OŚRODEK BADAWCZO-ROZWOJOWY INFORMATYKI

OPROGRAMOWANIE DLA TRANSMISJI DANYCH

Europejski Program Badawczy Diebolda

54

Warszawa 1974

OŚRODEK BADAWCZO-ROZWOJOWY INFORMATYKI

OPROGRAMOWANIE DLA TRANSMISJI DANYCH

Europejski Program Badawczy Diebolda

Wyłącznie do użytku na terenie PRL

54

Warszawa 1974

Tytuł oryginału: SOFTWARE FOR DATA COMMUNICATIONS
Document No E 101, November 1972

Tłumaczenie: Joanna Chrzanowska-Murray
Redakcja: Andrzej Idźkiewicz

Wydawca
Działowy Ośrodek Informacji, Warszawa, ul.Marszałkowska 104/122

OBRI, Warszawa 1974 r. Nakład: 850+165 egz. Obj.ark.wyd. 4,60; ark.druk. 13. Format A4. Papier offsetowy kl. III, 80g, 61x86

Druk DOI. Zam. nr342/73 nr. zezw. DN-444-13/73 Cena zł. 92.-

SPIS TREŚCI

s.

STRESZCZENIE 7

I. KATEGORIE OPROGRAMOWANIA DLA TELEPRZETWARZANIA 8

II. CZEGO NALEŻY WYMAGAĆ OD OPROGRAMOWANIA DLA TRANSMISJI DANYCH? 20

III. PODSTAWOWY PAKIET OPROGRAMOWANIA 32

IV. PRZY WIĘKSZYCH MOŻLIWOŚCIACH PAMIĘCI WEWNĘTRZNEJ 39

V. OPROGRAMOWANIE DATA BASE /DATA COMMUNICATIONS/DBDC/..... 47

VI. ROZDZIAŁ WEJŚCIA/WYJŚCIA LOGICZNEGO OD FIZYCZNEGO 56

VII. OPROGRAMOWANIE W MASZYNACH PERYFERYJNYCH 60

VIII. JĘZYKI DLA ANALITYKA SYSTEMÓW 67

IX. SYSTEMY DIALOGOWE 73

X. GŁÓWNE KRYTERIA WYBORU OPROGRAMOWANIA 78

XI. ZESTAWIENIE CECH OPROGRAMOWANIA DLA TRANSMISJI DANYCH 83

XII. AKTUALNE PRZYKŁADY 91

BIBLIOGRAFIA 104

PODZIĘKOWANIE

Książki Jamesa Martina wymienione w bibliografii były szeroko wykorzystywane podczas przygotowania tego sprawozdania. Rozdziały II, III i IV wzięto z "Systems Analysis for Data Transmission", za zgodą wydawcy.

STRESZCZENIE

Zwiększona zdolność ostatnio wprowadzanych komputerów do zdalnego przetwarzania i stale rozbudowująca się infrastruktura urządzeń transmisji danych powodują rosnące zainteresowanie oprogramowaniem dla systemów, które wykorzystują te możliwości. Dostępny jest duży wybór oprogramowania dostarczanego przez producentów sprzętu, które zaspakaja tego rodzaju potrzeby i niniejsze sprawozdanie wyszczególnia klasy, na które można podzielić takie oprogramowanie.

Dokonano przeglądu funkcji, które powinien obejmować pakiet oprogramowania transmisji danych, a następnie zilustrowano te funkcje na przykładzie pakietu należącego do najbardziej popularnych z obecnie dostępnych. W sprawozdaniu dyskutuje się zagadnienie, jak dalece pożądane jest pisanie dodatkowych programów, uzupełniających pakiet podstawowy, a także kombinowanie oprogramowania transmisji danych z innym bardziej uogólnionym oprogramowaniem, jak np. pakiety baz danych. Kombinacja taka zapewnia bardziej kompleksową obsługę systemu. Dyskutuje się również inne tendencje dające się zauważyć w technice teletransmisji danych.

Różne pakiety oprogramowania dla teleprzetwarzania różnią się znacznie postawionymi celami, jak i możliwościami. Użytkownik musi się upewnić, że jego potrzeby pokrywają się z możliwościami rozważanych pakietów. Zbyt często spotyka się rozczarowania, spowodowane brakiem dostatecznego zrozumienia.

Użytkownik powinien zdecydować się, czy potrzebne mu jest oprogramowanie tylko do transmisji danych, czy też do transmisji danych i obsługi bazy danych. Wiele pakietów oprogramowania łączy te obie dziedziny, a wymagania stawiane z uwagi na eksploatację bazy danych przyczyniają się znacznie do skomplikowania samego procesu wyboru.

Jako podstawowe kryteria doboru oprogramowania wymieniane są:

- związane z wybranym wariantem koszty oraz możliwe do uzyskania oszczędności,
- koszty eksploatacji wraz z konserwacją i konwersją,
- korzyści wynikające z zaoszczędzenia czasu pracy własnych programistów,
- urządzenia, które będą obsługiwane przez oprogramowanie,
- wymagania co do szybkości reakcji systemu lub systemów,
- korzyści wynikające z wpływu oprogramowania na wykorzystanie końcówek,
- wpływ na wydajność programistów,
- niezawodność,
- bezpieczeństwo,
- elastyczność.

Właściwy wybór będzie dokonany tylko wówczas, jeśli poświęci się dość uwagi każdemu z powyższych problemów, pamiętając nie tylko o wymaganiach bieżących, ale również o skutkach, jakie wybór oprogramowania będzie miał na zdolność użytkownika do właściwego reagowania na nowe systemy lub wymagania techniczne.

I. KATEGORIE OPROGRAMOWANIA DLA TELEPRZETWARZANIA

Oprogramowanie dla teleprzetwarzania można podzielić na następujące kategorie:

Kategoria 1. Programy dla sterowania funkcją transmisji danych

Niektóre pakiety oprogramowania wykonują jedynie najważniejszą funkcję - sterują operacjami wejścia/wyjścia, niezbędnymi przy transmisji danych. Składa się na to: grupowanie otrzymywanych bitów w znaki i komunikaty i odwrotnie /przy wysyłaniu komunikatu/, plus wykrywanie i korekcja błędów, przepytywanie, /polling/, multipleksowanie, demultipleksowanie itd. Pełną listę tych funkcji podamy dalej. Dostępne obecnie programy reali-

zują je na różne sposoby. Niektóre np. ustawiają komunikaty w kolejki, inne zaś nie. Wiele ośrodków zastosowało programy z niewystarczającym kompletem funkcji i musiało uzupełniać programy w trudnym kodzie assemblera.

Pakiety tej kategorii tworzą często podzespół uzupełniający system operacyjny. Bywają one nazywane "metodą dostępu" systemu operacyjnego. Najbardziej znane są BTAM, QTAM i TCAM, firmy IBM.

Kategoria 2. Programy sterujące inicjowane transakcjami

Niektóre pakiety dla systemów, w których komunikaty teleprzetwarzania inicjują akcję przetwarzania, łączą funkcje transmisji danych z programem sterującym, który generuje harmonogram przetwarzania komunikatów. Do takich systemów należą TCS f-my UNIVAC oraz Transaction Processor Monitor Honeywell'a.

W systemach inicjowanych transakcjami często chodzi w pierwszym rzędzie o wydajność. Czy oprogramowanie i związany z nim hardware korzystają z mechanizmów, które pozwalają osiągnąć żądany czas odpowiedzi i przepustowość? Jest to pytanie kompleksowe i często angażuje czynniki spoza teleprzetwarzania, jak np. mechanizmy wykorzystywanej bazy danych. Aby uzyskać szybkie odpowiedzi dla wielkiej liczby końcówek, może być potrzebny specjalny program sterujący, obsługujący zarówno teleprzetwarzanie, jak i kierowanie kolejnością prac. Przykładem takiego programu jest program sterujący PARS firmy IBM, który został początkowo zaprojektowany dla obsługi rezerwacji miejsc lotniczych, następnie jednak został użyty do innych zastosowań, tam, gdzie wymaganie szybkiej odpowiedzi łączyło się z konieczną dużą przepustowością transakcji.

Kategoria 3. Pakiety Baza Danych/Transmisja Danych

Wiele pakietów zaprojektowano do obsługi zarazem bazy danych i operacji transmisji danych. Niektóre z nich są przede wszystkim zorientowane na strukturę bazy danych - np. IMS firmy IBM - podczas, gdy inne są w pierwszym rzędzie zorientowane na telekomunikację, przy stosunkowo prostych mechanizmach bazy da-

nych - np. CICS IBMu. /Ostatnio opublikowana druga wersja pozwala na dostęp CICS do zbiorów DL/I pakietu IMS/.

Wiele pakietów baza danych/transmisja danych daje użytkownikowi możność wyboru między korzystaniem z należącego do pakietu "front-endu", a użyciem innych środków obsługi teleprzetwarzania, które mogą być realizowane za pomocą osobnego pakietu oprogramowania. /Przykładem tego jest IMS,jak wspomniano wyżej lub TOTAL firmy CINCOM. TOTAL korzysta do obsługi teleprzetwarzania z INTERCOM-u/.Wielu użytkowników chiałoby mieć kilka pakietów oprogramowania w jednym komputerze, aby korzystać z różnych możliwości. Jednak w niektórych przypadkach wymaga to zbyt wielkiej pamięci operacyjnej.

Kategoria 4. Pakiety dialogowe

Czasami pakiet wykracza poza mechanikę teleprzetwarzania, zapewniając strukturę dialogu człowiek-komputer. Wiele z pakietów baza danych/transmisja danych dostarcza użytkownikowi końcówki język porozumiewania się, na przykład Interactive Query Facility IBM-u /IQF/, który pracuje z IMS II. Oprogramowanie przystosowane do specyficznej struktury dialogu może być wydajniejsze niż dialog tworzony przez oprogramowanie ogólnego przeznaczenia. Będzie tak zapewne coraz częściej, w związku z narastającą tendencją do instalowania "inteligentnych" komponentów raczej w peryferyjnej części sieci, aniżeli w centrum. Ważne jest jednak, aby pamiętać, że istnieje wiele możliwych struktur dialogu człowiek-komputer, o różnych zaletach i wadach. Typy oferowane przez określony pakiet nie muszą być najlepsze dla danego użytkownika.

Niektóre ze struktur dialogowych zostały zaprojektowane po to, aby dać język wyższego rzędu dla porozumiewania się z systemem, tak, aby sprawozdania /komunikaty/ mogły być przygotowywane przez nie-programistę. Niektóre ze struktur dialogowych są zorientowane na określoną końcówkę. Jeśli końcówka ma niekonwencjonalną budowę, może mieć swój specjalny pakiet oprogramowania. Niektóre urządzenia dające odpowiedź głosem,na przykład, dostarczane są z kompletnym oprogramowaniem wspierającym.

Kategoria 5. Pakiety do zastosowań

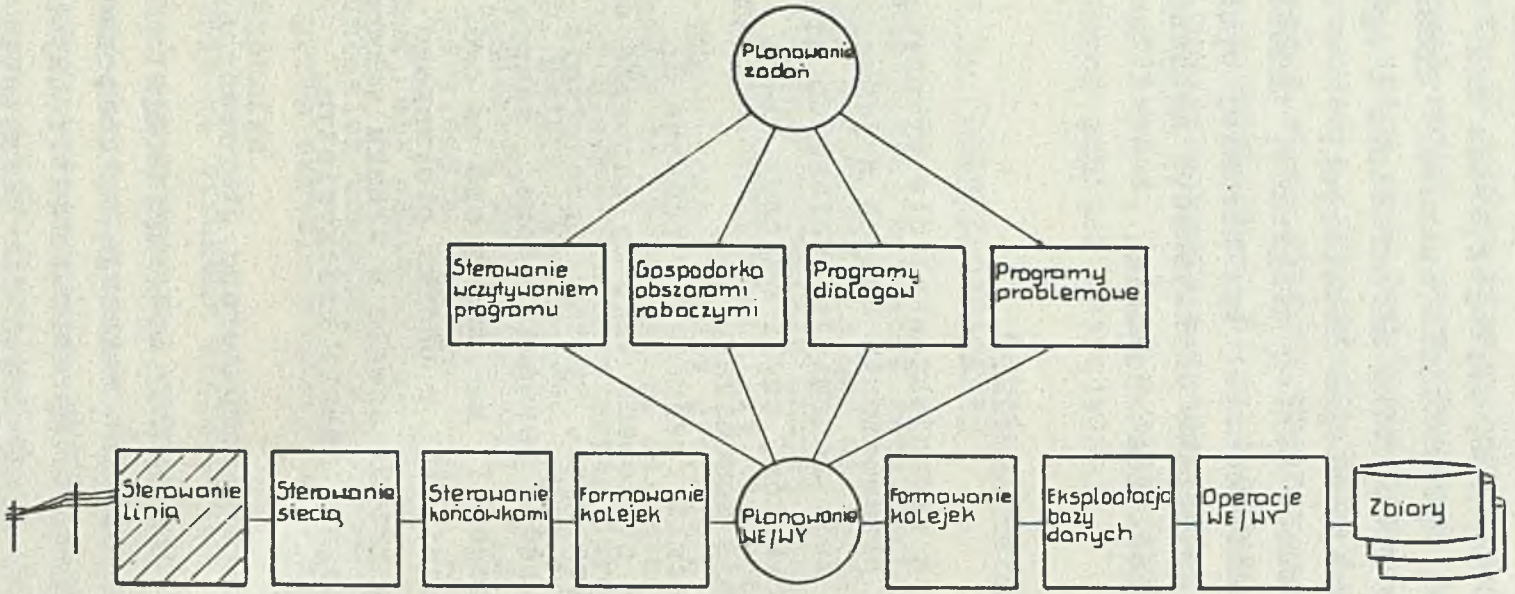
Pewne programy teleprzetwarzania są zaprojektowane dla określonego zastosowania. Czasem są zaprojektowane dla końcówki określonego przeznaczenia, np. końcówka bankowa. Czasem są zaprojektowane dla zastosowania, które ma określone i specyficzne wymagania, jak np. rezerwacja miejsc lotniczych.

Aby móc ocenić efektywność takich programów, trzeba umieć ocenić wymagania danego zastosowania, tym niemniej wymienione w tym sprawozdaniu pytania dotyczące efektywności mechanizmów teleprzetwarzania mają tu nadal zastosowanie.

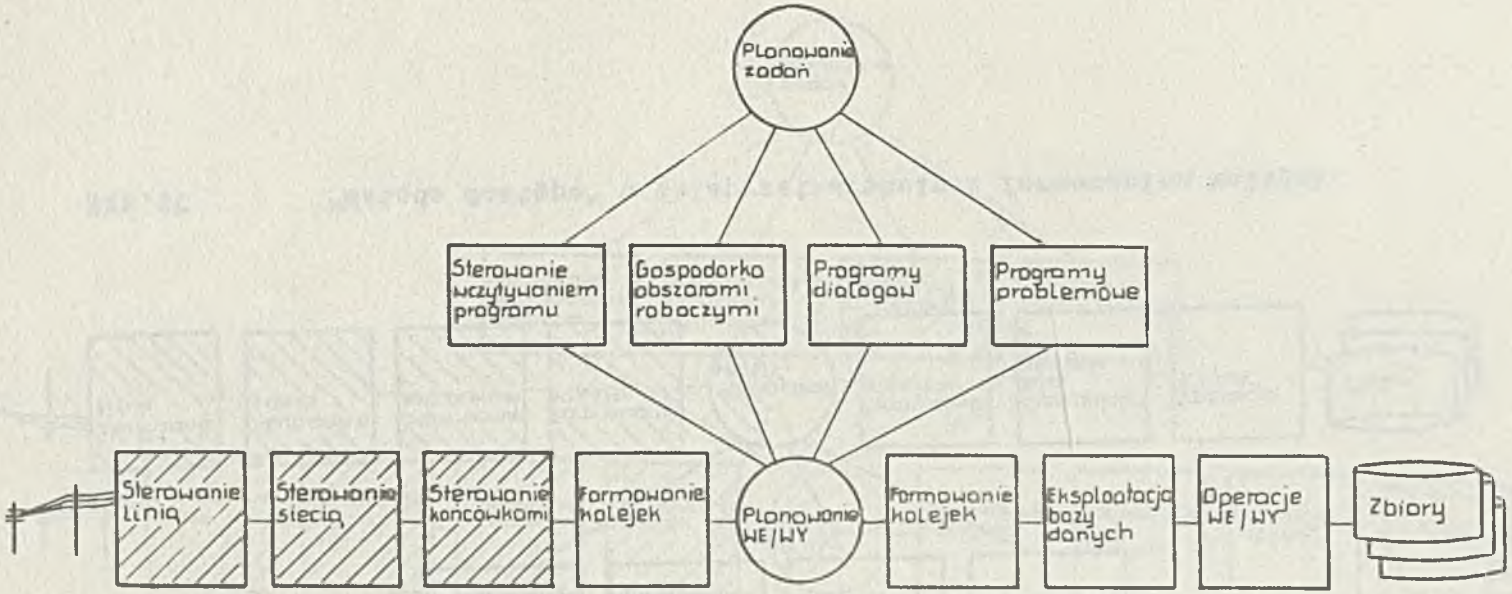
Kategoria 6. Systemy telekomunikacyjne

Niektóre systemy nie są zaprojektowane dla wykonywania przetwarzania danych, ale dla przenoszenia danych z jednego miejsca do drugiego, dla operacji teletransmisji. Należą do nich:
1/ systemy komutacji komunikatów, czyli systemy które zapamiętują, a następnie przesyłają komunikaty,
2/ systemy przesyłające paczki /packets/ - podobne do systemów komutacji komunikatów, ale zbudowane tak, by zapewnić szybkości konieczne dla sieci komputerowych. Systemy przesyłania paczek mogą transmitować dane w każdej postaci kodowej, zazwyczaj w formie paczek ustalonej długości. Nie muszą one na ogół gromadzić danych w pamięci pomocniczej, a paczki pierwszego priorytetu zwykle osiągają swoje przeznaczenie w ułamku sekundy - dostatecznie szybko dla dialogu w czasie rzeczywistym,
3/ systemy gromadzenia danych,
4/ komutacja linii w centralach komputerowo sterowanych.

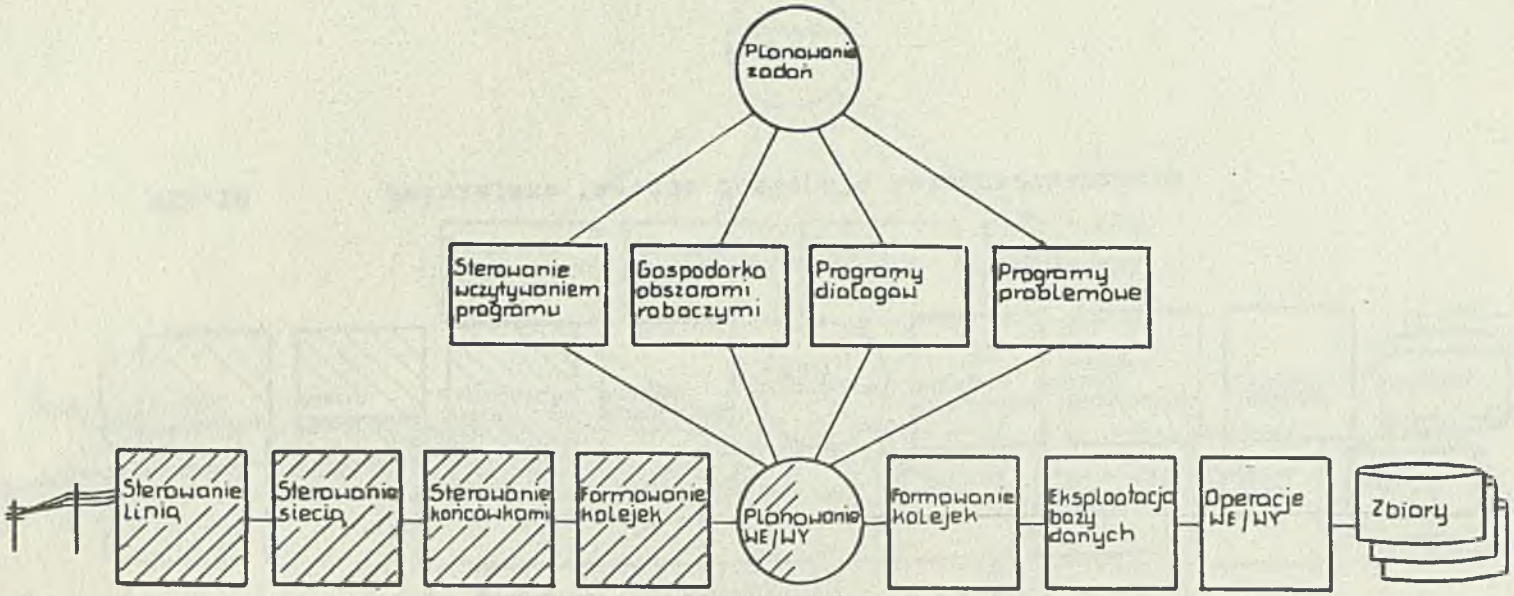
Rys. 1 pokazuje główne komponenty operacyjne systemu teleprzetwarzania. Obszary zacieniowane wskazują na komponenty, które są zapewniane przez pakiety teleprzetwarzania różnych szczebli. Programów pomocniczych, niezbędnych dla wdrażania i konserwacji systemów nie pokazano. Należą do nich kompilatory, pomoce w uruchamianiu, obsługa bibliotek programów, obsługa słowników danych i programy diagnostyczne.



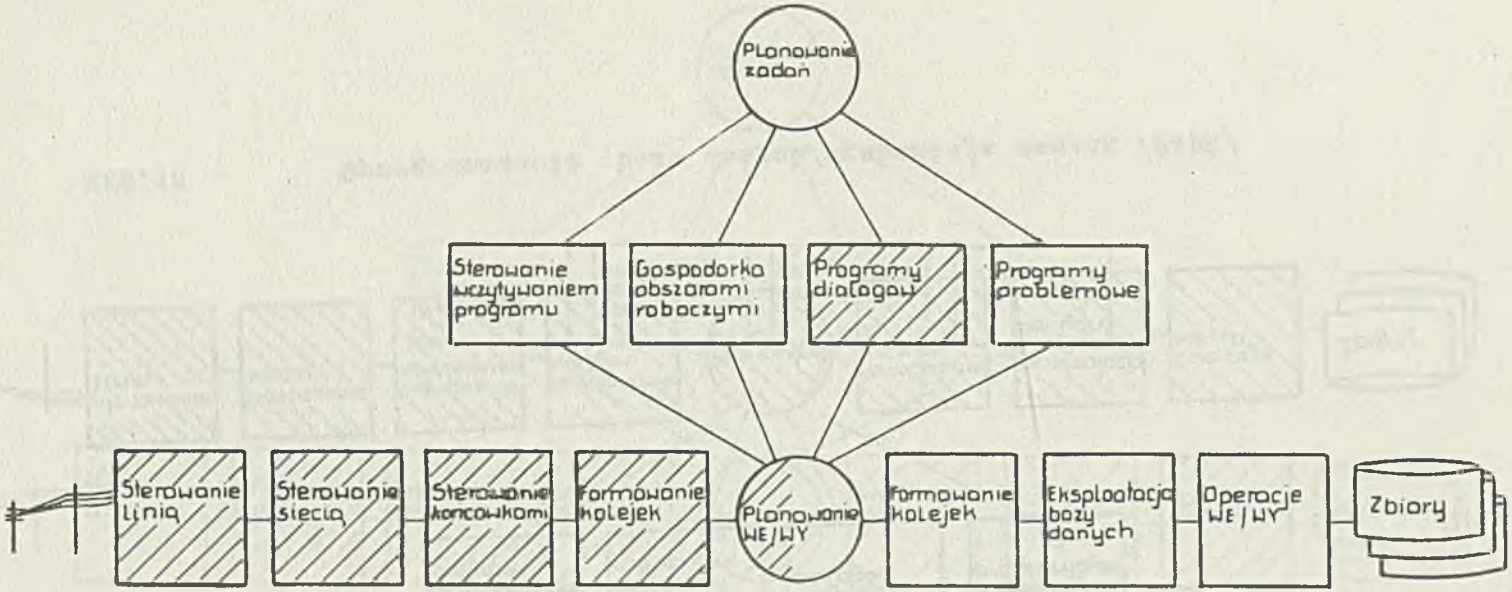
RYS.1A Prosta "metoda dostępu" w teleprzetwarzaniu



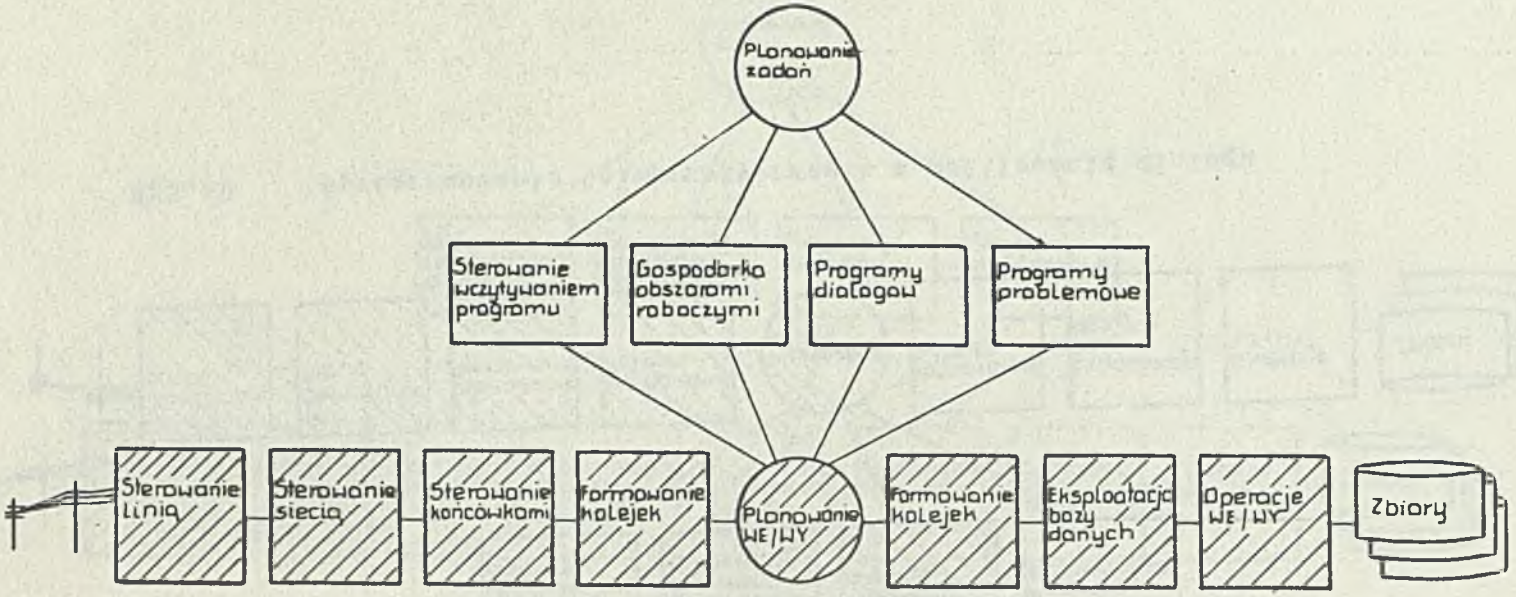
RYS.1B Pełniejsza "metoda dostępu"w teleprzetwarzaniu



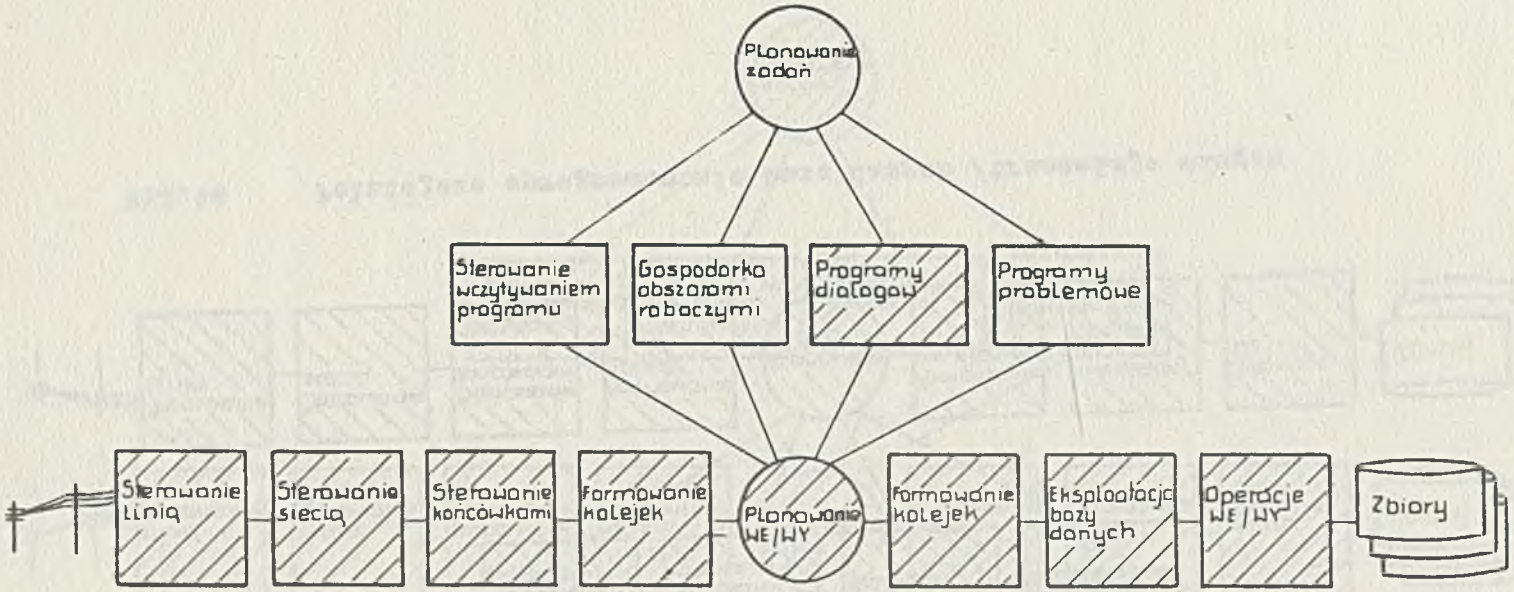
RYS.1C "Metoda dostępu" w teleprzetwarzaniu z formowaniem kolejek



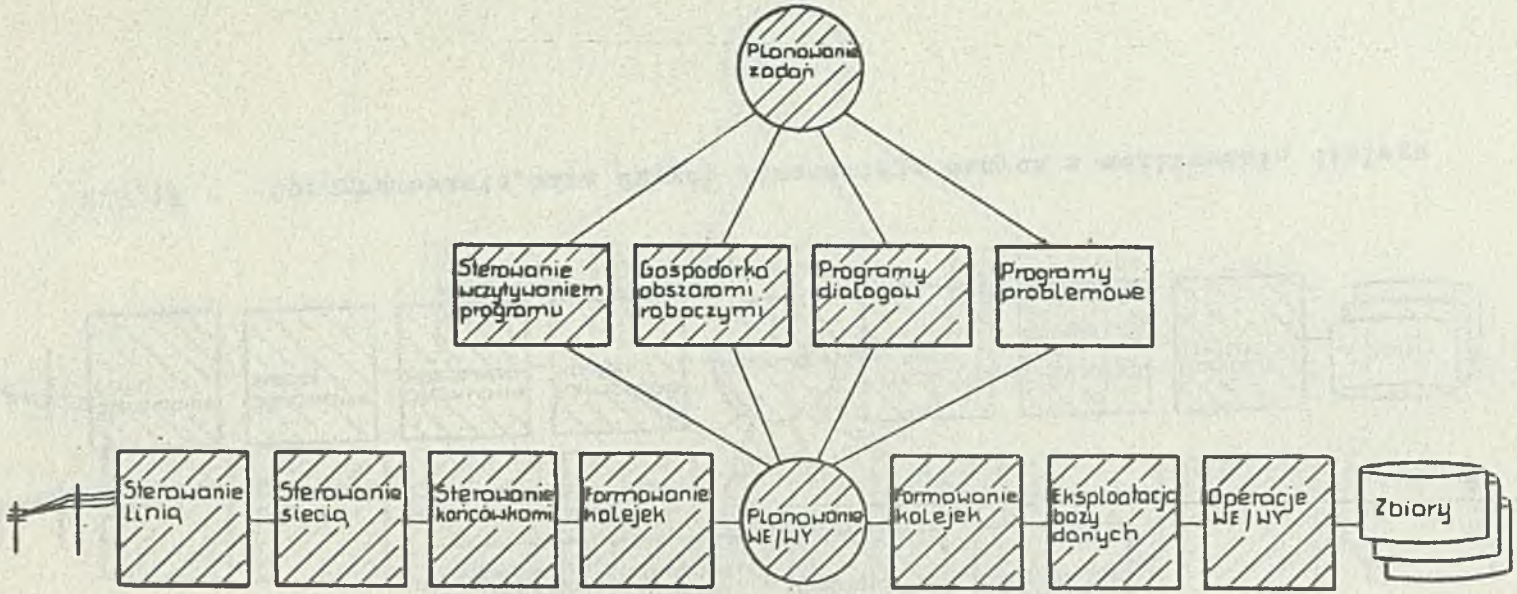
RYS.1D Oprogramowanie teleprzetwarzania z możliwością dialogu



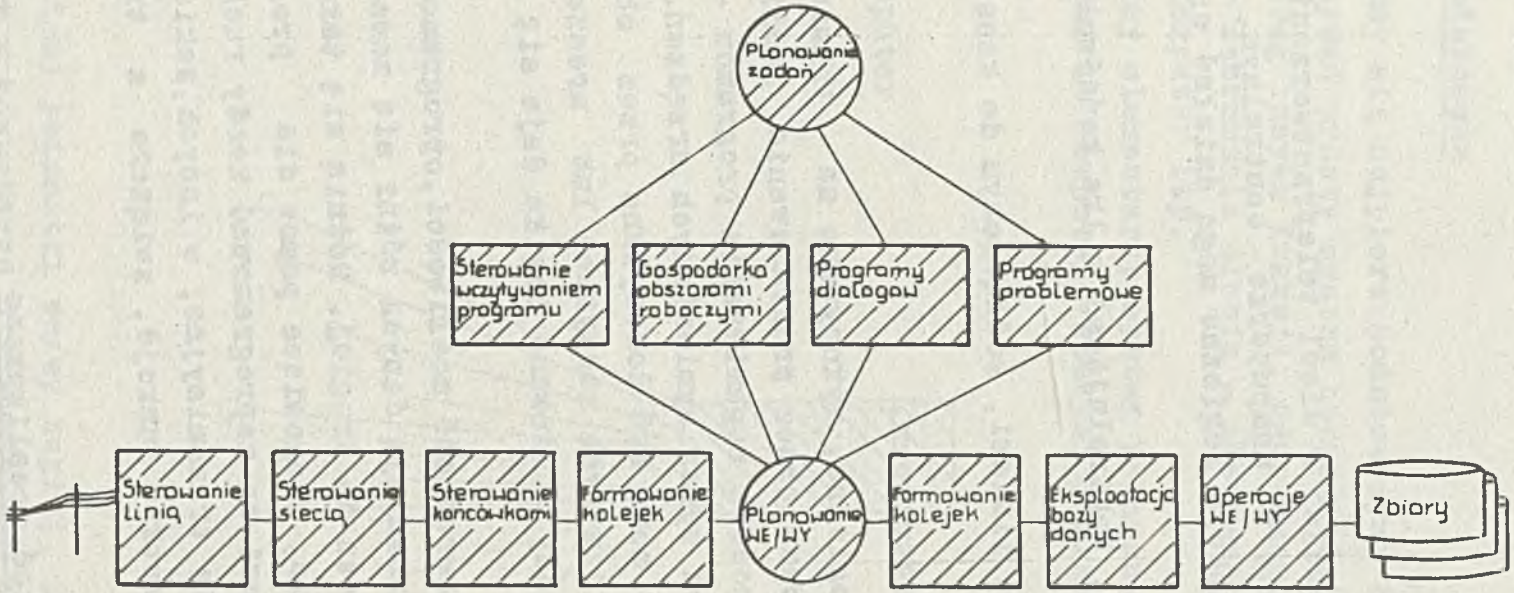
RYS.15 Oprogramowanie ,baza danych/transmisja danych /DBDC/



RYS.1F Oprogramowanie, baza danych /transmisja danych z możliwością dialogu



RYS.10 Pełniejsze oprogramowanie baza danych /transmisja danych



RYS.1II Kompletne pakiety programu problemowego z teleprzetwarzaniem

II. CZEGO NALEŻY WYMAGAĆ OD OPROGRAMOWANIA DLA TRANSMISJI DANYCH?

Tak zwana "inteligencja" w sieci teleprzetwarzania może rezydować niemal całkowicie w komputerze centralnym, bądź być rozproszona w sieci. Funkcje logiczne mogą działać w:

1/ końcówce,
2/ jednostce sterującej, kontrolującej kilka końcówek,
3/ koncentratorze,
4/ urządzeniu sterującym liniami, podłączonym do kanału komputera,
5/ w samym komputerze centralnym.

Wszelkie operacje logiczne, wykonywane na transmitowanych danych mogą być realizowane przez programowanie, jeśli dane urządzenie dysponuje możnością zapamiętania programów - co może odnosić się do wszystkich wyżej wymienionych urządzeń. Z drugiej strony, operacje te mogą być dokonywane przez obwody logiczne, zaszyte na trwałe w tych maszynach lub wreszcie przez mikro-oprogramowanie, kiedy wbudowana logika daje się modyfikować.

Ze względu na tę różnorodność możliwości, oprogramowanie potrzebne do sterowania transmisją danych różni się zasadniczo od jednej konfiguracji sprzętu do drugiej. Różnią się także znacznie pakiety oprogramowania, stanowiące pomoc dla programisty. W niektórych systemach musi on zaprogramować każdy ruch każdego bitu lub znaku przez linie transmisyjne; w innych, pakiety oprogramowania przejmują wszystkie funkcje, związane z transmisją danych.

Funkcje, które muszą zostać zrealizowane

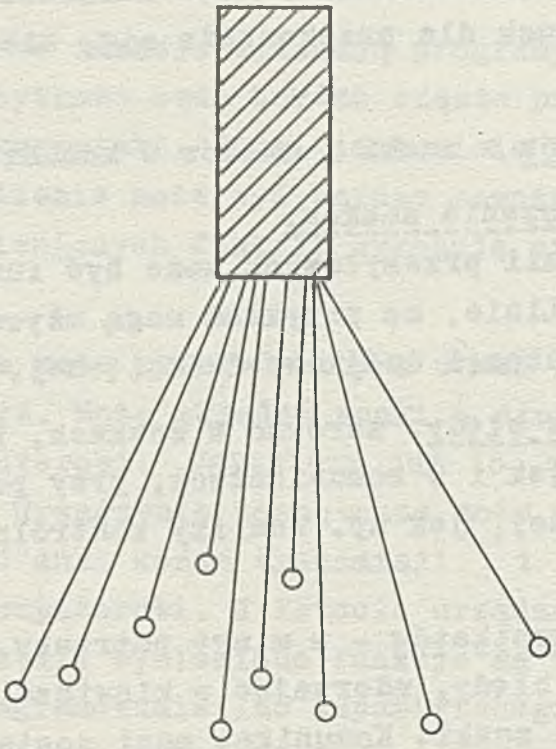
Omówimy wszystkie funkcje, które muszą być zrealizowane dla sterowania transmisją danych, a następnie różne sposoby, w jakie te funkcje mogą być przydzielone logice hardware'u, mikro-

oprogramowaniu, pakietom oprogramowania lub kodowi, napisanemu specjalnie dla systemu.

Funkcje podstawowe

Zajmiemy się najpierw podstawowymi funkcjami, które powinien wykonywać prosty system bezpośredni /on-line/. Następnie dodamy szereg innych funkcji, odnoszących się do bardziej kompleksowego sterowania siecią - sterowania liniami, dialogiem człowiek-komputer itp.

Rozważmy elementarny system pokazany na rys.2, z końcówkami podłączonymi do komputera stałymi liniami od punktu do punktu.



Rys. 2

Bez żadnej jednostki we/wy takiej, jak taśma lub dysk, główne programy w komputerze otrzymują lub przygotowują do wyjścia kompletny zapis lub komunikat. Krótki podprogram pisze lub czyta zapis i sprawdza, czy zostało to wykonane poprawnie. Operacja ta jest także potrzebna przy liniach transmisyjnych, ale wówczas podprogram wejścia/wyjścia może być bardziej skomplikowany.

Informacja cyfrowa przekazywana jest przez linie przesyłowe sekwencyjnie. A zatem komunikaty, które mają być przesłane, rozbijane są na bity i przesyłane z szybkością charakterystyczną dla danej linii. Podobnie, po otrzymaniu komunikatu, bity gromadzone są sekwencyjnie w znaki, a znaki formują komunikaty. Zarówno znaki, jak i komunikaty muszą być sprawdzane na błędy, a błędy, jeśli to możliwe, poprawione. Muszą być generowane odpowiednie sygnały sterujące dla operowania we właściwych momentach odległymi końcówkami.

Muszą być wykonywane następujące funkcje:

1/ inicjowanie i sterowanie odbiorem danych z linii.
Linie mogą mieć różne szybkości. Wiele linii może transmitować lub odbierać jednocześnie. Może zachodzić potrzeba zwracania się do końcówek dla przekonania się, kiedy są gotowe do transmitowania,

2/ składanie bitów w znaki i znaków w komunikaty,

3/ konwersja kodowania znaków.
Kodowanie dla linii przesyłowych może być inne, niż stosowane przez komputer. Linie, na przykład mogą używać kodu, podczas gdy komputer używa notacji dwójkowo-dziesiętnej,

4/ sprawdzanie na błędy, zarówno w znakach, przy pomocy kontroli parzystości, jak i w komunikatach, przy pomocy kontroli redundancji wzdłużnej, jak np. analizy kontrolnych znaków wielomianowych,

5/ redagowanie komunikatów - w miarę potrzeby. Na przykład, operator może robić błędy, uderzając w klawisze albo cofając karetkę lub wymazać znaki. Komunikat musi zostać przeredagowany we właściwą formę, zanim będzie gotowy do przetwarzania. Znaki kontrolne muszą zostać rozpoznane i usunięte,

6/ rozpoznawanie znaków "koniec zapisu" i "koniec transmisji" i "porządkowanie" przed następną transmisją, jeśli zajdzie potrzeba. Jeśli został wykryty błąd, ten sam komunikat musi być przesłany ponownie,

7/ przekazywanie komunikatów do głównego programu, po jednym, zredagowanych i po konwersji,

8/ przyjmowanie komunikatów od głównego programu, kiedy są gotowe do przesłania,do końcówek,

9/ przygotowywanie tych komunikatów do wyjścia. Będzie konieczna konwersja ich z kodu komputera na kod linii przesyłowej. Może zajść potrzeba dodania znaków kontrolnych,

10/ inicjowanie transmisji tych komunikatów,

11/ nadzorowanie procesu przesyłania, powtarzanie znaków lub komunikatów, jeśli końcówka wykryje błąd w transmisji,

12/ sygnalizowanie końca transmisji końcówce i wykonanie koniecznych funkcji porządkujących /house-keeping/ i funkcji sterowania liniami.

Jeśli wszystkie te funkcje wykonują programy w głównym komputerze, programy użytkowe będą bardzo często przerywane i obniży się sprawność komputera. Jeśli jest wiele linii lub duży ruch danych, to obniżenie może być bardzo poważne. Dlatego też zazwyczaj część wymienionych funkcji wykonują urządzenia zewnętrzne.

Takie urządzenie może składać bity w znaki i podawać znaki do głównego komputera. Może składać znaki w słowa lub bloki i przekazywać je komputerowi. Ograniczy już to znacznie liczbę przerw w programie. Urządzenie zewnętrzne może zapamiętać cały komunikat, rozpoznać znak końca transmisji i dopiero wtedy przerwać głównemu komputerowi. W istocie urządzenie zewnętrzne może wykonywać wszystkie wymienione funkcje za pomocą logiki zaszytej, mikro-oprogramowania lub zapamiętanego programu.

Które z wymienionych rozwiązań zostanie wybrane będzie w pewnym stopniu zależało od liczby obsługiwanych linii przesyłowych. Jeśli system ma tylko jedną linię transmisyjną, może ona wchodzić prosto do komputera albo do urządzenia o pamięci buforowej, które automatycznie składa lub rozkłada cały komunikat. Jeśli jednak jest 50 lub 100 linii transmisyjnych, najlepiej jest, jeśli kończą się one w osobnym komputerze, sterującym liniami.

Funkcje przepytywania i wywoływania /polling and dialing/

Jeśli używana jest bardziej złożona konfiguracja linii transmisyjnych niż przedstawiona na rys. 2, wzrastają wymagania w stosunku do oprogramowania. System może korzystać z linii przepytywanych, linii w pętlach lub linii z wybieraniem /komutowanych/. Funkcje niezbędne dla sterowania tymi liniami są następujące:

1/ wołanie końcówek

Program może wybrać potrzebne cyfry i nawiązać kontakt z końcówką lub z innym komputerem. Jeśli dana stacja jest "zajęta", żądanie połączenia może być zapamiętane i realizowane później,

2/ przeszukiwanie końcówek komutowanych /dial-ups/.

W niektórych przypadkach grupa końcówek, przyłączonych do linii dial-up będzie przeszukiwana dla stwierdzenia, czy któraś z nich nie ma zgromadzonych danych do przesłania,

3/ odpowiadanie, gdy komputer jest wzywany.

W przypadku otrzymania wezwania od końcówki lub innego komputera, program powinien nawiązać kontakt z urządzeniem, stosując właściwy "protokół" i powinien powiadomić potrzebny program problemowy,

4/ polling przez programowanie.

Jeśli linia jest przepytywana, może to być wykonane całkowicie przy pomocy programowania. Program ma listę adresów końcówek i kolejność, w jakiej są przepytywane,

5/ polling przez urządzenie autopoll.

Jeśli używane jest zewnętrzne urządzenie przepytujące, program nie wysyła sam komunikatów przepytujących; wydaje on polecenia urządzeniu przepytującemu,

6/ sterowanie liniami pętlowymi /looped/.

Jeśli do linii pętlowej podłączonych jest kilka urządzeń przy pracy z synchronicznym strumieniem słów, program może

składać i rozbijać potrzebne składniki znaków /bits of characters/.

I znowu wszystkie te funkcje mogą być realizowane albo przez centralny komputer, albo przez osobną maszynę. W celu zapewnienia elastyczności maszyna ta może być komputerem o stałym programie, zdolnym do obsługi komunikatów dowolnej długości i zdolnym do sterowania różną liczbą linii przesyłowych. Może ona mieć repertuar rozkazów odmienny od konwencjonalnych komputerów, zaprojektowany dla obsługi linii transmisyjnych. Może mieć możność rejestrowania komunikatów w swoim własnym zbiorze z dostępem wyrywkowym lub na taśmie. Może wysyłać komunikaty w języku angielskim do operatorów końcówek, w ramach swoich procedur sterowania. Ponieważ maszyna ta daje się programować, jej procedury mogą być modyfikowane stosownie do okoliczności.

Podstawowe pakiety oprogramowania

Wymienione wyżej funkcje stanowią podstawę wymagań, jeśli chodzi o sterowanie liniami transmisyjnymi. Niektóre pakiety oprogramowania zawierają te właśnie funkcje i niewiele poza tym. Uwalniają one programistów od potrzeby składania bitów w znaki, dokonują konwersji kodów i spełniają inne wymienione funkcje. Czasami programista programów zastosowań pisze po prostu tylko makrorozkazy GET i PUT i udostępnia mu się odpowiednie kody. Przy prostych systemach nie potrzeba wiele więcej. W systemach większych lub bardziej skomplikowanych trzeba poświęcić nieco więcej uwagi sprawności organizacji zarówno sterującego komputera, jak i sieci transmisyjnej. W wyniku powstają schematy kolejek dla komunikatów, mimroprogramy na wyższym szczeblu, dynamiczna alokacja buforów, kompresja komunikatów i prowadzi to do zastosowania koncentratorów linii transmisyjnych i innych poczynań zmierzających do optymalizacji sprawności sieci.

Rozdział i harmonogramy zasobów

Troska o sprawne i pełne wykorzystanie komputerów może wieść do wprowadzenia następujących funkcji:

1/ dynamiczna allokacja buforów

Jeśli otrzymywane są i wysyłane komunikaty zmiennej długości, bufory które je przechowują, mogą być zorganizowane w rozmaity sposób. Dynamiczna allokacja pamięci buforowej w blokach może zmniejszyć całkowite zapotrzebowanie na pojemność pamięci. Bloki będą łączone między sobą zgodnie z potrzebą; skoro tylko jakiś blok zostanie zwolniony, podlega znów allokacji do "rezerwuaru" bloków do dyspozycji,

2/ obsługa kolejek linii

Na przepytywanej linii transmisyjnej, jeśli się do tego dopuści, może powstać kolejka transakcji, czekających na transmisję. W systemach prostych kolejki nie są dozwolone: tzn.program nie może wysłać na linię komunikatu, jeśli inny komunikat już oczekuje na wysłanie. Takie postępowanie może czasem wstrzymać przetwarzanie, dlatego też pożądany jest mechanizm regulowania kolejek,

3/ obsługa kolejek programów

W podobny sposób, jeśli dopuści się do tego, formują się kolejki komunikatów na wejściu, oczekujących kiedy będą wolne pewne programy. Podobny mechanizm regulowania kolejek może więc działać na wejściu i wyjściu,

4/ komunikaty dla wielu odbiorców

Komunikat może być wysyłany nie do jednego, lecz do wielu odbiorców. Aby nie zajmować niepotrzebnie miejsca jednocześnie w wielu kolejkach, powinna raczej być tylko jedna kopia takiego komunikatu. Umieszcza się wówczas w różnych obszarach sterujących kolejkami instrukcje odsyłające do treści komunikatu,

5/ harmonogramy priorytetów

Niektóre komunikaty mogą mieć pierwszeństwo przed innymi. W takim przypadku potrzebny jest mechanizm regulujący kolejki, uwzględniający różne priorytety,

6/ multiprogramowanie

Jeśli programy zastosowań muszą czekać na zakończenie operacji teletransmisyjnych, pożądane jest, aby centralna jednostka

przetwarzania mogła zająć się inną pracą. Potrzebna jest zdolność łatwego przełączania się do zadań o niskim priorytecie i z powrotem.

Funkcje dotyczące urządzeń sieci

Do sieci transmisyjnej mogą być wbudowane rozmaite urządzenia, takie jak multipleksory, koncentratory i komputery czasowo przechowujące i wysyłające komunikaty /store - and - forward computers/. Oprogramowanie może kształtować komunikaty informacyjne i sterujące w taki sposób, aby były do przyjęcia przez takie urządzenia.

1. Formowanie bloków

W niektórych systemach istnieje optymalna długość bloku do transmisji. Poszczególne zapisy lub komunikaty mogą zatem być składane razem w jeden blok. W niektórych przypadkach urządzenia sieci przyjmują tylko bloki o ustalonej długości lub określonej wielkości pakiety,lub tylko kilka określonych wielkości. Oprogramowanie przeformuje dane do transmisji w wymagany sposób, a także wyodrębni poszczególne zapisy z otrzymywanych bloków.

2. Multipleksowanie

Jeśli używany jest multipleksor, przeplatający/interleaving/ znaki lub słowa z różnych końcówek, demultipleksowanie w centralnym komputerze może być dokonywane przez software lub hardware. Funkcja odwrotna odnosi się do transmisji z komputera.

3. Kompresja danych

Szereg technik może być stosowany do kompresji przesyłanej informacji, tak, aby mogła być przesłana przy pomocy mniejszej liczby bitów. "Upakowanie" danych do transmisji oraz "rozpakowanie" po odbiorze będzie należało do oprogramowania.

4. Funkcje "store - and - forward"

Niektóre systemy przechowują komunikaty do późniejszego wysłania lub do późniejszego pobrania od końcówki. Potrzebne jest sterowanie tą operacją przez oprogramowanie.

5. Komutowanie komunikatów

Wszystkie funkcje, normalnie wykonywane przez system komutowania komunikatów, mogą zostać wbudowane w oprogramowanie.

Rejestracja i zbieranie danych statystycznych

1. Rejestrowanie komunikatów

W niektórych systemach wszystkie komunikaty rejestrowane są na taśmie lub dysku, do ewentualnego późniejszego wykorzystania.

2. Gromadzenie statystyki

Ważną, a czasem zaniedbywaną funkcją, jest gromadzenie danych o wielkości ruchu na linii, przekłamaniach na linii i uszkodzeniach.

Wznowienie pracy po uszkodzeniach

1. Uszkodzenie linii

W wypadku uszkodzenia linii lub końcówki, programy mogą zapisać informację, aby ułatwić wznowienie pracy po naprawie.

2. Kolejna numeracja komunikatów

Komunikaty mogą otrzymywać kolejne numery od oprogramowania u źródła /front-end/, aby ułatwić wznowienie pracy po uszkodzeniu. Kolejne numery mogą być podawane operatorom końcówek.

3. Pomoc w przypadku uszkodzenia centralnego komputera

Oprogramowanie komputera sterującego linią może zarejestrować dane, potrzebne dla wznowienia pracy przez centralny komputer po jego naprawieniu. Może automatycznie przesyłać informacje dla operatorów.

4. Procedury wznowienia

Powinny być napisane programy pomagające przy wznowieniu pracy po awariach. Mogą one korzystać z rejestru komunikatów oraz systemu kolejnej numeracji.

5. Diagnostyka

Prawidłowe funkcjonowanie różnych części systemu może być sprawdzane przy użyciu programów diagnostycznych w centralnym komputerze. Prawidłowe działanie końcówki i jej podłączenia do komputera może być sprawdzane przez wywołanie z terminalu programu diagnostycznego, który sprawdzi wszelkie możliwe okoliczności.

6. Krosowanie w pętli /cross-patching/ oznacza, że linia wychodząca z komputera jest "patched", "zawrócona" tak, że natychmiast wchodzi z powrotem do komputera. Robi się tak dlatego, że przez wysłanie danych i przyjęcie ich od razu z powrotem, uzyskuje się sprawdzenie czy komputer zdolny jest do prawidłowego wysyłania i odbioru. Przy użyciu takich metod można stwierdzić przyczynę złego funkcjonowania, a oprogramowanie do automatycznej realizacji tej funkcji okazuje się bardzo cenne.

Generowanie odpowiedzi na zewnątrz

1. Sterowanie jednostką zewnętrzną, która generuje odpowiedzi

Jednostka zewnętrzna może przechowywać odpowiedzi, które mają być wysłane do końcówki i wysyłać te odpowiedzi, kiedy otrzyma polecenie z komputera. Taka jednostka zewnętrzna może być w pomieszczeniu komputera, w pomieszczeniu koncentratora lub w pomieszczeniu końcówki.

2. Sterowanie jednostką zewnętrzną, która formuje dokumenty lub obrazy

Jednostka zewnętrzna może przechowywać szczegóły dotyczące formularzy, tablic lub wykresów, które mają być wydrukowane lub wyświetlone. Informacja otrzymana z komputera spowoduje, że wygenerowany zostanie odpowiedni format i wypełniony danymi. Często taka jednostka będzie się mieścić w pomieszczeniu końcówki.

3. Sterowanie jednostką odpowiadającą głosem

Jednostce odpowiadającej głosem może być podany rozkaz wygenerowania wybranych słów. Dla pewnych urządzeń komputer przesy-

ła zakodowany odsyłacz lub adres każdego słowa lub frazy. U innych urządzeń przesyłany jest łańcuszek bitów, które urządzenie odpowiadające potrafi zamienić w dźwięk, natomiast samo urządzenie nie ma pamięci ze "słownikiem".

Aspekty bezpieczeństwa

Dwa aspekty oprogramowania transmisji danych, pożądane z punktu widzenia bezpieczeństwa to:
/1/ możność zidentyfikowania bez wątpliwości końcówki, z którą komputer jest w łączności oraz /2/ możność zidentyfikowania osoby obsługującej tę końcówkę. Dalszą funkcją, rzadko dziś stosowaną, jest szyfrowanie przesyłanych danych.

1. Identyfikacja końcówki

Końcówki, zbudowane z uwzględnieniem wymogów bezpieczeństwa, posiadają możliwość przesyłania specyficznego dla nich numeru identyfikacyjnego. Komputer może sprawdzić ten numer przy otrzymywaniu zapytania. Czasem, gdy wysyłane są dane do końcówki, oprogramowanie sprawdzi najpierw identyczność urządzenia, do którego będzie transmitować.

2. Identyfikacja użytkownika

Użytkownik końcówki może być rozpoznawany na różne sposoby - najczęściej w ten sposób, że podaje zastrzeżony klucz lub hasło, sprawdzane przez komputer. Hasło zmieniane jest w nieregularnych odstępach czasu. Użytkownik może też wkładać kartę identyfikacyjną, która spowoduje, że przesłany zostanie numer użytkownika. Program sprawdza, czy dany użytkownik uprawniony jest do wykonywania operacji, jakich zażąda.

3. Szyfrowanie i rozszyfrowywanie

Dla uchronienia "wrażliwych" danych przed podsłuchem lub przypadkowym niewłaściwym skierowaniem może być stosowana kryptografia. Oprogramowanie transmisyjne może dokonywać zaszyfrowywania i rozszyfrowywania komunikatów.

4. Nadzór nad bezpieczeństwem

W przypadku wystąpienia naruszeń bezpieczeństwa oprogramowanie może automatycznie wyłączyć końcówkę i powiadomić referenta bezpieczeństwa. Wszystkie potencjalne zagrożenia powinny być przez program rejestrowane dla przyszłej analizy lub kontroli.

Wiele programów transmisji danych nie posiada jeszcze wbudowanych funkcji zabezpieczających. Będą one jednak w przyszłości nabierały na ważności.

Dialog człowiek - maszyna

Niektóre funkcje związane z dialogiem człowiek-maszyna mogą być wbudowane w pakiety oprogramowania. Te funkcje, jednakże, mogą być bardziej zorientowane na zastosowanie, niż inne, omówione wyżej.

1. Kontrola sensowności

Dla pewnych pól mogą zostać ustalone granice dopuszczalnych wartości, aby ułatwić wyłapywanie pomyłek operatorskich.

2. Kontrola kompletności

Zapis może być składany pozycja po pozycji,a następnie sprawdzany, czy jakiejś pozycji nie pominięto.

3. Przegląd danych

Pewien przegląd danych może być dokonywany bez jakiegokolwiek ich przetwarzania. Może się to odbywać przy pomocy standardowych cech oprogramowania, bez uciekania się do programów zastosowaniowych. Mogą być przeglądane zapisy zbiorów. Masa danych na wielu "stronach" może być "przerzucana" przy pomocy urządzenia ekranowego. Może być kartkowany skorowidz,itp.

4. Wstępny dialog

Ściśle zdefiniowany wstępny dialog może zostać zastosowany, zanim zostaną uruchomione programy zastosowań. Ten dialog może przejść przez serię przeglądów "menu" dla ustalenia, który z

programów zastosowań powinien być użyty. Może on zażądać serii pozycji danych od operatora, aby utworzyć zapis gotowy dla programu zastosowania.

Możliwy jest szeroki wybór takich wstępnych, przygotowawczych operacji. Część tego oprogramowania dialogowego może znajdować się w koncentratorze lub jednostce sterowania końcówkami, a nie w komputerze centralnym.

Poziomy przejrzystości

Logika wykonywania tych funkcji może być upakowana różnymi sposobami. Może być rozdzielona między różne maszyny.

Będzie stawać się coraz ważniejszą sprawa uchronienia programisty zastosowań przed skomplikowaniem sieci i końcówek. W przyszłości będziemy prawdopodobnie widzieli całe warstwy oprogramowania zaprojektowanego po to, aby uczynić operacje transmisji tak przejrzystymi, jak tylko to możliwe, podczas gdy w rzeczywistości stają się one coraz bardziej skomplikowane.

Ważne będzie uzyskanie "przejrzystości kodowej" w sieci posługującej się kodami. Ważne będzie osiągnięcie "przejrzystości końcówek" tam, gdzie jeden typ końcówki może być zastąpiony przez inny lub gdzie końcówki niekompatybilne muszą komunikować się ze sobą. Będzie ważne osiągnięcie "przejrzystości sieci" z uwagi na narastające skomplikowanie i różnorodność sieci.

Funkcje bazy danych

Omawiane dotychczas funkcje odnoszą się wszystkie do transmisji danych w systemie. Wiele pakietów oprogramowania ma także operacje bazy danych i dla tych operacji potrzebny jest bardzo odmienny zespół funkcji, które będą omówione w rozdziale V.

III. PODSTAWOWY PAKIET OPROGRAMOWANIA

W tym i w następnym rozdziale opiszemy niektóre typowe programy teleprzetwarzania.

Większość użytkowników maszyn IBM stosuje systemy operacyjne OS/360 /lub/370/, systemy operacyjne na dyskach DOS, ich odpowiedniki z pamięcią wirtualną OS/VS lub DOS/VS lub Virtual Machine Facility VM/370. Ich oprogramowanie teleprzetwarzania musi pasować do tych systemów operacyjnych. Są tu dwie możliwości "metody dostępu" dla teleprzetwarzania. Są one typowe dla podstawowego oprogramowania, dostępnego na rynku, opiszemy je więc poniżej.

Pierwsza z nich, przedmiot niniejszego rozdziału, jest to prosty pakiet oprogramowania, zapewniający podstawowe funkcje, potrzebne dla sterowania liniami telekomunikacyjnymi. Nazywany jest BTAM /Basic Telecommunications Access Method/.Aby przesłać komunikat do końcówki, programista zastosowań wpisuje do swojego programu makroinstrukcję WRITE. Aby otrzymać komunikat z końcówki, używa makroinstrukcji READ. Program ten nie steruje kolejkami transakcji, jak trzeba by było w dużym systemie z liniami wielo-odczepowymi/multidrop/ i koncentratorami; nie wykonuje również żadnej redakcji komunikatów.

Druga metoda dostępu nazywana jest TCAM /Telecommunication Access Method/ i zastępuje dawną QTAM /Queued Telecommunications Access Method/. TCAM obsługuje kolejki transakcji i posługuje się swoim własnym odrębnym programem sterującym w systemie operacyjnym dla ustalenia kolejności operacji. Aby przesłać komunikat do końcówki, programista zastosowań wpisuje do swojego programu makroinstrukcję PUT. Aby otrzymać komunikat z końcówki, używa makroinstrukcji GET. Operacje te są następnie realizowane w kolejności, jaką ustala program sterujący TCAM. Pakiet TCAM zostanie omówiony w następnym rozdziale.

BTAM realizuje funkcje opisane poniżej.

1. Wywoływanie końcówek i otrzymywanie od nich komunikatów

Jeśli w programie zastosowania jest makroinstrukcja READ,następuje odgałęzienie do podprogramu BTAM, który wywołuje wskazany terminal i poleca mu nadawanie. Po wydaniu makroinstrukcji READ, program zastosowania może biec dalej. Osiągnie on wreszcie punkt, z którego nie może ruszyć dalej, dopóki nie otrzyma

informacji, będącej wynikiem tego READ. W tym punkcie wpisana jest do programu w assemblerze normalna makroinstrukcja WAIT. Wówczas program nadzorczy systemu operacyjnego przekaże sterowanie jakiemuś innemu programowi. Kiedy spełnione zostaną warunki dla WAIT - inaczej mówiąc, kiedy komunikat zostanie przyjęty i będzie gotowy do przetwarzania - program nadzorczy przekazuje z powrotem sterowanie oczekującemu programowi zastosowania.

2. Przepytywanie /polling/

Jeśli końcówka przyłączona jest do linii przepytywanej, wynikiem makroinstrukcji READ jest przegląd linii /line is being polled/; kiedy jakaś końcówka zgłasza, że ma komunikat, zostaje on odczytany. Aby tego dokonać, program BTAM musi posiadać listę przepytywania /polling list/, mówiącą, jakie są adresy końcówek i podającą kolejność, w jakiej muszą być przepytywane. Lista ta podana jest wcześniej w programie przez makroinstrukcję, która specyfikuje kolejność końcówek i mówi, czy przepytywanie powinno odbywać się w sposób ciągły /"wrap-around"/ , czy też nie.

Dla przykładu: na linii obsługującej końcówki IBM 2740 makroinstrukcja "READ INITIAL AND RESET" da w wyniku następujący ciąg operacji WE/WY:

1/ wpisz trzy znaki EOT dla zwolnienia linii,
2/ wpisz znak przepytywania, podając adres końcówki początkowej,
3/ wpisz znak spacji /aby dać niezbędną zwłokę/,
4/ odczytaj odpowiedź z końcówki,
5/ jeśli odpowiedź negatywna, powtórz trzy poprzedzające kroki dla końcówki następnej na liście,
6/ jeśli odpowiedź pozytywna, odczytaj znaki tekstu,
7/ wpisz znak EOA, a potem trzy razy EOT dla zakończenia operacji i ponownego zwolnienia linii.

Kiedy transmisja została pomyślnie zakończona, program BTAM powiadamia program zastosowania.

3. Przekazywanie komunikatów do końcówek

Podobnie, programista zastosowań może przesłać komunikat do końcówki, przy pomocy makroinstrukcji WRITE.

I znowu, przy niektórych układach sterowania linią, konieczna jest pewna wymiana transmisji tam i z powrotem, aby przesłać do końcówki jakiś komunikat. Znów, dla przykładu, na linii z IBM 2740 "WRITE INITIAL AND RESET" wywoła następujący ciąg operacji:

1/ wpisz trzy znaki EOT dla zwolnienia linii,
2/ wpisz adres końcówki, o którą chodzi,
3/ wpisz znak spacji /aby dać niezbędną zwłokę/,
4/ odczytaj odpowiedź z końcówki, mówiącą, czy jest gotowa odbierać,
5/ jeśli końcówka gotowa, wpisz znak EOA, a potem tekst,
6/ odczytaj potwierdzenie końcówki, mówiące, czy komunikat został odebrany prawidłowo,
7/ wpisz trzy znaki EOT, aby znów zwolnić linię.

Znowu, po pomyślnym zakończeniu transmisji, kod BTAM powiadamia program zastosowania.

4. Postępowanie z błędami transmisji

Jeśli używane są kody wykrywania błędów, z automatyczmą retransmisją błędnych komunikatów, kod BTAM zainicjuje retransmisję, jeśli jest konieczna.

Jeśli po określonej liczbie prób nie uzyskano poprawnej transmisji, kod BTAM powiadomi program zastosowania, aby mógł podjąć odpowiednią akcję.

5. Organizowanie zbiorów buforowych dla komunikatów o zmiennej długości

Jeśli BTAM pracuje w ramach OS/360 / lub /370/, przydziela dynamicznie bloki buforowe, zgodnie z zachodzącą potrzebą. Jeśli otrzymywany komunikat wypełni przydzielony mu blok, następny blok zostanie przyłączony do poprzedniego. Ten proces przyłączania będzie trwał dopóki nie zostanie odebrany cały komunikat. Odwrotny proces ma miejsce przy wysyłce. Skoro tylko blok zostanie opróżniony, zostaje znów udostępniony zbiorowi wolnych bloków do przydziału na bufory.

6. Wywoływanie tarczowe końcówek i odpowiadanie końcówkom, które wywołują komputer

Jeśli końcówka jest na linii wywoływanej tarczą numerową, makroinstrukcja WRITE może spowodować, że kod BTAM dokona operacji wywołania. Makroinstrukcja "WRITE INITIAL" spowoduje, że komputer wywoła końcówkę, nawiąże połączenie i przekaże, co trzeba. "WRITE CONTINUE" spowoduje przekazanie danych bez wywoływania /ponieważ połączenie jest już nawiązane/. Jeśli z wywoływanej końcówki otrzymywany jest sygnał "zajęte", kod BTAM powiadomi program zastosowania.

Podobnie, makroinstrukcja READ spowoduje wywołanie końcówki, aby sprawdzić, czy ta ma coś do nadania. W niektórych przypadkach BTAM otrzyma listę wywołań i będzie wywoływać końcówki na liście, jak w operacji przepytywania.

Kiedy użytkownik końcówki wywołuje komputer, BTAM może zostać ustawiony na odbiór wezwania, nawiązanie łączności z końcówką i powiadomienie programu zastosowania, że otrzymano wezwanie.

7. Konwersja kodów

BTAM dostarcza procedurę tłumaczącą i komplet tablic translacyjnych dla konwersji między kodem komputera i różnymi kodami, używanymi przez końcówki. W razie potrzeby, użytkownik może ustalić nowy zbiór znaków końcówek i podać tę tablicę translacyjną podprogramom BTAM.

8. Prowadzenie statystyki błędów

BTAM prowadzi statystykę wszystkich wykrytych błędów transmisji.

9. Zapewnienie bezpośrednich usług sprawdzania końcówek

BTAM posiada bezpośrednią /on-line/ diagnostykę ułatwiającą sprawdzanie urządzeń końcówek.

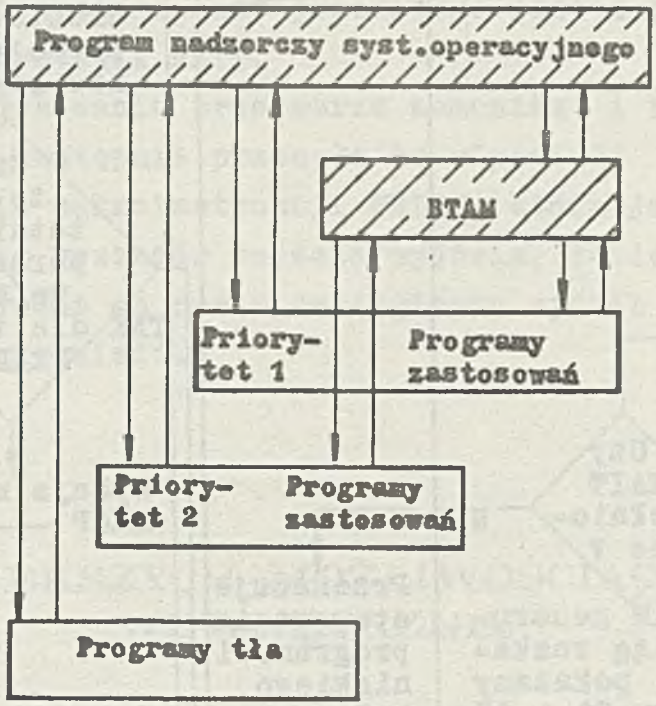
10. Inicjacja operacji

Makroinstrukcja OPEN w BTAM aktywuje grupę linii - tzn. dowolną określoną grupę linii, posiadających identyczne procedu-

ry sterowania. Musi to zostać wykonane, zanim zostanie przez te linie wysłany lub otrzymany jakikolwiek komunikat. Podobnie makroinstrukcja CLOSE zakańcza operacje. Makroinstrukcja OPEN uruchamia zbiory buforów i wykonuje program kanału, konieczny dla inicjacji jednostek sterowania teletransmisją i modemów.

Stosunek do innych programów

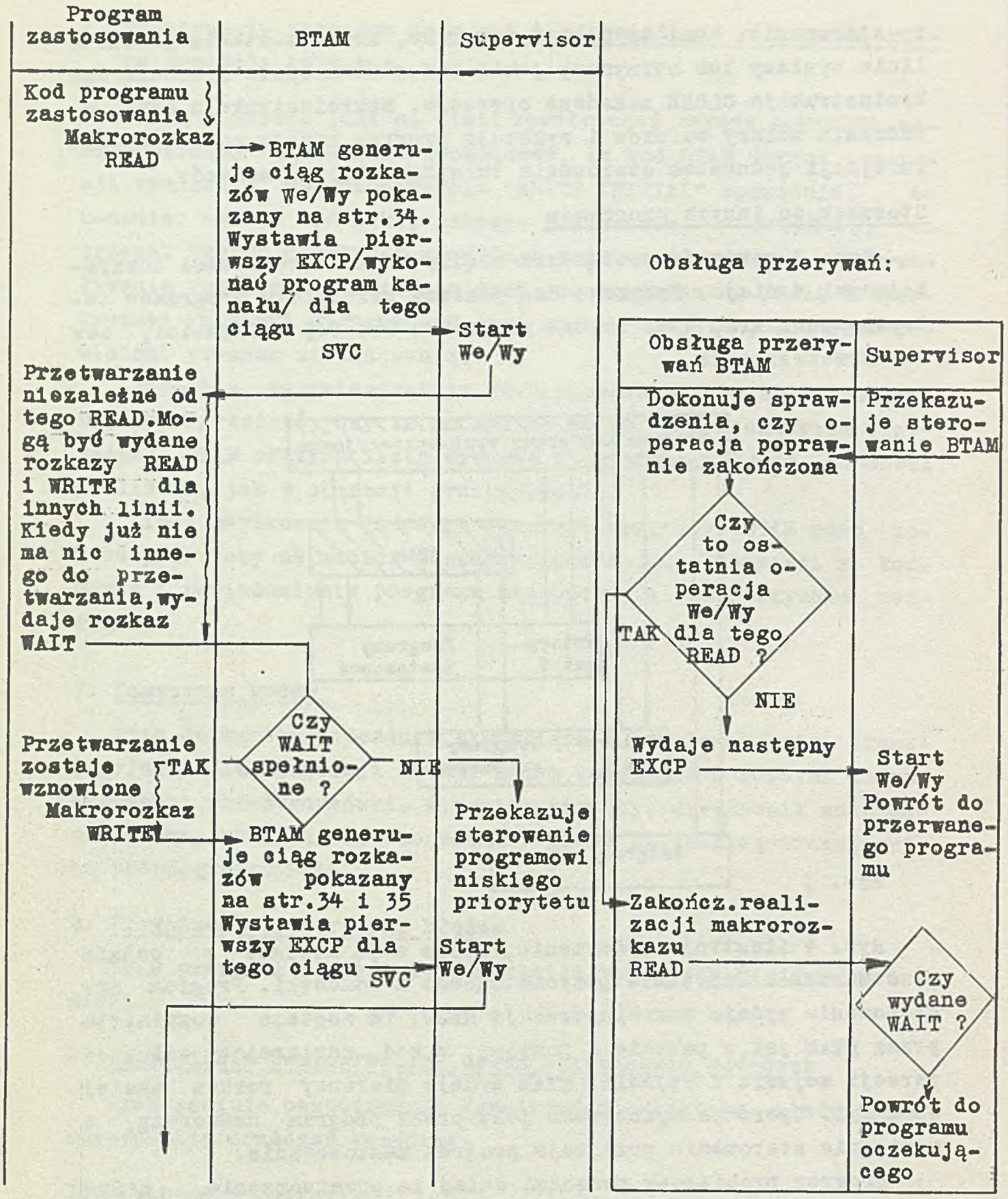
Rys. 3 pokazuje powiązania między BTAM i programami korzystającymi z niego. Pokazano dwa poziomy priorytetu programów teleprzetwarzania, wraz z tłem programów drugiej kolejności, bez teleprzetwarzania.



Rys. 3

Rys. 4 ilustruje wydarzenia, które mają miejsce w czasie przetwarzania zapytania przychodzącego z końcówki. Program zastosowania wydaje makroinstrukcję READ. Ta zostaje rozwinięta przez BTAM jak w punkcie 2 powyżej, w kod, zawierający kilka operacji wejścia i wyjścia. BTAM wydaje pierwszy rozkaz takiej operacji. Operacja wykonywana jest przez program nadzorczy, a następnie sterowanie przejmuje program zastosowania.

Program problemowy prowadzi dalej te przetworzenia, które mogą być dokonywane przed otrzymaniem wyników tego READ. Program zastosowania może wydawać makroinstrukcje READ i WRITE na



Rys. 4. Przebieg zdarzeń występujący, gdy BTAM obsługuje zapytanie przychodzące z końcówki

innych liniach, ale nie wolno mu tego robić na linii, dla której wydana została wyżej wspomniana READ. Kiedy program problemowy nie może już dalej przetwarzać, wydaje makroinstrukcję WAIT.

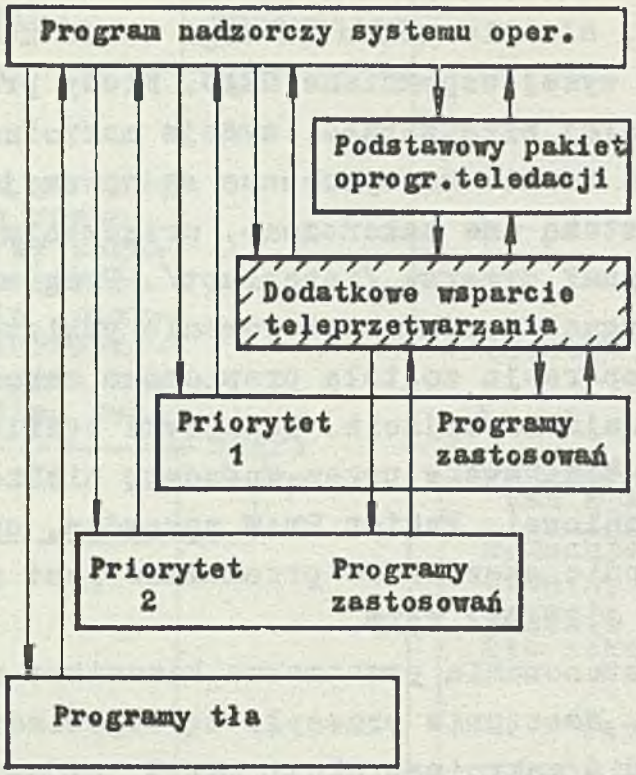
W tym samym czasie kontynuowane są operacje na wejściu/wyjściu. Kiedy zostaną one zakończone, urządzenia telekomunikacyjne przekażą sygnał przerwy /interrupt/. Program nadzorczy reagując na ten sygnał przekaże sterowanie pakietowi BTAM. BTAM sprawdzi, czy operacja została prawidłowo zakończona i wyda następny rozkaz wejścia/wyjścia. Taki cykl będzie trwał dopóty, dopóki komunikat nadawany przez końcówkę nie zostanie wczytany do pamięci rdzeniowej. Pakiet BTAM sprawdza, czy nie ma błędów linii, a następnie sterowanie przekazane jest z powrotem programowi, który wystawił WAIT.

Program zastosowania przetwarza komunikat i redaguje odpowiedź na niego. Następnie przesyła tę odpowiedź do końcówki przez wystawienie makroinstrukcji WRITE, która jest rozwijana przez BTAM w ciąg rozkazów wejścia/wyjścia, takich jak wyżej w pkt. 3. Wykonywane są one w analogiczny sposób i końcówka otrzymuje swoją odpowiedź.

IV. PRZY WIĘKSZYCH MOŻLIWOŚCIACH PAMIĘCI WEWNĘTRZNEJ

Opisane w poprzednim rozdziale wsparcie oprogramowania na poziomie BTAM czyni programowanie dla celów teleprzetwarzania łatwiejszym, ale przy skomplikowanych systemach programiście pozostaje nadal wiele trudnych problemów, które mogłoby przejąć oprogramowanie. Niektóre organizacje zdają sobie z tego faktu sprawę i opracowały swoje własne pakiety teleprzetwarzania, które pasują pomiędzy programy zastosowań i podstawowy pakiet teleprzetwarzania, jak pokazano to na rys.5. Taki pakiet może rezydować w części systemu operacyjnego o najwyższym priorytecie.

Powodami opracowania takiego pomocniczego pakietu teleprzetwarzania może być potrzeba pewnych standardowych aspektów dia-



Rys. 5

logu człowiek-maszyna, zapewnienie specjalnego traktowania błędów lub pomyłek, dalsze uproszczenie kodowania wejścia/wyjścia lub - powód szczególnie ważny - zaprogramowanie funkcji obsługi kolejek i wielokrotnych warunków WAIT w programach zastosowań, tak aby wiele wydarzeń asynchronicznych mogło następować z dostateczną sprawnością jednostki przetwarzającej. Innymi słowy - potrzeba uzyskania operacji wielowątkowej.

W niniejszym rozdziale opiszemy pakiet oprogramowania teleprzetwarzania, który jest bardziej wszechstronny niż ten, który opisaliśmy w rozdziale poprzednim. Jednak nawet przy dostępności oprogramowania wszechstronnego, będzie prawdopodobnie dość powodów w niektórych przypadkach, aby użytkownik opracował swoje własne pakiety, mieszczące się w schemacie oprogramowania jak na rys.5. Powody te będą się czasami wiązały z unikalnymi wymaganiami systemu lub potrzebami kontaktu człowiek-maszyna. W niektórych przypadkach powodami mogą być oszczędności finansowe, oszczędności na czasie lub oszczędność pamięci komputera.

TCAM

Pakietem IBM ogólnego zastosowania dla teleprzetwarzania w systemach real-time jest TCAM /Telecommunication Access Method/, który pracuje pod kierunkiem OS/360(na /360 lub /370/), lub VM/370.

TCAM będzie obsługiwał transmisję danych w systemach o wysokim stopniu multiprogramowania. W jednoczesnym przetwarzaniu może być wiele transakcji różnych typów i na różnych kategoriach sprzętu transmisji. Programista zastosowań jest odizolowany od operacji we/wy i po prostu przywołuje je w swoim programie przy pomocy makroinstrukcji, takich jak GET i PUT. Nie musi on zawracać sobie głowy skomplikowanymi rozważaniami nad układaniem kolejek i następstwa czasów.

TCAM, w przeciwieństwie do bardziej podstawowych programów teleprzetwarzania ma swój własny program sterujący,który przejmuje kierownictwo i ustala harmonogramy operacji obsługi ruchu. Przerywania i makroinstrukcje w programie zastosowania powodują przekazanie sterowania programowi sterującemu pakietu TCAM. Ten program sterujący rezyduje w części systemu pod kontrolą OS/360, OS/VS lub VM/370. Traktowany on jest w systemie jako zadanie najwyższego priorytetu. Programy problemowe są zasadniczo wykonywane w częściach o niższym priorytecie, jakkolwiek, jeśli trzeba, niektóre z nich mogą dzielić część o najwyższym priorytecie z programem sterującym.

Prace, nie biegnące w czasie rzeczywistym, mogą również być wykonywane w tym samym systemie i zazwyczaj będą zajmować części o najniższym priorytecie.

Komunikaty, docierające do programu sterującego TCAM, kierowane są przez niego do właściwego przeznaczenia, którym może być końcówka lub program zastosowania. Często potrzebna linia transmisyjna lub program mogą być zajęte i wówczas program sterujący TCAM organizuje kolejki oczekiwania na te linie lub program.

Dokonywanie połączeń i gromadzenie danych

Czasami TCAM może zająć się przychodzącym komunikatem sam, bez potrzeby przekazywania go programowi problemowemu. Należą

tu takie przypadki jak np. kiedy komunikat ma być po prostu przekazany innej końcówce lub komputerowi, tak jak w systemie komutowanym. W istocie TCAM sam wykonuje wszystkie funkcje,które można spotkać w systemie komutowanym. Jeśli komunikat z końcówki systemu komutowanego przeznaczony jest do przetwarzania przez program zastosowania, będzie to zaznaczone w nagłówku tego komunikatu.

Podobnie, TCAM może wykonywać funkcje gromadzenia danych, bez odsyłania do programów zastosowań. Operatorzy końcówek przekazują gromadzone dane, aby mogły być potem przetworzone w sposób wsadowy, podobnie jakby dziurkowali dane na kartach.TCAM albo ustawi te dane w kolejce dla określonego programu problemowego, albo też zapisze je w pamięci niskiej hierarchii, niezależnie od programu zastosowania. W tym ostatnim przypadku dane te zostaną później odczytane dla przetworzenia wsadowego przez inne programy, bez udziału TCAM.

Formaty komunikatu

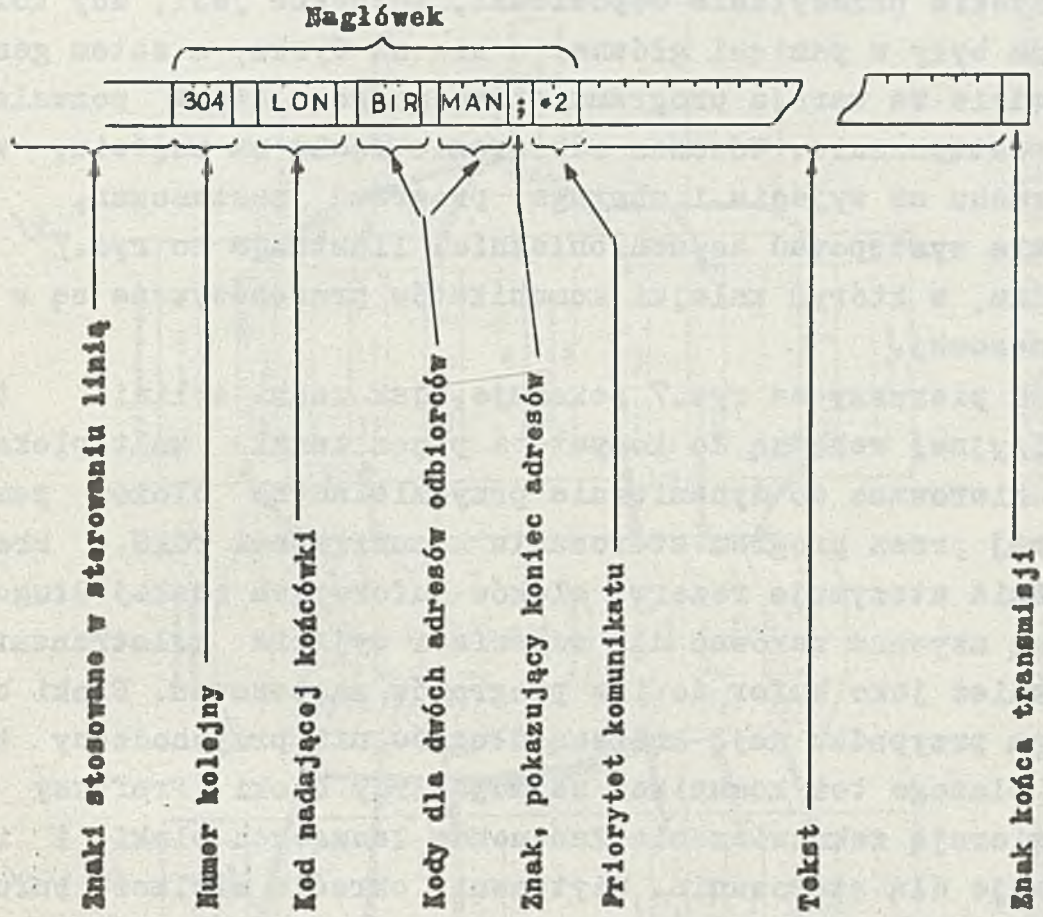
Tak więc program sterujący TCAM służy jako pośrednik pomiędzy programami zastosowań a końcówkami, a czasem między końcówkami i końcówkami,i między końcówkami a urządzeniami pamięci masowej. Końcówki mogą się różnić naturą i mogą być używane przy zastosowaniu różnych procedur sterowania liniami.

Programy problemowe nie potrzebują już zwracać się do końcówek, lecz mogą odsyłać do komunikatu używając makroinstrukcji takich jak GET i PUT.

Ze względu na rozliczne możliwości kierunkowe, niektóre z komunikatów będą potrzebowały nagłówka, z którego program sterujący TCAM będzie korzystał przy kierowniu komunikatu do właściwego miejsca przeznaczenia. Nagłówek zawiera następujące informacje:

1/ szyfr końcówki nadającej,
2/ szyfr/y/ dla miejsc/a/ przeznaczenia ,
3/ wskaźnik typu komunikatu,
4/ numer kolejny /dla orientacji przy awariach połączonych ze stratą całego komunikatu/,
5/ wskaźnik priorytetu.

Przykład komunikatu z nagłówkiem pokazany jest na rys.6. Mogą być stosowane bardzo różne formaty nagłówków.



Rys. 6. Komunikat z nagłówkiem

Wiele systemów w ogóle nie stosuje nagłówków dla komunikatów przychodzących, ponieważ wszystkie komunikaty przeznaczone są dla tego samego programu zastosowania. W innych systemach bywa używany jednoznakowy "szyfr akcji" dla wskazania właściwego programu zastosowania.

Formowanie kolejek

Komunikaty mogą przybywać o różnych porach. Często nie są one natychmiast przetwarzane, ponieważ zajęta jest jednostka przetwarzająca; a jeśli są przetwarzane, odpowiedź może nie być przesyłana natychmiast, ponieważ zajęta jest linia. Program sterujący TCAM dążąc do utrzymania wysokiego wykorzystania zarówno

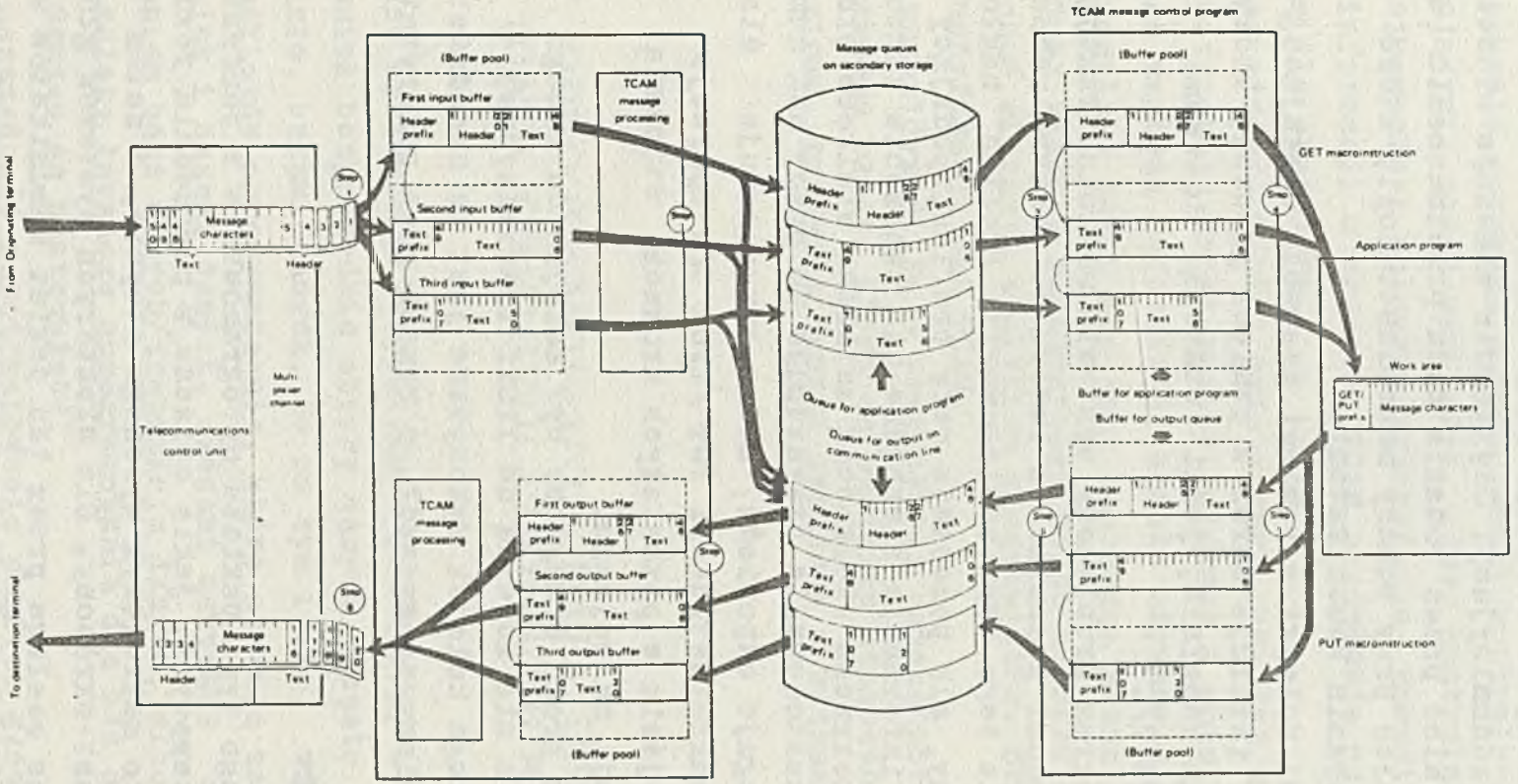
procesów jak i linii, buduje kolejki informacji przeznaczonych dla tych urządzeń. Kolejki te mogą znajdować się bądź w głównej pamięci komputera, bądź w pamięci masowej. Tam, gdzie ważne jest szybkie przesyłanie odpowiedzi, pożądane jest, aby kolejki tworzone były w pamięci głównej a nie na dysku, a zatem generowana będzie ta wersja programu sterującego, która pozwala na takie postępowanie. Wówczas odbieranie ruchu na wejściu, wysyłanie ruchu na wyjściu i obsługa programu zastosowań, będą wszystkie występować asynchronicznie. Ilustruje to rys.7[x] dla przypadku, w którym kolejki komunikatów przechowywane są w pamięci masowej.

Krok pierwszy na rys.7 pokazuje, jak znaki z linii teletransmisyjnej wchodzą do komputera przez kanał multipleksora. Są one kierowane do dynamicznie przydzielanych bloków pamięci buforowej przez program sterowania komunikatami TCAM. Program sterowania utrzymuje rezerwę bloków buforowych stałej długości, które są używane zarówno dla wejścia i wyjścia teletransmisji, jak również jako bufor do i z programów zastosowań. Bloki buforu w tym przypadku mają krótszą długość niż przychodzący komunikat. Dlatego też komunikat zajmuje trzy bloki. Prefiksy bloków zawierają zakotwiczenia łańcuchów łączących bloki i inne informacje dla sterowania. Użytkownik określa wielkość buforów, które chciałby mieć przydzielone przez program sterujący. Sposób obliczania optymalnej wielkości takich buforów podany jest w źródle bibliograficznym /7/.

Kiedy bufor się zapełni, TCAM rozpoczyna przetwarzanie danych. Może on dokonywać konwersji kodu i prostych form redagowania, takich, jak usuwanie zbędnych spacji i skasowanych znaków. Może on sprawdzać numer kolejny, ustawiać licznik komunikatu, dodawać informacje o czasie i dacie,i wykonywać inne tego rodzaju funkcje, wskazane przez użytkownika.Może zarejestrować komunikat na taśmie. Po wykonaniu tych funkcji /krok 2 na rys.7/ TCAM wpisuje komunikat w kolejce do pamięci na dysku, w tym przypadku dla określonego programu problemowego.

---

x/ Rys.7 z uwagi na trudności techniczne odbito bezpośrednio z oryginału /przyp.red./



Rys.7 Stosowane przez TCAM formowanie buforów i kolejek w pamięci roboczej

Komunikat ten, w systemie komutowanym mógłby zostać umieszczony w kolejce do przekazania na linię transmisyjną. Każda taka kolejka należy do osobnego programu zastosowania lub określonej linii transmisyjnej. Użytkownik wskazuje właściwy program zastosowania albo przez podanie kodu przeznaczenia w nagłówku komunikatu, albo przy pomocy makroinstrukcji, które generuje program sterowania komunikatami.

Dla każdej kolejki w pamięci masowej, czekającej na program zastosowania, istnieje bufor w pamięci głównej, który program sterujący TCAM zapełnia, jeśli istnieją dostępne komunikaty /krok 3 na rys.7/.

Kiedy program problemowy wystawia makroinstrukcję GET /krok 4/, TCAM przenosi komunikat z buforu do obszaru roboczego przewidzianego dla tego typu zastosowania. Prefiks nagłówkowy zostaje wymazany, ale pozostaje mały prefiks, podający wielkość komunikatu i informacje niezbędne dla tworzenia łańcuchów, gdy komunikat przekracza wielkością ustaloną wielkość bloku.

Program zastosowania przystępuje do pracy nad komunikatem i w wyniku redaguje odpowiedź, którą przygotowuje dla wyjścia przy pomocy makroinstrukcji PUT /krok 5/. Operacje formowania kolejek na wyjściu w pozostałych krokach stanowią odwrotność tychże operacji na wejściu.

Kolejka w kroku 6 mogłaby być kolejką do innego programu zastosowana, a nie kolejką do linii transmisyjnej.

Cała obsługa kolejek wykonywana jest na zasadzie: pierwszy przyszedł - pierwszy wyszedł, w ramach grup priorytetowych.

Zalety i wady

Zaletą tego typu pakietu oprogramowania w porównaniu z pakietami podstawowymi, jakie opisano w poprzednim rozdziale, jest to, że jest to makrojęzyk wyższego stopnia, a zatem czas programowania jest skrócony. Dla niektórych systemów główną zaletą jest kompletna obsługa przez ten pakiet komunikatów w systemach komutowanych lub też funkcje z zakresu gromadzenia danych. W wielu systemach pracujących w czasie rzeczywistym pierwszorzędne znaczenie może mieć czas reakcji /odpowiedzi/ i wówczas całe

formowanie kolejek komunikatów dokonywane będzie w pamięci głównej. Programy zastosowań mogą znajdować się w tej samej części pamięci głównej, co programy TCAM.

TCAM dysponuje gotowymi podprogramami kontrolnymi, rejestrującymi, datującymi, podającymi czas, numer kontrolny, podprogramami przejmowania i dalszego przesyłania komunikatu oraz transmisji informacji o błędach. TCAM przewiduje osobną końcówkę nadzorczą dla operatora.

Z drugiej strony, pakiet podstawowy, taki jak w poprzednim rozdziale, potrzebuje mniej pamięci głównej i mniej pamięci masowej. Z tego względu najczęściej będzie używany dla małych systemów, szczególnie jeśli nie jest konieczna praca wielowątkowa. Przy pomocy pakietu podstawowego można uzyskać większą elastyczność, choć większym wysiłkiem programistów. BTAM jest łatwiejszy do modyfikacji niż TCAM.

Należy przewidywać, że w przyszłości oszczędność siły roboczej i niezawodna praca, które są wynikiem stosowania pakietów wyższego poziomu, przeważą koszty związane z większymi narzutami.

V. OPROGRAMOWANIE DATA BASE (DATA COMMUNICATIONS/DBDC)

Większość systemów o bezpośrednio podłączonych końcówkach posiada również bezpośrednie zbiory danych, często nazywane, nieco dowolnie, bazami danych. Co za tym idzie, wzrasta wykorzystanie pakietów oprogramowania, które sterują zarówno przekazywaniem jak i bazami danych. Czasami nazywane one bywają oprogramowaniem bazą danych/transmisja danych, albo DBDC.

Główne zalety łącznego oprogramowania dla baz danych i dla transmisji danych są następujące:

1/ znaczna oszczędność zapotrzebowania pamięci głównej lub oszczędność czasu pracy procesora, gdy stosowana jest pamięć wirtualna,

2/ kombinowane opakowanie może być tak dobrane, aby zaspokoić określone wymagania co do czasu odpowiedzi /reakcji/, jak również co do przepustosowści,
3/ możliwości w zakresie punktów kontrolnych oraz restartu mogą być zaplanowane w sposób zintegrowany,
4/ nieco uproszczona może zostać praca programistów systemów.

Główna niedogodność wynika z niebywałej różnorodności wymagań różnych użytkowników, a nawet, u tego samego użytkownika, dla różnych zastosowań. Żadnemu pakietowi DBDC nie udało się zapewnić wszystkim wszystkiego, a poza tym, przy obecnym stanie rzeczy, a szczególnie bieżącym sprzęcie pamięci masowej, każdy taki pakiet byłby z konieczności wysoce niewydajny. Jedno zastosowanie może, na przykład, wymagać dużej przepustowości dla przetwarzania wsadowego zbiorów seryjnych. Inne zastosowanie może być dominowane przez wymagania szybkiej reakcji dla nieregularnych transakcji. Trzecie znów może wymagać bardzo niskiej przepustowości, ale za to korzystać ze zbiorów o skomplikowanej strukturze z całymi sieciami wskaźników /asocjacji/ dla odpowiadania na zapytania. Dla zaspokojenia tych trzech typów wymagań można zastosować bardzo sparametryzowany pakiet,którego nie uda się zoptymalizować dla żadnego z tych trzech typów zastosowań, a za to wystąpi degradacja ogólnej sprawności i zwiększone narzuty. Albo też, użytkownik może dobrać sobie pakiet DBDC idealny dla danego zastosowania, którego sprawność przy innym typie zastosowania będzie bardzo niska lub też jego logika manipulowania zbiorami będzie nie do przyjęcia. Czasami dla spełnienia wymagań określonej instalacji potrzebny będzie więcej niż jeden pakiet oprogramowania. Tym niemniej, posługiwanie się pakietami DBDC szybko się rozpowszechnia, ich wydajność i ogólna zastosowalność ulegają poprawie, jak również rośnie wybór na rynku.

Ponieważ żaden system DBDC nie zaspokaja wszystkich wymagań, które są teoretycznie pożądane, użytkownik zmuszony jest iść na kompromis. Jakie cechy są najważniejsze dla jego konkretnych potrzeb?

To sprawozdanie nie usiłuje szczegółowo omówić wymagań stawianych bazom danych, ponieważ naszym przedmiotem jest transmi-

sja danych. Ponieważ jednak, wielu użytkowników wybiera dla zaspokojenia swoich potrzeb w zakresie transmisji danych pakiet DBDC, w rozdziale tym rozważymy główne kryteria w odniesieniu do bazy danych.

Ponieważ trudno jest, aby jeden pakiet DBDC zaspokoił wszystkie teoretycznie pożądane wymagania, każdy z nich stara się podkreślić jedną z następujących zalet:

1/ systemy do obróbki seryjnej bezpośredniej /on-line batch systems/, których końcówki używane są do zdalnego wprowadzania zadań,

2/ systemy DBDC, w których główny nacisk położony jest na sterowanie teleprzetwarzaniem, a operacje na zbiorach są stosunkowo proste /na przykład CICS firmy IBM/,

3/ systemy DBDC, kładące główny nacisk na bazę danych - jej organizację, optymalizację, przystosowalność i brak szkodliwej redundacji - i w których teleprzetwarzanie ma znaczenie drugorzędne /np. IMS Version II IBM-u,system 2000 firmy MRI,IDS firmy HIS/,

4/ systemy informowania kierownictwa /Executive interrogation systems/, w których głównie chodzi o uzyskiwanie informacji lub generowanie sprawozdań z bazy danych przez końcówkę, bez programowania. Istnieje język dla zapytywania lub przeszukiwania zbiorów, dostosowany do określonych wymagań. Konstrukcja programu nie koncentruje się na dużej przepustowości. Zadaniem systemu jest zaopatrzenie kierownika lub użytkownika nieprogramisty w szybką informację, mimo że nie można przewidzieć w szczegółach, jakiej informacji będą oni żądać.

Byłoby dużą zaletą, gdyby wszystkie te możliwości mogły występować w tym samy systemie. Można zresztą przewidywać, że w przyszłości ujrzymy łączenie możliwości różnych pakietów oprogramowania. W najbliższym czasie jednak całkowita komasacja jest mało prawdopodobna, a więc ten, kto kupuje system musi jasno zdawać sobie sprawę, na co chce położyć nacisk. Należy zauważyć, że firmy opracowujące pakiety DBDC,zaczynają już kombinować wiele różnych cech w pojedynczym opakowaniu i oferują wyniki swej pracy w postaci modułów do wyboru, tak aby użytkownicy o różniących się żądaniach mogli sobie dobrać odpowiednie kombinacje.

Na przykład IBM ogłosił ostatnio połączenie TICS z modułem DL/I systemu INS[x/] i planuje również wypuszczenie pierwszej wersji języka zapytywania dla kierownictwa zwanego IQF /interactive query facility/ ze swoim IMS II; również MRI ostatnio wypuściła swój własny język zapytań do zastosowania z systemem 2000.

Oprogramowanie DBDC zazwyczaj ma trzy mechanizmy sterowania, w przeciwieństwie do jednego w czystym oprogramowaniu do transmisji danych. Pierwszy, to program sterujący transmisją danych /lub siecią transmisji/. Drugi, to program sterowania zbiorami, który obsługuje dostęp do bazy danych. Trzeci, jest to mechanizm sterujący wczytywaniem programów zastosowań i manipulowaniem obszarami roboczymi.

Każdy z tych trzech mechanizmów sterujących posiada zapisaną informację, odnoszącą się do jego działania. Dla programu sterowania siecią zanotowana jest pełna informacja o wszystkich urządzeniach sieci i program posługuje się "tablicą przepytywania" lub "tablicą kolejności obsługi" dla ustalenia kolejności, w jakiej należy kontaktować się z poszczególnymi urządzeniami sieci. Mechanizm wczytywania programów potrzebuje biblioteki programów. Programowi sterowania zbiorami potrzebne są opisy wszystkich typów danych. Opisy te są w różnym stopniu skomplikowane - od prostych spisów zawartości tomów /VTOC w terminologii IBM/ do biblioteki opisu danych, wyszczególniającej wszystkie elementy /pola/ danych i pokazującej zależności między nimi, zależności które stają się bardzo skomplikowane w przypadku zintegrowanych baz danych o wielorakich hierarchiach i/lub sieciach wskaźników. Biblioteka opisu danych jest podstawową częścią składową pakietów DBDC, których zadaniem jest obsługa wielu powiązanych ze sobą zbiorów.

W dodatku,do mechanizmów sterowania pakiety oprogramowania dostarczają języki zarówno dla analityków systemów, jak i dla użytkowników /patrz rozdz. VIII i IX/, a także szereg programów pomocniczych i usługowych. Dla kategorii czwartej - systemów informowania kierownictwa - język dla użytkownika ma pierwszorzęd-

---

x/ Chyba IMS? /przyp.red./

ne znaczenie. Dla kategorii trzeciej,ważne są języki opisu danych /rozdz.IX/. Podobnie, jak przy oprogramowaniu dla transmisji danych, niektóre z podstawowych operacji obsługi bazy danych mogą mieć swoją rezydencję w różnych składnikach systemu, na przykład w:

1/ centralnej jednostce logicznej,
2/ peryferyjnym komputerze obsługi zbiorów,
3/ logice kanałów,
4/ pamięci masowej.

I znowu operacje te mogą być wykonywane przez software, hardware lub mikrooprogramowanie /"firmware"/.

Podstawowe funkcje, wykonywane przy obsłudze zbiorów,są zawarte w "metodach dostępu" systemu operacyjnego.Należą do nich:

1/ operacje we/wy: uruchom /mechanizm dostępu/, przejrzyj /aby znaleźć żądany zapis/, czytaj, pisz,

2/ kontrola we/wy: kontrola zapisu może być wykonywana przez odczytywanie danych, potem jak zostały zapisane. Kontrola czytania powinna być wykonywana przez kod wykrywania i korekcji błędów. Dane błędne mogą być korygowane automatycznie,

3/ adresowanie: jest to centralny problem związany z dostępem do urządzeń bezpośredniego dostępu. Klucz, taki jak nazwa, numer konta lub odsyłacz symboliczny, musi zostać zamieniony na adres maszyny. Jest to osiągalne przy pomocy wielu różnych technik,

4/ gospodarka pamięcią główną: dynamiczna alokacja buforów lub jakiś prostszy schemat przydzielania pamięci,

5/ blokowanie i rozblokowywanie zapisów: od postaci fizycznej do logicznej i vice versa,

6/ obsługa kolejek: kolejki do kanałów, urządzeń we/wy oraz do obsługi przez programy główne; mogą być stosowane schematy priorytetu,

7/ kompilacja instrukcji programisty: GET/PUT. Programisty zastosowań nie powinny obchodzić komplikacje adresowania lub ustawiania w kolejki. On po prostu pisze GET i PUT i inne makroinstrukcje w języku wyższego rzędu, a te są tłumaczone na odpowiednie akcje.

Pakiety gospodarki bazy danych zawierają inne grupy funkcji, stanowiące dalszą pomoc dla programisty i umożliwiają uwzględnianie bardziej skomplikowanych zależności między danymi. Do tych grup funkcji należą następujące:

1/ bardziej skomplikowane zależności:

A. Adresowanie wg kluczy wielokrotnych

Jednym z najbardziej krytycznych problemów w niektórych systemach bazy danych jest adresowanie wówczas, gdy w każdym zapisie jest więcej niż jeden klucz. Niektóre zbiory muszą być tak skonstruowane, aby do danych prowadziły różne ścieżki. Wyszukiwanie wg kluczy wielokrotnych jest znacznie bardziej skomplikowane niż wg jednego klucza. Na przykład, jeśli dane mają tylko jeden klucz, są one często organizowane seryjnie wg tego klucza. Dane o dwóch kluczach nie mogą być organizowane seryjnie wg obu, tylko potrzebują kompletnego indeksu wtórnego /skorowidza/. Dane o kluczach wielokrotnych potrzebują zazwyczaj wielokrotnych indeksów. /Ideałem lub celem ostatecznym jest traktowanie każdego elementu danych jako potencjalnego klucza/.

B. Zbiory inwertowane

Zbiór inwertowany posługuje się adresowaniem według wielokrotnego klucza w ten sposób, że stosuje osobny indeks dla każdej pozycji, która została z góry ustalona jako klucz wyszukiwania. Niektóre systemy DBDC, zaprojektowane dla udzielania szybkich odpowiedzi na pytania nieprzewidzianej formy, są zbudowane głównie ze zbiorów inwertowanych /na przykład TDMS firmy System Development Corporation/.

C. Zbiory hierarchiczne

Niektóre struktury zbiorów mają wiele poziomów. W zbiorze części produkowanych, na przykład, wyrób składa się z zespołów, które z kolei składają się z podzespołów, itd. Dla każdego z tych poziomów potrzebne są zapisy, a struktura bazy danych musi wskazywać co do czego należy. Niektóre problemy związane ze strukturami hierarchicznymi są nadzwyczaj skomplikowane i wyma-

gają bardzo pomysłowej budowy zbiorów i programowania, np.krzyżowania między drzewami[x].

D. Łączniki między poszczególnymi zbiorami

Między poszczególnymi zbiorami mogą być łączniki logiczne. Jednym z powodów szerokiego stosowania takich łączników jest chęć zmniejszenia redundancji w bazie danych. Tam, gdzie używane są odrębne zbiory, bez łączników między nimi, ta sama informacja może znajdować się w różnych zbiorach, do różnych użytków. Bywa często aktualizowana z różnych źródeł, w różnych cyklach czasu, stwarzając w ten sposób niezgodności logiczne między jednym a drugim zbiorem, które muszą być rozwikłane w środowisku bazy danych. Drugim powodem są koszty związane z pamięcią nadmiarową tam, gdzie nawet przy stale obniżającym się koszcie jednostkowym zapamiętywania danych, kosztów redundancji pamięci dałoby się uniknąć.

2. Upakowywanie zbiorów

Pakiet DBDC może usiłować osiągnąć najbardziej wydajne wykorzystanie przestrzeni przez techniki takie, jak: /a/ eliminowanie redundancji danych - jakkolwiek w większości eliminacja redundancji jest pracą ludzką, odręczną; /b/ blokowanie i rozblokowywanie; /c/ redagowanie danych, aby uzyskać ich upakowanie /przy używaniu masek redagujących tylko po wyszukaniu/ oraz /d/ kompresja danych przez powtórne zakodowanie.

Redakcja i kompresja danych przez oprogramowanie może zredukować większość zbiorów danych handlowych do około 1/3 ich objętości. Te same techniki mogą być stosowane przy transmisji danych, po to, aby efektywnie zwiększyć szybkość transmisji. Techniki te jednakże są słabo rozumiane lub nieznane dla większości analityków systemów i dlatego, jak dotąd, rzadko stosowane. Wiele pakietów zaczyna oferować możliwości redakcji i kompresji i użytkownicy bardzo dużych zbiorów powinni się dobrze zapoznać z tymi możliwościami.

---

[x] w oryginale: "crossing between trees" /przyp.red./

3. Zabezpieczenie przed awariami

a. Oprogramowanie powinno zapewniać rejestry transakcji i wydruki zbiorów oraz środki konieczne dla odtworzenia zbiorów z rejestrów i wydruków w przypadku masowego zniszczenia zbiorów.

b. W celu utrzymania systemu pracującego w czasie rzeczywistym w ruchu, na wypadek gdy zawiedzie jednostka pamięci masowej, potrzebne jest zapewnienie urządzeń rezerwowych oraz procedur odzysku informacji[x/].

c. Punkty kontrolne i punkty restartu[xx/] potrzebne są dla wznowienia pracy po awarii komputera.

d. Potrzebna jest kontrola dokładności, aby zapewnić, że transakcje przybywające z końcówek nie zostały zagubione lub też nie aktualizowały przypadkowo jakiegoś zbioru dwukrotnie, po awarii linii lub podobnej.

4. Bezpieczeństwo

Bezpieczeństwo danych w systemie DBDC jest bardzo ważne i dotychczas rzadko kiedy było właściwie zagwarantowane. Powinny być stosowane zabezpieczenia programowe, uniemożliwiające osobom nieupoważnionym dostęp do elementów danych. Niektóre programy umożliwiają zabezpieczenie poszczególnych pól lub grup pól. Niektóre zabezpieczają poszczególne zapisy lub grupy zapisów. Inne jeszcze stosują zabezpieczenie tylko do całych zbiorów i nie jest to podejście właściwe. W wielu przypadkach stosowane zabezpieczenia są łatwe do obejścia. Można porównać je do typowego zamka w szufladzie biurka, ale nie można ich uważać za zabezpieczenie przed włamywaczem. Aby było istotnie efektywne, bezpieczeństwo musi być wbudowane w nierozdzielny sposób, w system operacyjny i oprogramowanie DBDC.

5. Reorganizacja i strojenie

Dynamiczna baza danych musi być periodycznie reorganizowana, jeśli ma zapewnić oczekiwaną sprawność. Jest to szczególnie

---

x/ "fallback and recovery"
xx/ "checkpoint and restart capability"

ważne przy ruchliwych zbiorach o licznych zapisach dodawanych i skreślanych. Skreślenia zazwyczaj pozostawiają "dziury" do wypełnienia, a dodatkowe zapisy umieszczane są w miejscach suboptymalnych, dopóki nie nastąpi reorganizacja zbiorów lub indeksów.

Periodyczna reorganizacja zbioru należy do normalnej gospodarki związanej z bazą danych. Może być również pożądana przebudowa bazy danych co pewien okres czasu. Budowa bazy wymaga zmian wówczas, gdy sposób korzystania z bazy staje się bardziej zrozumiały. Ma to na celu polepszenie sprawności bazy. Proces ten bywa czasem nazywany "strojeniem" bazy danych[x/].

Oprogramowanie bazy danych powinno ułatwiać zarówno reorganizację, jak i strojenie.

6. Zbieranie danych statystycznych

Dla zdobycia niezbędnych danych potrzebnych do strojenia bazy powinny być stale gromadzone informacje o tym, jak system i zapisy są użytkowane. Informacje te będą wykorzystane do rozważenia możliwych ulepszeń, zarówno w budowie zbiorów, jak i w budowie sieci transmisyjnej. Dostępne są monitory sprzętowe do gromadzenia takich danych statystycznych, te jednak zazwyczaj nie dostarczają wystarczających szczegółów co do tego, jakie zapisy i programy są używane. Niektóre pakiety oprogramowania przewidują funkcję gromadzenia statystyki. Informacje potrzebne do strojenia i reorganizacji danych powinny zawierać częstotliwość dostępów i aktualizacji, liczbę dodatków i skreśleń itp.

7. Język opisu bazy danych

Kilka pakietów gospodarki bazą danych zapewnia języki, którymi może posługiwać się analityk systemów do opisu elementów danych i łączników logicznych między nimi. Opisy w tych językach używane są wówczas, gdy skomplikowane są programy korzystające z bazy danych. Opisy te mogą być poddane rekompilacji wtedy, gdy dokonywane są zmiany w strukturze danych i w ten

---

x/ "tuning"

sposób mogą odizolować programistę zastosowań od zmian wprowadzonych do struktury danych. Omawiamy je bliżej w rozdziale VIII.

8. Języki zapytywania /Interrogation Languages/

Użytkownicy końcówek mogą korzystać ze specjalnych języków do zapytywania lub aktualizacji bazy danych. Języki te zostaną szerzej omówione w rozdziale IX.

VI. ROZDZIAŁ WEJŚCIA/WYJŚCIA LOGICZNEGO OD FIZYCZNEGO

Użytkownicy komputerów powinni być przygotowani na poważne zmiany w budowie i działaniu podsystemów teleprzetwarzania i podsystemów obsługi baz danych. Zmiany te nastąpią w ciągu najbliższych pięciu lat i prawdopodobnie będą następować przez wiele lat potem. Pożądane jest wybieranie takiego oprogramowania, które, jak dalece to możliwe, zabezpieczy nakłady, jakie użytkownik poniesie na programowanie, przed wpływem poważniejszych zmian technologicznych. Dla większości użytkowników byłoby po prostu zbyt kosztowne, aby przepisywać programy zastosowań wówczas, gdy zmieniają się struktury baz danych lub dostępne usługi telekomunikacyjne.

Aby zapewnić użytkownikowi tę ochronę oprogramowanie we/wy powinno dążyć do tego, aby jakiekolwiek przyszłe zmiany w teleprzetwarzaniu lub organizacji bazy danych były "przejrzyste" dla programisty zastosowań. Ujmując to innymi słowami, chodzi tu o rozróżnienie między fizyczną strukturą danych, które są zapisywane lub przesyłane i ich "wyglądem logicznym", który widzi programista. Kiedy programista wysyła dane wyjściowe lub odbiera dane wejściowe za pomocą instrukcji PUT i GET lub równoważnej, dane, które adresuje, występują w formie właściwej dla danego zastosowania. Nazywa się to "logiczną strukturą danych". Dane te mogą być przesyłane lub przechowywane w odmiennej formie fizycznej, na przykład kilka zapisów logicznych mo-

gło zostać połączonych w jeden zapis fizyczny, mogło ulec zmianie kodowanie znaków, mogły zostać dodane znaki kontrolne, odsyłacze do pamięci wirtualnej zostały zamienione w odsyłacze do pamięci rzeczywistej itd. Zadaniem oprogramowania we/wy jest dokonywanie konwersji między danymi "logicznymi" i "fizycznymi".

Dla ochrony nakładów na programy zastosowań należy wybierać oprogramowanie, w którym możliwa jest zmiana przesyłanych lub zapamiętywanych danych fizycznych bez zmiany programów zastosowań. Programy zastosowań używane przy teleprzetwarzaniu, dla przykładu, powinny być niewrażliwe na zmianę szybkości linii, układu sieci, przejście z sieci multipleksorowej na sieć koncentratorową itd. Programy problemowe, używane z bazą danych, powinny być niezależne od zmian w metodzie adresowania zbiorów, przejścia do innego sprzętu pamięci masowej,wprowadzenia wspólnych danych dla równolegle biegnących programów itd. W transmisji danych, w przechowywaniu danych, powinna być możliwa zmiana metody blokowania lub kodowania znaków, jak również wprowadzenie systemu kompresji danych, bez potrzeby zmiany kodu zastosowania.

Rozdzielenie operacji logicznych i fizycznych jest stosunkowo łatwe do osiągnięcia na łączu transmisyjnym od punktu do punktu. Trudne jest do osiągnięcia w systemie bazy danych o skomplikowanej organizacji strukturalnej, obsługującej licznych użytkowników. A jednak, dla tego właśnie ostatniego systemu, okaże się ono najbardziej wartościowe. Oprogramowanie baza danych/transmisja danych zawdzięcza swoje skomplikowanie głównie temu właśnie rozdziałowi między tym, jak dane wyglądają dla programisty, a tym jaka jest rzeczywista ich struktura fizyczna. Zagadnienie to może stać się szczególnie zawiłe wówczas, gdy dotyczy stosunków wzajemnych między różnymi bazami danych lub zespołami danych. Fizyczna struktura danych może zawierać zawikłaną sieć wskaźników i łańcuchów, która musi być utrzymywana, jeśli elementy danych mają być dodawane lub wymazywane. Programista zastosowań może nie widzieć tych wskaźników lub może widzieć znacznie prostszy zestaw, zorientowany na jedno zastosowanie - jego własne. Innymi słowy, logiczna struktura danych jest znacznie prostsza niż fizyczna struktura danych.

Kiedy fizyczna struktura danych się zmienia, co może się często zdarzać w kompleksowym systemie danych, logiczna struktura danych musi pozostać ta sama.

Oddzielenie prostego logicznego obrazu danych od rzeczywistości fizycznej ma trzy ważne zalety. Po pierwsze, jak już mówiliśmy, umożliwia to zmianę urządzeń peryferyjnych, sieci i bazy danych bez konieczności przepisywania programów zastosowań i ponoszenia związanych z tym kosztów. Po drugie, ponieważ logiczny obraz danych jest prosty - uproszczone są programy zastosowań. W środowisku skomplikowanej bazy danych uproszczenie to może być dość znaczne. Po trzecie, ponieważ programiści zastosowań nie mają styczności ze skomplikowaną strukturą sieci teleprzetwarzania oraz bazy danych, stosowanie struktur sieci i bazy danych o wysokim szczeblu złożoności jest całkowicie uzasadnione, o ile wyda się korzystne.

Tendencje przyszłości będą prawdopodobnie jeszcze bardziej uwypuklać te zalety, ponieważ skomplikowanie struktur sieci teleprzetwarzania i bazy danych będzie coraz bardziej wzrastać. Jedynym praktycznym sposobem budowy takich skomplikowanych systemów jest dzielenie ich na stosunkowo proste, odrębne moduły - i to musi stanowić cel, do którego dążyć powinno oprogramowanie.

Jednak za podział skomplikowanych systemów fizycznych na stosunkowo proste systemy "logiczne" trzeba płacić. Po pierwsze wzrosną koszty sprzętu i jego eksploatacji z uwagi na wymagania dużej pamięci operacyjnej i czas pochłaniany przez cykle jednostek przetwarzania. Po drugie oprogramowanie z konieczności będzie skomplikowane i jeśli nie zostanie szczególnie sprytnie zaprojektowane, wymagania, jakie postawi przed programistami systemu, którzy mają go wdrożyć i konserwować, będą olbrzymie. Już przy dzisiejszym oprogramowaniu DBDC płaci się za jedno i drugie, i na jedno i drugie się narzeka. Niektóre z tych pakietów wymagają wielkiej pamięci wewnętrznej, a praca nad ich wdrożeniem i konserwacją jest tak uciążliwa, że wielu użytkowników nie wykorzystuje w pełni zalet tych pakietów.

Trzeba sobie odpowiedzieć na pytanie: czy to warto? Czy użytkownik powinien płacić za dodatkową pamięć, jakiej wymaga

pakiet do uproszczenia i oddzielenia programów zastosowań albo za dodatkowy czas maszyny, wynikający ze stosowania systemu pamięci wirtualnej? Wielu dużych użytkowników zdecydowało,że jednak muszą tę cenę zapłacić. Stosują oni oprogramowanie DBDC na najwyższym poziomie i okazuje się to korzystne choćby tylko ze względu na zmniejszenie przyszłych kosztów konserwacji. Dla małych użytkowników cena wydaje się często zbyt wysoka. Często dowodzą oni, że są w stanie obejść się prostszą strukturą zbiorów i teleprzetwarzania, niż struktura potrzebna użytkownikowi dużemu, a zatem wystarczy im oprogramowanie prostsze. Może to być, jednakże, złudzeniem. Małe przedsiębiorstwo może pozostawać w tyle za dużą firmą w rozwoju przetwarzania danych,z czasem jednak i u niego powstanie potrzeba bazy danych dzielonej między wieloma programami zastosowań i zdalnymi użytkownikami. Zbiory mogą być mniejsze, ale zależności między nimi równie skomplikowane. Na przykład, małe linie lotnicze potrzebują równie skomplikowanego systemu rezerwacji miejsc, jak duże. A jednak musiały czekać dopóki nie stało się dostępne kompletne oprogramowanie pakietu zastosowań. Podobnie, inne typy mniejszych firm będą potrzebowały systemów bezpośrednich, odznaczających się większością tych samych cech, których potrzebują ich więksi odpowiednicy. Mogą oni jednak czekać z ich wdrożeniem, dopóki nie skończą się czasy pionierów.

Pamięć wirtualna na niektórych komputerach przyczyniła się do usunięcia jednego z głównych zastrzeżeń w odniesieniu do oprogramowania DBDC wysokiego szczebla, a przyszłość prawdopodobnie przyniesie znaczny wzrost wielkości pamięci głównej komputerów. U niektórych pakietów oprogramowania /ale prawdopodobnie nie wszystkich/ powinno nastąpić znaczne polepszenie jasności konstrukcji i modularności tak, że będą one łatwiejsze do wdrożenia i konserwacji i będą stawiać mniejsze wymagania talentowi programistów systemów.

Wydaje się pewne, że kompleksowość sieci teleprzetwarzania i struktur baz danych będzie wzrastać. Będzie większy wybór końcówek, urządzeń sieci /takich, jak inteligentne koncentratory i programowane łącznice/ oraz technik organizacji zbiorów. W planowaniu i budowie są nowe typy linii komunikacyjnych i

sieci teleprzetwarzania, które będą musiały współistnieć ze starymi. Ta rosnąca różnorodność będzie coraz bardziej zwiększała wagę rozdziału logicznych operacji wejścia/wyjścia od fizycznych.

VII. OPROGRAMOWANIE W MASZYNACH PERYFERYJNYCH

W większości systemów teleprzetwarzania lat sześćdziesiątych oprogramowanie znajdowało się całkowicie w centralnym procesorze. W wielu systemach instalowanych dzisiaj do sterowania niektórymi funkcjami wejścia/wyjścia używane są komputery peryferyjne. Najpospolitszym przykładem jest komputer sterowania linią transmisyjną, który obsługuje sieć teleprzetwarzania, przekazuje kompletne, sprawdzone a nawet częściowo zredagowane komunikaty do jednostki centralnej i otrzymuje od niej komunikaty do nadania. Używane są także koncentratory o stałym programie i kontrolery końcówek. Ten podział funkcji przetwarzania będzie stawał się coraz bardziej powszechny dzięki szybko spadającemu kosztowi minikomputerów i przyszłemu potencjałowi mikrokomputerów, mieszczącym się całkowicie na jednym lub dwu "chipach" obwodów scalonych /LSI/. W niektórych systemach komputery peryferyjne będą prawdopodobnie używane zarówno do dostępu do bazy danych, jak i dla wykonywania funkcji teletransmisji; w istocie właśnie tego wymaga idea poliprzetwarzania.

Tam, gdzie stosowane są komputery peryferyjne, funkcje oprogramowania muszą zostać między maszynami podzielone. Pożądane jest uzyskanie najbardziej ekonomicznego podziału pracy, takiego, który ułatwi przyszły rozwój i rozbudowę systemu.

Stosowanie komputerów peryferyjnych przyniesie szereg korzyści:

1. Komputer centralny może zostać odciążony od prac rutynowych i zwolniony od przetwarzania danych na większą skalę. Niektóre funkcje peryferyjne są to zabiegi powtarzalne, lepiej wykonywane

przez małe oddzielne maszyny, jak na przykład ciągłe przepytywanie sieci o małej ruchliwości. Kiedy funkcja taka, jak przepytywanie, kontynuowana jest przez jednostkę centralną, odbiera to wiele czasu od innych zadań, a licznik kosztownej maszyny nabija czas zmarnowany na bezproduktywne /negatywne/ przepytywanie.

Jeśli użytkownik płaci na podstawie czasu zegarowego, jest to rzeczywiście droga metoda przepytania.

2. Systemy operacyjne dla układów bezpośrednich /on-line/ są wysoce skomplikowane, a będą prawdopodobnie jeszcze bardziej. Przesunięcie wielu funkcji do maszyn peryferyjnych może zmniejszyć komplikacje, a zatem może zredukować liczbę zacięć programów oraz zmniejszyć konieczność zaangażowania programistów systemów.

3. Komputer sterujący linią może utrzymywać swoją sieć w stanie gotowości, nawet gdy komputer centralny nie pracuje. Może on składować przechodzące komunikaty dla późniejszego przetworzenia lub polecić operatorom końcówek, aby przesłali je później. Kraksa programowa w komputerze centralnym nie musi wyłączyć sieci i spowodować uciążliwej procedury uruchamiania jej na nowo.

4. Operacje wejścia/wyjścia pisane przez programistę zastosowania mogą być uproszczone. Użycie komputerów peryferyjnych jest jedną z dróg do rozdziału logicznych operacji wejścia/wyjścia od fizycznych.

5. Sieć lub baza danych może zostać zmieniona przez zmianę przydzielonego komputera peryferyjnego lub jego oprogramowania, bez potrzeby zmiany komputera centralnego lub jego programu.

6. Komputer peryferyjny może być prosty i skonstruowany dla określonej funkcji, na przykład z repertuarem rozkazów lub mechanizmem podziału pamięci, właściwymi dla teleprzetwarzania wejścia/wyjścia.

7. Stosowanie małych komputerów dla sterowania peryferyjnymi elementami systemów ułatwia przyłączanie wyrobu niezależnego wytwórcy urządzeń peryferyjnych do jednostki przetwarzającej

wielkiego producenta. Można tak zrobić, by peryferyjne sieci lub maszyny wydawały się komputerowi centralnemu proste. Komputer peryferyjny może symulować taśmę lub dysk, a zatem być adresowany przy pomocy języka wyższego rzędu, takiego jak COBOL lub PL/1.

Umożliwia to większą elastyczność projektowania, ponieważ wytwórcy urządzeń peryferyjnych oferują szeroki wybór końcówek, jednostek odpowiadających głosem, sprzętu gromadzenia danych, rejestrów kasowych i tym podobnych.

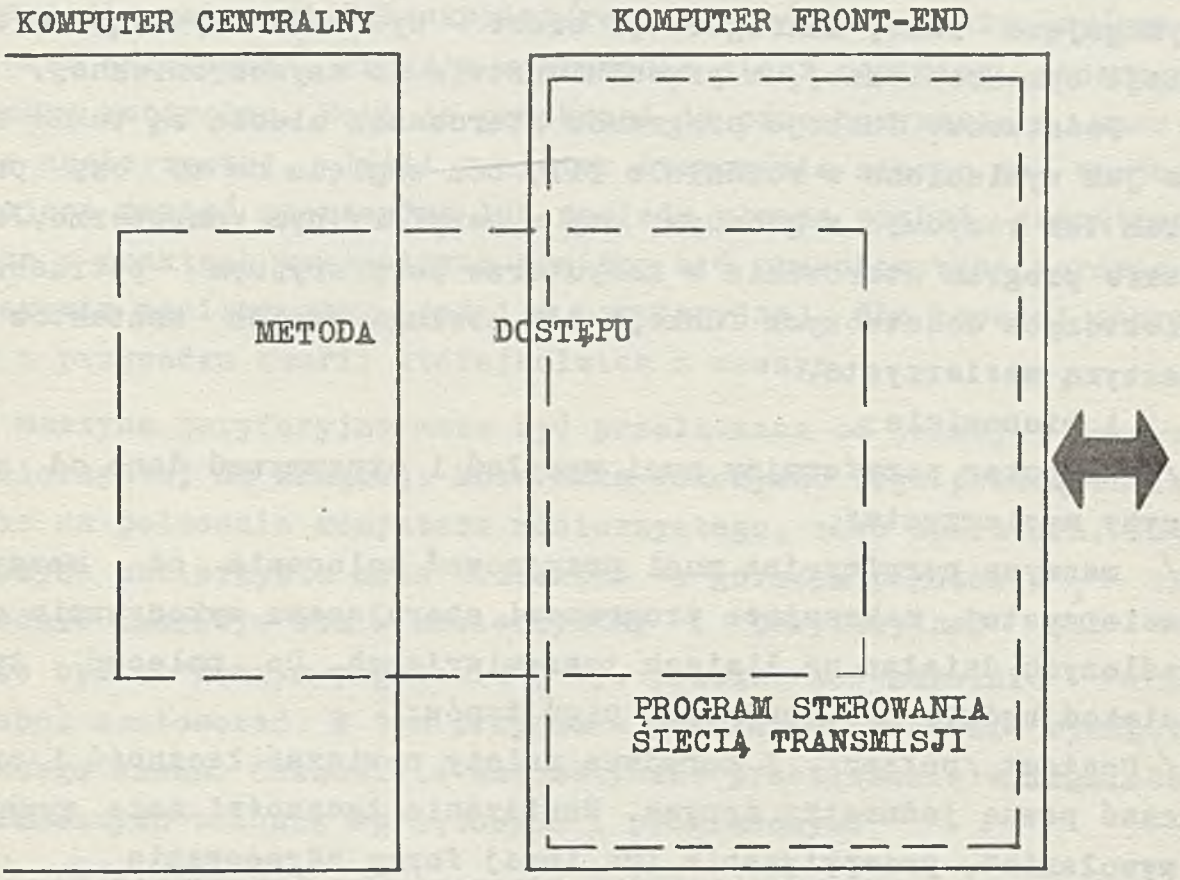
W latach sześćdziesiątych wysuwano dwa główne argumenty przeciwko komputerom peryferyjnym. Pierwszy związany był z zasadą Grosch'a, że koszt operacji jest tym mniejszy, im większa jest maszyna. Opłacało się obsługiwać operacje wejścia/wyjścia, tak samo jak wszystko inne, w maszynie centralnej. Argument ten został nieco osłabiony spadkiem kosztu minikomputerów,który został umożliwiony przez masową produkcję. Nadal jest jednak prawdą, że tam, gdzie występują tylko niewielkie potrzeby w teleprzetwarzaniu, na przykład jedna lub dwie linie, odrębny komputer nie opłaca się i linie mogą kończyć się jednostką adaptera linii, albo też bity z linii mogą być odprowadzane bezpośrednio do głównej pamięci jednostki centralnej.

Drugi argument przeciwko komputerom peryferyjnym dotyczy niezawodności. Dwa komputery połączone szeregowo są mniej niezawodne niż jeden. Jednakże wzrastający stopień scalania obwodów przyniósł ze sobą wzrost niezawodności obwodów aż do momentu, gdy gotowość systemu zależy bardziej od usterek oprogramowania niż od usterek sprzętu. Tak więc zastosowanie komputerów peryferyjnych przyniosło lepsze zabezpieczenie od usterek oprogramowania.

Oprogramowanie może być rozplanowane w ten sposób, że istnieje czysty i prosty rozdział między komputerami peryferyjnymi i centralnym. Nie zawsze jednak ma to miejsce.

IBM/370 może dla sterowania swoją siecią teleprzetwarzania zatrudnić komputer o stałym programie IBM 3705. Może on mieć pamięć wewnętrzną w granicach od 16K do 240K, a zatem jest w stanie pomieścić dość pokaźną porcję oprogramowania. 3705 może zostać zaprogramowany tak, aby podjąć inteligentne działanie

wówczas, kiedy /370 "wysiądzie", jak również może przejąć znaczną część przetwarzania wejścia/wyjścia od maszyny macierzystej. "Metoda dostępu" teleprzetwarzania, na przykład TCAM, zostaje podzielona między komputer macierzysty i 3705 jak pokazano na rys. 8. Wygenerowanie metody dostępu zarówno dla /370, jak i dla 3705 następuje w komputerze macierzystym.



Rys.8. Schematyczny podział oprogramowania pomiędzy komputerem centralnym i komputerem front-end

Sieć transmisji danych sterowana jest przez program sterowania sieci, rezydujący w 3705. Dla każdego komputera sterowania sieci ten program kontrolny gra kluczową rolę, jeśli chodzi o efektywność eksploatacji i kiedy dokonuje się wyboru sprzętu i oprogramowania, należy mu się bacznie przyjrzeć.

Program kontroli sieci 3705 generowany jest i załadowywany przez komputer macierzysty. Może zostać wygenerowanych szereg

wersji tego programu, przechowywanych w komputerze macierzystym i nazwanych symbolicznymi imionami. Jeśli do sieci wprowadza się od czasu do czasu zmiany lub jeśli różne zastosowania teleprzetwarzania, w różnym stopniu wykorzystujące możliwości sieci, są realizowane w różnych porach dnia, istnieje potrzeba wielu programów stosowania. Na przykład końcówki uwarunkowane czasowo mogą pod koniec dnia roboczego być wyłączane, a linie zostać przekazane do wykorzystania przez przetwarzanie wsadowe, wymagające innej konfiguracji sieci - być może używające transmisji synchronicznej, w przeciwieństwie do asynchronicznej.

Podstawowe funkcje programów sterowania siecią są takie same jak wymienione w rozdziale III, bez względu na to czy program ten rezyduje w głównym, czy w peryferyjnym komputerze. Jednakże program sterowania w komputerze peryferyjnym potrzebuje niektórych dodatkowych funkcji dla obsługi swoich kontaktów z maszyną macierzystą.

A mianowicie:

1/ komputer peryferyjny musi wysyłać i otrzymywać dane od maszyny macierzystej,

2/ maszyna peryferyjna musi przyjmować polecenia od maszyny macierzystej, nakazujące programowi sterującemu wykonywanie określonych działań na liniach transmisyjnych. Do poleceń tych należeć będzie następujących pięć typów:

a/ Contact /połącz/. Z końcówką należy nawiązać łączność i przekazać pewną jednostkę danych. Nawiązanie łączności może wymagać "wywołania", przepytywania lub innej formy adresowania.

b/ Invite /zaproś/. Znów należy nawiązać łączność z końcówką, na przykład przez wywołanie lub przepytywanie, ale w tym wypadku końcówka otrzymuje polecenie przesłania jednostki danych, jeśli taką posiada.

c/ Read /czytaj/. Jednostka danych zostaje przesłana od końcówki do maszyny macierzystej.

d/ Write /wpisz/. Jednostka danych zostaje przesłana od maszyny macierzystej do końcówki.

e/ Disconnect /rozłącz/. Spowoduje to zerwanie połączenia z urządzeniem /tak, jak odłożenie słuchawki telefonicznej/ lub przesłanie znaku sterującego, który spowoduje rozłączenie logiczne od linii wielokońcówkowej.

3/ maszyna peryferyjna musi powiadomić końcówki, w wypadku gdy maszyna macierzysta lub kanał zawiedzie. Powiadomienie powinno instruować dokładnie operatora, co ma robić,

4/ maszyna peryferyjna musi powiadomić komputer macierzysty o jakiejkolwiek, nie nadającej się usunąć awarii lub ciągłych przekłamaniach na linii w sieci,

5/ maszyna peryferyjna musi współpracować z macierzystą dla zapewnienia możności "checkpoint/restart" /punkt kontrolny/wznowienie działania/. Program sterowania sieci powinien generować punkty kontrolne. Może je przekazać do przechowywania w maszynie macierzystej. Jeśli komputer sterowania siecią ma wystarczającą pamięć operacyjną lub posiada własną pamięć zewnętrzną, dane o punktach kontrolnych powinny być przechowywane zarówno w maszynie macierzystej, jak i w peryferyjnej, dla lepszej ochrony w przypadku awarii którejkolwiek z maszyn,

6/ maszyna peryferyjna może być przełączana od jednej maszyny macierzystej do drugiej. Może sama dokonywać tego przełączania albo na polecenie komputera macierzystego, albo operatora. Nowa maszyna macierzysta może oczekiwać "w gorącym pogotowiu", tj. śledzić operacje obu - macierzystej i peryferyjnej lub też może być w "zimnym pogotowiu", tj. przetwarzać zupełnie inny zespół zastosowań. W tym przypadku przełączenie będzie wymagało pewnego czasu. Całkowicie automatyczne przełączenie wymaga skomplikowanych technik sprzętowych i programowych,

7/ może następować przełączenie z jednej maszyny peryferyjnej na drugą. Potrzebna jest wówczas umiejętność prawidłowego wyłączenia jednej maszyny i uruchomienia drugiej,

8/ maszyna peryferyjna musi często emulować inne urządzenia, z którymi może współpracować sprzęt lub oprogramowanie maszyny macierzystej. Na przykład na IBM/360 maszyna peryferyjna 3705 często emuluje jednostkę sterowania transmisją 2701, 2702 lub 2703. Wiele innych komputerów sterowania siecią emuluje standardowe urządzenia peryferyjne takie, jak jednostka taśmowa lub dyskowa.

... Przy zastosowaniu emulacji mogą być łączone ze sobą maszyny, które normalnie nie byłyby kompatybilne. Mogą być łączone

w system maszyny różnych producentów. "Kompatybilne na wtyczkę" maszyny peryferyjne od mniejszych producentów są podłączane do komputerów wielkich firm. Maszyna peryferyjna może zachowywać się dla maszyny macierzystej, tak jak konwencjonalna jednostka taśmowa lub dyskowa. Oprogramowanie maszyny centralnej może wówczas adresować ją, używając języka wyższego rzędu, takiego jak COBOL.

Szereg systemów używa małych komputerów, udających jednostki taśmowe. Mogą to być minikomputery lub maszyny takie, jak PDP11 firmy Digital Equipment Corporation. Mogą one być zaprogramowane do sterowania całą siecią, do wykonywania funkcji komutowania komunikatów, funkcji zbierania danych lub wykonywania jakiejś formy współdziałania między człowiekiem a maszyną.

Komputer peryferyjny znajduje się czasem w sali obliczeniowej, obok swojej maszyny macierzystej. Coraz częściej jednak będziemy znajdowali jego zdalne zastosowanie, jako inteligentnego koncentratora, maszyny sterującej zbieraniem danych, kontrolującej grupę końcówek lub sterującej inteligentnym urządzeniem takim, jak końcówka wykreślająca graficznie /koordynatograf/.

Komputer zdalnego sterowania siecią lub koncentrator potrzebuje tych samych funkcji programowych, jak wymienione wyżej, z tą różnicą, że jego kanał do komputera macierzystego będzie sam linią transmisji danych. Pomimo tego, że komputer sterowania siecią znajduje się daleko, ładowanie do niego programu może być nadal dokonywane przez maszynę macierzystą. Zaletą komputerów zdalnych do sterowania teleprzetwarzaniem jest to, że na niektórych systemach można uzyskać znaczne oszczędności w kosztach linii transmisyjnych. Czasami uda się uniknąć konieczności używania kosztownej lub trudno dostępnej linii szeroko pasmowej, właśnie dzięki użyciu do sterowania teleprzetwarzaniem komputera zdalnego.

VIII. JĘZYKI DLA ANALITYKA SYSTEMÓW

Wszystkim nam znane są języki, których normalnie używa programista. Drugą ważną klasą języków są te, które powstają dla analityków i programistów systemów, przy pomocy których możemy uściślać wymagania systemu. Należą do nich nowo powstałe języki systemowe w dziedzinach teleprzetwarzania i systemów DBDC. Oto niektóre z nich:

1/ języki do generowania programu sterującego siecią transmisji,

2/ języki do opisu organizacji zbiorów lub metod dostępu /powszechnie znane jako języki opisu danych - data definition languages DBL/,

3/ języki opisujące przechowywane dane oraz logiczne i fizyczne zależności między elementami danych /powszechnie znane jako języki manipulowania danymi - data manipulation languages DML/,

4/ języki do sterowania rozplanowaniem zadań w komputerze,

5/ języki do przepytywania bazy danych, ustalające metody poszukiwania dla formułowania odpowiedzi na skomplikowane pytania. Mogą one być zaprojektowane dla interaktywnej eksploatacji końcówek lub też dla przetwarzania wsadowego /znane ogólnie jako języki zapytań/,

6/ dialogi dla użytkowników systemów, którzy nie są programistami. Języki te nie mają formalnej struktury "języków" komputerowych,dlatego też będziemy je tu raczej nazywać "dialogami", a nie językami.

Podjęzyki

Tak zwane podjęzyki przyjmują kilka różnych form. Mogą to być języki tabelarne, przy pomocy których użytkownik wypełnia tabele o określonych elementach.Mogą to być języki poleceń, które na ogół rozpoczynają każdą instrukcję od czasownika, po któ-

rym następują parametry. Stosowane są również języki swobodne, nieformalne, o deklaracjach przypominających angielski. Mogą one przyjmować formę dialogu między człowiekiem a komputerem, w którym komputer wypytuje użytkownika, aby otrzymać potrzebną mu informację.

Języki proceduralne a języki przystosowane do problemu

Podjęzyk może wymagać od użytkownika, aby określał postępowanie, przy pomocy którego ma być coś osiągnięte. Nazwiemy go "językiem proceduralnym". Język "nieproceduralny" lub przystosowany do problemu żąda po prostu od użytkownika, aby podał co ma zostać dokonane, a nie, jak ma to być zrobione.

Języki proceduralne są korzystne dla użytkownika, który dokładnie wie czego chce, na przykład jak zorganizowanej struktury pamięci potrzebuje dla optymalnej efektywności albo jaki schemat priorytetów powinien być używany do transmisji danych. Języki nieproceduralne służą dla tych użytkowników, którzy nie znają lub nie rozumieją postępowania optymalizującego albo też nie mają czasu ani przeszkolenia, aby je przeprowadzić.

Poniższa tablica sugeruje, jakie kategorie użytkowników mogą stosować jakie różne formy podjęzyków - choć oczywiście mogą tu występować odmiany i odchylenia:

	PROCEDURALNE	NIEPROCEDURALNE
TABELARNE	Wyszkoleni urzędnicy i programiści	Niewyszkoleni urzędnicy i analitycy
POLECENIOWE	Wyszkoleni programiści i analitycy	Niewyszkoleni programiści, analitycy i kierownictwo
SWOBODNE	Wyszkoleni analitycy i kierownictwo	Niewyszkoleni analitycy i kierownictwo
DIALOGOWE	Wyszkoleni urzędnicy, analitycy i kierownictwo	Niewyszkoleni analitycy i kierownictwo

Tablica 1. Języki proceduralne i nieproceduralne

W tym rozdziale omówimy języki dla programów sterujących, a w następnym rozdziale przedyskutujemy języki przepytywania.

Języki programów sterujących

Istnieje duża rozmaitość konfiguracji programów sterujących, które sterują peryferyjnymi częściami systemów. Prawie dla każdego systemu inny jest program sterujący. Dlatego też do generowania programów sterowania istnieją specjalne języki. Niektóre języki generowania programów sterujących używane są dla baz danych, a inne dla podsystemów teleprzetwarzania. Programy sterujące, generowane przez nie, mogą rezydować albo w komputerze peryferyjnym, albo też w centralnej jednostce, zależnie od tego, które z nich spełnia funkcję sterowania. W obu przypadkach najprawdopodobniej maszyna sprawujące sterowanie będzie również wykonywać pracę generowania.

Programy sterowania, otrzymane przy pomocy języka generowania programów, będą prawdopodobnie mniej sprawne niż programy w asemblerze, specjalnie napisane dla danego systemu - podobnie jak program w FORTRANIE jest mniej sprawny, niż odpowiadający mu program w asemblerze. Mają one jednak tę wielką zaletę, że mogą szybko i łatwo zostać zmienione, kiedy zmieniają się wymagania. Dla większości systemów, z wyjątkiem najmniejszych, wymagania takie, jak konfiguracja sieci lub rozplanowanie danych, zmieniają się dostatecznie często, aby to było warte zachodu. Dla bardzo małego, niezmiennego, niedynamicznego systemu specjalnie przystosowany program sterujący może być bardziej korzystny. Przyszłość jednak należy do języków programów sterowania siecią, języków opisu bazy danych itp.

Za typowy może być uważany język do generowania programu sterowania siecią dla IBM 3705. Ma on trzy główne typy makrorozkazów. Typ pierwszy nazywany jest "system macro". Podaje on informacje o samej jednostce 3705, na przykład wielkość pamięci 3705, szczegóły adaptera kanału i funkcji checkpoint-restart oraz wielkość buforów tworzących zapas buforów. Użytkownik systemu może zadecydować zmianę wielkości buforu, jeśli zmieni się długość jego komunikatu.

Drugi typ makrorozkazu nazywany jest "configuration macro"

i opisuje elementy sieci teleprzetwarzania. Makrorozkazy tego typu podają generatorowi wystarczające informacje do budowy tablic, potrzebnych programowi sterowania do kontroli sieci. Dla każdego elementu sieci jest jeden makrorozkaz. Kolejność, w jakiej one są ustawione, wskazuje na konfigurację sieci przy hierarchii idącej od najwyższego do najniższego poziomu, na przykład: grupa linii, linia, gniazdo końcówek /kilka końcówek przyłączonych do jednej jednostki sterującej/, końcówka i element końcówki. Inne makrorozkazy konfiguracji podają szczegóły o procesorze macierzystym i metodzie dostępu do niego, o urządzeniach do przepytywania telekomunikacyjnego itd.

Trzeci typ nazywany jest makrorozkazem manipulacji blokiem /"block handling macro"/. Te makrorozkazy opisują przetwarzanie, które program sterowania siecią może wykonywać na bloku danych, przed przekazaniem tego bloku albo do komputera macierzystego, albo do urządzenia sieci. Program może na przykład dodać godzinę i datę do komunikatu. Może on zredagować komunikat. Dla jednego systemu może istnieć cały wybór specjalnie napisanych podprogramów manipulowania blokami.

Bardzo ważnym elementem niektórych systemów DBDC jest <u>język opisu bazy danych.</u> Język ten umożliwia opisanie danych wraz z zależnościami lub połączeniami pomiędzy segmentami, lub elementami /zapisami lub polami/ danych. Często używane są dwa rodzaje opisu danych; pierwszy - jest to opis logiczny /jakie elementy danych wiążą się z jakimi segmentami w logicznym obrazie danych, jaki ma programista/, drugi - jest to opis fizyczny /jak elementy i segmenty danych są powiązane fizycznie i w jakich urządzeniach rezydują/. Język opisuje długość wszystkich elementów danych, rodzaj znaków /BCD[x], binarne, upakowane dziesiętne itp./i wszystkie związki między elementami danych. W wyniku otrzymuje się <u>bibliotekę opisu danych</u> albo rodzaj elementarnego <u>słownika danych</u>, który w tym wypadku zawiera tylko fizyczne cechy danych, a nie ich charakterystyki.

Ta biblioteka opisu danych ma służyć <u>systemowi</u> przy generowaniu modułów jego programu sterującego. Pełniejsza forma

---

x/ BCD, binary coded decimal, zapis dziesiętny kodowany dwójkowo /przyp.red./

katalogu opisu danych, właściwy słownik danych, jest cennym narzędziem dla analityków systemów. Podaje on opis słowny wszystkich elementów danych i powiada kiedy są one używane,jakie jest ich źródło, jak często są aktualizowane itd. Jego zasadniczym przeznaczeniem jest wyeliminowanie redundancji i dwuznaczności, jakie cechują dane w większości dzisiejszych systemów komercyjnych. Te same elementy danych często są przechowywane zbytecznie w różnych zbiorach dla różnych zastosowań. Elementy powtarzające się są często aktualizowane w różnych przebiegach, powodując niezgodności w przechowywanych danych. Często szczegóły i przeznaczenie nadmiarowych elementów danych są różnie opisywane przez różne grupy. W wielu firmach normalizacja elementów danych i wyeliminowanie szkodliwej redundancji są największymi problemami przy opracowywaniu bazy danych. Dobre pomoce do projektowania bazy danych mogą dopomóc w wyjaśnieniu, a zatem i rozwiązaniu tego problemu. Oprogramowanie DBDC może służyć pomocą w utworzeniu słownika danych. Słownik danych w większości takich systemów jest używany pośrednio /off-line/, na kilka z nich - on-line. Słownik danych może stanowić rozwinięcie biblioteki opisu danych, a inwersje i listy kluczowych słów w kontekście /key word in context KWIC/ mogą pomóc w eliminowaniu redundancji. Bardziej szczegółowe opisy można znaleźć w bibliografii na końcu tego sprawozdania.

Systemy DBDC zazwyczaj eksploatowane są w środowisku gwałtownie zmieniających się wymagań i techniki,i to jest głównym powodem, dla którego język opisu danych i biblioteka danych są tak ważne. Konieczne są ciągłe zmiany w zawartości i organizacji zgromadzonych danych, zmiany częstsze nawet, niż występują w sieciach transmisji danych. Zanim zjawiło się dzisiejsze oprogramowanie DBDC programy zastosowań zawierały w rzeczywistości opis danych w swojej treści. Takie programy nie mogą być dostatecznie szybko adaptowane. Dzięki zgromadzeniu informacji o elementach danych, ich zależnościach i ich wymaganiach w zakresie bezpieczeństwa poza programami zastosowań, te ostatnie mogą się uniezależnić od zmian w danych i w mechanizmie ich przechowywania. Ta pożądana cecha nazywana jest "niezależnością od danych" /data independence/. W niektórych systemach programy są

logicznie wiązane z danymi /"binding"/ w czasie kompilacji. W innych, binding następuje w czasie załadowywania programów. Ten drugi sposób może nieco pogarszać czas reakcji on-line, ale poprawia bezpieczeństwo i elastyczność systemu i im później stosowany jest binding, tym większy stopień niezależności od danych.

Poniższa tablica stanowi listę podjęzyków występujących w niektórych znanych systemach DBDC:

SYSTEM	WYTWÓRCA	RODZAJ	PROCEDURALNY
MARK IV	Informatics	T	Nie
MANAGE	Xerox	T	Nie
COGENT	Computer Sciences Corporation	T	Nie
GIS	International Business Machines	C	Tak
IMS V2	International Business Machines	C	Nie
CICS	International Business Machines	C	Nie
DB Subsystem	Honeywell	C	Tak
UL/1	Radio Corporation of America	F	Nie
GIM	Trans World Airlines	F	Nie
TDMS	Systems Development Corporation	C	Nie
ISL-1	Information Systems Leasing Corporation	C	Nie
IDS	Honeywell	C	Tak
	T: Tabularny C: Poleceniowy F: Swobodny		

Tablica 2. Podjęzyki DBDC

IX. SYSTEMY DIALOGOWE

Istnieje zasadniczy związek pomiędzy strukturą dialogu człowiek-komputer oraz urządzeniami teleprzetwarzania, które powinny być zastosowane do wdrożenia tego dialogu. Martin w swojej książce "Projektowanie dialogów człowiek-komputer" głosi, że projektowanie systemu teleprzetwarzania powinno zaczynać się od badania użytkownika i rodzaju dialogu, który będzie mógł on najefektywniej zastosować. Skoro zostanie zrozumiana struktura dialogu, dopiero wówczas może zostać wybrany najlepszy do wdrożenia tego dialogu układ sprzętu i oprogramowania teleprzetwarzania. Oprogramowanie zaprojektowane dla ściśle określonej struktury dialogowej może być znacznie bardziej efektywne w osiąganiu swoich celów niż oprogramowanie teleprzetwarzania o charakterze uogólnionym. Istnieje wiele możliwości wyważenia korzyści pomiędzy konstrukcją struktury dialogowej i konstrukcją sieci i oprogramowania teleprzetwarzania.

Istnieje więcej możliwych kategorii struktury dialogu i struktury podsystemu teleprzetwarzania, niż się wydaje przeciętnemu analitykowi systemów - opłaca się więc zapoznać ze wszystkimi wariantami. W tablicy 2 mieliśmy zestawienie struktur podsystemów teleprzetwarzania. Tablica 3 podaje wykaz popularnych form struktury dialogowej, z których każda może mieć jeszcze wiele odmian. Daje się odczuć silna potrzeba przechodzenia do form dialogu, które są zarówno łatwiejsze, jak i bardziej efektywne.

Oprogramowanie teleprzetwarzania dla danej struktury dialogowej może być zaprojektowane w sposób umożliwiający osiągnięcie celów charakterystycznych dla danego dialogu, takich jak rozkład czasów reakcji, punkty kontrolne /checkpoints/, oraz metody postępowania z błędami i wznawiania programu /restarts/. Tak dostosowane oprogramowanie może używać optymalnych metod przydziału pamięci i sterowania linią, jak również optymalnej struktury priorytetów.

Następujące systemy pozwalają na używanie języków przepytywania z końcówek:

SYSTEM	WYTWÓRCA	TYP
IQF	International Business Machines	
AKSESS	Response Technology	
QL/1	Nortel International	
Model 103/104	Computer Corporation of America	
MARS VI	Control Data Corporation	Interpreter
Data Query System	Honeywell	
DMS	Burroughs	
System 2000	MRI, Inc.	Interpreter
SERIES	Information Systems Management	Precompiler
Datamaster	Systems Development Corporation	
TDMS	Systems Development Corporation	Compiler
DS/2	Systems Development Corporation	Interpreter
ORBIT II	Systems Development Corporation	
IMS, UNIMS	Univac	Interpreter
GIM	TRW, Inc.	Compiler
MUSE	Meta Language Products Inc.	Interpreter
UL/I	Radio Corporation of America	Compiler

Tablica 3. Struktury dialogowe

Korzyści wynikające z takiego zintegrowanego projektowania oprogramowania są najlepiej widoczne w programach zaprojektowanych dla konkretnych zastosowań, takich jak systemy bezpośrednie w bankowości i w rezerwacji miejsc lotniczych. Na przykład program sterujący PARS, opracowany przez IBM, został zaprojektowany pierwotnie dla systemu rezerwacji miejsc lotniczych przy użyciu /360 lub /370. Był jednak używany do całkiem innych zastosowań, odznaczających się dużym ruchem w czasie rzeczywistym i podobną strukturą dialogową, takich jak systemy finansowe i policyjne. W tych zastosowaniach okazuje się on znacznie bardziej wydajny /albo, mówiąc inaczej, uzyskuje się znacznie niższy koszt systemowy dla danej przepustowości i czasu reakcji/ niż uogólnione programy teleprzetwarzania, takie jak BTAM,TCAM, CICS lub IMS. Z drugiej strony, ten program sterujący z trudnością daje się zastosować lub jest całkowicie bezużyteczny dla systemów teleprzetwarzania o charakterze daleko odbiegającym od jego założeń projektowych. Jest jednak prawdopodobne, że w przyszłości będziemy widzieć coraz więcej oprogramowania zaprojektowanego dla potrzeb danego przemysłu lub przykrojonego do określonej kategorii zastosowania. W takim oprogramowaniu mechanizmy dialogu, a czasem nawet cały dialog, mogą zostać opakowane wraz z programami teleprzetwarzania.

Szczególną klasą podsystemu dialogowego jest system przepytywania bazy danych. Dostępnych jest wiele języków przepytywania bazy danych, często stanowiących część określonego systemu DBDC. Niektóre z nich zaprojektowano do użytku pośredniego /off-line/, a niektóre, włączając w to wymienione w tablicy 3, zaprojektowano do stosowania za pośrednictwem końcówki. Efektywność języka przepytywania bazy danych w znacznym stopniu zależy od struktury bazy danych, co ma szczególne znaczenie w systemach czasu rzeczywistego. Struktura bazy danych, konstrukcja końcówki, a wreszcie struktura sieci teleprzetwarzania /jeśli jest taka sieć/ - wszystkie zależą od struktury dialogu.

Popularna forma języka przepytywania bazy danych umożliwia użytkownikowi poszukiwanie informacji, która odpowiada określonym kryteriom. Instrukcje na wejściu, takie jak podana niżej, są typowe:

p o d a j w y k a z w s z y s t k i c h o d d z i a-
ł ó w b r a n ż o w y c h o /wydatkach/dochodach/ z a
g r u d z. 1972 w i ę k s z y c h n i ż 0,05

Większość takich języków zaprojektowana jest tak, aby ignorować pewne słowa zerowe, które użytkownik wstawia do zapytania po to, aby bardziej przypominało angielski. Większość z nich stosuje również następującą listę skrótów i znaków:

PORÓWNANIA	ARYTMETYCZNE		LOGICZNE
EQ: równe	+	dodać	AND
NQ: nierówne	-	odjąć	OR
LT: mniejsze od	x	pomnożyć	NOT
GT: większe od	/	podzielić	IF
LE: mniejsze lub równe	E	potęgowanie	ELSE
GE: większe lub równe	SQR /x/	pierw.kwadr.	
	LN /x/	log /nat/	
	SIN /x/	sinus	

Niektóre języki przepytywania pozwalają na szybkie i łatwe przygotowywanie dobrze sformalizowanych odpowiedzi. Oszczędzają one pracę programisty, która byłaby konieczna gdyby takich języków nie było i umożliwiają odpowiadanie bardzo szybko na żądanie informacji, wyrażane przez kierownictwo. Jak dalece się to udaje, zależy od struktury i zawartości bazy danych.

Wiele programów teleprzetwarzania opracowano z myślą o wydajności - tj. zaprojektowano je dla dużego ruchu lub krótkich czasów reakcji, lub jednego i drugiego naraz. Jednak zazwyczaj nie wydajność jest głównym kryterium dla systemów, które mają pozwolić dyrektorom lub ich sztabowi na stawianie pytań bazie danych. Tutaj głównym kryterium jest, aby przedstawiały żądane odpowiedzi i sprawozdania szybko, bez potrzeby programowania ich indywidualnie. Częstotliwość i liczba takich zapytań mogą być niskie i użytkownik na ogół nie żąda natychmiastowej odpowiedzi, tak więc wydajność ma drugorzędne znaczenie. Z tego względu programy sterujące transmisją danych mogą być prostsze.

Przekład z języka przepytywania wykonywany jest w różnych oprogramowaniach na trzy różne sposoby.

1. Może być używany program-tłumacz /interpreter/
2. Może być zastosowany kompilator on-line
3. Może być używany wstępny kompilator, który generuje kod pośredni, nadający się dla standardowego kompilatora on-line, takiego jak COBOL.

Tłumacz bezpośredni zapewnia szybką pracę. Jednakże jego zastosowanie ogranicza się na ogół do prostych pytań, zazwyczaj pytań z jednego zbioru. Nie jest generowany i przechowywany kod do powtórnego użycia, tak więc te same pytania mogą wymagać każdorazowego przetwarzania na nowo. Kompilator on-line może przetwarzać szybko bardzo złożone pytania, ma jednak duże wymagania w stosunku do pamięci. Jego cena jest też zwykle najwyższa. Kompilator wstępny /pre-compiler/ bywa tańszy, jest jednak powolny, z uwagi na potrzebną operację dodatkowej translacji. Ma również duże zapotrzebowanie na pamięć, szczególnie, jeśli stosowany on-line. Pre-kompilatory są zwykle używane dla pytań w seriach.

Istnieje niebezpieczeństwo, że wtedy, kiedy języki przepytywania są łatwe w stosowaniu, wówczas użytkownik może posługiwać się nimi nie biorąc pod uwagę ich wpływu na pracę maszyny. Pytania, które stawia się w takim języku bardzo łatwo i prosto, mogą zmusić komputer do nad wyraz długotrwałych operacji przeszukiwania zbiorów, które mogą kosztować mnóstwo czasu maszyny. Przekład niektórych pytań jest kosztowny, gdy idzie o czas komputera. Szczególnie odnosi się to do kompilatorów i pre-kompilatorów, przy których niewprawny użytkownik może nadużywać czasu maszyny. Większość systemów przepytywania DBDC nie posiada wbudowanej, wystarczającej funkcji obrachunkowej, która obciążałaby użytkownika rachunkiem za obliczenia, proporcjonalnym do czasu maszyny i w ten sposób nieco moderowała jego działalność. W wyniku końcowym może okazać się pożądane opracowanie takich funkcji obrachunkowych, które będą przedstawiały zapytującemu

różne zakresy odpowiedzi na jego pytanie /np. w ciągu minut, w ciągu godzin lub następnego dnia/ i koszty związane z różnymi wariantami /które powinny być tym niższe, im krótszy czas reakcji/x/.

X. GŁÓWNE KRYTERIA WYBORU OPROGRAMOWANIA

Rozdział ten ma przedstawić główne kryteria wyboru oprogramowania. Następny rozdział poda szczegółową listę kontrolną funkcji, które mogą być włączone do oprogramowania.

GŁÓWNE KRYTERIA

1. Koszty bieżące

Koszty związane z oprogramowaniem powinny być zaliczane albo do kosztów bieżących, albo do kosztów przyszłych. Koszty bieżące mają trzy główne składniki:

a/ cena zakupu lub koszt miesięcznej dzierżawy oprogramowania

b/ różnice w kosztach sprzętu związane z użyciem oprogramowania

Uogólniony pakiet oprogramowania zwykle wymaga więcej cyklów maszynowych i więcej pamięci wewnętrznej niż oprogramowanie zaprojektowane specjalnie dla ściśle określonego celu. Program napisany specjalnie dla danego zastosowania w asemblerze potrzebuje zazwyczaj mniej pamięci wewnętrznej i mniej cykli maszyny, niż pakiet oprogramowania.

---

x/ Jest to zapewne omyłka w oryginale. Powinno być na odwrót. /przyp.red./

c/ różnice w koszcie programowania

Użycie pakietu oprogramowania dla teleprzetwarzania daje oszczędności na pensjach programistów, którzy mogą być dość wysoko płatni. Z drugiej jednak strony, oprogramowanie,które jest nazbyt skomplikowane, wymaga do jego wykorzystywania wykwalifikowanych programistów systemowych. Z wyjątkiem szczególnie prostych przypadków, oszczędności na robociźnie przeważają prawdopodobnie inne koszty.

2. Przyszłe koszty

a/ przyszłe koszty dzierżawy oprogramowania,

b/ przyszłe koszty zakupu oprogramowania, jeśli maszyna zostanie zmieniona,

c/ przyszłe różnice w kosztach sprzętu, związane z użyciem oprogramowania,

d/ przyszłe koszty konserwacji programów[x/],

e/ ewentualne koszty konwersji.

3. Koszty utraconych szans wykorzystania siły roboczej

Wykwalifikowanych programistów jest mało i budżet wydatków na siłę roboczą jest zwykle ograniczony. Jeśli programiści zajęci są pracą przy pakiecie oprogramowania i jego konserwacją, tracone są szanse użycia ich do opracowywania innych,przyszłych zastosowań. Należy postawić pytanie: jaki jest koszt tych utraconych szans? W niektórych przypadkach jest on wysoki i może okazać się, że warto zwolnić wszystkie siły fachowe do prac nad rozwojem zastosowań, nawet kosztem zwiększonego kosztu pakietów oprogramowania i większych narzutów.

---

x/ W wielu instalacjach koszty konserwacji programów są bardzo wysokie. W niektórych /najgorszych/ przypadkach ponad 80% dostępnej siły roboczej zatrudnione jest przy konserwacji istniejących programów, a nie przy opracowywaniu nowych programów. Oprogramowanie, które pozwala na zmniejszenie kosztu konserwacji i konwersji przynosi ze sobą znaczne przyszłe korzyści kosztowe.

4. Jakie urządzenia obsługiwać będzie oprogramowanie?

Czy istnieje możliwość, że w przyszłości zostaną dodane końcówki, których oprogramowanie nie obsłuży? Czy jest możliwe, że użyte zostaną dla obniżenia kosztu sieci teletransmisyjnej takie techniki, których pakiet nie obsłuży /np.koncentratory,multipleksory, PABX[x], techniki pętlowe/? Czy, jeśli zostanie zaawizowany ulepszony model końcówki, nasze oprogramowanie będzie mogło z nią współpracować /np. jeśli do monitora ekranowego zostanie dodana funkcja "inteligentnego" redagowania/?

5. Czy zapewni ono oczekiwany czas reakcji?

Dla niektórych typów struktury dialogowej konieczny jest szybki czas reakcji. Czy dostatecznie szybki czas reakcji zostanie uzyskany, będzie zależało od struktury sieci teleprzetwarzania i jej mechanizmów oprogramowania, sposobu koordynacji pracy przez system operacyjny oraz od mechanizmu bazy danych i technik oprogramowania.

Metody oprogramowania mają decydujące znaczenie w odniesieniu do uzyskania wymaganego czasu reakcji bazy danych przy dzisiejszym sprzęcie pamięci masowej. Szczególnie ma to miejsce wówczas, gdy baza danych używa skomplikowanych mechanizmów adresowania, wielopoziomowych kluczy lub inwertowanych zbiorów. Jeśli chodzi o "front-end", zasadniczy wpływ na czas reakcji ma dobór sprzętu, linii transmisyjnych i struktury sieci, a oprogramowanie ma znaczenie drugorzędne.

Na czas reakcji ma wpływ pewna liczba stosunkowo drobnych czynników w oprogramowaniu teleprzetwarzania. Na przykład: jaki jest używany schemat priorytetów? Czy komunikaty typu A mają pierwszeństwo przed typem B? Gdy używane jest przepytywanie sieci, czy pierwszeństwo jest dawane komunikatom wchodzącym czy wychodzącym? Czy jednostka centralna blokuje linię na czas między przyjęciem komunikatu wejściowego i przesłaniem odpowiedzi

---

x/ PABX, Private Automatic Branch Exchange, prywatna centrala telefoniczna, pozwalająca na przyjmowanie i przekazywanie rozmów z i do publicznej sieci telefonicznej /przyp.red./

na ten komunikat? Przy niektórych pakietach oprogramowania pozostawia się wybór między takimi wariantami. Wybór optymalny będzie różny dla różnych użytkowników i konieczne jest przeprowadzenie kalkulacji, jaki wybór jest najlepszy.

Mechanizmy uzyskiwania szybkiej reakcji za pomocą sprzętu oraz oprogramowania powinny być dobierane we wzajemnym związku. Nie ma na przykład sensu płacić ekstra za pełną linię dupleksową, zamiast za pół-dupleks, jeśli i oprogramowanie i końcówki nie są w stanie wykorzystać możliwości transmisji w obu kierunkach w tym samym czasie.

Obliczenia, związane z ustaleniem potrzebnego czasu reakcji, mogą być dość skomplikowane. Szczegóły wykonywania takich kalkulacji można znaleźć w bibliografii /7/. Pożądane jest skonstruowanie modelu operacji teleprzetwarzania, czy to przez symulację, czy też metodami matematycznymi i zbadanie wpływu różnych mechanizmów sprzętu i oprogramowania na zachowanie się modelu.

6. Wpływ na wydajność użytkownika

W miarę, jak maleją koszty komputerów, a systemy dialogowe dla końcówek stają się coraz bardziej sprawne, coraz ważniejsza staje się wydajność, z jaką użytkownik może pracować, w stosunku do wydajności wykorzystania sprzętu. Możliwości użytkownika końcówki w znacznym stopniu zależą od rodzaju dialogu człowiek-maszyna. Wchodzą tu czynniki takie, jak:

a/ czas reakcji /mierzony od chwili kiedy ukaże się pierwszy znak odpowiedzi/,

b/ szybkość odpowiedzi w znakach na sekundę. /Jednostka ekranowa jest znacznie szybsza, niż końcówka typu maszyny do pisania; linia głosowa jest znacznie szybsza, niż linia dalekopisowa/. Ma to zasadniczy pływ na rodzaj możliwej struktury dialogowej,

c/ jedno-wymiarowe czy dwu-wymiarowe. Użytkownik pracujący z ekranem, korzystającym z dwóch wymiarów przestrzeni, ma więcej możliwości, niż użytkownik z końcówką - maszyny do pisania. Niektóre struktury dialogowe są znacznie bardziej efektywne przy interakcji dwuwymiarowej.

d/ języki. Niektórzy użytkownicy mogą używać języków programowania lub specjalnych języków o podobnej strukturze. Prawdopodobnie większość przyszłych użytkowników komercyjnych /przeciętni urzędnicy i kierownicy/ - nie. Wiele z najbardziej efektywnych dialogów komercyjnych unika całkowicie używania struktur mnemonicznych oraz składni, które byłyby charakterystyczne dla "języka" komputera. Tam, gdzie oprogramowanie zapewnia "dialog" dla potrzeb użytkownika, natura takiego dialogu powinna być dokładnie zbadana i, o ile to możliwe, wypróbowana z żywymi ludźmi. Psychologowie - behawioryści prawdopodobnie zaliczyliby niektóre z dzisiejszych dialogów końcówkowych do "nie nadających się do ludzkiej konsumpcji".

7. Wpływ na wydajność programisty

System powinien zapewniać możliwie najłatwiejszą pracę programisty przy adaptowaniu systemu do potrzeb użytkowych, a także /czasem jest to ważniejsze/, przy konserwowaniu i aktualizacji, w sytuacji, gdy zarówno struktura sieci, jak i struktura bazy danych ulegają częstym zmianom. Języki opisu sieci i bazy danych oraz ich kompilatory odgrywają wielką rolę w osiągnięciu tego celu.

8. Niezawodność

Przyszli nabywcy powinni dokładnie zbadać oprogramowanie z punktu widzenia jego zdolności do utrzymywania całego systemu w stałej sprawności eksploatacyjnej. Należy przeanalizować takie funkcje, jak punkty kontrolne i wznawianie pracy /checkpoint and restart/ oraz moce rezerwowe i procedury ponownego uruchomienia /fallback and recovery/. Należy się przekonać, jak system radzi sobie z błędami na liniach transmisyjnych oraz z uszkodzeniami linii. Tam, gdzie oprogramowanie zapewnia możliwości dialogu, należy dokładnie zapoznać się ze sposobami postępowania w przypadku omyłek użytkownika oraz jego "guzdrania się". Niezawodność samego oprogramowania ma podstawowe znaczenie; systemy zawodzą z powodu oprogramowania częściej, niż z jakiegolwiek innego. Kluczem do tego zagadnienia może być reputacja producenta oprogramowania, jeśli idzie o niezawodność jego produktów.

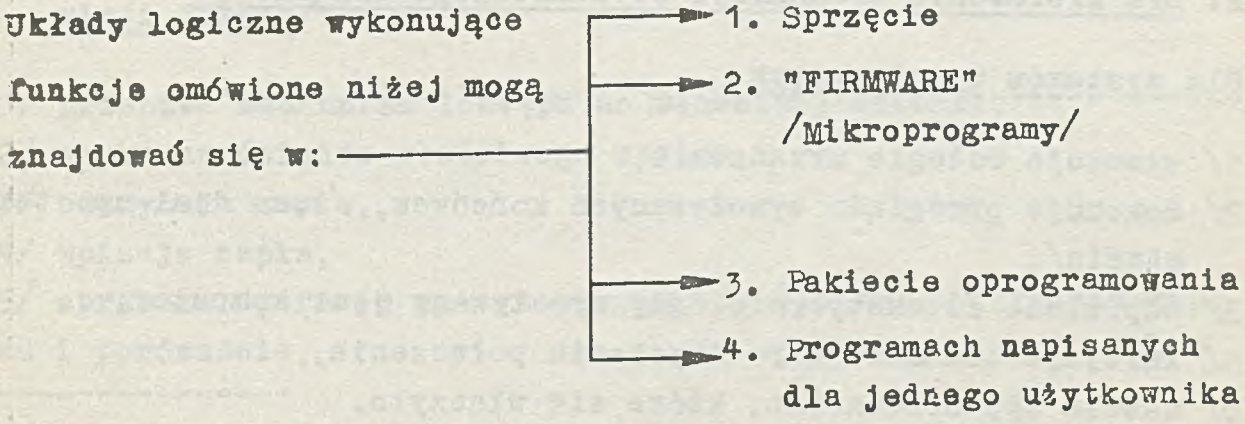
9. Bezpieczeństwo

W wielu systemach przystosowanych do telekomunikacji sprawy bezpieczeństwa i ochrony tajemnicy prywatnej nabierają rosnącego znaczenia. Liczni specjaliści od spraw bezpieczeństwa, którzy orientują się w wielu urozmaiconych, możliwych sposobach naruszania bezpieczeństwa systemu, będą twierdzić, że bezpieczeństwo w większości dzisiejszych systemów teleprzetwarzania jest nie wystarczające. Technika zapewniania bezpieczeństwa zaczyna znajdować zrozumienie. Ważne jest, aby była stosowana.

10. Elastyczność

Wymagania systemu teleprzetwarzania zmieniają się często szybciej niż to przewidywali projektanci systemów. Dlatego ważne jest, aby używany przez nas pakiet oprogramowania był w stanie przystosować się do dużych zmian. Jak dalece jest on elastyczny? W jakim stopniu będzie pomocny przy ewentualnej rozbudowie systemu?

XI. ZESTAWIENIE CECH OPROGRAMOWANIA DLA TRANSMISJI DANYCH



A. Wymagania logiczne dla podstawowych funkcji transmisji

Na odbiorze:

1/ inicjuje i steruje odbiorem danych,
2/ grupuje bity w znaki,
3/ grupuje znaki w komunikaty,
4/ rozpoznaje znaki "Koniec zapisu" i "Koniec transmisji",
5/ dokonuje konwersji kodu znaków,
6/ sprawdza na błędy,
7/ poprawia błędy lub powoduje powtórne nadanie błędnych komunikatów,
8/ redaguje wchodzące komunikaty, usuwając zbędne znaki,
9/ przekazuje komunikaty głównym programom.

Na wysyłce:

10/ przyjmuje komunikaty z głównych programów,
11/ przygotowuje do nadania, dodając odpowiednie znaki sterowania i adresowe,
12/ inicjuje transmisję,
13/ przerywa nadawanie, gdy przesłany znak "Koniec transmisji",
14/ nadzoruje proces transmisji,
15/ przyjmuje potwierdzenie od odbierającej końcówki,
16/ nadaje ponownie komunikaty, które zostały nie otrzymane lub otrzymane z błędami.

B. Dla kierowania transmisji do właściwego urządzenia

Dla systemów wywoływanych:

1/ wywołuje odległe urządzenie,
2/ dokonuje przeglądu wywoływanych końcówek, /scan dial-up terminals/,
3/ odpowiada automatycznie, gdy wywoływany jest komputer,
4/ inicjuje transmisję po uzyskaniu połączenia,
5/ ustala typ urządzenia, które się włączyło.

Dla systemów przepytywanych:

6/ przepytuje końcówki, czy która ma coś do przekazania,
7/ przyjmuje odpowiedź końcówki i inicjuje odbiór,
8/ przesyła komunikat do końcówki, prosząc o pozwolenie nadawania,
9/ przyjmuje odpowiedź końcówki i inicjuje nadawanie,
10/ zwalnia linię po zakończeniu transmisji,
11/ przerywa kontakt, jeśli urządzenie nie odpowiada /np.w ciągu 15 sekund/,
12/ aktualizuje listę przepytywań, jeśli urządzenie jest uszkodzone lub wyłączone z sieci,
13/ jeśli stosowane jest przepytywanie, zmienia adres urządzenia, gdy urządzenie pod tym adresem zawodzi /hub polling/,
14/ koryguje błędy w adresowaniu.

Dla systemów w pętli:

15/ ustala i utrzymuje synchronizację pętli,
16/ umieszcza znaki we właściwych szczelinach transmisyjnych[x/],
17/ odbiera znaki z właściwych szczelin transmisyjnych,
18/ wykrywa błędy po odbiorze i poleca urządzeniu powtórne nadanie,
19/ przyjmuje powiadomienie o błędach w nadawaniu i powtarza nadawanie.

C. Wymagania logiczne dla podstawowego wejścia/wyjścia zbioru /które może być lub nie być włączone do oprogramowania transmisji danych/:

1/ przesuwa mechanizm dostępu do właściwej ścieżki,
2/ szuka na ścieżce właściwego zapisu,
3/ odczytuje zapis,
4/ wpisuje zapis,
5/ sprawdza operację wpisywania przez powtórne odczytanie zapisu i porównanie,

---

x/ w oryginale "transmission slots" /przyp.red./

6/ znajduje alternatywną ścieżkę, jeśli wpisanie nie udaje się,
7/ sprawdza operację czytania przy pomocy kodu wykrywającego lub korygującego błędy. Jeśli stosowany jest kod korekcji błędów, podejmowane jest usiłowanie automatycznego poprawienia zapisu, który jest odczytywany.

D. Metody adresowania zbiorów /do adresowania zbiorów stosuje się wiele różnych technik, między którymi są podane niżej/:

1/ przeszukiwanie szeregowe,
2/ przeszukiwanie binarne,
3/ adresowanie bezpośrednie przy użyciu konwersji arytmetycznej,
4/ adresowanie bezpośrednie przy użyciu randomizacji,
5/ adresowanie indeksowo-sekwencyjne /zbiór jest sekwencyjny według klucza, dla którego istnieje indeks/,
6/ indeksowe nie-sekwencyjne /jeśli zbiór nie jest sekwencyjny względem indeksowanego klucza, indeks będzie większy/,
7/ indeksy wtórne lub wielokrotne /w każdej pozycji jest więcej niż jeden klucz, tak aby był do niej dostęp przy użyciu więcej niż jednego parametru wejścia/,
8/ listy odwrócone /inverted/,
9/ zawarte odsyłacze /embedded pointers/ między zapisami "ojciec" i "syn",
10/ zawarte odsyłacze między zapisami w hierarchii wielopoziomowej,
11/ zawarte odsyłacze między odrębnymi zbiorami,
12/ pierścieniowa struktura zawartych odsyłaczy,
13/ któreś z czterech wyżej wymienionych, ale z odsyłaczami usuniętymi do osobnego modułu, a nie zawartymi /embedded/.

E. Ustalanie harmonogramów i przydział zasobów:

1/ przydzielanie buforów transmisji. Dynamiczna alokacja buforów pozwala na wydajne wykorzystanie pamięci głównej,

2/ przydzielanie miejsca w pamięci głównej dla zapisów w zbiorach może być dokonywane w podobny sposób,

3/ przydzielanie pamięci dla programów,

4/ wczytywanie programów,
5/ obsługa kolejek dla linii,
6/ obsługa kolejek dla zbiorów,
7/ obsługa kolejek dla programów,
8/ obsługa elementów przesyłanych do kilku różnych miejsc przeznaczenia,
9/ ustalenie priorytetów, tj. nadawanie pierwszeństwa pewnym typom komunikatów,
10/ funkcje przechowania i późniejszego przekazania /store and forward/,
11/ komutowanie komunikatów /p. lista funkcji na następnych stronach/.

F. Rozdział danych "logicznych" od "fizycznych":

/Termin "logiczne" odnosi się do widzianego przez programistę zastosowań obrazu danych, który może odbiegać od fizycznej natury danych/.

1/ łączenie kilku zapisów "logicznych" w jeden zapis "fizyczny" dla umieszczenia w pamięci,
2/ wyodrębnianie "logicznych" komunikatów po transmisji,
3/ wyodrębnianie "logicznych" zapisów z zapamiętanych danych,
4/ konwersja odsyłaczy "logicznych" w odsyłacze fizyczne zawarte w danych,
5/ "przekładanie" obrazu struktury danych widzianego przez programistę na fizyczną strukturę danych i vice versa,
6/ konwersja odnośników do pamięci wirtualnej w odnośniki do pamięci rzeczywistej.

G. Kompresja danych:

1/ blokowanie i rozblokowywanie komunikatów i zapisów,
2/ redagowanie komprymujące danych,
3/ konwersja zakodowania danych dla zmniejszenia ich objętości do przekazania lub transmisji, a następnie dla przywrócenia oryginalnej postaci, gdy potrzeba,

4/ transmitowane mogą być kody wstępnie skomponowanych elementów danych, na podstawie których elementy te zostaną wygenerowane w odległym urządzeniu /co znacznie redukuje objętość transmitowanych danych/.

H. Gromadzenie danych statystycznych:

1/ prowadzenie statystyki korzystania z zapisów danych dla umożliwienia wydajnego "strojenia" bazy danych,
2/ prowadzenie statystyki woluminów ruchu,
3/ prowadzenie statystyki błędów na linii,
4/ prowadzenie statystyki uszkodzeń linii.

I. Sterowanie zewnętrzną jednostką odpowiadającą /udzielającą informacji/

1/ sterowanie zewnętrzną jednostką generującą odpowiedzi,
2/ sterowanie zewnętrzną jednostką formującą dokumenty i obrazy ekranowe,
3/ sterowanie zewnętrzną jednostką odpowiadającą głosem.

J. Postępowanie awaryjne:

1/ generowanie kolejnej numeracji komunikatów dla ułatwienia procedury odzyskania treści komunikatów,
2/ podprogramy odzyskiwania związane z kolejną numeracją,
3/ rejestrowanie komunikatów /dla późniejszego odnalezienia i odtworzenia/,
4/ podprogramy odzyskiwania związane z rejestracją komunikatów,
5/ inicjowanie wywołania numerowego /dialing/ w przypadku uszkodzenia linii dzierżawionej,
6/ sporządzanie sum kontrolnych wchodzących transakcji,
7/ odpowiadanie końcówkom, gdy częściowo zawodzi system centralny/,
8/ wykonywanie periodycznych,programowych zatrzymań kontrolnych /checkpoints/ i inicjowanie wznowienia po wystąpieniu usterki,

9/ przetwarzanie zastępcze /fallback/ przy operacjach w czasie rzeczywistym, kiedy zawodzi jednostka pamięci masowej,

10/ procedury rekonstrukcyjne po masowej utracie danych,

11/ procedury rekonstrukcyjne dla poprawienia odosobnionych zapisów,

12/ kontrole dokładności dla zapobieżenia przypadkowym lub umyślnie spowodowanym stratom,

13/ działania zabezpieczające przed tym, aby po okresie usterki system nie zaniedbał aktualizacji odnośnego zapisu albo przypadkowo nie aktualizował podwójnie.

K. Funkcje zabezpieczenia

1/ identyfikacja końcówki nadsyłającej materiały wejściowe,
2/ identyfikacja końcówki przed przesłaniem materiałów wyjściowych,
3/ identyfikacja użytkownika końcówki, na przykład za pomocą kodu indywidualnego,

4/ tablice upoważniające, wskazujące co danemu użytkownikowi wolno robić /jakiego rodzaju komunikaty może wprowadzać, jakich programów wolno mu używać lub jakie dane może odczytywać albo wpisywać/,

5/ blokady danych. Blokada może dotyczyć pojedynczych zapisów, pojedynczych pól lub całych zbiorów /to ostatnie jest na ogół nie wystarczające/,

6/ użycie kryptografii do tajnych komunikatów.

L. Diagnostyka:

1/ diagnostyka sprawdzania prawidłowej pracy linii i końcówki z innej, oddalonej końcówki,

2/ próby krzyżowe /cross-parching/ dla sprawdzenia, czy sprzęt i oprogramowanie komputera sterującego linią pracują prawidłowo,

3/ oprogramowanie dla wysyłania i automatycznego odbioru odpowiedzi diagnostycznej z odległego urządzenia,

4/ ustalenie, które urządzenia nie działają na linii wielourządzeniowej,

5/ utrzymywanie w aktualności tablic stanu sieci i urządzeń.

M. Funkcje dialogowe /na ogół dostosowane do problemu/

1/ kontrola sensowności na wejściu,
2/ kontrola kompletności na wejściu,
3/ redakcja kontekstowa wejścia,

4/ kartkowanie, przesuwanie strony, szperanie, przeszukiwanie indeksowe i inne operacje przeglądania, ale nie przetwarzania danych,

5/ dialogi z operatorem przed użyciem programu zastosowania, np. dla zbierania danych.

N. Możliwości językowe

1/ makrorozkazy wejścia/wyjścia dla programisty zastosowań, możliwe w języku wyższego rzędu, jak COBOL lub PL/1,

2/ programista zastosowań powinien mieć możność używania symboli dla oznaczania końcówki lub zbioru,

3/ obraz zarówno struktury danych w zbiorach, jak i struktury przekazywanych komunikatów, jaki widzi programista zastosowań, powinien być obrazem "logicznym", który ma prawo różnić się od rzeczywistości fizycznej,

4/ język opisu treści teleprzetwarzania, którym posługuje się analityk systemów przy generowaniu programu sterowania siecią,

5/ język opisu bazy danych, którym posługuje się analityk systemów dla określenia struktur zapisu i elementu danych oraz zależności pomiędzy zapisami i elementami danych. Język powinien formułować odrębnie logiczne i fizyczne opisy bazy danych,

6/ język użytkownika, umożliwiający przepytywanie bazy danych, a w niektórych przypadkach pozwalający na jej aktualizację.

Tzw. "kierownicze" języki przepytywania bazy danych mogą generować skomplikowane przeszukiwanie zbiorów przez stosunkowo proste sformułowanie wymagań i potrzeb.

XII. AKTUALNE PRZYKŁADY

A. Przedsiębiorstwo użyteczności publicznej, dostarczające gaz do wielkiego miasta na wybrzeżu Nowej Anglii oraz otaczających gmin miejskich /w sumie 46/ miało obrót brutto 75,5 mln dolarów i dochód przed opodatkowaniem 5,4 mln dolarów w 1971 r. Przedsiębiorstwo ma obecnie około 325 tys. odbiorców. W 1970 r. połączyło się z małym przedsiębiorstwem produkcji gazu, a inna fuzja, z trzema dalszymi przedsiębiorstwami o 25 tysiącach odbiorców, została dokonana w styczniu 1972 r. Przedsiębiorstwo, które jest zarejestrowane na giełdzie nowojorskiej, jest z kolei kontrolowane przez holding, na którego czele stoi młody,postępowy prezydent. Popiera on nowe idee w tym znanym z konserwatyzmu i zacofania przemyśle i chętnie wypróbowuje nowe rozwiązania, jeśli są dobrze przemyślane. Holding, jednakże, choć zachowuje sprzyjający klimat, nie bierze udziału w sugerowaniu lub wdrażaniu nowości.

Historia

Przedsiębiorstwo, podobnie jak wiele innych, od lat posługiwało się sprzętem małej mechanizacji, potem przeszło na system IBM 650, potem na serię 1400, a wreszcie,w styczniu 1970 r. zdecydowało się na śmiały krok przejścia na systemy w czasie rzeczywistym z maszyną 360/50. Później, awansowano do maszyny 370/55.

Decyzja przejścia na czas rzeczywisty była, jak stwierdza wydział obliczeniowy, "trudna do uzasadnienia czym innym niż lepszymi usługami /dla naszych klientów/. Większość korzyści

jest nieuchwytna, a nawet mogą one nie być bezpośrednio oczywiste". Po decyzji przejścia na czas rzeczywisty przedsiębiorstwo spędziło 6 miesięcy na szczegółowym badaniu i porównywaniu różnych wariantów systemów. Oprogramowanie własnej roboty, produkty niezależnych przedsiębiorstw programujących oraz pakiety producentów sprzętu zostały zbadane z trzech głównych punktów widzenia:

1/ czy system już pracował zadowalająco?

2/ czy wdrożenie było odczuwalnie bardziej kłopotliwe, kosztowne i czasochłonne niż przewidywano?

3/ czy firma uzyskała to, na co liczyła, w stosunku do rzeczywiście wydanych pieniędzy?

Ponieważ duże przedsiębiorstwa pokrewnej branży,które opracowały swoje własne oprogramowanie, miały problemy i nadal dokładają do interesu, zdecydowano zakupić gotowy pakiet. Wybór został zawężony do systemów IMS i CICS produkcji IBM.

System ma być przeznaczony do obsługi klientów i reklamacji i ma być w stanie dostarczyć szybką odpowiedź na wszelkie pytania klientów, dotyczące usług i rachunków. Głównym kryterium projektowania był krótki czas reakcji przy czerpaniu informacji z przechowywanych zapisów o charakterze historycznym. Ze względu na wielką liczbę abonentów i duży ruch usług informacyjnych, potrzebna była baza danych z dobrym teleprzetwarzaniem.

Wydział obliczeniowy rozpoczął od przyglądania się IMS, ale wstępne próby i wyliczenia wykazały, że duże wymagania w stosunku do pamięci rdzeniowej, stosunkowo niewielka liczba końcówek, które system mógł obsłużyć /wiosną 1970 r./ i słaby czas reakcji na pytania tych końcówek, są poważnymi wadami tego systemu. Pakiet diagnostyczny GPSS, oferowany przez producenta jako narzędzie projektowania, użyty był do symulowania ruchu danych, z którym system będzie musiał sobie radzić.

Bieżąco system ma maksymalny czas reakcji -trzy sekundy, w okresach największego obciążenia. Obsługuje on w tej chwili 48 końcówek 2260 i 3 końcówki 2740. Czterdzieści jeden 2260-tek znajduje się w biurze obsługi i reklamacji i są one używane do odpowiadania na zapytania klientów, dotyczące usług lub rachun-

ków. Dwie znajdują się w sekcji obliczeniowej /dla użytku wewnętrznego/, dwie w sekcji kredytu i inkasa, gdzie zainstalowane będą dalsze, a dwie używane są dla służby dyspeczerskiej ekip naprawczych, nowych połączeń, odłączeń i innych problemów. Ma być podłączonych 16 dalszych jednostek 2260, większość w nowych sekcjach.

W tej chwili wszystkie końcówki zlokalizowane są w bezpośrednim sąsiedztwie biura centrali. Z uwagi na przyłączenie trzech nowych przedsiębiorstw i ich abonentów, zostaną zainstalowane końcówki zdalne. Nie została dotychczas podjęta decyzja, czy używane będą prywatne linie dzierżawione, czy publiczna sieć telekomunikacji. W toku są badania nad porównaniem kosztów i sprawności działania. Producent /IBM/ dopomaga przedsiębiorstwu oferując TPAD, narzędzie symulacji linii i ruchu, jako pomoc w rozbudowie systemu dla przyjęcia nowych abonentów i sprzętu, wynikającej z fuzji.

Jakie są przyczyny sukcesu? System uruchomiono z opóźnieniem nie przekraczającym dwóch miesięcy i zdobył on sobie popularność w przedsiębiorstwie.

Wydział obliczeniowy podaje cztery powody:

1/ próbowaliśmy namówić kierownictwo na konkretny, ograniczony zakres zastosowania, którego trzymaliśmy się twardo. Ich oczekiwania zostały spełnione, a fakt, że system może dokonać więcej niż oczekiwano, mile wszystkich zaskoczył,

2/ sprzedawaliśmy ideę lepszej obsługi klientów i nie próbowaliśmy uzasadnić systemu ekonomicznie,

3/ od początku wdrażania staraliśmy się wciągnąć do pracy personel wydziału, który użytkuje system, dzięki czemu spotkaliśmy się z dobrym i szybkim przyjęciem,

4/ pozytywne doświadczenia ludzi z innych wydziałów, którzy pomagali w uruchomieniu systemu, stały się znane w przedsiębiorstwie tak, że wydział obliczeniowy może obecnie oczekiwać zrozumienia i pomocy w całej korporacji.

Przedsiębiorstwo jest obecnie w trakcie automatyzacji wydziału inżynieryjnego, przy pomocy maszyny IBM/370.

Uprzednio, rozmieszczenie podłączeń do sieci gazowej pokazane było na ogromnej mapie. Czasami, po zaprzestaniu dostawy gazu, sprzęt należący do przedsiębiorstwa pozostawał na terenie abonenta zapomniany i zagubiony, choć ciągle figurujący w księgach. Czasami również nie było można znaleźć umiejscowienia poszczególnych podłączeń, a obecność lub nieobecność przyłącz i zaworów nanoszona bywała na mapę w sposób niekompletny lub błędny. Kiedy ekipa naprawcza w terenie nie mogła znaleźć określonego punktu, dzwoniła do biura i następowało długie przeszukiwanie zbiorów i rejestrów, podczas gdy ekipa musiała czekać bezczynnie lub wracać ponownie do tej samej pracy. I jedno, i drugie było stratą czasu i pieniędzy. Obecnie przystąpiono do opracowania numerycznego trzech milionów zapisów dotyczących wykazu usług oraz sprzętu przedsiębiorstwa w terenie. Każdy element sprzętu, gazomierz, zawór itp. będą miały nadany numer trzynastocyfrowy, w którym mieścić się będą również współrzędne x i y dla bardzo dokładnego, automatycznego zlokalizowania na mapie sieci. W przyszłości, ekipy naprawcze w terenie być może będą mogły zabierać ze sobą przenośne końcówki z dostępem do kartotek komputera; w tej chwili ciągle jeszcze muszą porozumiewać się telefonicznie, a operator jednostki 2260 pomaga im w znalezieniu potrzebnych danych.

Fuzje

Pierwsza fuzja miała miejsce w 1970 roku, kiedy została przyłączona do przedsiębiorstwa mała firma o całkowicie odmiennym i mniej nowoczesnym systemie księgowości. Księgi małej firmy były dokładnie studiowane, podczas, gdy dokonywano ich konwersji odręcznie. Wkrótce wydział obliczeniowy opracował program, pozwalający na automatyczną konwersję zapisów. Program ten pracował tak dobrze, że przy obecnej fuzji z trzema dalszymi firmami średniej wielkości, uzyska się znaczną oszczędność czasu i pieniędzy. /Te trzy przedsiębiorstwa miały system księgowania podobny do owej małej firmy/.

Rezerwa awaryjna i wznawianie pracy /Back-up and recovery/

Obecnie, w przypadku usterki komputera system obsługiwany jest ręcznie. Operatorzy obsługujący telefony reklamacyjne ko-

rzystają z dokumentacji na papierze, która jest ciągle aktualizowana, w skróconej formie, przez komputer. Jeśli nastąpi jakaś transakcja, jest ona rejestrowana odręcznie, a później wprowadzana do maszyny z końcówek przez kilku troskliwie dobranych, zaufanych i przeszkolonych operatorów, korzystających z końcówek o ograniczonym dostępie. /Prawo nakazuje, aby rejestrowany był dokładny czas, w którym została zgłoszona reklamacja, tak więc specjalne końcówki pozwalają na wprowadzenie informacji z właściwym czasem ich przyjęcia, a nie faktycznym czasem kiedy są fizycznie wprowadzane do komputera/. Pakiet CICS zapewnia duże bezpieczeństwo pracy przy pomocy haseł, tablic upoważniania i sprawdzania podpisów. Tak więc, tylko określeni operatorzy, korzystający z określonych końcówek, mogą wprowadzać dane do systemu z chwilą wystąpienia uszkodzenia.

Wydział obliczeniowy ma nadzieję uzyskać czytnik optyczny. Wówczas, w czasie przerw pracy maszyny, operatorzy będą mogli wypełniać formularze odręcznie. Następnie będą one mogły być automatycznie odczytywane i wprowadzane do komputera, z chwilą uruchomienia systemu. Dokumentacja na papierze będzie wkrótce niemal całkowicie wyeliminowana, ponieważ taśmy komputera są obecnie umieszczane na mikrofiszkach. W czasie, kiedy przedsiębiorstwo dokona przeprowadzki w październiku 1973 r., wydział obliczeniowy ma nadzieję posiadać już kompletną bibliotekę na mikrofiszkach.

Przyszłość

W miarę jak następował rozwój systemu przedsiębiorstwa, zaczęto przyglądać się nowszym wersjom IMS. Firma, w idealnym przypadku potrzebowałaby teleprzetwarzania, obsługi końcówek i krótkiego czasu reakcji, zapewnianych przez CICS,natomiast możliwości obsługi bazy danych, jakimi dysponuje IMS. Z drugiej strony CICS, jakkolwiek ma słabsze oprogramowanie bazy danych i mniejsze wymagania w stosunku do pamięci, może obsłużyć więcej końcówek i zapewnia znacznie krótszy czas reakcji. Jest to szczególnie cenne przy niecierpliwych,a czasem wzburzonych klientach.

W czerwcu 1970 roku podjęto decyzję wprowadzenia CICS i natychmiast rozpoczęto prace projektowe. Jako termin uruchomienia przyjęto czerwiec 1971 r. W rzeczywistości system był w ruchu na początku sierpnia 1971 r. System był wdrażany przez czterech programistów. Jeden z nich pracuje w przedsiębiorstwie od wielu lat, dwóch od ponad dwóch lat, a jednego przyjęto do pracy i "po prostu został programistą". Superintendent działu informacji, sam były programista, opisuje ten zespół jako "wyważony". Jeden z programistów to jest "prawdziwy technik", dwaj opisywani są jako "dobrzy programiści, którzy związują luźne końce i wypełniają luki w myśleniu tego technika", a czwarty programista określany jest jako "przeciętny", chętny do sumiennej pracy nad każdym zadaniem jakie otrzyma, nie wyłączając tych, które inni uważają za nudne i nieciekawe.

Grupa pracowała dobrze jako zespół. Można było zatrudnić więcej programistów, ale dyrektor usług informacyjnych, były programista, uważał, że jeśli tych czterech pracuje dobrze w zespole, byłoby głupotą naruszać tę harmonię, a zbyt wielu ludzi spowodowałoby zamieszanie i trudniej byłoby ich nadzorować i ściśle z nimi współpracować.

Uzyskano bardzo ścisłą współpracę i koordynację z użytkownikami - to jest tymi wydziałami, które miały korzystać z systemu. Powodzenie tej współpracy należy przypisać, co najmniej w pewnym stopniu, osobowości dyrektora systemów informacyjnych.

Wydział stosunków z klientami, który odpowiada na wszystkie telefony w sprawie usług i rachunków, poproszono o zaprojektowanie jednego "ekranu", "formularza", który mógłby zawierać informacje wystarczające do odpowiedzi na większość zapytań dotyczących rachunków. Drugi ekran "historyczny", dla informacji rzadziej potrzebnej, dotyczącej przeszłości, również został zaprojektowany. Wreszcie sekcja obsługi zaprojektowała ekran,który miał dopomóc przy załatwianiu większości telefonów, dotyczących obsługi i napraw.

Z uwagi na ograniczenie do 480 znaków pojemności ekranu końcówki 2260, wydziały użytkowników wypracowały, przy pomocy wydziału obliczeniowego, specjalny czteroznakowy kod nazw różnych ekranów. Na przykład wezwanie BILL wywołuje pierwszy ekran

przed urzędnikiem dokonującym obsługi klienta. Jeśli, z jakiegoś powodu potrzebne są dalsze informacje, u samego dołu ekranu "BILL" widnieje informacja, jak otrzymać następny ekran: "proszę wystukać "HIST" po dalsze informacje".

Personel obsługi klientów, korzystający z końcówek 2260 lubi ten system. Został on zaprojektowany tak, by mogli z niego łatwo korzystać i tak rzeczywiście jest. Wystąpił interesujący efekt uboczny. Operatorzy końcówek uważają, że praca ich została awansowana i zażądali trzydziestodolarowej podwyżki tygodniowo.

Psychologicznie, dzięki temu, że poproszono ich od początku do pomocy i że ich użyteczne sugestie zostały przyjęte, pracownicy wydziałów użytkujących stali się od początku projektu ważnym sprzymierzeńcem: rzeczywiście pragnęli oni sukcesu projektu, przy którym pracowali.

WNIOSKI

1. Użycie prostego, ale kompletnego pakietu oprogramowania wydaje się być kluczem do bezbolesnego wdrożenia, bez nadmiernych poślizgów w harmonogramie.

2. Dobrze wyważona grupa, pracująca jako zespół, wykona pracę szybciej i lepiej. Użycie małego pakietu pozwoliło na utworzenie małego zespołu.

3. Dzięki zaangażowaniu do zaprojektowania dialogu ludzi, którzy mają korzystać z systemu, wydział obliczeniowy zapewnił sobie, że:

a/ system może, i będzie, odpowiadać na pytania, na które powinien odpowiadać - jest przystosowany do swojej pracy,

b/ dialogi z końcówką i ekranem są prostę i nieskomplikowane, a zatem nie irytują operatorów, którzy mogą skoncentrować się na miłej i szybkiej obsłudze klientów.

4. Jeśli pracownicy uważają, że obsługa terminalu awansuje ich w hierarchii zawodowej, mogą zechcieć więcej pieniędzy lub innych przywilejów.

5. Dzięki szczegółowamu przestudiowaniu doświadczeń, nie tyle całego rynku, ile podobnych innych przedsiębiorstw, posługujących się systemami w czasie rzeczywistym, przedsiębiorstwo mogło rozważyć plusy i minusy różnych podejść.

a. Uznano, że lepszy będzie gotowy pakiet, ponieważ pozwala na szybsze uruchomienie systemu /a zatem na zaoszczędzenie pieniędzy/.

b. Zbyteczna była rozbudowa wydziału programowania lub angażowanie niewypróbowanego i kosztownego personelu, który mógłby z trudnością zgrać się w zwarty zespół.

c. Dwa przedsiębiorstwa, które opracowały swoje własne oprogramowanie nadal mają kłopoty, a wdrożenie okazało się znacznie bardziej czasochłonne niż przypuszczano. Obie te firmy znacznie przekroczyły wstępne wyceny kosztowe.

6. Dzięki wdrożeniu systemu, stosunkowo prostego, w terminie i zgodnie z planem /tzn. system wykonuje to, co do wykonania zaplanowano/ wydział obliczeniowy zyskał zaufanie kierownictwa i ma obecnie wolną rękę do wprowadzania systemu w innych działach przedsiębiorstwa, takich jak wydział inżynieryjny, co niemal na pewno przyniesie znaczne oszczędności.

7. Nie robiono poważniejszych usiłowań w kierunku ekonomicznego uzasadnienia systemu. Jako zasadniczy cel postawiono zadowolenie klienta z obsługi.

B. Szybko rozwijający się bank w południowo-wschodniej części Stanów Zjednoczonych z trudem radził sobie z nawałem klientów, i związanym z tym napływem zapytań, często żądających natychmiastowej informacji o saldach, pożyczkach, kredycie itp.W 1967 roku zarząd banku zdecydował zastąpienie starego sprzętu serii 1400 systemu 360, model 40, pracującym pod OS/MFT przy obsłudze teleprzetwarzania pakietem QTAM. Początkowo komputer miał wykonywać dwa główne zadania:

1. system odpowiedzi głosowej dla okienek kasowych /wyszukiwanie sald klientów i inne informacje/,

2/ system monitorów ekranowych dla wydziału kredytów, umożliwiający także odpowiadanie na pytania sklepów i instytucji,związane z posługiwaniem się kartami kredytowymi /z paskiem magnetycznym/, wydawanymi klientom przez bank.

W miarę,jak wydział obliczeniowy banku realizował te dwa początkowe zadania, nabierał doświadczenia w pracy systemu. Wkrótce wydział zwrócił uwagę na inne możliwości zastosowania posiadanego sprzętu. Obecnie wydział obliczeniowy dokonuje na zlecenie obliczenia około 800 list płacy i działa jako konsultant dla lokalnych przedsiębiorstw w zagadnieniach związanych z komputeryzacją, ponieważ dysponuje najbardziej nowoczesną instalacją na tym terenie. Niezależnie od tego, znacznej rozbudowie uległy usługi na terenie samego banku. Do stycznia 1973 r. wydział obliczeniowy pracował na budżecie. Od tego czasu przeszedł na rozrachunek gospodarczy. Nowy ośrodek usług obliczeniowych ma 46 osób personelu, z których około 20 to programiści, wszyscy o trzech lub więcej latach pracy w wydziale. Większość może programować w asemblerze.

Obecny sprzęt składa się z jednego komputera 360/40, jednego 360/65 i jednego 370/155. Drugi 370/155 jest dzierżawiony do czasu otrzymania zamówionego już 370/158. W styczniu 1973 r.zakończono wymianę 130 dzierżawionych końcówek 2260 na 150 nowych, również dzierżawionych końcówek 3270. Także w filiach banku większość sprzętu z serii 1400 jest zastępowana sprzętem systemu 3. Kiedy w 1967 roku podejmowano decyzję zakupu sprzętu 360 powstało pytanie: kto opracuje dla niego oprogramowanie? Kierownictwo zdecydowało, że praca powinna zostać wykonana przez rozbudowany zespół własny. Ludzie ci lepiej znali operacje banku, jak również znali wymagania rewidentów i prawa stanowego, a także potrzeby personelu obsługującego i klientów.

System odpowiedzi głosowej

Jakkolwiek część systemu, dotycząca obsługi głosem, korzystała z opracowanego przez IBM pakietu HEAR, w okresie instalacji nie był jeszcze dostępny kompletny program - monitor. Trzeba było użyć dla okienek kasowych jednostek IBM 7770. Personel wy-

wydziału obliczeniowego banku musiał napisać swój własny monitor. Został on napisany w asemblerze 360 i zajął około 1000 wierszy kodu. Został napisany przez dwóch nadzwyczaj uzdolnionych programistów przy pracochłonności nieco ponad trzech osobomiesięcy. Uruchomienie zajęło znacznie więcej czasu. Program monitor obsługuje obecnie 32 linie telekomunikacyjne i zajmuje 56 K pamięci /łącznie z pamięcią na bufory/.

Kiedy zapytano ich, czy zrobiliby to samo obecnie w ten sam sposób, czy też woleliby zakupić dostępny obecnie pakiet dla tego zastosowania, programiści odpowiedzieli bez wahania,że woleliby zakupić gotowy pakiet.

System wideograficzny

System kredytowy, korzystający z monitorów ekranowych, miał być również sterowany przez QTAM i grupa programistów banku zasiadła do opracowania oprogramowania problemowego dla końcówek 2260, które miały znajdować się w głównym biurze w Atlancie, a następnie również w czterech odległych miejscach,w całym stanie.

Potrzeba zmiany

W miarę, jak rosłą liczba operacji i liczba klientów, system stawał się przeciążony wielkim wolumenem zapytań, korzystających z odpowiedzi głosowych i z monitorów ekranowych, których pod koniec było aż 130. Mała wydajność systemu stawała się widoczna, w miarę jak gwałtownie zaczął rosnąć czas reakcji.Klienci musieli czekać na otrzymanie swojego salda po 5 i więcej minut. Użytkownicy kart kredytowych oraz sklepy ich obsługujące, musiały czasem czekać po 15 do 20 minut na potwierdzenie kredytu za towar wartości 15 dolarów. Średni czas zwłoki wynosił 5 do 10 minut. W obliczu pogarszających się stosunków z klientami i obawiając się straty zysków , kierownictwo zostało zmuszone do działania. Potrzebny był, i to prędko, system zapewniający krótki czas reakcji na szybko rosnącą liczbę i różnorodność pytań nadsyłanych ze zdalnych końcówek.

Wśród rozważanych alternatyw były: MINERVA, pakiet DBDC o cenie zakupu 25 tysięcy dolarów oraz CICS. Zdecydowano zastosować CICS /program sterujący IBM/, pozwalający na równoczesną obsługę różnych typów zapytań /multi-thread operation/.

Zalety były następujące:

1/ oszczędność - zbędna była nadmierna rozbudowa pamięci rdzeniowej,

2/ kompatybilność - większość oprogramowania opracowanego na miejscu oraz program-monitor odpowiedzi głosem w okienkach kasowych, mogły być użyte pod CICS bez większych zmian,

3/ różnorodność - różnorodność zastosowań, z którymi mógł sobie poradzić system została znacznie zwiększona,

4/ wolumen - możliwości obsługi zapytań można było znacznie zwiększyć bez wydłużania czasu rakcji,

5/ szybkość - można było uzyskać zadowalające czasy reakcji, nawet przy liczniejszych zastosowaniach i znacznie większym ruchu.

Jakkolwiek pakiet CICS pozwala na korzystanie z haseł, tablic upoważniania użytkowników itp., ta część pakietu dotychczas nie była używana. Typy zapytań, jakie mogą stawiać operatorzy końcówek, są ograniczone, a informacje które są zastrzeżone jako poufne, są specjalnie oznaczane, tak by nie mogły być wydobyte. To co wchodzi, nie zawsze wychodzi. Własna lista płacy banku oraz dane personalne zostaną jednak podłączone bezpośrednio i wówczas dostęp do pewnych elementów będzie ograniczony.

Filie banku w całym stanie przechodzą ze sprzętu serii 1400 na sprzęt systemu 3. Filie prowadzą swoją własną księgowość i własne rachunki bieżące, ale muszą używać zdalnego systemu zapytywania dla uzyskania informacji, dotyczącej rachunków bieżących prowadzonych w innych filiach i centrali pożyczek, rachunków oszczędnościowych, informacji poufnej itp. z centralnej bazy danych w głównym oddziale. Nowe informacje, dotyczące rachunków, mogą być wprowadzane do systemu bezpośrednio przez końcówki 3270. Z chwilą kiedy zostanie wprowadzony zdalny system MICR/Magnetic Ink Character Recognition/ - rozpoznawania znaków

pisanych atramentem magnetycznym - wszystkie rachunki w filiach będą również podłączone on-line do centralnej bazy danych. Wówczas informacja o dowolnym rachunku będzie mogła być uzyskana w dowolnej filii /w tej chwili tylko rachunki bieżące w centrali i nowe rachunki są dostępne on-line/.

W obecnym czasie końcówki 3270 mają bezpośredni dostęp do zbiorów centralnej bazy danych z dziewięciu odległych lokalizacji w całym stanie. Cztery filie mają jeszcze końcówki 2260, a sześć otrzyma sprzęt systemu 3 z bezpośrednim dostępem do głównych zbiorów za pomocą sześciu dzierżawionych linii, transmitujących z szybkością 7200 bitów na sek. Końcówki 2260 transmitują 2400 bitów na sek., a 3270 transmitują 4800 bitów na sek. Dla systemu 3 rozważano użycie mikrofal, jednak najprawdopodobniej użyte będą linie dzierżawione. Wszystkie linie, zgodnie z prawem stanowym, muszą być dzierżawione od miejscowej kompanii telefonicznej. Używane są modemy Western Electric i C2 Conditioning. Okienka kasowe nie są podłączone jeszcze on-line,z wyjątkiem wyszukiwania informacji, ponieważ brak jeszcze dostatecznego sprzętu dla obsłużenia ich wszystkich funkcji.Nie jest również jeszcze on-line system wypłaty gotówki, powiązany z planem kart kredytowych o paskach magnetycznych. Końcówka Docutel może zrobić to wszystko, co końcówka w okienku kasowym: przyjmować depozyty na rachunki bieżące i oszczędnościowe, dokonywać przelewu między rachunkami i wypłacać 25 dolarów lub 50 dolarów dziennie dla posiadaczy kart, mających dobry kredyt. Z czasem, gdy system przejdzie na pracę bezpośrednią tak, że cała gotówka może być wycofywana z rachunku na życzenie, zamiast tylko krotności 25 dolarów.

Problemy

Na początku głównym problemem była częstotliwość błędów na liniach. Pospolite były częstotliwości dochodzące do 5 błędów na minutę,na trzech liniach, a "więcej błędów,kiedy pada". Kiedyś pracownik obsługi "po prostu przeciął linię". Jednakże synchroniczna transmisja binarna jednostek 3270 oraz ulepszona obsługa przez kompanię telefoniczną obniżyły częstotliwość błędów do około 1 błędu na 15 minut,na jednej linii. Oprogramo-

wanie dostarczone przez producenta było wykonane przez kilka różnych zespołów, które "wydają się nie rozmawiać ze sobą" tak, że integracja części oprogramowania była bardzo trudna dla programistów banku.

Jednym z głównych problemów, kiedy coś się psuje, jest "znalezienie kogoś, kto coś o tym wie /o teleprzetwarzaniu/". Każdy po prostu "stoi tylko, pokazując palcem na wszystkich innych".

Nie istnieje właściwa rezerwa awaryjna, na przypadek, gdyby wszystkie komputery banku były niesprawne, poza możliwością wypożyczenia urządzeń miejscowego centrum systemów terenowych IBM /IBM Field System Center/. Na szczęście system ma bardzo niewiele przestojów.

Inny problem związany z przestojami, to sprawa wrażliwych stosunków z klientami. W okresie przestoju klienci nie otrzymują odpowiedzi na swoje pytania, NIE dlatego, że informacji brak lub jest niedostępna, ale ponieważ kasjerzy i obsługa kredytu tak przywykli do systemu i tak są od niego zależni, że zapominają albo odmawiają korzystania z kartotek rezerwowych, które jest czasochłonne. Należy się dziwić, że bankowi nie udało się dotychczas nauczyć swoich kasjerów, aby telefonowali do obsługi rezerwowej, kiedy komputer "wysiądzie" /zbiory rezerwowe generowane są przez komputer na mikrofilmie i obsługiwane przez personel, postawiony specjalnie do tego, by odpowiadać na pytania telefonicznie, gdy system nie pracuje/.

Bank korzysta ostatnio z on-line, z centrum diagnostycznego IBM w Raleigh, North Carolina, z dobrym rezultatem. /Centrum w Raleigh wywoływane jest tarczą numerową, a następnie do linii należy podłączyć uszkodzone urządzenie. Centrum dokonuje na sprzęcie różnych prób, dokonuje diagnozy problemu, a następnie przepisuje "receptę" poprzez wydruk w lokalnym oddziale. Centrum przeprowadza również konserwację zapobiegawczą. Producent nie pobiera za te usługi dodatkowej opłaty.

Bank myśli już o użyciu IMS do zarządzania swoją rosnącą bazą danych, ma jednak zamiar zachować CICS dla potrzeb teleprzetwarzania. Nie można, niestety, używać naraz obu tych pakietów, z uwagi na ograniczoną pojemność pamięci.

BIBLIOGRAFIA

1. A Survey of Generalized Data Management Systems, CODASYL Systems Committee, ACM, Maj 1969 r.
2. Brackett, Gilbert R., Telephone Traffic Engineering Handbook, Telephone Publishing Company, Chicago, 1963 r.
3. Data Base Task Group, CODASYL Programming Language Committee, ACM, kwiecień 1971r.
4. Data Structures in Programming Languages, Proceedings of a Syposium, ACM SIGPLAN i University of Florida
5. Data Transmission, Western Union, 1961 r.
6. EIA Standard RS-232-A, EIA, New York, październik 1963 r.
7. Expenditure Patterns for Management Information Systems - 1971r,Diebold Research Program-Europe, Doc. No E-91
8. Feature Analysis of Generalized Data Base Management Systems, CODASYL Systems Committee, ACM, maj 1971 r.
9. GUIDE/SHARE Data Base Management Systems Requirement, New York, styczeń 1970 r.
10. Lefkowitz, Dawid, File Structures for On-Line Systems, Spartan Books, New York 1969 r.
11. Martin, James, Telecommunications and the Computer,Prentice-Hall, Englewood Cliffs, N.J. 1970 r.
12. Martin, James, Teleprocessing Network Organization,Prentice-Hall, 1970 r.
13. Martin, James, Design of Man-Computer Dialogues, Prentice-Hall, 1972 r.
14. Martin, James, Systems Analysis for Data Transmission,Prentice-Hall, 1972 r.
15. Pullen, Keats A.Jr. Design of Communications Systems, Hayden Book Co, New York 1963 r.
16. Schwartz, Leonard S., Principals of Coding, Filtering and Information Theory, New York Uniwersity Press, New York 1970 r.

Cena zł 92.-