II

KONFERENCJA UŻYTKOWNIKÓW MINIKOMPUTERA MERA-400

DO UŻYTKU WEWNĘTRZNEGO

GDAŃSK 1985

INSTYTUT OKRĘTOWY
POLITECHNIKI GDAŃSKIEJ

POROZUMIENIE UŻYTKOWNIKÓW
MINIKOMPUTERA MERA-400
PRZY POLSKIM TOWARZYSTWIE
CYBERNETYCZNYM
ODDZIAŁ W GDAŃSKU

II KONFERENCJA UŻYTKOWNIKÓW MINIKOMPUTERA MERA 400

Materiały

25 - 27 października 1984 r.

Komitet Organizacyjny Konferencji

Przewodniczący :
Andrzej Braniecki, Instytut Okrętowy PG

Sekretarz naukowy i redaktor materiałów :
Stefan Zieliński, PTC Oddział Gdański, Instytut Okrętowy PG

Sekretarz organizacyjny :
Grzegorz Cerkaski, Instytut Okrętowy PG

Rada Programowa Konferencji -
Tymczasowa Rada Programowa Porozumienia
Użytkowników Minikomputera Mera - 400

1. Andrzej Braniecki /przewodniczący/
2. Grzegorz Cerkaski /sekretarz/
3. Krystyna Pietrzak
4. Tadeusz Grall
5. Hanna Krzyszczuk
6. Kazimierz Jojczyk
7. Andrzej Stóś
8. Marian Waksman
9. Zbigniew Czerniak
10. Tomasz Rawiński
11. Stefan Zieliński

SPIS TREŚCI

x/ Referat ma być opublikowany w "Informatyce"

IV. SYSTEMY APLIKACYJNE

WSTĘP

II Konferencja użytkowników minikomputera MERA-400 odbyła się w dniach 25-26 października 1984 r. w Gdańsku w budynku NOT-u oraz 27 października w budynku Instytutu Okrętowego Politechniki Gdańskiej. W drugim dniu Konferencji odbył się pokaz zastosowania MERY-400 w Stoczni Remontowej "Radunia".

Konferencja zgromadziła około 200 uczestników i obserwatorów.

Na Konferencji wygłoszono 38 referatów i komunikatów. Ze względu na różnorodność tematyczną, wystąpienia zgrupowano w następujące sesje:

1. Systemy operacyjne
2. Bazy danych
3. Problemy sprzętowe
4. Języki programowania
5. Ogólny przegląd prac i zastosowań
6. Systemy aplikacyjne /dwie równolegle sesje/.

W trakcie Konferencji odbyła się prezentacja sprzętu firmy ICL. W drugim dniu Konferencji odbyło się zebranie plenarne członków Porozumienia użytkowników minikomputera MERA-400.

W podsumowaniu Konferencji można stwierdzić, że:

- Zrealizowano postulat I Konferencji - powołano, przy Polskim Towarzystwie Cybernetycznym Oddział Gdański oraz przy współudziale Instytutu Okrętowego Politechniki Gdańskiej, Porozumienie użytkowników minikomputera MERA-400, które rozpoczęło działalność w lipcu 1984 r.
- Nierozwiązany pozostaje model problemu sprzętu i serwisu.
- Występuje potrzeba doprowadzenia do podobnych narzędzi programistycznych na systemach operacyjnych typu CROOK i SOM. Program realizacji tzw. etapu zerowego rozwiązania tego problemu, wypracowany w ramach Porozumienia, przedstawiono i przedyskutowano w czasie sesji plenarnej dotyczącej systemów operacyjnych.

- Osiągnięto znaczny dorobek w zakresie języków programowania /FORTRAN, PASCAL, LOGLAN/.
- Utrzymuje się dalsza aktywność różnych ośrodków w zakresie oprogramowania aplikacyjnego.

Materiały pokonferencyjne zawierają referaty, komunikaty i oferty prezentowane na Konferencji i zgłoszone na piśmie. W materiałach zamieszczono także informację o działalności Porozumienia użytkowników minikomputera MERA-400. Ze względu redakcyjnych podział materiałów odbiega nieco od tematów poszczególnych sesji Konferencji.

I SYSTEMY OPERACYJNE

inż Bogdan Machowiak
Biuro Projektów Budownictwa Wiejskiego
Poznań, ul.Piekary 17
tel. 33-05-81 wew. 569

WYNIKI NARADY ROBOCZEJ PROJEKTANTÓW SYSTEMÓW OPERACYJNYCH DLA MERY-400

Minikomputer MERA-400, który jest najbardziej powszechną maszyną cyfrową eksploatowaną w Polsce - ilość wyprodukowanych jednostek centralnych ok. 800 szt /informacja Działu Zbytu producenta ERA/ - doczekał się wielu nowych implementacji systemów operacyjnych nie licząc znanego SOM-3 i jego mutantów produkcji Fabryki Mierników i Komputerów ERA w Warszawie, która jest jeszcze producentem tych maszyn - do końca 1984 r.

Te nowe systemy operacyjne dla MERY-400 powstały w kilku ostatnich latach, głównie w ośrodkach akademickich Warszawa, Gdańsk, Kraków, Kędzierzyn-Koźle- i dość skutecznie wypierają znany wszystkim użytkownikom /chociażby w początkowym okresie eksploatacji maszyny/ - SOM 3.

Opracowanie tych nowych systemów operacyjnych jest udaną próbę lepszego i pełniejszego wykorzystania procesora MERY-400 i jego urządzeń, niż to miało miejsce w znanym SOM-3.W opracowywaniu nowych systemów, projektanci kładli główny nacisk na wielodostęp i wieloprogramowość, lepsze wykorzystanie pamięci operacyjnej i zewnętrznej, jak również, rozwój i ulepszenie oprogramowania podstawowego - narzędziowego.

Do tej pory najbardziej znanymi nowymi opracowaniami są:

1. Uzdatniony Wielodostępny System Operacyjny SOM-3

opracowany w Instytucie Informatyki Uniwersytetu Warszawskiego. Istotą wielodostępu w tym opracowaniu jest przyznanie każdemu z zadań użytkowych pewnej /początkowej/ puli pamięci operacyjnej oraz wydzielenie fragmentu pamięci dyskowej w postaci sekcji roboczych przy czym zadania użytkowników pracują z cyklicznym podziałem czasu procesora.
Jedną z istotnych zmian w stosunku do SOM-3 jest możliwość korzystania z przestrzeni adresowej wynoszącej 64 k słów dla zadania użytkowego. Oferowane są jednocześnie środki pozwalające na dynamiczne rozszerzenie lub redukowanie przestrzeni adresowej zadania, co pozwala na elastyczne wykorzystanie pamięci operacyjnej maszyny dla pracy wielodostępnej.

Do zarządzania zawartością dyskowych kaset wymiennych służy podsystem - FLS /File System/. Nowy JOB CONTROL wyposażony jest w dyrektywy, których celem jest manipulacja zbiorami, "widzianymi" tylko przez danego użytkownika, który dodatkowo, może dostęp do swych zbiorów zabezpieczyć odpowiednim hasłem lub je zmienić.

Instytut Informatyki Uniwersytetu Warszawskiego z oprogramowaniem podstawowego oferuje następujące produkty:

- kompilator języka LOGLAN,
- kompilator języka PASCAL,
- makroassembler GASS, LINKEDYTOR,
- edytor tekstów EDM.

Omawiany system operacyjny eksploatowany jest obecnie w około 35 ośrodkach, które uprzednio używały SOM-3.

Pod tym systemem pracuje poprawnie całe oprogramowanie podstawowe i aplikacyjne wygenerowane pod standardem SOM-3 /wszystkie kompilatory, translatory, procesory itp /.

2. Drugim znanym systemem operacyjnym dla MERY-400 jest Wielodostępny System Operacyjny SOM-3.P, opracowany w Instytucie Informatyki Uniwersytetu Jagiellońskiego w Krakowie.

System ten jest podobnie zorganizowany jak poprzednio omawiany , nie jest jeszcze dystrybuowana wersja z dynamicznym podziałem puli pamięci operacyjnej dla zadań użytkowych. Przestrzeń adresowa dla każdego użytkownika jest określana w czasie generacji systemu.

Produkt ten posiada zewnętrzny File System dla przechowywania plików tekstowych i binariów oraz dostęp do niego realizowany jest przez nazwę użytkownika.

System operacyjny SOM-3.P posiada rozbudowany Parametryzowany Interpreter Komend, który realizuje makroinstrukcje między innymi: różnych wersji kompilacji i konsolidacji, formatowania programów źródłowych, obsługi różnych typów bibliotek, zmiany w konfiguracji terminali, itp.

Omawiany SOM-3.P eksploatowany jest obecnie w ok. 20 ośrodkach i praca wygerowanych programów binarnych i użytkowych, kompilatorów, procesorów pod standardowym SOM-3 realizowana jest poprawnie.

3. Następnym systemem, którego chciałbym przedstawić jest System Operacyjny DSM-1 opracowany w Instytucie Cieplnej Syntezy Organicznej "Blachownia" w Kędzierzynie-Koźlu.

Produkt ten jest systemem nowym, opracowanym tak, aby obsługa jego zadań przypominała jak najbardziej działanie standardu SOM-3. Spełnia on wszystkie funkcje SOM-3, pozwala wykorzystywać procesory SOM-3 oraz opracowane dotychczas oprogramowanie aplikacyjne w postaci binarnej jak i źródłowej.

DSM-1 jest systemem wielodostępnym, różni się od poprzednio omawianych tym, że gdy żądania zadań przekraczają wielkość dostępnej wolnej przestrzeni adresowej PAO, uruchamiane są automatycznie "wymiany" pomiędzy PAO a pamięcią dyskową, pozwalające na zwolnienie obszaru PAO dla aktualnie obsługiwanego zadania.

DSM-1 posiada prawie dwukrotnie rozszerzoną listę ekstrakodów. Oprócz prawie wszystkich ekstrakodów SOM-3 /za wyjątkiem operacji na sygnałach, REMOVE i CREATE / system posiada ekstrakody realizujące między innymi otwieranie dostępu do zbiorów dyskowych, uzyskiwanie i aktualizację informacji o zbiorach i sekcjach w zbiorze, tworzenie i kasowanie sekcji oraz wydruki informacji o różnych postaciach.

Integralną częścią DSM-1 jest podsystem zbiorów, który umożliwia korzystanie z sekcji dyskowych, które identyfikowane są poprzez ciąg nazw symbolicznych katalogu nadrzędnego oraz nazwą symboliczną sekcji. Podsystem zbiorów DSM-1 zapewnia między innymi automatyczne rozszerzenie w razie potrzeby obszaru przedzielonego do sekcji, dostęp do informacji zawierających datę utworzenia i ostatniego zapisu, adres i wielkość obszaru sekcji, ilość i wielkość zapisanych rekordów oraz gospodarkę wolnymi obszarami pamięci dyskowych - prowadzona jest ich ewidencja i istnieje możliwość

ich scalenia. Ponadto sekcje robocze są automatycznie likwidowane po zakończeniu działania zadania.

Przejście z SOM-3 na system DSM-1 polega jedynie na przekopiowaniu informacji zapisanych na pakietach dyskowych / wymiennych/ na zbiory DSM-1 pod kontrolą DSM-1.

Wielodostępny DSM-1 wymaga conajmniej:

- pamięci operacyjnej 32 K słów,
- jednej jednostki pamięci dyskowej,
- jednego urządzenia we/wy /terminal/,
- czytnika taśmy perforowanej.

Natomiast rozszerzony zestaw urządzeń w MERA-400, może zawierać:

- pamięć operacyjną do 512 k słów,
- większej ilości pamięci dyskowych,
- dowolną ilością urządzeń DZM-180, DZM-180 KSR lub podobnych np. monitory ekranowe MERA 7952/7953.

Handlery innych urządzeń nie zostały opracowane w DSM-1 tylko z powodu braku dostępu do tych urządzeń. Jednak w razie potrzeby jednostka autorska może podjąć się uzupełnienia systemu o dodatkowe handlery.

Pod systemem DSM-1 można wykorzystywać procesory SOM-3, jednak ze względu na wady i małe możliwości tych produktów, opracowano szereg nowych,pozwalających w pełni wykorzystać możliwości DSM-1 i MERY-400 są to:

- JOB CONTROL - procesor podstawowy,
- CHO DSM - operacje na pakietach dyskowych,
- AKT DSM - aktualizator tekstów,
- MAC DSM - translator rozszerzonego języka MACROASSEMBLER,
- LIB DSM - bibliotekarz sekwencyjny.

- CAT DSM - bibliotekarz słownikowy,
- EDI DSM- konsolidator,
- FSY DSM - obsługa podsystemu zbiorów dyskowych

i inne.

Praktycznie pod DSM-1 przetestowano /niestety nie w sposób pełny ze względu na brak danych testowych/ systemy PRO BUS i OWIG, które pracują poprawnie.

Dotychczas omawiane systemy operacyjne przejęły filozofię i architekturę logiczną znanego SOM-3, zwłaszcza te, dotyczące podziału pakietów dyskowych oraz ekstradody.
W związku z tym, pod tymi przedstawionymi skrótowo systemami pracują dobrze procesory SOM-3 w tym kompilatory jak również oprogramowanie aplikacyjne.

4. Na koniec tej krótkiej prezentacji najbardziej znanych nowych systemów operacyjnych dla MERY-400, chciałbym przedstawić systemy CROOK-3 i jego nowa wersję CROOK-4 autorstwa Instytutu Okrętowego Politechniki Gdańskiej. Systemy te były szczegółowo prezentowane przez autorów opracowania na ubiegłorocznej Konferencji Użytkowników MERY-400 ,która odbyła się w listopadzie 1983 r.

Koncepcyjnie rodzina CROOK-3 i CROOK-4 różni się w zasadniczy sposób od standardu SOM-3.

Różnice te głównie dotyczą sposobu organizacji wielodostępu- brak jest wyróżnionej konsoli operatora - wszyscy użytkownicy posiadają wrażenie wyłączności pracy na MERA-400, inna jest organizacja zbiorów dyskowych i odmienna jest metoda zarządzania pamięcią operacyjną, istnieje też możliwość pracy wieloprogramowej na każdym terminalu szczególnie uwydatniona w CROOK-4.

Wszystko to wpłynęło na inne zaimplementowane ekstrakody w CROOK-3 i CROOK-4, które nie mają żadnego odzwierciedlenia przy porównaniu z SOM-3.

W sensie stricto systemu operacyjnego, uważa się, że CROOK-3 a zwłaszcza CROOK-4 jest rozwiązany najlepiej w stosunku do prezentowanych tutaj systemów. System CROOK-4 zwłaszcza, jest bardzo podobnie zorganizowany koncepcyjnie i architektonicznie do znanego i szeroko stosowanego na wielu maszynach cyfrowych w świecie, systemu UNIX.

Pewna wada dotycząca braku możliwości przenoszenia binariów SOM-3 na systemy CROOK-3 i CROOK-4, została w znacznej części pomniejszona przez opracowanie symulatora SOM-3, pod którym pracuje cała fabryczna biblioteka oprogramowania podstawowego. Pewne perturbacje związane z eksploatacją binariów programów aplikacyjnych wygenerowanych pod SOM-3 związane są z niedoskonałością symulatora, który jednak jest doskonalony przez autora tego opracowania.

Jednakże istnienie symulatora SOM-3 uważa się z pozycji widzenia użytkownika za rozwiązanie tymczasowe ułatwiające tylko przejście z SOM-3 na system CROOK-3 lub CROOK-4.
Trudno sobie wyobrazić potrzebę opracowywania nowych produktów oprogramowania pod symulatorem SOM-3 na systemie CROOK-3 lub CROOK-4 - chociaż takie rozwiązanie jest w pełni możliwe.

Z oprogramowania narzędziowego autorzy rodziny CROOK-3 i CROOK-4 oferuję następujące produkty:

- kompilator języka C,
- kompilator języka FORTRAN

- kompilator języka LISP,
- kompilator języka CSL,
- interpreter rozszerzonego języka BASIC,
- assembler ASSM
- edytor tekstów EDI, i inne które są jeszcze w opracowaniu

Przedstawiając powyższą krótką prezentację nowych znanych i szeroko eksploatowanych systemów operacyjnych dla MERY-400, chciałbym podkreślić następujący problem:

Systemy te zostały opracowane w różnych Ośrodkach w Polsce całkowicie niezależnie od siebie. Powodem tej powstałej sytuacji były próby lepszego wykorzystania MERY-400 / i jej sprzętu niż to miało miejsce w SOM-3/ podejmowane przez różne ośrodki eksploatujące ten powszechny mały komputer. Próby unowocześnienia oprogramowania systemowego zostały osiągnięte we wszystkich przedstawionych tutaj opracowaniach.

Jednakże, nawet w grupie nowych systemów operacyjnych SOM-3 podobnych, wiele problemów związanych z zaprojektowaniem zostało rozwiązanych odmiennie przez te ośrodki autorskie. Dotyczy to głównie rozszerzonej listy ekstrakodów w nowych opracowaniach. Ponieważ ośrodki te opracowały i opracowują obecnie nowe oprogramowanie narzędziowe /kompilatory nowych języków / powoduje to brak możliwości a nawet całkowicie uniemożliwia przenoszenia tych produktów na inne implementacje systemów operacyjnych - oczywiście na ten sam komputer MERA-400.

Dodatkowo ,ponieważ jednostki te, opracowują kompilatory nowych szeroko często stosowanych Języków /PASCAL, LOGLAN,C, MODULA2 i inne/, może dojść w niedalekiej przyszłości do paradoksalnej sytuacji, takiej w której może nastąpić brak możliwości wymiany oprogra-

mowania aplikacyjnego nawet na poziomie programów źródłowych wśród użytkowników tej samej MERY-400.

W związku z powyższym, narodziła się inicjatywa zorganizowania spotkania roboczego na temat oprogramowania systemowego i podstawowego. Naradę tę zorganizowała Tymczasowa Rada Programowa Porozumienia Użytkowników Minikomputera MERA-400 i odbyła się ona w dniach 25-28 września 1984 r w Gdańsku.

Na to robocze spotkanie zaproszeni zostali przestawiciele ośrodków omawianych systemów operacyjnych:

- Instytut Informatyki UW,
- Instytut Informatyki UJ,
- Instytut Ciężkiej Syntezy Organicznej - Kędzierzyn - - Koźle
- Instytut Okręgowy PG,
- Instytut Techniki Konstrukcji Budowlanych PP.

Należy podkreślić, że było to pierwsze zorganizowane spotkanie, na które przybyli projektanci nowych znanych szeroko eksploatowanych systemów operacyjnych. Spotkanie to odbyło się pod hasłem wymiany doświadczeń i informacji na temat oprogramowania podstawowego MERY-400, jak również dokonania próby skoordynowania nowo podejmowanych prac.

Na naradzie tej jednogłośnie stwierdzono, że bezsensowne byłyby próby podejmowania wspólnych prac nad nowym systemem operacyjnym dla MERY-400, który zadowoliłby wszystkich użytkowników, chociażby w świetle pracochłonności i czasochłonności tych prac oraz w świetle zaprzestania produkcji tych maszyn.

Zarysowała się możliwość szerokiej współpracy i wymiany kompilatorów języków oprogramowania pomiędzy ośrodkami:

- PASCAL, LOGLAN- Instytut Informatyki UW,
- C - Instytut Okrętowy PG,
- MACROASSEMBLER SOM-3 - Instytut Ciężkiego Syst.Org- - Kędzierzyn
- FORTRAN - Inst.Technologii Konstrukcji Bud. PP.

Zrezygnowano z podjęcia próby poszerzenia warstwy symulatora SOM-3 w systemie CROOK-4 i przeniesienia pod ten symulator wszystkich nowych produktów oprogramowania podst. /aplikacyjnego/ systemów SOM-3 podobnych. Rozwiązanie takie byłoby nieefektywne w eksploatacji pomimo atrakcyjności dla pewnej grupy użytkowników MERY-400.

Ostatecznie ustalono, że w pierwszym etapie współpracy podejmie się następujące prace:

1. Przeniesieniem kompilatora języka PASCAL autorstwa Instytutu Informatyki UW bezpośrednio pod system CROOK-4.
2. Przeniesieniem kompilatora języka C autorstwa Instytutu Okrętowego PG na systemy SOM-3 podobne.

Zarysowała się potrzeba podjęcia dalszych tego typu prac i w następnej kolejności mogłyby być przenoszone następujące produkty:

- kompilator języka LOGLAN,
- kompilator języka FORTRAN / CROOK-4,

i inne w zależności od potrzeb członków Porozumienia Użytkowników Minikomputera MERA-400.

Wszystkie te prace łącznie z wymienionymi pracami tzw. etapu I współpracy byłyby finansowane przez Porozumienie Użytkowników Minikomputera MERA-400.

inż.Bogdan Machowiak jest autorem opracowania:

"Analiza Systemów Operacyjnych dla MERY-400", która zawiera analizę następujących systemów:

1. SOM-3 standard
2. Monitor MAS - opracowany przez FMiK-ERA w Warszawie.
3. MAX - opracowany przez FMIK - ERA w Warszawie.
4. Uzg. System Op SOM-3 - oprac.przez Inst.Informatyki UW
5. Wielodostępny SOM-3.P-oprac.przez Inst.Informatyki UJ.
6. Wielodostępny DSM-1 oprac. na ICSO -Kędzierzyn - Koźle.
7. CROOK-3 - oprac. przez Inst.Okrętowy PG.
8. CROOK-4 - oprac. przez Inst.Okrętowy PG.

Praca ta została wykonana na zlecenie BISTYP w Warszawie w ramach prac Klubu Użytkowników MERY-400.

dr Paweł Gburzyński
Instytut Informatyki Uniwersytetu Warszawskiego
00-901 PKiN, p. 850

POKONFERENCYJNE REFLEKSJE

W trakcie tegorocznej konferencji miało miejsce wiele wystąpień, których myślą przewodnią była konieczność wychodzenia z "zaścianka", jaki stanowi nasza informatyka w porównaniu z tym, co dzieje się na Zachodzie. Reprezentując placówkę naukową, uprawiającą niejako "z definicji" działalność stricte informatyczną poczułem się zobligowany do wypowiedzenia się w tej sprawie na piśmie. Duża liczba ulotek zawierających treść mojego konferencyjnego referatu (nie mającego zresztą wiele wspólnego z wychodzeniem z zaścianka) wystarczyła zapewne dla wszystkich uczestników konferencji. Sądzę zatem, iż zamiast powielać tu jej treść wolno mi będzie spróbować zrobić lepszy użytek z przynależnego mi miejsca.

Podstawowym hasłem, wokół którego koncentrowały się głosy wołające o nawrócenie się użytkowników na drogę postępu był UNIX (znak tow. zastrz. przez Bell Labs Inc.). Wszyscy zapewne już wiedzą, iż jest to system operacyjny, którego koncepcja zrodziła się na początku lat siedemdziesiątych, i począwszy od ich połowy wchodziła dość burzliwie w życie poprzez implementacje na praktycznie wszystkich liczących się typach komputerów. Przyznam się od razu: mój głos będzie głosem krytycznym; może nie tyle w

stosunku do samego UNIX'a, który jest niewątpliwie doskonałym systemem operacyjnym (zwłaszcza dla minikomputerów), ile w stosunku do wasi, jaka w wielu kresach nadaje sie temu systemowi w kontekście kryzysu naszej informatyki.

Po pierwsze chciałbym zauważyć, iż nie jest prawda, że UNIX posiada same zalety. System ten jest przeznaczony dla doświadczonego i inteligentnego użytkownika, który potrafi rozumieć to co sie dzieje w maszynie i czuć jej zasoby. Nieograniczona dynamika UNIX'a stwarza cały szereg zagrożeń związanych z możliwością zapchania systemu. Obsługa wielu sytuacji wyjątkowych jest niewystarczająca dla bezpiecznej wielodostępnej pracy dużych systemów użytkowych. System ten nie nadaje sie zupełnie dla zastosowań typu real-time (przetwarzanie w czasie rzeczywistym). Wad takich mógłbym przytoczyć wiecej mógłbym powołać sie na głosy wielu doświadczonych użytkowników, którzy doceniając zalety UNIX'a wyrażają sie krytycznie o wielu jego elementach. Nie jest jednak moim celem doprowadzenie do konkluzji, że UNIX jest zły i że być może jakiś inny system stanowiłby tutaj lepsze hasło. Chciałbym jedynie tę część użytkowników, która uważa, iż poza UNIX'em świat nie istnieje nakłonić do bardziej trzeźwego spojrzenia. Ogromne powodzenie UNIX'a nie jest wyłączną zasługą niespotykanej nigdzie poza nim jakości oferowanych przezeń środków. Wydaje mi sie, iż przede wszystkim jest ono rezultatem niezwykle silnej mobilności tego systemu, którego implementacja na praktycznie dowolnej rozsądnej maszynie nie trwa nigdy dłużej niż kilka miesięcy. Rzecz jasna chodzi mi o implementacje wykonywaną w sposób profesjonalny, przy dostępności szeregu specjalnych narzędzi programowych.

Często spotykanym sformułowaniem w stosunku do UNIX'a jest stwierdzenie, iż wprowadza on pewien "nowy standard". Mówi się też często, że jakiś inny system jest pod jakimś względem (lub w całości !?) kompatybilny z UNIX'em. Ze sformułowań takich wynikać może sugestia jakoby użytkownik, który zaakceptuje UNIX uzyskiwał coś więcej niż tylko możliwość wygodnej pracy pod systemem operacyjnym z prawdziwego zdarzenia. Tymczasem słowo "standard" oznaczać tu może jedynie tyle ile mniej-więcej w określeniu "standard życiowy". Czyż bowiem każdy program uruchomiony w Fortranie pod systemem UNIX, dajmy na to na maszynie IBM-370, będzie natychmiast przenaszalny na maszynę PDP-11 działającą pod tymże systemem? A może jest tak z programami napisanymi w C? Może chodzi tu o wymienność kaset dyskowych, czy może binarnych taśmek papierowych? Jest oczywiste, że na każde z powyższych pytań odpowiedź brzmi "nie". Cóż więc może tu być standardem w ścisłym rozumieniu tego słowa? Jedynie odpowiednio wysoki komfort pracy użytkownika, osiągalny zresztą obecnie pod wieloma systemami operacyjnymi, które z UNIX'em mają niewiele wspólnego. Wszelkie zaś problemy związane np. z przenoszeniem oprogramowania pomiędzy maszynami, lub nawet jedynie spod innego systemu operacyjnego, pozostają na poziomie tychże problemów znanych nam dobrze z dotychczasowych zmagań. W szczególności nie słyszałem, aby którykolwiek kompilator Fortranu działający pod systemem UNIX dopuszczał CAN-kod.

Można bez trudu wyliczyć grupy użytkowników, którym wprowadzenie UNIX'a na MERE-400 przyniosłoby rozczarowanie. Są to np. orędownicy podziału dysków na sekcje, fanatycy CAN-kodu, miłośnicy asemblera MAC, konwerterzy taśmek papierowych,

zwolennicy "firmowego" Job-control'a jako standardu dla tego typu programów, itp. Z drugiej strony są to także ci użytkownicy, którzy posiadają gotowe zestawy programów obsługujące często niestandardowe urządzenia, czy też (a są i tacy), którzy pracują na tzw. gołej maszynie.

Wydaje się dość oczywiste (i taka zresztą była konkluzja spotkania w Gdańsku), że mowa o UNIX'ie może być jedynie w kontekście zupełnie nowej maszyny. W końcu CROOK-4 po wyposażeniu go w kompilator języka C z prawdziwego zdarzenia przestanie w istotny sposób odbiegać od interesującego nas "standardu". System OOPS, przechodzący obecnie w IIUW fazę testowania, zawiera w sobie wszystkie istotne elementy UNIX'a dopuszczając jednocześnie kilka niespotykanych pod nim finezji (np. pracę dwuprocesorową).

Jeżeli już mowa o być może niedalekiej przyszłości, kiedy to pojawi się na horyzoncie jakiś nowy sprzęt wymagający oprogramowania, to przyznać muszę, iż odczuwam głęboki niepokój dostrzegając wokół siebie objawy UNIX'owej nagonki. Ileż tu pokrzykiwania o standardach światowych i o kompatybilności. Mało kto natomiast zauważa, że poruszając się w ten sposób nie zakreślamy nawet tzw. krzywej goniącego psa lecz jakąś łamaną o kątach prostych. Przecież nie utracimy żadnych standardów i nie zaprzepaścimy żadnej kompatybilności gdy pokusimy się o zmontowanie systemu trochę innego (nie śmiem powiedzieć lepszego) niż UNIX. Wystarczy jeżeli pod tym systemem będą dostępne kompilatory języków programowania uznawanych powszechnie za standardowe. Nie rozumiem dlaczego każda praca, która nie jest powieleniem czegoś, co zrobiono w USA piętnaście

lat temu ma być u nas a priori uznana za bezcelowa.

Niektóre głosy zabrane na konferencji dotyczyły kłopotów z przenoszeniem programów z systemu SOM-3 do systemu CROOK-4, bądź też kłopotów z adaptacja programów uruchomionych na innych maszynach. Chodziło tu głównie o programy napisane w Fortranie. Problemy tego typu sa powszechnie spotykane na całym świecie, a fakt, iż rysuja sie one nieco ostrzej w przypadku MERY-400 nie posiada wiekszego znaczenia. Nie jest bowiem prawda, iż u nas istnieje jakaś szczególnie niska kultura informatyczna. Być może argument ten nie jest specjalnie pocieszajacy, ale to przecież nie u nas wymyślono Fortran, Basic, . Cobol, i nie my zapoczatkowaliśmy ekspansje tych języków prowadzaca do znanego wszystkim obecnego stanu. Fakt, że ta ogólnie niska kultura informatyczna powoduje u nas dodatkowe szkody bierze sie stad, że tradycyjnie nie próbujemy wyciagnać żadnych wniosków z błędów popełnionych przez prekursorów. Od dobrych dwudziestu lat Fortran podlega krytyce, która ostatnio, ze wzgledu na istotne postępy w metodologii programowania, staje sie szczególnie zajadła. O Basic'u nie da sie zreszta powiedzieć nic wiecej. Tymczasem u nas sa to na ogół pierwsze, jeśli nie jedyne, języki, z którymi styka sie student wyższej uczelni technicznej czy uniwersytetu. Jeżeli już koniecznie chcemy domagać sie światowych standardów to chyba właśnie tu, tym bardziej że ich osiagniecie znajduje sie w naszym bezpośrednim zasiegu. Programista, który poznał Logian, czy choćby Pascal, niezależnie od ogólnego wzbogacenia swojej kultury infromatycznej, posiadł pewne określone umiejetności praktyczne pozwalajace mu bez wysiłku opanować np. Fortran, a nawet lepiej zrozumieć o co

naprawdę chodzi w tym języku. Z drugiej strony student, którego na samym początku nauczono Basic'u czy Fortranu ma poważne szanse na to, iż cała reszta jego programistycznej kariery przebiegać będzie wśród rozpaczliwych zmagań ze zmieniającym się kompilatorem, sprzętem czy systemem. W sytuacji, gdy stoimy przed zadaniem zaadaptowania gotowego systemu użytkowego napisanego np. w Fortranie nie mamy na ogół żadnego wyboru. Mamy natomiast ten wybór przystępując do pisania czegoś od nowa. Zadajmy sobie wówczas pytanie: czy to na pewno musi być Fortran, Basic lub Asembler?

UNIX powstał w kręgu ludzi o określonej kulturze, którzy wcześniej niż inni zdołali pozostawić za sobą podstawowe problemy natury metodologicznej. To właśnie dlatego, że system ten napisany został w strukturalnym języku wysokiego poziomu mógł on powstać szybko, uzyskać dużą popularność, pozostać otwartym na modyfikacje wykonywalne nawet przez użytkowników, stać się pewnym "standardem". Ale prawdziwym standardem stać się powinien ten sposób myślenia, który doprowadził do powstania UNIX'a, co, jak sądzę, winno być naszym celem nadrzędnym.

II JĘZYKI PROGRAMOWANIA

mgr Piotr Findeisen
Instytut Informatyki Uniwersytetu Warszawskiego
Pałac Kultury i Nauki, pok. 850
00-901 Warszawa

CHARAKTERYSTYKA JEZYKA PROGRAMOWANIA PASCAL

Jezyk programowania Pascal został zaprojektowany przez Prof. N. Wirtha na przełomie lat sześćdziesiatych i siedemdziesiatych. Jezyk ten miał spełniać cały szereg kryteriów, które wydawały sie ważne w tym czasie (i które do dziś nie straciły znaczenia). Był to okres w którym coraz wieksza wage przykładano do metodologii programowania zalecając hierarchiczne projektowanie, programowanie strukturalne itd. Duże znaczenie przywiązywano też do czytelności programów i łatwości ich modyfikacji. Zdawano sobie również sprawe z wad ówczesnego jezyka programowania lat sześćdziesiatych Algolu-60, a zwłaszcza z jego niewystarczających mechanizmów budowy struktur danych.

Powszechnie uważa sie, że Pascal spełnił pokładane w nim nadzieje. Jezyk ten umożliwia pisanie programów eleganckich, i to nie tylko w przypadku małych zadań. Znane sa próby pisania dużych systemów w Pascalu. Tradycyjna już metoda implementacji Pascala jest pisanie kompilatora tego jezyka w nim samym. Po wprowadzeniu do Pascala pewnych niezbędnych mechanizmów (np.

współbieżność) pisano w nim nawet systemy operacyjne.

W stosunku do innych języków tego typu Pascal umożliwia dość eleganckie konstruowanie odpowiednich struktur danych. Listy, kolejki, grafy czy drzewa są definiowane w prosty, czytelny sposób.

Wirth pragnął, aby jego język był szczególnie przydatny dla dydaktyki, ucząc dobrego stylu programowania i "myślenia w Pascalu". Faktycznie i ten cel został osiągnięty. Pascal jest językiem łatwym do opanowania, wszystkie jego konstrukcje są naturalne i nie wprowadzają do programów "nienormalnych" zjawisk, jak np. wołanie przez nazwę w Algolu-60. Pascal został bardzo szybko zaakceptowany przez wyższe uczelnie, które widziały w nim właściwe narzędzie do nauki programowania. Uważano, że Pascal powinien być pierwszym językiem programowania z jakim styka się programista.

Przejdźmy zatem do krótkiego przedstawienia samego języka. W Pascalu wyróżniamy dwa zasadnicze rodzaje instrukcji: proste i strukturalne. Instrukcje proste to: instrukcja przypisania, instrukcja procedury i instrukcja skoku, występujące niemalże w każdym języku programowania. Instrukcje strukturalne powstają przez pewnego rodzaju składanie instrukcji (zarówno prostych jak i strukturalnych). Typowym przykładem instrukcji strukturalnej jest instrukcja złożona, będąca ciągiem dowolnych instrukcji oddzielonych od siebie znakiem ';' i ujętym w nawiasy BEGIN i END. Pascal oferuje też trzy rodzaje instrukcji iteracyjnych, służących do organizowania pętli, instrukcję warunkową i instrukcję wyboru.

Najsilniejszy aparat Pascala jest związany z definiowaniem

typów danych. Podobnie jak w przypadku instrukcji, typy dzielimy na skalarne i strukturalne. Typy skalarne to typy standardowe jak np. INTEGER, CHAR a także typy wyliczeniowe definiowane przez użytkownika poprzez podanie wszystkich wartości danego typu. Typy strukturalne buduje się z innych typów (zarówno skalarnych jak i strukturalnych). Przykładem typu strukturalnego jest tablica, która komponuje się używając typu (typów) indeksu i typu elementu tablicy. Innym przykładem są rekordy, stanowiące ciąg poszczególnych pól (zmiennych), którym przypisane są nazwy i typ wartości, jakie można w danym polu umieścić. Na szczególną uwagę zasługują typy wskaźnikowe, które w połączeniu z rekordami pozwalają na budowanie wyrafinowanych struktur danych. Zmienna takiego typu może wskazywać na inną zmienną, utworzoną dynamicznie w trakcie działania programu. Zmienne dynamiczne można też usuwać, co pozwala na racjonalne modyfikacje uzyskanej w trakcie obliczeń konkretnej struktury danych.

Wszystkich zainteresowanych językiem Pascal odsyłam do książki z serii Biblioteka Inżynierii Oprogramowania pt. "Pascal", autorstwa M.Iglewskiego, J.Madey'a i St.Matwina wydanej przez WNT w 1979 roku.

W latach siedemdziesiątych powstały kompilatory Pascala dla większości stosowanych wówczas komputerów, jednak język ten poza uczelniami i nielicznymi instytutami nie był szeroko stosowany. Dotyczy to zwłaszcza rynku amerykańskiego. Jedna z przyczyn tego stanu rzeczy była ogólna niechęć Amerykanów do wszystkiego co przychodzi z Europy. Bez poparcia największych koncernów informatycznych Pascal ciągle był w Stanach Zjednoczonych pewnego rodzaju ciekawostką, znaną jedynie nielicznym kręgom

hobbystów. Wydawało się wówczas, że Pascal podzieli los innych "europejskich" języków takich jak Algol-60, Simula-67 czy Algol-68, które mimo pewnych sukcesów i zainteresowania nimi ginęły pod wpływem amerykańskich systemów, w których dominującą rolę odgrywały Fortran, Cobol lub Basic.

Przełom nastąpił na początku lat osiemdziesiątych, kiedy to gwałtowny rozwój mikroelektroniki i komputerów osobistych spowodował konieczność zaoferowania niewyszkolonemu klientowi silnego języka programowania, łatwego do nauczenia i odpowiadającego współczesnym wymaganiom światowym. Spośród wielu rynkowych propozycji amerykański klient zaakceptował Pascal i dziś jest to jeden z trzech najpopularniejszych języków dla najmniejszych systemów komputerowych (pozostałe to Basic i C).

Popularność Pascala stale wzrasta, zwłaszcza wśród użytkowników minikomputerów. Decydują o tym nie tylko zalety samego języka, ale też i pewne cechy jego kompilatorów. Napisanie efektywnego kompilatora Pascala jest zadaniem stosunkowo łatwym. Wśród licznych implementacji tego języka znanych w Polsce wiele pochodzi ze wspólnego źródła, jakim jest kompilator U.Ammanna z ETH w Zurychu. Kompilator ten został napisany w Pascalu i generował kod w abstrakcyjnym języku zwanym P-kod. Do kompilatora był dołączany interpreter P-kodu, dzięki czemu cały translator był łatwo przenaszalny na praktycznie dowolną maszynę.

Kompilator Pascala dla Mery-400 opracowany w Instytucie Informatyki Uniwersytetu Warszawskiego również pochodzi z tego źródła. Program U.Ammanna, po wprowadzeniu licznych modyfikacji stanowi pierwszy przebieg kompilatora i generuje kod pośredni

(rozszerzony P-kod). Drugi przebieg kompilatora, również napisany w Pascalu, tłumaczy kod pośredni na język asemblera.

Kompilator działa pod zmodyfikowanym w Instytucie Informatyki UW systemem operacyjnym SOM-3 i wymaga od 32k do 40k pamięci. Program zawierający około 1000 linii tekstu tłumaczy się (do postaci gotowej do wykonania) 6 minut. Język akceptowany przez kompilator prawie nie odbiega od standardu opisanego w raporcie tego języka, co powinno ułatwiać przenoszenie programów z innych maszyn na Mera-400. Dla licznych zastosowań nie bez znaczenia pozostaje też możliwość dołączania do programu pascalowego podprogramów w asemblerze.

doc. dr hab. Antoni Kreczmar

Instytut Informatyki Uniwersytetu Warszawskiego

Palac Kultury i Nauki, pok. 850

00-901 Warszawa

JEZYK PROGRAMOWANIA

LOGLAN 82

Jezyk Loglan-82 jest w pelni nowoczesnym, uniwersalnym jezykiem programowania. Skladnia jego wzorowana jest na skladni jezyka Pascal i nie odbiega od typowych przyzwyczajen programistow, natomiast semantyka tego jezyka jest bardzo bogata.

Wyliczmy podstawowe konstrukcje i cechy charakterystyczne jezyka Loglan-82.

1) Wygodny zestaw instrukcji strukturalnych,

2) Modularna struktura (z mozliwoscia rozszerzania i zagniezdzania modulow),

3) Procedury i funkcje (z pelna rekurencja), a takze procedury i funkcje jako parametry formalne,

4) Tablice dynamiczne (rozmiary tablic wyznaczane sa w trakcie wykonywania programu, dzieki czemu tablice wielowskaznikowe moga byc roznych ksztaltow, trojkatne , pasmowe itp.),

5) Klasy (uogolnienie pojecia rekordu), pozwalajace definiowac nowe typy i struktury danych,

6) Wspolprogramy,

7) Prefiksowanie - mechanizm pochodzacy od Simuli-67, znacznie w Loglanie-82 uogolniony, ktory pozwala budowac hierarchie typow i struktur danych, jezyki zorientowane problemowo itp.,

8) Typy formalne pozwalajace na parametryzacje modulow (procedur, funkcji, klas, wspolprogramow i procesow) ze wzgledu na dowolny, zlozony typ danych,

9) Mechanizmy ochrony, ktore zabezpieczaja moduly przed niewlasciwym ich uzyciem,

10) Programowane odsmiecanie - narzedzie pozwalajace w sposob skuteczny, a jednoczesnie w pelni bezpieczny, realizowac optymalna strategie zarzadzania pamiecia przydzielona uzytkownikowi w trakcie wykonywania programu,

11) Obsluga sygnalow, tzn. obsluga przerwan systemowych takich jak np. przekroczenie zakresu liczb zmiennopozycyjnych lub przekroczenie zakresu indeksu tablicy, jak rowniez przerwan swiadomie spowodowanych przez uzytkownika,

Ponizej prezentujemy kilka przykladowych programow i procedur napisanych w Loglanie. Na poczatek przyjrzyjmy sie prostemu programowi kopiujacemu we sciowy tekst zamieniajac przy tym male litery na duze:

```
program convert;
var c : char;
begin
    while not eof do
        read(c);
        if ord(c) >= ord('a') andif ord(c) <= ord('z') then
            c := chr (ord(c)-32)
        fi;
        write(c)
    od
end
```

Procedury i funkcje stanowia jedno z podstawowych narzedzi programowania w Loglanie. Ich skladnia jest nieco zblizona do pascalowej, np:

```
unit Euclid: function(i,j:integer):integer;
var k:integer;
begin
    do
        if j=0 then exit fi;
        k:=i mod j; i:=j; j:=k
    od;
    result:=i
end;
```

Ponizej przedstawiamy procedure korzystajaca z mechanizmow obslugi wyjatkow. Mechanizmy te umozliwiaja eleganckie (strukturalne) programowanie obslugi sytuacji marginalnych, zwiazanych np. z bledami w danych wejsciowych.

```
unit squareeq: procedure(a,b,c:real;output xr,xi,yr,yi:real);
(* Procedura rozwiazuje rownanie:        a*x**2 + b*x + c = 0, *)
(* xr,xi - odp. czesc rzeczywista i urojona I-go pierwiastka *)
(* yr,yi - to samo dla drugiego pierwiastka.                 *)
```

```
var delta: real;
handlers
(* Handler deklaruje obsluge  sytuacji wyjatkowych.   W przy- *)
(* padku naszej  procedury  sytuacja taka  ma miejsce gdy a=0 *)
(* (wyjatek  'division_by_zero' zgloszony przez system opera- *)
(* cyjny przy probie dzielenia ponizej).  Jezeli ponadto b=0, *)
(* co oznacza, iz rownanie nie ma pierwiastkow, wowczas zgla- *)
(* szany jest kolejny wyjatek - 'no_roots' przechwytywany np. *)
(* przez procedure (program) korzystajaca ze squareeq.        *)
when division_by_zero :
    if b=/=0 then
        xr:=-c/b;  terminate
    else
        raise No_roots
    fi;
begin
    a:=2*a; c:=2*c; delta:=b*b-a*c;
    if delta <= 0 then
        xr,yr:=-b/a;
        if delta=0 then  return fi;
        (* Standardowa wartoscia zmiennej,  na ktora nie zos- *)
        (* tala podstawiona zadna wartosc jest 0. Stad xi, yi *)
        (* sa poprawnie okreslone  po  powrocie z procedury w *)
        (* tym miejscu.                                       *)
        delta:=sqrt(-delta);
        xi:=delta/a; yi:=-xi;
        return
    fi;
    delta:=sqrt(delta);
    if b=0 then
        xr:=delta/a; yr:=-xr;
        return
    fi;
    if b>0 then b:=b+delta else b:=b-delta fi;
    xr:=-b/a; yr:=-c/b;
end squareeq;
```

Ponizej prezentujemy przyklad klasy definiujacej prosta strukture danych (stos) oraz standardowe operacje na tej strukturze:

```
unit push_down :class(n:integer);
var top:integer, A:arrayof integer;

unit pop: function :integer;
begin
    if top>0 then
        result:=A(top); top:=top-1
    else
        raise stack_underflow
    fi
end pop;

unit push:procedure(x:integer);
begin
```

```
    if n > top then
        top:=top+1; A(top):=x
    else
        raise stack_overflow
    fi
end push;

begin
    array A dim(1:n);
end push_down;
```

Przyklad uzycia powyzszej klasy moze wygladac nastepujaco:

```
var s,t,z: push_down;
 ...
    s:=new push_down(100);
    t:=new push_down(1000);
    z:=new push_down(5);
     ...
    call s.push(7);
    call t.push(100);
    i:=z.pop;
     ...
    kill(s.A);
    kill(s);
    (* Ostatnie  dwie instrukcje dokonuja programowej deallo- *)
    (* kacji  obiektu s.  W  pierwszym kroku  deallokowane sa *)
    (* wszystkie dynamiczne atrybuty obiektu (w tym przypadku *)
    (* tablica A, a nastepnie ten obiekt.                     *)
    etc.
```

Na koniec przyklad procedury wykorzystania typow formalnych. Ponizsza procedura sortuje obiekty dowolnego typu zlozonego, przy czym porzadek zadany jest funkcja 'less' stanowiaca jej parametr:

```
unit Gsort : procedure (type T; A : arrayof T;
                function less(x,y : T) : boolean);

var n,i,j:integer;
var x:T;
begin
    n:=upper(A);
    (* 'lower' i 'upper'  sa standardowymi atrybutami tablicy *)
    (* reprezentujacymi odpowiednio dolna i gorna wartosc jej *)
    (* indeksu.                                               *)
    for i:=2 to n do
        x:=A(i); j:=i-1;
        do
            if less(A(j),x) then exit fi;
            A(j+1):=A(j); j:=j-1;
            if j=0 then exit fi;
        od;
```

```
        A(j+1):=x;
    od
end Gsort;
```

Dzięki szerokiemu repertuarowi środków pozwalających między innymi na elastyczne organizowanie pętli oraz obsługę sytuacji szczególnych (wyjątków) można było całkowicie wyeliminować z Loglanu instrukcje skoku. Faktycznie, w Loglanie nie ma takiej instrukcji a jej brak szybko przestaje być zauważalny nawet przez początkującego programiste.

Prof.dr hab. Andrzej Salwicki
Instytut Informatyki UW
PKiN P-1210, 00-901 Warszawa

LOGLAN - narzędzie produkcji oprogramowania

Abstract

The arguments are given that LOGLAN offers new possibilities in production of software: programming by extending modules enables eary implementation of problem oriented languages developed from specification of /abstract/ data types, definition of hierarchics of data structures and inheritance of protocols of cooperation.

I Rynek software'owy to ok. 85% światowego rynku informatycznego. Znaczenie odpowiednich narzędzi i metod wytwarzania systemów programistycznych staje się coraz większe. Dla przykładu wspomnijmy kilka cyfr: znany język ADA pochłonął do tej pory nakłady przekraczające 2 mld. dolarów, ale zleceniodawca z Departamentu Obrony jest zadowolony bo do tej pory płaci przeszło 8 mld rocznie za produkty softwareowe i ma nadzieję że ADA zmniejszy te koszty.

Co jest potrzebne dla sprawnej produkcji oprogramowania?

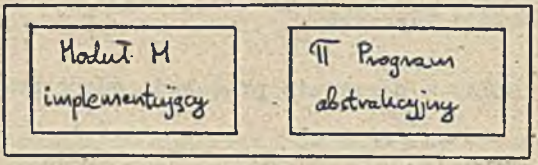
specyfikacja ← logika algorytmiczna
implementacja ← LOGLAN
weryfikacja ← logika algorytmiczna

My oferujemy dwa oryginalne narzędzia opracowane w Instytucie Informatyki UW. Z braku miejsca nie będziemy się rozwodzić na temat logiki algorytmicznej. Poprzestańmy na krótkim stwierdzeniu: nieodpowiednia specyfikacja powoduje ca 45% błędów programistycznych. Ich usuwanie jest kosztowne i mało skuteczne. Specyfikacja systemu za pomocą formuł algorytmicznych jest bardzo zwięzła, umożliwia abstrahowanie od detali implementacyjnych i ma naturalne przedłużenie w LOGLANowskiej implementacji systemu.

Nowoczesne metody programowania, by wyliczyć kilka przykładów, to

1. Abstrakcyjne struktury danych,
2. Hierarchie typów i systemów,
3. Dziedziczenie własności określonych w prototypach.

AD1. Zasada faktoryzacji - proponuje by moduł programistyczny / np. pewien program grafiki komputerowej/ potraktować jako parę



i odpowiednio podzielić pracę. Korzyści płynące z tej faktoryzacji są oczywiste. Moduł implementujący może być wielokrotnie wykorzystywany dla różnych programów. Program abstrakcyjny nie zajmuje się detalami implementacji, koncentruje się na istocie zadania, wreszcie - niebagatelna to sprawa - może być stowarzyszony z innym, lepszym modułem implementującym. Dla powodzenia tego podejścia niezbędne jest dysponowaniem należytą specyfikacją implementowanego typu danych.

AD2. Z hierarchiami typów i struktur inżynier oprogramowania spotyka się bardzo często. Wiele jest sytuacji w których mamy do czynienia z bogatym zestawem typów i takim dodatkowo, że widać w zestawie hierarchię, widać potrzebę rozbudowywania tej hierarchii. Dla przykładu wymieńmy systemy projektowania np. układów scalonych gdzie ilość typów jest ograniczona tylko inwencją projektantów, ale gdzie widać hierarchię to pewne układy są rozwinięciem innych. Podobnie rzecz się ma w wielu innych systemach np. podczas symulacji. Nie ma nic w tym dziwnego. Świat wokół nas jest zorganizowany hierarchicznie. Hierarchie są najważniejszym narzędziem do zapanowania nad chaosem informacyjnym jaki nam zagraża. Naturalne jest więc, że systemy informatyczne rozwijają hierarchiczne typy danych i wykorzystuję je.

AD3. W wielu sytuacjach mamy do czynienia z ogólnymi schematami postępowania, które trzeba tylko nieznacznie modyfikować w odniesieniu do zmieniających się typów argumentów. Nie zawsze możemy w procedurze wykorzystywać parametry formalne - typy argumentów a pomimo to chcemy korzystać z procedur tzw. generycznych. Idzie nam często o narzucenie pewnych schematów postępowania /np. protokołów/ modułom programistycznym. Inaczej mówiąc idzie o to by dane moduły na pewno odziedziczyły pewne wcześniej określone cechy i ew. rozwinęły się w ramach pozostawionej im swobody.

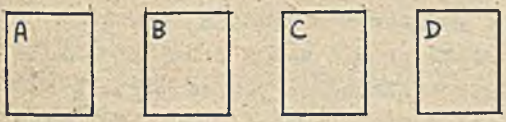
W jaki sposób urzeczywistnieją się wyliczone powyżej meto-

-dy programowania?
Oferowane systemy zawierają jako konstrukcje programotwórcze: chroniczne /encapsulated/ typy danych, pakiety /packages/, struktury dynamiczne lub wskaźnikowe /access types/, procesy współbieżne, narzędzia synchronizacji procesów i in.

LOGLAN oferuje zestaw jeszcze mocniejszy: poczynając od dynamicznych tablic /instr. A:=B jest legalna również dla tablic o różnej długości/, poprzez moduły - klasy umożliwiające tworzenie dowolnych struktur, współprogramy /coroutines/, procesy współbieżne, obsługę sygnałów /exception handling/ do specjalnej operacji składania modułów zwanej prefiksowaniem. Operacja ta ma wielorakie zastosowania. Umożliwia szybką realizację języków problemowo zorientowanych, tworzenie hierarchii typów i struktur, narzucanie procedurom pewnych protokołów i in. pożądanych własności.

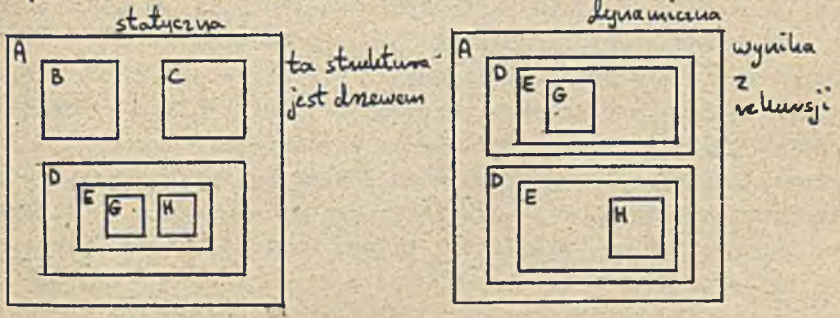
Prefiksowanie jest operacją dwuargumentową na modułach. Pierwszy argument powinien być modułem rodzaju klasa, drugi argument może być dowolnym modułem: blokiem, procedurą, funkcją, klasą, współprogramem lub procesem. Wynik operacji zachowuje rodzaj modułu prefiksowanego.

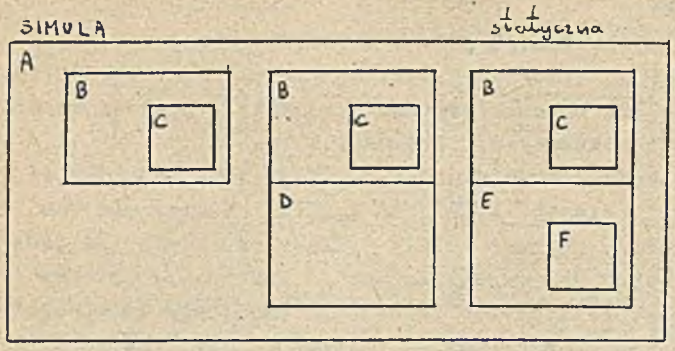
MODULARYZACJA /przykładowe struktury modułów programów w innych językach/
FORTRAN i inne assemblery:

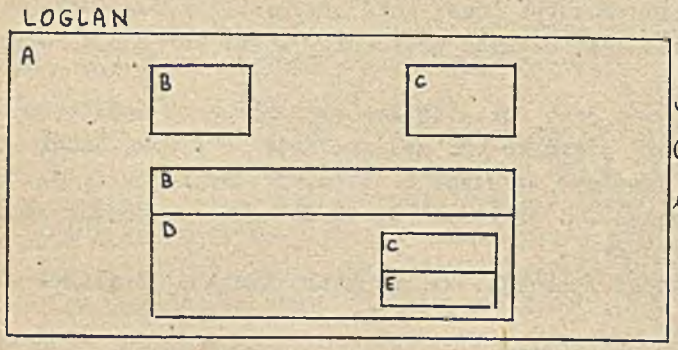


w assemblerach, FORTRANie, BASICu moduły tworzą prostą strukturę zbioru

Algol, Pascal i inne:







Rysunki modulowych struktur programów wyglądają abstrakcyjnie, ale mają znaczenie bardzo praktyczne i tak w programie LOGLAN-owskim moduł B może być klasą definiującą pewien język problemo-wo - zorientowany, C może być protokołem pracy pewnego urządze-nia. Wtedy D prefiksowane przez B jest prawdopodobnie blokiem - programem abstrakcyjnym napisanym w języku B, a pewną procedu-rą współpracy z urządzeniem zgodną z protokółem C.
LOGLAN zawiera w sobie semantycznie wszystkie języki /nie zapo-minajmy jednak o różnicach ortograficznych/ i dlatego możemy również zinterpretować odpowiednio wcześniejsze rysunki.
Rysunek struktury programu w SIMULI może być więc obrazem nas-tępującej sytuacji. W programie A mamy do czynienia z obiektami dwu typów D i E np. urzędnik i klient banku. Ich wspólne cechy np. operacja C wejście do banku są opisane w modelu B który dzięki prefiksowaniu jest dalej rozszerzany odpowiednio do struktur D lub E.

Mgr Piotr Gucwa, Zakład Oprogramowania Mini i Mikrokomputerów. Warszawa, ul. Jasna 14/16, pok. 362, tel. 26 00 08.

INTERPRETER PROLOGU DLA MERY-400

PROLOG (PROgramming in LOGic) jest praktycznym narzędziem do programowania w logice, t.z.n. program może być interpretowany jako zbiór aksjomatów, w oparciu o które dowodzi się twierdzenia (dyrektywy) pojawiające się na wejściu.

Z punktu widzenia użytkownika główną zaletą języka jest łatwość programowania. Program w Prologu jest zwarty i czytelny, może być szybko napisany i szybko uruchomiony.

Prolog uwalnia programistę od wielu czynności mechanicznych związanych z tworzeniem programu, umożliwia tym samym koncentrację na twórczych aspektach programowania.

Dla poparcia tych stwierdzeń przytoczymy cztery przykłady typowych zastosowań Prologu, zapisane w najbardziej rozpowszechnionej wersji tego języka zwanej - DECProlog /4/.

1. Przetwarzanie list.

```
concatenate ( X.L1,L2,X.L3)  :- concatenate ( L1,L2,L3) .
concatenate ( nil,L,L) .

member ( X,X.L) .
member ( X,Y.L) :- member ( X,L) .

reverse ( L,L1) :- reverse_concatenate ( L,nil,L1) .

reverse_concatenate ( X.L1,L2,L3) :-
                    reverse_concatenate ( L1,X.L2,L3) .
reverse_concatenate ( nil,L,L) .
```

Komentarz

Symbol ' :- ' czytamy ' jeśli'.

Identyfikatory zaczynające się dużą literą oznaczają zmienne. Pozostałe identyfikatory oznaczają predykaty, funktory lub stałe.

Zapis [illegible] ogólnie jest [illegible] klauzuli
t.n.n. do wyrażenia postaci:

[illegible] lub [illegible] :- [illegible]{,[illegible]}*.

Term X.L oznacza listę której pierwszym elementem jest
X a L jest resztą (ogonem).

Term nil oznacza listę pustą.

Powyższy program definiuje cztery procedury (relacje):

1 concatenate (L1,L2,L3) - lista L3 jest połączeniem list
L1 i L2.
"

2 member (X,L) - X jest elementem listy L.

3 reverse (L1,L2) - lista L2 jest odwróceniem listy L1.

4 reverse_concatenate (L1,L2,L3) - lista L3 jest połączeniem
odwróconej listy L1 i listy
L2.

2. Mała baza danych.

```
descendant (X,Y) :- offspring (X,Y) .
descendant (X,Z) :- offspring (X,Y) , descendant (Y,Z) .

offspring (abraham, ishmael) .
offspring (abraham, isaac) .
offspring (isaac, esau) .
offspring (isaac, jacob) .

?- descendant (abraham, X) .
```

Komentarz

descendant (X,Y) - czytamy 'Y jest potomkiem X'.
offspring (X,Y) - czytamy 'Y jest potomkiem X w pierwszym
pokoleniu (dzieckiem)'.

Wyrażenie postaci ?-TWIERDZENIE czytamy 'podaj wartości
wszystkich zmiennych dla których zachodzi TWIERDZENIE'.
W powyższym przykładzie na wyjściu otrzymamy:

X = ishmael
X = isaac
X = esau
X = jacob

3. Różniczkowanie symboliczne.

```
:- op ( 300,xfy,^) .

d ( U+V,X,Du+Dv)  :- !,  d ( U,X,Du) , d ( V,X,Dv) .
d ( U-V,X,Du-Dv   :- !,  d ( U,X,Du) , d ( V,X,Dv) .
d ( U*V,X,Du*V+U*Dv)  :- !, d ( U,X,Du) , d ( V,X,Dv) .
d ( U^N,X,N*U^N1*Du) :- !, integer(N), N1 is N-1, d(U,X,Du) .
d ( -U,X,-Du) :- !, d ( U,X,Du) .
d ( exp(U),X,exp(U)*Du) :- !,d(U,X,Du) .
d ( log(U),X,Du/U) :- !, d ( U,X,Du) .
d ( X,X,1) :- !.
d ( C,X,0) :- atomic(C), C \== 0, !.
```

Komentarz

Wyrażenie postaci :-PREDYKAT(TERM{,TERM}*) . oznacza wywołanie procedury o nazwie PREDYKAT.

Procedura op(300,xfy,^) definiuje operator infiksowy ^ (potęgowanie) o priorytecie 300 i łączeniu w lewo.

Operatory +, -, *, /, mają typy i priorytety zdefiniowane standardowo.

Oprócz procedury 'op' w przykładzie występują wywołania nie zdefiniowanych procedur: integer, is, atomic, \==, !. Są to procedury standardowe w DECPrologu i mają one znaczenia:

integer(N) - N jest liczbą całkowitą,

X is Y - wartość wyrażenia Y jest równa X,

atomic(C) - C jest stałą,

X \== Y - termy X i Y nie są identyczne,

! - procedura odcięcia, jest to specyficzna procedura Prologu, której wykonanie kończy się zawsze powodzeniem, a skutkiem jej wykonania jest odcięcie możliwości nawrotu do pewnych fragmentów programu. Dokładniejsze informacje znajdzie czytelnik w podręcznikach Prologu /2/, /3/.

d (U,X,V) - czytamy 'V jest pochodną U względem X'.

Aby obliczyć pochodną wyrażenia x^2+e^x względem x piszemy:

```
?- d(x^2+exp(x),x,WYNIK ) .
```

4. Translacja zdań języka angielskiego na formuły logiczne.

```
:- op(900,xfx,=>), op(800,xfy,&) , op(300,xfx,:) .

sentence(P) --> noun_phrase(X,P1,P) , verb_phrase(X,P1) .

noun_phrase(X,P1,P) --> determiner(X,P2,P1,P) , noun(X,P3) ,
          rel_clause(X,P3,P2) .
noun_phrase(X,P,P) --> name(X) .

verb_phrase(X,P)  --> trans_verb(X,Y,P1) , noun_phrase(Y,P1,P)
verb_phrase(X,P)  --> intrans_verb(X,P) .

rel_clause(X,P1,P1&P2) -->"that", verb_phrase(X,P2) .
rel_clause(X,P,P) --> "".

determiner(X,P1,P2,all(X):(P1=>P2)) -->"every".
determiner(X,P1,P1,exists(X):(P1&P2)) -->"a".

noun(X,man(X)) -->"man".
noun(X,woman(X)) -->"woman".

name(john) --> "john".

trans_verb(X,Y,loves(X,Y)) -->"loves".
intrans_verb(X,lives(X)) -->"lives".
```

Komentarz

Przykład ilustruje zastosowanie gramatyk metamorficznych. Gramatyki takie są rotszerzeniem notacji BNF dla gramatyk bezkontekstowych, o następujące konstrukcje:
- zmienne symbole terminalne i nieterminalne,
- symbole nieterminalne mogą mieć argumenty,
- reguły mogą zawierać warunki logiczne.

Reguły gramatyki metamorficznej są translowane na klauzul i wykonywane przez interpreter Prologu. Mogą być użyte zarówno do syntezy jak i analizy wyrażeń dowolnego języka.

W powyższym przykładzie aby przetłumaczyć zdanie:

Every man that lives loves a woman.

należy napisać:

```
?- sentence(P, "every man that lives loves a woman","").
```

otrzymaną odpowiedzią będzie:

```
P = all(X):(man(X)&lives(X) =>
            exists(Y):(woman(Y)&loves(X,Y)))
```

Prolog-400 jest wersją języka zbliżoną do DECPrologu i zrealizowaną w naszym zakładzie.

Interpreter Prologu-400 został napisany w asemblerze Gasc i uruchomiony pod systemem SCM-3 zmodyfikowanym przez IIUW. Interpreter wymaga od 20K do 64K pamięci, w zależności od rozmiaru programu.

Prolog-400 zawiera następujące mechanizmy:

- 16 bitowa arytmetyka dla liczb całkowitych,
- procedury bezpośredniego dostępu do pamięci dyskowej,
- edytor tekstów źródłowych,
- śledzenie wykonywania programów,
- przerywanie, modyfikowanie i wznawianie wykonywania programu w dowolnym miejscu,
- automatyczny dealokator pamięci (garbage collector),
- optymalizacja wykorzystania pamięci przy rekurencyjnym wywołaniu na końcu procedury (tail-recursion optimisation).

Literatura:

1 Bowen L. D.: DECSystem-10 Prolog User's Manual. Version 3.43. University of Edinburg, Department of Artificial Intelligence, December 1981.

2 Clocksin W. F., Mellish C. S.: Programming in Prolog. Springer-Verlag, 1982.

3 Kluźniak F., Szpakowicz S.: Prolog. Warszawa. WNT 1983.

4 Pereira L. M., Pereira F., Warren D. H. D.: User's Guide to DECSystem-10 Prolog. LNEC, Lisboa 1978.

Janusz Gocałek
Jacek Klauziński
Politechnika Poznańska - ITKB

FORTRAN - CRROK - PODSUMOWANIE

1. Wstęp

Niniejszy tekst jest niejako przedłużeniem referatu wygłoszonego na ubiegłorocznej I Konferencji Użytkowników [illegible] Mera- 400. Wobec dużego zainteresowania użytkowników systemem operacyjnym CROOK-4 a tym samym kompilatorem języka FORTRAN-CROOK to opracowanie stanowi jakby podsumowanie prac dotyczących tego kompilatora. Język Fortran stanowi dzisiaj podstawowe narzędzie pracy we wszystkich niemal instytucjach wykorzystujących komputer. W szczególności minikomputer Mera-400.

Istnieje więc nadal potrzeba rozwijania kompilatora pracującego w systemie Operacyjnym CROOK-4 i wzbogacania go o nowe mechanizmy i narzędzia w celu ułatwiania oraz polepszania pracy w tym języku.

Dla zaprezentowania tych mechanizmów wybraliśmy te, które budzą wśród użytkowników komputera Mera-400 najwięcej kontrowersji, uwagi i ciekawości, jako narzędzi nieodzownych dla lepszego wykorzystania maszyny.

2. Arytmetyka mieszana

Mieszanie typów w wyrażeniach arytmetycznych w początkowej wersji Kompilatora nie było przewidywane. Użycie arytmetyki mieszanej w wyrażeniach arytmetycznych jest często elementem wielu błędów popełnianych przez użytkowników w ich programach wydłuża ono również czas obliczeń programu.
Po wielu jednak naciskach ze strony użytkowników(co wynika z przyzwyczajeń wyniesionych z pracy na e.m.c Odra i Mera-400 -SOM-3) Kompilator został wzbogacony w arytmetykę mieszania typów.

Przyjęto zasadę, że operacje arytmetyczne wykonywane na liczbach dwóch różnych typów daje w wyniku wartość typu o poziom wyższego tzn. jeżeli wykonane jest mnożenie wartości typu REAL, INTEGER lub odwrotnie to wynik jest wartością typu REAL itd.

3. Nakładkowanie

Stanowi dla programisty jeden z podstawowych a może nawet najważniejszych elementów, w celu oszczędnego wykorzystywania pamięci operacyjnej.

Pomimo, że obecnie problem zainstalowanej pamięci w maszynie staje się elementem drugoplanowym, to jednak z punktu widzenia ekonomicznego pracy systemu i odpowiedniej pracy pozostałych użytkowników (wielodostęp) nakładkowanie nadal cieszy się dużym zainteresowaniem programisty.

Sposób tworzenia programów nakładkowanych nie wymaga od programisty wielkiego nakładu pracy oraz wiedzy. Mechanizm nakładkowania jest w tym kompilatorze bardzo elastyczny, ponieważ ten sam program może być przez dwóch różnych programistów, niewielkim nakładem pracy, różnie skonstruowany. Efekt uzyskany zależy głównie od użytkownika, od sposobu utworzenia nakładek i poziomów.

Programista ma do dyspozycji główny poziom nakładkowania (część rezydującą) oraz 9 poziomów nakładkowania. Każdy poziom może zawierać 99 nakładek. W deklaracji poziomów nakładkowania oraz identyfikacja nakładki odbywa się przy pomocy numerów. Do ściągania nakładki służy jedna procedura biblioteczna, która w swoich parametrach wołania potrzebuje numer poziomu nakładkowania oraz numer nakładki.

Najniższy poziom nakładkowania jest niedostępny dla programisty. Realizuje on wszystkie odwołania do programów bibliotecznych współpracujących z programem i również na stałę rezyduje w pamięci.

Ciało nakładki może zawierać dowolną liczbę segmentów, a początkiem każdej nakładki jest użyta dyrektywa #OVL. Użycie tej dyrektywy poraz pierwszy określa koniec części rezydującej a początek pierwszej nakładki. Występowanie dalszych dyrektyw #OVL oznacza koniec nakładki a początek następnej.

4. Zbiory binarne

Zbiory binarne są realizowane przez Kompilator jako zbiory

- bezpośredniego dostępu
- sekwencyjne

Deklaracja zbioru binarnego bezpośredniego dostępu jest realizowana przez dyrektywę #DEFINE FILE.

Dyrektywa posiada dwa parametry, numer strumienia oraz długość rekordu w słowach. Instrukcje wejścia i wyjścia są w tej implementacji inaczej rozwiązane. Instrukcja SETFILE ustala zbiór skojarzony ze srumieniem, na którym wykonywane będą operacje czytania i pisania. Instrukcje READ i WRITE natomiast zawierają jako swój parametr, numer transmitowanego rekordu. Operacje wejścia i wyjścia są wykonywane na danym zbiorze do momentu pojawienia się następnej instrukcji SETFILE, która ustala inny zbiór itd.

Zbiory binarne tworzone przez kompilator FORTRAN-CROOK, odpowiadają swoją strukturą odpowiednim zbiorom binarnym tworzonym przez język BASIC i CEMMA. Istnieje więc możliwość przekazywania informacji między programami napisanymi w różnych językach.

5. Programy o pamięci 64 K

Kompilator generuje programy fortranowskie, które jako spójne ciało może zajmować do 64 K pamięci operacyjnej.

Dla realizacji tak dużych programów nie trzeba wykonywać żadnych dodatkowych deklaracji. Nie muszą być przy tym dokonywane żadne zmiany w konstrukcji maszyny. Jedynym ograniczeniem jest stosowanie dużych tablic, dla których wskaźnik przekroczy dopuszczalną wartość dla liczb typu INTEGER.

6. Biblioteki i programy pomocnicze

Kompilator jest wyposażony w biblioteki systemowe i użytkowe. Na uwagę zasługują: biblioteka łańcuchowa, która umożliwia wszystkie operacje na tekstach. Duża biblioteka podprogramów matematycznych, która zawiera niemal wszystkie istotne algorytmy z różnych dziedzin matematyki.

Opracowana jest duża biblioteka umożliwiająca współpracę urządzeń kreślących typu Plotter, pisak X-Y itp.

Biblioteka jest 3-poziomowa i nie zależy od typu używanego sprzętu. Najniższy bowiem poziom tej biblioteki napisany jest w Assemblerze i może być dostosowany do odpowiedniego typu urządzenia.

Jako komplet XFOR-a stanowią również programy pomocnicze do czytania taśm papierowych przygotowanych w systemach SOM-3 i GEORGE-3. Program do tworzenia bibliotek assemblerowskich oraz procedury kopiujące.

L I T E R A T U R A:

[1] Z. Czerniak, M. Nikodemski, A. Bobcow:
-System operacyjny CROOK-4 dla minikomputera Mera-400
Instytut Okrętowy Polit. Gd. Gdańsk 1984

[2] J. Gocałek, J. Klauziński:
- Kompilator języka FORTRAN-CROOK dla minikomputera Mera -400 w systemie operacyjnym CROOK-4.
ITiKB Polit. Pozn. Poznań 1984

r inż. Ryszard Tuziemski
uro Projektów
downictwa Komunalnego
Gdańsku

RATFOR - PREPROCESOR STRUKTURALNEJ WERSJI JĘZYKA FORTRAN

1. WSTĘP

RATFOR jest strukturalną wersją języka FORTRAN opracowaną i zrealizowaną w Bell Laboratories przez Briana W. Kernighana [1]. Jego nazwa pochodzi od słów "Rational Fortran", co w wolnym przekładzie oznacza "rozsądny" lub "sensowny" Fortran.

RATFOR z pewnością zasługuje na popularyzację, szczególnie w warunkach polskich, ponieważ jest tanim i łatwym w użyciu narzędziem umożliwiającym natychmiastowe zastosowanie metod programowania strukturalnego na wszystkich maszynach posiadających kompilatory Fortranu, bez oczekiwania na powszechną dostępność języków takich jak Pascal.

Pomimo istnienia nowocześniejszych języków programowania Fortran jest wciąż najbardziej popularny i najszerzej rozpowszechniony, co nadaje mu cechy języka uniwersalnego umożliwiającego pisanie programów przenośnych. Nie umniejsza to jednak jego wad.

Fortran posiada prymitywne instrukcje warunkowe. Arytmetyczna instrukcja IF wymusza zastosowanie przynajmniej dwóch etykiet i dwóch implikowanych instrukcji skoku, co prowadzi

do pogorszenia czytelności kodu. Logiczna instrukcja IF umożliwia określenie bezpośrednio za warunkiem zależnej od niego akcji, lecz może to być jedynie pojedyncza instrukcja Fortranu z jeszcze dalszymi ograniczeniami. Poza tym instrukcja IF nie może mieć części ELSE - nie ma sposobu na określenie akcji alternatywnej w przypadku, gdy warunek nie jest spełniony.

Z powyższych przyczyn programy pisane w Fortranie zawierają wiele etykiet i instrukcji skoku, przez co są mało czytelne i trudne do zrozumienia, a co za tym idzie - trudne w uruchamianiu i modyfikacji.

Gdy ma się do czynienia z niewygodnym językiem programowania, jednym z możliwych wyjść jest zdefiniowanie nowego języka nie posiadającego wad starego i przetłumaczenie go na stary język przy pomocy preprocesora. To właśnie podejście zostało zastosowane przy opracowaniu RATFORU.

2. NIEFORMALNY OPIS JĘZYKA

RATFOR zachowuje zalety Fortranu takie jak uniwersalność, przenośność, efektywność usuwając jednocześnie jego niedogodności. Różnią go od Fortranu dwie cechy.

Po pierwsze, RATFOR ukrywa przed użytkownikiem instrukcje sterujące Fortranu dostarczając w to miejsce własne, znacznie wygodniejsze.

Po drugie, ponieważ preprocesor musi przejrzeć cały program źródłowy w celu przetłumaczenia struktur sterujących, może w tym samym czasie dokonać szeregu operacji "kosmetycznych", dzięki czemu język źródłowy może być dla programisty wygodniejszy w użyciu.

W rezultacie powstał język, w którym /znając Fortran/ można zacząć programować niemal natychmiast po przeczytaniu jego opisu.

Oto cechy języka RATFOR, które czynią go tak wygodnym narzędziem programowania.

2.1. Swobodny format źródłowy

Instrukcje RATFORU mogą pojawiać się w dowolnym miejscu wiersza. Ponadto w jednym wierszu może znajdować się kilka instrukcji, o ile oddzieli się je średnikami, np.

```
A = 1; B = 2; C = 3
```

Każda instrukcja rozpoczynająca się cyframi jest traktowana jako opatrzona etykietą fortranowską i odpowiednio formatowana. Pozwala to na umieszczanie takich instrukcji w dowolnym miejscu wiersza, jak również na nieliczenie spacji między etykietą, a instrukcją, np.

```
WRITE (5,100);    100 FORMAT (" HELLO ! " )
```

Konsekwencją swobodnego formatu jest możliwość czytelniejszego pisania programów poprzez eksponowanie warunków, od których uzależniona jest dana akcja, np.

```
IF (warlog)
        WRITE (5,900) X, Y, Z
```

2.2. Wyrażenia logiczne i relacji

Zapis operatorów logicznych i relacji jest /w ramach dostępnego repertuaru znaków/ maksymalnie zbliżony do notacji

matematycznej. Fragment instrukcji

```
IF (A <= B & C > D) ...
```

jest z pewnością bardziej czytelny niż równoważny mu zapis

```
IF (A.LE.B.AND.C.GT.D) ...
```

który również jest w RATFORZE dopuszczalny.
Pozostałym operatorem relacji, tj. ".NE.", ".LT.", ".GE." i ".EQ." odpowiadają znaki "!=", "<", ">=" i "==" - podwójny znak równości odróżnia operator relacji od operacji przypisania. Operatorowi ".OR." w oryginalnej wersji języka odpowiada znak kreski pionowej "|", w realizacji dla [illegible]RY-400 zastąpiony znakiem "!" z uwagi na dostępny alfabet.

2.3. Kontynuacja wiersza

RATFOR automatycznie kontynuuje wiersze kończące się jednym ze znaków:

```
+ - * / , ! & (
```

zakładając, że są to niezakończone instrukcje arytmetyczne bądź warunki logiczne. Pozwala to uniknąć żmudnego liczenia znaków w długich instrukcjach arytmetycznych, listach formatu, deklaracjach typu itp., np.

```
X = A * B +
    C/D - E
```

jest jedną instrukcją, podobnie jak

```
900 FORMAT (1X,"BARDZO DLUGI TEKST-CZESC I",
"BARDZO DLUGI TEKST-CZESC II")
```

Oczywiście istnieje możliwość zapobieżenia automatycznej kontynuacji tam, gdzie nie jest ona pożądana.

2.4. Komentarze

Początek komentarza sygnalizowany jest znakiem " # " występującym w dowolnym miejscu wiersza. Pozostała część takiego wiersza uważana jest za komentarz.

Umożliwia to umieszczania komentarzy w jednym wierszu z instrukcjami, co w wielu przypadkach zwiększa czytelność programu, np.

```
L = STL (S); P = STP (S); S = S-1  # ZDEJMIJ ZE STOSU
```

2.5. Pseudoinstrukcja DEFINE

Stosowanie stałych liczbowych występujących w postaci jawnej w wielu miejscach programu od dawna uważane jest za złą praktykę programowania. Przy pomocy pseudoinstrukcji DEFINE można zdefiniować taką stałą jako nazwę, co oprócz zwiększenia przejrzystości kodu pozwala na jego parametryzację i ułatwia przyszłe modyfikacje, np.

```
DEFINE   ROWS  100  # MAKS. LICZBA WIERSZY
DEFINE   COLS   50  # MAKS. LICZBA KOLUMN

DIMENSION  A (ROWS), B (ROWS, COLS)
...
IF (I <= ROWS & J <= COLS/ ...
```

Ogólnie dowolny ciąg znaków alfanumerycznych może być zdefiniowany jako nazwa; w konsekwencji gdy taka nazwa pojawi się na wejściu /ograniczona znakiem niealfanumerycznym/,

zastępowana jest pozostałą częścią wiersza definiującego /po odrzuceniu ewentualnego komentarza/.

2.6. Instrukcje złożone

Grupa instrukcji jest równoważna pojedynczej instrukcji po zamknięciu jej w parę nawiasów kwadratowych [...] lub klamrowych {...}. Nawiasy wydają się ze względu na ich bardziej zwartą formę niż słowa kluczowe begin ... end np. z języka Pascal; oprócz tego "end" ma określone znaczenie w Fortranie.
Ogólnie gdziekolwiek w RATFORZE może być użyta pojedyncza instrukcja, tam może wystąpić ich grupa zamknięta w nawiasy, np.

```
IF ( J>ROWS)
   [ERR=1; CAL ERROR ("OVERFL") ; RETURN]
```

Wewnątrz instrukcji złożonej mogą występować dowolne instrukcje, a więc również instrukcje złożone.

2.7. Instrukcje sterujące

O ile poprzednio wyliczone cechy języka RATFOR miały charakter głównie "kosmetyczny", o tyle instrukcje sterujące są czynnikiem decydującym o łatwości i przejrzystości wyrażania algorytmów w danym języku programowania. RATFOR posiada instrukcje umożliwiające programowanie strukturalne, co czyni go zbliżonym pod tym względem np. do języka Pascal.

2.8. Instrukcja DO

Instrukcja DO w RATFORZE jest analogiczna do instrukcji DO w Fortranie z jednym wyjątkiem - nie wymaga podania etykiety instrukcji granicznej cyklu, np.

```
DO  I=1, IMAX    # INICJALIZACJA  X(I)
      X(I) = Ø.
```

lub

```
DO  J=1, JMAX    # ZAMIANA  A(J)  Z  B(J)
[ T = A(J) ; A(J) = B(J) ; B(J) = T ]
```

Składnie instrukcji DO jest następująca:

DO Fortran - DO - Tekst

instrukcja-RATFORU

Po słowie kluczowym DO muszą wystąpić zmienna sterująca i parametry cyklu zgodne z regułami używanego kompilatora Fortranu.

2.9. Instrukcja FOR

Jest ona zbliżona do analogicznych konstrukcji w języka Algol czy Pascal lecz bardziej od nich uniwersalna, ponieważ jej krok nie jest ograniczony do postępu arytmetycznego. Składnia instrukcji FOR jest następująca:

FOR (ini; war; inkr)

instrukcja-RATFORU

gdzie "ini" jest pojedynczą instrukcją RATFORU wykonywaną raz przed rozpoczęciem pętli, "war" jest dopuszczalnym warunkiem dla logicznej instrukcji IF sprawdzanym na początku pętli, za "inkr" jest pojedynczą instrukcją RATFORU wykonywaną na końcu każdego obiegu pętli. Każdy z członów "ini", "war" i "inkr" może zostać opuszczony, jednak średniki muszą wystąpić. Opuszczony warunek jest traktowany jako zawsze spełniony, tak więc:

```
FOR (; ;)
```

definiuje nieskończoną pętlę.

Instrukcja FOR jest szczególnie użyteczna w pętlach "od N do 1", w przetwarzaniu struktur listowych, w pętlach mogących wykonać się zero razy i innych trudnych do wyrażenia za pomocą instrukcji DO, zaś niewygodnych do bezpośredniego zaprogramowania. Oto przykład programu sumowania elementów listy, której koniec sygnalizowany jest zerową wartością wskaźnika do następnego elementu /lista implementowana jest przy pomocy tablicy wskaźników i równoległej tablicy wartości elementów/:

```
SUM = Ø
FOR (I = FIRST; I > Ø ;  I = PTR (I))
   SUM = SUM + VALUE (I)
```

2.10. Instrukcja NEXT

Umożliwia ona wymuszenie następnego iteracji w dowolnym miejscu pętli DO lub FOR, np.

```
DO  I = 1,80   # PRZETWARZANIE ZNAKOW ROZNYCH OD SPACJI
   [ IF (CARD (I) = = BLANK)  NEXT
      ...
      BARDZO DUŻO INSTRUKCJI
      ... ]
```

2.11. Instrukcja BREAK

Pozwala ona na opuszczenie pętli DO lub FOR przed wyczerpaniem zadanej liczby iteracji, np.

```
DO  I = 1,80  # POMINIECIE POCZATKOWYCH SPACJI
   IF (CARD (I) != BLANK)  BREAK
```

2.12. Instrukcja IF...ELSE

Umożliwia ona warunkowy wybór drogi obliczeń, zaś jej składnia jest następująca:

```
IF (war)  instrukcja RATFORU
ELSE      instrukcja RATFORU
```

"war" analogicznie jest w instrukcji FOR i w następnych instrukcjach musi spełniać kryteria legalności dla fortranowskiej logicznej instrukcji IF, zaś"instrukcja RATFORU" może być podobnie jak w Pascalu dowolną instrukcją, a więc także instrukcją złożoną zawierającą inne instrukcje IF-ELSE, np.

```
IF ( A <= B)
  [ SW = Ø ; IF (LO == 1) WRITE (5,900) A,B ]
    ELSE [ SW = 1; WRITE (5,910) B,A ]
```

Część ELSE jest oczywiście opcjonalna.
Przy pomocy serii instrukcji ELSE IF można zasymulować nie występującą w RATFORZE instrukcję case:

```
IF  ...  ---
ELSE IF  ...  ---
...
ELSE  ---
```

2.13. Instrukcja WHILE

Analogicznie jak w języku PASCAL instrukcja ta służy do organizacji pętli o liczbie powtórzeń uzależnionej od warunku logicznego testowanego na początku pętli, np.

```
WHILE (A(I)>B(J)) [I = I+1;  J = J-1]
```

Składnia instrukcji WHILE ma postać:

```
WHILE (war)  instrukcja RATFORU
```

2.14. Instrukcja REPEAT - UNTIL

Używana jest ona do organizacji pętli o liczbie powtórzeń uzależnionej od warunku logicznego testowanego na końcu pętli, np.

```
REPEAT
  WHILE (A>B)   A = A-B
  WHILE (B>A)   B = B-A
UNTIL (A == B)
```

zaś jej składnia ma postać

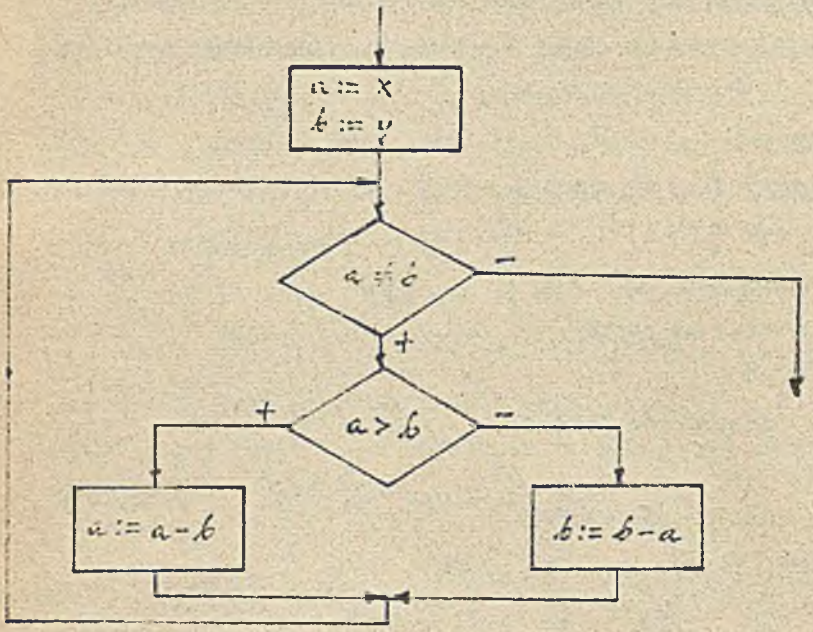
```
REPEAT
    instrukcja RATFORU
    ...
    instrukcja RATFORU
UNTIL (war)
```

2.15. Wiersze niestandardowe

Dowolny wiersz rozpoczynający się znakiem "%" jest przepisywany na wyjście bez jakichkolwiek zmian oprócz usunięcia znaku "%" i przesunięcia pozostałej części wiersza o jeden znak w lewo. Zapobiega to próbom interpretacji przez RATFOR np. dyrektyw sterujących oraz umożliwia włączanie do programu w RATFORZE fragmentów kodu, który nie powinien być modyfikowany, np. istniejących już programów w Fortranie.

3. PRZYKŁADY

Aby zademonstrować przewagę RATFORU nad Fortranem rozpatrzmy schemat blokowy przedstawiający algorytm obliczania największego wspólnego podzielnika liczb naturalnych X i Y .



Oto odpowiadający powyższemu schematowi fragment programu w RATFORZE /z pominięciem deklaracji typów zmiennych/:

```
A = X ;  B = Y
WHILE (A! = B)
    IF (A > B)  A = A-B
    ELSE   B = B-A
```

oraz równoważny schematowi ciąg instrukcji w "klasycznym" Fortranie:

```
      A = X
      B = Y
 10   IF (A.NE.B) GOTO  20
      GOTO  50
 20   IF (A.GT.B) GOTO  30
      B = B-A
      GOTO  40
 30   A = A-B
 40   GOTO  10
 50   ...
```

Bardziej wszechstronnie możliwości RATFORU demostruje drugi przykład będący adaptacją procedury sortowania szybkiego opisanej w [5] /program 2.11/.
Tabulogram kodu tej procedury zamieszczony jest w dodatku A. Porównanie tej wersji procedury z oryginałem wykazuje, że adaptacja programów z języka Pascal na RATFOR jest bardzo prosta /o ile programy te nie zawierają jawnej lub niejawnej rekursji/.

4. IMPLEMENTACJA

Pierwsza wersja RATFORU została napisana w języku C [6] pod systemem operacyjnym UNIX [7] przy wykorzystaniu kompilatora kompilatorów YACC [8].
Następnie powstała nowa wersja RATFORU napisana w nim samym i skompilowana przy pomocy wersji poprzedniej. W rezultacie wykorzystania przenośnego podzbioru Fortran uzyskano przenośną wersję RATFORU, która działała bez zmian na maszynach sześciu

różnych producentów.

Preprocesor RATFORU zrealizowany przez autora niniejszego artykułu napisany został w celu zapewnienia przenośności w FORTRANIE i działa na minikomputerze MERA-400 pod systemami operacyjnymi SOM-3 i CROOK-4. Planowane są wersje RATFORU dla maszyn serii ODRA oraz SM.

5. UWAGI KOŃCOWE

Użytkownicy uważają RATFOR za język znacznie wygodniejszy od Fortranu, ponieważ umożliwia on pisanie programów tak czytelnie, jak w nowocześniejszych językach programowania takich jak Pascal. Panuje opinia, że kodowanie w RATFORZE jest przynajmniej dwukrotnie szybsze niż w Fortranie, a co ważniejsze - szybsze i łatwiejsze jest uruchamianie i poprawianie programów. Jako przyczynę podaje się głównie przejrzystość otrzymywanego kodu i łatwość stosowania metod programowania strukturalnego. Od momentu powstania krąg użytkowników RATFORU rozszerza się. Świadczy o tym n.in. fakt, że RATFOR wchodzi w skład oprogramowania, z którym rozprowadzany jest system UNIX. Ponadto w oparciu o RATFOR zrealizowano interesującą koncepcję wirtualnego systemu operacyjnego [9] definiującego jednolite środowisko dla programów użytkowych na kilkunastu typach maszyn z różnymi oryginalnymi systemami operacyjnymi, począwszy od mikrokomputerów zaś skończywszy na dużych modelach maszyn IBM/370. Należy przy tym pamiętać, że w USA powszechnie dostępne są nowocześniejsze języki programowania.

W odczuciu autora niniejszego artykułu, w warunkach krajowych RATFOR jest szansą na poprawę jakości programowania bez ponoszenia nakładów finansowych i szkoleniowych związanych z wdrożeniem kompilatora nowego języka programowania: przejście z Fortranu na RATFOR jest natychmiastowe, ześ efekt w postaci wzrostu wydajności programowania - bardzo duży.

LITERATURA

[1] B.W. Krenighan: "RATFOR - a Preprocessor for a Rational Fortran", Softwave - Practice and Experience Vol. 5, 395-406 /1975/.

[2] ISO Recomendation R 1539-72 Programming Language FORTRAN .

[3] American National Standard Fortran, American National Standards Institute, New York 1966.

[4] PN-76/T-42111. Język programowania FORTRAN.

[5] N.Wirth: "Algorytmy + struktury danych = programy", WNT, W-wa, 1980.

[6] D.M. Ritchie, S.C. Johnson, M.E. Lesk, B.W. Krenighan: "The C Programming Language", The Bell System Technical Journal, Vo. 57, Jul-Aug 1978, No 6, Part 2.

[7] D.M.Ritchie, K.Thompson: "The UNIX Time - Sharing System", BSTJ, Vol. 57, No 6.

[8] S.C.Johnson, M.E. Lesk: "Language Development Tools", BSTJ Vol. 57, No 6.

[9] D.E.Hall, D.K.Scherrer, J.S.Sventek: "A Virtual Operating System", Communications of the ACM, Sept. 1980, Vol. 25, No 9.

```
PROGRAM RF12
;
=  TEST 12 - SORTOWANIE SZYBKIE
;
DEFINE MAX 50
;
REAL A(MAX); DATA N/MAX/;
;
;  GENERACJA DANYCH TESTOWYCH
;
FOR(I=1;I(=N;I=I+1)
  [ II=N-I+1; A(I)=II ]

WRITE(5,900)A; 900 FORMAT(/" DANE PRZED SORTOWANIEM"/5(1X,10F10.2/))

CALL QS1(A,N)

WRITE(5,910)A; 910 FORMAT(/" DANE PO SORTOWANIU"/5(1X,10F10.2/))
END
;
;
SUBROUTINE QS1(A,N)
;
;   PODPROGRAM SORTUJE N-ELEMENTOWA TABLICE A
;*********************************************
;
DEFINE MAX 15  ; MAKS. ROZMIAR STOSU DLA N=2**15 ELEMENTOW
;
REAL A(N); INTEGER STOSL(MAX),STOSP(MAX),I,J,L,P,S
;
S=1; STOSL(1)=1; STOSP(1)=N
REPEAT   ; WEZ ZADANIE Z WIERZCHOLKA STOSU
  L=STOSL(S); P=STOSP(S); S=S-1
    REPEAT   ; DZIEL A(L)...A(P)
      I=L; J=P; I1=(L+P)/2; X=A(I1)
        REPEAT
          WHILE( A(I)(X ) I=I+1
          WHILE( X(A(J) ) J=J-1
          IF(I(=J)
            [ W=A(I); A(I)=A(J); A(J)=W
              I=I+1; J=J-1 ]
        UNTIL( I)J )
        IF(J-L(P-I)
          [ IF( I(P ) ; ZAPAMIETAJ ZADANIE PODZIALU PRAWEJ CZESCI
            [ S=S+1; STOSL(S)=I; STOSP(S)=P ]
          P=J  ; SORTUJ LEWA CZESC
          ]  ELSE
          [ IF( L(J ) ; ZAPAMIETAJ ZADANIE PODZIALU LEWEJ CZESCI
            [ S=S+1; STOSL(S)=L; STOSP(S)=J ]
            L=I  ; SORTUJ PRAWA CZESC
          ]
    UNTIL(L)=P)
UNTIL(S==0)
;  END QUICKSORT1
RETURN; END
$JOB LFILE - KONIEC
$ASS SI ASC LO CSL
$EXE DLS USL
```

Janusz Gocałek
Jacek Klauziński
Politechnika Poznańska

MODULA-2: Między Pascalem a Adą.

1. Geneza powstania języka.

Coraz większą popularność zdobywa sobie nowy język programowania Modula-2 (MODUlar programing LAnguage).

Opracowany został pod koniec lat siedemdziesiątych przez samego twórcę języka Pascal Niklausa Wirtha w ETH Zurich.

Język Modula-2 pochodzi bezpośrednio z języków Pascal [1] i Modula [2]. Pascal został zaprojektowany jako język ogólnego zastosowania i po wdrożeniu w 1970 roku osiągnął szerokie zastosowanie. Natomiast Modula powstała na skutek eksperymentów z programowaniem wieloprocesowym i dlatego koncentrowała się na istotnych problemach nierozerwalnie związanych z tego typu zakresem zastosowania. Język ten został zdefiniowany i wdrożony eksperymentalnie w 1975 roku.

W roku 1977 w Instytut Fur Informatik ETH w Zurichu rozpoczęto program badawczy mający na celu zaprojektowanie systemu komputerowego (hardware i software) w pojęciu zintegrowanym. System ten (później nazwany LILITH) miał być programowany w jednym języku programowania wysokiego poziomu, który musiał sprostać wymaganiom zarówno programowaniu na poziomie języka wysokiego poziomu jak i programowaniu na niskim poziomie (tzw. assemblerowym).

Modula-2 powstała w wyniku starannych rozważań konstrukcyjnych jako język zawierający w sobie wszystkie aspekty Pascala i rozszerzający je o nowe pojęcia modułu. Ponieważ składnia nowo zaprojektowanego języka była bardziej zbliżona do składni języka Modula niż do Pascala , wybrano więc nazwę tego języka Modula-2.

2. Modula a Pascal.

Modula-2 jest językiem uniwersalnym , zawierającym najlepsze elementy swych poprzedników (Algol,Fortran,Pascal)

Główne różnice w stosunku do języka Pascal to:

- pojęcie modułu , oraz możliwość podziału programu na część definicyjną i implementacyjną. Każdy program napisany w Moduli jest modułem, który sam może składać się z kilku modułów. Obiekty zadeklarowane i zdefiniowane w jednym module mogą być użyte w drugim .
- Modula posiada bardziej systematyczną składnię od Pascala , ułatwiającą szybkie nauczenie się tego języka. A decyduje o tim fakt iż każda struktura rozpoczynająca się słowem kluczowym kończy się również słowem kluczowym tzn. jest odpowiednio zamknięta w nawias.

- możliwość pełnego wykorzystania mechanizmów wieloprogramowości, tzn.że może być realizowanych równocześnie kilka procesów. Istnieje również możliwość komunikowania się miedzy tymi procesami.
- dostęp do sprzętu. Modula-2 umożliwia dostęp z języka wysokiego poziomu do praktycznie wszystkich elementów sprzętu. Udogodnienia takie zawarte programach (modułach) bibliotecznych są maszynowo zależne a więc różne w różnych implementacjach tego języka.
- typ proceduroWy umożliwiający dynamiczne przypisywanie procedur do zmiennych. Typ proceduroWy umożliwia modułowi klienta włączenie swoich własnych procedur do modułu bibliotecznego.

Modula-2 jest językiem stanowiącym rozszerzenie Pascala ale posiada też szereg innych udogodnień oraz problemów , które w Pascalu były uciążliwe a niekiedy nie możliwe do prostego rozwiązania . Dlatego Pascal stawał się językiem niewygodnym i mało elastycznym.

Praktycznie niemożliwe było w Pascalu tworzenie bibliotek matematycznych. Zmienne o charakterze globalnym w Pascalu były widoczne zbyt szeroko i doprowadzało to często do nieprzewidzianych skutków. Modularna struktura Moduli umożliwia zarówno tworzenie bibliotek jak i ukrycie zmiennych lokalnych - tak , że stają się one niewidoczne w pozostałej części programu.

Brak rozłącznej kompilacji w pascalu uniemożliwiał często tworzenie dużych programów. Modula ma ten problem rozwiązany dzięki swojej strukturze modułowej.

Bardzo ograniczone w Pascalu były możliwości w zakresie wejścia/wyjścia. Modula nie posiada żadnych instrukcji wejścia/wyjścia. Operacje te są wykonywane przez moduły biblioteczne i mogą być w nieograniczony sposób modyfikowane i rozszerzane lub zmieniane. Kompilator języka Modula-2 rozpoznaje duże i małe litery.

Istnieje jeszcze szereg innych udoodnień języka Modula-2 w stosunku do Pascala.

3. Ogólny opis składni języka.

Słownik języka Modula-2 składa się z identyfikatorów, liczb , napisów ,operatorów i ograniczników. Liczby dzielą się na całkowite (bez znaku) i rzeczywiste. Jeżeli liczba je zakończona znakiem B, jest traktowana jako liczba ósemkowa. Znak H oznacza liczbę w zapisie szesnastkowym. Zakończenie liczby znakiem C wskazuje reprezentację znaku , którego wartość ósemkowa przedstawia liczba . Jest to liczba typu CHAR.

Operatory i ograniczniki są specjalnymi znakami,parami znaków lub słowami kluczowymi. Słowa kluczowe składają się tylko z dużych liter i nie mogą być użyte w charakterze identyfikatorów. Wyróżniami następujące słowa kluczowe języka Modula-2:

AND,ARRAY,BEGIN,BY,CASE,CONST,DEFINITION,DIV ,DO,ELSE,ELSIF,END,EXIT,EXPORT,FOR,FROM,IF, IMPLEMENTATION,IMPORT, IN,LOOP,MOD,MODULE, NOT,OF,OR,POINTER,PROCEDURE,QUALIFIED,RECORD, REPEAT,RETURN,SET,THEN,TO,TYPE,UNTIL,VAR, WHILE,WITH.

Każdy identyfikator występujący w programie musi być poprzedzony deklaracją. Niżej przedstawione identyfikatory są predefiniowane i ważne we wszystkich zdaniach języka:

ABS,BITSET,BOOLEAN,CAP,CARDINAL,CHAR,CHR, DEC,DISPOSE, EXCL,FALSE,FLOAT,HALT,HIGH,INC, INCL,INTEGER,NEW,NIL,ODD, ORD,PROC,REAL, TRUE,TRUNIC,VAL.

W Module istnieje pięć typów podstawowych, które są predefiniowane i opisane standardowymi słowami (identyfikatorami):

INTR,CARDINAL,BOOLEAN,CHAR,REAL.

Każdy programista pisz cy program w języku Modula ma do dyspozycji procedury standardowe. Do najważniejszych z nich należy zaliczyć podprogramy obliczające wartość bezwzględną (ABS), konwersji liter małych na duże (CAP), zamiany liczb typu CADINAL na REAL i odwrotnie. Do obsługi podstawowych funkcji matematycznych (SIN,COS,LOG,EXP,itd.) służy specjalnie zdefiniowany moduł biblioteczny.

4. Biblioteki.

W skład biblioteki systemowej wchodzą standardowe moduły. Wejście/wyjście jest realizowane przez moduł o nazwie InOut, w którym znajdują się procedury umożliwiające wprowadzenie i wyprowadzenie informacji w różnej postaci. (Read,ReadInt,ReadReal,WriteReal,WriteString, WriteLn,ReadHex,ReadOct,itd.)

Działanie na plikach jest możliwe dzięki modułowi bibliotecznemu Files, który posiada procedury biblioteczne umożliwiające otwieranie zbiorów, zmiany nazwy zbioru, usunięcie zbioru, kopiowanie bloku itp. (FileName, Create,Delete,Close, Rename,SetBlock,WriteBlock itd.)

Każda implementacja tego języka , a nawet każdy kto programuje w tym języku , mogą mieć własne sposoby obsługi i realizacji funkcji standardowych,podprogramów obsługi wejścia/wyjścia oraz obsługi zbiorów. Jednak dla przenaszalności programów przyjęto pewien standard (moduły wymienione wyżej), do którego należy się dostosować.

5. Przykład programu.

Niżej przedstawiony program jest napisany w języku Modula-2.

```
MODULE log2;
 FROM InOut IMPORT WriteString,WriteLn;
 FROM RealInOut IMPORT ReadReal,Done,WriteReal;
 VAR x,a,b,sum:REAL;
BEGIN
 WriteString("x= ");ReadReal(x);
 WHILE Done DO
  (* 1.0<=x<2.0 *)
  WriteReal(x,15);
  a:=x;b:=1.0;sum:=0.0;
  REPEAT
   (* log2(x):=sum+b*log2(a) *)
   a:=a*a;b:=0.5*b;
   IF a>=2.0 THEN
    sum:=sum+b;a:=0.5*a;
   END
  UNTIL b<1.0E-7;
  WriteReal(sum,16);WriteLn;
  WriteString("x= ");ReadReal(x)
 END;
 WriteLn
END Log2.
```

6. Uwagi końcowe.

Pierwsza implementacja języka Modula-2 była na komputerze PDP-11 w 1979 roku, a definicja tego języka została opublikowana jako Raport Techniczny w marcu 1980 roku. Od tego czasu język ten jest w codziennym użyciu w instytucie ETH w Zurichu. Po latach stosowania i testowania w różnych zastosowaniach rozpoczęto rozpowszechnianie kompilatora tego języka do innych ośrodków. Zainteresowanie tym kompilatorem rosło szybko ponieważ zawiera on silne narzędzie projektowania systemowego wdrażane na szeroko stosowanych minikomputerach.

Modula jest jeszcze jednym krokiem do przodu w rozwoju nowoczesnych języków programowania. Obok tak popularnych obecnie języków jak BASIC,FORTRAN 77 o..z nowoczesnych i bardziej rozbudowanych jak C, FORTH czy ADA ; Modula jest napewno językiem bardziej czytelnym , łatwym w nauce, oraz wygodniejszym ,w niektórych zastosowaniac. Modula jest łudząco podobna do Pascala a więc przestawienie się na styl programowania z pascala jest bardzo proste. Można przypuszczać , że popularność tego języka będzie rosła i tak samo jak Pascal stanie się językiem codziennego użytku.

Jest niemożliwe w tak krótkim artykule przedstawić wszystkie zalety Moduli. Przedstawione zostały jedynie ogólne uwagi na temat tego języka. Niżej przedstawiona literatura wzbogaci napewno wiadomości dotyczące Moduli.

Literatura:

[1]. N.Wirth
The Programming language PASCAL.
Acta Informatica 1,35-63 (1971).

[2]. N.Wirth
Modula: a language for modular multiprogramming.
Software-Practice and Experience,7,3-35 (1977).

[3]. N.Wirth
The Use of Modula.
Software-Practice and Experience,7,37-66 (1977).

[4]. N.Wirth
Design and Implementation of Modula.
Software-Practice and Experience,7,67-84 (1977).

[5]. N.Wirth
Programming in Modula-2.
Springer-verlag,Berlin Heidelberg New York 1983.

[6]. J.Holden,I.C.Wand
An Assessment of Modula
Software-Practice and Experience,10,593-622 (1980)

[7]. J.Hoppe
A Simple Nucleus Written in Modula-2:A Case Study.
Software-Practice and Experience,10,697-706 (1980)

[8]. P.Fuglewicz
Modula-2:Język lat osiemdziesiątych.
Informatyka 1,2,3 (1984)

[9]. W.Abramowicz
Modula-2 i LILITH zgodność metod i narzędzi informatycznych.
Informatyka 4 (1984)

Zenon Kapała
Instytut Okrętowy
Politechniki Gdańskiej

CHARAKTERYSTYKA I ZASTOSOWANIA JĘZYKA PROGRAMOWANIA C

Na początku lat sześćdziesiątych powstało wiele prac systematyzujących różne techniki programowania. Na bazie tych doświadczeń opracowano takie języki programowania jak Algol, Simula, Pascal. Nieco później C, Loglan, Ada.

Język C opracowali B.W. Kernighan i D.M. Ritchie w Bell Telephone USA. Był on projektowany z myślą o tworzeniu w nim podstawowego oprogramowania systemowego. Pierwsza wersja tego języka powstała na maszynie PDP 11. W nim również został napisany system UNIX.

Język ten posiada następujące właściwości:

1/ Cechy wysoce zorientowanych języków a więc:

- zapewnia modułowe pisanie algorytmów /posiada niezbędne mechanizmy komunikacji pomiędzy modułami/;
- daje pełną swobodę definiowania nowych typów zmiennych złożonych z wcześniej zdefiniowanych lub z podstawowych takich jak: znakowe, słowowe /całkowite jako liczba ze znakiem lub bez/, zmiennopozycyjne pojedyńczej precyzji i podwójnej, ponadto można definiować pola bitowe;
- posiada bogaty zestaw instrukcji warunkowych, instrukcji organizacji pętli, instrukcji wyboru, rekurencyjne wołanie funkcji;

- z instrukcji prostych można składać instrukcje złożone z możliwością tworzenia zmiennych lokalnych.

2/ Język C posiada również mechanizmy o konstrukcjach spotykanych w assemblerach, ale na znacznie wyższym poziomie abstrakcji. Ten wysoki poziom abstrakcji uwalnia użytkownika od architektury maszyny /reprezentacji liczb, sposobu adresowania, itp./. Programista sam może podpowiedzieć, które zmienne chciałby trzymać w rejestrach, która zmienna /rejestr/ jest adresem i do jakiego obiektu się odwołuje. Użytkownik sam zatem może dysponować zasobami pamięci np.: tworzyć listy, dowolne struktury danych.
Podstawowy repertuar operacji logicznych, arytmetycznych, operacji przesuwania, preferuje język ten do zastosowań specjalistycznych, takich jak pisanie systemów operacyjnych, języków programowania, przetwarzania danych. Wymienione cechy oraz takie jak statyczne /tylko podczas kompilacji/ nadawanie wartości zmiennym, przyporządkowanie adresów, powodują wysoce efektywną generację kodów maszynowych, zapewniają dużą szybkość obliczeń.

3/ Silnym narzędziem tego języka jest preprocesor.
Dyrektywy preprocesora pozwalają na: definiowanie makroinstrukcji, stałych, zmiany argumentów funkcji, odwołanie i łączenie procedur bibliotecznych.

Opisane cechy języka, a także duża czytelność napisanych w nim programów, uniwersalność, łatwość nanoszenie zmian, przejrzysta składnia spowodowały, że stał się językiem powszechnie używanym. Dzięki temu, że kompilator tego języka został napisany w nim samym, można język ten w krótkim czasie

zaimplementować na inną maszynę, zmieniając jedynie nieduże fragmenty programu.

O uniwersalności języka C świadczyć możne następujący przykład. Kompilator tego języka opracowywany w Instytucie Okrętowym został napisany w języku Basic. Wybór taki nie jest przypadkowy. Motywuje go fakt, iż Basic mimo wad ma niewątpliwą zaletę. Jest językiem konwersacyjnym, pozwalającym na szybkie, sprawne testowanie i uruchamianie nowych algorytmów. Basic - ext opracowany w I.O. był projektowany z myślą o takich pracach. W napisanym kompilatorze języka C pod Basicem, można było przygotować procedury konwersji liczb, wejścia i wyjścia, niezbędne przy budowaniu języka C. Następnie, przy niewielkich zmianach w składni języka Basic oraz niewielu dyrektywach procesora języka C dopasowujących składnię języka Basic do składni języka C otrzymano kompilator języka C napisanego w nim samym.

Takie rozwiązanie problemu pozwoliło przede wszystkim na skrócenie czasu projektowania i uruchamiania kompilatora języka C. Nie jest to oczywiście produkt optymalny. Ale, jak wiadomo, pielęgnacja oprogramowania zajmuje mniej więcej tyle czasu, co samo projektowanie. W następnym etapie, kiedy język testowany jest, można będzie gu usprawniać.

Ponadto powstały kompilator języka Basic jest dobrym narzędziem, nawet lepszym aniżeli sam kompilator języka C, jeśli chodzi o zastosowanie nieprofesjonalne /pisanie systemów operacyjnych/. Bowiem konwersacyjno-interpretacyjny język Basic - ext z wieloma funkcjami standardowymi pozwala

na łatwe budowanie algorytmów a jego cross-kompilator /z uwzględnieniem typów zmiennych zaproponowanych w komentarzach/ zwiększa wydatnie czas obliczeń.

mgr Leszek Czerwosz
Centrum Medycyny Doświadczalnej i Klinicznej
Polskiej Akademii Nauk

Makroassembler MAX

Makroassembler MAX jest nowym jednonakładkowym od 2.5 do 4 razy szybszym assemblerem od standardowego assemblera MAC. Dotyczy to zwłaszcza programów składających się z wielu małych segmentów.

Pomimo tego,że stanowi jeden moduł ładowania to nie wymaga wiele pamięci. Minimalnie potrzebuje 12 k słów. Aby przetłumaczyć zaś samego siebie wymaga 16 k słów.

Makroassemler MAX spełnia wszystkie wymagania opisane w podręczniku firmowym - opisie makroassemblera MAC. Dodatkowo posiada kilka cech rozszerzających działanie.

Możliwe jest definiowanie własnych makrodefinicji tzw makrosów wewnątrz segmentu,poza dyrektywą INS. Są one traktowane jako nierezydujące. W szczególności w programach fortranowskich możliwe jest zdefiniowanie wstawki assemblerowej zawierającej makrodefinicję a następnie używanie jej w tym segmencie w tej lub następnych wstawkach assemblerowych.

W operandach instrukcji dopuszczalne są łańcuchy znakowe,tak dla dyrektywy DFC.

Tabela symboli i tabela treści makrodefinicji zajmują wspólny obszar pamięci,są elastyczne. Tabela symboli drukowana jest w kolejności alfabetycznej.

Listing został nieco skrócony,jeżeli linia programu źródłowego nie generuje więcej niż 3 słowa to na wydruku zajmuje jeden wiersz.

Błąd jest zawsze sygnalizowany w wierszu w którym wystąpił.

Assembler stawia automatycznie znacznik EOF na strumieniu BO po przetłumaczeniu całego programu. Komunikat o końcu tanslacji wysyłany jest na lokalny strumień CO.

Obecna już kilkumiesięczna próbna eksploatacja assemblera wykazuje jego dużą przydatność w pracy,skracając nudne godziny oczekiwania na zakończenie kompilacji.

III BAZY DANYCH

Mgr Aleksander Pietrow
Mgr Jacek Skorupski
Ośrodek Informatyki Urzędu
Wojewódzkiego w Tarnobrzegu

FUNKCJE I STRUKTURA OPROGRAMOWANIA BD-83[1/]

Oprogramowanie BD-83 jest przeznaczone to tworzenia systemów przetwarzania danych na minikomputerze MERA-400. Ze względu na organizację zbiorów zapewniającą bardzo szybki dostęp do rekordu wg dowolnie zbudowanego identyfikatora, oprogramowanie może być stosowane do tworzenia złożonych systemów przetwarzania danych np. systemu planowania na podstawie rozwinięć technologicznych wyrobów.

W systemach przetarzania danych realizowanych na minikomputerze MERA-400 zbiory danych są przechowywane na dyskach i taśmach magnetycznych. Dane są wprowadzane do zbiorów za pośrednictwem monitorów ekranowych. Oprogramowanie BD-83 zapewnia równoczesne i niezależne wprowadzanie danych przez wielu użytkowników. Równolegle z wprowadzaniem danych mogą być wykonywane programy obliczeniowe oraz programy sporządzania wydruków. Wprowadzanie danych, które w niewielkim stopniu obciąża moźc obliczeniową jednostki centralnej, jest wykonywane z wyższym priorytetem. Dzięki temu rozwiązaniu czas obsługi użytkownika wprowadzającego dane jest krótki a jednostka centralna w pełni wykorzystana.

Oprogramowanie BD-83 może być użytkowane przy minimalnej konfiguracji minikomputera MERA-400 /pamięć operacyjna 32k słów, 1 jednostka dyskowa, 1 monitor, 1 drukarka znakowo-mozaikowa/. W tym przypadku równoległe wprowadzanie danych i przetwarzanie nie jest możliwe ze względu na zbyt małą pamięć operacyjną. Efektywne wykorzystanie oprogramowania BD-83 zapewnia konfiguracja: pamięć operacyjna 64k słów, 2 jednostki dyskowe, 4-8 monitorów ekranowych, 1-2 drukarki znakowo-mozaikowe. Oprogramowanie BD-83 może być również stosowane przy szerszych konfiguracjach minikomputera MERA-400.

Oprogramowanie BD-83 działa na zbiorach dyskowych i taśmowych. Każdy zbiór składa się z rekordów. Rekord składa się z pól /danych/.

1/Informacja o oprogramowaniu autorstwa dra A.Ziółkowskiego z Instytutu Badań Systemowych PAN.

Długość pola,liczba pól w rekordzie,maksymalna liczba rekordów w zbiorze są parametrami dla każdego zbioru określanymi w mimencie jego tworzenia.Każdy rekord posiada wyodrębniony identyfikator składający się z jednego lub kilku pól /identyfikator wieloczłonowy/.Oprogramowanie BD-83 zapewnia sekwencyjny i bezpośredni dostęp do rekordów.W przypadku dostępu bezpośredniego rekordy są odszukiwane po podaniu numeru rekordu lub po podaniu identyfikatora.Przy odszukiwaniu dokumentu wg identyfikatora wykorzystuje się rozproszoną tablicę indeksową /kodowanie mieszające/.Budowę zbiorów oraz zasady dostępu przedstawiono w rozdziale 2.

Oprogramowanie BD-83 składa się z programów działających w trybie konwersacyjnym oraz procedur,które mogą być dołączane do programów użytkowych w języku FORTRAN.Programy realizują dyrektywy użytkownika wprowadzone z klawiatury monitora ekranowego.Programy umożliwiają tworzenie i aktualizację zbiorów,sporządzanie wydruków kontrolnych zawartości zbiorów oraz proste przetwarzanie danych np.sortowanie, wykonywanie operacji arytmetycznych na polach.Przy relacji złożonego przetwarzania danych w systemach użytkowych,które wymaga napisania własnych programów,stosuje się procedury ułatwiające korzystanie ze zbiorów.

W skład oprogramowania BD-83 wchodzą następujące programy:

BAZA - program tworzenia i reorganizacji zbiorów.Program tworzy zgodnie z podanymi parametrami opisy zbiorów oraz rezerwuje potrzebne miejsce na dysku lub taśmie magnetycznej.Za pomocą programu można usuwać i kopiować zbiory.Można również reorganizować zbiory to znaczy zmieniać ich opisy /np.wprowadzić nowe pola do rekordów/.Na żądanie program sporządza wydruki opisów zbiorów.

DANE - wielodostępny program wypełniania zbiorów danymi. Program pozwala tworzyć,usuwać i poprawiać rekordy w zbiorach,których opisy utworzono wcześniej programem BAZA.

Program umożliwia równoczesną pracę wielu użytkowników zapewniając ochronę zbiorów.

DRUK - program drukowania zawartości zbiorów.
Program umożliwia drukowanie wybranych pól z rekordu. Użytkownik może określić warunki selekcji rekordu do wydruku. Na żądanie program oblicza sumy kontrolne lub wartości średnie. Program umożliwia sporządzanie wydruku zawartości zbioru wg pomocniczego zbioru indeksów.

LACZ - program przetwarzania danych.
Program ten realizuje w trybie konwersacyjnym kilka funkcji przetwarzania danych często występujących w systemach użytkowych jak łączenie dwóch zbiorów, uzupełnianie jednego zbioru danymi z drugiego zbioru, wykonywanie prostych operacji arytmetycznych, usuwanie dokumentów spełniających określone warunki selekcji.

KOMP - program kompresowania i dekompresowania zbiorów.
Zbiory mogą być składowane w formie skompresowanej, co zapewnia znaczną oszczędność pamięci dyskowej lub taśmowej potrzebnej do przechowywania /archiwowania/ zbioru.

Pisząc własne programy użytkowe programista może korzystać z następujących procedur oprogramowania BD-83:

OTWORZ - otwarcie zbioru. Przy otwieraniu zbioru określa się rekord logiczny przez podanie jakie pola będą używane w programie.

CZYTAJ - czytanie rekordu o podanym identyfikatorze.

PISZ - wpisanie nowej informacji do rekordu o podanym identyfikatorze.

CZYTNAST - czytanie kolejnego rekordu.

PISZNAST - wpisanie nowej informacji do kolejnego rekordu.

DOPISZ - zapisanie nowego rekordu na końcu zbioru.

USUN - usunięcie rekordu o podanym identyfikatorze lub wymazanie wybranych pól.

SZUKAJ - odszukanie pierwszego rekordu o podanym niepełnym identyfikatorze.

ZAMKNIJ - zamknięcie zbioru.

PORZ - porządkowanie rekordów wg zadanego klucza.

COPY - procedura pomocnicza do kopiowania łańcucha znaków.

Operowanie rekordem logicznym,który składa się z części pól rzeczywistego rekordu ułatwia stopniową rozbudowę systemu. Dodanie nowych pól do rekordu nie powoduje żadnych zakłóceń w uruchomionych wcześniej programach użytkowych.Programy te będą działały w dalszym ciągu poprawnie.Żadne modyfikacje nie są potrzebne.Określenie,które pola rekordu są używane w programie jest również istotne dla potrzed dokumentacyjnych.

W systemie BD-83 stosuje się 2 metody ochrony zbiorów. Pierwsza metoda polega na ograniczeniu dostępu do zbiorów w programie DANE tylko z wybranych monitorów ekranowych.W drugiej metodzie o uzyskaniu dostępu do zbioru decyduje znajomość wcześniej wprowadzonego hasła.Dla każdego zbioru wprowadza się osobne hasło dostępu do odczytu informacji ze zbioru oraz osobne hasło umożliwiające modyfikowanie zawartości zbioru.

Oprogramowanie BD-83 ułatwia uruchamianie programów użytkowych.Na żądanie można uzyskać wydruk śladu wykonania programu zawierający nazwy wywołanych procedur BD-83,wartości parametrów aktualnych oraz nazwy ewentualnych błędów.Za pomocą programu BAZA można wydrukować opisy zbiorów,które stanowią istotny element dokumentacji systemów użytkowych.Ze wzg względu na łatwość tworzenia zbiorów do testowania /za pomocą programu BAZA i DANE/programy użytkowe tworzące system użytkowy mogą być uruchamiane jednocześnie i niezależnie.W systemach użytkowych wykorzystujących oprogramowanie BD-83,wprowadzanie danych jest oddzielone od przetwarzania co znacznie ułatwia ich eksploatację.

Oprogramowanie BD-83 zapewnia współpracę z procesorem FMC.W programach konwersacyjnych systemu stworzono możliwość wykonywania dyrektyw systemu operacyjnego CREATE,REMOVE i ASSIGN co znacznie ułatwia operowanie na sekcjach założonych procesorem FMC.

BUDOWA ZBIOROW

W minikomputerze MERA-400 całą powierzchnię dysków magnetycznych podzielono na sekcje.Sekcje są logicznymi urządzeniami we-wy.Sekcja składa się z sektorów po 256 słów /512 znaków/. W pierwszym słowie zerowego sektora sekcji zawierającej zbiory oprogramowania BD-83 znajdują się znaki VV.Jedna sekcja może zawierać wiele zbiorów.Każdy zbiór ma 6-cio znakowy kod /identyfikator/ oraz 28-mio znakową nazwę.

Zbiór składa się z rekordów. Maksymalną liczbę rekordów określa się podczas tworzenia opisu zbioru, ale nie może ona być większa od 32000. Rekord składa się z pól. W jednym rekordzie może występować do 100 różnych pól.

Każde pole ma 6-cio znakowy kod /identyfikator/ oraz 28-mio znakową nazwę. Dla każdego pola określa się format. W programie DANE /wypełnianie zbiorów danymi/ długość pola nie może przekraczać 40 znaków. Dłuższe pola mogą być wypełniane jedynie za pomocą procedur. Procedura PORZ /i program SORT/ wprowadza ograniczenia na długość rekordu. Nie może on przekraczać 512 znaków. Zbiory o długości rekordu do 2048 znaków /programu/ można porządkować nieco wolniejszą wersją - programu SORT.

Każdy zbiór składa się z 4 części:
- opisu zbioru
- opisów pól
- tablicy adresowej
- informacji właściwej /rekordów/

Każda część zajmuje całkowitą liczbę sektorów.
Opis zbioru zawiera ogólne informacje o zbiorze i zajmuje 1 sektor. Na kolejnych słowach sektora zapisane są następujące dane:

1 -NAST -następny zbiór - nr pierwszego sektora następnego zbioru. Jeżeli nie ma więcej zbiorów na sekcji zmienna przyjmuje wartość zero.
2 -PTA -początek tablicy adresowej - nr pierwszego sektora tablicy adresowej.
3 -PINF -początek informacji właściwej - nr pierwszego sektora zawierającego rekordy.
4 - 5 -KODZ -kod zbioru - 6-cio znakowy identyfikator zbioru zapisany na 2 słowach formatem 2B3.
6 -19 -NAZZ -nazwa zbioru.
20-22 -DAT --data aktualizacji - wprowadzona automatycznie na podstawie datownika komputerowego data wprowadzenia ostatnich zmian w zbiorze /dopisania, usunięcia, poprawienia dokumentów/.
23 -NGEN -nr generacji zbioru.
24-26 -HASŁO-hasło /odczyt/ - 6 dowolnych znaków.
27-29 -HASŁZ-hasło /zapis/ /odczyt/ - j.w.
30 -MASO -maska dostępu użytkownika /odczyt/ - zmienna

Zmienna zawierająca zakodowaną informację czy użytkownik pracujący przy kolejnym monitorze /pierwszemu monitorowi odpowiada najmniej znaczący bit/ może odczytywać informację ze zbioru /bit=1/ czy nie /bit=0/.

31 -MASZ -maska dostępu użytkownika /zapis/ /odczyt/ - j.w. lecz dotyczy zapisu.
32 -MLD -maksymalna liczba rekordów w zbiorze.
33 -DLD -długość rekordu w znakach.
34 -DTA -liczba elementów tablicy adresowej.
35 -LDA -liczba pól w rekordzie.
36 -DKLU -długość identyfikatora w znakach.
37 -SEK -nazwa sekcji dyskowej.
38 -ALD -aktualna liczba rekordów w zbiorze.
39 -TYP -typ zbioru.
40 -KAS -nazwa kasety dyskowej.
41-43 -SOR -numery klucza pól ostatniego sortowania.
44 -nieużywane.
45 -ULD -liczba usuniętych rekordów.
46-50 -zmienne pomocnicze używane w procedurach.

Słowa 51-256 są nieużywane.W programach użytkowych opis zbioru jest dostępny za pośrednictwem 50-cio elementowej tablicy Z, będącej parametrem formalnym procedury OTWORZ.

Opisy pól rozpoczynają się od drugiego sektora zbioru.Każde pole rekordu posiada swój opis.Opis jednego pola ma 20 słów.W jednym sektorze mieści się 12 opisów /240 słów/.Słowa 241-256 są nieużywane.Tak więc opisy pól rekordu złożonego z 12 pól zajmują 1 sektor a rekordu złożonego z 13 pól 2 sektory.W kolejnych słowach opisu pola zapisane są następujące dane:

1- 2 - KODD - kod pola - 6-cio znakowy identyfikator pola zapisany na 2 słowach formatem 2C3
3 - LZD - długość pola w znakach
4 - RODZ - format pola F ≤ RODZ = 0,1,2,3,4 - liczba znaków po kropce
A - RODZ = 5
I ≤ RODZ = 9
5- 18 - NAZD -nazwa pola
19 - ZNPD - nr pierwszego znaku pola w rekordzie

20 - ZMP - zmienna pomocnicza używana w programie BAZA
Tablica_adresowa składa się z wielu elementów.Liczbę elementów określa dana DTA w opisie zbioru.Każdy element tablicy adresowej składa się z identyfikatora rekordu oraz numeru rekordu w zbiorze.Długość rekordu,nr pierwszego sektora zawierającego rekordy /dane DLD i PINF z opisem zbioru/ oraz nr rekordu w zbiorze wyznaczają położenie pierwszego znaku rekordu.Długość identyfikatora rekordu jest parametrem określanym przy tworzeniu opisu zbioru /dana DKLU z opisem zbioru/.Długość elementu tablicy adresowej zależy od długości identyfikatora.Identyfikator w tablicy adresowej zajmuje pełną liczbę słów /parzystą liczbę znaków/. W przypadku gdy długość identyfikatora jest nieparzysta, identyfikator w tablicy adresowej jest uzupełmiany znakiem#. Przy zapisywaniu elementów tablicy adresowaj granice sektorów są ignorowane.Element tablicy adresowej może zaczynać się w jednym sektorze a kończyć w następnym.Aby zapewnić szybki dostęp do rekordów elementy tablicy adresowej są wypełniane przy wprowadzaniu rekordów wg algorytmu kodowania mieszającego.Ten sam algorytm jest następnie używany przy odszukiwaniu rekordów.Podczas tworzenia opisu zbioru tablica adresowa jest wstępnie zerowana.Wartość funkcji mieszającej zależy od wszystkich znaków identyfikatora.W przypadku wystąpienia kolizji /przepełnienie/ przeglądane są kolejne elementy tablicy adresowej.Po dojściu do ostatniego elementu tablicy przeglądanie jest kontynuowane od pierwszego elementu.
Informacja właściwa stanowi ostatnią część zbioru.Składa się ona z rekordów umieszczanych w kolejności ich wprowadzania.Kolejność ta może być zmieniona w wyniku porządkowania zbioru.Rekordy są przechowywane w postaci alfanumerycznej.Granice sektorów są ignorowane.Rekord może zaczynać się w jednym sektorze a kończyć w następnym.Wszystkie sektory są wstępnie wypełniane znakami#. Rekordy Usunięte są oznaczane przez zapisanie znaku # jako pierwszego znaku rekordu.Rekordy są usuwane podczas kompresowania zbioru,które ma miejsce przed sortowaniem oraz przy otwieraniu zbiorów procedurą OTWORZ.

Dr inż. Witold Rekuć
Instytut Organizacji i Zarządzania
Politechniki Wrocławskiej

UNIWERSALNY SYSTEM ZARZADZANIA BAZA DANYCH DLA MINIKOMPUTEROW SERII MERA-400

Celem niniejszego referatu jest prezentacja podstawowych cech zrealizowanego w Instytucie Organizacji i Zarządzania Politechniki Wrocławskiej systemu zarządzania bazą danych dla minikomputerów serii MERA-400.

Uniwersalny system zarządzania bazą danych /USZBD/ o nazwie STEP-400 jest pakietem programów i procedur służących do definiowania struktury bazy danych, przygotowania programów użytkowych przetwarzających bazę danych oraz kontroli i reorganizacji bazy danych. System ten został opracowany jako oprogramowanie minikomputera MERA-400 o podstawowej /i większej/ konfiguracji z systemem operacyjnym SOM-3. Bazowym językiem programowania w systemie STEP-400 jest język SIMBOL-400 [2].

1. Struktura bazy danych

USZBD STEP-400 umożliwia zarządzanie bazą danych o uproszczonej strukturze sieciowej /modelu sieciowym [1]/. Podstawową jednostką danych jest rekord będący ciągiem zmiennych prostych i tablic jednowymiarowych. Zarówno zmienne proste jak i tablice mogą być typu: TEXT, INTEGER, REAL, DOUBLE, LOGICAL lub CAN. Rekordy o identycznej strukturze i nazwie tworzą zbiory rekordów jednakowego typu.

Każdy rekord zapamiętany w bazie danych posiada identyfikator

wewnętrzny - klucz bazy danych, który jest unikalnym adresem tego rekordu w bazie danych.

Na określonych typach rekordów /zbiorach rekordów jednakowego typu/ można zdefiniować relację zwaną typem setu. Set jest strukturą charakterystyczną dla podejścia sieciowego, w której jednemu rekordowi danego typu przyporządkowane jest n rekordów innego typu. W danym typie setu jeden typ rekordu pełni rolę właściciela typu setu, drugi typ rekordu - rolę członka danego typu setu.

Wprowadzono stały podział rekordów na dwie kategorie: właściciel /OWNER/ i członek /MEMBER/. Typ rekordu zdefiniowany w jednym typie setu jako OWNER/MEMBER, ma tę samą kategorię w pozostałych typach setów , w których uczestniczy.

Przy tworzeniu typów setów obowiązują następujące zasady:

1/ typ setu jest relacją zdefiniowaną dokładnie na dwóch typach rekordów,

2/ właścicielem typu setu musi być typ rekordu kategorii OWNER, natomiast członkiem - typ rekordu kategorii MEMBER,

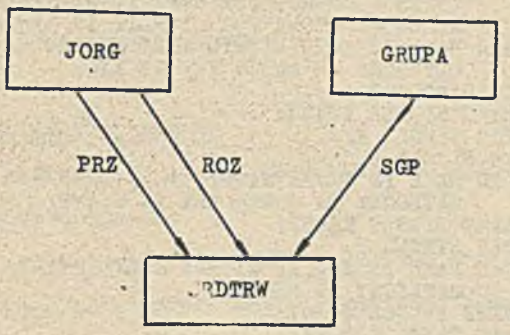
3/ dany typ rekordu kategorii OWNER może być właścicielem wielu typów setów i analogicznie typ rekordu kategorii MEMBER może być członkiem wielu typów setów,

4/ na parze typów rekordów /OWNER i MEMBER/ można definiować wiele typów setów,

5/ typ setu jest relacją jeden do wielu, tzn. jeden rekord kategorii MEMBER posiada tylko jednego właściciela w danym typie setu.

Opisane wyżej zasady przyjęte zostały także np. w systemie TOTAL [3].

Przykład prostej struktury bazy danych przedstawia rys. 1.



JORG - rekord "Jednostka organizacyjna"
GRUPA - rekord "Grupa środków trwałych"
SRDTRW - rekord "Srodek trwały"
PRZ - set "Przychód środków trwałych"
ROZ - set "Rozchód środków trwałych"
SGP - set "Srodki trwałe w grupie"

Rys. 1. Diagram struktury przykładowej bazy danych.

Powiązania rekordów w setach realizowane są przy pomocy łączników adresowych. We wszystkich setach przyjęto jednakową realizację powiązań: każdy rekord MEMBER posiada wbudowany łącznik wskazujący na poprzedni /PRIOR/, następny /NEXT/ rekord w łańcuchu oraz na rekord właściciel /BACK/.

Rekordy poszczególnych typów pamiętane są w jednostkach pamięci bazy danych zwanych obszarami. W systemie STEP-400, realizacją obszaru bazy danych jest sekcja dyskowa zdefiniowana w systemie operacyjnym SOM-3. Do jednego obszaru można zapisywać rekordy od 1 do 15 typów dowolnej kategorii. Rekordy danego typu pamiętane są w obrębie jednego obszaru. Obszar podzielony jest na fragmenty zwane segmentami, które stanowią macierzysty nośnik rekordów jednego typu. Podczas eksploatacji następuje zapełnianie

```
SCHEMA NAME IS TESTBA;
       ACCESS-CONTROL FOR COPY IS MARY;
       ACCESS-CONTROL FOR DISPLAY IS ZENEK.
RECORD NAME IS JORG WITHIN AREA AR1;
       CATEGORY IS OWNER KEY IS KJO;
  KJO SINGLE T(5);
  JNAME SINGLE T(30);
  STAN SINGLE R;
  SAM SINGLE L.
RECORD NAME IS GRUPA WITHIN AREA AR1;
       CATEGORY IS OWNER KEY IS GNAME;
  GNAME SINGLE T(45);
  STAN SINGLE D.
RECORD NAME IS SRDTRW WITHIN AREA AR1;
       CATEGORY IS MEMBER;
  SNAME SINGLE C(50);
  DATP TABLE T(2,3);
  DATR TABLE T(2,3);
  WARTP SINGLE R;
  WARTR SINGLE R;
  INWENT SINGLE T(10);
  DOKUM SINGLE T(3);
  NDOK SINGLE I;
  RESER SINGLE T(400).
SET NAME IS ROZ; OWNER IS JORG; MEMBER IS SRDTRW.
SET NAME IS SGP; OWNER IS GRUPA; MEMBER IS SRDTRW.
SET NAME IS PRZ; OWNER IS JORG; MEMBER IS SRDTRW.
AREA NAME IS AR1;
     ASSIGNED TO STREAM BZD;
     ALLOCATED 960 SECTORS ON-DEVICE 9425;
     OPEN-CONTROL IS ZENEK;
  RECORD JORG IS OWNER WITH 10 ENTRIES LENGTH 22 WORDS
       NUMBER OF-RECORDS IS 30;
  RECORD GRUPA IS OWNER WITH 5 ENTRIES LENGTH 29 WORDS
       NUMBER OF-RECORDS IS 99;
  RECORD SRDTRW IS MEMBER LENGTH 237 WORDS
       NUMBER OF-RECORDS  300.
END-SCHEMA.
```

Rys. 2. Przykład schematu bazy danych.

segmentów rekordami, a w przypadku przepełnienia - wypożyczanie pozycji od segmentów przypisanych innym typom rekordów. Gospodarowanie pamięcią obszaru bazy danych realizowane jest metodą tworzenia łańcuchów pozycji wolnych, zajętych i skreślonych.

Strukturę bazy danych definiuje się przy pomocy języka opisu danych /JOD/. Definicja bazy danych jest oddzielona od specyfikacji procedur przetwarzania bazy danych i nosi nazwę schematu

bazy danych. Rys. 2. przedstawia przykład schematu bazy danych o strukturze uwidocznionej na rys. 1.

2. Operacje manipulacji danymi

Operacje manipulacji danymi podzielone są na trzy grupy:

1/ operacje odczytu danych:

a/ odczyt rekordu kategorii OWNER o zadanym identyfikatorze zewnętrznym /pole wyróżnione w rekordzie, będące typu TEXT/,

b/ odczyt pośredni rekordu kategorii MEMBER jako kolejnego rekordu, w stosunku do aktualnie odczytanego /następny, poprzedni, pierwszy lub ostatni w secie/,

c/ odczyt rekordu dowolnej kategorii o zadanym kluczu bazy danych,

d/ odczyt rekordu w sekwencji wzrastającej wartości klucza bazy danych /odczyt seryjny/,

e/ odczyt rekordu kategorii OWNER jako właściciela określonego setu przy odczytanym rekordzie członkowskim tego setu,

2/ zapis danych:

zapis rekordu dowolnej kategorii do bazy danych,

3/ modyfikacja bazy danych:

a/ włączenie zapisanego rekordu MEMBER do określonego setu,

b/ wyłączenie rekordu MEMBER z określonego setu,

c/ skreślenie z bazy danych rekordu dowolnej kategorii,

d/ zmiana zawartości rekordu dowolnej kategorii.

Operacje manipulacji danymi działają na bazie danych w trybie jednorekordowym.

Do specyfikacji operacji w bazie danych służy język manipulacji danymi /JMD/, Każdej operacji odpowiada jedna komenda JMD.

Komendy JMD wstawiane są do tekstu procedur pisanych w języku SIMBOL-400 na zasadach obowiązujących dla instrukcji tego języka. Rys. 3. prezentuje przykład tekstu programu przetwarzającego bazę danych o strukturze uwidocznionej na rys. 1 i 2.

```
PROGRAM LJORG;
 /* LADOWANIE REKORDOW JORG
SINGLE TEXT ( KJO 5, JNAME 30, OPCONT 12='ZENEK'),
       REAL (STAN),
       INTEGER (STATUS),
       LOGICAL (SAM=.T.);
SINGLE INTEGER (STRONA=0, LINIA=60);
ARRAY INTEGER (KBDCU (3))
BEGIN
      %INVKE (TESTBA*JORG);
      %OPENA (AR1*OPCONT*STATUS);
      CALL BLAD(20); CALL NEXPAG;
      PUT LO & 33,'LADOWANIE  =JORG=';
      REPEAT
            GET SI /,(KJO),(JNAME);
            ONERROR ET1;
            STAN:=STAN+1000;
            IF SAM=.T. THEN SAM:=.F. ELSE SAM:=.T.;
            %WRITO (JORG*KJO=KJO,
                        JNAME=JNAME,
                        STAN=STAN,
                        SAM=SAM*KJO*KBDCU*STATUS);
            CALL BLAD(3); CALL NEXPAG;
            PUT LO (KJO(1,5));
      FOREVER;
      GOTO ET2;
ET1:  IF ERR=58 THEN BEGIN CALL NEXPAG;
                           PUT LO '$$ - KONIEC ZBIORU SI'END
                ELSE BEGIN CALL NEXPAG;
                           PUT LO 'ERROR = ',(ERR) END;
ET2:  %CLSEA(AR1*STATUS);
      CALL BLAD(22);
END
```

Rys. 3. Przykład programu przetwarzającego bazę danych.

3. Oprogramowanie USZBD STEP-400

W skład oprogramowania USZBD STEP-400 wchodzą następujące procesory:

DDLCOM - translator języka opisu danych: przekształca schemat bazy danych w metabazę, tzn. opis bazy danych

w postaci dogodnej do odczytu przez procesory systemu,

PRESIM - preprocesor języka SIMBOL-400: zamienia komendy JMD zawarte w tekscie procedury na standardowe wywołania procedury manipulacji danymi,

ALOCAT - procesor inicjujący obszar bazy danych,

DISPLA - procesor drukujący definicje wskazanych jednostek danych, zdefiniowanych w schemacie /metabazie/,

PRINT - procesor drukujący informacje o zawartości wskazanych fragmentów bazy danych,

REORG - procesor reorganizujący obszar bazy danych,

STEP. - procedura realizująca operacje manipulacji danymi.

4. Wykorzystanie USZBD STEP-400

Administrator projektujący bazę danych specyfikuje jej strukturę przy pomocy JOD, tworząc schemat bazy danych. Schemat ten podlega weryfikacji przy pomocy procesora DDLCOM, który przekształca go w metabazę.

Opracowanie schematu i utworzenie metabazy jest pierwszym krokiem realizacji systemu przetwarzania danych, wykorzystującym USZBD STEP-400. Administrator może na bieżąco uzyskiwać informacje o strukturze bazy danych posługując się procesorem DISPLA. Procesor ten umożliwia drukowanie wskazanych definicji jednostek danych celem ich przekazania np. projektantom i programistom zastosowań.

Programy użytkowe specyfikuje się w j. SIMBOL-400 i JMD. Tworzenie kodu wynikowego programów użytkowych odbywa się przy pomocy preprocesora PRESIM oraz następnie przy pomocy procesorów systemu SOM 3 BCS: SIM, MAC i EDI. Procesor EDI dołącza do programu użytkowego procedurę STEP., realizującą operacje manipulacji

danymi.

Pierwotne ładowanie bazy danych musi być poprzedzone alokacją /inicjacją/ obszarów bazy danych. Służy temu procesor ALOCAT, który zapisuje do wskazanej sekcji dyskowej dane organizacyjne specyficzne dla alokowanego obszaru. Zapisu danych użytkowych do bazy danych dokonują programy użytkowe.

W procesie eksploatacji bazy danych, administrator może korzystać także z procesorów: PRINT i REORG.
Procesor PRINT umożliwia uzyskiwanie wydruków dotyczących zawartości bazy danych: wskazanych rekordów lub wskazanych obszarów.
Procesor REORG, na zlecenie administratora reorganizuje obszary bazy danych, gdy ich wypełnienie rekordami nie spełnia przyjętych wymogów. Wtedy tworzony jest nowy obszar, który może mieć inną wielkość, inny podział na segmenty itp. Reorganizacja ta nie dotyczy struktury logicznej bazy danych.

5. Uwagi końcowe

Na zakończenie prezentacji systemu dwie uwagi dotyczące możliwości jego zastosowania i rozwoju.

Zrealizowany system STEP-400 może być zastosowany w dziedzinach zastsowania języka SIMBOL-400. Główne jego zastosowanie to systemy przetwarzania danych różnego typu. Mechanizmy oferowane przez system nie nakładają istotnych ograniczeń na struktury danych oraz na funkcje /operacje/ określone na tych strukturach. W tym sensie system STEP-400 jest uniwersalny.

Rozwój systemu STEP-400 mógłby być prowadzony w kilku kierunkach, określonych przez typowe funkcje systemów tego typu:
- kontrola współbieżności przetwarzania bazy danych,
- odtwarzanie bazy danych,

- reorganizacja bazy danych,
- ochrona integralności i poufności bazy danych.

Innym możliwym kierunkiem rozwoju systemu mogą być języki opisu i manipulacji danymi.

Przedstawiony tu system jest rozwiązaniem kompromisowym między pożądanymi cechami USZBD i środkami przeznaczonymi na jego realizację. Wykonawcy systemu są przekonani o jego dużej przydatności do realizacji systemów przetwarzania danych.

Literatura

1. Date C.J. Wprowadzenie do baz danych. WNT, Warszawa 1981.
2. Majewski J.T., Wagner K. Język SIMBOL dla MERY-400. Informatyka Nr 10, 1983.
3. Tsichritzis D.C., Lochovsky F.H. Data Base Management Systems. New York, San Francisco, London, Academic Press 1977.

mgr Maria Meler-Kapcia
Instytut Okrętowy
Politechnika Gdańska

RELACYJNA BAZA DANYCH SELKO
- KIERUNKI ROZWOJU

1. WSTĘP

Relacyjna baza danych SELKO opracowana została w Instytucie Okrętowym Politechniki Gdańskiej na minikomputer MERA-400 pracującym pod systemem operacyjnym CROOK-4.
Od ponad roku funkcjonuje ona w Stoczni Remontowej "RADUNIA" w zakresie fakturowania usług oraz rozrachunków z kontrahentami. Wdrażana jest w Gdańskich Zakładach UNIMOR, WZSP, Elektrowni KOZIENICE oraz w dyrekcji naszego instytutu. Ponadto testowana jest w kilkunastu ośrodkach, np. w Warszawie, Łodzi, Gliwicach, Zgierzu, Raszynie. Do realizacji typowych zadań wystarczająca jest konfiguracja MERY-400 z pamięcią operacyjną 32 KS, jedną jednostką dyskową, drukarką, monitorem ekranowym i ewentualnie jednostką taśmową do zrzutów.

Wszystkie programy bazy danych wykonują się w trybie konwersacyjnym w języku zbliżonym do naturalnego, zapewniając prostotę obsługi, co jest istotne zwłaszcza w przypadku użytkowników nie będących programistami. Łatwość formułowania nawet złożonych zadań uzyskano dzięki wprowadzeniu, obok typowych funkcji baz danych, dodatkowych, nistandardowych funkcji, praktycznie eli-

minując konieczność pisania programów użytkowych.

Wśród kilku, prezentowanych na minikomputer MERA-400, systemów baz danych najbardziej zbliżona do przedstawionej jest baza danych VITRON; zbliżona dlatego, że posiada również cechy organizacji relacyjnej.

Zasadnicze różnice to:

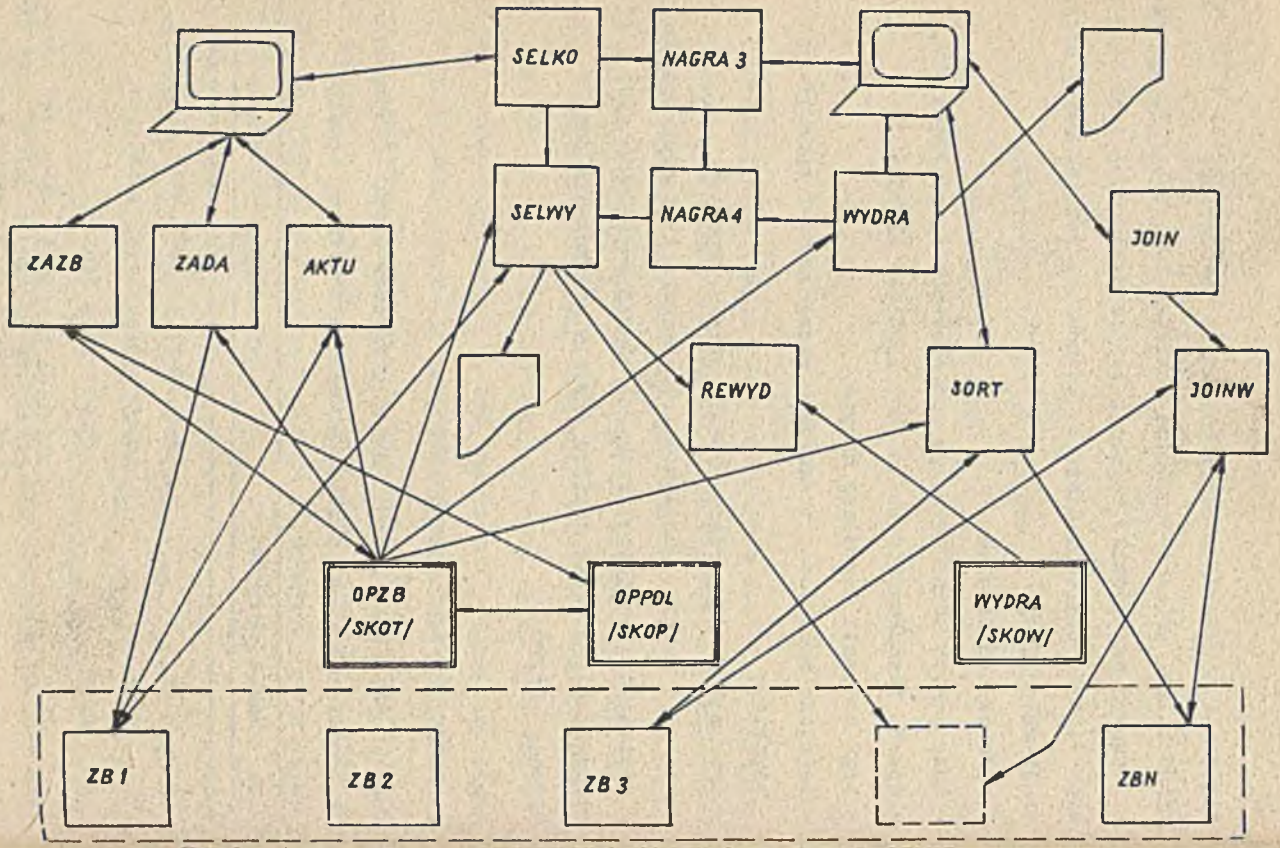
- baza danych VITRIN pracuje pod systemem operacyjnym SOM,
- sposób formułowania zadań jest bardziej złożony,
- brak kontroli formalnej danych,
- mniejsze możliwości w zakresie redagowania i sporządzania raportów,
- brak możliwości przechowywania opisów raportów,
- znacznie mniejsze możliwości w zakresie łączenia zbiorów.

W odniesieniu do pozostałych baz danych porównanie jest o tyle utrudnione, że są to dopiero systemy gromadzenia danych i utrzymywania zbiorów.

2. STRUKTURA BAZY DANYCH SELKO

Baza danych SELKO stanowi zbiór programów /rys. 1/ opracowanych do realizacji zadań przedstawionych w poniższej tabeli:

Lp.	Nazwa programu	Realizowane zadania
1	2	3
1	ZAZB	zakładanie zbiorów bazy danych
2	ZADA	zapisywanie danych w zbiorze
3	AKTU	aktualizacja danych
4	SELKO	selekcja danych i wydruk lub przepisywanie do innego zbioru bazy danych - część konwersacyjna


SELKO
NAGRA 3
SELWY
NAGRA4
WYDRA
JOIN
ZAZB
ZADA
AKTU
REWYD
JORT
JOINW
OPZB
/SKOT/
OPPOL
/SKOP/
WYDRA
/SKOW/
ZB1
ZB2
ZB3
ZBN

1	2	3
5	SELWY	j.w. - część wykonawcza
6	NAGRA3	formułowanie nagłówka i zakończenia raportu
7	REWYD	rejestracja opisów raportów jako wydruków standardowych
8	WYDRA	sporządzanie wydruków standardowych
9	NAGRA4	tworzenie nagłówka i zakończenia wydruków standardowych
10	SORT	sortowanie zbioru wejściowego z zapisem w zbiorze wyjściowym
11	JOIN	łączenie zbiorów o różnych strukturach rekordu - cz. konwersac.
12	JOINW	j.w. - część wykonawcza
13	KATZ	wydruk skorowidzów: zbiorów /OPZB-SKOT/ oraz pól /OPPOL-SKOP/
14	KATW	wydruk skorowidza wydruków /WYDRA-SKOW/
15	KAZB	kasowanie zbiorów bazy danych
16	AWAR	zmiana liczby rekordów w opisie zbioru.

3. FUNKCJE STANDARTOWE DZIAŁANIA NA ZBIORACH BD

3.1. Zakładanie zbiorów

W bazie danych SELKO istnieją dwa poziomy opisu zbiorów:

- fizyczny, zawierający tytuł i typ zbioru, liczbę rekordów, długość w bajtach oraz lokację zbioru na dysku. Opis ten przechowywany jest w skorowidzach zbiorów systemu operacyjnego,
- logiczny, zawarty w dwóch skorowidzach bazy danych: w pierszym - typ rekordu oraz lista pól, w drugim - parametry posz-

czególnych pól.

Odrębny opis zbiorów i pól pozwolił na wyeliminowanie wielokrotnego opisywania pól należących do wielu zbiorów. Zakładając zbiór użytkownik deklaruje: nazwę zbioru, typ zbioru oraz listę pól zbioru.

Dla pól, nie figurujących jeszcze w skorowidzu pól, musi określić nazwę, typ i długość pola oraz format dla pól numerycznych typu F.

Przewidziano następujące typy pól: X - alfanumeryczny - dopuszczalne wszystkie znaki; A - alfabetyczny - wyłącznie litery 9 - numeryczny - cyfry, bez możliwości działań arytmetycznych; F - numeryczny formatowany oraz D - data. Każdej czynności związanej z wprowadzaniem lub aktualizacją danych towarzyszy kontrola poprawności formalnej w zależności od typu pola i jego parametrów.

3.2. Zapisywanie danych

Celem dokonania zapisu w zbiorze bazy danych użytkownik określa nazwę zbioru, a następnie numery bądź nazwy pól, w dogodnej dla siebie kolejności, do których będzie wprowadzał dane. Rekordy mogą być umieszczane za istniejącymi już w zbiorze lub zapisywane od początku zbioru, jeśli dotychczasowe rekordy mają zostać skasowane.

Zapis w zbiorze bazy danych uwarunkowany jest poprawnością danych zgodnie z zadeklarowanymi parametrami pola, gdyż dane poddawane są szczegółowej kontroli formalnej, a ewentualne pomyłki mogą być korygowane na bieżąco dzięki wprowadzeniu różnorodnych udogodnień.

3.3. Aktualizacja danych

Aktualizacja zbiorów bazy danych SELKO obejmuje:
- uzupełnienie wybranych rekordów, w których nie wpisano danych do wszystkich pól,
- modyfikację danych zapisanych w polach,
- kasowanie rekordów.

Przystępując do aktualizacji użytkownik podaje: nazwę zbioru, listę pól aktualizowanych oraz opcjonalnie listę pól, według wartości których będą wyszukiwane rekordy.

W trakcie aktualizacji zbiór może być przeszukiwany sekwencyjnie bez podawania parametrów lub wyrywkowo na podstawie numeru rekordu bądź wartości pól, zadeklarowanych jako klucz do wyszukiwania. Po analizie wyświetlonych wartości pól użytkownik ma możliwość:
- pozostawić rekord bez zmian,
- wpisać nowe wartości do pól, wyszczególnionych do aktualizacji,
- skasować rekord.

W procesie aktualizacji prowadzona jest również formalna kontrola danych, a w przypadku wystąpienia błędu użytkownik może skorzystać z udogodnień korekcyjnych.

3.4. Sortowanie zbiorów

Sortowanie zbiorów w bazie danych SELKO odbywa się przy użyciu algorytmu sortowania szybkiego. Sortowane są wartości pól kluczowych lub ich części, a następnie według indeksów zapisywane są w zbiorze wynikowym rekordy w porządku rosnącym lub malejącym na żądanie.

Zbiory sortowane mogą być według dowolnych pól lub ich fragmentów w kolejności zadanej przez użytkownika.

W celu posortowania zbioru użytkownik określa nazwę zbioru wejściowego, nazwę zbioru wyjściowego, listę pól kluczowych oraz porządek sortowania.

4. SELEKCJA I WYDRUKI INFORMACJI Z BAZY DANYCH

Selekcję połączoną z wydrukiem informacji z bazy danych realizuje program SELKO, który umożliwia w bardzo łatwy sposób projektowanie i sporządzanie raportów przez użytkownika. Raporty mogą obejmować nie tylko część zasadniczą o postaci tabelarycznej informującą o zawartości zbioru, ale ponadto mogą posiadać złożony nagłówek i zakończenie /poniżej części zasadniczej/.

Nagłówek może zawierać elementy stałe jako dowolne ciągi znaków, jak również elementy zmienne, którymi mogą być wartości ze zbioru, zależności logiczne oraz wartości deklarowane na bieżąco np. aktualna data lub nazwisko wystawiającego.

W zakończeniu mogą zostać obliczone i wydrukowane statystyki dotyczące przetwarzanego zbioru.
Rozplanowanie nagłówka i zakończenia może być na bieżąco wielokrotnie korygowane i wyświetlone aż do uzyskania postaci akceptowanej przeh użytkownika.

Część zasadnicza raportu oprócz informacji ze zbioru może zawierać dane wyliczone w trakcie selekcji, przeformatowane, stałe numeryczne lub alfanumeryczne. Mogą one podlegać selekcji i wydrukowi na podobnych zasadach, jak pola ze zbioru, a ponadto

służyć do dalszych obliczeń.

Pola mogą być drukowane w dowolnym układzie /dowolnej kolejności/ poziomym oraz pionowym /jedne pod drugimi/.

Zasadnicza część wydruku może obejmować wszystkie bądź wybrane, w oparciu o zdefiniowaną przez użytkownika zależność logiczną, rekordy zbioru.

Istnieją następujące możliwości wydruku na podstawie zawartości zbioru:

- najprostsza postać wydruku, w której pola drukowane są w kolejności ich występowania w rekordzie. Ta postać wydruku nie wymaga od użytkownika żadnych deklaracji,
- postać wydruku, w której deklaruje się listę i układ pól drukowanych.
  Możliwe jest drukowanie jednego pola pod drugim, tj. na wielu poziomach,
- j.w. oraz lub wyłącznie wartości pól obliczonych, części pól i ewentualnie stałych. Pola takie traktowane są równorzędnie z polami występującymi w rekordzie,
- j.w., a ponadto sum pionowych grup rekordów uporządkowanych według klucza sumowania np. sumy zarobków pracowników według wydziałów itp.,
- sum częściowych, w których wydruk dotyczy wyłącznie grup rekordów posortowanych według klucza sumowania. W obrębie tej grupy mogą być zsumowane i wydrukowane wartości pól spełniających zdefiniowane zależności logiczne.

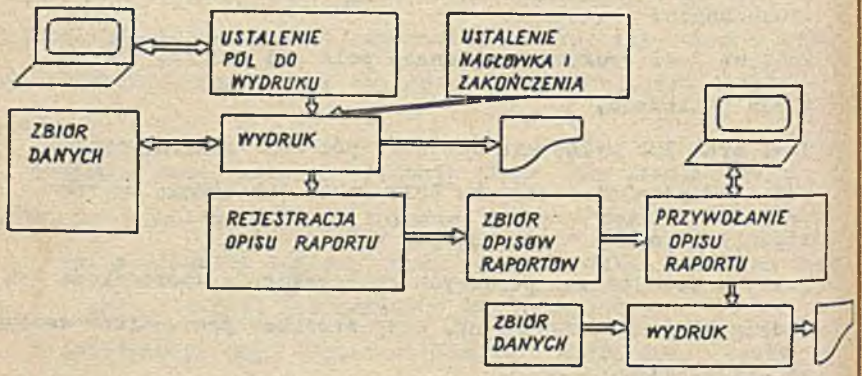
Niezależnie od selekcji użytkownik może zdecydować od którego rekordu ma nastąpić wydruk.

Informacja może być wyprowadzana nie tylko na drukarkę lub

monitor, ale również do zbioru w celu dalszego przetwarzania lub drukowania.

W przypadku, gdy zaprojektowany raport ma złożoną strukturę i będzie często sporządzany, wówczas jego opis można umieścić w specjalnym zbiorze opisu wydruków nadając mu nazwę. Sporządzenie raportu na podstawie przechowywanego opisu wymaga jedynie podania jego nazwy, nazwy zbioru wejściowego oraz urządzenia lub zbioru wyjściowego. /rys. 2/.

Opcjonalnie możliwa jest zmiana lub deklaracja zależności logicznych. Ponadto jeśli w nagłówku lub zakończeniu występowały wartości zmienne wymagane jest ich wprowadzenie.



Rys. 2. Przebieg tworzenia i rejestracji raportów

5. ŁĄCZENIE ZBIORÓW

Istotną rolę w oprogramowaniu systemu relacyjnej bazy danych odgrywa łączenie zbiorów o różnych strukturach rekordu. W bazie danych SELKO warunkiem połączenia rekordów dwu zbiorów jest spe-

nienie zależności logicznych zdefiniowanych przez użytkownika. Zależności logiczne zawierają operandy, będące wynikami porównań pól rekordów.

Łączenie polega na przeniesieniu zawartości określonych pól z jednego rekordu do drugiego i ewentualnie dokonaniu podstawień, jeżeli spełniona jest zależność logiczna. Obliczenia i przeniesienia mogą być realizowane w obrębie rekordów jednego zbioru lub w obrębie rekordów dwóch, jak również trzech zbiorów.

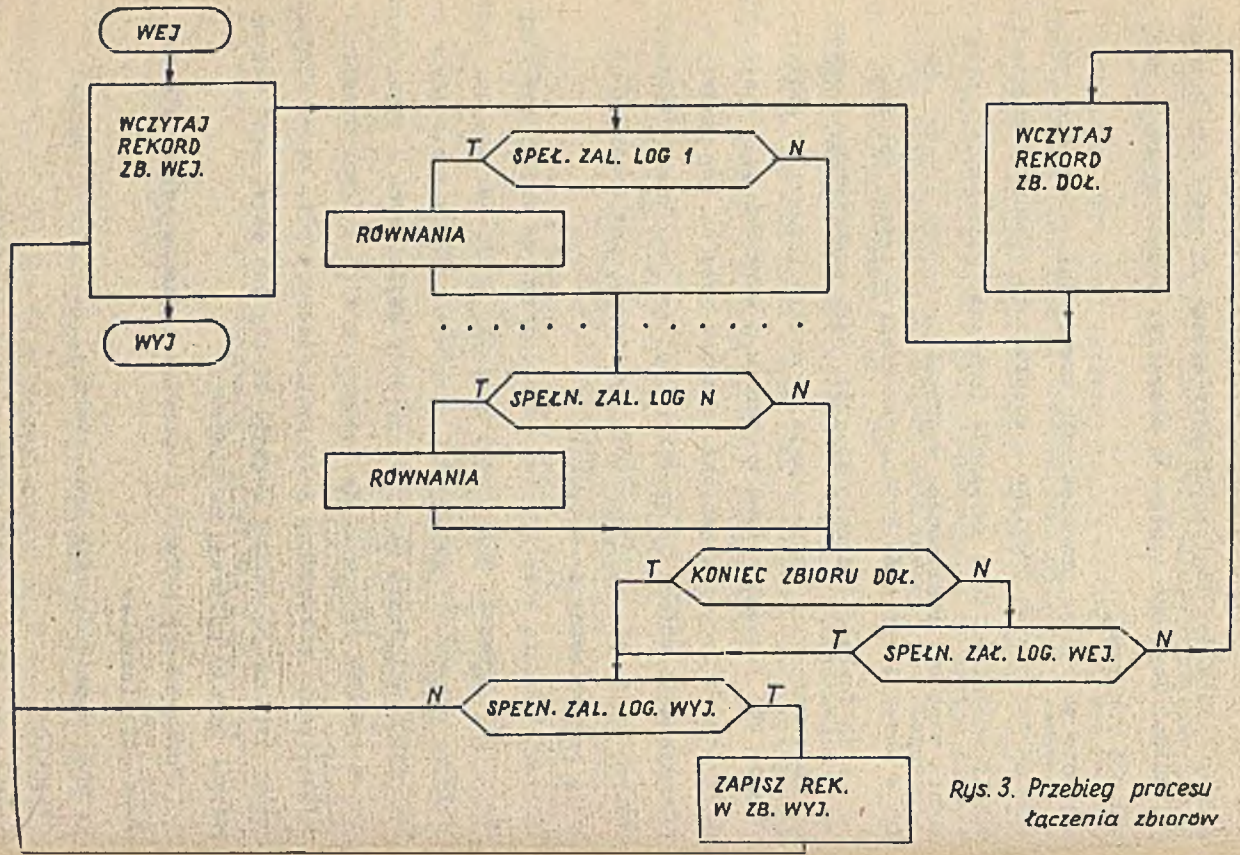
Poza funkcją łączenia zbiorów, jako zasadniczą, program łączenia może mieć zastosowanie do kontroli wartości poszczególnych pól jednego zbioru w oparciu o wartości pól innego zbioru /kontrola logiczna danych/. Ponadto może on być pomocny w przygotowaniu zbioru do wydruku; możliwe jest bowiem łączenie różnych typów pól w jednym polu wynikowym, zawierającym np. linię przygotowaną do wydruku.

Program łączenia może również pośredniczyć we współpracy bazy danych z programami napisanymi w języku BASIC lub innych, umożliwiając usuwanie i wstawianie dowolnych znaków kontrolnych celem przeformatowania rekordów zbiorów wejściowych i wyjściowych.

W trakcie łączenia, na żądanie, a w niektórych wypadkach automatycznie dokonywane jest przekodowanie pól, które obejmuje: zmianę typu pola, zmianę formatu liczby lub daty, bądź zmianę postaci liczby cyfrowej na znakową.

Formułowanie zadania w zakresie łączenia odbywa się poprzez definiowanie:

- równań bezwarunkowych oraz stałych,
- warunków,
- zależności logicznych, opartych na w/w warunkach,



Rys. 3. Przebieg procesu łączenia zbiorów

- równań w tym podstawień warunkowych, tj. równań, które będą liczone w przypadku spełnienia zależności logicznej,

a w fazie wykonawczej:

- wskazanie numerów zależności logicznych do określenia zakresu przeszukiwania zbioru dołączanego oraz umieszczenia rekordów w zbiorze wyjściowym.

Przebieg procesu łączenia zbiorów w b.d. SELKO został przedstawiony na rys. 3.

6. UWAGI OGÓLNE

Baza danych SELKO została zaprojektowana jako proste i dogodne narzędzie dla szybkiego wdrażania systemów przetwarzania danych z zakresu dowolnych zastosowań. Celem sprawdzenia walorów użytkowych przyjętej koncepcji i założeń bazy danych opracowano i oprogramowano wersję modelową systemu w języku BASIC, jedynym dostępnym do niedawna języku wyższego rzędu na minikomputer MERA-400.

Oprogramowanie bazy danych SELKO ma strukturę modularną, w której poszczególne moduły realizują typowe funkcje relacyjnych baz danych. Obok typowych funkcji wprowadzono wiele dodatkowych, niestandardowych mechanizmów eliminujących potrzebę pisania programów użytkowych. W każdym z modułów zastosowano konwersacyjny tryb pracy umożliwiający łatwe formułowanie nawet złożonych zadań, co jest szczególnie użyteczne na etapie wdrażania systemu. W późniejszym okresie eksploatacji, po dopracowaniu trybu przetwarzania, możliwe jest stworzenie z dostępnych modułów programowych specjalizowanego pakietu, przeznaczonego do przetwarzania w trybie wsadowym.

Opracowany model bazy danych posiada pewne ograniczenia, wynikające z oprogramowania systemu w języku BASIC, a dotyczące głównie czasu przetwarzania.
Rzutuje to na wielkość przetwarzanych zbiorów, które nie powinny przekraczać kilku tysięcy rekordów, aby czas przetwarzania nie był większy od kilku do kilkunastu minut w przypadku typowych zadań użytkowych.

W trakcie eksploatacji bazy danych SELKO wprowadzono dodatkowe udogodnienia, sugerowane przez użytkowników. Aktualnie zasadniczym kierunkiem prowadzonych prac jest zwiększenie efektywności działania systemu, co można osiągnąć przechodząc z języka BASIC na assembler. Kompleksowe przeprogramowanie systemu z języka BASIC na assembler wymagałoby znacznego nakładu pracy, dając w efekcie jedynie skrócenie czasu dostępu, co dla większości użytkowników nie jest tak istotne. Dla szybkiej poprawy efektywności działania bazy danych realizuje się dwa kierunki prac, a mianowicie:

- oprogramowanie wersji assemblerowej podstawowych modułów systemu /jak: selekcja informacji, sortowanie szybkie, dużych zbiorów/ kompatybilnych z istniejącymi, ale o ograniczonym zakresie możliwości.
  Do realizacji w pierwszej kolejności wybrano moduły, w których poprawa efektywności daje największe korzyści czasowe i efektywny wielodostęp przy minimalnej konfiguracji pamięci.
- zastąpienie w istniejących modułach procedur /realizujących działania czasochłonne/ napisanych w języku BASIC procedurami napisami w assemblerze.

Obydwa kierunki zmierzają w konsekwencji do przejścia z języka BASIC na assembler w taki sposób, aby użytkownik mógł na bieżąco otrzymywać szybciej działające wersje programów, a sposób pracy i obsługi pozostał bez zmian.

Wykaz literatury:

1. Atre S. "Data Base Structured Techniques for Design Performance and Management", A Wiley - Int. Publ., 1980.
2. Date C.J. "An Introduction to Database Systems" Addison - Wesley Publishing Company Inc. 1977.
3. Hutt A.T.F. "A Relational Data Base Management System" A Wiley - Int. Publ., 1979.
4. Meler-Kapcia M. "Realizacja funkcji systemów zarządzania bazą danych jako podstawa ich klasyfikacji. Informatyka 1979 nr 4.
5. Meler-Kapcia M. "Oprogramowanie systemu bazy danych materiałów i części zamiennych statku", Mat. Konferencji naukowej Cybernetyka w gospodarce morskiej, Sopot 1981.
6. Meler-Kapcia M. "Opis realizacji modelu relacyjnej bazy danych na minikomputer MERA-400". Oprac. wewn. I.O. PG nr 389/83, Gdańsk, 1983.

IV SYSTEMY APLIKACYJNE

Mgr Urszula Woźniak
Mgr inż. Andrzej Boboow
Stocznia Remontowa "Radunia"
Mgr inż. Wojciech Ziółkowski
Ośrodek Obliczeniowy
Politechniki Gdańskiej

ZASTOSOWANIE MERY - 400 W STOCZNI REMONTOWEJ "RADUNIA"

WPROWADZENIE

Stocznia Remontowa "Radunia" do roku 1980 była typowym przedsiębiorstwem usługowym pod względem organizacji i zastosowań informatyki. Były tu eksploatowane użytkowe systemy statystyczne takie jak "gospodarka materiałowa", "f-k", "pracochłonność" oraz parę innych dodatkowych jednostek systemowych charakterystycznych tylko dla stoczni remontowych. Informatyka w Stoczni organizacyjnie była oparta na schemacie: rejestracja i kontrola danych -- transport danych do Ośrodka Informatyki oddalonego od zakładu o około 10 km -- przetwarzanie danych -- transport tabulogramów wynikowych do Stoczni.

Taki schemat eksploatacji systemów użytkowych powodował zbyt długi obieg dokumentów źródłowych ze względu na czas oraz ilość szczebli organizacyjnych. Brak było możliwości nałożenia odpowiedzialności na określone stanowiska pracy za prawidłowe przygotowanie dokumentów oraz właściwe wprowadzenie danych do komputera.

Ponadto, systemy użytkowe, jak wspomniano wyżej, miały charakter typowo statystyczno-sprawozdawczy gdyż przetwarzanie informacji odbywało się najkrócej w cyklach dekadowych.

Z uwagi na fakt, że przetwarzanie odbywało się w Ośrodku Obliczeniowym oddalonym od Stoczni, brak było możliwości rozwoju systemów o jednostki dostarczające informacji niezbędnych do bieżącego sterowania zakładem.

Ważnym elementem rozwoju informatyki jest również czynnik psychologiczny. W 90 % zakładów wdrożenie jakiegokolwiek systemu użytkowego napotyka na bariery ludzkie, psychologiczne, tym bardziej, że wyniki przetwarzań służyły przeważnie do kontroli obliczeń wykonanych ręcznie.

Sytuacja gospodarcza kraju oraz całkowicie odmienny sposób zarządzania przedsiębiorstwem spowodowany wdrażaniem reformy gospodarczej, zmusiły przedsiębiorstwo do zainteresowania się takimi urządzeniami, które pozwalałyby na prawie natychmiastowe określenie stanu produkcji i sytuacji finansowej Stoczni.

Sama Stocznia jako przedsiębiorstwo usługowe charakteryzuje się nietypowością swych usług ze względu na krótkie terminy remontów wykonywanych w tzw. międzyrejsach, czyli w czasie załadunku lub wyładunku statku w porcie. Oczywiście są to remonty nietypowe, różnorodne. W tej sytuacji przewidywanie rzetelnego wyniku finansowego opartego na pełnym portfelu zamówień staje się wręcz niemożliwe.

Należało przejść do badania wyniku finansowego od strony wykonania a więc do codziennej analizy wartości, ilości i jakości wykonanych usług oraz stanu konta finansowego.

1. AKTUALNY STAN INFORMATYKI W STOCZNI

W roku 1981 Stocznia podjęła decyzję o całkowitej modernizacji sposobu wykorzystania informatyki w zakładzie. Do współpracy w tym zakresie zaproszono specjalistów z Instytutu Okrętowego Politechniki Gdańskiej.

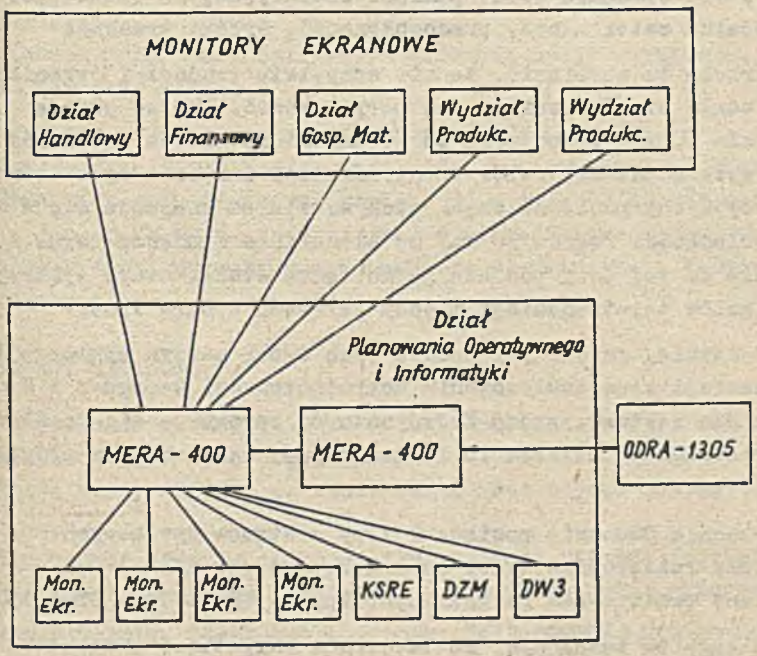
Wtedy też zdecydowano się na zakup minikomputera MERA-400.

Za racjonalnością tego zakupu przemawiały następujące argumenty:

1. Możliwość zakupu za polskie złotówki.
2. Krajowy serwis.
3. Wykluczenie pośrednich etapów obiegu dokumentów - zejście z końcówkami do twórcy dokumentu i jednocześnie odbiorcy informacji.

4. Potrzeba automatu, który służyłby nie tylko do obróbki i przetwarzania danych.
5. Istniał wykonawca takiego oprogramowania systemu operacyjnego i systemów użytkowych, który zapewnił nam pokonanie barier typu wielodostępność, wieloprogramowość. Wykonawca ten zapewnił nam również możliwość zautomatyzowania obróbki i przetwarzania informacji za pomocą funkcjonalnych makrozleceń.

Organizacja informatyki w Stoczni na dzień dzisiejszy przedstawiona została na rys. 1.



Rys. 1. Schemat organizacji informatyki w Stoczni

Rola Działu Planowania Operatywnego i Informatyki sprowadza się obecnie do nadzoru nad eksploatacją systemów użytkowych i właściwym wykorzystaniem minikomputerów oraz do doradztwa i pomocy ściśle specjalistycznej, natomiast odpowiedzialnością za prawidłowe wykonanie obliczeń obciążono inne zainteresowane komórki organizacyjne.

Z uwagi na odzyskane moce przerobowe, ci sami pracownicy, którzy wprowadzali dane oraz byli obciążeni kontrolą tych danych zostali wykorzystani do celów projektowo-programowych, wdrożeniowych jak również do koordynowania tych części systemów użytkowych, które wymagają dużej pamięci operacyjnej na Odrze-1305 /gospodarka materiałowa, pracochłonność, środki trwałe/.

Trzeba tu nadmienić, że nie wszystkie trudności organizacyjne udało się Stoczni do tej pory pokonać. Chodzi przede wszystkim o brak zezwolenia ze strony Ministerstwa Łączności na tzw. łącze stałe dla celów teletransmisji danych z wydziałów produkcyjnych, ponieważ część tych wydziałów znajduje się w dalszej odległości /około 20 km/ od stanowiska minikomputera. Stocznia do tej pory posiada jedno łącze stałe, które wykorzystuje do celów teletransmisji między Merą-400 a Odrą 1305.

Jednakże, dzięki połączeniu tych dwóch maszyn systemem teletransmisji oraz dostosowaniu makroinstrukcji Georga-3 i Crooka do takiego zestawu, praca koordynatorów sprowadza się do śledzenia przebiegu przetwarzania i ewentualnej ingerencji w przypadku awarii.

Obecnie Stocznia posiada 2 typy systemów użytkowych:
- systemy realizowane tylko na minikomputerze MERA-400,
- systemy realizowane na bazie współpracy MERA-400 - ODRA 1305.

Należy tu zaznaczyć, że te części wszystkich programów użytkowych, które dotyczą rejestracji danych z kontrolą formalną są realizowane na MERZE-400 poprzez uniwersalne programy o nazwie ZAP napisane w Assemblerze i Fortranie.

Program ZAP w Assemblerze jest bardziej uniwersalny, szybszy oraz zajmuje mniej pamięci. Wadą jego jest jednak fakt, że może być konserwowany tylko i wyłącznie przez jego autora.

Program ZAP w Fortranie zajmuje nieco więcej pamięci, jest mniej uniwersalny lecz łatwiejszy w konserwacji.

Obydwa te programy posiadają następujące możliwości:
- umożliwiają szybkie, w dowolnym układzie, wprowadzanie danych z jednoczesną ich kontrolą,
- pozwalają na szybką aktualizację dowolnego pola lub określonej ilości pól,
- umożliwiają szybkie dojście do dowolnej, zarejestrowanej informacji,
- posiadają możliwość dopisywania własnych opcji użytkowych oraz ich poprawianie.

Na bazie wyżej omówionych programów ZAP zarówno w języku Assembler jak i Fortran oraz możliwości systemu CROOK-4 udało się Stoczni w roku bieżącym prawie w 95 % zmienić sposób wprowadzania danych. W zasadzie wszystkie dane wejściowe do systemów użytkowych eksploatowanych w Stoczni są wprowadzane z końcówek monitorowych.

2. SYSTEMY UŻYTKOWE

W Stoczni "Radunia" tylko na MERZE-400, dzięki zainstalowaniu dodatkowej pamięci operacyjnej o pojemności 128 K-słów produkcji Firmy AMEPOL, są realizowane następujące systemy użytkowe:
- Fakturowanie usług - system pozwalający na bieżące /w dosłowym tego słowa znaczeniu/ rozliczanie wszystkich usług wykonywanych przez Stocznię. Możliwość bieżącego fakturowania swoich usług ma dla Stoczni ogromne znaczenie:
  1/ uniemożliwia sztuczne kredytowanie kontrahentów poprzez szybkie ściąganie należności,
  2/ pozwala na dokładne orientowanie się Kierownictwa Stoczni w wysokości uzyskanej "sprzedaży", a więc w możliwościach finansowych zakładu,

3/ pozwala na pełne wywiązywania się z 3-dniowego cyklu fakturowania usług, który to cykl został Stoczni narzucony przez bank jako obowiązujący.

- Planowanie operatywne. W roku 1984 Stocznia, w wyniku reorganizacji zmieniła okres planowania operatywnego kroczącego z okresu dekadowego na tygodniowy. Było to podyktowane przede wszystkim tym, że jak wspomniano wyżej, usługi tego zakładu w ich przeważającej większości opierają się na tzw. "międzyrejsach", a więc remontach bardzo krótkich, nieraz kilkugodzinnych. Są to remonty wykonywane na statkach w czasie za i wyładunku. Armatorzy współpracujący ze Stocznią nie są w stanie na początek roku planowanego określić ilości statków, ich typów i rodzaju remontu, który winien być wykonany. W ten sposób plany mogły być opracowywane na bazie przybliżonych wielkości wykonanych w tym samym okresie roku ubiegłego co wiązało się z dużym przybliżeniem. Wraz z upływem okresu badanego, planiści musieli dokonywać wielu skomplikowanych korekt aby te plany urealnić. Takie korekty mogły następować w okresach nie krótszych jak miesiąc, gdyż zagłębianie się w okres dekadowy czy tygodniowy było niemożliwe ze względu na czas. Trzeba sobie zdawać sprawę z tego, że każda korekta planu narusza wielorakie struktury:
- - asortymentową,
- wydziałową,
- w zakresie mocy przydzielonej armatorom,
- w zakresie rodzaju remontu.

Oprogramowanie na minikomputerze MERA-400 tego skomplikowanego algorytmu pozwala obecnie Stoczni na cotygodniowe emitowanie planu operatywnego dokonując bieżących korekt, a więc urealniając go do bieżących potrzeb armatorów i możliwości zakładu. System ten pozwala obecnie na emitowanie planów operatywnych w różnorakich przekrojach, a więc:
- rok i dowolny kwartał, miesiąc, tydzień w roku dla całej Stoczni,

- rok i dowolny kwartał, miesiąc, tydzień w roku dla wydziału,
- kwartał i dowolny miesiąc, tydzień w kwartale dla Stoczni,
- kwartał i dowolny miesiąc, tydzień w kwartale dla wydziału,
- miesiąc i dowolny tydzień w miesiącu dla Stoczni,
- miesiąc i dowolny tydzień w miesiącu dla wydziału itp.

- Analiza ekonomiczna PULWAR. Jest to system pozwalający na przewidywanie wyniku finansowego oraz innych wskaźników ekonomicznych na zadany okres czasowy. Bardzo pomocny zarówno na etapie budowy planu techniczno-ekonomicznego Stoczni /nie związany ze strukturą usług lub wyrobów/ jak i na etapie przewidywania możliwością osiągnięcia właściwych wyników ekonomicznych w trakcie realizacji roku planowanego.

Omówione wyżej systemy użytkowe, dostępne na własnym komputerze, a więc systemy do uruchomienia w każdej chwili wielokrotnie, konwersacyjne, nie wymagające obecności specjalisty informatyka czy elektronika zostały wdrożone w taki sposób by służyły bezpośrednio planistom, ekonomistom czy też kierownictu zakładu. Wszystkie te cechy zostały podporządkowane wymogom reformy gospodarczej, możliwościom szybkiego orientowania się w sytuacji zakładu oraz celowi, który przyświeca działalności informatycznej Stoczni, a więc - dojściem informatyki do każdego stanowiska pracy.

Inną, dużą grupę zagadnień stanowią systemy użytkowe oprogramowane na bazie połączenia komputerów Odra-1305 i Mera-400 za pomocą łącza stałego.

Do systemów tych należą:

Gospodarka materiałowa - typowy system gospodarki materiałowej z unowocześnionymi jej cechami takimi jak:

- możliwość bieżącego emitowania tabulogramów obrazujących wysokość poniesionych kosztów materiałowych na danej usłudze,
- możliwość bieżącego emitowania tabulogramów obrazujących stany zerowe i tzw. stany alarmowe w magazynach,
- możliwość bieżącego emitowania tabulogramów ukazujących poniesione koszty materiałów z importu w cenach dewizowych.

Jest to bardzo ważna cecha tego systemu, pozwalająca na obciążenia wysokością wkładów dewizowych kontrahentów współpracujących ze Stocznią.

System pracochłonności - stanowiący niejako podstawę do wdrożenia systemu ewidencji osobowej i w ostateczności systemu płac. System ten jest obecnie na etapie zmian rozliczania okresu czasowego z dekadowego na tygodniowy. Stanowi on w zasadzie ewidencję przepracowanego czasu przez pracowników bezpośrednio i pośrednio produkcyjnych w układzie na stanowiska pracy w odniesieniu do praconików, brygady, zespołu mistrzowskiego lub wydziału oraz ukazuje koszty robocizny na wykonywanej lub wykonanej usłudze. Ponadto, system ten dokładnie ewidencjonuje czas stracony w rozbiciu na rodzaje.

Te dwa systemy, wyżej omówione, stanowią podstawę do wyceny danej pracy oraz obciążenia kosztami kontrahenta. System pracochłonności tak jak i system gospodarki materiałowej umożliwia bieżące emitowanie tabulogramów ukazujących koszty robocizny na danej usłudze.

System środków trwałych - system oparty o obowiązujące przepisy gospodarki środkami trwałymi, ukazujący ich ruch, stan, wartości amortyzacji czy umorzeń.

System rozliczania sprzedaży i wartości produkcji globalnej - specyficzny dla branży remontu statków. Dane wejściowe oparte o omówiony wyżej system fakturowania usług, a więc o wartość sprzedaży usług Stoczni oraz o zaliczenia pracochłonności wyrażonej w złotówkach na pracach w toku. Różnica między zaliczoną a sprzedaną robocizną stanowi w zakładzie podstawę do wypłacenia tzw. premii motywacyjnej. Jest to bardzo ważny, z punktu widzenia bodźcowania wydajności pracy, system użytkowy.

System finansowo-księgowy - typowy dla wszystkich zakładów, umożliwiający rozliczanie bilansów finansowo-księgowo-ekkonomicznych.

3. PERSPEKTYWY I PLANY DALSZYCH PRAC

Należy tu zaznaczyć, że Stocznia posiada obecnie pełne i szybkie oprogramowanie umożliwiające wprowadzanie danych wejściowych z monitorów ekranowych do minikomputera MERA-400 oraz oprogramowanie umożliwiające przesyłanie danych z MERY-400 do ODRY-1305 na bazie łącza stałego z kontrolą tej transmisji.

Oprogramowanie to zostało wykonane przez specjalistów Politechniki Gdańskiej, przy współudziale pracowników Stoczni. Zawiera ono w sobie pełną kontrolę formalną oraz część kontroli logicznej uwzględniającą specyfikę dokumentu jak i systemu żytkowego.

Informacje te dotyczą wszystkich omówionych powyżej systemów.

Możliwości techniczne sprzętu będącego w posiadaniu Stoczni obecnie zaczynają się bardzo kurczyć przy dynamicznych pracach w zakresie oprogramowania użytkowego.

Dodatkowe trudności nastręcza fakt całkowitego braku serwisu technicznego minikomputera MERA-400 przy dużej awaryjności tego sprzętu. Dlatego też przyszły rok Stocznia pragnie poświęcić na rozbudowę i unowocześnienie tego sprzętu w zakresie:

- uzyskania łącza stałego /od roku otrzymujemy decyzje negatywne/ z wydziałami gdyńskimi,
- powiększenia pojemności pamięci zewnętrznych przez dostawienie dysków o większej pojemności,
- uzyskanie i zainstalowanie procesora zewnętrznego /MULTIX/ do obsługi znakowych urządzeń peryferyjnych takich jak: łącza synchroniczne, monitorów itp.,
- uzyskania i zainstalowania procesora zewnętrznego do umożliwienia szybkiego dostępu do pamięciowych modułów zewnętrznych /PLIX/,
- modyfikacji systemu operacyjnego CROOK-4 do współpracy z procesorami wymienionymi wyżej CROOX/.

Wykonanie tych zadań umożliwi Stoczni podjęcie dalszych prac projektowo-programowych w zakresie budowy jednolitego użytkowego systemu informacyjnego, który pozwoli na informatyzację w przyszłości całości prac planistycznych i rozliczeniowych oraz umożliwi sterowanie je działalnością z uwzględnieniem optymalizacji.

mgr inż. Jan Wierzbicki
mgr inż. Maria Perek
Huta Szkła Okiennego
"Szczakowa"

SYSTEM INFORMOWANIA KIEROWNICTWA W HUCIE SZKŁA OKIENNEGO "SZCZAKOWA"

Na wstępie krótka charakterystyka Ośrodka i Zakładu: Aktualnie Ośrodek liczy 12 osób, z tego 8 osób w sekcjach projektowo - programowej i eksploatacji. Huta Szkła Okiennego "Szczakowa" zatrudnia średnio 1200 osób. Ośrodek istnieje od 4 lat. Prowadzimy eksploatację na jednym systemie Mera 400 w następującej konfiguracji: 64 ksłów pamięci operacyjnej, 2 pamięci dyskowe, 2 drukarki, 7 monitorów ekranowych. W najbliższym czasie zainstalujemy drugi system Mera 400. Pracujemy pod systemem 2 zadaniowym SOM 3 ze spulerem drukarki.
Cała oprogramowanie użytkowe wykonywane jest we własnym zakresie w oparciu o Bazę Danych "Vitrin BD-83".

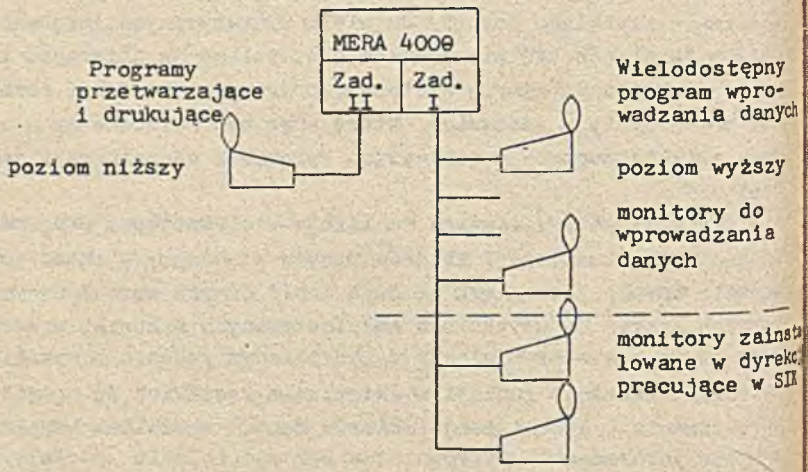
Ostatnio w naszym Ośrodku powstał System Informowania Kierownictwa w skrócie "SIK".
Został on opracowany w celu ułatwienia pracy dyrekcji przez umożliwienie szybkiego dostępu do wielu syntetycznych informacji. Zasada działania ogólnie rzecz biorąc polega na zbieraniu informacji z eksplo atowanych systemów, przetwarzaniu ich i redagowaniu przejrzystych zestawień, które mogą być oglądane na monitorach zainstalowanych w gabinetach dyrektora naczelnego i jego zastępców.

Wyświetlanie zestawień realizuje wielodostępny program wprowadzania i aktualizacji zbiorów danych wchodzący w skład oprogramowania BD-83. Ze względu na dużą ilość danych wprowadzanych dla potrzeb system ów użytkowych eksploatowanych w hucie, program ten jest używany w sposób ciągły w wydzielonym zadaniu o wysokim priorytecie. Włączanie funkcji wyświetlania zestawień do programu wprowadzania i aktualizacji zbiorów danych umożliwia eksploatację systemu informowania kierownictwa bez wydzielania specjalnych zadań systemu o peracyjnego dla obsługi użytkowników tego systemu.

Użytkownicy mogą żądać informacji od systemu w dowolnym momencie. Wysoki priorytet zadania, w którym wykonywany jest wielodostępny program wprowadzania i aktualizacji zbiorów danych, zapewnia krótki czas oczekiwania na informację.
Równolegle z tym zadaniem w drugim zadaniu o niższym priorytecie i w spulerze wykonywane są programy przetwarzania danych i drukowania, co zapewnia pełne wykorzystanie zasobów komputera w konkretnej konfiguracji.
Zestawienia systemu informowania kierownictwa mają postać pełnych ekranów i są przechowywane w pamięci dyskowej jako rekordy zbiorów SIK /BD-83/.
Struktura rekordu jest następująca:

- identyfikator 2 lub 4 znakowy
- raport czyli dana o długości 1760 znaków, która przy wyświetlaniu zajmuje dokładnie cały ekran monitora.

Wprowadzanie nowych zestawień nie nastręcza żadnych trudności ponieważ jest równoznaczne z wprowadzeniem nowych rekordów do zbiorów. Oprogramowanie BD-83 pozwala ograniczyć dostęp do zbiorów z poszczególnych monitorów a także wg haseł. Umożliwia to odseparowanie systemu informowania kierownictwa od innych systemów użytkowych.



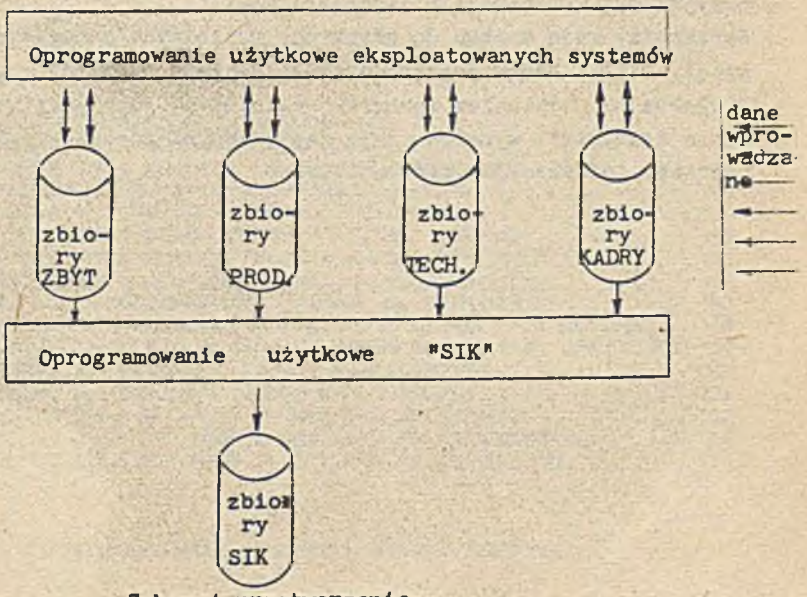
Realizacja komputerowa systemu

Aktualnie w naszym Ośrodku są eksploatowane następujące ważniejsze podsystemy:

- ZBYT - fakturowanie, sprawozdawczość, analizy, księgowanie, drukowanie żądań zapłaty na inkaso.
- PRODUKCJA - rozliczanie produkcji w poszczególnych fazach procesu produkcyjnego, analizy, statystyka, obliczanie akordów.
- TECHNOLOGIA - prowadzenie ewidencji analiz technologicznych szkła i surowców, obliczanie korekt dla procesu technologicznego, obliczanie parametrów fizykochemicznych szkła.
- Kadry - ewidencja danych osobowych pracowników, statystyka kadrowa.

Dzięki Bazie Danych "Vitrin" posiadamy jednolite struktury zbiorów we wszystkich podsystemach.

Dane do SIK otrzymywane są w trakcie przetwarzania aktualnie eksploatowanych podsystemów w sposób automatyczny bez ingerencji operatora systemu, dzięki czemu zestawienia SIK są aktualizowane na bieżąco. Odbywa się to przy pomocy odpowiednich procedur JOB'a.



Schemat przetwarzania

Oprogramowanie użytkowe SIK /napisane w języku FORTRAN/ zbierające automatycznie dane z podsystemów i tworzące raporty-zestawienia do SIK składa się z kilkunastu programów o łącznej długości panad 3000 linii programowych.

Zbiór zestawień SIK uzupełniony jest o tzw. zbiór archiwalny raportów, gdzie przechowuje się najważniejsze zestawienia za poszczególne miesiace.

Jak już wspomniano system SIK nie jest systemem zamkniętym. W trakcie wdrażania kolejnych podsystemów np. Gospodarka Materiałowa, Podsystem Finansowy itp. można rozbudować zbiory raportów. Obecnie w zbiorze SIK jest około 50 zestawień, w najbliższym czasie udostępnimy następne.

Użytkowanie systemu jest niezwykle proste. Aby uzyskać potrzebne zestawienie użytkownik podaje jego numer. Numery zestawień są podane w spisach z estawień które również mogą być wyświetlane na ekranie. Zestawienie z numerem 2 zawiera spis grup tematycznych w którym podane są numery zestawień zawierające spisy zestawień z danej grupy.

System Informowania Kierownictwa w HSO "Szczakowa" bardzo dobrze przyjął się w Dyrekcji przedsiębiorstwa.

Dyrektorzy mają dostęp do szybkich, rzetelnych i bogatych informacji, dzięki czemu mogą podejmować decyzje bardziej operatywnie. Ciekawym udogodnieniem oprogramowania BD-83 jest możliwość drukowania statystyki wykorzystania zestawień co pozwala lepiej poznać potrzeby informacyjne kierownictwa.

```
2:R     :NR RAPORTU

  W Y K A Z  D Z I A L O W

                                NR RAPORTU
   I. ZBYT . . . . . . . . . . . . . . ?3

  II. PRODUKCJA. . . . . . . . . . . . ?4

 III. TECHNOLOGIA. . . . . . . . . . . ?5

  IV. KADRY. . . . . . . . . . . . . . ?6



FUNKCJA? /+DOPISZ,$USUN,=POPRAW,?WYSWIETL,$KONIEC,!WIADOMOSC/


3:R     :NR RAPORTU
                          *** Z B Y T ***
 W Y K A Z   R A P O R T O W
                                                          NR RAPORTU
  1. SPRZEDAZ NARASTAJACO WG. ASORTYMENTOW. . . . . . . . . . . . . ?7
  2. SPRZEDAZ SZKLA OKIENNEGO NARASTAJACO ILOSCIOWO . . . . . . . . ?8
  3. SPRZEDAZ SZKLA OKIENNEGO NARASTAJACO WARTOSCIOWO . . . . . . . ?9
  4. SPRZEDAZ DLA KOOPERACJI NARASTAJACO ILOSCIOWO. . . . . . . . . ?10
  5. SPRZEDAZ DLA STOLBUDOW NARASTAJACO ILOSCIOWO . . . . . . . . . ?11
  6. SPRZEDAZ SZKLA OKIENNEGO NARASTAJACO WG. GATUNKOW. . . . . . . ?12
  7. SPRZEDAZ SZKLA OKIENNEGO NARASTAJACO WG. TYPU ROZKROJU . . . . . ?13
  8. SPRZEDAZ SZKLA OKIENNEGO NARASTAJACO WG. POWIERZCHNI . . . . . ?14
  9. SPRZEDAZ ZAFAKTUROWANA OD POCZ.M-CA WG. ASORTYMENTOW . . . . . ?17
 10. SPRZEDAZ ZAFAKTUROWANA SZKLA OKIENNEGO OD POCZ.M-CA ILOSCIOWO. . ?18
 11. SPRZEDAZ ZAFAKTUROWANA SZKLA OKIENNEGO OD POCZ.M-CA WARTOSCIOWO. ?19
 12. SPRZEDAZ DLA KOOPERACJI OD POCZ,M-CA ILOSCIOWO . . . . . . . . ?20
 13. SPRZEDAZ DLA STOLBUDOW OD POCZ.M-CA ILOSCIOWO. . . . . . . . . ?21
 14. SPRZEDAZ SZKLA OKIENNEGO OD POCZ.M-CA WG. GARUNKOW . . . . . . ?22
 15. SPRZEDAZ SZKLA OKIENNEGO OD POCZ.M-CA WG. TYPU ROZKROJU. . . . . ?23
 16. SPRZEDAZ SZKLA OKIENNEGO OD POCZ.M-CA WG. POWIERZCHNI. . . . . ?24


FUNKCJA? /+DOPISZ,$USUN,=POPRAW,?WYSWIETL,$KONIEC,!WIADOMOSC/
```

RAPORT NR 22 ZA OKRES 01-28.11.84

SPRZEDAZ SZKLA OKIENNEGO OD POCZ.M-CA WG GATUNKOW

KIERUNEK	GAT.I (%)	GAT.II (%)	GATIII(%)	RAZEM %	M2EF
RYNEK	65.44	30.97	3.59	100.00	3280
KOOPERACJA	100.00	0.00	0.00	100.00	11500
CENTRALE	51.20	48.80	0.00	100.00	8169
STOLBUDY	98.80	1.20	0.00	100.00	8239
POZARYNEK	47.07	52.93	0.00	100.00	3966
EXPORT FW	0.00	100.00	0.00	100.00	4475
RAZEM	70.24	29.46	0.30	100.00	39631
OGOLEM(+NOTY+D.S)					39631

FUNKCJA? /+DOPISZ,$USUN,=POPRAW,?WYSWIETL,$KONIEC,!WIADOMOSC/

RAPORT NR 50

SKLAD TEORETYCZNY SZKLA STAN NA DZIEN 28.11.1984

NAWAZKI

PIASEK	DOLOMIT	WAPIEN	SODA	TL.GLINU	SULFAT	WEGIEL D.	SUMA
1490.000	358.000	88.000	518.000	20.800	24.000	1.400	2500.200

UDZIAL PROCENTOWY TLENKOW / % /

NAWAZKI	SIO2	AL203	CAO	MGO	FE203	NA20	SUMA
17.903	72.398	1.121	7.782	3.635	0.097	14.968	100.000

D = 2.47716
NA/CA = 1.92348
AL/MG = 0.30856
RO =11.41616
RO + R2O =26.38391

JEDNORODNOSC:

SODA = 20.718
SODA+SULFAT = 21.678
CZESCI NIEROZP. W HCL = 60.734
CZESCI ROZP. W HCL = 17.588

FUNKCJA? /+DOPISZ,$USUN,=POPRAW,?WYSWIETL,$KONIEC,!WIADOMOSC/

mgr inż. Jan Ustaszewski
Instytut Dróg i Mostów
Politechnika Warszawska

R E F E R A T

Temat: "SOFTWARE CREATION - środowisko softwareowe do tworzenia i eksploatowania na EMC Mera 400 oprogramowania inżynierskiego".

Wstęp.

SC /Software Creation/ jest oprogramowaniem systemowym organizującym pracę w języku FORTRAN na EMC Mera 400. SC działa pod kontrolą systemu operacyjnego SOM 3. SC jest dostosowany do standardowej konfiguracji: pamięć operacyjna /użytkowa/ 20 K słów, monitor /KSR/, drukarka, dysk, SOM 3 wersja EMC. Zadaniem SC jest sterowanie pracą procesorów systemowych SOM'u /FOR, MAC, EDI, edytor tekstowy/, obsługa pamięci dyskowej /system zbiorów/ oraz startowanie programów użytkowych.

Założenia metodologiczne.

SC "odsoparowuje" użytkownika od takich pojęć jak: procesory systemowe, sekcje, opcje /SOM'u 3/. Jednostkami informacji, którymi manipuluje użytkownik, są zbiory źródłowe. Dostęp do zbioru uzyskuje się poprzez podanie nazwy zbioru. Zmiany w zbiorze dokonywane są za pomocą edytora tekstowego.

W dialogu użytkownika z komputerem wyróżniamy:

- dyrektywy SC,
- pracę w edytorze tekstowym,
- ewentualną konwersację z oprogramowaniem użytkowym /rys. 1/.

Oprogramowanie użytkowe oznacza pewien zbiór programów, które mogą się wzajemnie wywoływać. Programy, zbiory danych oraz obszar roboczy do przekazywania danych pomiędzy programami tworzą system użytkownika /rys. 2/.

System użytkownika posiada swoją strukturę statyczną /rys. 3/, która determinuje sposób pracy w systemie SC.

Praca w systemie SC.

Do startowania gotowych programów użytkowych służy dyrektywa E /Execute/. Jeżeli użytkownik , oprócz nazwy programu, poda nazwę zbioru danych, to program może czytać ten zbiór z bazy danych za pomocą standardowej /SC/ procedury READAT. Parametrem tej procedury jest tablica mogąca pomieścić do 80 znaków ISO-7. Każde wywołanie procedury READAT powoduje wypełnienie tej tablicy kolejnym rekordem zbioru danych.
Program może wywołać inny program za pomocą standardowej /SC/ procedury EXE, której parametrem jest 6-cio znakowa nazwa programu.

Programy użytkowe tworzone są za pomocą dyrektywy P /Program/. Konsolidacja /linkowanie/ programu odbywa się na podstawie opisu struktury programu. Opis struktury programu jest to zbiór źródłowy, który określa z jakich modułów /podprogramów/ składa się dany program./rys. 4/.

Do modyfikacji modułów źródłowych służy dyrektywa M /Modify/. Powoduje ona zainicjowanie pracy edytora tekstowego. Po wprowadzeniu poprawek w zbiorze źródłowym i wyjściu z edytora, SC uruchamia automatycznie translację i kompilację poprawionego modułu, poczym zapisuje nową binarną wersję modułu do bazy danych./rys. 5/.

Tworzenie i poprawianie zbiorów danych, jak również poprawki nie wymagające kompilacji modułów /np. komentarze/, umożliwia dyrektywa ED /EDit/, która inicjuje edytor tekstowy do pracy na określonym zbiorze.

Dalsze możliwości systemu SC.

Oprócz FORTRAN'u, możliwa jest również praca w językach MACROASSEMBLER i BASIC /wersja bezdyskowa/. Wybór języka odbywa się poprzez dyrektywę LAN /LANguage/. Jeżeli jest ustawiony tryb pracy w języku MACROASSEMBLER'a, to dyrektywa M /Modify/ po skończonej edycji modułu uruchamia procesor MAC z pominięciem procesora FOR. Ustawienie języka BASIC powoduje pobranie zbioru, którego nazwa jest parametrem dyrektywy M /Modify/ z bazy danych na sekcję roboczą i następnie automatyczne wystartowanie procesora BAS. Reżim pracy procesora BAS zostaje ustawiony w ten sposób, że dyrektywa

BASIC'u LOA=1 powoduje załadowanie programu z dysku zamiast z czytnika taśmowej papierowej, a dyrektywa LIS=2 - wylistowanie programu na dysk zamiast na perforator. Po zakończeniu pracy w BASIC'u, wylistowany program zapamiętywany jest automatycznie w bazie danych. Program ten można poprawiać edytorem tekstowym tak samo, jak każdy inny zbiór źródłowy.

Dyrektywy specjalne.

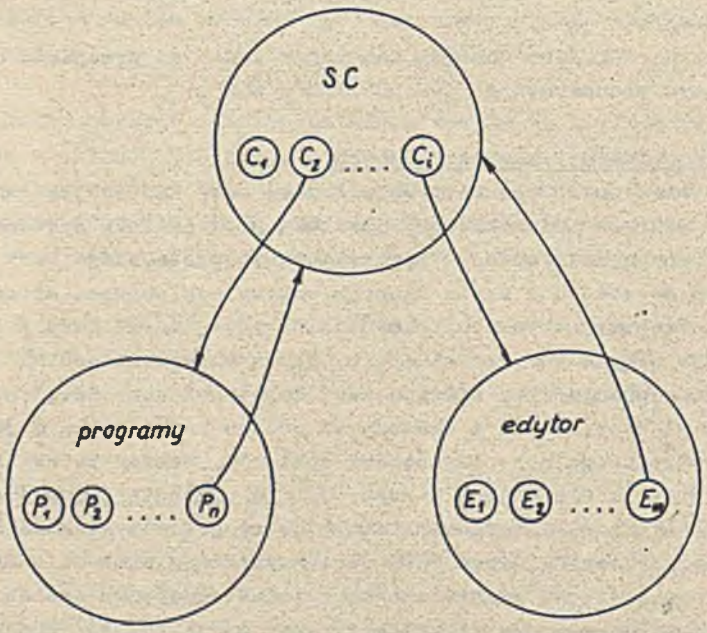
Istnieje kilka dyrektyw niedostępnych dla zwykłego użytkownika. Przeznaczone są one dla programisty systemowego nadzorującego pracę na danym komputerze.
Dyrektywa LOG /LOGin/ służy do ustawienia pracy dla określonego użytkownika. Zbiory każdego użytkownika są niedostępne dla innych użytkowników, mimo iż znajdują się w tej samej bazie danych /sekcji dyskowej/.
Dyrektywa JCL /Job Control Language/ służy do przejścia na nasłuch dyrektyw standardowego JOB CONTROL'a SOM'u 3.

Struktura wewnętrzna bazy danych.

Aczkolwiek opis struktury wewnętrznej bazy danych jest ukryty przed użytkownikiem, podany jest dla programistów systemowych pracujących nad rozwojem oprogramowania systemowego Mery 400.
Dostęp do zbioru w bazie danych odbywa się poprzez czteropoziomową strukturę strukturę skorowidzów /rys. 6/. Zbiory i skorowidze zajmują ciągłe obszary na dysku. Zapis nowego zbioru na dysk wymaga przeszukania tzw. bitowej mapy zajętości sektorów, aby znaleźć wolne miejsce. Zapis i skasowanie zbioru związane są z jednoczesną modyfikacją mapy zajętości sektorów. Sektor zerowy służy jako słownik dla LOADER'a /rys. 7/. Za pomocą trzyznakowych identyfikatorów startowane są tylko programy systemowe SC / w przypadku, gdy rezydują one na kasecie wymiennej a nie w LMD/.
Programy użytkowe startowane są w nieco odmienny sposób. Nazwa programu podana jako parametr dyrektywy E /Execute/ znajdowana jest w skorowidzu skonsolidowanych programów i odpowiadający adres programu na dysku wpisywany jest w standardową /SC/ pozycję słownika oznaczoną identyfikatorem EXE. Następnie wywoływany jest ekstrakod CHAIN dla identyfikatora EXE. W rezultacie program użytkownika rozpoczyna swoje działanie. Metoda ta umożliwia stosowanie pełnych 6-cio znakowych nazw dla programów.

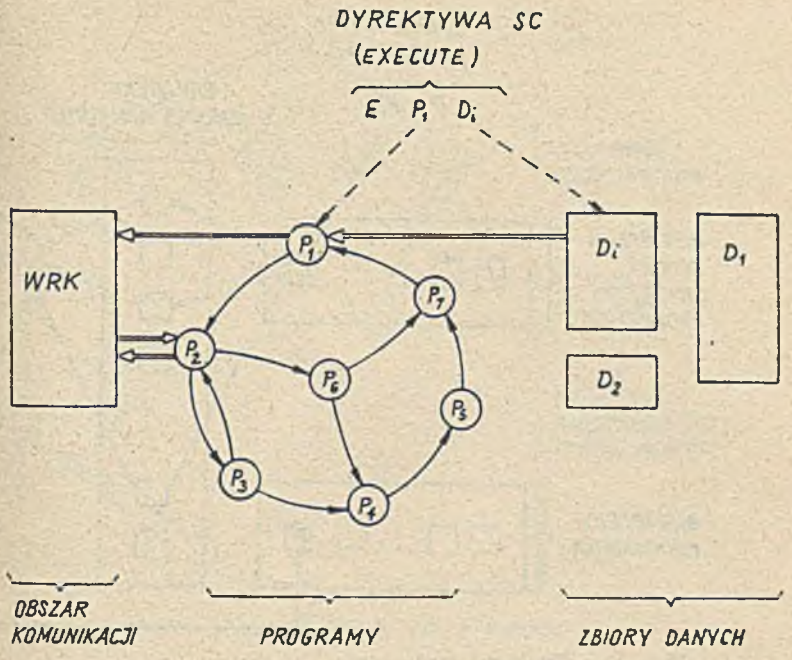
Zakończenie.

System SC eksploatowany jest w Instytucie Dróg i Mostów Politechniki Warszawskiej już ponad rok. Praktyka wykazała, że jest on skutecznym narzędziem do tworzenia nawet bardzo rozbudowanego oprogramowania /kilkaset modułów/. Użytkownicy pracowali samodzielnie z komputerem już po kilkunastominutowym instruktażu.
SC umożliwia ominięcie "bariery" jaką stawia przed początkującym użytkownikiem Mery 400 SOM 3 - obszerna dokumentacja, dosyć złożony i żmudny sposób uruchamiania procesorów systemowych, brak systemu zbiorów. SC jest przeznaczony dla szerokiego kręgu zwykłych użytkowników chcących policzyć coś w FORTRAN'ie i nie mających czasu na studiowanie zawiłości systemu operacyjnego.

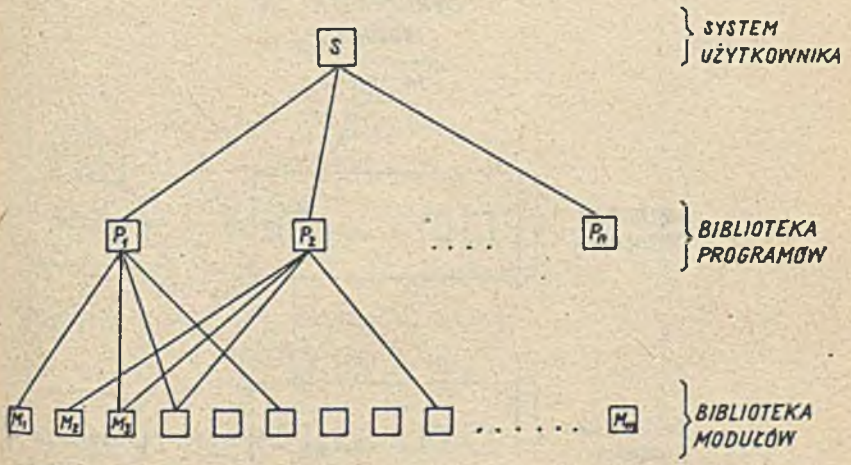


C_i - dyrektywa SC
E_i - dyrektywa edytora tekstowego
P_i - dyrektywa (jeżeli występuje) programu użytkownika

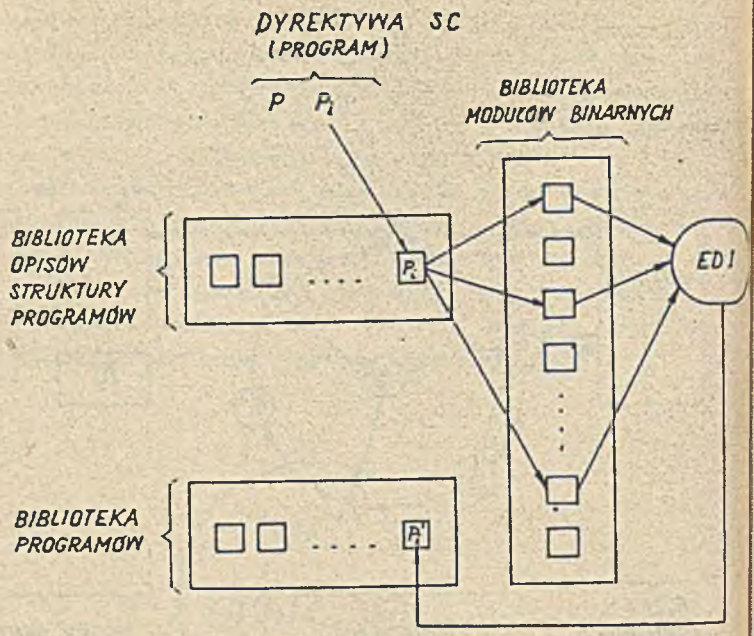
Rys. 1. Elementy dialogu użytkownika z komputerem



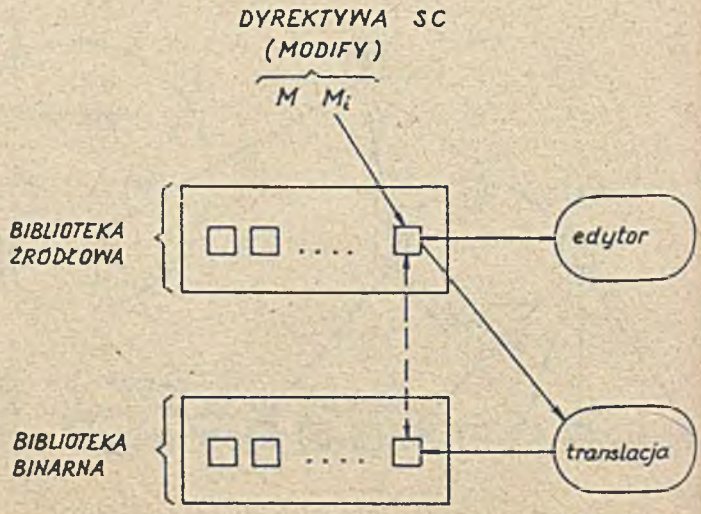
Rys. 2. Przekazywanie sterowania pomiędzy programami i przepływ danych w systemie użytkownika



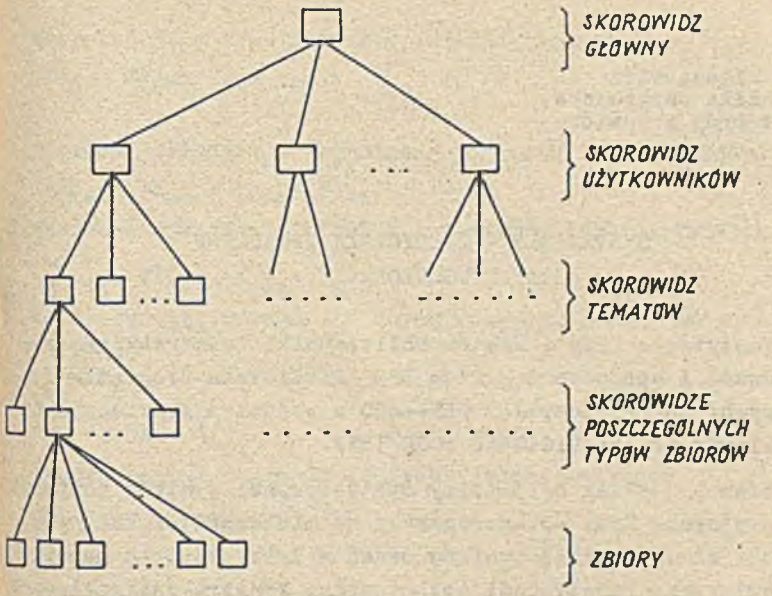
Rys. 3. Schemat statyczny systemu użytkownika



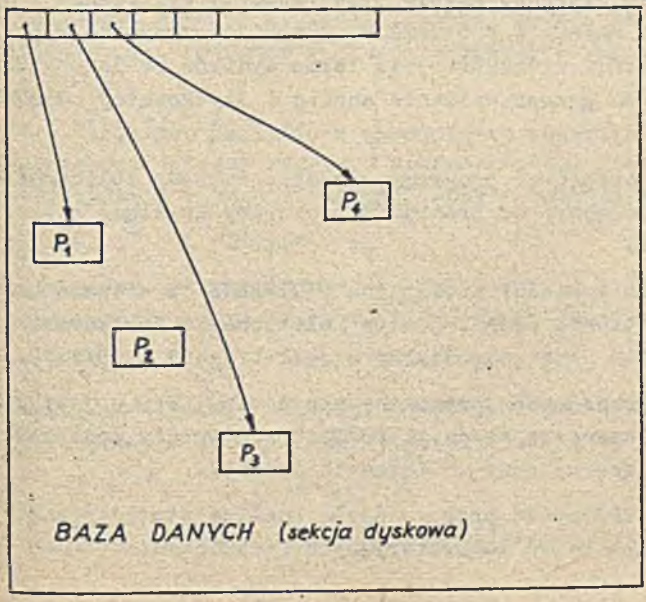
Rys. 4. Tworzenie programu



Rys. 5. Modyfikacja modułu



Rys. 6. Struktura skorowidzów w bazie danych



Rys. 7. Organizacja słownika LOADER'a

Barbara Piłasiewicz
Politechnika Warszawska
Instytut Dróg i Mostów

SYSTEM BPS - BIBLIOTEKA PROGRAMÓW STATYSTYCZNYCH

W Instytucie Dróg i Mostów Politechniki Warszawskiego zaprojektowano i opracowano system BPS /Biblioteka Programów Statystycznych/ na minikomputer MERA-400 z wyjściem graficznym na autokreślarkę czeską DIGIGRAF 1008/1612.

System BPS służy do analizy statystycznej wyników pomiarów laboratoryjnych. Brak oprogramowania na minikomputer MERA-400 do obróbki statystycznej wyników badań w laboratoriach materiałów budowlanych, konstrukcji żelbetowych, konstrukcji stalowych i mechaniki gruntów był bodźcem do opracowania tego systemu.

Pierwsze programy systemu BPS opracowane zostały w języku BASIC, ponieważ należało szybko przeprowadzić obliczenia i wprowadzano ciągłe zmiany w programach. Wyniki otrzymywano w postaci dużej ilości stron wydruków. Taka forma wyników obliczeń okazała się uciążliwa do przeprowadzenia analiz i użytkownicy odczuwali brak rysunków wykresów otrzymywany z obliczeń funkcji.

Należało opracować programy rysujące wykresy obliczanych zależności i dołączyć do programów procedury kreślące z biblioteki DIGI-BENS.

Biblioteka procedur kreślących DIGI-BENS opracowana została również w Instytucie Dróg i Mostów Politechniki Warszawskiej w języku FORTRAN na autokreślarkę czeskł DIGIGAF 1008/1612.

Całe dotychczasowe oprogramowanie systemu BPS w języku BASIC zostało przetłumaczone na język FORTRAN i powstały dodatkowo nowe programy kreślące wykresy obliczanych funkcji.

Systemem BPS można przeprowadzić analizę statystyczną otrzymanych wyników z badań laboratoryjnych i obliczyć:

- wartości średnie z pomiarów laboratoryjnych

$x^1_{śr}, x^2_{śr}, x_{śr}, y_{śr} = \sum_{i=1}^{n} y_i / n$

- wyszukać wartości maksymalne z pomiarów laboratoryjnych

$x^1_{max}, x^2_{max}, x_{max}, y_{max}$

- wyszukać wartości minimalne z pomiarów laboratoryjnych

$x^1_{min}, x^2_{min}, x_{min}, y_{min}$

- współczynniki równań liniowych regresji jednej zmiennej i obliczyć

$y_x = a\,x + b$

$x_y = c\,y + d$

- współczynniki regresji liniowej jednej zmiennej

$\beta_{yx} = (\bar{y}_x - y)/(x - \bar{x})$

$\beta_{xy} = (\bar{x}_y - x)/(y - \bar{y})$

- współczynniki korelacji liniowej jednej zmiennej

$r_{xy} = \pm\sqrt{\beta_{xy}\ \ \beta_{yx}}$

- i narysować wykresy regresji liniowej jednej zmiennej
- współczynniki równania regresji krzywoliniowej parabolicznej i obliczyć

$y_x = a0 + a1\,x + a2\,x^2$

- i narysować wykresy regresji krzywoliniowej parabolicznej
- współczynniki równania regresji krzywoliniowej w postaci

$y_x = a\ x^b$ i obliczać y_x

- i narysować wykresy regresji krzywoliniowej tego równania
- współczynniki regresji krzywoliniowej wykładniczej i obliczać

$y_x = b\,a^x$

- i narysować wykresy regresji krzywoliniowej wykładniczej
- współczynniki równania regresji krzywoliniowej w postaci

$y_x = a\,e^{b\,x}$ i obliczać y_x

- i narysować wykresy regresji krzywoliniowej tego równania

- współczynniki równania regresji krzywoliniowej hiperbolicznej i obliczyć

$$y_x = (a / x) + b$$

- i narysować wykresy regresji krzywoliniowej hiperbolicznej
- stosunek korelacyjny regresji krzywoliniowej jednej zmiennej

$$\eta_{yx} = \left(\sum_{i=1}^{r} n_{xi} (y_i - y)^2 \right) / \left(\sum_{j=1}^{r} n_{yj} (y_j - y)^2 \right)$$

- współczynniki równania liniowego regresji wielorakiej /wielokrotnej/

$$y_x = a\ x1 + b\ x2 + c$$

- łączny współczynnik regresji wielorakiej /wielokrotnej/

$$R = \sqrt{(r_{zx}^2 + r_{zy}^2 - 2\ r_{zx}\ r_{zy}\ r_{xy}) / (1 - r_{xy}^2)}$$

- i narysować wykresy regresji wielorakiej /wielokrotnej/
- odchylenie standardowe

$$\sigma = \sqrt{\sum_{i=1}^{n} (y - y_{śr})^2 / (n - a)}$$

- średnie odchylenie względne
- współczynnik zmienności

$$\nu = (\sigma / y_{śr})\ 100\ \%$$

- i wydrukować tabele różnych zależności np. właściwości betonu
- i wydrukować tablice korelacyjne
- współczynniki równania nieliniowego regresji wielorakiej /wielokrotnej/

$$y_{x_1 x_2} = a\ x1 + b\ x2 + 2c\ x\ 1\ x\ 2 + 2\ d\ x1 + 2\ e\ x\ 2 + f$$

W dalszym ciągu system BPS będzie rozwijany w zależności od potrzeb użytkowników systemu.

Wprowadzenie danych z pomiarów do systemu BPS może odbywać się w trybie konwersacyjnym, z taśmy papierowej lub dysku. Taśmę papierową można przygotować na teletape. Dane na dysk mo

zapisać używając procesorów systemowych UPD, SED lub EDM.

Dane podaje się bez uprzedniego porządkowania ich, w prosty sposób w postaci:

- przy zależności jednej zmiennej
  - n - liczba pomiarów
  - x_i, y_i - pary wartości z pomiarów
  - $i = 1,2,\ldots, n$
- przy zależności dwóch zmiennych
  - n - liczba pomiarów
  - x_{1i}, x_{2i}, y_i - wartości z pomiarów
  - $i = 1,2,\ldots, n$

Przykłady zastosowania systemu BPS

Systemem BPS przeprowadzono analizę statystyczną właściwości betonów z kruszyw węglanowego i łupkoporowatego.
Dla kruszywa łupkoporowatego dodatkowo liczono:

- gęstość pozorną kruszywa ρ_k ze wzoru

$$\rho_B = (c + k + p + w) / (c / \rho_c + k / \rho_k + p / \rho_p + w)$$

- wskaźnik cementowo-wodny c / w
- wskaźnik k / z

Dane i obliczenia przedstawiono w tabeli nr 1.

Wytrzymałość betonu piaskowego obliczona została w postaci zależności jednej zmiennej jako krzywe regresji pierwszego i drugiego stopnia, wyniki obliczeń przedstawiono w tabelach Nr 2, Nr 3 i Nr 4.

Wyniki obliczeń krzywych regresji parabolicznej w postaci graficznej przedstawiono dla:

- cementu C
- gęstości pozornej betonu ρ_B
- gęstości pozornej kruszywa ρ_k
- wskaźnika cementowo-wodnego c/w
- wskaźników k/c, k/z, k/p, (k + p) / c oraz p / c

na rysunku Nr 1.

Przykład obliczeń regresji wielorakiej /wielokrotnej/ na zależność liniową dwóch zmiennych x_1 i x_2 podano w tabeli Nr 5, a postać graficzną wyników obliczeń regresji wielorakiej przedstawiono na rysunku Nr 2.

Systememe BPS można szybko przeprowadzić analizę statystyczną otrzymanych wyników z pomiarów laboratoryjnych poprzez ocenę wykresów szukanych zależności, zamiast żmudnej i długiej analizy wielu stron wydruków.

Wykaz tablic i rysunków

Tabl. Nr 1 - Fragment wydruku tabeli dla kruszywa łupkoporowatego.
Tabl. Nr 2,3,4 - Fragment wydruku równań regresji krzywoliniowej jednej zmiennej.
Rys. Nr 1 - Wykres wytrzymałości betonu w zależności od: C, ϱ_B, ϱ_k, c / w, k / c, k / z, k / p, (k + p) / c, p / c.
Tabl. Nr 5 - Fragment wydruku równań regresji liniowej dwóch zmiennych.
Rys. Nr 2 - Wykres zmiennej y w zależności od x_1 i x_2.

Wykaz piśmiennictwa

1. J. Greń: Statystyka matematyczna modele i zadania, PWN, Warszawa 1982 r.
2. L. Brunarski, L. Runkiewicz: Podstawy i przykłady stosowania metod nieniszczących w badaniach konstrukcji z betonu, ITB, Warszawa 1975 r.
3. M.Cieciura, J. Pińkowski: Teoria decyzji statystycznych i statystyka matematyczna, WAT, Warszawa 1982 r.
4. Praca zbiorowa: Zastosowanie rachunku prawdopodobieństwa w geotechnice, Ossolineum, Warszawa 1982 r.
5. H.E. Kryński: Matematyka dla ekonomistów, PWN, Warszawa 1971 r.
6. C.Radhakrishna Rao: Modele liniowe statystyki matematycznej, PWN, Warszawa 1982 r.
7. Praca zbiorowa: FORTRAN IV-IDIM Programowanie systemu cyfrowego ODRA 1304, Politechnika Warszawska, Warszawa 1976 r.

TABELA NR 4

```
 SSSSS    YY    YY    SSSSS    TTTTTTTT   EEEEEEE  MM       MM
SS   SS   YY  YY    SS   SS      TT      EE       MMM     MMM
SS         YYYY     SS           TT      EE       MM M   M MM
 SSSSS      YY       SSSSS       TT      EEEEEE   MM  M M  MM
     SS     YY           SS      TT      EE       MM   M   MM
SS   SS     YY     SS    SS      TT      EE       MM       MM
 SSSSS      YY      SSSSS       TT      EEEEEEE  MM       MM
```

```
BBBBBB    PPPPPP     SSSSS
BB   BB   PP   PP   SS   SS
BB   BB   PP   PP   SS
BBBBBB    PPPPPP     SSSSS
BB   BB   PP             SS
BB   BB   PP        SS   SS
BBBBBB    PP         SSSSS
```

BIBLIOTEKA PROGRAMOW STATYSTYCZNYCH

LABORATORIUM INFORMATYKI I INFOGRAFIKI INSTYTUT DROG I MOSTOW
POLITECHNIKA WARSZAWSKA AL. ARMII LUDOWEJ 16 00-637 WARSZAWA

WLASCIWOSCI BETONU WYKONANEGO Z KRUSZYWA LUPKOPOROWATEGO

C	P1	P2	K1	K2	W	@B	C/W	K/Z	@K	WB
225	164.0	100	328.0	328	180	1285	1.25	.98	966	99.0
330	297.5	90	297.5	255	210	1443	1.57	.60	981	170.5
350	482.0	80	241.0	241	240	1572	1.46	.42	1016	259.0
450	485.0	80	242.0	242	290	1717	1.55	.37	1230	309.3
225	158.0	100	316.0	316	171	1215	1.32	.97	881	103.0

TABELA NR 2

WYDRUK ROWNAN REGRESJI I STOPNIA

Y(X) = .31159X + 11.68762

WYTRZYMALOSC BETONU OBLICZONA WG WZORU NA REGRESJE I STOPNIA

NR	X(I)	Y(I)	Y(I) OBLICZO
1	21.80000	20.40000	18.48018
2	24.00000	18.60000	19.16567
3	21.80000	18.50000	18.48018
4	22.00000	19.00000	18.54250
5	21.30000	17.70000	18.32439
6	19.00000	16.30000	17.60775
7	21.90000	17.10000	18.51134
8	24.30000	18.70000	19.25915
9	25.30000	19.70000	19.57073
10	23.80000	16.50000	19.10336
11	24.50000	19.20000	19.32147
12	24.30000	18.90000	19.25915
13	22.60000	18.70000	18.72945
14	23.90000	18.90000	19.13451
15	24.30000	20.10000	19.25915
16	21.70000	18.90000	18.44903
17	21.60000	19.80000	18.41787
18	21.20000	20.00000	18.29323
19	23.90000	19.80000	19.13451
20	24.00000	19.90000	19.16567
21	23.60000	19.30000	19.04104
22	24.30000	18.90000	19.25915
23	25.30000	20.40000	19.57073
24	25.50000	18.80000	19.63305
25	25.80000	20.80000	19.72653
26	25.30000	20.40000	19.57073
27	24.90000	17.60000	19.44610
28	24.80000	18.70000	19.41494
29	26.20000	21.40000	19.85116
30	25.00000	18.20000	19.47726

X(Y) = .63153Y + 11.57231

WSPOLCZYNNIK REGRESJI Y WZGLEDEM X @YX= .311585
WSPOLCZYNNIK REGRESJI X WZGLEDEM Y @XY= .631531

WSPOLCZYNNIK KORELACJI LINIOWEJ R=SQR(@(YX)*@(XY)) = .4435[illegible]

ODCHYLENIE STANDARDOWE WYTRZYMALOSCI BETONU S(R) = 1.202[illegible]

V(R) = 6.31699
WSKAZNIK ZMIENNOSCI LICZB ODBICIA V(L) = 7.25664 E-02

S(K) = 1.07794

SREDNIE KWADRATOWE ODCHYLENIE WZGLEDNE V(K) = [illegible]

D = T*S(K) = 1.76783

ROWNANIE REGRESJI PARABOLICZNEJ: Y(X)

Y(X) = 11.97520 + .28644*X + .00055*X*X

WYTRZYMALOSC BETONU OBLICZONA WG WZORU NA REGRESJE II STOPNIA

NR	I	X(I)	I	Y(I)	I	Y(I) OBLICZONE
1	I	21.80000	I	20.40000	I	18.47924
2	I	24.00000	I	18.60000	I	19.16447
3	I	21.80000	I	18.50000	I	18.47924
4	I	22.00000	I	19.00000	I	18.54132
28	I	24.80000	I	18.70000	I	19.41496
29	I	26.20000	I	21.40000	I	19.85499
30	I	25.00000	I	18.20000	I	19.47769

S(K) = 1.07794

SREDNIE KWADRATOWE ODCHYLENIE WZGLEDNE V(K) = 5.69121

PARABOLICZNY WSPOLCZYNNIK REGRESJI: N' = .983437

D = T*S(K) = 1.76782

TABELA NR 4

ROWNANIE REGRESJI KRZYWOLINIOWEJ WYKLADNICZEJ

Y(X) = .99903 * X .93229

WYTRZYMALOSC BETONU OBLICZONA WG WZORU NA REGRESJE WYKLADNICZA

NR	I	X(I)	I	Y(I)	I	Y(I) OBLICZONE
1	I	21.80000	I	20.40000	I	17.67706
2	I	24.00000	I	18.60000	I	19.33471
3	I	21.80000	I	18.50000	I	17.67706
4	I	22.00000	I	19.00000	I	17.82821
5	I	21.30000	I	17.70000	I	17.29878
6	I	19.00000	I	16.30000	I	15.55068
24	I	25.50000	I	18.80000	I	20.45897
25	I	25.80000	I	20.80000	I	20.68328
26	I	25.30000	I	20.40000	I	20.30934
27	I	24.90000	I	17.60000	I	20.00982
28	I	24.80000	I	18.70000	I	19.93489
29	I	26.20000	I	21.40000	I	20.98208
30	I	25.00000	I	18.20000	I	20.08473

S(K) = 1.31688

SREDNIE KWADRATOWE ODCHYLENIE WZGLEDNE V(K) 7.1238

WSPOLCZYNNIK KORELACJI WYKLADNICZEJ N' = S(Y)/SY = .983437

D = T*S(K) = 2.15968

REGRESJA 1-EJ ZMIENNEJ

RYSUNEK NR

TABELA NR 5

WYDRUK ROWNAN REGRESJI DWUCH ZMIENNYCH

Y = F (X1,X2)
Y (X1,X2) = .0002 X1 + -.0591 X2 + .5619

Y (X1,X2) OBLICZONE WG WZORU NA REGRESJE WIELORAKA DWUCH ZMIENNYCH

	I	XI	I	X2	I	Y (X1,X2)	I	Y(X1,X2) OBLICZ.
1	I	1700.0000	I	3.0000	I	.7180	I	.6993
2	I	1300.0000	I	3.0000	I	.6820	I	.6252
3	I	1400.0000	I	3.0000	I	.6640	I	.6437
4	I	1900.0000	I	3.0000	I	.7600	I	.7363
5	I	1500.0000	I	3.0000	I	.7340	I	.6622
6	I	500.0000	I	3.0000	I	.4970	I	.4771
7	I	600.0000	I	3.0000	I	.5340	I	.4956
8	I	1250.0000	I	3.0000	I	.5770	I	.6160
9	I	1050.0000	I	3.0000	I	.4540	I	.5789
10	I	1700.0000	I	5.0000	I	.5630	I	.5810
11	I	1300.0000	I	5.0000	I	.5400	I	.5070
12	I	1400.0000	I	5.0000	I	.5290	I	.5255
13	I	1900.0000	I	6.0000	I	.5290	I	.5589
14	I	1500.0000	I	6.0000	I	.5150	I	.4849
15	I	500.0000	I	3.0000	I	.4970	I	.4771
16	I	600.0000	I	3.0000	I	.5340	I	.4956
17	I	1250.0000	I	4.0000	I	.5190	I	.5568
18	I	1050.0000	I	3.0000	I	.4540	I	.5789

ODCHYLENIE STANDARDOWE = .054682

LACZNY WSPOLCZYNNIK KORELACJI WIELORAKIEJ R = .547095

WARTOSCI MAKSYMALNE I MINIMALNE Z POMIAROW

	MIN	MAX
X1	500.0000	1900.0000
X2	3.0000	6.0000
Y	.4540	.7600

REGRESJA 2-CH ZMIENNYCH

$$Y = A*X1 + B*X2 + C$$

RYSUNEK NR 1

Dr inż. Jan Duda
Akademia Górniczo-Hutnicza im. St. Staszica
Instytut Automatyki Inżynierii Systemów
i Telekomunikacji, Kraków

PRZEMYSŁOWE SYSTEMY CRPD I STEROWANIA NA M.Cś MERA-400 Z SYSTEMEM OPERACYJNYM SOM-3

I. Wstęp

W latach 1980-1984 w Instytucie Automatyki, Inżynierii Systemów i Telekomunikacji AGH opracowano na zlecenie przemysłu trzy systemy cyfrowe CRPD i sterowania pracujące w konfiguracji sprzętowej: MERA-400,z systemem operacyjnym SOM-3, kanał przemysłowy Inteldigit PI, stacja dysków sztywnych i znakowe urządzenia we/wy.

W referacie omówiono poszczególne systemy, głównie z punktu widzenia realizowanych zadań i oprogramowania, następnie zaprezentowano pewne przyjęte rozwiązania programowe, a w zakończeniu przedstawiono główne problemy związane z projektowaniem takich systemów na m.c. MERA-400.

II. Omówienie zrealizowanych systemów CRPDiS.

System I. Centralna rejestracja i przetwarzanie danych pomiarowych sieci gazowej kombinatu metalurgicznego.

System I przeznaczony jest do realizacji następujących zadań:

a/ odczyt i przetwarzanie w cyklu minutowym 180 analogowych sygnałów pomiarowych,

b/ bieżąca korekcja przepływów gazów z przeliczaniem liczby ekspansji i liczby przepływu dla małych przepływów,

c/ rejestracja awarii czujników pomiarowych i przekroczeń ograniczeń technologicznych,

d/ drukowanie raportów zużycia gazów za okres zmiany i doby z podaniem stanów awaryjnych czujników, okresów przekroczeń ograniczeń technologicznych i przekroczeń limitów zużycia gazów,

e/ drukowanie raportów na życzenie jak w /d/ za dowolny okres w dobie bieżącej oraz raportów przebiegów czasowych pomiarów w żądanym okresie doby,

f/ obsługa dialogu operatorów z m.c. , w tym zmiany i odczyt

ograniczeń technologicznych i limitów zużycia, odczyt wartości wybranych pomiarów, zlecanie raportów itp.
Jak widać system cyfrowy ni oddziaływuje automatycznie na obiekt w związku z tym wykorzystano jednozadaniowy system operacyjny SOM-3 w wersji INPI OUPI. Pojemność PaO dla zadania użytkownika wynosi 20 K słów.

Oprogramowanie systemu CRPD stanowi jedno zadanie napisane prawie w całości w języku FORTRAN IV S. Liczba instrukcji programu wynosi około 5000, liczba podprogramów - 62.

W systemie CRPD wydzielono podsystemy:

a/ przyjmowania z klawiatury alfanumerycznej poleceń dwóch niezależnych operatorów oraz wyprowadzania informacji wyjściowych systemu na urządzenia wyjściowe /drukarka i trzy monitory ekranowe /
b/ przetwarzania sygnałów pomiarowych w cyklu minutowym oraz generowania raportów minutowych o stanie sieci gazowej
c/ obsługi poleceń operatorów,
d/ generowania raportów okresowych i na życzenie,
e/ obsługi startu i restartu systemu.

Program główny steruje pracą podsystemów /a-d/ w czasie rzeczywistym. Podsystemy realizują swoje zadania zgodnie z ustalonym priorytetem, malejącym od a do d, przy czym podsystem /a/ może przerwać pracę wszystkich pozostałych podsystemów , a praca podsystemu /d/ może być przerwana przez każdy inny. Odczytywanie znaków z klawiatur oraz drukowanie informacji wyjściowych w podsystemie /a/ realizowane jest w trybie QUICK -RETURN.

Zgłaszanie przerwań zegarowych oraz przerwań koniecznych dla realizacji dialogu operatora z m.c. odbywa się w programie użytkowym w drodze cyklicznej kontroli czasu zegara m.c. oraz stanu operacji we/wy zapoczątkowanych w podsystemie /a/. Stwierdzenie odpowiedniego zdarzenia powoduje wykonanie wynikających z niego operacji lub zgłoszenie zlecenia przez modyfikację tablicy zleceń i przejście do programu głównego. Będzie to nazywane przerwaniem realizowanym programowo.

Program główny kontroluje cyklicznie stan tablicy zleceń

i wywołuje podprogramy realizujące zgłoszone zlecenia.

Instrukcje umożliwiające wykrywanie zdarzeń umieszczone są w segmentach podsystemów /b/ i /d/ w taki sposób aby zapewnić płynny przebieg dialogu operatorów z m.c. oraz synchronizację pracy systemu z czasem rzeczywistym.

Oprogramowanie systemu stanowi 3-poziomową strukturę overlay'ową, w której na poziomie 0 umieszczono program główny oraz segmenty podsystemu /a/. Oprogramowanie pozostałych podsystemów nie rezyduje na stałe w PaO. Liczba nakładek wynosi 31. System wykorzystuje całą dostępną PaO oraz dodatkowo na dysku przechowuje: dane pomiarowe za okres pełnej doby, stałe systemu oraz wyniki obliczeń podsystemów /b/ im/d/ na zbiorach bezpośredniego dostępu, a na dziesięciu zbiorach sekwencyjnych -zredagowane rekordy informacji wyjściowych /tzw. bufory wyjśc./

Czas przetwarzania sygnałów pomiarowych ze 180-ciu czujników w jednym cyklu wynosi około 5 s. Maksymalny czas oczekiwania na reakcję systemu po zgłoszeniu polecenia nie przekracza 3 s. /parametr ten decyduje o komforcie pracy operatora/. Maksymalny czas generowania raportu osiąga 40 min, średni - około 5 min.

System II. Centralna rejestracja danych pomiarowych i sterowanie procesem trójstopniowej rektyfikacji.

Założone funkcje systemu cyfrowego:

a/ odczytywanie 112-tu analogowych sygnałów pomiarowych z maksymalną możliwą częstością i przeliczanie ich na jednostki fizyczne,
b/ okresowa identyfikacja charakterystyk czujników,
c/ bieżąca rejestracja stanów awaryjnych i przekroczeń ograniczeń technologicznych,
d/ obsługa dialogu operatora z m.c. , w tym: wprowadzanie okresowych danych pomiarowych, wydruk raportów na żądanie i innych informacji o stanie procesu,
e/ okresowe uśrednianie i wyznaczanie trendów wartości pomiarów,
f/ drukowanie raportów dobowych,
g/ identyfikacja on-line modelu stanu ustalonego procesu,
h/ optymalizacja stanu ustalonego procesu,
i/ automatyczne ustawianie wyliczonych sterowań.

Do wprowadzania danych wykorzystuje się pulpit operatora procesu technologicznego /POPT/ oraz klawiaturę alfanumeryczną z monitorem ekranowym. Informacje wyjściowe przesyłane są na wyświetlacze POPT-a, na monitor ekranowy i drukarkę DZM.

Dla realizacji wymienionych funkcji wygenerowano 3-zadaniową wersję systemu operacyjnego SOM-3 i wykonano oprogramowanie złożone z trzech zadań;

Zadanie 1,o najwyższym priorytecie,obsługuje dialog operatora z m.c. poprzez POPT. Zadanie jest inicjowane przerwaniem zgłaszanym poprzez kanał PI lub programowo z pozostałych zadań. Odpowiednie oprogramowanie składa się z trzech nakładek napisanych w języku ASSEMBLER i zajmuje około 3.5 K słów PaO.

Zadanie 2,o niższym priorytecie,realizuje funkcje /a,b,c oraz i/. Jest odwieszane przerwaniem zegarowym dla odczytu i przetworzenia dwóch sygnałów pomiarowych oraz obsługi jednego regulatora, po czym zawiesza się na okres 50 ms. Średnia odczytu pomiaru wynosi około 8/min. Oprogramowanie napisane prawie w całości w języku ASSEMBLER a częściowo w języku FORTRAN IV S, zajmuje około 2 K słów w PaO.

Zadanie 3,o najniższym priorytecie,realizuje pozostałe funkcje systemu, w tym obsługę dialogu operatora z m.c. poprzez klawiaturę i monitor ekranowy, jak również wyprowadzanie informacji wyjściowych z buforów w pamięci dyskowej na urządzenia drukujące. Zadanie to,napisane w całości w języku FORTRAN IV S składa się z 84 segmentów /łącznie ok. 4400 instrukcji/, tworzy 4-poziomową strukturę overlayową złożoną z 42 nakładek i wykorzystuje około 8 K słów PaO.

W zadaniu tym wydzielono podsystemy:
a/ obsługi dialogu z m.c.,
b/ okresowego uśredniania pomiarów,
c/ generowania raportów,
d/ Identyfikacji procesu,
e/ optymalizacji,
f/ obsługi startu i restartu systemu,
które realizują swoje zadania według ustalonych w programie priorytetów, malejących od /a/ do /e/.

Operacje odczytu znaków z klawiatury alfanumerycznej oraz wyprowadzania informacji wyjściowych /w podsystemie a/ realizowane są w trybie QUICK&RETURN.

Praca podsystemów /d/ i /e/ może być przerywana , przy czym przerwania są zgłaszane programowo, podobnie jak w systemie I. Obsługą przerwań steruje segment główny.

Żaden z podsystemów ni e rezyduje na stałe w PaO /ze względu na ograniczenia pojemności PaO/. Powoduje to pewne zakłócenia rytmiczności dialogu operatora z m.c. i znacznie wydłużą czas obliczania sterowań optymalnych.

Wszystkie istotne dane systemu umieszczono w bloku GLOBAL COMMON o wymiarze 4400 słów.

System III. Centralna rejestracja danych pomiarowych i sterowanie procesem wielkopiecowym.

Założone funkcje systemu:

a/ odczytywanie w cyklu minutowym 40-tu analogowych sygnałów pomiarowych i przeliczenie ich na jednostki fizyczne,

b/ odczytywanie i przetwarzanie 17-tu asynklicznych sygnałów cyfrowych,

c/ rejestracja danych pomiarowych przesyłanych znakowo linią dalekopisową,

d/ obsługa dialogu operatora z m.c. , w tym: wprowadzanie okresowych danych pomiarowych, drukowanie żądanych informacji o stanie procesu,

e/ bieżąca rejestracja stanów awaryjnych i przekroczeń ograniczeń technologicznych,

f/ aktualizacja baz danych w cyklu godzinowym dla potrzeb raportowania i sterowania,

g/ obliczanie , w cyklu godzinowym, wartości zmiennych sterujących,

h/ drukowanie raportów okresowych.

Spośród omawianych sytemów , system III charakteryzuje się największym stopniem złożoności. Decydują o tym: złożona struktura i duża liczba danych wprowadzanych automatycznie i ręcznie, duża różnorodność wymaganych raportów oraz złożone i wymagające obszernych baz danych, algorytmy obliczania sterowań.

POjemność PaO dostępna dla użytkownika wynosi 32 K słów. Jako urządzenia znakowe we/wy wykorzystuje się dwie drukarki DZM/KSR. Dla potrzeb omawianego systemu wygenerowano 4-zadaniową wersję systemu operacyjnego SOM-3. Zadania realizujące wymienione funkcje zostaną omówione w kolejności malejących priorytetów.

Zadanie 1, napisane w języku ASSEMBLER , odczytuje w trybie WAIT dane znakowe przesyłane linią dalekopisową. Wykorzystuje około 200 słów PaO.

Zadanie 2, napisane w języku ASSEMBLER , jest inicjowane przerwaniem kanału PI i odczytuje sygnały cyfrowe. Wykorzystuje około 200 słów PaO.

Zadanie 3 obsługuje dialog z m.c. dwóch niezależnych operatorów, wyprowadza informacje wyjściowe systemu z buforów dyskowych na drukarki oraz przetwarza sygnały odczytane przez zadania 1 i 2.

Oprogramowanie zadania 3 napisane w języku FORTRAN IV S składa się z 26 podprogramów i tworzy 3-poziomową strukturę overlay'ową. Wykorzystuje około 9 K słów PaO i pamięć dyskową. Po wykonaniu operacji zleconych przez inne zadania lub operatorów, zawiesza się na okres 1 s. Operacje odczytu znaków z klawiatur i drukowania na drukarce realizuje się w trybie QUICK-RETURN.

Zadanie 4 realizuje pozostałe funkcje systemu. Oprogramowanie napisane w języku FORTRAN IV S zawiera 90 podprogramów /około 5000 instrukcji/ i tworzy strukturę overlay'ową złożoną z 40 nakładek. Wykorzystuje 14.5 K słów PaO i pamięć dyskową. Program główny w oparciu o cykliczną, programową kontrolę stanu zegara m.c. i stanu tablicy zleceń operatorów, steruje przebiegiem obliczeń.

Wszystkie istotne zmienne i parametry systemu są umieszczone w bloku GLOBAL COMMON o wymiarze 5400 słów.

W systemie nie występuje deficyt czasu.

Powierzenie obsługi dialogu operatorów z m.c. zadaniu inicjowanemu przez zegar gwarantuje rytmiczność przebiegu dialogu i praktycznie niezauważalne opóźnienie reakcji systemu na zgłoszone polecenie.

III. Procedury pomocnicze dla potrzeb systemów CRPD

Oprogramowanie omawianych systemów ma w swojej zasadniczej części charakter unikalny dla każdego z nich. Wynika to ze specyfiki stawianych im wymagań i ograniczeń pojemności PaO. Nie mniej, zasługuje na uwagę sposób rozwiązania problemu koordynacji operacji wejścia/wyjścia.

W systemach CRPDiS konieczne jest prowadzenie dialogu z m.c. oraz agregacja informacji wyjściowych w spójne zbiory. Liczba urządzeń drukujących jest przy tym na ogół mniejsza niż liczba wymaganych zbiorów wyjściowych. Poza tym część z tych urządzeń jest sprzężona z klawiaturami na których mogą być rozpoczęte operacczytania danych. Konieczne jest zatem buforowanie informacji wyjściowych w pamięci dyskowej i koordynacja przesyłania zawartości tych buforów na urządzenia wyjściowe z procesem wprowadzania danych.

W omawianych systemach przyjęto następujące rozwiązanie:

a/ informacje wyjściowe podzielono na rozłączne zbiory wyjściowe w taki sposób aby każdy z nich był zapełniany przez co najwyżej jedno zadanie,

b/ każdemu zbiorowi przydzielono:
- bufor w postaci zbioru sekwencyjnego na dysku,
- nazwę strumienia wejściowego,
- nazwę strumienia wyjściowego,
- numer urządzenia na którym ma być drukowany dany zbiór,
- licznik rekordów wpisanych do zbioru,
- licznik rekordów odczytanych ze zbioru /wydrukowanych/

c/ w procedurach generujących wydruki wyjściowe, przed wydrukowaniem każdego rekordu wywołuje się podprogram, który:
- podstawia do zmiennej NU nazwę strumienia wyjściowego deklarowanego zbioru,
- sprawdza czy strumień ten odpowiada zbiorowi sekwencyjnemu na dysku: jeśli TAK - modyfikuje licznik rekordów wpisanych
  jeśli NIE - kończy działanie.

d/ drukuje się rekord w trybie WAIT na strumieniu NU

e/ każdemu urządzeniu drukującemu przypisuje się :
- standardową tablicę UFT,

- indeks sprzężenia z klawiaturą /0 lub 1/
- indeks blokady programowej urządzenia,
- całkowitą liczbę rekordów zgłoszonych do wydrukowania na danym urządzeniu

f/ wpisywanie kolejnych znaków z klawiatur oraz drukowanie rekordów ze zbiorów wyjściowych realizuje się w trybie QUICK-RETURN w specjalnym podprogramie, który ma za zadanie:
- sprawdzanie stanu kolejnych zbiorów wyjściowych,
- sprawdzenie wszystkich warunków wymaganych do zainicjowania operacji wyjścia /stan operacji wejścia, stan pozostałych zbiorów wyjściowych/
- przepisanie rekordu z dysku do bufora wPAO w trybie WAIT
- zainicjowanie drukowania w trybie QUICK-RETURN
- modyfikacja licznika rekordów odczytanych
- jeśli liczniki rekordów wpisanych i odczytanych są równe to dodatkowo następuje zerowanie liczników, anulowanie blokad urządzenia i przewinięcie strumieni wejściowego i wyjściowego oraz zainicjowanie operacji wejścia jeśli urządzenie jest sprzężone z klawiaturą.

Przyjęte rozwiązanie zapewnia:

1. spójność zbiorów wyjściowych na wydrukach
2. możliwość doraźnego przesyłania informacji wyjściowych bezpośrednio na drukarkę lub zaślepkę
3. możliwość równoczesnego drukowania na kilku urządzeniach wyjściowych
4. możliwość blokowania procesu drukowania np po zapełnieniu ekranu monitora /odblokowanie wymaga wpisania specjalnego polecenia
5. możliwość kopiowania na papierze całego zbioru rekordów zgłoszonych do wydruku na monitorze ekranowym.

IV. Problemy związane z projektowaniem dużych systemów CRPD na m.c. MERA-400

Podstawowym problemem przy projektowaniu dużych systemów CRPD na m.c. MERA-400 z systemem operacyjnym SOM-3 jest ograniczenie pojemności bezpośrednio adresowalnej PAO do 32k słów. Zmusza to do komplikowania struktury overlayowej oprogramowania i do minimalizacji liczby zadań. Każde zadanie

wymaga zadeklarowania bloku komunikacji, a jeśli realizuje operacje we/wy, to w trakcie konsolidacji może zostać dodatkowo dołączona odpowiednia biblioteka systemowa. W efekcie wydzielenie zadania powduje utratę do kilku tys. słów PAO.

Konsekwencją minimalizacji liczby zadań jest konieczność projektowania systemów przerwań zgłaszanych programowo, co znacznie komplikuje oprogramowanie.

Ograniczenia zasobów PAO utrudniają także unifikację oprogramowania.

Duże utrudnienia stwarza brak możliwości deklarowania wielu globalnych bloków COMMON. w systemie wielozadaniowym.

Pewne problemy wynikają z ograniczeń języka FORTRAN IV S dotyczących głównie indeksów tablic, braku danych dotyczących wymagań pamięciowych procedur systemowych, niedostatków procedury LINK EDITOR i usterk oprogramowania systemowego.

Autor referatu jest autorem projektu i w znacznym stopniu wykonawcą tej części przedstawionego oprogramowania ,która jest zapisana w języku FORTRAN IVS. Pozostałe prace związane z omawianymi systemami wykonał zespół pracowników IAISiT AGH

Mgr MARIA UFNALSKA B.P.B.B.O. MIASTOPROJEKT 2 ŁÓDŹ -
WŁODZIMIERZ KARCZEWSKI - " -
Mgr JERZY ZIELONKA - " -

SPOSÓB ROZWIĄZANIA PROBLEMU KOSZTORYSOWANIA PRZY POMOCY EMC MERA 400 NA PODSTAWIE DOŚWIADCZEŃ BPBBO MIASTOPROJEKT-2

Wstęp

Kosztorys jest nieodłączną częścią dokumentacji projektu technicznego, stanowi podstawę do zawarcia umowy i dokonania rozliczeń pomiędzy inwestorem a wykonawcą. Wprowadzenie zasad reformy gospodarczej w budownictwie wymusiło przyjęcie systemu kosztorysowania zapewniającego ustalenie bardziej zindywidualizowanych niż w latach ubiegłych cen za roboty budowlano-montażowe tak, aby wartości kosztorysowe realizowanych obiektów stanowiły w maksymalnym stopniu autentyczną pochodną ich struktury przestrzenno-materiałowej oraz warunków budowy. Do niedawna automatyzacja kosztorysowania była sprawą swobodnego wyboru, w chwili wprowadzenia zasady szczegółowego rozliczania robót budowlano-montażowych stała się wręcz koniecznością. Duża pracochłonność obowiązującej metody przy równoczesnym braku wykwalifikowanych kosztorysantów w biurach projektowych wymusza intensyfikację prac programowych i wdrożeniowych w zakresie automatyzacji kosztorysowania. Sądzimy, że krótka prezentacja stanu prac nad automatyzacją kosztorysowania w BPBBO Miastoprojekt-2 w Łodzi i wynikające stąd wnioski mogą być interesujące dla innych przedsiębiorstw zajmujących się tymże zagadnieniem, a wymiana poglądów i doświadczeń pomoże nam wszystkim w optymalnym ukierunkowaniu dalszego działania.

Zespół ETO przy BPBBO MIASTOPROJEKT-2 sprawę automatyzacji kosztorysowania przy pomocy EMC zajmuje się od 1978 r., to jest z chwilą zakupu zestawu komputerowego MERA-400. W ciągu tego okresu opracowano własny, w miarę elastyczny system automatycznego kosztorysowania o nazwie KOSZTORYS, który w wielkim stopniu ułatwia i przyśpiesza pracę kosztorysantów. System "KOSZTORYS" służy do wykonywania kosztorysów budowlanych i branżowych oraz zestawień robocizny, materiałów i sprzętu na podstawie podanych przez kosztorysantów danych

wejściowych /przedmiaru/.
W pierwszym etapie prac w oparciu o KATALOGI CEN KOSZTORYSOWYCH przyjmując jako założenie wyjściowe uniwersalność programów i selektywność zbiorów danych /zdeterminowaną potrzebami naszego biura/ opracowano bazę danych i zapisano ją za pomocą odpowiednich programów na zbiorach bezpośredniego dostępu. Zmiany w przepisach dotyczących metod kosztorysowania spowodowały dokonywanie ciągłych modyfikacji istniejących programów oraz powstawania nowych. W wyniku prawie 6-letnich prac dotyczących tych zagadnień powstał plik 11 programów oraz bogata baza danych stałych zawierająca Katalogi Cen Kosztorysowych i odpowiadające im Katalogi Norm Kosztorysowych.
Przy opracowaniu systemu przyjęto jako wiążące Rozporządzenie Ministra do Spraw Cen z dnia 31 grudnia 1982 r. w sprawie zasad i metod ustalania kosztów uzasadnionych robót budowlanych /Dziennik Ustaw PRL 1984 r. nr 3 poz. 20/ oraz Zarządzenie nr 6 Ministra Budownictwa i PMB z dnia 28.04.1983 r. w sprawie kosztorysowania szczegółowego robót budowlanych i montażowych Programy nasze były przedstawione do zaopiniowania w Instytucie Organizacji Zarządzania i Ekonomiki Przemysłu Budowlanego "ORGBUD" w ramach Klubu Użytkowników MK MERA-400. Uznane zostały jako spełniające wymogi obowiązujących zarządzeń oraz stwierdzono poprawność formy wydawanych wydruków. System został przewidziany do obsługi wszystkich branż przy kosztorysowaniu obiektów budowlanych. Obejmuje on następujące programy i zbiory:

- program KCK
- program WYKAZ RMS
- program LIST
- program KOSZT RMS
- program KOSZIN
- program METODA
- 5 programów pomocniczych
- zbiory KCK
- zbiory KNK
- zbiory nazwisk, jednostek, robót, inwestorów, działów i wykonawców.

Po odpowiedniej aktualizacji ostatnio wymienionych zbiorów

system może być eksploatowany w dowolnym biurze czy przedsiębiorstwie wykonawczym posiadającym zestaw MK MERA-400. Z uwagi na wielkość zbiorów KCK i KNK a z drugiej strony ograniczoną pojemność pamięci dyskowych dokonano podziału dotychczas założonych zbiorów na cztery kasety:

kaseta I

Zbiory KCK i KNK budowlane i roboty ziemne
biblioteka programów.

Kaseta II

Zbiory KCK i KNK instalacje sanitarne
biblioteka programów

kaseta III

zbiory KCK i KNK instalacje elektryczne
biblioteka programów

kaseta IV

zbiory KCK i KNK roboty drogowe
biblioteka programów.

6 głównych programów systemu pozwala na realizację kosztorysów oraz zestawień robocizny, materiałów i sprzętu w następujących wariantach wynikających każdorazowo z potrzeb i ustaleń między inwestorem a wykonawcą w oparciu o jednakowo przygotowane dane.

a/ Program KCK służy do sporządzenia tradycyjnego kosztorysu według cen jednostkowych z KCK z uwzględnieniem mnożnika M-82

b/ Program WYKAZ RMS służy do sporządzenia zestawień robocizny, materiałów i sprzętu wg KNK w odniesieniu do poszczególnych elementów robót oraz dla całego obiektu

c/ Program LIST służy do sporządzania tzw. "kosztorysu ślepego" to znaczy kosztorysu w układzie pozycji wg KCK nie uwzględniając cen

d/ Program KOSZT RMS służy do sporządzenia kosztorysu zagregowanego w układzie elementów robót wg norm KNK oraz jednostkowych cen indywidualnych oraz stawek jednostkowych użytkownika

e/ Program KOSZIN służy do sporządzenia kosztorysu w układzie pozycji KCK ale wg norm KNK oraz jednostkowych cen indywi-

dualnych oraz stawek jednostkowych użytkownika

f/ Program METODA służy do sporządzenia kosztorysu szczegółowego zgodnego z metodą zaprezentowaną w załączniku do Zarządzenia nr 6 w dwóch wariantach

- uwzględniający nakłady robocizny, materiałów i sprzętu oraz ceny wykonawcy w rozbiciu na pozycje kosztorysowe
- tzw. "ślepy", uwzględniający tylko nakłady robocizny, materiałów i sprzętu w rozbiciu na pozycje kosztorysowe

W każdym z tych wariantów może być drukowane sumaryczne zestawienie materiału i zestawienie sprzętu dla całego obiektu.

Wszystkie programy napisane są w języku FORTAN IV S i mogą być realizowane na EMC MERA-400 posiadające minimalną konfigurację.

Wcześniejsze zapowiedzi wprowadzenia nowych norm kosztorysowych - KNR-ów jak i nowej metody szczegółowego kosztorysowania robót budowlano-montażowych, spowodowało podjęcie tematu przez Zespół.

Po konsultacjach u zżytkownikami naszych programów i przeanalizowaniu własnych doświadczeń podjęliśmy prace programowe, poważnie zaawansowane w chwili obecnej, nad stworzeniem nowego systemu.

Przyjęcie cech dotyczcasowego systemu uważanych przez użytkowników za zalety takich jak:

- proste przygotowanie danych
- elastyczność zakładanych zbiorów
- możliwość ich adaptacji dla potrzeb każdego użytkownika
- możliwość uwzględnienia w obliczeniach kosztorysowych dodatkowych pozycji spoza zbiorów przy jednocześnie posiadanej przez Zespół możliwości wglądu do powstakących KNR-ów pozwala na domniemanie, iż w momencie ukazania się nowych norm będziemy w stanie oferować zainteresowanym programy sterujące i sukcesywnie zbiory danych KNR.

WNIOSKI

Na podstawie dotychczasowych doświadczeń z eksploatacji programów kosztorysowych w BPBBO Miastoprojekt 2 w Łodzi oraz opinii współpracujących z nami biur projektów i przedsiębiorstw budowlano-montażowych można sformułować następujące wnioski:

1. Pracochłonność obowiązującej metody szczegółowego kosztorysowania robót budowlano-montażowych przy równoczesnym braku kosztorysantów wymusza powszechność automatyzacji kosztorysowania.

Logicznym następstwem tego stwierdzenia jest:

2. Konieczność uwzględnienia wymogów ETO przez autorów nowych norm.

3. Wdrożenie automatyzacji kosztorysowania w przedsiębiorstwie powoduje określone reperkusje:

- struktury zatrudnienia /znaczne odciążenie w pracy kosztorysantów i całkowite hali maszyn/
- zmniejszenie czasu opracowania dokumentacji, co zamyka drogę odwrotu od tak przyjętego toku działania.

Znaczne już wyeksploatowanie posiadanych MERA 400 i z drugiej strony zamierzone w najbliższym czasie zaprzestanie produkcji MERY 400 i niepodjęcie w to miejsce kompatybilnego innego sprzętu stawia użytkowników przed koniecznością podjęcia w najbliższym czasie wspólnej decyzji wyboru innego sprzętu o odpowiednich parametrach technicznych i cenie, który stałby się powszechnym narzędziem naszego dalszego działania.

dr inż Zbigniew Mrozek
Instytut Elektrotechniki i Elektroniki
Politechnika Krakowska
Kraków, ul. Warszawska 24

WYGŁADZANIE I APROKSYMACJA WYNIKÓW POMIAROWYCH Z UŻYCIEM FUNKCJI SKLEJANEJ

WSTĘP

Jednym z zastosowań komputera jest przetwarzanie danych liczbowych będących wynikami pomiarów. Pomiary są obarczone błędami, co niekorzystnie wpływa na wyniki obliczeń. W niniejszym artykule rozważa się sytuacje, gdy:

1. Dyskretne wartości wyników pomiarów będą aproksymowane przez funkcję sklejaną (spline function),

2. Pomiary z błędami grubymi odrzucono, a pomiary wykonane dla tej samej wartości zmiennej niezależnej, zostały zastąpione przez wartość średnią tych pomiarów.

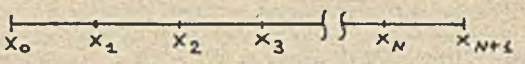
WIELOMIANOWA FUNKCJA SKLEJANA

Wyniki pomiarów będą aproksymowane przez funkcję sklejaną daną wzorem

$$Y = \sum_{i=1}^{m} a_i \Phi_i(x) \tag{1}$$

przy czym
- Y - wielomianowa funkcja sklejana aproksymująca
- x - zmienna niezależna;
- a_i - współczynniki - do obliczenia;
- $\Phi_i(x)$ - bazowe funkcje sklejane [2];
- m - wymiar przestrzeni aproksymującej = liczba funkcji bazowych

Przyjęto, że zmienna niezależna x należy do przedziału $[x_0, x_{N+1}]$. Wewnątrz tego przedziału umieszcza się N punktów zwanych węzłami funkcji sklejanej (rys. 1)



Rys 1. Węzły funkcji sklejanej.

Funkcja sklejana nie jest funkcją analityczną. Każdy współczynnik a oddziaływuje tylko na pewien fragment funkcji sklejanej. Daje to znaczne możliwości kształtowania krzywej aproksymującej.

W celu uproszczenia założono stałą odległość h między węzłami.

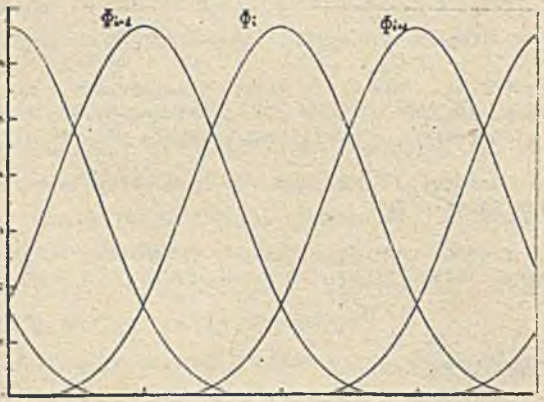
$$h = x_i - x_{i-1}, \qquad i = 1,2,\ldots,N+1. \tag{2}$$

Przyjęto, że do aproksymacji będzie stosowana funkcja sklejana stopnia

trzeciego. Dla takich funkcji wymiar przestrzeni aproksymującej wynosi m=N+4. Jest to zarazem liczba bazowych funkcji sklejanych i liczba nieznanych współczynników q_i.

FUNKCJE BAZOWE.

Wykresy bazowych funkcji sklejanych stopnia trzeciego przedstawiono na rys. 2.



Rys 2. Bazowe funkcje sklejane.

Każda z funkcji bazowych $\Phi_i(x)$ jest utworzona z czterech wielomianów trzeciego stopnia o wzorach [2,5]

$$(3)\qquad \Phi_i(x) = \begin{cases} w(h-t)^3 & \text{dla } x\in[x_{i-1}, x_i] \\ w(4h^3+3t^3-6ht^2) & \text{dla } x\in(x_{i-2}, x_{i-1}] \\ w(h^3+3(h^2t+ht^2-t^3)) & \text{dla } x\in(x_{i-3}, x_{i-2}] \\ wt^3 & \text{dla } x\in(x_{i-4}, x_{i-3}] \\ 0 & \text{dla } x\notin(x_{i-4}, x_i) \end{cases}$$

przy czym
$i=1,2, \ldots, N+k+1=7$;
$h=(x_i-x_{i-1})$ = const;
$t=(x-x_n)/h$; x_n - pierwszy węzeł na lewo od x;
$w = 1/6h$

Powyższe wzory są tak dobrane, że uzyskuje się ciągłość funkcji sklejanej (1), oraz jej pierwszej i drugiej pochodnej - w całym przedziale $[x_0, x_{n+1}]$. Trzecia pochodna ma, poza węzłami x_i, stałą wartość. W węzłach dopuszczalne są skokowe zmiany wartości tej pochodnej.

APROKSYMACJA I WYGŁADZANIE WYNIKÓW POMIAROWYCH

Mając określone współrzędne węzłów, oblicza się wartości wszystkich funkcji bazowych (3) dla każdej wartości zmiennej niezależnej. Kombinacja liniowa wartości tych funkcji bazowych i nieznanych współczynników q ma

dać wartość jednego z pomiarów - zgodnie ze wzorem (1).

Powtarzając te obliczenia dla kolejnych wartości zmiennej niezależnej, otrzymuje się układ r równań o m niewiadomych współczynnikach a_i. Przedstawiony sposób postępowania nosi nazwę metody kollokacji. Otrzymane równania można zapisać w postaci macierzowej

(4) $$A\underline{a} = \underline{y}$$

przy czym

A - macierz o wymiarze r x m, zawierająca wartości funkcji bazowych;
r - liczba punktów pomiarowych;
a - wektor o wymiarze m nieznane współczynniki;
y - wektor o wymiarze m zawiera wyniki pomiarów.

Dalsze postępowanie będzie zależne od wymiarów macierzy A. Rozważa się następujące przypadki:

a) ilość pomiarów jest równa ilości współczynników (macierz A kwadratowa) - współczynniki oblicza się metodą eliminacji Gaussa. Jeśli macierz jest osobliwa, należy usunąć odpowiednią ilość wierszy macierzy, co prowadzi do przypadku b.

b) ilość pomiarów jest mniejsza od ilości współczynników. Proponuje się zmniejszenie wymiaru (ilości współczynników) funkcji aproksymującej. Można też zadać dowolne wartości pewnej liczbie współczynników, a pozostałe wyliczyć.

c) ilość pomiarów większa od ilości współczynników funkcji aproksymującej. Jest to przypadek najciekawszy, dający możliwość częściowej eliminacji błędów pomiarowych (wygładzenie wyników pomiarowych)

Dla przypadku c , gdzie mamy do czynienia z nadokreślonym układem równań algebraicznych, można zastosować:

1. skreślenie takiej ilości wierszy macierzy A, aby uzyskać macierz kwadratową. Istotna wada takiego postępowania, będzie otrzymywanie różnych rozwiązań równania (4), w zależności od tego, które wiersze macierzy A zostały skreślone. Stosując tę metodę, traci się bezpowrotnie informacje, zawarte w pomiarach, które odpowiadają skreślonym wierszom macierzy. Pogarsza to dokładność aproksymacji.

2. korzystniejszym wariantem jest zastosowanie specjalnej procedury do rozwiązywania nadokreślonego układu równań algebraicznych. Wykorzystano do tego celu metodę obliczania pseudoodwrotności macierzy prostokątnej [1]. W rezultacie, dla każdego założonego wymiaru przestrzeni funkcji sklejanych, otrzymuje się odpowiednią liczbę współczynników, nie tracąc przy tym informacji związanej z nadmierną liczbą punktów pomiarowych. Współczynniki oblicza się ze wzoru

(5) $$a = A^{\#}y$$

przy czym $A^{\#} = (A^T A)^{-1} \cdot A^T$ jest pseudoodwrotnością macierzy A

Istotną zaletą powyższej metody (w porównaniu do prezentowanej w [3]), jest uśrednienie (wygładzenie) danych pomiarowych. Zmniejsza to znacznie wrażliwość metody na niedokładności pomiarów.

WNIOSKI

W artykule przedstawiono metodę aproksymacji i wygładzania wyników pomiarowych - z użyciem funkcji sklejanej. Zaprezentowano specjalną procedurę do rozwiązywania nadokreślonego układu równań algebraicznych. Przedstawiona metoda jest stosowana na komputerach MERA 400 i MERA 60 i była testowana z dobrymi rezultatami dla kilkudziesięciu punktów pomiarowych.

Niniejszy artykuł redagowano na MERZE 400, wykorzystując program TEXT, opracowany przez Instytut Informatyki UJ w Krakowie.

LITERATURA

[1] Bjorck A., Dalquist G.: Numerical methods, Prentice Hall 1974, /Warszawa 1983 WNT/.

[2] Boor C.: Package for calculating with B-splines. SIAM Num. Anal. Vol.14: 1977 No 3 str. 441-472

[3] Mrozek B., Mrozek Z.: Uwzględnienie nieliniowości parametrów modelu układu silnik indukcyjny - falownik prądu. II Sympozjum: Podstawowe Problemy Energoelektroniki, Bielsko-Biała, 10-12 listopad 1983.

[4] Mrozek B., Mrozek Z.: Metody aproksymacji krzywej magnesowania silnika indukcyjnego. III Krajowa Konferencja Energoelektroniki, Napędu Elektrycznego i Trakcji, Warszawa 9-11 kwietnia 1984.

[5] Mrozek Z.: Wybrane, skończenie wymiarowe metody aproksymacji obiektów o parametrach rozłożonych - praca doktorska. Kraków 1982, AGH.

adres domowy
Na Skarpie 9/30
31-909 Kraków

mgr inż. Roman Barecki
mgr inż. Witold Czarnecki
mgr inż. Krzysztof Makowiecki
Wrocławskie Biuro Proj.-Bad.
Budownictwa Przemysłowego

WB - system projektowania konstrukcji prętowych

System WB do obliczeń statycznych i wymiarowania płaskich układów prętowych opracowano na emc MERA-400 i ODRA-1305.

Posiada automatyczną kontrolę danych wejściowych, generatory geometrii i obciążeń układu oraz katalogi profili walcowanych i spawanych.

Długie sekwencje obliczeń stanowią o przydatności oprogramowania w praktyce projektowej.

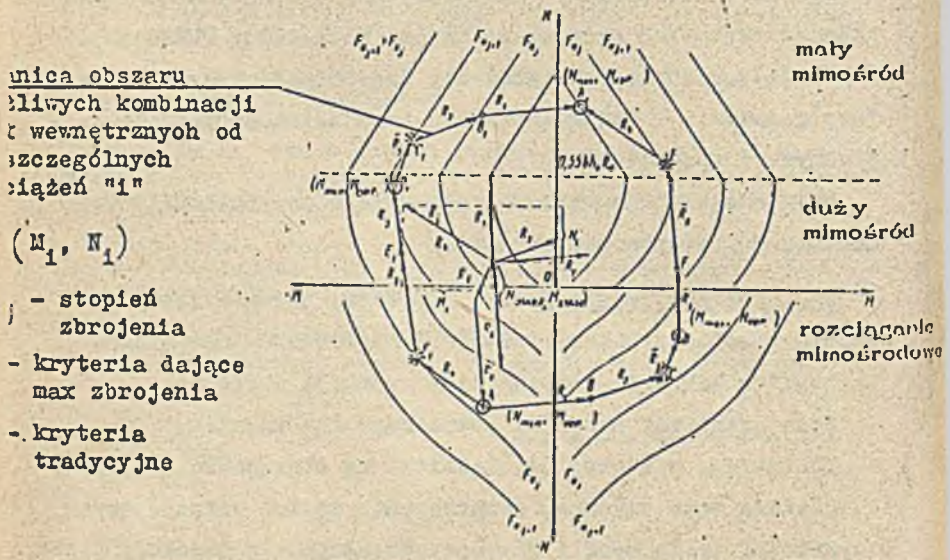
Wartości ekstremalne: sił, naprężeń, przemieszczeń i oddziaływań, spełniają wymogi PN-82/B-02000 "Obciążenia budowli" w zakresie kombinacji i redukcji obciążeń.

Stan graniczny nośności rozpatruje kombinacje:

- podstawową, w której uwzględnione są obciążenia obliczeniowe stałe oraz zmienne, uszeregowane wg ich wzrastających wartości liczbowych i przyporządkowanych im współczynników jednoczesności ψ_0,
- wyjątkową, uwzględniającą obciążenia stałe, zmienne ze współczynnikiem jednoczesności $\psi_0 = 0.8$ i jedno obciążenie wyjątkowe.

Typowe kryteria doboru ekstremalnych sił wewnętrznych (M_{max} - maks. moment i N - towarzysząca siła osiowa lub M i N_{max}) jako prowadzące do znacznego niedowymiarowania przekroi mimośrodowo-ściskanych zastąpiono dla elementów:

- stalowych, kryterium ekstremalnych naprężeń z uwzględnieniem wyboczenia, zwichrzeniem i wpływu obciążeń działających prostopadle do płaszczyzny ramy,
- żelbetowych mimośrodowo-ściskanych bądź rozciąganych, kryterium wyboru decydującym o max stopniu zbrojenia przekroju.



unica obszaru
liwych kombinacji
t wewnętrznych od
szczególnych
ciążeń "i"

(M_1, N_1)

- stopień zbrojenia
- kryteria dające max zbrojenia
- kryteria tradycyjne

Stan_graniczny_użytkowania_ obejmuje sprawdzenie następujących kombinacji od obciążeń charakterystycznych:

- w elementach stalowych kombinacji podstawowej, obejmującej obciążenie stałe i jedno z najniekorzystniejszych obciążeń zmiennych

w elementach żelbetowych - kombinacji podstawowej, oraz "obciążeń długotrwałych" w której przemieszczenia obliczane są od obciążeń stałych i długotrwałych części obciążeń zmiennych.

Wymiarowanie konstrukcji stalowych spełnia warunki normowe PN-80/B-03200 w zakresie stanów granicznych nośności i użytkowania. Wymiarowaniu podlegają profile walcowane jedno i dwugałęziowe oraz blachownice spawane zawarte w katalogu systemu. Program sprawdza stany graniczne wybranych elementów stalowych /z uwzględnieniem wyboczenia, zwichrzenia, stateczności miejscowej/ i dobiera profile w taki sposób, aby naprężenia mieściły się w przedziale 95-102% wytrzymałości obliczeniowej. Profile dwugałęziowe projektowane są wraz z przewiązkami lub skratowaniem. Obliczane są również grubość i długość spoin łączących przewiązki lub krzyżulce skratowania z gałęziami pręta. Dla profili spawanych program dobiera rozstaw żeber poprzecznych oraz spoiny między półkami a środnikiem. Dobór nowych elementów konstrukcji odbywa się w sposób pozwalający na zachowanie ciągłości między wskazanymi jej częściami.

Organizacja systemu

WB-21 do obliczeń statyczno-wytrzymałościowych układów ramowo-kratowych oraz WB-22 do obliczeń rusztów, stanowią niezależne programy napisane w języku FORTRAN. W wersji emc MERA-400 obejmują 15 samowywołujących się programów w których dane pośrednie przekazywane są za pomocą dyskowych zbiorów o bezpośrednim dostępie. W celu skrócenia czasu obliczeń część procedur napisano w języku ASSEMBLER

W chwili obecnej eksploatowane są następujące wersje oprogramowania:

- dla emc MERA-400 o wyposażeniu standardowym i pod systemem operacyjnym SOM-3, gdzie zbiory zewnętrzne i nakładki programu przechowywane są na dysku,
- j.w. lecz z dodatkową pamięcią półprzewodnikową 128 K, pod systemem SOM-3 poszerzonym o ekstrakody PUTPAO i GETPAO. W wersji tej część zbiorów oraz częściej używane nakładki umieszczono w szybkiej pamięci zewnętrznej i zwiększono pamięć operacyjną o 8 K. W wyniku uzyskano znaczne skrócenie czasu obliczeń.

Powyższe wersje nie obejmują wymiarowania konstrukcji stalowych. Odpowiednie moduły są w przygotowaniu.

- dla emc ODRA-1305 w systemie operacyjnym GEORG-3 i pod EXECUTION zrealizowano pełny zakres obliczeń.

Ograniczenia podyktowane są wielkością zbiorów dyskowych = 32 K. Muszą być spełnione:

- $(r + 1) \times W \leqslant 3640$
- $W \times P_k \leqslant 10920$
- $E \times P_k \leqslant 5461$

gdzie: r - szerokość pasma/rozpiętość numeracji końców pręta/
W - ilość węzłów
P_k - ilość przypadków obciążeń
E - ilość elementów

Zamierzenia rozwojowe systemu obejmują:

- wymiarowanie elementów żelbetowych z uwzględnieniem stanu g... nośności i użytkowania, doborem wkładek i strzemion, w tym wymiarowanie słupów dwukierunkowo - mimośrodowo-ściskanych,
- wprowadzenie elementów mieszanych prętowo-powierzchniowych,
- obliczenia i wymiarowanie układów z obciążeniem statycznym i dynamicznym.

mgr Jerzy Grall
Instytut Włókiennictwa
Łódź, ul. Gdańska 91/93

SYSTEM PERCY - WSPOMAGANA KOMPUTEREM ANALIZA ROZPŁYWU ENERGII ELEKTRYCZNEJ, STANU SIECI I MOŻLIWOŚCI OPTYMALIZACJI KOMPENSACJI MOCY BIERNEJ

W Instytucie Włókiennictwa w Ośrodku Energetyki Przemysłu Lekkiego zbudowano elektroniczne, cyfrowe urządzenie "PERCY-01", które umożliwia wielopunktowe, równoczesne pomiary rozpływu energii elektrycznej w dowolnie wybranej i dowolnie rozgałęzionej elektroenergetycznej sieci zakładowej. Do aparatu PERCY-01 podłączane są przystawki pomiarowe, wmontowane w liczniki energii elektrycznej czynnej lub biernej. Aparat zbiera i rejestruje informacje o obciążeniu sieci elektrycznej w poszczególnych wybranych punktach pomiarowych według zadanego programu czasowego; wstępnie przetworzone informacje zapisywane są za pośrednictwem podłączonego do aparatu perforatora DT105S na taśmie papierowej w kodzie zerojedynkowym.

Aparat ma możność współpracy z 32 licznikami z wbudowanymi przystawkami pomiarowymi i zbiera informacje o energii elektrycznej z wszystkich punktów pomiarowych jednocześnie i w sposób ciągły w zadanych przedziałach czasowych. Rejestracja danych pomiarowych dotycząca zliczonej energii w każdym przedziale czasowym następuje na żądanie operatora lub zgodnie z zadanym programem. Pomiary dokonywane są z dokładnością wielokrotnie większą, niż jest to możliwe przy odczycie wzrokowym z licznika. Aparat m.in. eliminuje potrzebę licznej obsługi, niezbędnej przy klasycznym pomiarze przez odczyt w przypadku pomiarów wielopunktowych.

Urządzenie oparte jest na układach scalonych produkcji krajowej, jest łatwe do uruchomienia i nie wymaga stałego nadzoru, przy czym może współpracować z licznikami o dowolnych stałych. Aparat przystosowany jest do przenoszenia, a dzięki budowie panelowej możliwa jest szybka naprawa.

Oprogramowanie systemu PERCY wykonane zostało w Zespole d/s ETO Instytutu Włókiennictwa na m.k. MERA-400. Programowa obróbka danych pomiarowych przebiega w trzech fazach:

a/ czytanie z tasiemki perforowanej i dekodowanie danych pomiarowych na kod akceptowany przez m.k. MERA-400;
b/ pierwotne przetwarzanie danych pomiarowych;
c.1./ wyznaczanie optymalnych wartości mocy baterii kondensatorów stosowanych do kompensacji mocy biernej w elektrycznej sieci promieniowej,
c.2./ j.w. w odniesieniu do sieci rozgałęzionej.

Ad.a. Program #MØ53 wczytuje taśmę perforowaną z zarejestrowanymi na niej danymi, przyjmuje dane wprowadzane w trybie dialogowym przez operatora z monitora systemowego, dotyczące wartości współczynników przeliczeniowych właściwych dla każdego z podłączonych czujników oraz informację dotyczącą liczby aktualnie podłączonych czujników /maks. 32/ zwiększonej o 1 /jeden czujnik rejestruje czas pomiaru/, następnie realizuje konwersję kodu i wstępne przetwarzanie danych pomiarowych, wyniki przetwarzania zapisuje na dysku, emitując jednocześnie tabulogram kontrolny zawierający następujące dane: nr rejestru /punktu pomiarowego/, czas pomiaru k-tego, nr rejestru n-tego, zawartość rejestru n-tego. Błąd w numerze rejestru sygnalizowany jest jako 99, a w zawartości rejestru jako 9999, co oznacza, że na tasiemce perforowanej wystąpił znak różny od Ø i 1.

Program #MØ53 napisany został w języku FORTRAN IV, a z programu głównego występuje odwołanie do podprogramu w języku ASSEMBLER, realizującego ww konwersję kodu. Opracowane zostały dwie wersje programu - dla przedziałów czasowych 15-minutowych i 30-minutowych. Czas pracy programu dla wersji 15-minutowej i danych całodobowych, przy pełnym wykorzystaniu możliwości programu - ok.40 min.

Ad.b. Program #MØ54 korzysta z danych zapisanych na dysku przez program #MØ53, oraz z danych wprowadzanych w trybie dialogowym przez operatora z klawiatury monitora systemowego. Program systematyzuje dane na podstawie pomierzonej energii: czynnej, biernej rzeczywistej oraz biernej pobieranej przez baterię kondensatorów, wylicza moce czynne, bierne rzeczywiste, bierne

naturalne, pozorne rzeczywiste i pozorne naturalne dla każdej godziny pomiarów. Program liczy również wartości średnie tych wielkości fizycznych oraz znajduje ich wartości maksymalne i minimalne. Emitowany jest tabulogram z wynikami obliczeń, tabele z pełnymi danymi i wynikami oraz wykresy punktowe.

Następujące dane wprowadzane są z monitora systemowego: data obliczeń, nr zlecenia, zleceniodawca, miejsce pomiarów /oddział, stacja/, obiekt zasilania, nr zasilacza, typ przewodu, długość przewodu, napięcie, data wykonania pomiarów, wykonawcy pomiarów, parametry punktu pomiarowego. Dla parametrów punktu pomiarowego możliwe są trzy warianty:

1/ przypadek, gdy mierzona jest energia czynna, energia bierna oraz energia pobierana przez kondensatory,
2/ przypadek, gdy mierzona jest tylko energia czynna oraz energia bierna,
3/ znak końca działania programu, w przypadku niższej od dopuszczalnej liczby punktów pomiarowych.

Drukowanie tabel i wykresów dla poszczególnych punktów pomiarowych jest opcjonalne.

Program #MØ54 napisany jest w języku FORTRAN IV. Czas pracy programu przy pełnym wykorzystaniu jego możliwości: ok.10-12 minut.

Ad.c. Wyznaczanie wartości optymalnych mocy baterii kondensatorów do kompensacji mocy biernej realizowane jest za pomocą następujących programów:

- #MØ48 w odniesieniu do sieci promieniowej,
- #MØ49 /w trakcie uruchomiania/ w odniesieniu do sieci rozgałęzionej.

Oba programy obliczają ponadto następujące wielkości energetyczne, związane z zagadnieniem optymalnej kompensacji mocy biernej:

- ogólną moc baterii kondensatorów w całej sieci energetycznej zakładu,
- wartość obciążeń czynnych i biernych w określonych punktach sieci przemysłowej,
- straty mocy czynnej w poszczególnych gałęziach oraz w całej

sieci w stanie naturalnym, rzeczywistym i optymalnym,
- oszczędność energii w wyniku zastosowania optymalnej kompensacji mocy biernej.

Dla każdego z programów występują dwa warianty procedur obliczeniowych sterowanych programem, uzależnione od miejsca usytuowania rozliczeniowego układu pomiarowego: po stronie górnego napięcia - wariant I i po stronie dolnego napięcia - wariant II.

Programy operują na danych wyliczonych przez program #LØ54 /moce czynne, moce bierne rzeczywiste, moce bierne naturalne/ oraz na danych wprowadzanych przez operatora z monitora systemowego: liczba promieni /lub gałęzi/, napięcie odniesienia U, współczynnik W, tangens ustalony, znamionowe straty jałowe transformatora sprzęgłowego, znamionowe straty obciążeń transformatora sprzęgłowego, moc znamionowa transformatora sprzęgłowego, prąd biegu jałowego transformatora sprzęgłowego, napięcie zwarcia transformatora sprzęgłowego, ogólna moc baterii kondensatorów, rozpatrywany okres czasu w godzinach, cena jednostkowa energii elektrycznej, rezystancja linii zasilających, straty jałowe transformatorów, moc znamionowa transformatorów, straty obciążeniowe transformatorów, prąd stanu jałowego transformatorów, przewodność właściwa ~~kkkk~~ materiału linii zasilającej, przekrój przewodu zasilającego.

Tabulogramy wyników zawierają następujące informacje:
- miejsce, czas i sposób wykonywanych pomiarów kompensacji mocy biernej w sieci promieniowej /ew. rozgałęzionej/,
- tabelę z wynikami pomiarów oraz parametrami transformatorów i linni zasilających,
- tabelę z wartościami mocy baterii kondensatorów, mocy czynnych w punkcie zasilania promienia /gałęzi/ oraz strat mocy czynnej,
- wartości wielkości charakterystycznych służących do wykazania efektów,
- efekty zastosowania optymalnej kompensacji.

Oba programy napisane są w języku FORTRAN IV i występują w nich odwołania do podprogramów z biblioteki obliczeń matematyczno-statystycznych. Czas obliczeń dla maksymalnej liczby

promieni /gałęzi/ przy pełnym wykorzystaniu możliwości programu wynosi ok. 3 minut.

Bliższych informacji udzielić mogą:

- w Ośrodku Energetyki Przemysłu Lekkiego IW
  Łódź. Al.Mickiewicza 24 tel. 36-31-73 :
  inż Henryk Jurczak
  mgr inż Wojciech Lewicki
  inż Tadeusz Piątkiewicz
  mgr inż Piotr Ruszkiewicz
- w Zespole d/s ETO IW
  Łódź, ul. Gdańska 91/93 tel. 33-96-00 wewn. 175 lub 214:
  mgr Jerzy Grall
  mgr Krystyna Białas
  mgr inż Jolanta Isalska

Mgr Bogumił Domalanus
Instytut Techniki Cieplnej
i Silników Spalinowych
Politechniki Poznańskiej

CROSS-ASEMBLER CAMASS M400 - ASEMBLER DO PROGRAMOWANIA PROCESORA AUTONOMICZNEGO 131 Z WYKORZYSTANIEM MERY-400

Asembler CAMASS ułatwia pisanie programów w języku symbolicznym procesora autonomicznego CAMAC 131. Asembler jest napisany w języku BASIC-EXT.

Przygotowanie programów dla zestawu sterowanego procesorem 131 jest zbliżone do przygotowywania programów w języku BASIC. Asembler przyjmuje zlecenia, których składnia jest identyczna jak w języku BASIC. A zatem jeżeli dysponujemy już zbiorem w postaci symbolicznej /przygotowanym asemblerem CAMASS/, który chcemy poprawić czy uzupełnić ściągamy program do pamięci za pomocą zlecenia:

OLD ⟨nazwa programu⟩

Ściągnięty do pamięci program możemy wydrukować na końcówce lub drukarce zleceniem:

LIST [⟨wskazanie we/wy⟩␣] [⟨lista numerów wierszy⟩]

⟨wskazanie we/wy⟩ : : = {# ⟨nr we/wy⟩}

⟨lista numerów wierszy⟩ : : = { -⟨nr wiersza⟩ / ⟨nr wiersza⟩[-⟨nr wiersza⟩] / [,⟨nr wiersza⟩[-⟨nr wiersza⟩]

Nowe wiersze programu wprowadzamy podając nr wiersza i rozkaz zgodnie z listą rozkazów procesora 131.

Wiersze można usuwać przy pomocy zlecenia:

REJ [⟨lista numerów wierszy⟩]

⟨lista numerów wierszy⟩ - identycznie jak w zleceniu LIST

Wydanie zlecenia bez listy numerów wierszy powoduje usunięcie całego programu.

Program zawarty w pamięci możemy umieścić na dysku przy pomocy zlecenia:

SAVE

Każdy program posiada swoją nazwę, którą możemy zmienić zleceniem:

PROG < nazwa programu >

Jeżeli nie wydamy zlecenia program będzie miał nazwę PROG. Program przenoszony na dysk zleceniem SAVE jest umieszczany w zbiorze o nazwie takiej jak nazwa programu.

Użytkownik asemblera C..SS ma do dyspozycji podobnie jak w języku BASIC zlecenie przenumerowywania wierszy:

RES

Umożliwia ono jednakże tylko standardowe przenumerowywanie od nr 100 co 10.

Asembler interpretuje ponadto następujące zlecenia, które są niezbędne przy pisaniu programów w języku symbolicznym oraz 2 dodatkowe zlecenia ułatwiające pracę; są to:

ADR < wartość >

< wartość > : : = jest to adres, od którego zostanie umieszczony program przy wprowadzaniu go przez loader. systemu CAMAC.

< nr wiersza > SET < wartość >

Zlecenie to a właściwie instrukcja spowoduje umieszczenie w odpowiedniej komórce pamięci CAMAC wartości podanej oktalnie parametrem < wartość >

NUM

zlecenie to powoduje zgłoszenie się asemblera znakiem ":" po którym należy podać liczbę dziesiętną lub oktalną /poprzedzoną znakiem %/ po czym na końcówce zostaną podane obydwie postacie tzn. dziesiętna i oktalna wprowadzonej liczby.

HELP

zlecenie powoduje wydruk na końcówce zleceń przyjmowanych przez asembler i opis ich działania.

Do zakończenia pracy w asemblerze służy zlecenie:

BYE

Jeżeli program jest gotowy do translacji wprowadzamy znaki /$ lub DC4 i w zbiorze o nazwie takiej jak nazwa programu + litera B będzie umieszczony program gotowy do wprowadzenia do pamięci kasety sterowanej procesorem 131 przy pomocy loadera systemu CAMAC.

Wiersze programu są interpretowane zgodnie z listą rozkazów procesora 131. Dla asemblera niezbędne jest podanie najpierw numeru wiersza a następnie kodu rozkazu. Pozostałe parametry niezależnie od kolejności ich podania są odpowiednio porządkowane i umieszczane w pamięci, chyba że ich wartości są niewłaściwe co jest sygnalizowane odpowiednim komunikatem.

Pracę wykonano w ramach Problemu Międzyresortowego MR.I.10 pt. "Optymalizacja procesów termodynamicznych i przepływowych" koordynowanego przez Instytut Techniki Cieplnej i Silników Spalinowych Politechniki Poznańskiej."

dr Wojciech Malinowski
Instytut FizykiMolekularnej PAN, Poznań

PRÓBA ZASTOSOWANIA MERY-400 DO TOMOGRAFII NMR

Tomografia jest to technika otrzymywania obrazów prezentujących przekrój badanego przedmiotu w danej płaszczyźnie , ze względu na pewną własność fizyczną. Dziedziną zastosowania tomografii jest diagnostyka medyczna. Tomografia NMR jest jedną z szeregu technik tomograficznych.

NMR (Nuclear Magnetic Resonance) - jądrowy rezonans magnetyczny jest jedną z technik spektroskopowych. Tomograf NMR w swej idei różni się od spektrometru rezonansu jądrowego jedynie dodaniem układu cewek gradientowych. Badanie rezonansu jądrowego w polu magnetycznym z liniowym gradientem pozwala rozróżnić i jednoznacznie przyporządkować linie rezonansowe różnym fragmentom badanego obiektu. Wykonanie pomiarów przy różnych kierunkach gradientu w danej płaszczyźnie pozwala na rekonstrukcję obrazu z projekcji.

Zadaniem komputera w tomografii jest realizacja czterech zasadniczych procesów : kontroli i sterowania układem pomiarowym, akwizycji danych, transformacji fourierowskiej danych wejściowych oraz rekonstrukcji obrazu z projekcji i jego wyświetlania. Przy odpowiednim doborze algorytmów przetwarzania wszystkie te procesy mogą być realizowane współbieżnie.

W projekcie budowy tomografu do realizacji procesów sterowania i przetwarzania wybrana została MERA-400 . Konfiguracja podstawowa z pamięcią 128K rozszerzona została o monitor graficzny o wielu poziomach szarości, preprocesor akwizycji danych, procesor pomiarowy MIEC-400 . Opracowywany jest procesor fourierowski (ITM Kraków) . Dla zapewnienia fizycznej współbieżności przetwarzania danych należałoby jeszcze uzupełnić podaną konfigurację o procesory macierzowy i obrazowy, co nie jest chwilowo przewidywane. Oprogramowynie przygotowywane jest pod systemem SOM-3 (wersja II UW) .

Praca nad konstrukcją tomografu NMR prowadzona jest w ramach problemu węzłowego PW 06.8 , planowe zakończenie prac w roku 1985 w formie modelu użytkowego.

V INFORMACJE OGÓLNE I OFERTY

Andrzej Braniecki
Przewodniczący Rady Programowej
Porozumienia Użytkowników
Minikomputera MERA-400

DZIAŁALNOŚĆ POROZUMIENIA UŻYTKOWNIKÓW MINIKOMPUTERA MERA - 400

WPROWADZENIE

I Konferencja Użytkowników MERY-400, która odbyła się w listopadzie 1983 r. wykazała potrzebę zorganizowania się użytkowników do wspólnych działań w celu poprawy eksploatacji i rozwoju minikomputera MERA-400.

Już na tej Konferencji przedstawiono i przedyskutowano wstępne propozycje form organizacyjnych dla takiego przedsięwzięcia. Komisja Wnioskowa, powołana w czasie Konferencji w porozumieniu z Zarządem Oddziału Gdańskiego PTC opracowała w kwietniu 1984 r. "Tymczasowy regulamin Porozumienia Użytkowników Minikomputera MERA-400".

Do najistotniejszych ustaleń tego regulaminu można zaliczyć:

1/ Porozumienie jest zespołem problemowym przy Oddziale Gdańskim PTC i zrzesza tzw. członków wspierających PTC /instytucje/. Przyjęcie członka do Porozumienia dobywa się na podstawie złożenia deklaracji.

2/ Cele Porozumienia są następujące:
- tworzenie warunków do podejmowania wspólnych działań uczestników Porozumienia w zakresie rozwoju sprzętu i oprogramowania oraz koordynacja tych działań;
- organizowanie wymiany doświadczeń oraz nowych rozwiązań sprzętowych i programowych;
- podejmowanie wspólnych działań dla zapewnienia lepszej obsługi serwisowej sprzętu;

- organizowanie w Gdańsku dorocznej Konferencji Użytkowników MERY-400 oraz innych ich spotkań;
- wydawanie materiałów informacyjnych w zakresie zainteresowań uczestników Porozumienia.

3/ Podstawą działania Porozumienia są uchwały Zebrań Plenarnych Porozumienia, a pracami Porozumienia kieruje Rada Programowa /wybierana na zebraniach plenarnych/.
Porozumienie posiada Sekretariat zorganizowany przez Zarząd Oddziału Gdańskiego PTC w uzgodnieniu z Radą Programową przy współpracy Instytutu Okrętowego PG.

4/ Fundusz Porozumienia tworzony jest ze składek członkowskich, których wysokość ustalana jest corocznie przez Zebranie Plenarne Porozumienia.

Tymczasowy regulamin został zatwierdzony jako obowiązujący na Zebraniu Plenarnym Porozumienia w czasie II Konferencji. Zebranie to ustaliło także składkę członkowską Porozumienia na rok 1985 w wysokości minimalnej 10000 zł.

Celem zabezpieczenia właściwego poziomu podejmowanych przez Porozumienie przedsięwzięć dotyczących oprogramowania systemowego i zastosowań informatyki Zebranie Plenarne Porozumienia wybrało na konsultantów naukowych Porozumienia prof. A.Salwickiego i doc. A. Kreczmara z Instytutu Informatyki Uniwersytetu Warszawskiego.

1. DOTYCHCZASOWA DZIAŁALNOŚĆ POROZUMIENIA

Porozumienie rozpoczęło działalność w lipcu 1984 r. tzn. w momencie, gdy zgodnie z "Regulaminem Porozumienia" swój akces zgłosiło 10 użytkowników minikomputera MERA-400. Do końca 1984 r. w ramach "Porozumienia" przeprowadzono:

- zorganizowanie w Gdańsku we wrześniu trzydniowej narady roboczej twórców systemów operacyjnych i kompilatorów języków programowania na minikomputer MERA-400,

- akcję informacyjną i zorganizowanie II Konferencji Użytkowników Minikomputera MERA-400 w Gdańsku w dniach 25-27 październik 1984 r.,
- działalność organizatorska: powołanie Sekretariatu "Porozumienia", przyjmowanie nowych członków "Porozumienia", opracowywanie wyników II Konferencji /przygotowanie materiałów pokonferencyjnych do wydawnictwa/, organizacja kilku spotkań Rady Programowej "Porozumienia".

Do końca 1984 r. akces do "Porozumienia" zgłosiły 58 instytucji.

2. PLANOWANA DZIAŁALNOŚĆ POROZUMIENIA

Do najważniejszych zadań ustalonych na Zebraniu Plenarnym Porozumienia w czasie II Konferencji i przewidzianych do realizacji w 1985 r. należą:

1/ Realizacja tzw. etapu zerowego B. Machowiaka polegającego na:
   - przeniesieniu kompilatora języka PASCAL opracowanego w Instytucie Informatyki UW pod system operacyjny CROOK-4,
   - przeniesienie kompilatora języka C opracowywanego w Instytucie Okrętowym PG na systemy SOM-3 podobne.

2/ Organizacja spotkań kierunkowych, seminariów problemowych i narad roboczych dotyczących:
   - zastosowań informatyki w poszczególnych typach instytucji i przedsiębiorstw /w celu decentralizacji obiegu informacji/;
   - narzędzi programowania takich jak systemy operacyjne, języki programowania, bazy danych;
   - możliwości sprzętowych.

3/ Współpraca przy organizacji szkoleń z zakresu narzędzi programowania.

4/ Inicjowanie i kierowanie opracowywaniem wstępnych koncepcji, przygotowywania założeń do planowanych prac, ekspertyz i opinii.

5/ Organizowanie grup problemowych działających na zasadzie sekcji w Porozumieniu /zrzeszających użytkowników o podobnych zastosowaniach minikomputera MERA-400/.

6/ Upowszechnianie informacji o sprzęcie i oprogramowaniu.

7/ Organizacja III Konferencji Użytkowników w październiku 1985 r. w Gdańsku.

Ważnym problemem wielokrotnie podnoszonym na II Konferencji jest uzyskanie kompatybilności oprogramowania systemowego MERY-400 ze standardami światowymi. Najłatwiejszą drogą do uzyskania tego efektu jest ściśle dostosowanie systemu operacyjnego CROOK-4 do standardu systemu operacyjnego UNIX oraz opracowanie dla MERY-400 kompilatora języka C.

Porozumienie będzie czyniło starania aby problem ten został rozwiązany. Podobnie, podejmowane będą starania o poprawę sytuacji odnośnie sprzętu i serwisu.

wrzesien 1984

OPROGRAMOWANIE MINIKOMPUTERA

MERA-400

POWSTALE W

INSTYTUCIE INFORMATYKI UNIWERSYTETU WARSZAWSKIEGO

NA BAZIE SYSTEMU OPERACYJNEGO

SOM-3

Oprogramowanie, ktoreso elementy zostana pokrotce przedstawione w ninieJszeJ ulotce, bazuJe na systemie operacyJnym SOM-3, a dokladnieJ na wersJi teso systemu zmodyfikowaneJ w Instytucie Informatyki Uniwersytetu Warszawskieso. Warto Jednak zdawac sobie sprawe, iz slowne frasmenty teso oprosramowania powstawaly w ramach osolneso projektu, ktoreso celem bylo przysotowanie noweso systemu operacyJneso dla minikomputera MERA-400. System ten, noszacy nazwe OOPS (ObJect Oriented Parallel System) znaJduJe sie aktualnie w stadium testowania. Ze wzsledu na fakt, iz system SOM-3 Jest obecnie (i prawdopodobnie bedzie przez dlusi czas) najbardzieJ rozpowszechnionym systemem operacyJnym maszyny MERA-400, zdecydowalismy sie na pewien wysilek w celu wydatneso uzdatnienia teso systemu dla potrzeb mozliwie wsodneJ i bezpiecznej wielodostepneJ pracy interakcyJneJ.

W dalszeJ czesci zakladac bedziemy, iz czytelnik zaznaJomiony Jest (byc moze pobieznie) z systemem SOM-3 w wersJi standardoweJ (ofarowaneJ przez producenta).

Jedna z najbardzieJ istotnych zmian w stosunku do standardoweso systemu SOM-3 Jest mozliwosc skorzystania z przestrzeni adresoweJ wynoszacej 64K slow dla Jedneso zadania. Pomimo pewnych osraniczen wymuszonych architektura maszyny, Jak na przyklad osraniczenie zasiesu instrukcJi bajtowych do dolnych 32K slow pamieci, uzyskuJe sie w ten sposob praktycznie podwoJenie rozmiarow pamieci operacyJneJ bezposrednio wykorzystywalneJ przez prosram uzytkowy. Oferowane sa Jednoczesnie srodki pozwalaJace dynamicznie rozszerzac badz redukowac przestrzen adresowa zadania, co w przypadku pracy wielodostepneJ pozwala elastycznie wykorzystywac pamiec operacyJna maszyny. IstnieJa takze narzedzia, szczesolnie przydatne w przypadku konfisuracJi o duzym zasobie pamieci operacyJneJ, umozliwiaJace wirtualizacJe przestrzeni adresoweJ zadania w ramach istnieJacej aktualnie puli wolnych sesmentow pamieci. Narzedzia te (dostepne takze z prosramow w Fortranie) umozliwiaJa dostep z pojedynczeso prosramu uzytkoweso do caleso wolneso obszaru pamieci operacyJneJ, wydatnie rozszerzaJac klase problemow rozwiazywalnych za pomoca minikomputera.

Istota wielodostepu osiasnieteso w zmodyfikowaneJ wersJi systemu SOM-3 polesa na przyznaniu kazdemu z zadan uzytkowych pewneJ (poczatkoweJ) puli pamieci operacyJneJ (minimalny rozmiar obszaru pamieci umozliwiaJacy zdefiniowanie zadania wynosi 4K slow) oraz wydzieloneso frasmentu pamieci dyskoweJ w postaci sekcJi roboczych.

Zadania uzytkowe pracuja w systemie Round Robin z cyklicznym podzialem czasu. Istnieja dwie klasy zadan, mianowicie zadania interakcyjne oraz zadania realizowane w tzw. tle (background). Podzial czasu odbywa sie w ramach kazdej z wymienionych klas, przy czym priorytet zadania tla jest zawsze nizszy od priorytetu zadania interakcyjnego. Uzytkownik posiada mozliwosc zmiany klasy zadania przy pomocy stosownej dyrektywy Job Control'a lub zadania komunikacji.

Przewidziana jest takze mozliwosc przekazywania komunikatow pomiedzy zadaniami, co np. umozliwia synchronizacje przy pracy wielu zadan o wspolnych zasobach. Przyjete rozwiazanie, kompatybilne z podsystemem wejscia-wyjscia, pozwala w szczesolnosci na wspolprace zadan sterowana przeplywem danych (tzw. pipeline).

Istnieje opcja systemu pozwalajaca w prosty sposob prowadzic rozliczanie uzytkownikow z globalnego czasu sesji. W wersji tej, na stalym pakiecie dysku systemowego znajduje sie centralna kartoteka uzytkownikow uprawnionych do korzystania z maszyny.

Na bazie zmodyfikowanego systemu SOM-3 pracuje Job-Control AJCL bedacy calkowicie nowym (w stosunku do standardowej wersji systemu) jezykiem sterowania zadaniem. Nowy Job-Control charakteryzuje sie przede wszystkim wygodna (zwarta i czytelna) forma dyrektyw, ktorych repertuar zostal wyraznie rozszerzony w stosunku do Job-Control'a w wersji firmowej. Znajduja sie tu miedzy innymi dyrektywy alokacji pamieci, zmiany klasy oraz caly szereg dyrektyw o charakterze informacyjnym, pozwalajacych uzytkownikowi zorientowac sie o aktualnych zasobach i parametrach zadania. Uzytkownik posiada takze mozliwosc praktycznie dowolnego kreowania sekcji dyskowych w ramach przydzielonej mu puli sekcji roboczych.

Naturalna dyrektywa wywolania programu (uzytkowego badz systemowego) pozwala na jednoczesny przydzial strumieni oraz ustawienie opcji a takze umozliwia przekazanie parametru tekstowego wywolywanemu programowi. Uzytkownik moze definiowac prywatne biblioteki programow standardowych przeszukiwane przed biblioteka systemowa LMB. Istnieje takze mozliwosc definiowania (w momencie generacji systemu) wiekszej liczby standardowych bibliotek systemowych przeszukiwanych automatycznie w momencie wywolania programu.

Nowy Job-Control dostarcza rowniez makroaparat pozwalajacy na naturalne definiowanie sparametryzowanych sekwencji dyrektyw identyfikowanych przez nazwy. Istnieje przy tym mozliwosc okreslania trojstopniowej hierarchii bibliotek-makrodyrektyw (poprzez FLS), w tym biblioteki globalnej - dostepnej dla wszystkich uzytkownikow.

Do zarzadzania zawartoscia dyskowych kaset wymiennych sluzy specjalny podsystem, stanowiacy w pewnym sensie rozszerzenie Job-Control'a AJCL, o nazwie FLS. Niektore dyrektywy Job-Control'a traktowane sa automatycznie jak odwolania do tego podsystemu i sluza do manipulowania odpowiednio sformatowana zawartoscia kasety wymiennej. Z logiczna struktura pakietu obslugiwanego przez FLS zwiazane jest pojecie jego wlasciciela oraz uzytkownikow. Wlasciciel, bedacy rowniez jednym z uzytkownikow pakietu, posiada pewne szczegolne uprawnienia niedostepne pozostalym uzytkownikom. Z kazdym uzytkownikiem zwiazana jest pula jego zbiorow (plikow) identyfikowanych poprzez (maksymalnie czternastoznakowe) nazwy. Uzytkownikowi dostarczone sa srodki pozwalajace na ochrone zbiorow przed przypadkowym zniszczeniem a takze przed ingerencja ze strony

innych uzytkownikow. W konfiguracjach wielodyskowych istnieje mozliwosc wymiany zbiorow badz calych kartotek pomiedzy pakietami (program MVF).

Zbiory obslugiwane przez FLS nie sa bezposrednio dostepne programom systemowym badz uzytkowym. W celu skorzystania z zawartosci takiego zbioru nalezy go najpierw sciasnac na jedna z sekcji roboczych. Posrod dyrektyw podsystemu znajduja sie dyrektywy sluzace do przechowywania zbiorow, pobierania ich zawartosci, usuwania zbiorow oraz informowania uzytkownika o zawartosci jego kartoteki. Istnieja takze dyrektywy pomocnicze sluzace na przyklad do inicjowania pakietu, wprowadzania badz usuwania uzytkownikow, itp. Ze wzgledu na zastosowanie specjalnych technik kopiowania (transmisje sciezkami) komunikacja z podsystemem jest szybka, nawet w przypadku przechowywania bardzo duzych zbiorow.

Istnieje mozliwosc zorganizowania kasety wymiennej w taki sposob, by jej wydzielona czesc dostepna byla w sposob tradycyjny - za posrednictwem sekcji dyskowych. W momencie inicjowania kasety dla takiego trybu pracy konieczne jest jedynie okreslenie globalnego obszaru przeznaczonego na sekcje. Dokladny podzial tego obszaru odbywac sie moze w sposob dynamiczny.

Rozpoczynajac prace z maszyna, uzytkownik posiadajacy kartoteke zbiorow na kasecie wymiennej podaje swoj (maksymalnie osmioznakowy) identyfikator oraz opcjonalne haslo. Uzytkownik posiada mozliwosc zdefiniowania pliku zawierajacego prywatna biblioteke makrodyrektyw Job-Control'a. Kolejnosc przeszukiwania bibliotek jest taka, ze najpierw przeszukiwana jest prywatna biblioteka danego uzytkownika, nastepnie ewentualna biblioteka wlasciciela pakietu i dopiero na koncu biblioteka systemowa. Specjalna dyrektywa o nazwie profile wywolywana jest (o ile istnieje) automatycznie przy wlaczaniu sie uzytkownika do systemu. Umozliwia to wstepne okreslenie zasobow uzytkownika (przydzial pamieci operacyjnej, utworzenie dodatkowych sekcji roboczych, itp.) zgodnie ze specyfika jego aktualnej pracy.

Podsystem FLS posiada zabezpieczenia na okolicznosc czesciowego zniszczenia zawartosci kasety wymiennej, np. w przypadku awarii dysku. Opis struktury zbiorow pakietu jest dublowany, przy czym kopia tego opisu aktualniana jest automatycznie po zakonczeniu pracy przez uzytkownika. Nalezy podkreslic, iz prawdopodobienstwo omylkowego zniszczenia zawartosci kasety na skutek bledu uzytkownika lub jego programu jest znikome.

Dla konfiguracji wyposazonej w stacje tasm magnetycznych atrakcyjna moze okazac sie mozliwosc rozszerzenia podsystemu FLS o podsystem MTS sluzacy do polautomatycznej archiwacji oraz skladowania na tasmach magnetycznych badz poszczegolnych zbiorow, badz tez calych zawartosci kaset. Skladowanie ma charakter przyrostowy, co oznacza iz skladowane sa jedynie te zbiory, ktorych zawartosc ulegla zmianie od czasu poprzedniego skladowania.

Innym elementem oferowanego oprogramowania, czesciowo związanym z podsystemem FLS jest spooler. Pozwala on drukowac zawartosci zbiorow bez angazowania w tym celu zadan uzytkowych. Spooler umozliwia jednoczesna obsluge wielu drukarek, przy czym wybor drukarki odbywa sie w sposob automatyczny (pierwsza wolna drukarka), badz tez zgodnie z preferencjami uzytkownika. Zgloszenia naplywajace do spooler'a są przezen kolejkowane i obslugiwane w kolejnosci naplywania. Kazdy

wydruk rozpoczyna sie specjalnym naglowkiem (banner) zawierajacym nazwe zbioru oraz czas i date zgloszenia go do wydruku. Drukarki obslugiwane przez spooler maja byc takze dostepne w sposob bezposredni. Maja one byc takze wykorzystywane przez spooler firmowy. Na czas trwania wydruku drukarka jest rezerwowana, co uniemozliwia odwolanie sie do niej z innego zadania.

Kolejnym elementem oferowanego oprogramowania jest edytor tekstowy EDM stanowiacy podstawowe narzedzie pracy przy opracowywaniu tekstow programow oraz danych. Podstawowe cechy tego programu, ktorego koncepcja wywodzi sie od koncepcji edytorow tekstowych ED oraz EX wystepujacych w systemie UNIX (znak handlowy zastrzezony przez Bell Labs Inc.), to bezposredni i blyskawiczny dostep do dowolnego fragmentu opracowywanego tekstu oraz silne, wyrafinowane dyrektywy pozwalajace przy minimalnej liczbie znakow wprowadzanych z klawiatury wykonywac zlozone modyfikacje zarowno na calym tekscie jak i na jego wybranych fragmentach. Zaawansowane metody wyszukiwan kontekstowych oraz kontekstowego wprowadzania poprawek pozwalaja na stosowanie EDM'u (choc zasadniczo nie jest on do tego przeznaczony) w sytuacjach wymagajacych zwykle uzycia makroprocesorow. Warto jednoczesnie zaznaczyc, iz elementarne srodki oferowane przez program sa mozliwe do natychmiastowego niemal opanowania nawet przez poczatkujacego uzytkownika.

Odmienna koncepcja opracowywania tekstow znajduje swoj wyraz w edytorze EDS zorientowanym na wspolprace z monitorem MERA-7910 dostepnym poprzez jednostke grupowa MERA-7905. Wykorzystujac specyficzne mozliwosci tego monitora pozwalajace na wyrywkowy dostep do poszczegolnych fragmentow ekranu EDS dostarcza srodkow na manipulowanie tekstem poprzez bezposrednia ingerencje w jego wyswietlane fragmenty. Rowniez i tym przypadku uzytkownik posiada mozliwosc bezposredniego odwolywania sie do odleglych kawalkow tekstu oraz kontekstowego wyszukiwania interesujacych fragmentow.

Kolejnym programem sluzacym do przetwarzania tekstow jest DOC, ktory sluzy do nadawania tesktom postaci wydawniczej. Dostepnosc drukarki wyposazonej w male litery wydatnie podnosi atrakcyjnosc programu. Wsrod prostych dyrektyw DOC'a znajduja sie srodki pozwalajace miedzy innymi wyrownywac marginesy, tworzyc akapity, numerowac strony, podkreslac oraz wytluszczac wybrane fragmenty, itp. W szczegolnosci, niniejszy dokument przygotowany zostal przy pomocy programu DOC.

Przedstawiane oprogramowanie obejmuje rowniez kompilatory dwoch jezykow programowania wysokiego poziomu. Jednym z tych jezykow jest LOGLAN-82 - stanowiacy uniwersalne narzedzie programowania przydatne zwlaszcza do zapisywania skomplikowanych algorytmow strukturalnych o charakterze nienumerycznym. Jezyk ten (o skladni nieco zblizonej do PASCAL'a) opracowany zostal w IIUW w ramach projektu, ktorego celem bylo zaprojektowanie i zaimplementowanie koncepcyjnie nowego uniwersalnego jezyka programowania dla perspektywicznych maszyn cyfrowych. Wsrod roznorodnych srodkow programotworczych dostarczonych przez LOGLAN znajduje sie bogaty zestaw narzedzi umozliwiajacych definiowanie zlozonych (takze hierarchicznych) typow danych (znacznie bardziej wyrafinowanych niz np. rekordy) oraz ulatwiajacych programowanie strukturalne. Na podkreslenie zasluguje duza dynamika LOGLAN'u, ktory umozliwia miedzy innymi dynamiczne organizowanie pamieci z programowana kontrola jej wykorzystania. Wszystkie obiekty strukturalne, w tym takze tablice, tworzone sa dynamicznie i maja byc

kasowane (z odzyskaniem zajmowanego przez nie obszaru pamieci) gdy nie sa juz potrzebne.

Drugim jezykiem, ktorego kompilator zostal rowniez wykonany w IIUW jest PASCAL. Biezaca wersja kompilatora implementuje wszystkie standardowe elementy tego powszechnie znanego jezyka programowania, ktorego przedstawiac nie trzeba.

Podstawowym jezykiem uzywanym w IIUW do przygotowywania oprogramowania systemowego jest asembler GASS. Asembler ten, zrealizowany wedlug calkowicie nowej koncepcji, rozni sie zasadniczo od asemblera MAC oferowanego przez producenta. Mnemoniczne kody instrukcji zostaly wydluzone (sa one rownej dlugosci) i pozwalaja na naturalne, latwe do utrwalenia skojarzenie ich z faktycznie wykonywana operacja. Skladnia tak zwanego drugiego argumentu instrukcji zostala ujednolicona. Oznacza to, iz stopien posrednioscl (bedacy w przypadku asemblera MAC atrybutem instrukcji oraz ostatniej litery jej kodu mnemonicznego) jest w GASS'ie jednoznacznie okreslony skladnia argumentu. Nowy asembler jest takze wyposazony w stosowny makroaparat oraz w mozliwosc warunkowej translacji wskazanych sekwencji kodu. Diagnostyka bledow generowana przez GASS (bledne linie zawsze pojawiaja sie na listingu) oraz wydruk posortowanej tablicy symboli ulatwiaja uruchamianie programow. GASS pozwala rowniez na automatyczne optymalizowanie kodu polegajace na zastepowaniu dlugich (dwa slowa) instrukcji ich krotkimi odpowiednikami, tam gdzie jest to mozliwe.

Format programow binarnych generowanych przez GASS jest calkowicie odmienny od standardowego formatu binariow dla systemu SOM-3 (docelowo GASS jest przeznaczony do pracy w systemie OOPS). Razem z asemblerem dostarczany jest program BFC dokonujacy konwersji binariow GASS'a na format akceptowalny przez SOM-3. Innym programem, rowniez wchodzacym w sklad wyposazenia nowego asemblera, jest FMS sluzacy do formatowania tekstow zrodlowych w GASS'ie.

Podczas eksploatacji systemu SOM-3 w IIUW do systemu tego wprowadzono caly szereg mniej lub bardziej drobnych modyfikacji i poprawek, ktore, niezaleznie od przedstawionych powyzej elementow oprogramowania, wyraznie poprawiaja komfort pracy pod tym systemem. W szczegolnosci poprawiono wiele handlerow niejednokrotnie rozszerzajac ich funkcje oraz lagodzac ich rozmaite wady. Jedna z globalnych zmian dotyczacych wszystkich handlerow urzadzen znakowych jest zrezygnowanie z zaokraglania przesylanych sekwencji do parzystej liczby znakow. Poza ta roznica, na ogol akceptowalna przez programy uruchamiane pod standardowa wersja systemu, zachodzi pelna kompatybilnosc z systemem standardowym (oczywiscie w zakresie funkcji oferowanych przez ten system). Podsystem wejscia-wyjscia zostal rowniez nieco usprawniony, co przejawia sie miedzy innymi w mozliwosci bezpiecznej wymiany kaset dyskowych w trakcie dzialania programu. Liczba bledow typu OFF LINE ... pojawiajacych sie podczas normalnej pracy zostala wydatnie zredukowana, zas koniecznosc ustawicznego uzywania dyrektywy ONL zadania komunikacji zostala praktycznie wyeliminowana.

W IIUW opracowano rowniez interfejs pozwalajacy podlaczyc do minikomputera MERA-400 pewne niestandardowe urzadzenia. Do urzadzen tych naleza:

- Pisak analogowy (np. wesierski NE-240) o rozdzielczosci 1024*1024 punktow plaszczyzny,

- Monitor graficzny o bardzo duzej szybkosci (interfejs pamieciowy) wykonany na bazie odbiornika telewizyjnego, o rozdzielczosci 256*256 punktow,

- Czytnik kart ARITMA EC-6112 produkcji czeskiej.

W sklad systemu operacyjnego wchodza handlery powyzszych urzadzen, a takze (w przypadku pisaka i monitora graficznego) podprogramy standardowe pozwalajace na bezposrednie wykorzystanie ich z programow Fortranowych.

Dysponujemy rowniez pelnym programowym interfejsem do dyskow 30Mb (dotyczy to handlera oraz podsystemu FLS/MTS). W razie pojawienia sie na rynku oczekiwanego interfejsu fizycznego do tych dyskow (prototyp funkcjonuje w IIUW) jestesmy w stanie zapewnic ich natychmiastowe skonfigurowanie.

Wymagania odnosnie konfiguracji minikomputera akceptujacej zmodyfikowana wersje systemu SOM-3 zalezą od liczby zadan (uzytkownikow) pracujacych jednoczesnie oraz od repertuaru urzadzen obslugiwanych przez system. Przy malej liczbie typowych urzadzen (KSR/DZM-180, monitory 7952, czytnik tasmy, perforator) oraz niewielkiej liczbie zadan (1-2) system operacyjny miesci sie w 12K slow pamieci operacyjnej. W przypadku powiekszenia liczby zadan, lub koniecznosci obslugiwania wiekszej liczby bardziej roznorodnych urzadzen obszar pamieci systemu nalezy powiekszyc do 16K lub nawet (zwlaszcza w przypadku wystepowania w konfiguracji jednostki grupowej MERA-7905) do 20K slow. Kazde zadanie uzytkowe powinno standardowo dysponowac pula pamieci operacyjnej o rozmiarze 20K slow, wystarczajaca dla swobodnego korzystania z prawie wszystkich programow systemowych. W przypadku bardziej specyficznych zastosowan rozmiar ten mozna zmniejszyc nawet do 4K slow. Istnieje dyrektywa Job-Control'a pozwalajaca w sposob bezpieczny wymieniac pamiec miedzy zadaniami.

Ze wzgledu na fakt, iz kazde zadanie uzytkowe otrzymuje do wylacznej dyspozycji prywatna pule pamieci dyskowej wskazane jest, aby w przypadku wiekszej liczby zadan dysponowac wiecej niz jedna stacja dyskow. W praktyce okazuje sie, ze maksymalna sensowna liczba uzytkownikow dla konfiguracji jednodyskowej jest dwa.

Mozliwe jest przygotowanie (i zainstalowanie) kilku wersji systemu dla pojedynczej konfiguracji minikomputera. Sciagajac system (bootstrap) nalezy wowczas ustawic na kluczach procesora stosowny numer wersji.

Zamowienia na oprogramowanie lub uslugi prosimy skladac na adres:

Dom Handlowy Nauki, Sp z o.o.
ul. Miodowa 2
00-080 Warszawa
"KOMPEKS"

mgr Czesław Kamzol
Zakłady Systemów Automatyki
"MERAMONT"
Zakład Informatyki i Obsługi Technicznej
ul.Armii Czerwonej 66/72
61-807 POZNAŃ
tel. 67-43-19 lub 699-151 w. 144

INFORMACJA O STANIE SERWISU SYSTEMU MERY-400

W dniu 2.11.1983 r pomiędzy Fabryką Mierników i Komputerów "ERA" w Warszawie a Zakładami Systemów Automatyki "MERAMONT" w Poznaniu, została zawarta umowa na prowadzenie obsługi serwisowej systemu "MERA-400". W myśl umowy obsługa serwisowa dotyczy napraw gwarancyjnych i pogwarancyjnych urządzeń wchodzących w skład systemu MERA-400 oraz uruchamienia urządzeń stanowiących rozszerzenie posiadanego systemu.

Umowny rejon obsługi serwisowej obejmuje następujące województwa: szczecińskie, koszalińskie, słupskie, gdańskie, elbląskie, gorzowskie, pilskie, bydgoskie, toruńskie, włocławskie, poznańskie, konińskie, zielonogórskie, leszczyńskie, kaliskie, legnickie i wrocławskie.

Realizację umowy serwisowej rozpoczęliśmy od 1.7.84r o czym użytkownicy zostali pisemnie powiadomieni przez producenta.

W okresie 4-miesięcznej obsługi przyjęto wiele zgłoszeń awarii urządzeń, najczęściej dotyczących pamięci dyskowych i ich współpracy z jednostką centralną oraz drukarek DZM 180. Odnośnie DZM 180 prowadzimy w pełnym zakresie naprawy podzespołów elektronicznych / płyty: logiki bufora, przekaźników DC, DL, transportu, zasilacza/, natomiast w ograniczonym zakresie mechanizmu, z uwagi na

brak części zamiennych. Niestety,ze względu na niedostateczne zaopatrzenie w pakiety i części zamienne, wiele z tych zgłoszeń, a przede wszystkim dotyczących awarii pamięci dyskowych i jednostki centralnej, nie realizowaliśmy i przekazywaliśmy do serwisu producenta.

Jakość i skuteczność obsługi serwisowej zawsze wiązaliśmy z właściwym wyposażeniem w niezbędne części zamienne i dlatego też w naszym wystąpieniu do Dyrektora Fabryki Mierników i Komputerów "ERA" inż.Wojciecha Mikulskiego, ten element obsługi serwisowej został szczególnie podkreślony.
Mamy jednak nadzieję, że problem zaopatrzenia w części zamienne zostanie pozytywnie rozwiązany przez producenta,co w naszym przekonaniu przyczyni się w znacznej mierze do poprawy jakości świadczonych przez nas usług serwisowych.

mgr Bronisław Maciuk
Politechnika Śląska
Katedra Organizacji Produkcji
Wydziału Metalurgicznego

BDMB - WIELODOSTĘPNY PROGRAM ZAKŁADANIA I AKTUALIZACJI KARTOTEKOWYCH ZBIORÓW NA STACJACH DYSKOWYCH MERA 9425.

Program BDMB przyjmując narzuconą przez system SOM3FMC organizację pamięci dyskowej MERA 9425,umożliwia tworzenie w sekcjach dyskowych,własnych bibliotek słownikowych kartotekowych zbiorów danych.Dane te mogą być przetwarzane przez użytkowników bezpośrednich z oddalonych końcówek,którymi mogą być monitory ekranowe MERA 7952 lub urządzenia DZM-KSR 180.Użytkownicy bezpośredni korzystają ze wszystkich sposobów przetwarzania danych tj.wyszukiwania,aktualizowania i usuwania danych.Program BDMB w trybie wielodostępnym dopuszcza równoczesną pracę 4 użytkowników bezpośrednich. Każdemu z użytkowników zostaje przydzielony,przez operatora systemu,oddzielny /może być różnej wielkości /obszar pamięci operacyjnej.Przydział takich zasobów zadania dla poszczególnych użytkowników jak:drukarka,czytnik tasiemki papierowej, sekcje robocze dysku stałego,dokonuje program BDMB w chwili zgłoszenia takich potrzeb.

Kartotekowe zbiory danych mogą zawierać do 32 768 dokumentów.Pola dokumentów mogą być grupowane w oddzielnych rekordach,które pamiętane są w oddzielnych plikach,co umożliwia tworzenie kartotek wieloindeksowych.Pliki będące indeksami mogą być sortowane.Dla szybkiego wyszukiwania rekordów wg.numerów dokumentów w posortowanych plikach indeksowych,mogą zostać użyte tablice inwersyjne.Efektywne wykorzystanie pamięci dyskowej umożliwia zastosowanie oprócz znakowego zapisu danych takich zapisów jak:binarny,CAN-kod, dziesiętny/4 cyfry w 1 słowie/ oraz formatów specjalnych w takich przypadkach jak konto bankowe czy data.

Sterowanie pracą programu BDMB realizowane jest przy

pomocy dyrektyw podawanych z klawiatury monitora operatora systemu, oraz z klawiatur końcówek użytkowników bezpośrednich. Przetwarzanie danych realizowane jest przy pomocy takich dyrektyw jak:

GET - wyszukiwanie danych wg. dowolnie skonstruowanego klucza, sporządzanie selektywnych zestawień danych z wyjściem na monitor lub drukarkę

DEL - usuwanie dokumentów

REP - aktualizacja wybranych pól dokumentów

INS - wprowadzanie dokumentów

SORT - sortowanie plików.

Ponadto wprowadzono takie dyrektywy jak: kopiowania zbiorów, usuwania zbiorów, oraz kompresji bibliotek.

dr inż. Zbigniew Mrozek
Instytut Elektrotechniki i Elektroniki
Politechnika Krakowska im. T. Kościuszki
31-155 Kraków, ul. Warszawska 24.

DIAGRA - FORTRANOWSKI PODPROGRAM RYSUJACY WYKRESY

Opracowano krótki i bardzo efektywny podprogram rysujący wykresy na drukarce wierszowej lub mozaikowej. Uzyskuje się maksymalnie do 20 wykresów - każdy rysowany innym znakiem. Każda krzywa jest automatycznie skalowana, a następnie optymalnie rozmieszczona na powierzchni papieru.

Współrzędne punktów do rysowania są zadawane w formie macierzy. Jej wymiary są deklarowane w programie głównym. Nie ma żadnych ograniczeń ilości rysowanych punktów. Oznacza to, że w zależności od potrzeby, uzyskuje się wykres o długości na przykład pół strony lub kilkanaście stron.

Poniżej przedstawiono wydruk omawianego podprogramu, wraz z programem testującym RTEST. Program generuje wartości zmiennej niezależnej x oraz tablicę YY, zawierającą wartości trzech funkcji: $\exp(x)$, $x \cdot \cos(x)$, $\sin(x)/(1+x)$. Następnie podany jest wydruk podprogramu rysującego DIAGRA oraz wykresy podanych wyżej funkcji, uzyskane na minikomputerze MERA 400 z drukarką DZM 180. Program przygotowano w fortranie standardowym. Może być on uruchomiony bez przeróbek na innych komputerach.

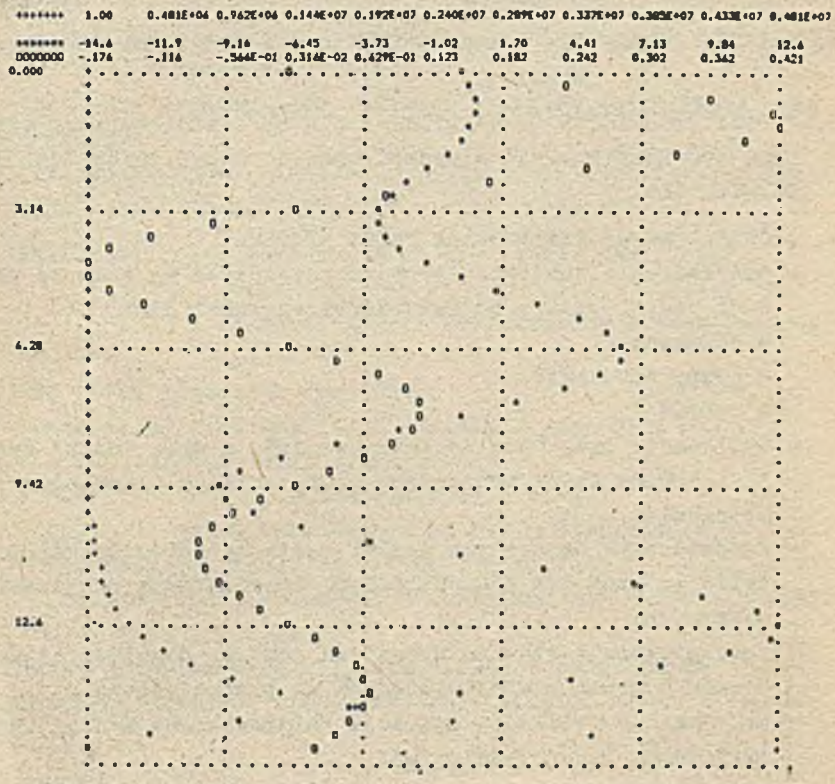
```
FORTRAN IV-S                        25. 04. 1984          STRONA  1

    1     1          PROGRAM RTEST
    2     2          REAL X(50),YY(3,50)
    3     3          DX=.314

    4     4          XX=0.
    5     5          DO 1 I=1,30
    6     6          YY(1,I)=EXP(XX)
    7     7          YY(2,I)=COS(XX)*XX
    8     8          YY(3,I)=SIN(XX)/(XX+1.)
    9     9          X(I)=XX
   10    10    1     XX=XX+DX
   11    11          CALL DIAGRA(X,YY,3,50,2)
   12    12          STOP 1
   13    13          END
```

```
FORTRAN IV-S                    25. 04. 1984            STRONA   1

   1        1          SUBROUTINE DIAGRA  (X,YY,IY,NX,LR)
   2                C     PROGRAMOWAL  ZBIGNIEW MROZEK - KRAKOW 1974
   3                C     WERYFIKACJA - JAN WEREWKA, ZBIGNIEW MROZEK - V2.0/84

   4                C        PODPROGRAM ROBI WYKRES DO DWUDZIESTU FUNKCJI ZADANYCH W TABELI
   5                C        AUTOMATYCZNE SKALOWANIE WYKRESU DLA KAZDEJ FUNKCJI
   6                C        PARAMETRY FORMALNE:
   7                C             X=X(NX)  TABLICA DO OPISANIA SKALI NA OSI ARGUMENTU
   8                C             YY=YY(IY,NX)  TABLICA WARTOSCI WYKRESLANYCH FUNKCJI
   9                C             IY -ILOSC WYKRESLANYCH FUNKCJI
  10                C             NX  ILOSC PUNKTOW WYKRESU DLA KAZDEJ FUNKCJI
  11                C             LR -NUMER PROGRAMOWY DRUKARKI WIERSZOWEJ
  12        2            REAL DS(11),SY(20),YMIN(20),X(NX),YY(IY,NX),Y(20)
  13        3            INTEGER ER(20),'104),WN,B
  14        4            LOGICAL HGRID
  15        5            DATA WN/1H./,NUL/1H /
  16        6            DATA ER(1),ER(2),ER(3) /1H+,1H*,1HO/
  17        7            DATA ER(4),ER(5),ER(6) /1H!,1HX,1H$/
  18                C   TU ZADAC DALSZE ZNAKI DO RYSOWANIA
  19                C          SKALOWANIE
  20        8            WRITE(LR,5)
  21        9            DO 6 J=1,IY
  22       10            AMAX=YY(J,1)
  23       11            AMIN=AMAX
  24       12            DO 7 JX=2,NX
  25       13            IF(AMAX.LT.YY(J,JX)) AMAX=YY(J,JX)
  26       14    7       IF(AMIN.GT.YY(J,JX)) AMIN=YY(J,JX)
  27       15            HGRID=AMIN.EQ.AMAX
  28       16            IF(HGRID) AMAX=AMAX+J
  29       17            IF(HGRID)AMIN=AMIN-1.
  30       18            D=(AMAX-AMIN)*.1
  31       19            DS(1)=AMIN
  32       20            DO 8 ID=2,11
  33       21    8       DS(ID)=DS(ID-1)+D
  34       22            B=ER(J)
  35       23            WRITE(LR,4)B,B,B,B,B,B,B,D,(DS(I),I=1,11)
  36       24            SY(J)=10./D
  37       25    6       YMIN(J)=AMIN
  38                C GENERACJA SIATKI UKLADU WSPOLRZEDNYCH
  39       26            DO 10 IX=1,NX
  40       27            DO 9 I=1,104
  41       28    9       R(I)=NUL
  42       29            HGRID=MOD(IX,10).EQ.1
  43       30            IF(HGRID) GO TO 12
  44       31            DO 11 KN=1,11
  45       32            IR=1+(KN-1)*20
  46       33   11       R(IR)=WN
  47       34            GO TO 15
  48       35   12       CONTINUE
  49       36            DO 14 KN=1,51
  50       37            IR=1+(KN-1)*2
  51       38   14       R(IR)=WN
  52       39   15       CONTINUE
  53                C GENERACJA WYKRESOW FUNKCJI
  54       40            DO 13  L=1, IY
  55       41            DYY=(YY(L,IX)-YMIN(L))*SY(L)
  56       42            DYYA=DYY-AINT(DYY)
  57       43            IF(DYYA.GE..5) IE=INT(DYY)+2
  58       44            IF(DYYA.LT..5) IE=INT(DYY)+1
  59       45            R(IE)=ER(L)
  60       46   13       CONTINUE
```

```
   61      47          IF(.NOT.HGRID)WRITE (LR,2) (R(I),I=1,104)
   62      48          IF(HGRID) WRITE(LR,3)X(IX),(R(I),I=1,104)
   63      49  10      CONTINUE

   64      50          WRITE(LR,1)
   65      51          RETURN
   66      52    1     FORMAT(13X,52(2H. )///
   67                 *1H /,10X,40HWYKRES WYKONANY PRZEZ SUBROUTINE DIAGRA //)
   68      53    2     FORMAT(13X,104A1)
   69      54    3     FORMAT(1X,G10.3,2X,104A1)
   70      55    4     FORMAT(3X,7A1,2X,11(G9.3,1X));
   71      56    5     FORMAT(///)
   72      57          END
```

INTEGER	NAME	SPECIFICATION	LENGTH
	R	ARRAY	104
	B	VARIABLE	1
	IX	VARIABLE	1
	KN	VARIABLE	1
	JX	VARIABLE	1
	IT	D VARIABLE	1
	NUL	I VARIABLE	1
	IR	VARIABLE	1
	I	VARIABLE	1
	WN	I VARIABLE	1
	J	VARIABLE	1
	ID	VARIABLE	1
	NX	D VARIABLE	1
	LR	D VARIABLE	1
	L	VARIABLE	1
	IE	VARIABLE	1
	ER	I ARRAY	20
REAL	NAME	SPECIFICATION	LENGTH
	D	VARIABLE	3
	YY	D ARRAY	
	DYYA	VARIABLE	3
	YMIN	ARRAY	60
	AMIN	VARIABLE	3
	X	D ARRAY	
	SY	ARRAY	60
	Y	ARRAY	60
	DYY	VARIABLE	3
	AMAX	VARIABLE	3
	DS	ARRAY	33
LOGICAL	NAME	SPECIFICATION	LENGTH
	HGRID	VARIABLE	1

```
0 NOLO NOMA NOU0 NOU1 NOU2 NOU3 NOU4 NOU5 NOU6
```



WYKRES WYKONANY PRZEZ SUBROUTINE DIAGRA

SYSTEM EWIDENCJI I ROZLICZEŃ FINANSOWYCH
"SERF"
Karta informacyjna systemu

JEDNOSTKA AUTORSKA: OŚRODEK INFORMATYKI CZSP WARSZAWA

1. DZIEDZINA: gospodarka finansowa przedsiębiorstwa
   NAZWA SYSTEMU: System Ewidencji i Rozliczeń Finansowych - - "SERF".

2. PRZEZNACZENIE: dla przedsiębiorstw i spółdzielni dowolnych branż.

3. ŚRODKI TECHNICZNE DO REALIZACJI SYSTEMU: minikomputer serii MERA 400 konfiguracji standard oraz do 8-monitorów 7952 lub DZM-KSR 180.

4. GŁÓWNE FUNKCJE REALIZOWANE PRZEZ SYSTEM:
   - sporządzanie zestawień dowodów księgowych wg grup ze wszystkimi danymi oraz z podsumowaniem wartości dla:
     - dokumentu
     - grupy dokumentów
     - ogółem
   - wyliczanie tzw. "sum do księgowania", a więc rodzaj dekretu i ogólna kwota wynikająca ze wszystkich dokumentów źródłowych
   - sporządzanie skróconych wersji zestawień dowodów księgowych tylko wg grup lub poszczególnych dokumentów z zachowaniem podsumowań
   - sporządzanie zestawień wybranych kont wg dowolnej ilości znaków symbolu konta, służących do przeksięgowań
   - prowadzenie ewidencji operacji księgowych na kontach analitycznych z uwzględnieniem:
     - identyfikacji konta
     - numeracji kolejnej i zewnętrznej dokumentów
     - treści operacji
     - konta przeciwstawnego
     - nr pozycji w zestawieniu dowodów
     - obrotów i sald za okres: ubiegły, obliczeniowy, narastająco

- sporządzanie zestawień obrotów i sald kont analitycznych z tworzeniem sum obrotów i sald po 3,5 oraz "x" znakach symbolu konta z możliwością wybrania kont
- sporządzanie zestawień obrotów i sald kont syntetycznych /z badaniem czy suma kont analitycznych w zbiorach równa się sumie konta syntetycznego/
- automatyczne tworzenie rekordów transakcyjnych dla przeksięgowań kosztów, rozr chunków itp wg "indeksu przeksięgowań" przygotowanego przez konkretnego użytkownika oraz zaksięgowanie tych rekordów na kontach
- tworzenie archiwalnego zbioru dokumentów dotyczących rozrachunków
- emitowanie narastająco od początku roku rozliczeń dotyczących kont "rozrachunkowych" a więc grupy "2".

5. OGRANICZENIA: - symbol konta analitycznego max 12 zn. /w ramach grupy/ np. 00,01,03 itp./ jednakowej długości symbol poszczególnych kont analitycznych/
   - dla dwóch jednostek dyskowych:
     - max ilość kont analit. i syntetycznych 18800 w tym max kont syntetycznych 256
     - max ilość rekordów transakcyjnych 29000
     - oraz 200 cylindrowa sekcja na dysku stałym na sortowanie rekordów zb. tran. do wydruku kartoteki /długość sekcji roboczej w sektotach dla wydruku kartoteki = 0,14 x il.rek.TRR.

   Schemat przybliżonego obliczania ilości rekordów zb. TRR na wyznaczonej sekcji: il.rek.zb.TRR = długość sekcji w sektorach x 6.

6. DOKUMENTY WEJŚCIA:
   - polecenie księgowania
   - raport kasowy
   - lista płac
   - wyciąg bankowy
   - faktura własna
   - faktura obca

- rachunek
- inkaso
- nota bankowa

7. DOKUMENTY WYJŚCIA:
   - Wykaz otwartych kont analitycznych i syntetycznych
   - Bilans otwarcia kont analitycznych
   - Bilans otwarcia kont syntetycznych
   - Tabulogram otwarcia kont analitycznych
   - Tabulogram otwarcia kont syntetycznych
   - Zestawienie dowodów księgowych wg grup
   - Zestawienie dowodów księgowych
   - Zestawienie grup dowodów księgowych
   - Zestawienie błędnych dowodów księgowych
   - Zestawienie obrotów i sald wg "x" znaków symbolu konta
   - Tabulogram ewidencji finansowej
   - Zestawienie obrotów i sald kont analitycznych
   - Zestawienie obrotów i sald kont syntetycznych
   - Analiza kont za okres ..... do ....

INNE UWAGI:

1. JEDNOSTKA KONSERWUJĄCA SYSTEM: Ośrodek Informatyki CZSP, 00-213 Warszawa, ul. Bonifraterska 14. tel. 31-41-63 lub 31-68-72.

2. POSIADANA DOKUMENTACJA DLA UŻYTKOWNIKA:
   - Instrukcja eksploatacji systemu dla użytkownika
   - Dokumentacja eksploatacyjna dla operatora

3. ORIENTACYJNY KOSZT UDOSTĘPNIENIA SYSTEMU:
   - upowszechnienie ok. 300 tys.zł
   - modyfikacja systemu dla potrzeb użytkownika wg kosztów rzeczywistych
   - wdrożenie systemu u użytkownika wg kosztów rzeczywistych.

SYSTEM EWIDENCJI I URZĄDZEŃ PRZEDMIOTÓW NIETRWAŁYCH
W UŻYTKOWANIU - "PNU"
Karta informacyjna systemu

JEDNOSTKA AUTORSKA: OSRODEK INFORMATYKI CZSP WARSZAWA

1. DZIEDZINA: przedmiot nietrwały w użytkowaniu
NAZWA SYSTEMU: System ewidencji i umorzeń przedmiotów nietrwałych w użytkowaniu - "PNU"

2. PRZEZNACZENIE: dla przedsiębiorstw i spółdzielni dowolnych branż.

3. ŚRODKI TECHNICZNE DO REALIZACJI SYSTEMU: minikomputer serii MERA 400 konfiguracji standard oraz do 5-ciu monitorów 7952 lub DZM-KSR 180

4. GŁÓWNE FUNKCJE REALIZOWANE PRZEZ SYSTEM:
- założenie i aktualizowanie wykazu indeksów pnu
- ewidencja ilościowo-wartościowa przedmiotów nietrwałych w użytkowaniu w podziale na miejsce użytkowania
- emisja dokumentów transakcyjnych wg ich rodzajów
- wyliczanie wartości umorzeń przedmiotów nietrwałych w użytkowaniu
- sporządzanie zestawień obrotów i sald kont 351 i 361 z podziałem na miejsca użytkowania
- sporządzanie zestawień pnu wg stawek umorzeniowych /z wyszczególnieniem wartości umorzeń w cyklu, narastająco i wartości pozostałych do umorzenia/ oraz wg miejsca powstawania kosztów
- sporządzanie zestawień pnu, których okres używalności minął, z podziałem na miejsca użytkowania
- rozliczenie inwentaryzacji z aktualizacją stanów, w miejscach użytkowania wyliczonymi różnicami
- wyliczenie wartości umorzeń pnu przeszacowanych do 100 % stopy umorzeniowej

5. OGRANICZENIA:
- indeks "A" - max 11 znaków
"B" - max 20 znaków

- liczba symboli indeksów pnu - 10.000
- liczba kart ewid.ilościowo-wart.- 13.000
- liczba dokuemntów transakcyj. - 20.000
- liczba miejsc użytkowania - 999
- liczba stanowisk kosztów - 999

BUDOWA INDEKSÓW:

"A" 99 - 999999999
- grupa klasyfikacyj
- numer inwentarzowy

"B" 99 - 9999 - 999999,99 - 99,99
- grupa klasyfikacyjn
- 4 znaki SWW
- cena zakupu pnu
- m-c, rok wydania d użytkowania

6. DOKUMENTY WEJŚCIA:
kartoteka pnu - KT, dokumenty obrotu Rw.MN,LN,Wz,PK, korekty tych dokumentów SR,SM,SL,SW,SP oraz arkusz spisu z natury - I

7. DOKUMENTY WYJŚCIA:
- Wykaz KT przedmiotów nietrwałych w użytkowaniu na dzień wg miejsc użytkowania
- wykaz pnu wg indeksu
- wykaz dokumentów transakcyjnych wg indeksów i miejsc użytkowania
- dokumenty transakcyjne błędne
- dokumenty transakcyjne wg rodzajów dokumentów
- ewidencja ilościowo-wartościowa pnu wg miejsc użytkowania
- zestawienie obrotów i sald wartości pnu oraz wartości umorzeń wg miejsc użytkowania
- zestawienie pnu ze 100 % stawką umorzeniową wg st. kosztów
- zestawienie pnu umorzonych okresowo w podziale na stanowiska kosztów
- zestawienie pnu, których okres użytkowania minął wg miejsc użytkowania
- wykaz dokumentów KS
- wykaz różnic inwentaryzacyjnych
- wykaz pnu przeszacowanych do 100 % stopy umorzeniowej

NE UWAGI:

1. JEDNOSTKA KONSERWUJĄCA I DOSKONALAJĄCA SYSTEM:
Ośrodek Informatyki CZSP, 00-213 Warszawa, ul. Bonifraterska 14, tel. 31-41-63 lub 31-68-72.

2. POSIADANA DOKUMENTACJA DLA UŻYTKOWNIKA:
- instrukcja eksploatacji systemu dla użytkownika
- dokumentacja eksploatacyjna dla operatora

3. ORIENTACYJNE KOSZTY UDOSTĘPNIENIA SYSTEMU:
- upowszechnienie ok. 300 tysięcy
- modyfikacja systemu dla potrzeb użytkownika wg kosztów rzeczywistych
- wdrożenie systemu u użytkownika wg kosztów rzeczywistych.

mgr inż. Jerzy Nagórski
Zakład Elektroniki Komputerowej
"ZEKOM" ul. Makowa 8. Łódź.

OFERTA FIRMY "ZEKOM"

Zakład Elektroniki Komputerowej "ZEKOM" jest firmą prywatną specjalizującą się w produkcji monitorów i terminali komputerowych typu ekranowego oraz urządzeń i wyposażenia wspomagającego. Wszystkie aktualnie produkowane monitory mają pojemność ekranu 1024 znaki /w 16 wierszach po 64 znaki/ w matrycy 5 x 7, ekran 12" może być z luminoforem białym lub zielonym P31.
Produkujemy kilka typów monitorów, każdy w kilku odmianach po to, by spełniać wielorakie wymagania i oczekiwania użytkowników dotyczące ceny, dodatkowych funkcji, szybkości działania, rodzaju obudowy, miejsca zainstalowania.
Oferujemy następujące typy urządzeń:
MZ 1664

- monitor ekranowy odbiorczy,
- interfejs równoległy o strukturze analogicznej jak w interfejsie drukarki DZM 180 /7 bitowa magistrala danych wejściowych + 2 linie sterujące, hand-shaking/,
- kod ASCII /2 zestawy po 64 znaki/,
- szybkość pracy 60 znaków/s,
- kineskop A31-310W /czarno-biały/,
- obudowa telewizora VELA 203,
- typowe zastosowanie: niezależna współpraca z drukarką DZM 180,
- niska cena,
- wyposażenie dodatkowe:
  - sprzęgi serii SM do podłączenia monitora do:
    - drukarki DZM 180, DZM 180/325, monitora technicznego DC 7086 /konsola operatora systemu RIAD 32/ - typ SM 11,
    - terminala DZM 180KSR, monitorów technicznych DZM 180/05 i DZM 180/25 /konsole operatora systemów ODRA 1305 i 1325/ typ SM 12,
    - innych systemów np. L x 2500 /LOGABAX/, P6060 /OLIVETTI/, HP9830 /HEWLETT-PACKARD/,
  - uchwyt UN-1 mocujący obrotowo monitor do drukarki DZM 180,
  - uchwyt U-21 lub podstawa UP-1 w przypadku użycia monitora jako urządzenia wolnostojącego,

- przystawka funkcyjna UF-12 do sterowania stronicowaniem.

1664-RO

- terminal odbiorczy przeznaczony do pracy w systemach szeregowej asynchronicznej transmisji danych,
- interfejs szeregowy wyłącznie z obsługą linii R x D:
  - wg standardu RS-232C /V.24/ - typ monitora ME 1664-RO/V,
  - wg standardu RS-232C oraz pętla prądowa 20mA - typ monitora ME 1664-RO/VP,
- kineskop A31-310W czarno biały,
- szybkość pracy 150 lub 300 bodów,
- realizuje następujące funkcje specjalne:
  - LF - wysuw linii i powrót znacznika do początku linii /nie realizuje odrębnej funkcji CR/,
  - BS, FF, BEL, NUL,
  - ruch znacznika w górę po odebraniu znaku CTRL, p,
  - obudowa telewizora VEGA 203,
  - niska cena,
  - wyposażenie dodatkowe:
    - pakiet sprzęgu SV24/400 umożliwiający niezależne podłączenie do systemu MERA 400,
    - uchwyt UM-21 lub podstawa UP-1 w przypadku użycia monitora jako urządzenia wolnostojącego, umożliwiające zmianę ustawienia względem operatora,
    - przystawka funkcyjna UF-12 do sterowania stronicowaniem.

E 1664-KSR

- terminal nadawczo-odbiorczy przeznaczony do pracy w systemach szeregowej asynchronicznej dwukierunkowej transmisji danych,
- interfejs szeregowy wyłącznie z obsługą linii T x D oraz R x D:
  - wg standardu RS-232C /V.24/ - typ monitora ME 1664-KSR/VZ,
  - wg standardu RS-232C /V.24/ oraz pętla prądowa 20mA - typ monitora ME 1664-KSR/VKP,
- klawiatura z przełącznikami kontaktronowymi w standardowym układzie QWERTY z klawiszami ustalającymi reżim współpracy monitora z komputerem: typ KT 1664/K,

- niska cena,
- pozostała charakterystyka monitora jak dla MX 1664-RO.

MV 1664/E

- monitor ekranowy odbiorczy,
- interfejs równoległy o strukturze analogicznej jak w interfejsie drukarki DZM 180 /7 bitowa magistrala danych wejściowych + 2 linie sterujące, hand-shaking/,
- szybkość pracy do 100.000 znaków/s /dla znaków wyświetlalnych/,
- kod ASCII, 128 znaków,
- kineskop M31-310GH, luminofor zielony P31 /wyłącznie/,
- obudowa telewizora VEGA 203,
- typowe zastosowanie: niezależna współpraca z drukarką DZM 180
- wyposażenie dodatkowe:
  - sprzęgi SM,
  - uchwyt UM-1 mocujący obrotowo monitor do drukarki DZM 180,
  - uchwyt UM-21 lub podstawa UP-1 w przypadku użycia monitora jako urządzenia wolnostojącego,
  - przystawka funkcyjna UF-11 do sterowania stronicowaniem oraz spowalnianiem /do szybkości zbliżonej do szybkości drukarki DZM 180 czyli 180 znaków/s/.

MV 1664 R

- funkcjonalny odpowiednik monitora MV 1664,
- obudowa typu "ZEFIR" R,
- kineskop M31-310GH, luminofor zielony P31,
- wyposażenie dodatkowe:
  - sprzęgi serii SM,
  - uchwyt UM-22 umożliwiający zmianę ustawienia,
  - podstawa UP-2,
  - przystawka funkcyjna UF-2 do sterowania stronicowaniem oraz spowalnianiem.

MV 1664-RO

- terminal odbiorczy przeznaczony do pracy w systemach szeregowej, asynchronicznej transmisji danych,

- interfejs szeregowy wg standardu RS-232C /V.24 z pełną obsługą linii od nr 101 do 109 i/lub pętla prądowa 20/60/mA,
- szybkość pracy 150, 300, 600, 1200, 2400, 4800, 9600 bodów /przełączalna/ z kwarcem 153,6kHz albo 110 bodów z kwarcem 112,64kHz,
- kod ASCII, 128 znaków,
- kineskop M31-310GH, luminofor zielony P31,
- obudowa typu "ZEKOM" R,
- wyposażenie dodatkowe:
  - sprzęg SV24/400 umożliwiający podłączenie do systemu MERA 400,
  - uchwyt UK-22 umożliwiający zmianę ustawienia,
  - podstawa UP-2,
  - przystawka funkcyjna UF-2 do sterowania stronicowaniem,
- w wersji MV 1664-RO/ET /na zamówienie/ monitor realizuje funkcję tabulacji poziomej; po odebraniu kodu HT, następny znak interpretuje jako liczbę binarną /modulo 64/ ustalającą położenie znacznika w wierszu.

MV 1664-KSR

- terminal nadawczo-odbiorczy przeznaczony do pracy w systemach szeregowej, asynchronicznej dwukierunkowej transmisji danych,
- interfejs szeregowy wg standardu RS-232C /V.24/ z pełną obsługą linii od nr 101 do 109 i/lub pętla prądowa 20"60/mA,
- klawiatura w układzie standardowym QWERTY z częścią numeryczną i klawiszami ustalającymi reżim współpracy monitora z komputerem:
  - z przełącznikami kontaktronowymi - typ monitora MV 1664-KSR/KK,
  - z przełącznikami hallotronowymi - typ monitora MV 1664-KSR/KH,
- pozostała charakterystyka monitora jak dla MV 1664-RO.

SV24/400

- pakiet sprzęgu umożliwiający podłączenie do systemu MERA 400,
- funkcjonalnie spełnia te same zadania, co jednostka sterująca SM-UZDAT 11 w systemie MERA 400 oraz dodatkowo przyjmuje i obsługuje przerwania operatora: może współpracować z każdym ter-

minalem posiadającym interfejs V.24 jak monitor ekranowy, drukarka mozaikowa DZM 180-KSR, itd,

- możliwe są dwa następujące rodzaje przerwania operatora:
  - generowane przez naciśnięcie klawisza BREAK. Pakiet SV/400 wykrywa znak "BREAK" jako znak bez bitu stopu, generując sygnał błędu FE,
  - generowane dowolnie wybranym klawiszem, którego kod znaku jest ustawiony mikroprzełącznikami znajdującymi się na pakiecie, przerwanie to można uaktywnić lub nie mikroprzełącznikiem,
  - wymiary i konstrukcja pakietu odpowiada funkcjonalnie jednostce sterującej SM-UZDAT 11,
  - pakiet sprzęgu wyposażamy jedynie w złącza /bez kabla/ umożliwiające podłączenie go do terminala.

Terminale nadawczo-odbiorcze wyposażamy w klawiatury produkowane przez firmę prywatną z Łodzi "Zakład Elektroniczny W.Kobierzycki, T.Torczyński" na naszej licencji. Sprawy handlowe i serwisowe pozostają w gestii naszej firmy.
Oferta na klawiatury dotyczy następujących typów:

KT 1664

- klawiatura alfanumeryczna w dwóch zasadniczych wykonaniach:
  - z przełącznikami hallotronowymi - typ KT 1664/H,
  - z przełącznikami kontaktronowymi - typ KT 1664/K,
- standardowy układ klawiszy QWERTY,
- blok alfanumeryczny, blok numeryczny oraz blok klawiszy funkcyjnych CN LINE, RO /PAGING/, FULL DUPLEX ustalających reżim współpracy terminala z komputerem oraz klawisz SPEED do wyboru szybkości transmisji pomiędzy terminalem i komputerem,
- złącze 871 025 wraz z kablem długości 1 m,
- wyjście w kodzie ASCII, 7 bitów danych + 1 sygnał sterujący + sygnały z klawiszy funkcyjnych,
- zasilanie +5V/0,6A,
- obudowa typu "ZEMON" R,
- sterowanie funkcją stronicowania w terminalu NV 1664-KSR.

KA 1664

- klawiatura alfanumeryczna pochodna typu KB 1664, bez wydzielonego bloku numerycznego.

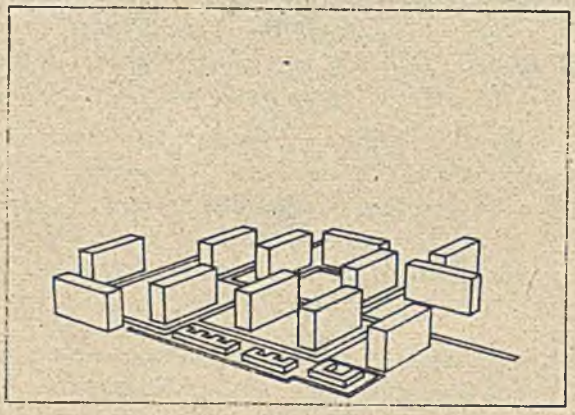
Podstawową dewizą naszej firmy jest wysoka jakość- dążymy do tego, aby nasze wyroby odznaczały się zwłaszcza wysokim poziomem technicznym /rozwiązań konstrukcyjnych i standardu wykonania/ oraz wysoką niezawodnością.
Dużą wagę przywiązujemy do spraw serwisowych i w każdym wypadku zapewniamy szybkie usunięcie zaistniałych niesprawności w działaniu dostarczonego przez nas sprzętu.

INSTYTUT DRÓG I MOSTÓW
POLITECHNIKA WARSZAWSKA

oferuje:

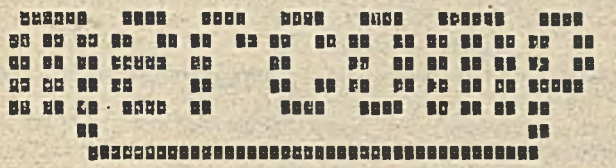
program P E R S

Program umożliwia graficzną prezentację układów brył w przestrzeni w postaci rzutu perspektywicznego z możliwością eliminacji linii niewidocznych.



Widoki obiektów można uzyskiwać z różnych punktów widzenia i w różnej skali. Dane wprowadza się wsadowo lub interakcyjnie. Rysunki można wyprowadzać praktycznie na dowolne urządzenie kreślące lub monitor graficzny.
Program działa na EMC Mera 400 w standardowym zestawie /wymagana pamięć 20 K słów/.

Autor programu : mgr inż. Jan Ustaszewski
Instytut Dróg i Mostów Politechniki Warszawskiej
Al. Armii Ludowej 16
00-637 Warszawa
tel. 25 75 40 lub 25 72 30



Centrum Badawczo-Wdrożeniowe "MERCOMP" Sp. z o.o.
04-994 Warszawa, Poezji 19, tel. 12-91-30

Centrum Badawczo-Wdrożeniowe "MERCOMP" Sp. z o.o. (jednostka gospodarki uspołecznionej) prowadzi modernizacje i kompletacje systemów minikomputerowych MERA-400 z wykorzystaniem środków technicznych własnych oraz poddostawców, w oparciu o projekty własne lub użytkowników systemów. Modernizacja taka pozwala na znaczne zwiększenie mocy obliczeniowej oraz wygody pracy w stosunku do znanych wersji standardowych.
Polega ona m. in. na rozszerzeniu konfiguracji, poza znanymi już rozwiązaniami, o moduły pamięci dyskowej 30 MB i taśm magnetycznych za pośrednictwem procesora PLIX, moduły monitora (MULTIX), pamięci operacyjnej do 512 k słów, przy współpracy m.in. PZ "Amepol" oraz specjalizowanych urządzeń sterowania i kontroli (np. CAMAC).
Równocześnie system MERA-400 wyposażany jest w odpowiednia do konfiguracji, wersje oprogramowania podstawowego tj. język sterowania zadaniami ZZK wraz z modyfikacjami systemu operacyjnego i oprogramowanie narzędziowe. Oprogramowanie to może być wzbogacone o kompilatory języków: LOGLAN, PASCAL, LISP, GASS oraz o oprogramowanie użytkowe w zależności od potrzeb.
W terminie późniejszym przewiduje się także wdrożenie nowej wersji procesora słowneso oraz wykorzystanie łączy światłowodowych do tworzenia lokalnych sieci mini -, mikro - komputerowych.
Mechanizm modernizacji jest otwarty na potrzeby użytkowników. Modernizacja może być realizowana w dwóch wariantach: pełnym i częściowym. Wariant pełny modernizacji obejmuje przejęcie przez CBW "Mercomp" całego sprzętu danego ośrodka i dostarczenie, w uzgodnionym terminie, kompletnego zmodernizowanego systemu MERA-400, łącznie z wyposażeniem ośrodka obliczeniowego. Wariant częściowy modernizacji może obejmować rozbudowę konfiguracji o ustalone moduły (np. pamięć dyskowa 30 MB) wraz z niezbędnym oprogramowaniem.

Wszelkich informacji udzielają:

mgr inż. Stanisław Gałązka
mgr Jerzy Dąbrowski

tel. 25-56-40 lub 25-80-16

LISTA UCZESTNIKÓW
II KONFERENCJI UŻYTKOWNIKÓW MINIKOMPUTERA MERA-400

1. ADAMCZAK Lucyna, Ośrodek ETO Huty Szkła Okiennego "Sandomierz" ul. Portowa 24, 27-600 Sandomierz, tel.sł.30-41 w. 143.
2. ANTOSIK Tomasz, Dział Informatyki PP Polmozbyt, ul. Strykowska 1/5, 91-729 Łódź, tel.sł. 849242.
3. BADURA Wojciech, Sekcja Informatyki Huty Szkła Okiennego "Kara", ul. Paplińskiego 1, 97-300 Piotrków Trybunalski, tel.sł. 16-11-41 w. 97.
4. BARCZYK Stanisław, Dział Informatyki Huty Szkła Okiennego "Jaroszowiec", ul. Kolejowa 1, 32-312 Jaroszowiec, tel. sł. Olkusz 309-14 w. 143.
5. BARECKI Roman, Pracownia ETO Wrocławskiego Biura Projektowo-Badawczego Budownictwa Przemysłowego, ul. Świdnicka 19, 50-066 Wrocław, tel.sł. 380-41 w. 258.
6. BAUER Zbigniew, Zakład Informatyki Spółdzielni Mieszkaniowych, ul. Kliny 2, 31-485 Kraków, tel.sł. 11-94-67.
7. BEDNARSKI Krzysztof, Dział Przygotowania i Analiz Gdańskich Zakładów Nawozów Fosforowych, ul. Kujawska 2, 80-958 Gdańsk, tel.sł. 438-263.
8. BESTYNSKI Piotr, Zespół Techniki Obliczeniowej ETO Poznańskiego Biura Projektów Budownictwa Przemysłowego, ul. Ratajczaka 10/12, 61-813 Poznań, tel.sł. 622-31 w. 151.
9. BIEGAŁA Ludwik, Ośrodek Obliczeniowy Instytutu Niskich Temperatur i Badań Strukturalnych PAN, ul. Pl. Katedralny 1, 50-950 Wrocław, tel.sł. 22-10-71, dom. 22-86-03.
10. BIEŃKOWSKI Andrzej, Zakład Informatyki Okręgowego Przedsiębiorstwa Geodezyjno-Kartograficznego, ul. Zwycięstwa 140, 75-313 Koszalin, tel.sł. 277-51 w. 121.
11. BIESIADOWSKI Jerzy, Ośrodek Informatyki Gdańskich Zakładów Elektronicznych Unimor, ul. Rzeźnicka 54/56, 80-822 Gdańsk, tel.sł. 375-330.
12. BOBCOW Andrzej, Dział Planowania i Informatyki Stoczni Remontowej "Radunia", ul. Na Ostrowiu 1, 80-873 Gdańsk, tel.sł. 31-68-31 w. 278,223,225.
13. BOBIŃSKA Bożena, Zakład Aparatów Elektrycznych Politechniki Łódzkiej, ul. Żwirki 36, 90-921 Łódź, tel.sł. 36-55-22 w. 274.
14. BONIECKI Marek, Ośrodek Informatyki Centralnego Związku Spółdzielczości Pracy, ul. Bonifaterska 14, 00-213 Warszawa, tel.sł. 31-41-63, 31-68-72.
15. BRANIECKI Andrzej, Instytut Okrętowy Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80-952 Gdańsk, tel.sł.471-718.

16. BRESLER Karol, Pracownia TO Wojewódzkiego Biura Projektów, ul. Wolności 286, 41-800 Zabrze, tel.sł. 71-20-21 w. 10.

17. BRYCZKOWSKI Krzysztof, Centrum Badawczo-Wdrożeniowe "Mercomp" - "Mera - Pnefal", ul. Poezji 19, 04-994 Warszawa, tel.dom. 13-65-96.

18. BRZESKI Michał, Ośrodek Elektronicznego Przetwarzania Danych Warszawskiej Spółdzielni Mieszkaniowej, ul. Krasińskiego 16, 01-581 Warszawa, tel.sł. 39-94-54.

19. BUJAŁOWSKA Arleta, Stacja Przetwarzania Danych Centrum Komputeryzacji Rynku "Dekar", ul. Stary Rynek, 61-773 Poznań, tel.sł. 205-411 w. 60.

20. BYCZKOWSKI Lech, Dział Informatyki Morskiej Obsługi Radiowej Statków, ul. Zygmunta Augusta 3-5-7, 81-359 Gdynia, tel.sł. 20-04-08.

21. CABIŃSKI J.Grzegorz, Ośrodek Badawczo-Rozwojowy Elektronicznych Układów Specjalizowanych, ul. Grudziądzka 46, 87-101 Toruń., tel.sł.330-45 do 47 e. 303.

22. CHMIELEWSKA Danuta, Zakład Produkcyjno-Montażowy Urządzeń Górnictwa Odkrywkowego "Fugo", Dział Postępu Ekonomiczno i Organizacyjnego, ul. Przemysława 85, 62-510 Konin, tel.sł. 215-81 w. 540.

23. CICHOCKI Mieczysław. BPBBO "Miastoprojekt-Łódź", Zespół Automatyzacji i Projektowania, ul. Traugutta 21, 90-133 Łódź, tel.sł. 328920 w. 248,137.

24. CICHOWSKA Gracjanna, Komórka Obliczeń Numerycznych Instytutu Chemii Fizycznej PAN, ul. Kasprzaka 44/52, 01-224 Warszawa, tel.sł. 32-32-21 w. 331.

25. CZAJKOWSKA Monika, Ośrodek Postępu Techniczno-Organizacyjnego PP "Orbis", ul. Marszałkowska 142, 00-061 Warszawa, tel.sł. 26-02-71 w. 56-16.

26. CZAJKOWSKI Piotr, Ośrodek Informacji i Obliczeń Elektronicznych Biura Projektów Budownictwa Komunalnego, ul. Grunwaldzka 2, 82-300 Elbląg, tel.sł. 273-96.

27. CZERNIAK Zbigniew, Instytut Okrętowy Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80-952 Gdańsk, tel.sł. 471-656.

28. CZERWIŃSKI Michał, Ośrodek Badawczo-Rozwojowy Elektronicznej Aparatury Medycznej, ul. Wolności 345, 41-800 Zabrze, tel.sł. 71-64-21 w. 255.

29. CZERWOSZ Leszek, Zakład Neurofizjologii Centrum Medycyny Doświadczalnej i Klinicznej PAN, ul. Dworkowa 3, 00-784 Warszawa, tel.sł. 49-74-88, 49-66-51.

30. DEPTUŁA Bohdan, Instytut Techniki Cieplnej i Silników Spalinowych Politechniki Poznańskiej, ul. Piotrowo 3, 60-965 Poznań, tel.sł. 782-201, 782-215.

31. DOBROWOLSKI Bogdan, PKP Oddział Geodezyjny w Katowicach, al. Roździeńskiego 1, 40-202 Katowice, tel.sł. 57-55-43.

32. DOKLMIUK Jerzy, Zakład Aparatów Elektrycznych Politechniki Łódzkiej, ul. Żwirki 36, 90-539 Łódź, tel.sł. 36-55-22 w. 938.

33. DRYJANSKI Waldemar, Dział Technologiczny Zakładu Produkcyjno-Montażowego Urządzeń Górnictwa Odkrywkowego "Fugo", ul. Przemysłowa 65, 62-510 Konin, tel.sł. 21-581 w.537

34. DUDA Jan, Instytut Automatyki, Inżynierii Systemów i Telekomunikacji AGH, al. Mickiewicza 30, 30-059 Kraków, tel. sł. 33-81-00 w. 28-51.

35. DUDEK Jan, Zakład Informatyki Spółdzielni Mieszkaniowych, Spółdzielnia Osób Prawnych, ul. Kliny 2, 31-465 Kraków tel.sł. 11-94-67.

36. DZIADURA Stanisław, Pracownia Systemów Minikomputerowych Ośrodka Badawczo-Rozwojowego Metrologii Elektrycznej, ul. Sulechowska 1, 65-022 Zielona Góra, tel.sł. 63054 w. 5434.

37. DŻOGA Jerzy, Przedsiębiorstwo Zagraniczne "Amepol", Plac Żelaznej Bramy 1, 00-136 Warszawa, tel.sł. 20-34-75.

38. FINDEISEN Piotr, Instytut Informatyki Uniwersytetu Warszawskiego, Skrytka pocztowa 1210, 00-901 Warszawa, tel.sł. 20-02-11 w. 2103.

39. GBURZYNSKI Paweł, Instytut Informatyki Uniwersytetu Warszawskiego, Skrytka pocztowa 1210, 00-901 Warszawa, tel. sł. 20-02-11 w. 2103.

40. GLURA Wiesław, Ogólnouczelniany Ośrodek Obliczeniowy UMK, ul. Gagarina 7, 87-100 Toruń, tel.sł. 260-17 w. 44.

41. GŁOWACZ Grzegorz, Dział Przetwarzania Danych Przemysłowego Centrum Optyki, ul. Ostrobramska 75, 04-175 Warszawa, tel.sł. 13-73-94.

42. GOCAŁEK Janusz, Instytut Technologii i Konstrukcji Budowlanych Politechniki Poznańskiej, ul. Piotrowo 5, 60-965 Poznań, tel.sł. 78-24-50.

43. GOTWALD Jerzy, PKP Oddział Geodezyjny w Katowicach, ul. Roździeńskiego 1, 40-209 Katowice, tel.sł. 57-55-43.

44. GRALL Jerzy, Zespół d/s ETO Instytutu Włókiennictwa, ul. Gdańska 91/93, 90-950 Łódź, tel.sł. 33-96-00.

45. GRALL Tadeusz, Zespół EPD Centralnego Ośrodka Badawczo-Rozwojowego Maszyn Włókienniczych "Cenaro", ul. Wólczańska 55/59, 90-950 Łódź, tel.sł. 32-85-70 w. 243.

46. GRZYBOWSKI Ryszard, BSiPE "Energoprojekt", Zespół Informatyki, ul. Piekary 19, 60-967 Poznań, tel.sł. 22-00-11 w. 298.

47. GUCWA Piotr, Zakład Oprogramowania Centrum Projektowania i Zastosowań Informatyki, Al. Niepodległości 190, 00-608 Warszawa, tel.sł. 27-12-20.

48. HLEBOWICZ Bogusław, Wydział Budownictwa i Maszyn Rolniczych Politechniki Warszawskiej, ul. Narbutta 86, 02-524 Warszawa, tel.sł. 260-61.

49. HORAK Andrzej, Centrum Elektronicznego Przetwarzania Danych Śląskiej Akademii Medycznej, ul. Medyków 18, 40-752 Katowice-Ligota, tel. sł. 526-081 w. 1466.

50. JAREMA Marek, Ośrodek ETO Huty Szkła Okiennego, ul. Portowa 24, 27-500 Sandomierz, tel.sł. 30-41 w. 229.

51. JEWULSKI Tadeusz, Dział Teleinformatyki Miejskiego Przedsiębiorstwa Komunikacyjnego, ul. Brożka 3, 30-405 Kraków, tel.sł. 66-10-22 w. 263.

52. JEZIERSKA-ZIEMKIEWICZ Elżbieta, Przedsiębiorstwo Zagraniczne "Amepol", Plac Żelaznej Bramy 1, 00-136 Warszawa, tel. sł. 20-34-75.

53. KACZOR Andrzej, Zespół ETO Biura Projektów Budownictwa Wiejskiego, ul. J. Nowickiego 32, 87-100 Toruń, tel. sł. 21051 w. 88.

54. KAIPER Barbara, Ośrodek Elektronicznego Przetwarzania Danych Warszawskiej Spółdzielni Mieszkaniowej, ul. Krasińskiego 16, 01-581 Warszawa, tel.sł. 39-94-54 w. 54.

55. KAŁUŻNA Bożena, Dział Postępu Ekonomicznego i Organizacyjnego Zakładu Produkcyjno-Montażowego Urządzeń Górnictwa Odkrywkowego "Fugo", Konin, tel.sł. 215-81 w. 540.

56. KAPAŁA Zenon, Instytut Okrętowy Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80-952 Gdańsk, tel.sł. 471-869.

57. KAPCIA Jerzy, Instytut Telekomunikacji Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80-952 Gdańsk, tel.sł. 471-643.

58. KARCZEMSKI Włodzimierz, Biuro Projektowo Badawcze Budownictwa Ogólnego "Miastoprojekt 2", ul. Więckowskiego 20,90-722 Łódź, tel.sł. 32-81-00.

59. KAREWICZ Bogusław, Zakład Elektroniki Komputerowej "Zekom", ul. Makowa 8, 91-480 Łódź,

60. KASPRZAK Andrzej, Pracownia Systemów Technologicznych Instytutu Obróbki Skrawaniem, ul. Wrocławska 37a, 30-011 Kraków, tel.sł. 33-93-33 w. 219.

61. KAZIMIERCZAK Bogdan, Instytut Elektroenergetyki i Automatyki Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80-952 Gdańsk, tel.sł. 471-619.

62. KLATKA Piotr, Wydział Budownictwa i Maszyn Rolniczych Politechniki Warszawskiej, Zespół Matematyki, ul. Narbutta 86, 02-524 Warszawa, tel.sł. 260-61 w. 227.

63. KLINCEWICZ Walerian, Zakład Oprogramowania Mini i Mikrokomputerów Centrum Projektowania i Zastosowań Informatyki, Al. Niepodległości 190, 00-608 Warszawa, tel.sł.27-1220.

64. KOBZA Marian, Zakład Maszyn Energetycznych Instytutu Techniki Cieplnej, ul. Dąbrowskiego 113, 93-208 Łódź, tel.sł. 43-26-50 w. 260.

65. KOCHEL Tadeusz, Górnośląski Okręgowy Zakład Gazownictwa, Ośrodek Elektronicznej Techniki Obliczeniowej, ul.Gwardii Ludowej 11, 41-800 Zabrze, tel.sł. 71-52-21 w. 5415.

66. KONECKA Ewa, Dział Informatyki Wojewódzkiego Przedsiębiorstwa Energetyki Cieplnej, ul. Dzierżyńskiego 5, 85-315 Bydgoszcz, tel.sł. 342-81 w. 288.

67. KONIECZNY Roman, Instytut Transportu Samochodowego Politechniki Śląskiej, ul. Krasińskiego 8, 40-012 Katowice, tel.sł. 539-107 w. 18.

68. KOSSOWSKI Marek, Dział Urządzeń Informatyki i Ruchu Załogi Kopalni Węgla Kamiennego "Siersza", 32-541 Trzebinia, tel.sł. Trzebinia 9 w. 542.

69. KOWALCZYK Bogdan, Instytut Technologii Mechanicznej Politechniki Warszawskiej, ul. Narbutta 86, 02-524 Warszawa, tel. sł. 49-98-71 w. 206.

70. KOWALSKI Stanisław,Centralny Ośrodek Informatyki Drogownictwa, Zakład Terenowy COID, ul. Pretficza 9/11, 50-984 Wrocław, tel.sł. 61-59-70.

71. KOZIOŁ Andrzej, Dział Urządzeń Informatyki i Ruchu Załogi Kopalni Węgla Kamiennego "Siersza", 32-541 Trzebinia, tel.sł. Trzebinia 9 w. 251.

72. KRAJEWSKA Maria, Biuro Projektów "Naftoprojekt", ul.Mysia 3, 00-496 Warszawa, tel.sł. 28-40-21 w. 88.

73. KRECZMAR Antoni, Instytut Informatyki Uniwersytetu Warszawskiego, Skrytka pocztowa 1210, 00-901 Warszawa, tel.sł. 20-02-11 w. 26-90.

74. KRUPA Jacek, Zrzeszenie WKTiR, ul. Czackiego 3/5, 00-043 Warszawa, tel.sł. 27-39-01.

75. KRYCZKA Jan, Instytut Inżynierii Chemicznej Politechniki Łódzkiej, ul. Wólczańska 175, 90-530 Łódź, tel.sł. 36-55-22 w. 715.

76. KRYGOWSKI Józef, Ośrodek Obliczeniowy Biura Projektów Budownictwa Wiejskiego, ul. M. Fornalskiej, 35-959 Rzeszów, tel.sł. 368-81.

77. KRZYSZCZUK Hanna, Zakład Nowych Metod Projektowania Centralnego Ośrodka Badawczo-Projektowego Budownictwa Przemysłowego "Bistyp", ul. Parkingowa 1, 00-518 Warszawa,tel. sł. 28-94-71.

78. KUBOWICZ Marek, Instytutu Informatyki Uniwersytetu Jagiellońskiego, ul. Kopernika 27, 31-501 Kraków, tel.sł. 11-02-77 w. 31.

79. KUCHARSKI Józef, Huta Szkła Okiennego "Sandomierz", ul.Portowa 24, 27-600 Sandomierz, tel.sł. 30-41 w. 229.

80. KUTRA Jerzy, Dział Przetwarzania Danych Przemysłowego Centrum Optyki, ul. Ostrobramska 75, 04-175 Warszawa, tel. sł. 13-73-94.

81. KWAPINSKI Henryk, Zakładowy Ośrodek Obliczeniowy ZACh METALCHEM, ul. Chorzowska 44b, 44-100 Gliwice, tel.sł. 31-64-41 w. 102.

82. KWASNICKA Hanna, Dział Informatyki OBR Elektronicznej Aparatury Medycznej, ul.Wolności 345A, 41-800 Zabrze, tel. sł. 71-64-21.

83. LAUSZ Irena, Żyrardowskie Zakłady Tkanin Technicznych, ul. Okrzei 51, 96-300 Żyrardów, tel.sł. 20-31 w. 284.

84. LEWANDOWSKI Marek, Przedsiębiorstwo Zagraniczne "Amepol", Plac Żelaznej Bramy 1, 00-136 Warszawa, tel.sł.20-34-75.

85. ŁUKASZEWSKA Małgorzata, Ośrodek Informatyki Zachodniej Dyrekcji Okręgowej Kolei Państwowych, ul. Marchlewskiego 130/140, 60-967 Poznań, tel.sł. 69-37-52.

86. MACHOWIAK Bogdan, Zespół Techniki Obliczeniowej Biura Projektów Budownictwa Wiejskiego, ul. Piekary 17, 60-989 Poznań, tel.sł. 33-05-081 w. 569.

87. MACIUK Bronisław, Katedra Organizacji Produkcji Wydziału Metalurgicznego Politechniki Śląskiej, ul. Krasińskiego 8b, 40-019 Katowice, tel.sł. 516-671 w. 24.

88. MADEJ Mariusz, Zakład Akustyki Cybernetycznej Instytutu Podstawowych Problemów Techniki PAN, ul. Świętokrzyska 21, 00-049 Warszawa, tel.sł. 26-12-81 w. 168.

89. MAJCHRZAK Ewa, Instytut Odlewnictwa Politechniki Śląskiej, ul. Towarowa 7, 44-100 Gliwice, tel.sł. 31-60-31.

90. MAKOWIECKI Krzysztof, Wrocławskie Biuro Projektowo-Badawcze Budownictwa Przemysłowego, ul. Świdnicka 19, 50-066 Wrocław, tel.sł. 380-41 w. 258.

91. MALINOWSKI Marek, Wydział Budownictwa i Maszyn Rolniczych Politechniki Warszawskiej, Zespół Matematyki, ul. Narbutta 86, 02-524 Warszawa, tel.sł. 260-61 w. 227.

92. MALINOWSKI Wojciech, Pracownia Informatyki Instytutu Fizyki Molekularnej PAN, ul. Smoluchowskiego 17/19, 60-179 Poznań, tel.sł. 67-40-71.

93. MANKIEWICZ Jerzy, Pracownia Informatyki Ośrodka Badań Stanu Torów, ul. Chodakowska 50, 03-816 Warszawa, tel.sł. 18-36-25.

94. MARTIN Aleksander, Zakładowy Ośrodek Informatyki Huty Miedzi "Głogów", 67-231 Żukowice, tel.sł. 320-71 w. 6380.

95. MARTIN Włodzimierz, Instytut Okrętowy Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80-952 Gdańsk, tel.sł. 471-869.

96. MARASEK Krzysztof, Instytut Podstawowych Problemów Techniki PAN, ul. Świętokrzyska 21, 00-049 Warszawa, tel.sł. 26-12-81 w.223.

97. MACZYNSKI Sławomir, Pracownia Zastosowania Metod Obliczeniowych Instytutu Melioracji i Użytków Zielonych, 05-550 Raszyn-Falenty, tel. sł. 500-531 w. 203, 243.

98. MICHAŁOWSKI Henryk, Pracownia Zastosowania Metod Obliczeniowych Instytutu Melioracji i Użytków Zielonych, 05-550 Raszyn-Falenty, tel.sł. 500-531, w. 203, 243.

99. MELER-KAPCIA Maria, Instytut Okrętowy Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80-952 Gdańsk, tel.sł. 471-795.

100. MIKULSKA Maria, Zakład Maszyn Elektrycznych i Transformatorów Politechniki Łódzkiej, ul. Żwirki 36, 90-924 Łódź, tel. sł. 623-09.

101. MILLER Janusz, Instytut Automatyki Inżynierii Systemów i Telekomunikacji Akademii Górniczo-Hutniczej, Al. Mickiewicza 30, 30-096 Kraków, tel.sł. 33-81-00.

102. MORAWSKA Barbara, Zakład Elektroniki Profesjonalnej P.Z. "TOBI", ul. Jagiellońska 8, 05-121 Legionowo, tel.sł. 25-54-03.

103. MORAWSKI Jacek, Centrum Badawczo-Wdrożeniowe "Mercomp", ul. Poezji 19, 04-994 Warszawa, tel.sł. 12-91-30.

104. MROZEK Zbigniew, Instytut Elektrotechniki i Elektroniki Politechniki Krakowskiej, ul. Warszawska.24, 31-155 Kraków, tel.sł. 33-03-00 w.614.

105. NAGORSKI Jerzy, Zakład Elektroniki Komputerowej "Zekom", ul. Makowa 8, 91-480 Łódź.

106. NAZAROWSKI Jerzy, Ośrodek Postępu Techbicznego NOT, Centralny Ośrodek Informacji Naukowo-Technicznej i Ekonomicznej, ul. Czackiego 3/5, 00-043 Warszawa, tel. sł. 27-39-01.

107. NIKODEMSKA Joanna, Pracownia Usług Obliczeniowych Ośrodka Obliczeniowego WZSP, ul. Dyrekcyjna 5, 80-852 Gdańsk, tel.sł. 31-56-21 w. 241.

108. NIKODEMSKI Marek, Instytut Okrętowy Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80-952 Gdańsk, tel.sł.471-869

109. NIKIEL Wojciech, Instytut Mechaniki Technicznej Politechniki Warszawskiej, ul. Narbutta 86, 02-524 Warszawa, tel. sł. 49-98-71 w. 406.

110. NIWIŃSKI Stanisław, Zakład Miernictwa i Sterowania Elektrycznego Instytutu Elektrotechniki, ul. Pożaryskiego 28, 04-703 Warszawa, tel.sł. 12-31-53.

111. NOWAK Dorota, Centrum Elektronicznego Przetwarzania Danych Śląskiej Akademii Medycznej, ul. Medyków 18, 40-754 Katowice-Ligota, tel.sł. 526-081, w. 1966.

112. NOWAK Zofia, Zakładowy Ośrodek Obliczeniowy ZACh "METAL-CHEM", ul. Chorzowska 44b, 44-101 Gliwice, tel.sł. 31-64-41.

113. OLEJNICZAK Romuald, Dział Przetwarzania Danych Ośrodka Elektronicznej Techniki Obliczeniowej, ul. Kościuszki 11, 25-310 Kielce, tel.sł. 444-51.

114. OLKOWSKI Henryk, Pracownia Matematyki Stosowanej Instytutu Ciężkiej Syntezy Organicznej "Blachownia", 47-232 Kędzierzyn-Koźle, tel.sł. 332-41 w. 5459.

115. OLTON Janusz, Zakład Elektroniki Wojskowego Instytutu Medycyny Lotniczej, ul. Krasińskiego 58, 01-755 Warszawa, tel.sł. 308-2634.

116. PABIS Leszek, Zakład Automatyki Instytutu Przemysłu Gumowego "Stomil", ul. Harcerska 30, 05-820 Piastów, tel. sł. 586-025, w. 284.

117. PALECZNY Andrzej, Centrum Komputeryzacji Rynku "Cekar", ,Poznań, tel. sł. 526-17.

118. PALUSZKIEWICZ Anna, Ogólnouczelniany Ośrodek Obliczeniowy UMK, Pracownia Systemów Operacyjnych, ul. Gagarina 7, 87-100 Toruń, tel.sł. 260-17 w. 44.

119. PEREK Maria, Zakładowy Ośrodek Informatyki Huty Szkła Okiennego "Szczakowa", ul. Kolejarzy 81, 32-520 Jaworzno, tel.sł. 774-41 w. 120.

120. PESZK Jerzy, Ośrodek Elektronicznego Przetwarzania Danych Warszawskiej Spółdzielni Mieszkaniowej, ul. Krasińskiego 16, 01-581 Warszawa, tel.sł. 39-94-54 w.54.

121. PIETROW Aleksander, Ośrodek Informatyki Urzędu Wojewódzkiego, ul. Geodetów 1, 39-400 Tarnobrzeg, tel.sł.22-35-49.

122. PIETRZAK Krystyna, Zakład Ekonomiki i Informatyki Miejskiego Przedsiębiorstwa Komunikacyjnego, ul. Piotrkowska 147, 90-440 Łódź, tel.sł. 36-15-60.

123. PIŁASIEWICZ Barbara, Instytut Dróg i Mostów Politechniki Warszawskiej, ul. Armii Ludowej 16, 00-637 Warszawa, tel.sł. 25-75-40.

124. PISIEWICZ Andrzej, Wojewódzkie Biuro Projektów, ul. Wolności 286, 41-800 Zabrze, tel.sł. 71-20-21 w. 84.

125. PŁACZEK Dorota, Sekcja Informatyki Rybnickiego Zakładu Prefabrykacji, ul. Wiejska 7, 44-200 Rybnik, tel.sł. 264-51 w. 81.

126. POLEROWICZ Zenon, Dział Technologiczny Fabryki Urządzeń Górnictwa Odkrywkowego, ul. Przemysława 85, 62-510 Konin, tel.sł. 215-81 w. 537.

127. PIZON Bogusława, ZACh "METALCHEM", Zakładowy Ośrodek Obliczeniowy, ul. Chorzowska 44b, 44-101 Gliwice, tel.sł. 31-64-41.

128. PSIORZ Marek, Dział Informatyki Zarządu Portu Szczecin-Swinoujście, ul. Bytomska 7, 70-603 Szczecin, tel. sł. 308-90.

129. RACZEK Olgierd, Ośrodek Informatyki Centralnego Związku Spółdzielczości Pracy, Sekcja Techniczna, ul. Bonifatorska,14, 00-213 Warszawa, tel.sł. 31-41-63.

130. RADOMSKA Maria, Instytut Chemii Organicznej i Fizycznej Politechniki Wrocławskiej, ul. Wyspiańskiego 27, 55-370 Wrocław, tel.sł. 20-24-12.

131. RAWIŃSKI TOMASZ, Instytut Okrętowy Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80952 Gdańsk, tel.sł.471-643.

132. REKUĆ Witold, Instytut Organizacji i Zarządzania Politechniki Wrocławskiej, ul. Smoluchowskiego 25, 50-372 Wrocław, tel.sł. 20-33-87.

133. ROS Marek, Ośrodek Postępu Technicznego Organizacyjnego PP Orbis, ul. Marszałkowska 142, 00-061 Warszawa, tel. sł. 26-02-71 w. 56-16.

134. RYCHLIK Ewa, Dział Informatyki Żyrardowskich Zakładów Tkanin Technicznych, ul. Okrzei 51, 96-300 Żyrardów, tel.sł. 20-31/5 w. 284.

135. SALWICKI Andrzej, Instytut Informatyki Uniwersytetu Warszawskiego- Skrytka pocztowa 1210, 00-901 Warszawa, tel.sł. 200-211 w. 2394.

136. SITEK Stanisław, WZT "TELKOM-TELETRA", ul. Bułgarska 67/73 60-320 Poznań, tel. sł. 676-801 w. 412.

137. SKORUPSKI Jacek, Ośrodek Informatyki Urzędu Wojewódzkiego, ul. Geodetów 1, 39-400 Tarnobrzeg, tel.sł. 22-35-49.

138. SŁAWIŃSKI Władysław, Zakład Informatyki Spółdzielni Mieszkaniowych, ul. Kliny 2, 31-485 Kraków, tel.sł.11-94-67

139. SOBCZYK Wojciech, PKP Oddział Geodezyjny w Katowicach, ul. Roździeńskiego 1, 40-202 Katowice, tel.sł. 57-55-4

140. SOKALSKI Andrzej, Ośrodek ETO Huty Szkła Okiennego, ul. Portowa 24, 27-600 Sandomierz, tel.sł. 3041 w. 229.

141. SPIRYDOWICZ Jerzy, Centralny Ośrodek Informatyki Drogownictwa, Zakład Terenowy COID w Zielonej Górze, ul. Dąbrowskiego 200, 65-714 Zielona Góra, tel.sł.71-441.

142. STEFANIAK Tomasz, Pracownia ETO Biura Projektów Przemysłu Lekkiego "Bedete", Pl. Zwycięstwa 2, 90-312 Łódź, tel. sł. 36-33-44 w. 113.

143. STEFANIAK Włodzimierz, Pracownia Krystalizacji Instytutu Podstaw Metalurgii PAN, ul. Towarowy 7, 44-100 Gliwice tel.sł. 31-60-31.

144. SZARATA Danuta, Dział Informatyki Wojewódzkiego Przedsiębiorstwa Energetyki Cieplnej, ul. Dzierżyńskiego 5, 85-315 Bydgoszcz, tel.sł. 342-81 w. 288.

145. SZCZEPANIAK Barbara, Zespół Specjalistów ETO Biura Projektów Budownictwa Komunalnego, ul. Tuwima 22, 90-002 Łódź, tel.sł. 32-32-75.

146. SZCZESNY Ryszard, Dział Informatyki Żyrardowskich Zakładów Tkanin Technicznych, ul. Okrzei 51, 96-300 Żyrardów, tel.sł. 20-31/15 w. 284.

147. SZCZYPUŁA Janusz, Instytut Inżynierii Sanitarnej i Ochrony Środowiska Politechniki Krakowskiej, ul. Warszawska 24, 31-155 Kraków, tel.sł. 33-03-00.

148. SZCZEPAŃSKI Marian, Dział Informatyki Morskiej Obsługi Radiowej Statków, ul. Zygmunta Augusta 3-5-7, 81-359 Gdynia, tel.sł. 20-04-08.

149. SZEWCZYK Grzegorz, Zakład Inżynierii Chemicznej Ośrodka Badawczo-Rozwojowego Przemysłu Barwników "Organika", ul. A. Struga 29, 95-100 Zgierz, tel.sł. Łódź 57-11-57 w. 949, 157.

150. SZYMCZAK Elżbieta, Instytut Mechaniki Stosowanej Politechniki Łódzkiej, ul. Stefanowskiego 1/15, 90-924 Łódź.

151. SLĘZAK Elżbieta, Ośrodek Informatyki Przedsiębiorstw Budowlano-Montażowych Górnictwa, ul. Koneckiego 5, 40-010 Katowice, tel.sł. 57-37-51.

152. SWIRSKI Zbigniew, ZPT "Mera-Pnefal", ul. Poezji 19, 04-994 Warszawa, tel.dom. 44-39-63.

153. TLAGA Waldemar, Instytut Technologii Elektronicznej Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80-952 Gdańsk, tel.sł. 471-647.

154. TUZIEMSKI Ryszard, Biuro Projektów Budownictwa Komunalnego ul. H. Sawickiej 27, 80-237 Gdańsk, tel.sł. 41-40-11 w. 13.

155. UFNALSKA Maria, Zespół ETO Biura Projektowo-Badawczego Budownictwa Ogólnego "Miastoprojekt 2", ul. Więckowskiego 20, 90-722 Łódź, tel.sł. 32-81-00.

156. USTASZEWSKI Jan, Instytut Dróg i Mostów Politechniki Warszawskiej, Al. Armii Ludowej 16, 00-637 Warszawa, tel. sł. 25-75-40.

157. WARSZTOCKA Anna, Zakłady Mechanizacji Budownictwa "Zremb", ul. Przasmyska 2, 01-756 Warszawa, tel.sł. 32-22-71.

158. WIECZOREK Karol, Zakład Ekonomiki i Informatyki Miejskiego Przedsiębiorstwa Komunikacyjnego, ul. Piotrowska 147, 90-440 Łódź, tel.sł. 36-63-55.

159. WIERZBICKI Jan, Zakładowy Ośrodek Informatyki Huty Szkła Okiennego "Szczakowa", ul. Kolejarzy 81, 32-520 Jaworzno, tel.sł. 774-41 w. 120.

160. WINIARSKI Maciej, Instytut Dróg i Mostów Politechniki Warszawskiej, Al. Armii Ludowej 16, 00-637 Warszawa,tel. sł. 25-80-16.

161. WERBIŃSKI Ryszard, Dział Przygotowania i Analiz Gdańskich Zakładów Nawozów Fosforowych, ul. Kujawska 2, 80-958 Gdańsk, tel.sł. 438-263.

162. WŁODARSKI Ładysław, Pracownia ETO Biura Projektów Przemysłu Lekkiego "Bedete", Pl. Zwycięstwa 2, 90-950 Łódz,tel. sł. 36-49-81.

163. WOŹNIAK Urszula, Stocznia Remontowa "Radunia", ul. Na Ostrowiu 1, 80-873 Gdańsk, tel.sł. 31-68-31 w. 278.

164. WOJCICKI Andrzej, Zakład Informatyki Okręgowego Przedsiębiorstwa Geodezyjno-Kartograficznego, ul. Zwycięstwa 140, 75-613 Koszalin, telmsł. 277-51 w. 170.

165. WROŃIECKI Zbigniew, Centralna Techniczno-Handlowa Przemysłu Precyzyjnego "Prema", ul. Krakowskie Przedmieście 47/51, 00-950 Warszawa, tel.sł. 26-32-01 w. 295.

166. ZAMYSŁOWSKI Edward, Sekcja Automatyzacji Badań Fabryki Osprzętu Samochodowego, ul. Przybyszewskiego 99, 93-126 Łódź, tel.sł. 402-40 w. 388.

167. ZAŁESKA-POPOW Barbara, Zakład Zastosowań Matematyki i Informatyki Akademii Rolniczej, ul. Dr Judyma 24, Szczecin, tel.sł. 700-61 w. 91.

168. ZAWADZKI Wiesław, Zakłady Mechaniczne, ul. Przemysława 14/15 66-400 Gorzów, tel.sł. 272-21.

169. ZIELIŃSKI Krzysztof, Pracownia Informatyki Ośrodka Badań Stanu Torów, ul. Chodakowska 50- 03-816 Warszawa, tel. sł. 18-36-25.

170. ZIELIŃSKI Stefan, Instytut Okrętowy,Politechniki Gdańskiej, ul. Majakowskiego 11/12, 80-952 Gdańsk, tel.sł. 471-795

171. ZIELONKA Jerzy, Zespół ETO Biura Projektowo Badawczego Budownictwa Ogólnego Miastoprojekt, ul. Więckowskiego 20, 90-722 Łódź, tel.sł. 32-81-00.

172. ZIEMKIEWICZ Andrzej, Instytut Maszyn Matematycznych, ul. Krzywickiego 34, 02-078 Warszawa, tel.sł. 21-84-41 w. 543.

173. ŻELIŃSKI Jerzy, Dział ETO Cieszyńskiej Fabryki Farb i Lakierów "POLIFARB", 43-400 Marklowice k/Cieszyna, tel.sł. 217-67.

174. ŻYŁŁA Romualnd, Instytut Inżynierii Chemicznej Politechniki Łódzkiej, ul. Wólczańska 175, 90-530 Łódź, tel.sł. 36-55-22 w. 837.

LISTA CZŁONKÓW POROZUMIENIA UŻYTKOWNIKÓW MINIKOMPUTERA MERA-400

Lp.	Nazwa i adres instytucji	Pełnomicnik/zastępca	Tel.
1	2	3	4
1	Akademia Górniczo-Hutnicza, Międzyresortowy Instytut Fizyki i Techniki Jądrowej Al.Mickiewicza 30, 30-059 Kraków	doc.dr hab. Marta Wasilewska-Radwańska dr inż. Marek Książkiewicz	34-00-10 34-16-98
2	Biuro Projektów "NAFTOPROJEKT" ul. Mysia 3, 00-496 Warszawa	mgr inż. Maria Krajewska	28-40-21 w.88
3	Biuro Projektów Budownictwa Komunalnego w Łodzi ul. Tuwima 22/26, 90-002 Łódź	mgr inż. Barbara Szczepaniak mgr Marek Tyneński	32-32-75 32-32-75
4	Biuro Projektów Budownictwa Komunalnego ul. H.Sawickiej 27, 80-237 Gdańsk-Wrzeszcz	Roman Mysiek Ryszard Tuziemski	41-40-11 w.13 41-40-11 w.13
5	Biuro Projektów Budownictwa Komunalnego ul. Grunwaldzka 2, 82-300 Elbląg	mgr Piotr Czajkowski mgr inż. Roman Świtecki mgr Henryk Pudzyński	259-03 259-03 243-76
6	Biuro Projektów Budownictwa Wiejskiego w Toruniu ul. Juliana Nowickiego 32, 87-100 Toruń	mgr Marek Dobrowolski mgr Andrzej Kaczor	210-51 w. 58 210-51 w. 88
7	Biuro Projektów Budownictwa Wiejskiego ul. Piekary 17, 60-959 Poznań	mgr inż. Michał Gawecki inż. Bogdan Machowiak	33-05-81 w. 570 33-05-81 w.569
8	Biuro Projektów Przemysłu Lekkiego "BEDETE" Pl.Zwycięstwa 2, 90-950 Łódź	mgr Ładysław Włodarski mgr Tomasz Stefaniak mgr Marek Czajkowski	36-49-81 36-33-44 w.113 36-33-44 w.113

1	2	3	4
9	Biuro Projektów Służby Zdrowia ul. Astrow 10, Katowice	mgr Wiesław Nowak Zofia Kremer	582-840 582-840
10	Biuro Projektowo Badawcze Budownictwa Ogólnego "Miastoprojekt-2" ul. Więckowskiego 20, Łódź	dr inż. L. Maro Włodzimierz Karczemski	3281-00 w.36 3281-00 w.37
11	Biuro Projektowo-Badawcze Budownictwa Ogólnego MIASTOPROJEKT ul. Traugutta 21/23- 90-113 Łódź	Mieczysław Cichocki Janusza Oleskiego	33-81-19 lub 32-89-20 w.248 33-81-19
12	Biuro Studiów i Projektów Energetycznych "Energoprojekt" ul. Piekary 19, 60-967 Poznań	mgr inż.Ryszarda Grzybowskiego inż.Edmunda Łuczaka	22-20-11 w.298 22-20-11 w.216
13	Biuro Projektów Budownictwa Morskiego "BIMOR" Plac Batorego 4, 70-207 Szczecin	mgr Władysław Kowalczyk mgr inż.Mieczysław Kosecki	403305 403-305
14	Bydgoskie Zakłady Przemysłu Gumowego "STOMIL" ul. Toruńska 155, 85-950 Bydgoszcz	mgr inż. Henryk Stachowski mgr Andrzej Poziemski	61-16-14 w. 412 61-16-41 w.179
15	Centrala Techniczno-Handlowa Przemysłu Precyzyjnego "Prama" ul.Krakowskie Przedmieście 47/51 , Warszawa	mgr inż.J.Konczerowicz mgr A.Mierzejewski mgr Z.Wroniecki	263201 w.295 263201 w.295 263201 w.295
16	Centralne Biuro Studiów i Projektów Budownictwa Wodnego "Hydroprojekt" ul.Powstańców Warszawskich,202, 80-162 Gdańsk 1	mgr inż. Henryk Noga mgr inż.A.Danecka-Noga	41-20-61 41-60-80 -"-
17	Centralny Ośrodek Badawczo-Projektowy Budownictwa Przemysłowego ul. Parkingowa 1, 00-518 Warszaw	mgr inż. Hanna Krzyszczuk mgr inż. K. Kociatkiewicz	289471 289471

1	2	3	4
18	Centralny Ośrodek Badawczo-Rozwojowy Maszyn Włókienniczych POLMATEX-CENARO ul.Wólczańska 55/59, 90-950 Łódź	mgr inż. Tadeusz Grall mgr Włodz.Jastrzębski	32-85-70 w.243 32-85-70 w.243
19	Centrum Medycyny Doświadczalnej i Klinicznej Polskiej Akademii Nauk ul. Dworkowa 3, 00-784 Warszawa	mgr Leszek Czerwosz mgr Joanna Kulesza	49-74-88 49-66-51 -"-
20	Fabryka Osprzętu Samochodowego "POLMO" ul.Przybyszewskiego 99, 93-126 Łódź	mgr inż. Edward Zamysłowski	388
21	Gdańskie Zakłady Nawozów Fosforowych ul.Kujawska 2, 80-958 Gdańsk 1	Ryszard Werbiński mgr Juliusz Ogórkowski	438-263 438-272 438-310
22	Główne Biuro Studiów i Projektów Przeróbki Węgla SEPARATOR ul. Armii Czerwonej 2, 40-952 Katowice	mgr inż.Z.Krypczyk mgr Elżbieta Piwko	586-071 w.513 586-071 w.415 537-307
23	Huta Szkła Okiennego "Szczakowa" ul. Kolejarzy 81, 32-520 Jaworzno	mgr inż. Jana Wierzbickiego mgr inż. Maria Perek	774-41 w.120, 282 774-41 w.282
24	Huta Szkła Okiennego "KARA" Przedsiębiorstwo Państwowe ul. Paplińskiego 1, 97-300 Piotrków Trybunalski	mgr Wojciech Badura Jolanta Olejniczak	16-11-41 w. 97 16-11-41 w. 97
25	Huta Szkła Walcowanego "Jaroszowiec" ul. Kolejowa 1, 32-312 Jaroszowiec	mgr inż. Stanisław Barczyk	wewm 143

1	2	3	4
26	Instytut Chemii Fizycznej PAN ul. Kasprzaka 44/52, 01-224 Warszawa	mgr inż. Gracjana Cichowska	32-32-21 w.331
27	Instytut Fizyki Molekularnej PAN ul. Smoluchowskiego 17/19, 60-179 Poznań	dr Wojciech Malinowski dr Andrzej Dezor	67-40-71 w.264 67-40-71 w.219
28	Instytut Informatyki Uniwersytetu Jagiellońskiego ul. Reymonta 4/243, 30-538 Kraków	mgr Kazimierz Jojczyk mgr Mieczysław Guja	33-63-77 w.538 33-63-77 w.538
29	Instytut Inżynierii Chemicznej Politechniki Łódzkiej ul. Wólczańska 165, Łódź	mgr inż.Jan Kryczka mgr inż. Romuald Żyłła	36655-22 w.715 36655-22 w.843
30	Instytut Melioracji i Użytków Zielonych ul. Falenty, 05-550 Raszyn	mgr Henryk Michałowski mgr inż. Sławomir Maczyński	500-531 w.243 500-531 w.243
31	Instytut Maszyn Przepływowych Politechniki Łódzkiej ul. Stefanowskiego 1/15 Łódź	Roman Malinowski Horodko Longin	36-55-22 w.1252 36-55-22 w.1252
32	Instytut Niskich Temperatur Badań Strukturalnych PAN Pl. Katedralny 1, 50-950 Wrocław	dr Ludwik Biegała inż. Kazimierz Nawrot	22-10-71 w.72 22-10-71 w.72
33	Instytut Odlewnictwa Politechniki Śląskiej ul. Towarowa 7, 44-100 Gliwice	mgr inż. Ewa Majchrzak mgr Włodzimierz Stefaniak	31-60-31 31-60-31
34	Instytut Technologii Elektronowej Politechniki Warszawskiej ul. Koszykowa 75, 00-662 Warszawa	mgr inż. Michał Duchnowski mgr inż. Stanisław Dmowski	21007-693 21007-906
35	Instytut Włókiennictwa ul. Gdańska 91/93, 90-950 Łódź	mgr Grall Jerzy mgr Krystyna Białas	339-600 w.214 339-600 w.175

1	2	3	4
36	Kopalnia Węgla Kamiennego "SIERSZA" ul. Kopalniana, 32-541 Trzebinia	mgr inż. Marek Kossowski mgr Aleksander Krzemień mgr inż. Andrzej Kozioł	Trzebina 9, w.542 wew. 251 wewz 251
37	Ośrodek Analizy Wartości Inżynierska Spółdzielnia Pracy ul. Słupska 28, 80-392 Gdańsk	mgr inż. Andrzej Dobrzański	52-02-57
38	Okręgowe Przedsiębiorstwo Geodezyjno Kartograficzne ul. Zwierzyniecka 10, 60-813 Poznań	Kierownik Ośrodka ETO B. Biełkowska J. Łykowski	472-41 w.27 472-41 w.27 472-41 w.27
39	Okręgowe Przedsiębiorstwo Geodezyjno Kartograficzne ul. Zwycięstwa 140, 75-613 Koszalin	mgr inż. Andrzej Bieńkowski mgr inż. Andrzej Wojciechowski mgr inż. Maria Urbańska	277-51 w.121 277-51 w.170 277-51w.168
40	Ośrodek Badawczo-Rozwojowy Elektronicznej Aparatury Medycznej ul. Wolności 345a, 41-800 Zabrze	inż. M. Czerwiński mgr J. Kręcioch mgr J. Śledziewska	71-64-21-9 w.255 71-64-21-9 w.255 71-64-21-9 w.255
41	Ośrodek Badawczo Rozwojowy Metrologii Elektrycznej ul. Sulechowska 1, 65-022 Zielona Góra	mgr Stanisław Dziadurę mgr inż. Urszula Bereza	65036 w. 5371 65036 w. 5434
42	Politechnika Gdańska Instytut Okrętowy ul. Majakowskiego 11/12 80952 Gdańsk	mgr inż. Andrzej Branicoki mgr inż. Zbigniew Czerniak	471-718 471-656
43	Politechnika Poznańska Instytut Techniki Cieplnej i Silników Spalinowych Zakład Techniki Cieplnej ul. Piotrowo 3, 60-965 Poznań	dr inż. Bohdan Deptuła mgr Bogumił Domalanus mgr Wacław Gołaś	782-215 782-215, 782-201
44	Politechnika Wrocławska Instytut Chemii Organicznej i Fizycznej I-4 ul. Wybrzeże Wyspiańskiego 27 50-370 Wrocław	dr inż. Ryszard Radmowski dr inż. Maria Radomska mgr inż. Mirosław Biernat	20-24-12 20-24-12 20-37-83

1	2	3	4
45	Polskie Górnictwo Naftowe i Gazownictwo Górnośląski Okręgowy Zakład Gazownictwa ul. Gwardii Ludowej 11, 41-800 Zabrze	mgr inż. Tadeusz Kochel Teodor Gorczyk mgr inż. Garstka-Maćkowiak Joanna	71-52-21 w.5415 71-52-21 w.5415 71-52-21 w.5415
46	Przemysłowe Centrum Optyki w Budowie ul. Ostrobramska 75, Warszawa	Grzegorz Głowacz Jerzy Kutra	13-73-94 13-73-94
47	Rybnicki Zakład Prefabrykacji ul. Wiejska 7, 44-200 Rybnik	mgr inż. Dorota Płaczek Wiesława Malowana	26451/81 26451/81
48	Śląska Akademia Medyczna Centrum Elektronicznego Przetwarzania Danych ul. Medyków, 40-752 Katowice-Ligota	dr inż. Janusz Pietkiewicz mgr Andrzej Horak mgr Dorota Nowak	527-081 w.1466 527-081 w.1466 527-081 w.1466
49	Warszawska Spółdzielnia Mieszkaniowa ul. Krasińskiego 16, 01-581	dyr. Zdzisław Jagodziński mgr inż. Michał Brzeski mgr Jerzy Peszek	39-51-00 39-94-54 w.54 39-94-54 w.54
50	Wojewódzki Ośrodek Informatyki ul.Jaktorowska 20/22, Żyrardów	Renata Opłatek Andrzej Malinowski	31-12 31-12
51	Wojewódzkie Biuro Projektów ul. Wolności 286, 41-801 Zabrze	mgr inż. Karol Bresler inż. Andrzej Pisiewicz	71-20-21 w.10 71-20-21 w.84
52	Wojskowy Instytut Higieny i Epidemiologii im. gen.Karola Kaczkowskiego ul. Kozielska 4, 01-163	mgr inż. Zbigniew Drzewiecki	250441 prosić 63-112 z Warszawa 3083112
53	WrocławskiegBiuro Projektowo-Badawoze Budownictwa Przemysłowego ul.Świdnicka 19, 50-950 Wrocław	Witold Czarnecki Grażyna Bednarek Roman Barecki	380-41 w.252 380-41 w.253 380-41 w.253

1	2	3	4
54	Zakłady Automatyki Chemicznej "METALCHEM" ul. Chorzowska 44, 44-101 Gliwice	inż. H.Kwapiński Z. Nowak mgr inż. B. Pizoń	31-64-41/102
55	Zakład Ekonomiki i Informatyki Miejskiego Przedsiębiorstwa Komunikacyjnego ul. Piotrowska 147/149, Łódź	mgr Krystyna Pietek mgr Lucjan Durkiewicz mgr Konstanty Owczarek	3648-12 32-19-32 3663-55.
56	Zakład Informatyki Spółdzielni Mieszkanio- wych, Spółdzielnia Osób Prawnych ul. Kliny 2, 31-465 Kraków	Bauer Zbigniew Rewiiak Zygmunt Dudek Jan	11-94-67 11-94-67 11-94-67
57	Zespół Opieki Zdrowotnej ul. Kamieniec 10, 34-500 Zakopane	mgr inż. Anna Smol inż. Kazimierz Slusarczyk	20-21 w.288 20-21 w.288
58	Żyrardowskie Zakłady Tkanin Technicznych ul. Okrzei 51, 96-300 Żyrardów	Ryszard Szczęsny Adam Rychlik Jacek Owczarek	20-31- do 35 w. 284 20-31 do 35 w. 284 20-31 do 35 w.284

Prorem/57/85/250