ramach kontroli jakości jest odbiór jakoś-

x) to jest pewną funkcję, której argumentami są zmienne losowe

ciowy wyrobów lub partii wyrobów. Przez "odbiór" rozumie się dokonywanie klasyfikacji wyrobów na przyjęte lub nie przyjęte przez odbierającego. Odbiór taki jest więc biernym rodzajem kontroli jakości, nie może bowiem oddziaływać na wadliwość przedstawionej do odbioru partii towaru przez zmiany w sposobie wytwarzania.

Skuteczność odbioru określa się przez:

- prawdopodobieństwo α popełnienia błędu pierwszego rodzaju, tj. błędu polegającego na odrzuceniu wyrobów lub partii odpowiadających wymaganiom (ryzyko dostawcy),
- prawdopodobieństwo β popełnienia błędu drugiego rodzaju, tj. błędu polegającego na przyjęciu wyrobów lub partii nie odpowiadających wymaganiom (ryzyko odbiorcy).

Stosowanie kontroli wyrywkowej jest więc związane z pewnym ryzykiem, którego wysokość może być jednak z góry świadomie ustalana przez dobór odpowiednich warunków kontroli.

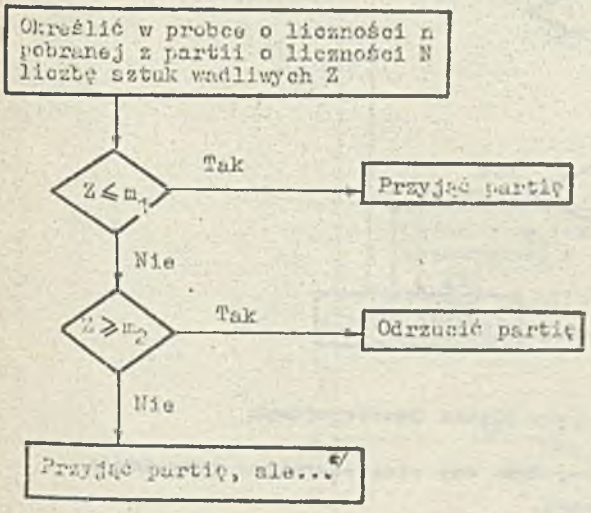
Jak już wspomniano, kontrola stuprocentowa nie zawsze jest możliwa do przeprowadzenia. Istotne trudności w jej stosowaniu, a zatem sytuacje, w których zaleca się stosowanie odbiorów statystycznych występują przy odbiorach:

- dużych partii wyrobów, wobec dużej pracochłonności i związanego z tym zmęczenia kontrolerów,
- produktów sypkich lub ciągłych, wobec niemożności wykonania kontroli całej partii,
- pojedynczych wyrobów, gdy liczba parametrów wymagających kontroli jest bardzo duża,
- dowolnych partii, gdy badania mają charakter niszczący.

Plan odbioru, tak jak wszystkie plany kontroli jakości, dzieli się na plany odbioru według oceny alternatywnej i plany odbioru według właściwości liczbowych.

Plany według oceny alternatywnej dzielą się na plany:

- jednostopniowe (rys.2),
- dwustopniowe (rys. 3),
- wielostopniowe (rys. 4),
- sekwencyjne.



Rys. 2. Schemat procedury odbioru w planie jednostopniowym

Innym kryterium podziału metod planów odbioru według oceny alternatywnej jest sposób, w jaki otrzymywane są gotowe wyroby, tzn. czy są one dzielone na partie, czy też proces produkcyjny traktuje się jako proces ciągły. Wyżej przytoczony podział dotyczy produkcji dzielonej na partie. Metody kontroli jakości produkcji ciągłej, jako szczególnie interesujące z punktu widzenia "komputeryzacji", będą omówione w osobnym rozdziale.

Plany odbioru według oceny własności liczbowych dzielą się na:

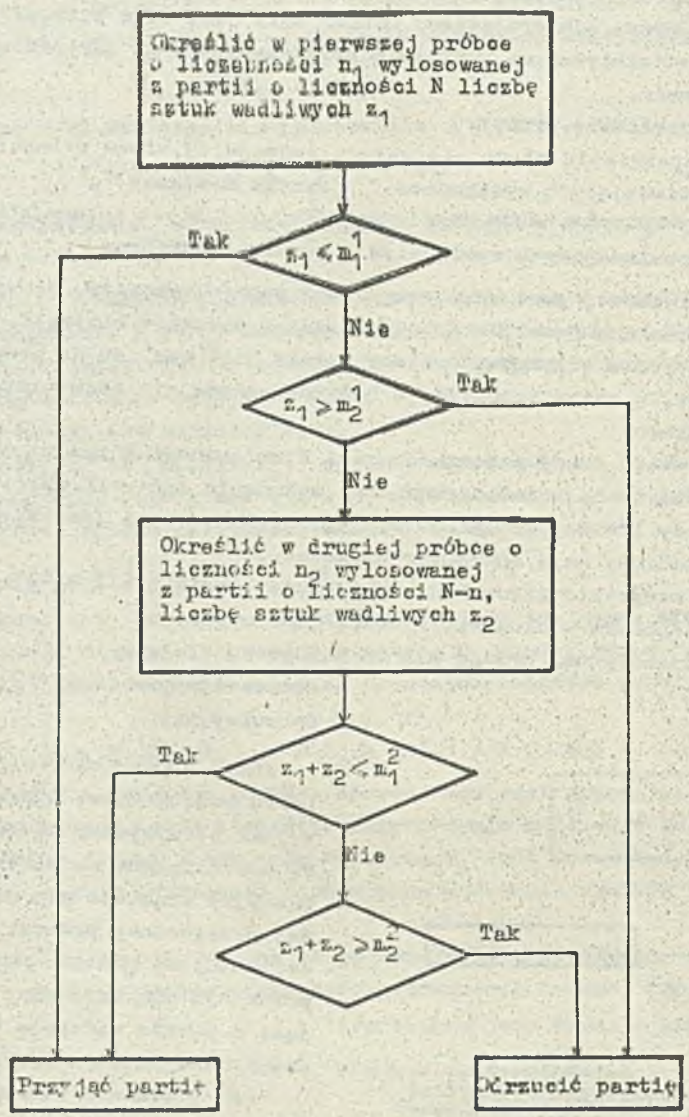
- plany dla cechy ograniczonej jednostronnie przy znanym lub nieznanym odchyleniu średnim,
- plany dla cechy ograniczonej dwustronnie przy znanym lub nieznanym odchyleniu średnim.

Plany odbioru według oceny alternatywnej

Szczegółowe przepisy dotyczące stosowania alternatywnych planów badaniu jakości zawarte są w normie PN-73/N-03021. Dotyczy ona planów jedno-, dwu- i wielostopniowych.

Pierwszym zagadnieniem kontroli wyrywkowej jest ustalenie wielkości próbki w zależności między innymi od liczności partii. Zależność ta jest określona przez tzw. poziom kontroli, tj. stosunek liczności próbki do liczności partii. W obowiązującej normie przyjęto trzy ogólne i cztery specjalne poziomy kontroli. Norma ta zawiera specyfikację oraz zalecenia dotyczące stosowania poszczególnych poziomów.

x) Sytuacja, że $m_1 < z < m_2$ może zajść tylko w kontroli ulgowej (ponieważ wtedy $m_2 = m_1+1$), należy wtedy tę partię przyjąć, ale już do kontroli następnej partii zastosować plany kontroli normalnej (dotyczy to również innych planów wielostopniowych)



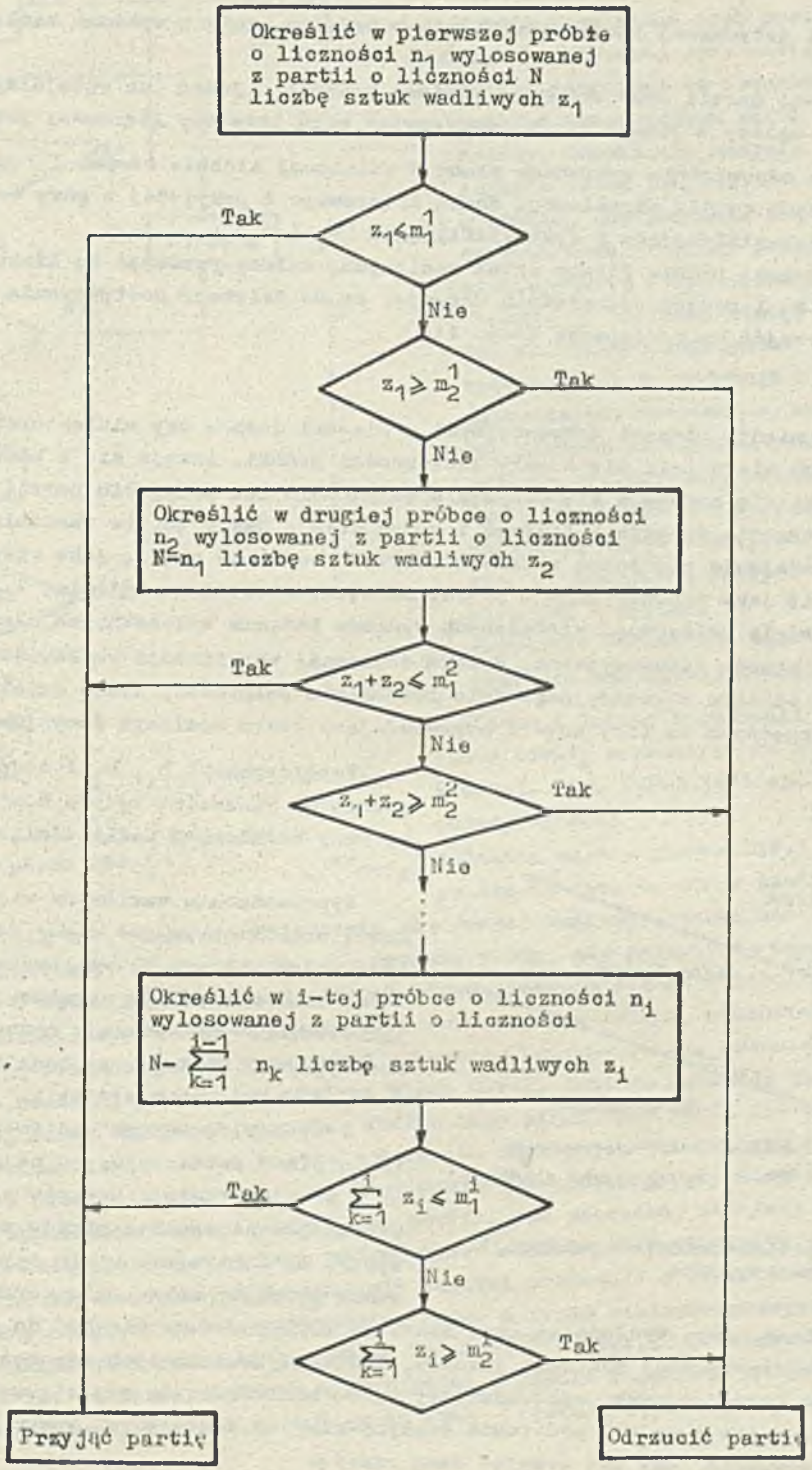
Rys. 3. Schemat procedury odbioru przy planie dwustopniowym

Dla każdego z planów badania, czy to będzie plan jedno-, dwu- czy wielostopniowy przewiduje się trzy rodzaje kontroli, tj. normalną, obostrzoną i ulgową.

Różnią się one między sobą przyjmowanymi wielkościami: liczbą kwalifikującą m_1 i liczbą dyskwalifikującą m_2. Przy kontroli obostrzonej liczba dyskwalifikująca jest mniejsza niż przy innych rodzajach kontroli. Kontrola normalna stanowi zasadniczy rodzaj kontroli i jest zawsze stosowana do sprawdzania pierwszych partii danego wyrobu. Cytowana wyżej norma zawiera dokładny opis algorytmu przechodzenia z jednego rodzaju kontroli na drugą.

Po to, żeby na podstawie normy PN-73/N-03021 wybrać odpowiedni plan badania należy ustalić najpierw, czy ma to być plan jedno-, dwu-, czy wielo- (maksymalnie siedmio-) stopniowy. W myśl zaleceń normy, plany jednostopniowe charakteryzujące się większą licznością próbki należy stosować wtedy, gdy koszt badania sztuki produktu jest niewielki i czas badania jest zbyt długi, aby stosować plany wielostopniowe. Plany wielostopniowe charakteryzują się najmniejszą oczekiwaną liczbą badanych sztuk (w porównaniu z innymi planami więcej trzeba sztuk pobrać, ale mniej skontrolować) i dlatego norma zaleca stosować je wówczas, gdy czas potrzebny do pobrania i zbadania jednej sztuki produktu jest krótki, a koszt badania jest duży.

Przy wyborze liczby stopni planu badania decydującą rolę odgrywa także znana lub przewidywana wadliwość procesu produkcyjnego. I tak na przykład, z tego powodu, że pierwsza próbka w planie dwustopniowym jest mniej liczna niż w planie jednostopniowym, stosunek liczb sztuk badanych przy



Rys.4. Schemat odbioru przy planie wielostopniowym

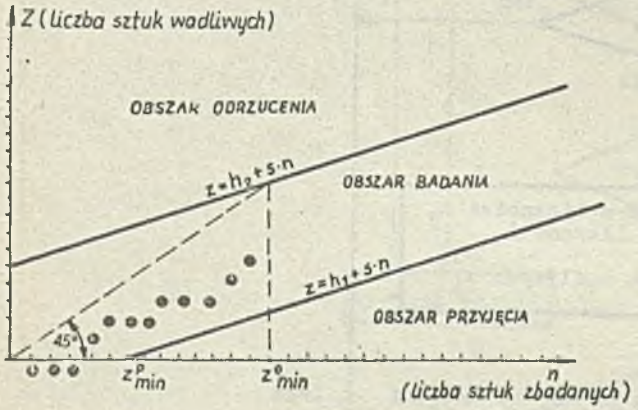
kontroli próbek w tych dwóch planach zależy od jakości przedstawionego do odbioru produktu. Jeżeli jakość ta jest tak dobra, że do drugiego stopnia badania trzeba brać tylko niewiele próbek, to liczba operacji kontrolnych będzie wyraźnie mniejsza przy planie dwustopniowym. Cytowani przez Granta [3] Dodge i Romig podają w jednej ze swoich prac wykres porównawczy kontroli według planów jednostopniowego i dwustopniowego dla różnych liczności partii i różnych stosunków średniej wadliwości w procesie - do wadliwości dyskwalifikującej. Stwierdzają oni, że według tablic najczęściej stosowanych w praktyce, oszczędność w nakładzie pracy na kontrolę w wyniku stosowania planów dwustopniowych wynosi zwykle ponad 10%, a może sięgać aż do 50%.

Po podjęciu decyzji dotyczącej liczby stopni planu badania, należy wykonać następujące czynności:

- na podstawie liczności partii oraz obranego poziomu kontroli (jeden ze specjalnych, czy ogólnych poziomów) należy z tabeli w normie odczytać znak literowy liczności próbki,
- znaleźć w normie dla odpowiednio wybranego planu o ustalonej liczbie stopni i rodzaju kontroli tabelę, w której podane są dla określonego znaku literowego i przyjętej z góry wadliwości dopuszczalnej wartości kwalifikująca i dyskwalifikująca (m_1 i m_2);
- po zbadaniu w wylosowanej próbie liczby sztuk wadliwych, należy porównać tę liczbę z odczytanymi wartościami m_1 i m_2 i podjąć odpowiednią decyzję, co do dalszego postępowania (przyjąć partię, odrzucić czy przejść do następnego etapu itp).

Plany sekwencyjne

Zasadnicza różnica między planami sekwencyjnymi a planami jedno- czy wielostopniowymi polega na tym, że w pierwszych nie ustala się z góry liczebności próbki. Losuje się i bada kolejne elementy tak długo, dopóki nie podejmie się decyzji o odrzuceniu lub przyjęciu partii, ponieważ jest jeszcze możliwa trzecia decyzja - losować i badać dalej. Tak więc, po zbadaniu kolejnego wylosowanego elementu podejmuje się jedną z trzech decyzji: odrzucić partię jako niezgodną z wymaganiami, przyjąć partię jako zgodną, albo - pobrać następny element do badania. I to, że decyzję podejmuje się na podstawie kolejności otrzymanych wyników badania wylosowanych elementów jest cechą charakterystyczną planów sekwencyjnych. W celu dokonania weryfikacji wadliwości produkcji, kreśli się na arkuszu odbioru sekwencyjnego dwie równoległe półproste, które dzielą obszar I ćwiartki układu współrzędnych na trzy części odpowiadające trzem możliwym decyzjom.



Rys.5. Graficzne przedstawienie realizacji badania sekwencyjnego

Współczynniki h_1, h_2 i s dobiera się stosownie do wielkości ryzyka dostawcy i odbiorcy przy ustalonych wadliwościach AQL[x/] i LTPD[xx/].

Wyprowadzenie wzorów na wszystkie wielkości charakteryzujące plany sekwencyjne znaleźć można w pracy K.Wasilewskiego [5].

Porównanie planów sekwencyjnych z planami jednostopniowymi wykazuje przewagę planów sekwencyjnych. Stosując je bada się przeważnie, średnio mniej sztuk z partii.

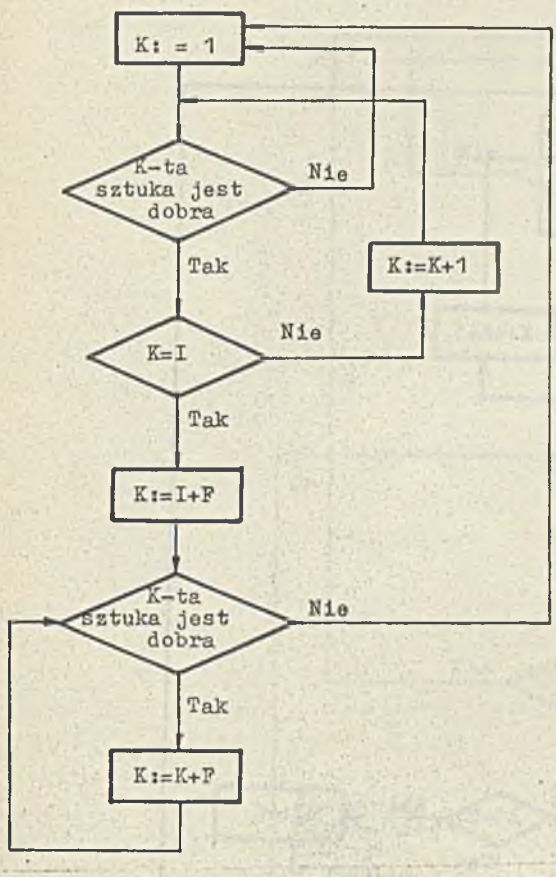
W sytuacji niskich wadliwości przed kontrolą plany sekwencyjne są najefektywniejszymi z punktu widzenia kosztów planami odbioru [5]. W praktyce stosowanie planów sekwencyjnych wymaga od kontrolera wielu czynności: narysowania wykresu, obliczenia parametrów, naniesienia punktów na wykres, itp. Wydłuża to czas kontroli oraz powoduje niechęć do stosowania metod sekwencyjnych w statystycznej kontroli jakości. Zautomatyzowanie tych czynności, to znaczy oprogramowanie planów sekwencyjnych, sprowadziłoby pracę kontrolera do rejestrowania czy dana sztuka jest wadliwa czy nie i na tej podstawie otrzymywałby on decyzję z komputera, co ma robić: badać dalej, odrzucić, czy też przyjąć daną partię.

Plany odbioru odnoszące się do produkcji ciągłej

W produkcji taśmowej wytwarzany wyrób otrzymuje się w sposób ciągły. Dzielenie kontrolowanego produktu na partie może być wtedy technologicznie trudne lub niewspółmiernie kosztowne. Dla tej sytuacji zostały opracowane przez Dodge'a specjalne plany odbioru zwane planami CSP (Continuous Sampling Plan). Podstawowym rodzajem tych planów jest plan CSP-1 (rys.6).

x/ AQL (Acceptable Quality Level) - taka wadliwość dopuszczalna, przy której dostawca ponosi małe ryzyko równe α nieprzyjęcia partii

xx/ LTPD (Lot Tolerance Per Cent Defective) - taka wadliwość, przy której istnieje małe ryzyko odbiorcy β przyjęcia partii [5].



Rys.6. Schemat planu CSP-1

W myśl tego planu tryb postępowania przy kontroli odbiorczej jest następujący[x]:

1) należy rozpocząć od stuprocentowej kontroli kolejno wykonywanych sztuk i prowadzić ją dopóty, dopóki nie natrafi się na i kolejnych sztuk bez wad;
2) z chwilą, gdy znalazła się taka seria i dobrych sztuk, należy przerwać kontrolę stuprocentową i kontrolować tylko pewną frakcję f ogólnej liczby sztuk, wybierając do badania pojedyncze sztuki spośród spływających z produkcji wyrobów w sposób zapewniający bezstendencyjność próbki;
3) gdy pobrana do badania próbka okaże się wadliwa, należy natychmiast przywrócić kontrolę stuprocentową kolejnych sztuk i prowadzić ją znowu dopóty, dopóki nie znajdzie się i kolejnych dobrych sztuk, analogicznie, jak w punkcie 1;
4) wszystkie znalezione wadliwe sztuki należy poprawić lub zastąpić dobrymi.

Dodge i Torrey opracowali dwie modyfikacje planu CSP-1, nazwane przez nich CSP-2 (rys.7) i CSP-3. Plan CSP-2 jest opisany[xx] w następujący sposób:

"Różnica między planem CSP-2 i planem CSP-1 polega na tym, że gdy w planie CSP-2 wprowadziło się już kontrolę wyrywkową, to pojawienie się jednej wadliwej sztuki nie powoduje powrotu do kontroli stuprocentowej. Wraca się do niej dopiero wtedy, gdy pojawi się druga wadliwa sztuka w serii następnych S sztuk lub wcześniej. Innymi słowy, powraca się do kontroli stuprocentowej wtedy, gdy przy kontroli wyrywkowej zaobserwowano dwie wadliwe sztuki, przedzielone liczbą dobrych sztuk mniejszą lub równą S. W przeciwnym przypadku kontynuuje się badania wyrywkowe".

Plan CSP-3 jest bardziej precyzyjną odmianą planu CSP-2. Jest on bardziej wrażliwy na pojawienie się większej serii wadliwych wyrobów. Według tego planu, gdy trafi się wadliwa próbka (sztuka), przeprowadza się dodatkową kontrolę czterech następnych wchodzących z taśmy sztuk. Jeżeli żadna z tych sztuk nie wykazuje wad, kontynuuje się kontrolę wyrywkową w sposób podany w planie CSP-2. W wypadku znalezienia wśród tej czwórki już pierwszej wadliwej sztuki, przechodzi się od razu na kontrolę stuprocentową według zasad odpisanych w planie CSP-2.

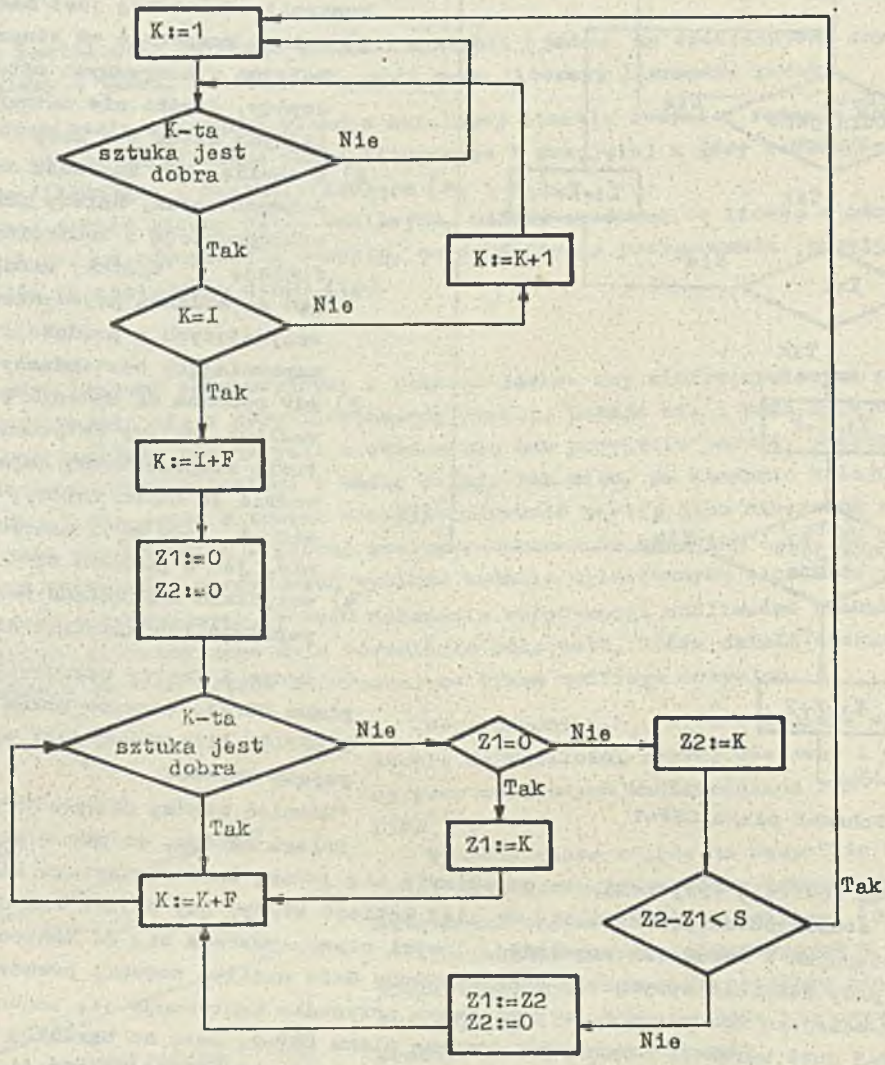
Innym rodzajem planów stosowanych do kontroli jakości produkcji o charakterze ciągłym, są wielopoziomowe plany ciągłych badań wyrywkowych. Jeden z typów wielopoziomowych planów został opracowany i przeanalizowany przez Liebermana i Solomona[xxx] (rys.8). Działanie tych planów polega na odpowiednim, to znaczy zgodnym z przyjętymi w planie zasadami, przechodzeniu z jednej wielkości kontrolowanej frakcji produktów na inną (frakcji tych może być dwie lub więcej).

---

x/ Cytowane z Granta [3]. Dodge H.F.: A Sampling Inspection Plan for Continuous Production. The Annals of Mathematical Statistics 1943 t. 14 Sept. s. 264-279; Dodge H.F.: Sampling Plans for Continuous Production. Industrial Quality Control 1947 t.4 nr 3 s.5-9

xx/ Dodge H.F., Torrey N.M.: Additional Continuous Sampling Inspection Plans. Industrial Quality Control 1951 t.7 nr 5 s. 7-12

xxx/ Lieberman, Solomon: Multi-level Continuous Sampling Plans. The Annals of Mathematical Statistics 1955 t.26 s. 686-704

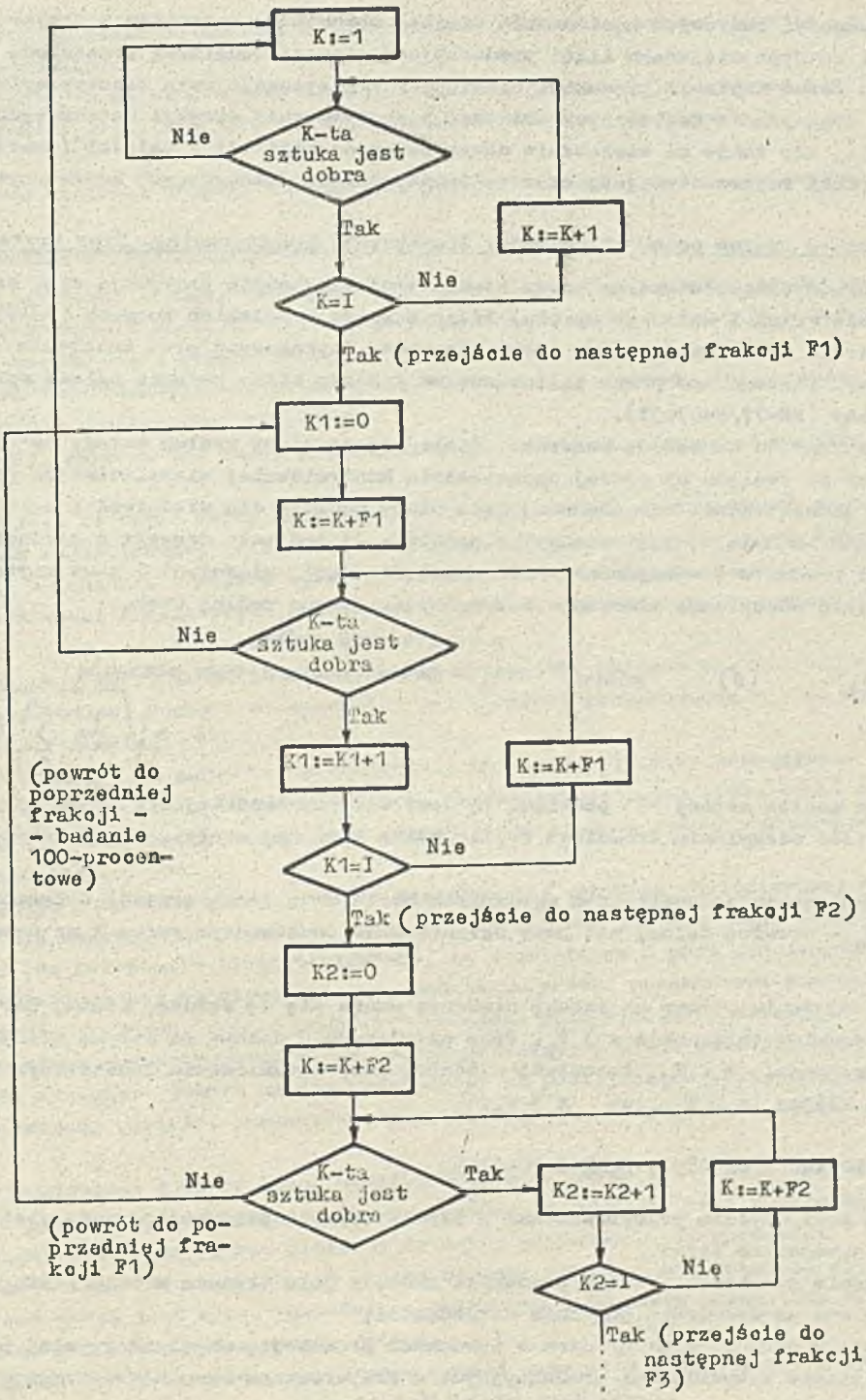


Rys.7. Schemat planu CSP-2

Plan Libermana i Solomona, tak samo jak plany CSP, zaczyna się od kontroli stuprocentowej, prowadzonej do momentu znalezienia serii i dobrych sztuk. Rozpoczynając kontrolę wyrywkową, zakłada się pewną wartość f frakcji badanych sztuk. Jeżeli przy takiej kontroli wyrywkowej znaleziono i kolejnych dobrych sztuk, przyjmuje się do dalszej kontroli frakcję f_2. Następna seria i kolejnych dobrych sztuk uprawnia do przyjęcia frakcji f_3, itd. Gdy przy tej procedurze trafi się wadliwa sztuka, powraca się do poprzedniej reakcji.

Opisane wyżej plany ciągłych badań wyrywkowych nie wyczerpują naturalnie wszystkich istniejących możliwości tworzenia takich planów. Można układać plany, w myśl których odbiór rozpoczyna się po kontroli wyrywkowej, a nie stuprocentowej. Plany takie opierają się na zaufaniu, że przy aktualnym procesie produkcyjnym uzyska się równie dobrą jakość, jak ta, która wynikała z poprzednio badanych próbek. Podstawą do decyzji przejścia z wyrywkowej kontroli na stuprocentową są łączne dane z poprzednich badań próbek.

Plany ciągłych badań wyrywkowych powinny być stosowane pod warunkiem, że jakość przedstawionego do odbioru produktu jest na tyle dobra, że większa jego część będzie mogła przejść przez kontrolę bez potrzeby segregowania. Zasadnicza różnica między planami odbioru partii po partii i planami ciągłych badań wyrywkowych jest ta, że w pierwszym wypadku niepomyślny wynik badania próbki pociąga za sobą kontrolę stuprocentową całej partii, czyli działanie "do tyłu", natomiast w drugim - kontrolę stuprocentową stosuje się do dalszej produkcji, czyli działa się



Rys.8. Schemat planu Liebermana i Solomona

"do przodu". W pewnych sytuacjach uwarunkowanych rodzajem produkcji, można zmodyfikować procedurę planów odbioru ciągłego tak, aby należało w wypadku natrafienia na złą próbkę kontrolować stuprocentowo pewną liczbę sztuk wykonanych bezpośrednio przed wadliwą próbką.

Plany odbioru odnoszące się do produkcji ciągłej są, jak to już zostało wspomniane, niezwykle interesujące z punktu widzenia zastosowań maszyn cyfrowych w dziedzinie statystycznej kontroli jakości.

Ponieważ algorytm tych planów jest dość zawiły (rys. 6,7,8) , wymaga od kontrolera ciągłej uwagi i częstego podejmowania decyzji, jest on w obecnej praktyce prawie wcale nie stosowany.

Wobec trudności kadrowych konieczność ciągłej obecności kontrolera w miejscu dokonywania kontroli, czyli różnych miejscach linii produkcyjnej stanowi dodatkową przeszkodę w praktycznym zastosowaniu metod kontroli produkcji ciągłej. W tej sytuacji rola maszyny cyfrowej nie polegałaby tylko na dokonywaniu koniecznych obliczeń i podejmowaniu decyzji dotyczących przebiegu dalszej kontroli, ale także na sterowaniu samym procesem kontroli w całości (zautomatyzowanie procesu wyboru próbki reprezentacyjnej oraz zautomatyzowanie dokonywanych koniecznych pomiarów).

Plany odbioru według oceny właściwości liczbowych. Zasady ogólne. Typy kryteriów

W procedurach odbiorów według oceny właściwości liczbowych przyjmuje się, że kontrolą objęta jest cecha mierzalna X badanego wyrobu. Plany zawarte w Polskich Normach (PN-77/N-03031), a dotyczące kontroli odbiorczej według oceny liczbowej, opracowano przy założeniu, że cecha ta ma rozkład normalny. Dlatego też przed zastosowaniem takiego planu badania należy sprawdzić normalność rozkładu cechy (PN-77/N-03031).

Plany, ze względu na metodę kontroli dzieli się na plany według metody "s" i plany według metody "δ", zaś ze względu na rodzaj ograniczenia kontrolowanej właściwości na plany badania dla właściwości jednostronnie ograniczonej oraz plany badania dla właściwości ograniczonej dwustronnie. W planach badania według metody "s" podstawą do podjęcia decyzji o zgodności lub niezgodności partii produktu z wymaganiami ustalonymi dla danej właściwości jest wartość średniej $\bar{X}$ z próby i wartość odchylenia średniego S z próby obliczane według wzorów:

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \qquad (6)$$

gdzie: n - liczność próby, X_i - pomiar cechy i-tego elementu

$$S = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2} \qquad (7)$$

W planach według metody "δ" podstawą tą jest wartość średniej $\bar{X}$ z próby, przy czym zakłada się, że wartość odchylenia średniego δ jest stała i znana, a nie dopiero estymowana, jak w metodzie "s".

Przy ograniczeniu jednostronnym wyróżniona cecha X ma jedną granicę tolerancji: T_g - granicę górną lub T_d - granicę dolną, zaś przy ograniczeniu dwustronnym cecha X ma narzucone obie granice T_g i T_d.

Przy ograniczeniu górnym za sztukę niedobrą uważa się tę sztukę, której wartość cechy X jest większa od granicy tolerancji $x > T_g$. Przy ograniczeniu dolnym za sztukę niedobrą uznaje się tę, dla której zachodzi $x < T_d$. Za sztukę niedobrą przy ograniczeniu dwustronnym uznaje się taką sztukę, dla której $x > T_g$ lub $x < T_d$.

BIEŻĄCA KONTROLA JAKOŚCI. CELE I PODZIAŁ

Kontrola statystyczna przeprowadzana w czasie trwania produkcji, zwana jest kontrolą bieżącą. Ma ona dwa zasadnicze cele:

1) obserwowanie przebiegu procesu produkcji podczas jego trwania w celu stwierdzenia, czy utrzymuje się ona na ustalonym poziomie dokładności;
2) zauważenie w porę istotnych zmian w warunkach produkcji, mogących wywołać powstawanie braków, współdziałanie z wydziałami produkcyjnymi w przywróceniu warunków wymaganych przez proces technologiczny.

Metody bieżącej SKJ można dzielić na podstawie różnych kryteriów:

- w zależności od rodzaju kontrolowanych cech wyrobu - na metody kontroli według oceny alternatywnej lub liczbowej;
- w zależności od wskaźników statystycznych, które przyjmuje się do oceny produkcji - na metody oparte na średnich wartościach cech (średnia arytmetyczna, mediana, itp.) i ich rozrzutach (odchylenie średnie, rozstęp, itd.) oraz metody oparte na wartościach cech poszczególnych sztuk wchodzących w skład próbki;
- w zależności od sposobu rejestrowania wyników - na metody tabelaryczne lub wykreślne.

Podstawy, na których opierają się metody bieżącej SKJ są takie same, jak podstawy procedur odbiorczych. W kontroli wadliwości korzysta się z rozkładu Bernoulli'ego, zaś w metodach kontro-

li opartych na ocenie cech mierzalnych zakłada się normalność rozkładu i stosuje się to samo estymatory parametrów tego rozkładu.

Dokładny opis poszczególnych metod bieżącej kontroli jakości, podobnie jak i metod kontroli odbiorczej, a także charakterystyki niektórych t nich, znaleźć można między innymi w pracy autorki [2].

Do prowadzenia bieżącej kontroli jakości w zakładach pracy służą tak zwane karty kontrolne.

Zakładanie i prowadzenie kart kontrolnych

Karta kontrolna jest to wykres służący do rejestracji wyników kontroli jakości produktów w miejscu ich wytwarzania. Formularz karty kontrolnej jednotorowej, gdzie tor jest to układ współrzędnych prostokątnych, składa się z trzech części.

Część pierwsza zawiera dane identyfikacyjne dotyczące produkowanego przedmiotu, miejsca i czasu wytwarzania, wykonawcy, kontrolera, itp.

Druga, zasadnicza część, zawiera tor karty przeznaczony do notowania liczbowych wyników kontroli.

Tor karty kontrolnej jest to część karty kontrolnej stanowiąca układ współrzędnych prostokątnych; oś pozioma służy do rejestrowania kolejności chronologicznej, na przykład przez notowanie kolejnego numeru próbki, związanego z czasem wytworzenia określonej części produktu. Na osi pionowej zaznacza się wielkości charakteryzujące jakość produktu, reprezentowaną przez wyniki badania kolejnej próbki.

Jeśli nie jest narzucona z góry, to na podstawie badań wstępnych, oblicza się wartość estymatora wartości oczekiwanej badanej cechy i na wysokości tej wartości przeprowadza się na wykresie linię poziomą, tzw. linię centralną.

W zwykłych warunkach kontroli na torze karty kontrolnej umieszcza się dwie zewnętrzne linie kontrolne równoległe do linii centralnej: górną i dolną. Przekroczenie w dół przez punkt kontrolny zewnętrznej dolnej linii oraz w górę zewnętrznej górnej linii stanowi sygnał zachodzącej zmiany w procesie produkcyjnym.

Linie kontrolne obejmują obszar, w którym powinny zawierać się wartości kontrolowanej cechy. Przy założeniu, że średnia wartość danej cechy w całej produkcji (partii) nie ulega zmianie i zachowuje wartość określoną położeniem linii centralnej, to z ustalonym z góry prawdopodobieństwem, np. 99%, punkt odpowiadający wyestymowanemu na podstawie próbki parametrowi powinien znaleźć się w obrębie linii kontrolnych. Jeżeli punkt ten wypadnie poza liniami kontrolnymi, to z tym samym prawdopodobieństwem można stwierdzić, że proces produkcyjny uległ rozregulowaniu. Jednak może się zdarzyć, średnio w jednym na sto przypadków, że punkt znajduje się poza liniami kontrolnymi przy niezmienionej wartości parametru w partii. Sygnał rozregulowania będzie wtedy mylny.

Trzecia część karty kontrolnej zawiera różne notatki dotyczące zmian w surowcu, narzędziach, przyczynach powstawania poszczególnych sygnałów i przeprowadzania rewizji linii kontrolnych.

Gdy wprowadza się kartę kontrolną po raz pierwszy do danej produkcji, gdy przedmiot produkcji uległ zmianie, gdy zmieniono surowiec lub ogólne warunki produkcyjne, dokonuje się badań wstępnych. Celem tych badań jest ocena wartości oczekiwanych nieznanych parametrów statystycznych badanej cechy na podstawie wylosowanych próbek.

Badania wstępne mogą być jednostopniowe, gdy trzeba ocenić parametry statystyczne i ustalona jest liczność próbki lub dwustopniowe, gdy liczność próbki nie jest dana.

Jeśli w procesie produkcyjnym nawet małe zmiany wartości parametru statystycznego wpływają istotnie na zmianę jakości produkcji, norma PN-65/N-03012 zaleca pobierać często duże próbki, jeśli natomiast można dopuścić duże zmiany wartości parametru statystycznego w procesie produkcyjnym, wówczas należy pobierać często[x/] małe próbki. Norma ta jednak nie podaje uniwersalnej metody ustalania liczności próbek, gdyż każdy rodzaj karty kontrolnej, ze względu na badane różne parametry statystyczne, wymaga osobnego ustalania tej liczności. Odgrywają tu również dużą rolę względy techniczne i ekonomiczne.

---

x/ Ocenę dotyczącą odstępu czasu pomiędzy próbkami pozostawiono intuicji ustalającego

Jednym z celów badań wstępnych jest wyznaczenie wartości normalnej nieznanego parametru statystycznego. Wartość normalna parametru, jest to wartość obliczana na podstawie przypadkowego zbioru obserwacji, to znaczy, że wyniki badań poszczególnych elementów próbki muszą stanowić zbiór przypadkowy. Badania przypadkowości dokonuje się za pomocą jednego z testów serii według zaleceń normy PN-55/N-03012.

Sposoby obliczania linii kontrolnych podawane są przez odpowiednie normy dla każdego rodzaju kart.

Metody kontroli jakości w badaniach podczas produkcji dzielą się w zależności od tego, jakie wskaźniki przyjmuje się za istotne. I tak dla oceny według średnich wartości cechy metody kontroli oparte są na badaniu zmienności dwóch wskaźników statystycznych, a mianowicie jednej ze średnich wartości kontrolowanej cechy oraz jednej z miar rozrzutu cechy. Pierwszy z tych wskaźników daje możność stwierdzenia, czy nie zaszło odchylenie od założonej wartości cechy w partii, drugi zaś świadczy o rozrzucie wartości cechy w partii.

Dla ocen dokonywanych na podstawie indywidualnych wartości cechy reprezentatywna jest metoda kontroli skrajnych wartości: najmniejszej i największej. W metodzie tej rejestruje się na wykresie jedynie skrajne wartości cechy w każdej próbce.

Przy cechach alternatywnych wyróżnia się metodę kontroli wadliwości - w oraz metodę kontroli liczby sztuk wadliwych w próbce - z. W pierwszej z nich wskaźnikiem statystycznym jest stosunek liczby sztuk wadliwych w próbce do liczności próbki; obliczoną wartość tego stosunku nanosi się w postaci punktu na wykres; w drugiej liczbę sztuk wadliwych w próbce porównuje się na wykresie z naniesionymi tam liniami ograniczającymi.

PORÓWNANIE METOD KONTROLI WEDŁUG OCENY ALTERNATYWNEJ Z METODAMI KONTROLI WEDŁUG OCENY CECHY MIERZALNEJ

Przy większości badań przeprowadzanych w ramach statystycznej kontroli jakości stosuje się ocenę alternatywną, mimo tego, że rozwój wiedzy w dziedzinie SKJ doprowadził do znacznego rozszerzenia zakresu stosowania w przemyśle badań według oceny właściwości liczbowych. Jedną z oczywistych przyczyn ograniczających stosowanie tej ostatniej metody jest fakt, że wiele właściwości charakteryzujących jakość produktu ocenianych jest jedynie kwalitatywnie. W takich wypadkach nie można stosować kontroli według oceny właściwości liczbowych. Zdarza się jednak często mylne rozpoznanie takiej sytuacji, związane z zastosowaniem niewłaściwych kryteriów oceny jakości lub brakiem rozeznania w istniejących możliwościach dokonywania pomiarów.

Stwierdzony został fakt [3], że przy kontroli cechy mierzalnej koszty kontroli na jednostkę produktu są zwykle mniejsze, gdy stosuje się ocenę alternatywną niż przy ocenie na podstawie wartości liczbowej tej cechy. Spowodowano jest to faktem, że kontrola na podstawie oceny wartości liczbowych wymaga wykonania licznych prac kreślarskich i obliczeniowych. Gdy o jakości produktu decyduje wiele właściwości, dla każdej z nich należy określić i stosować oddzielne kryteria. To właśnie powoduje wzrost kosztów kontroli i wydłuża czas jej trwania.

Mimo wymienionych wad i ograniczeń, metody statystycznej kontroli jakości oparte na ocenie wartości liczbowej cechy mają wyraźną przewagę nad innymi, szczególnie gdy kontrolowana cecha jakościowa jest źródłem poważnych trudności produkcyjnych. Przewaga ta polega na fakcie, że ocena liczbowa daje o wiele więcej informacji na temat procesu produkcyjnego i wytwarzanego wyrobu niż ocena alternatywna, a co za tym idzie stosowanie jej ułatwia oddziaływanie na jakość produktu. Poza tym, metody kontroli oparte na ocenie liczbowej badanej cechy są efektywniejsze w tym sensie, że pozwalają na osiągnięcie tych samych efektów w dziedzinie kontroli jakości wyrobu (tak samo zapewniają zachowanie odpowiedniego poziomu jakości) przy mniejszej liczności próbek, niż wymaga tego kontrola według oceny alternatywnej. W praktyce można nie ograniczać się do prowadzenia kontroli danej produkcji za pomocą jednej tylko z porównywanych tu dwóch metod. W pewnej fazie wprowadzania statystycznej kontroli jakości prowadzi się ją na podstawie oceny liczbowej cechy, a gdy już wszelkie uchwytne zakłócenia procesu produkcyjnego zostaną usunięte i proces będzie można uważać za ustabilizowany, można przejść na kontrolę alternatywną. Inny sposób takiej koegzystencji dwóch metod polega na stosowaniu kontroli według miary cechy tylko do pewnych cech wyrobu (mogą to być cechy, które stwarzają najpoważniejsze kłopoty natury jakościowej), zaś inne mniej ważne cechy tego samego wyrobu kontroluje się na podstawie oceny alternatywnej.

WNIOSKI

Przytoczone do tej pory uwagi ogólne na temat statystycznej kontroli jakości miały na celu stwierdzenie czy warto, a jeśli tak, to gdzie i do czego stosować w tej dziedzinie elektroniczne maszyny cyfrowe. Już sam fakt, że bezpośrednio w miejscu wytwarzania danego produktu stosuje się tylko najprostsze metody, nie dające wiele informacji na temat zmian zachodzących w procesie produkcyjnym wskazują na to, że należałoby skomputeryzować takie metody jak sekwencyjna, czy plany odbiorów ciągłych. Z opisu tych metod widać, że nie jest możliwa ich popularyzacja w wydaniu "ręcznym", gdyż ich algorytmy są zbyt zawile lub wymagają takich dodatkowych czynności, jak sporządzanie żmudnych wykresów. Rysunki 6, 7 i 8 przedstawiające schematy planów odbioru ciągłego wykazują, że napisanie odpowiednich programów nie byłoby zbyt trudne, natomiast zastosowanie tych metod przez kontrolera pozbawionego współpracy komputera wręcz niemożliwe.

Projekt zastosowania elektronicznych maszyn cyfrowych do wykonywania wszelkich obliczeń związanych ze statystyczną kontrolą jakości procesu produkcyjnego możnaby także rozszerzyć na sterowanie procesem produkcyjnym. Pod pojęciem sterowania rozumiem wyciąganie odpowiednich wniosków z przeprowadzonych badań statystycznych i na ich podstawie natychmiastowe interweniowanie w kontrolowany proces. Skomputeryzowany zostałby więc także sam proces podejmowania decyzji dotyczących dalszego przebiegu produkcji oraz dalszego przebiegu kontroli. Ideałem byłoby również zautomatyzowanie procesów pomiarowych, przez co osiągnęłoby się dopiero pełną automatyzację całokształtu statystycznej kontroli jakości.

Literatura

[1] Fisz M.: Rachunek prawdopodobieństwa i statystyka matematyczna. Warszawa 1958
[2] Friling T.: Statystyczna kontrola jakości. Rozpoznanie metod. Warszawa: IMM 1979 Archiwum Opracowań IMM nr 72
[3] Grant E.L.: Statystyczna kontrola jakości. Warszawa 1972
[4] Obalski J.: Statystyczna kontrola jakości podczas produkcji. Warszawa 1955
[5] Wasilewski K.: Metody kontroli jakości w przedsiębiorstwach przemysłowych. Warszawa 1974
[6] Wybrane problemy kontroli jakości produkcji. Warszawa: Zakład Badań Społecznych Instytutu Organizacji Przemysłu Maszynowego 1977.

TŁUMACZ-TEKST 0/56/82001A.

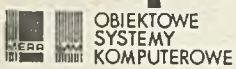
BRANŻOWY OŚRODEK INFORMACJI NAUKOWEJ TECHNICZNEJ I EKONOMICZNEJ
INSTYTUTU MASZYN MATEMATYCZNYCH
02-078 Warszawa, ul. Krzywickiego 34, tel. 21-84-41 w. 391

BOINTE udziela informacji

z zakresu techniki komputerowej

BOINTE wydaje

informacja ekspresowa



przegląd dokumentacyjny

biuletyn informacyjn

Materiały konferencyjne, szkoleniowe, prospekty

BOINTE gromadzi

wydawnictwa zwarte, czasopisma krajowe i zagraniczne, katalogi i prospekty, sprawozdania z prac naukowo-badawczych oraz inne materiały informacyjne

BOINTE wykonuje usługi reprodukcyjne i poligraficzne

fotokopie, mikrofilmy, kserokopie z zakresu posiadanych zbiorów

Cena zł. 140. —

WARUNKI PRENUMERATY

Prenumeratę na kraj przyjmują Oddziały RSW "Prasa-Książka-Ruch" oraz urzędy pocztowe i doręczyciele w terminie do dnia 25 listopada na rok następny.

Cena prenumeraty rocznej zł 840.

Jednostki gospodarki uspołecznionej, instytucje, organizacje i wszelkiego rodzaju zakłady pracy zamawiają prenumeratę w miejscowych Oddziałach RSW "Prasa-Książka-Ruch", w miejscowościach zaś, w których nie ma Oddziałów RSW - w urzędach pocztowych.

Czytelnicy indywidualni opłacają prenumeratę wyłącznie w urzędach pocztowych i u doręczycieli.

Prenumeratę ze zleceniem wysyłki za granicę przyjmuje RSW "Prasa-Książka-Ruch", Centrala Kolportażu Prasy i Wydawnictw, ul. Towarowa 28, 00-958 Warszawa, konto PKO Nr 1153-201045.

Prenumerata ze zleceniem wysyłki za granicę jest droższa od prenumeraty krajowej o 50% dla zleceniodawców indywidualnych i o 100% dla zlecających instytucji i zakładów pracy.

INDEKS 35362