3 1981

informatyka

WYDAWNICTWO PRT SIGMA

Poszukujemy współpracowników

W ciągu ostatnich miesięcy sytuacja w Polsce zmieniła się tak dalece, że realne stają się nawet takie sfery działań, których wcześniej nie sposób było nawet przewidywać. Praca naszej Redakcji również może zmienić swój charakter; INFORMATYKA ma teraz szansę stać się autentycznym forum środowiska, służącym wymianie istotnych poglądów i doświadczeń.

Rządy autorytatywne w Polsce, które spowodowały w efekcie m. in. całkowite uzależnienie informatyków od decyzji odgórnych (częściej politycznych niż gospodarczych), ograniczały do niedawna zakres aktywności pisma. Redakcja nie była choćby dopuszczana do źródeł prawdziwych informacji o informatyce, mogła się opierać jedynie na spreparowanych zestawieniach, przygotowanych przez poszczególne instytucje dla pokrzepienia decydentów. Ranga pisma — w następstwie — znacznie malała.

Brak ścisłego związku informatyki ze społecznym i gospodarczym stanem kraju przyniósł w jej ramach dość swobodny rozwój prac naukowych, dający w znacznej mierze świadectwo niefrasobliwości „twórców". Niedostatek jednoznacznych kryteriów oceny tych prac zwiększył obfitość dokonań pseudonaukowych. Tym piętnem naznaczona została również INFORMATYKA.

Większość przedstawicieli środowiska, którzy byli świadomi realiów, nie podjęła współpracy z pismem, sądząc zapewne, że ich głos i tak zostanie odrzucony przez Redakcję, reprezentującą interesy instytucji zarządzających, albo obawiając się przykrych reperkusji. Ich postawa utorowała drogę na łamy wielu pracom o niskiej wartości, które nie znalazły silnej konkurencji.

Dzisiaj, kiedy prasa zaczyna zbliżać się coraz bardziej do rzeczywistości, kiedy skutki wieloletniej irracjonalnej gospodarki, nauki i techniki zostają poddane ostremu społecznemu osądowi — pojawia się szansa odrzucenia w przeszłość złych doświadczeń. Ale — powtarzamy jeszcze raz — zależy to przede wszystkim od ś r o d o w i s k a. Jeśli ci informatycy, których poglądy i wiedza godne są upowszechnienia, nie zdecydują się na współpracę ze s w o i m pismem, nigdy nie zaspokoi ono ich oczekiwań. A oczekiwania te — jak wiemy — są duże.

Wzywamy zatem wszystkich informatyków, którzy czują się na siłach przedstawiać na łamach swoje poglądy i przekazywać swą wiedzę, by niezwłocznie podjęli współpracę z INFORMATYKĄ. Czekamy na kontakt. Zasady współdziałania ustalimy z każdą osobą indywidualnie. Oferujemy wszystkie formy stałej współpracy. Publikacje będą oczywiście honorowane zgodnie z obowiązującymi stawkami autorskimi. Prosimy o szybką decyzję.

REDAKCJA

WYDAWNICTWO
NACZELNA ORGANIZACJA TECHNICZNA
CZASOPISM I KSIĄŻEK TECHNICZNYCH
SIGMA
ul. Świętokrzyska 14a
00-950 Warszawa
skrytka pocztowa 1004

Redakcja: 00-041 Warszawa, ul. Jasna 14/16, pokój 326, tel. 27-71-40, dyżury redakcji 10.00—13.00

Zakł. Graf. „Tamka". Zam. 47. Obj. 5,5 ark. druk. Nakład 6200 egz. L-118.

Cena egzemplarza zł 30.— INDEKS 36124 Prenumerata roczna zł 360.—

Ślusarek M.: Realizacja programów współbieżnych w systemie ODRA 1305

INFORMATYKA 1981, nr 3, s. 4

Charakterystyka zasad działania oraz metod budowy programów wieloczłonowych w systemie ODRA 1305. W oparciu o teorię procesów współbieżnych omówiono sposoby komunikacji pomiędzy członami programu, podkreślając znaczenie subprogramowania w praktyce programistycznej.

Слюсарек М.: Реализация согласованных программ в системе ОДРА 1305

ИНФОРМАТИКА 1981, № 3, стр. 4

Характеристика принципов работы и методов построения многозвенных программ ОДРА 1305. На основе теории согласованных процессов обсуждаются способы сообщения между звенами программы, подчеркивается практическое значение субпрограммирования.

Libura L.: Minisystem wyszukiwania informacji o materiałach

INFORMATYKA 1981, nr 3, s. 6

Ogólna charakterystyka systemu wyszukiwania informacji o materiałach w oparciu o ich nazwy. System został opracowany w Gdyńskiej Stoczni Remontowej na komputerze ODRA 1305 dla potrzeb technicznego przygotowania produkcji.

Либура Л.: Мини-система поиска информации о материалах

ИНФОРМАТИКА 1981, № 3, стр. 6

Общая характеристика системы поиска информации о материалах на основе их названий. Система была разработана в Гдынской судоремонтной верфи на вычислительной машине ОДРА 1305 для технических потребностей приготовления производства.

Dobija M., Kolarzyk K.: Koszty informacji systemowej

INFORMATYKA 1981, nr 3, s. 8

Propozycja praktycznej metody badania efektywności obliczeń informatycznych. Metoda ta, opracowana w branżowym ośrodku informatyki i organizacji PETROINFORM, polega na porównywaniu kosztów uzyskania informacji metodami informatycznymi oraz metodą tradycyjną.

Добия М., Коляжик К.: Стоимость системной информации

ИНФОРМАТИКА 1981, № 3, стр. 8

Предложение практического метода исследования эффективности автоматизированных вычислений. Настоящий метод разработан в отраслевом центре вычислительной техники и организации ПЕТРОИНФОРМ. Главным образом, сравнивает он стоимости получения информации вычислительными методами и традиционным способом.

Belczak S., Czaja J.: Język BASIC dla mikrokomputera INTEL 8080

INFORMATYKA 1981, nr 3, s. 11

Ogólna charakterystyka rozwiązań interpretera języka BASIC opracowanego w Instytucie Informatyki Politechniki Gdańskiej dla potrzeb przemysłowego przetwarzania danych na mikrokomputerze INTEL 8080. Podano organizację i możliwości użytkowe interpretera, podkreślając jego efektywność oraz łatwość dalszej rozbudowy.

Бэльчак С., Чая Й.: Язык BASIC для малой вычислительной машины INTEL 8080.

ИНФОРМАТИКА 1981, № 3, стр. 11

Общая характеристика решений интерпретора языка BASIC, разработанного в Институте вычислительной техники Гданской Политехники для потребностей промышленной обработки данных на малой вычислительной машине INTEL 8080. Представляется организация и эксплуатационные возможности интерпретора, подчеркивается его эффективность и легкость дальнейшего развития.

Chomicki J.: Abstrakcyjne typy danych

INFORMATYKA 1981, nr 3, s. 13

Charakterystyka nowego narzędzia inżynierii oprogramowania, przeznaczonego do formalnego opisywania struktur danych. Podano definicje podstawowych pojęć oraz wskazano na dużą przydatność tego rodzaju metody opisu w dziedzinie projektowania baz danych.

Хомицки Й.: Абстрактные типы данных

ИНФОРМАТИКА 1981, № 3, стр. 13

Характеристика нового средства технологии математического обеспечения предназначенного для формального описывания структур данных. Даются определения основных понятий, а также подчеркивается значительная пригодность такого рода метода описания в области проектирования баз данных.

Lewoc J. B.: O zawodzie inżyniera informatyka

INFORMATYKA 1981, nr 3, s. 17

Krytyczna ocena aktualnego programu kształcenia informatyków w Politechnice Wrocławskiej, a także niewłaściwej praktyki obsadzania stanowisk kierowników zespołów wdrażania nowych przemysłowych zastosowań informatyki.

Левоц Й. Б.: О профессии инженера специалиста по вычислительной технике

ИНФОРМАТИКА 1981, № 3, стр. 17

Критическая оценка актуальной программы обучения специалистов по вычислительной технике во Вроцлавском политехническом институте, а также несоответствующей практики укомплектовывания постов заведующих коллективов внедрения новых промышленных применений вычислительной техники.

Ślusarek M.: Implementation of concurrent programs in the ODRA 1305 computer system

INFORMATYKA 1981, No 3, p. 4

Characteristics of operation principles and method of concurrent programs building in the ODRA 1305 computer system. Discussed the communication means between program members, based on the theory of concurrent processes, with emphasis on significance of subprogramming in everyday programmer's work.

Ślusarek M.: Die Realisation der verzahnt ablaufenden Programme im ODRA 1305 Rechnersystem

INFORMATYKA 1981, Nr. 3, S. 4

Die Charakteristik der Wirkungsweise und der Erarbeitungsmethoden für die mehrgliedrige Programme im ODRA 1305 Rechnersystem. Auf Grund der Theorie von verzahnt ablaufenden Prozessen wurde die Kommunikation zwischen Programmgliedern besprochen mit Unterstreichung der Subprogrammierung in alltäglicher Programmierungspraxis.

Libura L.: The minisystem of materials information retrieval

INFORMATYKA 1981, No 3, p. 6

General characteristics of the information retrieval system, which bases on search according materials designation. The system was elaborated in the Gdynia Repair Shipyard on the ODRA 1305 computer for technical production preparation.

Libura L.: Ein Mini-System für die Informationsrecherche von Materialien

INFORMATYKA 1981, Nr. 3, S. 6

Allgemeine Charakteristik eines Informationssystems für das Wiederauffinden von Materialien nach dem Benennungsmerkmal. Das System wurde in der Gdyniar Reparaturwerft auf dem ODRA 1305 Rechner für die technische Vorbereitung der Produktion erarbeitet.

Dobija M., Kolarzyk K.: Costs of system's information

INFORMATYKA 1981, No 3, p. 8

Proposal of the practical method for data processing efficiency analysing. The method, elaborated in the branch data processing and organization center PETROINFORM, bases on the comparison of information acquisition cost applying electronic data processing and manual work.

Dobija M., Kolarzyk K.: Die Kosten der Systeminformation

INFORMATYKA 1981, Nr. 3, S. 8

Ein Vorschlag der praktischen Methode für die Effektivitätsanalyse der EDV. Diese Methode, die im Fachzentrum für EDV und Organisation PETROINFORM erarbeitet wurde, beruht auf dem Kostenvergleich der Informationsgewinnung mit Hilfe der EDV-Technik und der Handarbeit.

Belczak S., Czaja J.: BASIC language for the microcomputer INTEL 8080

INFORMATYKA 1981, No 3, p. 11

General characteristics of solutions for the BASIC interpreter, elaborated in the Data Processing Institute of the Gdańsk Technical University for industrial data processing on the INTEL 8080 microcomputer. Presented the organisation and utilizable possibilities of the interpreter with emphasis on its effectiveness and facility for future extending.

Belczak S., Czaja J.: BASIC-Sprache für den Mikrorechner INTEL 8080

INFORMATYKA 1981, Nr. 3, S. 11

Allgemeine Charakteristik von Lösungen des BASIC-Sprache Interpreters, der im Informatik Institut der Technischen Universität in Gdańsk für die industrielle Datenverarbeitung auf dem Mikrorechner INTEL 8080 erarbeitet wurde. Es wurden die Organisation und die Gebrauchsmöglichkeiten des Interpreters mit Betonung seiner Effektivität und Ausbauzugänglichkeit angegeben.

Chomicki J.: Abstract data types

INFORMATYKA 1981, No 3, p. 13

Characteristics of the new software engineering tool, appropriated for formal description of data structures. Presented definitions of fundamental terms and pointed on great usability of this description method in the field of data bases designing.

Chomicki J.: Die abstrakten Datentypen

INFORMATYKA 1981, Nr. 3, S. 13

Die Charakteristik des neuen Hilfsmittels für die Softwaretechnologie, das für die formelle Beschreibung der Datenstrukturen bestimmt ist. Es wurden die Definitionen der Grundbegriffe angegeben und die grosse Brauchbarkeit dieser Beschreibungsmethode betont.

Lewoc J. B.: About profession of data processing engineer

INFORMATYKA 1981, No 3, p. 17

Critical evaluation of the actual data processing specialists education program in the Wrocław Technical University, as well of improper practice of appointing implementation group leaders for new industrial data processing applications.

Lewoc J. B.: Über den Beruf des EDV-Ingenieurs

INFORMATYKA 1981, Nr. 3, S. 17

Kritische Beurteilung des aktuellen Ausbildungsprogramms für die EDV-Ingenieure in Wrocławer Technischer Universität, sowie der falschen Praxis der Besetzungen von Einsatzgruppenleiterposten für die neuen industriellen EDV--Anwendungen.

Informatyka

zastosowania w gospodarce, technice i nauce

Nr 3
MIESIĘCZNIK
1981
ROK XVI
Marzec

ORGAN KOMITETU INFORMATYKI, MINISTERSTWA NAUKI, SZKOLNICTWA WYŻSZEGO I TECHNIKI ORAZ KOMITETU NAUKOWO-TECHNICZNEGO NOT DS. INFORMATYKI

W NUMERZE:

MACIEJ ŚLUSAREK
Instytut Informatyki
Uniwersytet Jagielloński
Kraków

Realizacja programów współbieżnych w systemie ODRA 1305

W okresie minionej dekady prowadzone były intensywne prace badawcze, zmierzające do opanowania metodologii programowania współbieżnego. Uzyskane wyniki pozwalają na stwierdzenie, że realizacja programów współbieżnych jest w zasięgu codziennej praktyki programistycznej. Do celów programowania współbieżnego skonstruowano pewną liczbę specjalizowanych języków (np. CONCURRENT PASCAL [5]), jednak żaden z nich nie jest dostępny w systemie ODRA 1300. Celem niniejszego artykułu jest prezentacja możliwości realizacji programów współbieżnych w języku PLAN. Na wstępie przytoczymy kilka istotnych pojęć związanych z programowaniem współbieżnym.

Procesem nazywamy zbiór operacji całkowicie uporządkowany w czasie. Wykonywane są one jedna po drugiej, a ich wynik jest niezależny od czasu [4]. Program skonstruowany w taki sposób jest procesem; mówimy wówczas o programie sekwencyjnym.

Procesy są współbieżne, jeżeli ich wykonanie pokrywa się w czasie. Nieistotne jest przy tym, czy operacje wchodzące w skład poszczególnych procesów przeplatają się czy też pokrywają w czasie [1, 4]. Program nazywamy współbieżnym, jeśli składa się z co najmniej dwóch współbieżnych procesów. Jeśli procesy współbieżne operują na wspólnym zasobie danych, to — aby wynik obliczeń był prawidłowy — konieczna jest synchronizacja ich działania. Podstawowymi pojęciami wykorzystywanymi przy rozwiązywaniu tego typu problemów są:

- sekcja krytyczna [1], służąca do wykluczania się procesów w czasie, jeśli operują na wspólnych danych
- semafor [1], zmienna synchronizacyjna, określająca liczbę zdarzeń (sygnałów), które wystąpiły (zostały nadane), a które nie zostały jeszcze obsłużone (odebrane); w przykładach przedstawionych w dalszej części artykułu do realizacji sekcji krytycznej wykorzystano pojęcie semafora binarnego
- impas (blokada, zastój), czyli stan oczekiwania przez jeden lub więcej procesów na zdarzenie, które nigdy nie nastąpi.

Programy wieloczłonowe

Komputer ODRA 1305 jest maszyną wieloprogramową. Oznacza to, że czas centralnego procesora jest dzielony pomiędzy kilka lub kilkanaście programów, znajdujących się równocześnie w pamięci operacyjnej. Każdy z nich zajmuje spójny, rozłączny z innymi obszar i nie ma możliwości oddziaływania na obszary pamięci przydzielone innym programom. Jeżeli jeden program oczekuje na wystąpienie jakiegoś zdarzenia (np. w wyniku wykonania ekstrakodu SUSBY), to centralny procesor jest przydzielany innemu programowi, gotowemu do wykonania. Do realizacji tego przydziału wykorzystuje się system przerwań [9], obsługiwanych przez program sterujący EGZEKUTOR.

Istnieje możliwość wyodrębnienia wewnątrz jednego programu do czterech procesów, zwanych subprogramami lub członami programu, identyfikowanych indeksami od Ø do 3. Przy przydziale centralnego procesora są one traktowane tak jak niezależne programy, mogą natomiast operować na pamięci w dowolnym miejscu programu — w szczególności na ściśle określonych danych, służących do komunikacji między procesami.

Każdy człon programu ma swój priorytet, akumulatory, licznik rozkazów, tryby skoku, adresacji i usuwania zer. EGZEKUTOR ma jedynie informację o liczbie członów i ich stanie oraz wskazówkę, który z nich jest aktualnie wykonywany. Przydział pamięci operacyjnej oraz urządzeń zewnętrznych odbywa się na poziomie programu, a nie jego członu.

W teorii procesów współbieżnych wyróżnia się podstawowe stany procesu:

- aktywny (posiadający centralny procesor)
- gotowy (oczekujący na przydział procesora)
- zawieszony (oczekujący na określone zdarzenie).

W dalszej części artykułu, w celu uproszczenia objaśnień, aktywnym nazywany będzie człon programu, posiadający centralny procesor lub oczekujący na jego przydział.

W systemie ODRA 1305 stan członu programu charakteryzowany jest przez wartości pięciu jednobitowych wskaźników, przechowywanych przez EGZEKUTOR [3, 6]. Są to następujące wskaźniki:

SL — ustawiany dla każdego członu o numerze różnym od zera przy ładowaniu programu do pamięci operacyjnej; zerowany przez pierwszy rozkaz AUTO odnoszący się do danego członu; przechowuje informację o tym, czy człon był kiedykolwiek aktywizowany (bezpośrednio po załadowaniu aktywny jest tylko człon zerowy)

SM — ustawiony oznacza, że dany człon „zawiesił się" wykonując ekstrakod SUSAR

SP — ustawiony oznacza, że dany człon „zawiesił się" wykonując ekstrakod SUSIN

MM — ustawiany na skutek wykonania odnoszącego się do danego członu rozkazu AUTO, jeśli w tym czasie człon jest aktywny; jest on przechowywany w celu uniknięcia impasu

PM — ustawiany podczas obsługi przerwania pochodzącego od urządzenia pracującego w trybie odpowiedzi bezpośredniej [9], jeśli przerwanie to pojawi się w momencie, w którym człon jest aktywny; jest on przechowywany również w celu uniknięcia impasu.

Sterowanie pracą programów wieloczłonowych

Do realizacji programów wieloczłonowych wykorzystywane są opisane poniżej ekstrakody. W opisie ich funkcji przyjęto, że jeśli α jest typem wskaźnika, a X numerem członu, to α_X oznacza wartość tego wskaźnika dla członu X, natomiast α oznacza wartość tego wskaźnika dla członu wykonującego ekstrakod.

a) Rozkaz o kodzie 163 w języku PLAN ma postać:
AUTO X N(M)
Realizuje on następujące dwie funkcje:

- pierwsze uaktywnienie członu o numerze X ($X \neq \emptyset$); niezerowy argument N(M) jest wprowadzany do licznika rozkazów członu X, a wskaźnik SL_X jest zerowany; argument tego ekstrakodu jest dwunastobitowy, powinien być zatem adresem stałej rozkazowej, zawierającej instrukcję skoku do właściwej części programu

- kolejne uaktywnienie członu X; w tym przypadku N(M) jest równe zeru i jeśli człon X jest nieaktywny, to jest on „odwieszany" (przez wyzerowanie wskaźników SM_X i SP_X), a jeśli jest aktywny, to ustawiany jest wskaźnik MM_X; Po otrzymaniu centralnego procesora człon X będzie wykonywany od rozkazu o adresie podanym w jego liczniku rozkazów.

Ekstrakod AUTO nie uruchamia aktywizowanego członu (może to zrobić tylko EGZEKUTOR), a jedynie zmienia wartość odpowiednich wskaźników.

b) Rozkaz o kodzie 164 występuje w następujących dwóch wariantach:
- SUSAR 1 (pole argumentu puste);

ekstrakod testuje wskaźnik MM członu, który go wykonuje i jeśli MM jest ustawiony, to jest on zerowany i człon pracuje dalej; jeśli MM ma wartość zero, to człon jest zawieszany (przez ustawienie wskaźnika SM)
- SUSIN 2 (pole argumentu puste);

jeśli co najmniej jeden ze wskaźników MM i PM jest ustawiony, to oba są zerowane i człon pracuje dalej; w przeciwnym przypadku człon jest zawieszany (przez ustawienie jego wskaźnika SP).

c) Rozkaz o kodzie 162 ma postać:

SUSMA X N(M)

Ekstrakod testuje słowo o adresie N(M) + 1 i jeśli zawartość tego słowa jest równa zeru, to zmienia ją na różną od zera (ustawiając w nim wszystkie bity), zawartość akumulatora X przesyła do słowa o adresie N(M) oraz zwiększa licznik rozkazów członu, dzięki czemu ominięty zostaje następny rozkaz. Jeśli zawartość słowa o adresie N(M) + 1 jest niezerowa, to sterowanie przechodzi do następnego rozkazu w programie.

Przerwania zewnętrzne i zegarowe w programach wieloczłonowych

Zdecydowana większość standardowego oprogramowania systemu ODRA 1300 wykorzystuje normalny tryb zawieszeń do sterowania pracą urządzeń zewnętrznych [9]. Przerwanie od urządzenia pracującego w takim trybie sprawia, że „odwieszany" jest każdy człon programu, który wykonał ekstrakod SUSBY odnoszący się do tego urządzenia.

Przerywanie pochodzące od urządzenia pracującego w trybie odpowiedzi bezpośredniej [6, 9] powoduje — dla każdego członu X w programie — sprawdzenie (przez wytestowanie wskaźnika SP_X), czy jest on zawieszony na skutek wykonania ekstrakodu SUSIN. Jeśli SP_X jest ustawiony, to jest on zerowany (co implikuje uaktywnienie członu X), natomiast jeśli SP_X jest równy zeru, to ustawiany jest wskaźnik PM_X.

Należy podkreślić, że ekstrakod SUSBY zawiesza tylko jeden człon programu w oczekiwaninu na koniec przesyłania [7], zatem normalny tryb zawieszeń może być stosowany również w programach wieloczłonowych.

Realizacja sekcji krytycznej

Jak wspomniano, do realizacji sekcji krytycznej może służyć semafor binarny [2, 4]. Załóżmy, że zmienna *mutex* może przybierać wartość 0 i 1 i rozważmy proces cykliczny składający się z następujących operacji:
- prolog
- sekcja krytyczna
- epilog.

Sekcję krytyczną można rozbić na następujące trzy operacje składowe:

- wait (mutex)
- operacja na wspólnym zasobie danych
- signal (mutex).

Operacja *wait* powoduje testowanie zmiennej *mutex* i jeśli jest ona równa zeru, to następuje zmiana jej wartości na 1 i kontynuacja procesu. Jeśli natomiast *mutex* ma wartość 1 (co oznacza, że inny proces operuje na wspólnym zasobie danych), to proces zostaje „zawieszony" i ustawiony w kolejce procesów oczekujących na zwolnienie wspólnego zasobu. Operacja *signal* zeruje zmienną *mutex* i „odwiesza" pierwszy proces z kolejki oczekujących na tym semaforze.

Załóżmy, że w obszarze zmiennych dolnych zarezerwowano obszar MUTEX (2). Operację *wait* realizuje następujący ciąg rozkazów:

```
#                                  POCZĄTEK SEKCJI KRYTYCZNEJ
WT      SUSMA      ϕ      MUTEX
        BRN                SUS
        BRN                EN
SUS     SUSAR      1
        BRN                WT
```

Rolę semafora binarnego pełni słowo o adresie MUTEX + 1, a EN jest etykietą początku operacji na wspólnym zasobie danych. Załóżmy, że ten ciąg rozkazów jest wykonywany przez człon X i rozważmy dwa przypadki, które mogą zajść w chwili wykonywania ekstrakodu SUSMA.

a) Zawartość słowa o adresie MUTEX + 1 jest różna od zera. Oznacza to, że inny człon Y operuje na wspólnym zasobie danych. Człon X wykona rozkaz skoku do etykiety SUS i „zawiesi się", a po uaktywnieniu ponownie wykona SUSMA. Jeśliby pomiędzy wykonaniami rozkazów SUSMA i SUSAR przekazano sterowanie do członu Y, to operacja *signal* (patrz poniżej), wykonana w członie Y, ustawiłaby wskaźnik MM_X i rozkaz SUSAR nie „zawiesiłby" członu X.

b) Przypadek przeciwny (zawartość słowa o adresie MUTEX + 1 równa zeru). Żaden człon nie operuje na wspólnym zasobie danych, zatem człon X zmienia zawartość słowa o adresie MUTEX + 1 na niezerową i wykonuje dalej sekcję krytyczną, omijając rozkaz skoku do etykiety SUS.

Z powyższego wynika, że w operacji signal należy wyzerować słowo o adresie MUTEX + 1, a następnie uaktywnić (za pomocą ekstrakodu AUTO) jeden lub więcej — w zależności od sytuacji — członów oczekujących na zwolnienie zasobu. W najprostszym przypadku realizowane jest to przez wykonanie ekstrakodu AUTO dla wszystkich innych członów korzystających ze wspólnego zasobu.

Z kolei przedstawiony zostanie przykład wykorzystania operacji *wait* oraz *signal* do realizacji prostej warunkowej sekcji krytycznej [4, 8]. Załóżmy, że człon o numerze 0, który jest nadawcą komunikatów (porcji informacji), został napisany jako segment o nazwie SEGA, natomiast człon o numerze 1, będący odbiorcą komunikatów, jest segmentem o nazwie SEGB, co odpowiednio opisano w dyrektywach # PROGRAM tych segmentów. W obszarze wspólnej pamięci dolnej zadeklarowano obszar MUTEX (2) dla semafora oraz bufor, w którym zorganizowano kolejkę cykliczną komunikatów. Schemat segmentu SEGA jest następujący (w nawiasy ujęto zapisane słownie operacje, których szczegółowa realizacja zależy od budowy i zadania programu).

```
{inicjalizacja}
CYCLE   {wytworzenie kolejnej porcji}
EN      SUSMA      Ø      MUTEX
        BRN               SU
        {jeśli bufor nie zapełniony
        skocz do          SEND}
        STOZ              MUTEX + 1
        AUTO       1      Ø
        BRN               SU
SEND    {dopisz porcję do kolejki}
        STOZ              MUTEX + 1
        AUTO       1      Ø
        {jeśli koniec, skocz do FIN}
        BRN               CYCLE
FIN     SUSAR      1
        BRN               FIN
SU      SUSAR      1
        BRN               EN
```

Segment SEGB działa analogicznie, to znaczy sprawdza, czy bufor jest niepusty (jeśli pusty, to zawiesza się poza sekcją krytyczną), pobiera kolejną porcję danych, opuszcza sekcję i opracowuje komunikat. Sprawdza, czy występują określone warunki zakończenia pracy i jeśli je stwierdzi, to zatrzymuje program.

Łatwo zauważyć, że powyższy schemat można rozszerzyć na typowy model programu: wczytywania, przetwarzania i zapisywania danych.

Zastosowanie programów wieloczłonowych

Konsekwencją tego, że w danym systemie komputerowym działa jeden centralny procesor (a jest to sytuacja najczęściej spotykana w systemie ODRA 1300), jest możliwość równoczesnego wykonywania tylko operacji wejścia-wyjścia i jednego procesu przetwarzania. W przypadku, gdy kolejne partie danych są przesyłane i przetwarzane z jednakową prędkością, podwójne buforowanie jest wystarczającym narzędziem usprawniającym program. W praktyce najczęściej jednak prędkości przesyłania i przetwarzania są zmienne. W takim przypadku elastyczność metody bufora cyklicznego obsługiwanego przez niezależny subprogram jest zaletą decydującą o jakości programu. W szczególności istotne zwiększenie efektywności przetwarzania uzyskuje się stosując tę metodę dla programów realizujących znaczną liczbę przesłań. Podział programu na współbieżnie działające człony jest również niezbędny przy obsłudze zwrotnych urządzeń wejścia-wyjścia, działających w trybie odpowiedzi bezpośredniej.

Metoda programów wieloczłonowych jest zatem skutecznym sposobem zwiększania efektywności wykorzystania centralnego procesora przez pojedynczy program. Aby czas komputera był dobrze wykorzystany przez programy sekwencyjne (jednoczłonowe), konieczne jest równoległe wykonywanie kilku lub kilkunastu takich programów. W praktyce, przy stosowanym najczęściej wsadowym trybie przetwarzania, taka sytuacja zdarza się bardzo rzadko. Zatem również efektywność wykorzystania systemu komputerowego ODRA 1300 może być istotnie zwiększona przez zastosowanie subprogramowania.

LITERATURA

[1] Dijkstra E. W.: Cooperating sequential processes. W: Programming Languages, F. Genuys (ed), Academic Press, London, NY, 1968

[2] Dijkstra E. W.: Hierarchical ordering of sequential processes. Acta Informatica, vol 1 (1971), no 2, pp. 115—138

[3] Egzekutor E6BM. WZE ELWRO, publikacja nr 1350101, Wrocław, 1972

[4] Hansen P. B.: Podstawy systemów operacyjnych. WNT, Warszawa, 1979

[5] Hansen P. B.: The programming language Concurrent Pascal. IEEE Transactions on Software Engineering, vol 1 (1975), No 2, pp. 199—207

[6] Liszyński Z.: SOI Politechniki Poznańskiej (informacja prywatna)

[7] PLAN. Podręcznik programowania. WZE ELWRO, publikacja nr 13004, Wrocław, 1970

[8] Ślusarek M.: Programy współbieżne w systemie ODRA 1300. Subprogramowanie. Raport Instytutu Informatyki Uniwersytetu Jagiellońskiego, Kraków, 1980

[9] Techniczny podręcznik programisty. WZE ELWRO, publikacja nr 137601, Wrocław, 1976

LUDWIK LIBURA

Biuro Projektowo-Technologiczne
Morskich Stoczni Remontowych
Gdańsk

Minisystem wyszukiwania informacji o materiałach

W zakładowych kartotekach materiałowych materiał określany jest najczęściej przez: nazwę, kod cyfrowy, cenę, stan magazynowy itp. Pojęcie materiału jest tu bardzo szerokie — obejmuje takie pozycje, jak np.: wyroby hutnicze, części maszyn, farby, drewno, narzędzia, elementy elektryczne, chemikalia. W warunkach komputerowego przetwarzania danych kod cyfrowy stał się podstawowym identyfikatorem materiału. Poszczególnym członom takiego kodu przypisuje się odpowiednie znaczenia. Przykładowo:

xx x ... xx .. xx ..
a b c d

gdzie: a — gałąź, b — branża, c — rodzaj wyrobu (blacha, rura, pręt itp.), d — inne cechy materiału.

Dla technicznego przygotowania produkcji w Gdyńskiej Stoczni Remontowej opracowano minisystem wyszukiwania informacji o materiałach oparty o ich nazwy. Aby było możliwe użycie nazwy materiału jako jego identyfikatora konieczne jest uściślenie jej formy. Dla systemu uściślono zasady budowy, kontroli i porządkowania nazw materiałów. Zasady te obowiązują tak dla nazw materiałów, jak i nazw wybierających materiały.

Prawidłowa nazwa materiału powinna zaczynać się rzeczownikiem. W praktyce można spotkać liczbę mnogą, pojedynczą i formy zdrobniałe rzeczownika, np. śruba, śruby, śrubka, śrubki. W nazwie materiału rozróżnia się rdzeń i końcówki, które nie muszą być zgodne z odpowiednikami gramatycznymi. Jednoznaczne nazwanie materiału wymaga dodania do rzeczownika pewnej liczby określeń, np.:

SRUBA
SRUBA STALOWA
SRUBA STALOWA M5
SRUBA STALOWA M5×20
SRUBA STALOWA M5×20 Z LBEM SZESCIOKATNYM.

Jak widać, ze wzrostem liczby określeń rzeczownika maleje zakres zbioru oznaczonego nazwą, co odgrywa dużą rolę przy wyszukiwaniu informacji. Dla powyższych przykładów można opracować wzór syntaktyczny, którego kolejne człony byłyby następujące: rzeczownik, materiał (jako surowiec), rodzaj gwintu, długość i kształt łba. Każdy człon takiego wzoru wyraża inny aspekt semantyczny. Opracowując wzór nazw dla poszczególnych grup materiałów możemy uporządkować jego człony. Nazwa uporządkowana zbudowana jest zgodnie ze wzorem przyjętym dla danego rzeczownika. Dla podanego przykładu nazwa uporządkowana przyjmuje postać:

SRUBA STALOWA M5 20 SZESCIOKATNY

Nazwy zawarte w kartotece materiałowej, jak i nazwy używane do wybierania — nie zawsze spełniają stawiane im wymagania. Aby sformalizować ich postać przekształca się je automatycznie według określonych algorytmów. W opisywanym systemie przyjęto następujące zasady tworzenia nazw uporządkowanych:

- jeżeli materiał ujęty jest przez normę, to należy bezwzględnie stosować się do niej — szczególnie w odniesieniu do oznaczeń
- rzeczownik stoi na pierwszym miejscu w nazwie
- nazwy materiałów, których rzeczowniki mają ten sam rdzeń, buduje się wg jednego wzoru syntaktycznego; ilość członów ograniczono do czternastu, a na każdy człon przeznaczono 16 znaków, co daje maksymalnie 224 znaki w nazwie
- występujące przy rzeczowniku określenia materiału ustawia się w taki sposób, że każde określenie dotyczy węższego zakresu materiałów
- wybrane grupy synonimów zastępuje się przez określone wyrażenia
- nazwy materiałów w zbiorze porządkuje się zgodnie z alfabetem.

Zasada ustawiania rzeczownika na pierwszym miejscu musi być przestrzegana również dla nazw źródłowych w kartotece. W stosunku do źródłowych nazw wybierających mogą być od niej odstępstwa. Przebieg porządkowania sterowany jest specjalnymi parametrami. Wykorzystuje się tu zbiory rdzeni, końcówek, symbole stali i stopów kolorowych, przyjęte wyrażenia dla poszczególnych określeń rzeczownika itp.

Nazwę, która ma służyć do wybierania materiału zaczyna się od rzeczownika, a dalsze określenia pisze się w dowolnej kolejności. Nazwa wybierająca, napisana przez użytkownika, analizowana jest przez system i gdy odpowiada ona określonym wymaganiom, zostaje przekształcona w formę umożliwiającą jej porównanie z nazwami w kartotece. Dla ułatwienia wybierania wprowadzono również możliwość zapisu nazwy materiału bez rzeczownika — w sposób uproszczony, np. zapis „ST4S = 5" oznacza blachę stalową o gatunku St4S i grubości 5 mm. W tym przypadku pełna nazwa ma postać:

BLACHA STALOWA ST4S GRUB.5MM.

System pozwala też na stosowanie symboli zastępczych, np. „PRPL" zamiast „PRET PLASKI".

Jak wspomniano wyżej — kolejność określeń w nazwie może być dowolna i tak np.

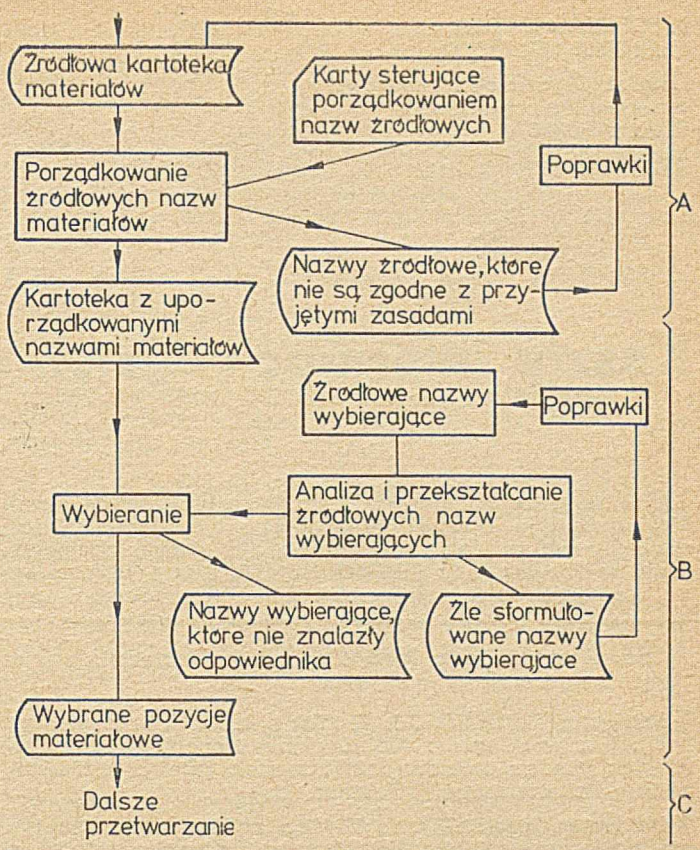
SRUBA STALOWA M5 i SRUBA M5 STALOWA

są nazwami równoważnymi (poszczególne określenia rozdzielane są co najmniej jednym odstepem — spacją).

Pierwszym warunkiem podobieństwa dwóch nazw materiałów jest pokrywanie się obu rzeczowników. Podobieństwo orzekane jest w oparciu o cechy formalne. Dwa rzeczowniki uznajemy za jednakowe, gdy mają ten sam rdzeń i dopuszczalne końcówki przynależne do tego rdzenia. Na przykład: dla rdzenia „SRUB" mogą być przyporządkowane końcówki: —A, —Y, —KI, —KA. Zatem rzeczowniki „SRUBA", „SRUBKI" uznajemy — w myśl powyższych zasad — za równe.

Drugim warunkiem tego podobieństwa jest natomiast równoznaczność określeń występujących w porównywalnych nazwach. Liczba określeń w nazwie z kartoteki musi być co najmniej równa liczbie określeń w nazwie wybierającej lub od niej większa. Podobieństwo między dwoma określeniami może polegać na identyczności lub relacji inkluzji, np. „GRUB" i „GRUBOSC", gdzie „GRUB" zawiera się w wyrażeniu „GRUBOSC". Dwa określenia będą jednakowe, gdy są synonimami (np. „WYMIAR" i „SREDNICA" w odniesieniu do przedmiotów okrągłych).

Nazwę z kartoteki, którą uznano za podobną do nazwy wybierającej, przyjmujemy jako nazwę wybraną.



Rys. Ogólny schemat funkcjonowania minisystemu wyszukiwania informacji o materiałach
A — część porządkująca nazwy źródłowe z kartoteki materiałów
B — część wybierająca
C — dalsze przetwarzania informacji o wybranych pozycjach materiałowych

Operując liczbą określeń rzeczownika w nazwie wybierającej możemy spowodować wyszukanie za jej pomocą jednego lub więcej materiałów. Chcąc wybrać wszystkie blachy — napiszemy tylko rzeczownik „BLACHA"; jeżeli będziemy chcieli wybrać wszystkie blachy miedziane, niezależnie od grubości — napiszemy „BLACHA MIEDZ". Tak wybrane zbiory możemy traktować jako wejściowe do dalszego przetwarzania.

Powyższe funkcje omawiany minisystem realizuje automatycznie dzięki odpowiednim parametrom sterującym. Przy uznaniu nazwy materiału jako jego identyfikatora istnieje możliwość porównania kartotek, które mają różne metody budowy kodu cyfrowego. Dodatkową zaletą posiadania kartoteki, z uporządkowanymi nazwami materiałów, jest możliwość drukowania informacji o nich w formie katalogu. Wyszukiwanie informacji o materiałach jest ułatwione, gdyż nazwy są kodami zorientowanymi na człowieka. Kod cyfrowy dogodny dla maszyny służy do przetwarzania w innych celach.

System jako całość składa się z części porządkującej i wybierającej. W części pierwszej dokonuje się porządkowania nazw materiałów z kartoteki, wg określonych wzorów budowy, oraz likwiduje się niektóre synonimy. Część druga realizuje jego główną funkcję, tj. wybieranie. Ogólny schemat działania systemu ilustruje rysunek.

Programy systemu napisano w języku PLAN-3. Pracują one pod kontrolą systemu operacyjnego GEORGE-3 na maszynie ODRA 1305. Informacje wyszukujące wprowadzane są wsadowo.

MIECZYSŁAW DOBIJA
KAZIMIERZ KOLARZYK
Ośrodek Organizacji i Informatyki
Przemysłu Petrochemicznego
PETROINFORM
Kraków

Koszty informacji systemowej

Główne prace PETROINFORMU, jako branżowego ośrodka informatyki i organizacji, wiążą się z doskonaleniem systemów informacyjnych w przedsiębiorstwach zgrupowanych w Zjednoczeniu PETROCHEMIA. Prace te polegają na rozwijaniu mocy obliczeniowych i teletransmisji oraz na tworzeniu i wprowadzaniu jednolitych systemów.

Obecne wyposażenie sprzętowe ośrodka stanowią następujące urządzenia:

- komputer HONEYWELL H-3200 o szybkości ok. 200 tys. operacji/s, w zestawie: 256 K pamięci operacyjnej, 13 jednostek pamięci taśmowej, 8 jednostek pamięci dyskowej, 3 drukarki wierszowe, 2 czytniki kart
- komputer HONEYWELL H-64 o szybkości ok. 600 tys. operacji/s, w zestawie: 256 K pamięci operacyjnej, 2 jednostki pamięci taśmowej, 3 jednostki pamięci dyskowej, 1 czytnik kart, 1 drukarka wierszowa
- 12 instalacji do przygotowania i transmisji danych z szybkością od 2400 do 4800 Bd.

W zakresie eksploatacji systemów informatycznych osiągnięto stan charakteryzujący się pewną równowagą między podstawowymi potrzebami a istniejącymi możliwościami obliczeniowymi. PETROINFORM dysponuje obecnie zestawem systemów informatycznych, które obsługują istotne dziedziny działalności przedsiębiorstw Zjednoczenia. Zestawienie tych systemów zawiera tabela 1.

Doświadczenia zebrane w okresie dziesięciu lat działalności PETROINFORMU, zwłaszcza w zakresie funkcjonowania wymienionych systemów, pozwalają podjąć próbę oceny efektywności informatyki w Zjednoczeniu PETROCHEMIA. W tym celu dokonano porównania kosztu informatycznego uzyskiwania informacji z kosztem wytworzenia tej samej informacji przy zastosowaniu tradycyjnych środków, tzn. przy zatrudnieniu pracowników do wykonania identycznego zakresu zadań w technologii pracy ręcznej. Porównanie takie polegało na ocenie zmniejszenia pracochłonności w wyniku zastosowania rozwiązań informatycznych oraz łącznego kosztu działalności informatycznej.

W związku z tym należy podkreślić niektóre istotne aspekty funkcjonowania PETROINFORMU, uwzględnione w modelu stanowiącym wzorzec dla ewentualnych porównań i uogólnień, a mianowicie to, że ośrodek ten:

- jest przedsiębiorstwem na własnym rozrachunku gospodarczym, co powoduje konieczność prowadzenia odpowiedniej ewidencji zdarzeń gospodarczych; m. in. ewidencja sprzedaży usług informatycznych jest również objęta systemem informatycznym
- świadczy kompleksowe usługi w zakresie informatyki; tworzy nie tylko branżowe systemy informatyczne, ale również wyposaża w sprzęt informatyczny przedsiębiorstwa podległe Zjednoczeniu PETROCHEMIA oraz zapewnia im pełny zakres usług konserwacyjno-remontowych
- współpracuje z Zakładowymi Ośrodkami ETO (ZOETO), które przygotowują dane i rozprowadzają wyniki obliczeń oraz współuczestniczą we wdrażaniu systemów informatycznych.

Stąd zastrzeżenie, że uzyskane wyniki porównania kosztów można ewentualnie zastosować tylko do organizacji działających na podobnych zasadach.

Tabela 1. Zestawienie systemów informatycznych eksploatowanych w Zjednoczeniu PETROCHEMIA

Symbol systemu	Nazwa systemu
RPZ	Realizacja Produkcji i Zbytu
GM	Gospodarka Materiałowa
GKP	Gospodarka Kadrowo-Płacowa
GST	Gospodarka Środkami Trwałymi
F-K	Finansowo-kosztowy
KOT	Kierowanie Obrotem Towarowym w AGROCHEMIE i WĘGLOPOCHODNYCH
Raport	Raporty o produkcji
BEN	Gospodarka Aparaturą Kontrolno-pomiarową
GSB	Gospodarka Sprzętem Budowlanym (NAFTOBUDOWA)
PPW	Planowanie produkcji (obliczenia optymalizacyjne)
CRD	Centralna Rejestracja Danych (Gdańskie Zakłady Rafineryjne)
SEM	System eksploatacji majątku trwałego
SEWINTE	System INTE
SEWIP	System informacji patentowej

Na całkowity koszt techniki informatycznej składają się następujące elementy kosztów:
— pracy komputerów (EMC)
— pracy urządzeń teletransmisji (T-T)

Dr MIECZYSŁAW DOBIJA ukończył w 1968 r. studia matematyczne na Uniwersytecie Jagiellońskim. W latach 1968—1973 pracował w Zakładach Chemicznych OŚWIĘCIM, projektując i wdrażając systemy informatyczne. Od 1974 r. pracuje w PETROINFORMIE, gdzie obecnie pełni funkcje Głównego Analityka. W 1975 r. uzyskał stopień doktora ekonomii z zakresu statystyki matematycznej.

Dr inż. KAZIMIERZ KOLARZYK ukończył w 1956 r. studia w Akademii Górniczo-Hutniczej w Krakowie. W roku 1972 doktoryzował się na Wydziale Wiertniczo-Naftowym tej uczelni, uzyskując stopień doktora nauk technicznych. Pracuje zawodowo w przemyśle chemicznym. Od 1974 r. kieruje Ośrodkiem Organizacji i Informatyki Przemysłu Petrochemicznego PETROINFORM w Krakowie.

— dzierżawy taśm magnetycznych (DTM)
— utrzymania systemów informatycznych — projektowanie, programowanie i konserwacja urządzeń (KUS)
— dzierżawy łącz teletransmisji (KLT)
— zatrudnienia personelu w ZOETO
— zużycia materiałów eksploatacyjnych — papieru drukarkowego, taśm i kart papierowych (PAP).
Źródłem dla uzyskania danych liczbowych w tym zakresie jest ewidencja sprzedaży oraz sprawozdania dla GUS.

METODA PORÓWNANIA KOSZTÓW UZYSKANIA INFORMACJI

Istotą metody porównania kosztów informatycznego i tradycyjnego uzyskania informacji jest ustalenie współczynnika Q, pozwalającego zamienić czas pracy komputerów na pracochłonność w technologii pracy ręcznej, niezbędną do wykonania obliczeń uzyskanych z systemów informatycznych. Jako jednostkę miary używaną do pomiaru pracochłonności przyjęto pracę jednego pracownika w ciągu miesiąca. Sposób przeliczania czasu pracy komputerów na liczbę pracowników zatrudnionych w ciągu miesiąca jest następujący:

$$\text{EMC [godz]} \cdot Q \frac{\left[\begin{array}{l}\text{liczba zatrudnionych}\\ \text{w miesiącu}\end{array}\right]}{\text{[godz]}} = I \left[\begin{array}{l}\text{liczba zatrud-}\\ \text{nionych w mie-}\\ \text{siącu}\end{array}\right]$$

Stosując współczynnik Q otrzymujemy zatem wielkość I, którą zamieniamy na koszt w oparciu o średni koszt zatrudnienia pracownika. Z kolei koszt ten porównuje się z kosztem obliczeń informatycznych. Przyjmuje się, że koszt zatrudnienia pracownika wynosi obecnie średnio ok. 10 000 zł. miesięcznie. Koszt ten można zastosować także do osób zatrudnionych w informatyce.

Określenie wielkości Q wskazuje na rolę, jaką współczynnik ten pełni w przedstawionej metodzie. Dlatego przedstawiamy sposób obliczania tego współczynnika dokonany na przykładach systemu Gospodarka Materiałowa oraz systemu Finansowo-kosztowego.

PRZYKŁAD 1. Obliczanie wartości współczynnika Q dla systemu gospodarki materiałowej

W eksploatowanym w PETROINFORMIE branżowym systemie gospodarki materiałowej tworzy się rozliczenia dla ewidencji zapasów magazynowych, przychodów i rozchodów materiałów, analizy zaopatrzenia, planowania zapasów i dostaw, sprawozdawczości oraz ewidencji przedmiotów nietrwałych. Obliczenie współczynnika Q oparto w tym przypadku na wynikach analizy skutków wprowadzenia fragmentu tego systemu, określonego głównie przez zakres pracy typowej sekcji księgowości materiałowej [1].

Ogólny zakres pracy takiej sekcji obejmuje:

- prowadzenie kartoteki ilościowo-wartościowej
- sporządzanie rozdzielnika kosztów
- sporządzanie noty syntetycznej
- sporządzanie sprawozdania G-11 (kwartalnie).

W tym celu pracownicy organizują odbiór dokumentów źródłowych, aktualizację indeksów z cenami oraz wycenę dokumentów. Następnie sporządzają wymienione zestawienia i ujednolicają kartotekę ilościowo-wartościową z kartoteką ilościową magazynów. Liczebność zatrudnienia w tej sekcji zależy w istotnym stopniu od liczby pozycji materiałów w magazynach, a zatem od wielkości przedsiębiorstwa.

W badanym przypadku sekcji księgowości materiałowej — w 1967 r. tzn. przed wprowadzeniem jakiejkolwiek mechanizacji, zatrudnionych było 17 osób. W okresie trzynastu lat wystąpił wzrost liczby pozycji i transakcji materiałowych, wynikający z rozbudowy kombinatu i zwiększenia się liczby asortymentów materiałów. Obecnie liczby te wynoszą odpowiednio ok. 25 tys. pozycji w kartotekach oraz ok. 30 tys. transakcji w okresie miesiąca.
Rozwinął się też znacznie system świadczeń socjalnych, wpływający na wzrost zatrudnienia, jak np. większy wymiar urlopów wypoczynkowych i macierzyńskich. Według oceny dokonanej z udziałem kierownictwa zakładu prace te wykonywałoby obecnie co najmniej 20 osób. W rzeczywistości prace te wykonują obecnie trzy osoby wspomagane przez system informatyczny gospodarki materiałowej.

Równocześnie szacuje się, że dla sporządzenia przedstawionego uprzednio zakresu informacji zużywa się miesięcznie 6—7 godzin pracy komputera. Oznacza to, że w badanym przypadku wartość współczynnika Q kształtuje się następująco:

- zmniejszenie liczby zatrudnionych wynosi $20 - 3 = 17$
- zużycie czasu komputera ok. 6,5 godz;

stąd współczynnik $Q = 17 : 6{,}5 \approx 2{,}8$.

Oznacza to, że w analizowanym systemie gospodarki materiałowej w dużym kombinacie jedną godzinę pracy komputera możemy zastąpić zatrudniając trzy osoby przez okres jednego miesiąca.

Wyniki te potwierdzają również rezultaty badania współczynnika Q w małym zakładzie, który eksploatuje jedynie system gospodarki materiałowej. Według oceny kierownictwa zakładu wprowadzenie systemu zastąpiło pracę ok. siedmiu osób. Ponieważ miesięczne wykorzystanie czasu komputera wynosi w tym przypadku ok. 3 godz., to wartość współczynnika Q wynosi: $7 : 3 \approx 2{,}3$. Dla systemu gospodarki materiałowej przyjęto więc średnią wartość współczynnika $Q = 2{,}5$.

PRZYKŁAD 2. Obliczanie współczynnika Q dla systemu finansowo-kosztowego

Wartość współczynnika Q dla eksploatowanego w PETROINFORMIE systemu finansowo-kosztowego (F-K) obejmującego całokształt problematyki rachunkowości, ustalono na podstawie analizy wyników wdrożenia systemu w Zakładach Azotowych w Puławach. Rachunek efektywności systemu przeprowadzono zarówno w aspekcie zamiany prac wykonywanych dotychczas w księgowości ręcznie na pracę wykonywaną przez komputer, jak również przy uwzględnieniu faktu, że w ujęciu tradycyjnym wiele zestawień (tabulogramów), materiałów analitycznych itp. nie było wykonywanych w ogóle, ze względu na ograniczone możliwości technik tradycyjnych.

System F-K wykorzystuje informacje z wielu innych systemów oraz obejmuje całokształt operacji księgowych (począwszy od księgowania cząstkowych operacji, aż do zestawienia danych do bilansu i rachunku wyników wraz z kalkulacją i analizą), dlatego też eliminuje on wiele zbędnych czynności wykonywanych dotąd przez komórki: księgowości kosztów, księgowości finansowej oraz działu finansowego. Eliminacja prac ręcznych i zastąpienie ich pracą komputera dotyczyła w szczególności takich czynności, jak:

- zakładanie obrotówek kont syntetycznych i analitycznych
- rejestrowanie (księgowanie) operacji
- rozliczanie kosztów produkcji
- kalkulacja kosztów
- uzgadnianie wewnętrznych ksiąg
- zamykanie i otwieranie kont na nowy rok obrachunkowy
- analiza i inwentaryzacja sald rozrachunków
- analiza kosztów produkcji i rentowności.

Wdrożenie i eksploatacja omawianego systemu na przestrzeni lat 1976—1978 wpłynęły na zmniejszenie stanu zatrudnienia w pionie głównego księgowego w ZA PUŁAWY o 18 osób. Zmniejszenie etatów nastąpiło głównie w komórce księgowości finansowej, księgowości kosztów oraz w dziale finansowym. Ponadto system daje w krótszym przedziale czasowym o wiele bogatszy serwis informacyjny niż to było możliwe w warunkach pracy ręcznej. Szczególnie na uwagę zasługują dodatkowe informacje z zakresu analizy sald oraz rozliczeń z odbiorcami, co zabezpiecza skutecznie przedsiębiorstwo przed przedawnieniami należności. Uzyskuje się również bogatszą informację z zakresu analizy kosztów poszczególnych wyrobów, z uwzględnieniem wpływu na koszty takich czynników, jak: zmiany wielkości produkcji, zmiany norm surowcowych oraz zmiany cen.

[1] Badania zostały przeprowadzone w sekcji księgowości materiałowej Zakładów Chemicznych OŚWIĘCIM

W ocenie użytkownika rozszerzenie tego zakresu prac w księgowości wymagałoby dodatkowego zatrudnienia ok. 17 pracowników, czyli łączne zmniejszenie obsady pracowników księgowości wyniosłoby 35 pracowników. Zwiększył się również serwis informacyjny dla celów rozrachunku wewnątrzzakładowego — o kształtowaniu się kosztów wg wydziałów i oddziałów oraz kosztów produkowanych wyrobów (analiza rentowności, akumulacji i zysku). Dla uzyskania dodatkowych informacji w tym zakresie należałoby zatrudnić dodatkowych pracowników również w służbach ekonomicznych w dyrekcji i na wydziałach.

Do ustalenia współczynnika *Q* przyjęto wyłącznie oszczędności etatowe dotyczące służb księgowości (35 pracowników). Łączny czas maszynowy potrzebny do eksploatacji systemu F-K wynosi średnio w miesiącu ok. 12 godzin: można w nim wyodrębnić następujące operacje (czas w godz.):

- zakładanie, aktualizacja i modyfikacja zbiorów głównych (5,0)
- rozliczanie kosztów produkcji (2,0)
- kalkulacja (0,25)
- analiza rozliczeń (0,50)
- analiza kosztów (0,50)
- wydruki (1,75)
- zabezpieczenie zbiorów (2,0)

Współczynnik przeliczeniowy *Q* jednej godziny pracy komputera na osobo-miesiące wyniesie ok. 2,9 (35 pracowników : 12 godzin).

Wartość średnia (arytmetyczna) współczynników *Q* obliczonych w obu opisanych przykładach wynosi 2,6. Zakładając, że pewne szacunki mogą być mniej dokładne ze względu na różną wydajność pracy różnych ludzi lub też inne czynniki, przyjęliśmy minimalną wartość współczynnika $Q = 2{,}3$, którą zastosujemy do przeliczenia efektów z wszystkich eksploatowanych systemów. Zauważyć należy, że postępowanie to polega na zastosowaniu współczynnika *Q* do przeliczenia łącznego czasu pracy komputerów, co pozwala oszacować ogólne zapotrzebowanie na wzrost zatrudnienia pracowników, niezależnie od faktycznych możliwości realizacji. Istniejące trudności zwiększania zatrudnienia ograniczają w praktyce tak duży wzrost. Powoduje to jednak, że system informacyjny przedsiębiorstwa zostaje ograniczony wyłącznie do elementarnych funkcji ochrony mienia, rachunkowości i sprawozdawczości. Brakuje wówczas wielu cennych informacji niezbędnych dla efektywnego zarządzania.

Zjawisko rozwoju informacji dla zarządzania obrazuje szczególnie wyraźnie praca służb zaopatrzenia, a zwłaszcza porównanie liczby informacji wykorzystywanych w latach sześćdziesiątych z analogiczną liczbą w latach siedemdziesiątych (po wprowadzeniu systemów informatycznych). Liczba ta jest różna w zależności od organizacji metod pracy służb zaopatrzenia, jednak obserwowany spadek wartości zapasów w stosunku do wartości produkcji oraz wzrost wskaźnika rotacji zapasów ma niezaprzeczalny związek z rozwojem systemu informacyjnego zarządzania. Podobna sytuacja występuje zresztą we wszystkich dziedzinach systemu informacyjnego przedsiębiorstwa.

PORÓWNANIE KOSZTÓW OBLICZEŃ INFORMATYCZNYCH Z KOSZTEM ZATRUDNIENIA PRACOWNIKÓW

Jak już wspomniano, w Ośrodku PETROINFORM istnieje zinformatyzowany system dokumentowania sprzedaży usług obliczeniowych. Główną cechą tego systemu jest automatyczne obliczanie czasu pracy komputerów, wykorzystywanych przez poszczególnych użytkowników, i automatyczne wytwarzanie faktur za te usługi. Na podstawie informacji z tego systemu opracowano zestawienie E-1, zatytułowane „Koszt informacji systemowej" (wydruk), które zawiera dane do analizy efektywności zastosowań informatycznych. Zestawienie to zawiera porównanie kosztu informacji systemowej z kosztem zatrudnienia pracowników, obliczonym przy zastosowaniu współczynnika $Q = 2{,}3$. Ostatnia rubryka zestawienia obrazuje korzyści zastosowania informatycznego przetwarzania. Zestawienie zawiera też podsumowanie przedstawiające korzyści oraz koszty w skali Zjednoczenia PETROCHEMIA.

ZPRP *PETROCHEMIA* OOITP PETROINFORM — KOSZT INFORMACJI SYSTEMOWEJ ZA M-C LIPIEC 1980 R. — Q = 2.30 — DATA 14 08 80 TAB. E-1

KOSZT PRAC INFORMATYCZNYCH

NUMER PRZEDS.		EMC	T-T	DTM	PAP	KUS	KLT	ZAT	RAZEM	RUB3*Q	KORZYSCI
01	ZCH OSWIECIM							23		155	
	ILOSC	4034.387	6705.900	114.000				10000.000		10000.000	
	CENA	58.330	1200/G	100.000				23000.00		1550000.00	
	WARTOSC	234934.51	134118.00	11400.00	76000.00	101000.00	12500.00		739952.51		750047.4
02	ZA CHORZOW							12		134	
	ILOSC	3491.472	6090.700	6.000				10000.000		10000.000	
	CENA	58.330	1200/G	100.000				120000.00		1340000.00	
	WARTOSC	203693.35	121814.00	600.00	30000.00	43000.00	90000.00		609082.35		73091.6
80	AGEOCHEM KRAKÓW							8		22	
	ILOSC	569.390	0.000	3.000				10000.000		10000.000	
	CENA	58.330	00000000	100.000				80000.00		220000.00	
	WARTOSC	33214.23	0.00	300.00	2000.00	10000.00	0.00		125514.23		94485.7
81	WEGLOPOCHODNE — GLIWICE							5		43	
	ILOSC	1119.877	1348.200	0.000				10000.000		10000.000	
	CENA	58.330	1200/G	0.000				50000.00		430000.00	
	WARTOSC	65325.96	26964.00	0.00	3000.00	13000.00	10000.00		168289.96		261710.0
	RAZEM ZJEDNOCZENIE										
	WARTOSC	195803.22	1020006.90	14200.00	339446.25	951000.00	456500.00	1580000.00	6319215.47	12920000.00	6600784.5

Q — PRZELICZNIK 1 GODZ. PRACY EMC NA OSOBO-MIESIACE
EMC — PRACA KOMPUTERA W MTY
T-T — PRACA URZADZEN TELETRANSMISJI W GODZ.
DTM — DZIERZAWA TASM MAGNETYCZNYCH W GODZ.
RUB3*Q — ROWNOWAZNIKI PRAC EMC WYKONYWANYCH TRADYCYJNIE PRZEZ LUDZI
PAP — ZUZYCIE PAPIERU
KUS — KOSZT UTRZYMANIA SYSTEMU (PROJ., PROG., SERWIS)
KLT — OPLATA ZA DZIERZAWE LACZA
ZAT — ILOSC ZATRUDNIONYCH W ZOETO

Analiza zestawienia E-1 prowadzi do stwierdzenia, że w warunkach Zjednoczenia PETROCHEMIA informatyczne uzyskiwanie informacji jest znacznie tańsze od zatrudnienia pracowników. Okazuje się też, że koszt wytworzenia informacji metodami informatycznymi będzie równy kosztowi zatrudnienia pracowników dopiero przy obniżeniu współczynnika Q do wartości ok. 1,4. Oznacza to, że wymienione koszty są zbliżone wtedy, gdy praca komputera w ciągu jednej godziny odpowiada pracy 1,4 osoby w ciągu miesiąca.

Nie oznacza to jednak, że przy $Q < 1{,}4$ informatyka stałaby się przedsięwzięciem nieopłacalnym. Nawet przy wyższym koszcie obliczeń informatycznych ogólny bilans może być dodatni, jeśli uwzględni się również inne korzyści wynikające z zastosowania systemów informacyjnych. Systemy te, rozwijane w Zjednoczeniu PETROCHEMIA, przyczyniają się dodatkowo do obniżki kosztów transportu towarów masowych, obniżki zapasów materiałów, przyśpieszenia wpływu środków finansowych, poprawy gospodarki aparaturą kontrolno-pomiarową itp. Efekty tego rodzaju znacznie przewyższają korzyści wynikające z zastąpienia pracy ludzkiej techniką informatyczną. Niemniej analiza kosztu informatyki w porównaniu z techniką ręczną jest sprawą podstawową. Systemy informacyjne można wprawdzie rozwijać różnymi środkami technicznymi, jednak przedstawione w niniejszym artykule obliczenia przekonują, że rozwój ten należy stymulować w pierwszym rzędzie przez wykorzystywanie informatyki.

Z proponowanej metody wynika, że efektywność obliczeń informatycznych jest wprost proporcjonalna do wartości współczynnika Q. Wymagane są zatem działania umożliwiające zwiększenie tego współczynnika, czyli — podwyższenie niezawodności pracy sprzętu, sprawnej organizacji i obsługi operatorskiej, jakości rozwiązań projektowo-programowych, rozpowszechnienie rozwiązań typowych oraz zwiększenie efektywnej współpracy użytkowników.

STANISŁAW BELCZAK, JANUSZ CZAJA

Instytut Informatyki
Politechnika Gdańska

Język BASIC dla mikrokomputera INTEL 8080

Rozwój zastosowań mikroprocesorów w różnych dziedzinach życia stwarza duże zapotrzebowanie na narzędzia do budowy oprogramowania. Dla większości zastosowań stosuje się obecnie języki symboliczne, które umożliwiają opracowywanie programów optymalnych pod względem zajętości pamięci i szybkości działania. Ten drugi czynnik ma szczególnie duże znaczenie przy pracy w trybie bezpośrednim.

Istnieje jednak wiele zastosowań mikroprocesorów, w których nie narzuca się tak ostrych wymagań na czas wykonywania programu, natomiast istotna jest efektywność kodowania, uruchamiania oraz użytkowania oprogramowania. Jako przykład może służyć przetwarzanie danych pomiarowych zarejestrowanych na taśmie magnetycznej lub papierowej.

W Instytucie Informatyki Politechniki Gdańskiej rozwiązywano to zadanie na różnych komputerach dla następujących urządzeń:

- rejestratorów DAS (Data Acquisition System)
- rejestratorów PRD (produkowanych obecnie przez zakłady „Kabid" pod nazwą „Cyfrowy rejestrator danych KB 5502")
- mikrokomputerowego systemu czasu rzeczywistego do badań silników wysokoprężnych.

We wszystkich wymienionych urządzeniach wyniki testowania obiektu dynamicznego są rejestrowane na nośniku zewnętrznym (taśma papierowa lub magnetyczna).

W celu sporządzenia odpowiedniego protokołu — zawierającego informację o obiekcie, warunkach jego testowania i wynikach badań — trzeba opracować zarejestrowane dane na komputerze.

Program, który będzie wykonywał te czynności, wygodnie jest napisać w języku programowania wyższego rzędu, dołączając procedurę wczytywania i przygotowania danych, nie zapisanych w standardowym formacie. Zapewnia to prostotę w użytkowaniu programu i łatwość jego modyfikacji, co ma szczególnie duże znaczenie przy sporządzaniu protokołów przez pracowników zakładu przemysłowego, którzy mają niewielkie przygotowanie informatyczne.

Zastosowanie mikroprocesorów w procesach pomiarowych pociąga za sobą wymaganie, aby przetworzenie danych pomiarowych wykonywane było również za pomocą mikrokomputera. Aby sprostać wymaganiom przemysłowego przetwarzania danych w Instytucie Informatyki opracowano interpreter języka BASIC dla mikrokomputera INTEL 8080.

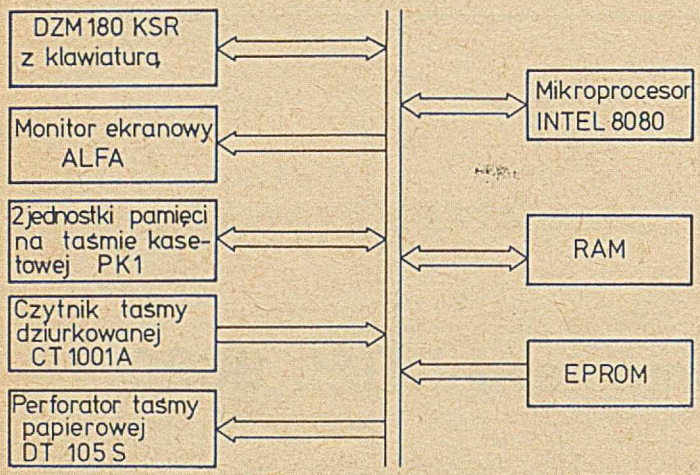
Mgr. inż. STANISŁAW BELCZAK ukończył w 1968 roku Wydział Elektroniki Politechniki Gdańskiej. Pracuje w Instytucie Informatyki Politechniki Gdańskiej, obecnie zajmuje się zagadnieniami wprowadzania technik mikroprocesorowych do aparatury kontrolno-pomiarowej w procesie automatyzacji badań.

Mgr inż. JANUSZ CZAJA ukończył w 1977 roku Wydział Elektroniki Politechniki Gdańskiej. Pracuje w Instytucie Informatyki Politechniki Gdańskiej nad zagadnieniami związanymi z budową oprogramowania użytkowego systemu sterowania próbami produkcyjnymi turbin gazowych oraz oprogramowaniem mikrokomputera INTEL 8080.

Zrealizowano w nim podstawowe elementy języka wzorcowego i pewne dodatkowe mechanizmy związane bądź to ze specyfiką mikrokomputera, bądź ze specyfiką systemu operacyjnego i konfiguracji sprzętowej, na której został uruchomiony. Umożliwia on pisanie i wykonywanie programów oraz prowadzenie obliczeń typu kalkulatorowego przez bezpośrednie wykonywanie instrukcji języka BASIC.

ORGANIZACJA INTERPRETERA

Interpreter języka BASIC został zrealizowany w systemie mikrokomputerowym o konfiguracji przedstawionej na rys. 1 i zajmuje 8 K bajtów pamięci (RAM, ROM). Jest on umieszczony w pamięci stałej ROM i wymaga zarezerwowania ok. 0,5 K bajta pamięci RAM na stos i obszary zmiennych. Wielkość potrzebnej pamięci RAM zależy od wymagań na liczbę nawiasów w linii i stopień zagnieżdżenia pętli FOR...NEXT.



Rys. 1. Konfiguracja sprzętu

W interpreterze można wyróżnić dwa tryby pracy:

- wykonywanie bezpośrednie poleceń i pojedynczych instrukcji języka BASIC wprowadzonych z klawiatury
- wprowadzanie programów z nośnika zewnętrznego (taśma magnetyczna, taśma papierowa) lub z klawiatury oraz ich interpretacja.

Instrukcje opatrzone numerem są traktowane przez interpreter jako instrukcje programu i dołączane do wstępnie przetworzonej wersji źródłowej programu, umieszczonej w pamięci RAM. Natomiast wprowadzenie z klawiatury instrukcji nie opatrzonej numerem powoduje sprawdzenie jej poprawności syntaktycznej, a następnie wykonanie. Błędy syntaktyczne lub semantyczne są sygnalizowane wydrukiem określającym typ błędu. Zawsze, gdy interpreter oczekuje na wprowadzenie kolejnej linii, można wprowadzać instrukcję programu lub instrukcję do bezpośredniego wykonania.

Interpreter umieszcza program w obszarze pamięci RAM, określonym przez programistę za pomocą zlecenia MEM. Przetwarza poszczególne wiersze na potrzeby interpretacji oraz wydruku, nie sprawdzając przy tym ich poprawności syntaktycznej. Przejście od fazy wczytywania do fazy interpretacji następuje po rozpoznaniu przez interpreter polecenia RUN lub instrukcji GOTO n (n — numer wiersza) we wczytywanym aktualnie wierszu. W fazie interpretacji pobierane są kolejno instrukcje programu, sprawdzana jest ich poprawność syntaktyczna, a następnie są one wykonywane. W momencie wystąpienia błędu syntaktycznego lub semantycznego typ błędu sygnalizowany jest odpowiednim wydrukiem, a interpreter przechodzi ponownie do fazy wczytywania.

Wykonywanie instrukcji w pierwszym trybie pracy pozwala inicjować lub zmieniać wektor stanu programu. Ma to szczególnie duże znaczenie w procesie uruchamiania programów. Po przerwaniu wykonywania programu (instrukcja STOP, błąd) można wypisać wartości zmiennych, zmienić je (jeśli istnieje konieczność) i przejść do interpretacji dalszej części programu za pomocą instrukcji GOTO n. Wykonywanie programów zapętlonych, lub wykonujących się zbyt długo, może zostać przerwane po wysłaniu określonego znaku z klawiatury.

W interpreterze istnieją również elementy pozwalające dołączyć do programu w języku BASIC wersje binarne podprogramów zapisanych np. w języku makroasemblera. Umożliwia to kodowanie w języku symbolicznym pewnych fragmentów programów nieefektywnych lub wręcz niemożliwych do zapisania w języku BASIC. Mechanizmy językowe umożliwiają przesłanie jednego parametru do (z) podprogramu binarnego, natomiast bliższa znajomość programu interpretera pozwala na wymianę dowolnej liczby danych pomiędzy programem w języku BASIC i podprogramem binarnym.

Zrealizowano następujące polecenia, ułatwiające uruchomienie i wykonanie programów.

SAVE	— zapamiętanie programu na nośniku zewnętrznym (taśma papierowa lub magnetyczna)
LOAD	— wczytanie programu z nośnika zewnętrznego
SCR	— kasowanie programu
CLEAR	— zerowanie zmiennych programu
NULL	— ustawienie lewego marginesu
LIST	— wydrukowanie programu
RES	— przenumerowanie wierszy programu
CLCRT	— kasowanie obrazu na monitorze ekranowym
BYE	— przekazanie sterowania do systemu operacyjnego.

MOŻLIWOŚCI UŻYTKOWE INTERPRETERA

Stałe i zmienne

Stałe mogą być przedstawione w postaci liczb całkowitych, dziesiętnych lub w postaci z wykładnikiem z zakresu: .1E-127 do .999999E+127.

Wartości wszystkich zmiennych i stałych są traktowane jako liczby zmiennoprzecinkowe, zajmujące 5 bajtów pamięci. Wyróżniono zmienne proste i zmienne wektorowe. Nazwa zmiennej składa się z litery lub litery i cyfry.

Repertuar instrukcji

- deklaracje: DIM (deklarowanie wektorów o rozmiarze od 1 — 10 000 elementów), DATA READ, RESTORE;
- instrukcja przypisana: LET; wyrażenie po prawej stronie instrukcji LET może zawierać zmienne, stałe, operatory arytmetyczne (+, —, *, /, ↑), funkcje standardowe oraz nawiasy ustalające porządek wykonywania działań;
- instrukcje sterujące: FOR TO STEP, STOP, END, GOTO, GOSUB, RETURN, IF THEN;
- instrukcja we/wy;

Jak wynika z rys. 1 program może wczytywać dane z 3 urządzeń wejściowych i wyprowadzać informację na 4 urządzenia wyjściowe. Do tego służą instrukcje INPUT oraz PRINT. Instrukcja INPUT umożliwia wczytanie danych zapisanych w określonym formacie, natomiast argumentem instrukcji PRINT mogą być: zmienna, wyrażenie arytmetyczne, tekst, funkcja standardowa, wyznacznik formatu drukowania liczb. Standardowo do programu jest przypisana klawiatura i monitor ekranowy. Wykonanie instrukcji LOUT lub LINT powoduje, że urządzeniem we/wy, przydzielonym do programu, jest jedno z pozostałych urządzeń (np. dla wydruku: pamięć kasetowa, perforator lub drukarka). Wybór konkretnego urządzenia dokonuje się poprzez odpowiednie polecenie do systemu operacyjnego. Za pomocą instrukcji COUT lub CINT do programu można przydzielić ponownie standardowe urządzenia. Instrukcje te wykonywane są również w przypadku przerwania działania programu;

- funkcje umożliwiające komunikację z podprogramami binarnymi: CALL, ARG;
- funkcje standardowe: ABS, COS, EXP, INT, LN, LOG, RND, SGN, SIN, SQR, TAB, TAN.

Czasy wykonywania operacji arytmetycznych i niektórych funkcji standardowych, określone doświadczalnie przy częstotliwości zegara 18 MHz, przedstawiają się następująco:

- dodawanie 1.2 ms
- mnożenie 3.0 ms
- dzielenie 9.0 ms
- SIN 40 ms
- EXP 32 ms
- LOG 45 ms.

Dołączenie do systemu mikrokomputerowego interpretera języka BASIC pozwoliło rozwiązać wiele zadań programowych bardzo małym nakładem pracy. Mocno rozbudowany mechanizm uruchamiania programów stwarza możliwości korzystania z systemu również osobom nie znającym szczegółowo tajników programowania.

Na uwagę zasługuje fakt, że program interpretera jest zbudowany modularnie, co pozwala zrealizować go na dowolnej konfiguracji sprzętowej oraz rozbudowywać o nowe instrukcje i polecenia.

LITERATURA

[1] Martin W. J.: Programowanie w języku BASIC, Politechnika Gdańska (wyd. wewnętrzne).

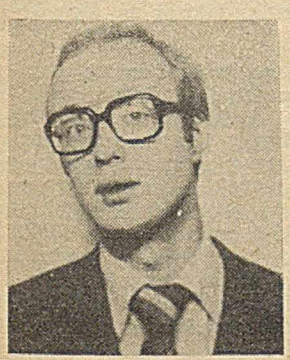
JAN CHOMICKI
Centrum Projektowania
i Zastosowań Informatyki
Warszawa

Abstrakcyjne typy danych

Błyskawiczny rozwój dziedziny baz danych stworzył wiele nowych zagadnień i problemów. Wśród nich wielką rolę odgrywa kwestia formalnego opisu i niezawodności wielkich zgromadzeń informacji. Od kilku lat toczy się dyskusja nad przydatnością rozmaitych metod specyfikacji przeznaczonych dla baz danych. Jedną z nich jest rozwijana od niedawna, wielce obiecująca technika, w której do opisu struktur danych stosuje się tzw. abstrakcyjne typy danych. Niniejszy artykuł jest próbą zgromadzenia i podsumowania różnych głosów na ten temat. Brakuje w nim rozstrzygnięć definitywnych, gdyż w dziedzinie baz danych aparat pojęciowy dość szybko się zmienia. To, co w tej chwili wydaje się nieprzezwyciężoną trudnością, może zniknąć po zastosowaniu nowego modelu danych.

Od kilku lat coraz bardziej popularne jest w informatyce pojęcie abstrakcyjnego typu danych (ang. *Abstract Data Type* — ADT). Okazało się, że jest to nadzwyczaj sprawna metoda opisu struktur danych. Naturalna wydaje się zatem próba zastosowania ADT w dziedzinie wielkich zgrupowań informacji, czyli — baz danych. Jednakże o wiele większa skala, skomplikowanie i stopień złożoności problemów baz danych sugerują, że w tym procesie mogą się pojawić trudności.

Omówienie problemu rozpoczniemy od prostego przykładu, od dobrze znanej programistom struktury stosu. Formalny opis semantyki tej struktury nastąpi jednak dopiero po wstępnych uwagach terminologicznych. Warto zastanowić się, czy ADT to istotnie nowe pojęcie. W tym celu porównamy je z bardziej rozpowszechnionymi terminami: abstrakcją, typem danych, modułem i procesem.



Mgr JAN CHOMICKI ukończył w 1979 r. studia na wydziale Matematyki Uniwersytetu Warszawskiego. Od tego czasu pracuje w Centrum Projektowania i Zastosowań Informatyki w zespole przygotowującym translator relacyjnego języka zapytań SEQUEL.

PRZYKŁAD

STOS
obiekty: stos (Stos), element stosu (Element)
operacje: połóż-element (POŁÓŻ), zdejmij-element (ZDEJMIJ), wierzchołek (WIERZCH), otwórz-stos (OTWÓRZ), czy-pusty (PUSTY)

TERMINOLOGIA

ADT i abstrakcje. Abstrakcja (abstrakcyjny) jest terminem używanym w informatyce nadzwyczaj często. Rozwój języków programowania umożliwił trzy rodzaje abstrakcji, pomocne przy budowaniu programów [7]: proceduralną (ukrycie szczegółów algorytmu w procedurze), sterowania (ukrycie detali sterowania) oraz interesującą nas tu abstrakcję danych (pominięcie szczegółów budowy lub przedstawienia danych). ADT dostarczają specyficznego typu abstrakcji danych, którą nazwać można abstrakcją realizacji. ADT bowiem to zbiór obiektów oraz zbiór operacji działających na obiektach. Poza tymi operacjami obiekty ADT są niedostępne. Abstrakcja realizacji, która pozwala nie troszczyć się o szczegóły reprezentacji obiektów i operacji, nie jest jedyną formą abstrakcji danych w informatyce; inne zostaną omówione w dalszej części artykułu.

ADT i moduły. Podane w poprzednim punkcie określenie ADT mogłoby sugerować, że są one czymś zbliżonym do modułów. Niektórzy autorzy (np. [15]) wręcz utożsamiają te dwa pojęcia. Sprawa jednak nie jest tak prosta. Moduł stanowi bowiem termin szerszy niż ADT (np. procedura jest również modułem). Pojęcia modułu używa w różnym znaczeniu niemal każdy programista. Abstrakcyjny typ danych jest terminem bardziej jednoznacznym.

ADT i typy danych. Typ danych definiuje się zazwyczaj jako klasę wartości zmiennych. W językach programowania określone są implicite zarówno fizyczne reprezentacje poszczególnych typów (np. typ „zbiór" w Pascalu jako wektor bitowy), jak i zestaw dostępnych na nich operacji. ADT byłyby uogólnieniem typów danych, umożliwiającym abstrahowanie od reprezentacji i pozwalającym programiście definiować operacje na danych, zgodnie ze swoimi potrzebami. Główną zaletą mechanizmu typów jest wzmożona kontrola semantycznej poprawności programu.

ADT i procesy. Według [13] różnią się one tylko rozłożeniem akcentów. „To, czy chcemy traktować stos jako proces akceptujący i dostarczający (w odwrotnej kolejności) dane, czy jako strukturę o specyficznym dostępie do danych elementarnych, może zależeć od rozwiązywania problemu, lecz jest zazwyczaj podyktowane określonymi zamiłowaniami i uprzedzeniami". Proces byłby zatem pewnym szczególnym sposobem widzenia (opisu) ADT, przydatnym np. przy synchronizacji działania w przypadku równoległego wykonywania ich operacji.

Podsumowując powyższe wyjaśnienia można stwierdzić, że ADT stwarzają następujące możliwości:

— abstrakcji (w sensie niezależności od reprezentacji)

— hierarchizacji problemu

— modularyzacji

— formalnej kontroli poprawności specyfikacji

— elastycznego dopasowania specyfikacji do potrzeb użytkownika.

FORMALNE METODY OPISU ADT

Definicje ADT określają zarówno składnię operacji ADT, tzn. liczbę i typ argumentów oraz typ wyniku, jak również ich semantykę, tzn. efekty działania. Dotychczas opracowano kilkanaście metod opisu ADT. Nie wszystkie znalazły zastosowanie, ze względu na swój zbyt ogólny charakter lub skomplikowanie. Omówimy specyfikację przez aksjomaty, której wariantem — wzbudzającym powszechne zainteresowanie — jest opis algebraiczny [2, 3, 4].

Opis składni operacji działających na stosie:
OTWÓRZ()→Stos
POŁÓŻ (Stos, Element)→Stos
ZDEJMIJ (Stos)→Stos ∪ {błąd (pusty-stos)}
WIERZCH (Stos)→Element ∪ {błąd (pusty-stos)}
PUSTY (Stos)→Boolean = {fałsz, prawda}

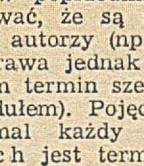
Zakłada się, że na danym obiekcie (stosie) działamy tylko poprzez operacje wymienione w definicji składni. Można zatem utożsamić obiekt z historią jego powstawania. Ponieważ jeden obiekt mógł powstać na wiele sposobów, konieczne jest więc wprowadzenie odpowiedniej relacji równoważności w zbiorze możliwych historii powstawania. Relacja ta definiowana zostaje przez zbiór równań. Równania te równocześnie wyznaczają semantykę operacji.

Algebraiczny opis semantyki operacji działających na stosie:

dla wszystkich p: Stos, e: Element,
ZDEJMIJ (POŁÓŻ (p, e)) = p,
ZDEJMIJ (OTWÓRZ) = błąd (pusty-stos),
WIERZCH (POŁÓŻ (p, e)) = e,
WIERZCH (OTWÓRZ) = błąd (pusty-stos),
PUSTY (POŁÓŻ (p, e)) = fałsz,
PUSTY (OTWÓRZ) = prawda,

Równania te określają wzajemne związki operacji. Każdemu obiektowi ADT (w naszym przypadku stosowi) odpowiada ciąg operacji POŁÓŻ o argumentach będących elementami tego obiektu. Taki ciąg nazywamy postacią kanoniczną.

Stos (a) i ciąg operacji go tworzących (b):



a)

POŁÓŻ(POŁÓŻ(POŁÓŻ(POŁÓŻ(OTWÓRZ,5)3),1),3)

b)

Postać kanoniczna jest nadzwyczaj ważna, gdyż umożliwia praktyczne konstruowanie algebraicznych specyfikacji [14]. Operacje nie tworzące postaci kanonicznej (w naszym przypadku: ZDEJMIJ, WIERZCH i PUSTY) są określane poprzez swój wpływ na postać kanoniczną argumentu. Można czytać równania od lewej do prawej strony i stosować je jako reguły redukcji napisów odpowiadających ciągom operacji do postaci kanonicznej (prawa strona równania jest zazwyczaj prostsza od lewej).

Powyżej opisana metoda nosi nazwę „algebraicznej", gdyż została sformalizowana w sposób elegancki i pełny, dzięki teorii algebr wielorodzajowych.

Specyfikacja algebraiczna struktury danych ma wiele cech nadzwyczaj pożądanych — jest formalna, minimalna (nie wprowadza zbytecznych detali), wszechstronna i elastyczna. Nietrudno też ją rozszerzyć, definiując nową operację poprzez odpowiednie reguły przekształcenia postaci kanonicznej. Natomiast rozumienie takiej specyfikacji wymaga pewnej wprawy, ponieważ przy budowaniu równań podobnie jak przy programowaniu powstawać mogą liczne błędy. Dlatego konieczne są systemy automatycznej weryfikacji specyfikacji. Taki system opracowano na Uniwersytecie Południowej Kalifornii [4]. Dla każdego definiującego operację zestawu równań generuje on pewien abstrakcyjny program w INTERLISPIE, działający na listach. Programy te mogą być następnie kompilowane i udostępniane jako moduły biblioteczne. System umożliwia interakcyjne budowanie dowodu poprawności specyfikacji.

ADT A BAZY DANYCH

Omówimy kolejno przydatność w dziedzinie baz danych (BD) tych cech ADT, które wymieniliśmy w poprzedniej części artykułu.

Abstrakcja. ADT zostały wprowadzone jako szczególnie pożyteczne narzędzia przy budowaniu oprogramowania, ponieważ pozwalały uwolnić się od szczegółów reprezentacji opisywanych obiektów i zajmować się wyłącznie ich własnościami logicznymi. ADT dostarczają możliwości pominięcia problemów realizacji. W bazach danych ADT mogłyby zatem służyć np. opisowi odwzorowania poziomu koncepcyjnego na poziom wewnętrzny, czy opisowi niezależnego od architektury sieci zachowania operacji w rozproszonych bazach danych [5]. Natomiast logika (czy semantyka) BD nie potrzebuje ADT. Rozumiemy czym jest dany przedmiot, gdy stwierdzimy, jakie ma on cechy i w jakich pozostaje związkach. Tak jest w codziennym życiu i tak jest w dziedzinie BD. Natomiast w programowaniu istotę konstrukcji rozumiemy naprawdę dopiero wówczas, gdy odniesiemy ją do czynności wykonywanych przez komputer (niekoniecznie rzeczywisty, raczej wirtualny). Pojęcie znaczenia w BD pokrywa się z pojęciem znaczenia w świecie fizycznym, natomiast w terminologii programowania znaczenie jakiejś konstrukcji — to po prostu jej funkcja. Nie było bowiem pętli „for" przed pojawieniem się ALGOLU 60; przedsiębiorstwa, biblioteki i inne obiekty pojawiły się znacznie wcześniej niż komputery.

Potrzebne w BD abstrakcje to nie tylko abstrakcje realizacji, związane z komputerową implementacją BD; to również abstrakcje ludzkiego myślenia, jakimi są uogólnienie i grupowanie.

Abstrakcja uogólnienia pozwala wprowadzić nowe pojęcie obejmujące całą klasę innych pojęć (np. „koń", „pies" i „kot" mogą być uogólnione do pojęcia „zwierzę"). Abstrakcja grupowania pozwala zastąpić jednym pojęciem zgrupowanie innych pojęć (np. „mąż" i „żona" mogą być ujęci razem jako „małżeństwo"). Abstrakcja realizacji pozwala utożsamić różne komputerowe reprezentacje jakiegoś pojęcia (np. „zbiór" może być reprezentowany jako „lista", „tablica" lub „wektor bitowy"). Te trzy rodzaje abstrakcji są niezależne i powinny być rozpatrywane osobno [11]. Sporo prac pomija różnice pomiędzy tymi abstrakcjami, co prowadzi nieraz do paradoksalnych wyników.

Przykład

Opiszemy bazę danych BAL za pomocą metody algebraicznej. Jakie elementy (typy danych) będą się na nią składać? Przyjmijmy, że „damy" (D), „kawalerowie" (K) oraz „tańczące na balu „pary" (P).

Damy. Operacje zmieniające zbiór dam na balu to:

BD()→D (brak dam)
PD (D, d)→D (pojawienie się damy d na parkiecie)
WD (D, d)→D (zniknięcie damy d z parkietu)
JD (D, d)→Boolean (czy dama d jest na parkiecie?)

Aby jakoś połapać się w tańcach dam, będziemy pisać Ob (..., d) dla zaznaczenia, że dama d jest na parkiecie. Skutki wyżej wymienionych operacji będziemy opisywać poprzez zmiany w Ob. Zbiór dam na parkiecie dl, d2, ..., dk opisywany jest za pomocą formuły

Ob(....Ob(Ob (Ø, dl), d2) ...., dk)
BD = Ø (pusty zbiór dam)
PD (Ob (D, d), dl) =
if d=dl then Ob (D, d)
else Ob (PD (D, dl), d)
(gdy dama d jest już na parkiecie to nie sposób jej zaprosić do tańca)
WD (Ø, d) = Ø (nie tańczą damy, których nie ma)
WD (Ob (D, d), dl) =
if d=dl then D
else Ob (WD (D, dl), d) (dama d znika z parkietu)
JD (Ob (D, d), dl) =
if d=dl then prawda
else JD (d, dl)
JD (Ø, d) = falsz

Kawalerowie. W naszym opisie różnią się od dam tylko nazwami operacji zmieniających ich zbiór:
— BK zamiast BD, BK()→K
— PK zamiast PD, PK (K, k)→K
— WK zamiast WD, WK (K, k)→K
— JK zamiast JD, JK (K, k)→Boolean
i dziedzinami tych operacji. Natomiast ich znaczenie jest identyczne i dlatego nie będziemy powtarzać opisu.

Pary.
BP()→P (pusty parkiet)
ZAPROSZENIE (P, d, k)→P (nowa para na parkiecie)
KONIEC (P, d, k)→P (koniec tańca damy d z kawalerem k)
PARY to zbiór par. Będziemy opisywać operacje na tym zbiorze poprzez operacje na zbiorach: D i K.
BP = PARA (BD, BK)
(pusty parkiet: brak dam i brak kawalerów)
ZAPROSZENIE (PARA (D, K), d, k) =
if JD (D, d) OR JK (K, k)
then PARA (D, K)
else PARA (PD(D, d), PK(K, k))
(na parkiecie pojawia się i dama, i kawaler)
KONIEC (PARA(BD, BK), d, k) = PARA(BD, BK)
(jeśli nikt nie tańczy, to nikt nie przestaje tańczyć)
KONIEC (PARA(PD(D, d), PK(K, k)), dl, kl) =
if d=dl and K=kl
then PARA (D, K)
else ZAPROSZENIE (KONIEC (PARA(D, K), dl, kl), d, k)
(tylko właściwa para znika z parkietu)

Podana powyżej specyfikacja bazy danych BAL jest poprawna tylko pod warunkiem, że jedynymi dostępnymi operacjami są ZAPROSZENIE i KONIEC. Inne możliwości są dla par wykluczone: dama nie może zwichnąć nogi, ani pozostawić kawalera samego na parkiecie i odwrotnie. Jeśliby dopuścić samodzielne zmiany w zbiorach dam lub kawalerów, to zostałaby pogwałcona zasada hierarchiczności: typ może używać tylko typów należących do jego reprezentacji. W naszym przykładzie oznacza ona, że operacje na zbiorach dam i kawalerów nie powinny w skutkach zmieniać zbioru par na parkiecie.

Czy nie warto zatem zrezygnować z zasady hierarchiczności albo ograniczyć zakres jej stosowania tylko do przypadków, gdy powiązanie między typami danych ma charakter abstrakcji realizacji? W naszym przykładzie oznaczałoby to, że obojętny jest sposób reprezentowania zbiorów (tablice, wektory bitowe itp.). Natomiast powiązanie między zbiorami dam i kawalerów oraz zbiorem par jest abstrakcją grupowania, do której zasada hierarchiczności się nie stosuje. W konsekwencji, aby tę abstrakcję wyrazić, należałoby rozszerzyć używany do tej pory w przykładzie aparat formalny. Do dokonania tego nie wystarcza jednak aparat algebraiczny. Potrzeba tu raczej technik używanych w zagadnieniach reprezentacji wiedzy, rozwiniętych najlepiej w dziedzinie „sztucznej inteligencji". Propozycję takiego rozszerzenia zawiera np. [1].

Hierarchizacja. Opis schematu koncepcyjnego BD musi zawierać oprócz abstrakcji (uogólnienia i grupowania) ogólniejsze relacje binarne. Np. powiązanie „jest-małżonkiem" wydaje się być, przynajmniej od pewnego czasu, powiązaniem nie hierarchicznym, lecz równorzędnym.

Modularyzacja czyli umieszczenie (pakowanie) danych razem z operacjami. Nurt programowania strukturalnego (z niego wywodzi się koncepcja ADT) preferuje rozpoczynanie rozwiązywania problemu od projektowania zestawu programów. Decyzję o strukturach danych odkłada się na moment najodleglejszy. Takie podejście znalazło skrajny wyraz w algebraicznej metodzie opisu ADT, w której strukturę danych utożsamia się po prostu z ciągiem tworzących ją operacji. Natomiast w BD najpierw mamy dużo struktur danych, powiązanych w szczególnie skomplikowany sposób. Zestaw operacji jest a priori nieznany i zwykle zmienia się z upływem czasu.

Formalna kontrola poprawności. W przypadku BD wystąpiłyby następujące problemy:

- weryfikacja schematu BD
- kontrola zgodności rzeczywistej BD ze schematem
- zabezpieczenie integralności BD.

Pewne teoretyczne rozwiązanie pierwszych dwóch problemów przyniosła dotychczas tylko praca [1]. Schemat BD w tej koncepcji to zestaw definicji typów: atrybutów, obiektów, odwzorowań i zbiorów, tworzonych na podstawie typów elementarnych poprzez operatory typów (np. suma, restrykcja). Aksjomatyczna definicja tych mechanizmów umożliwia wnikliwą kontrolę typów, wykrywającą w specyfikacji schematu BD znaczną liczbę błędów semantycznych.

Elastyczność to własność szczególnie cenna z punktu widzenia użytkownika. Jednakże płaci się za nią znacznym skomplikowaniem specyfikacji, co ze względu na rozmiar BD stanowi często zbyt wygórowaną cenę. Z tego powodu dotychczas konstruowane systemy BD opierały się na pewnym ustalonym modelu danych. Wydaje się, że na razie nieuniknione jest wyróżnienie prymitywnych struktur danych (np. relacja, powiązanie), a także możliwości ich strukturalizacji (np. poprzez grupowanie). Ważne jest tylko, aby były to mechanizmy jak najbardziej elastyczne, wszechstronne i wygodne.

Istnieją prace poświęcone algebraicznemu opisowi relacyjnego [9, 12] i binarno-relacyjnego [8] modelu danych. Sformułowane tam definicje mogą okazać się bardzo przydatne w implementacji odpowiednich systemów zarządzania BD. Natomiast jako narzędzie dla użytkownika ADT są chyba zbyt mocne, a w związku z tym również i kłopotliwe w implementacji.

W przypadku BD pojawiają się jeszcze inne problemy.

Zgodność podejść użytkowników. Nie bardzo wiadomo, czy można rozwiązać to zagadnienie inaczej niż w [1]. W tej koncepcji dokonuje się syntezy podejść (ang. *views*) i bada poprawność tak powstałego schematu.

Dziedziczenie środowiska. Praca [1] proponuje, by z różnymi metodami budowy elementów schematu BD (typów) wiązać różne reguły dziedziczenia. Np. zwykła relacja binarna nie daje możliwości dziedziczenia środowiska, natomiast powiązanie poprzez precyzowanie (np. „zwierzę"→„koń") powinno umożliwić dziedziczenie wszelkich atrybutów. „Koń" ma wszystkie własności „zwierzęcia", np. „imię", „liczbę nóg" czy „kolor".

Dotychczas omówiliśmy problemy związane z przydatnością ADT do opisu schematu BD, czyli jej aspektu s t a t y c z n e g o. Kolej na zagadnienia pojawiające się przy pracy z bazą danych (aspekt d y n a m i c z n y).

Synchronizacja. Szczególnie odpowiednia jest tu metoda pozwalająca traktować ADT jako procesy lub monitory [5]. Wówczas stosować można znane rozwiązania z dziedziny systemów operacyjnych.

Dzielenie danych. Filozofia ADT (szczególnie metody algebraicznej), lekceważąca budowę (morfologię) obiektów, bardzo utrudnia określenie wspólnego obszaru zainteresowań różnych użytkowników. Zgodnie z tą filozofią każdy ma swoje poletko, na którym coś tam robi. W metodologii projektowania takie podejście pozwala ustrzec się wielu błędów, natomiast w dziedzinie baz danych poletko rozrasta się do wielkiego pola, uprawianego przez całą grupę użytkowników. Ponadto w pracy [10] znaleźć można argumentacje uzasadniające tezę, że nawet strukturalna odpowiedniość podejść nie wyznacza jednocześnie dynamicznej interferencji użytkowników, gdyż ta zależy również od własności translacji operacji użytkownika.

Zróżnicowanie uprawnień do dostępu. Konieczne jest tutaj rozbudowanie aparatu pojęciowego, a w związku z tym i językowego, o pojęcie „uprawnienia". Praca [6] sugeruje, że można wprowadzić je do mechanizmu typów poprzez tzw. typ kwalifikowany. Definicja typu kwalifikowanego składa się z typu obiektu oraz podzbioru zbioru uprawnień dla tego obiektu. Jeślibyśmy np. chcieli ograniczyć działanie procesu p na stosie wyłącznie do odczytywania wartości wierzchołka stosu, powinniśmy zadeklarować:

p: Stos [WIERZCH]

Jak widać, naturalne jest powiązanie uprawnień z odpowiednimi operacjami wchodzącymi w skład definiujących dany obiekt ADT.

Powyższy mechanizm okazuje się bardzo elastyczny, gdyż w przypadku struktur złożonych (tablica, plik) umożliwia odróżnienie uprawnień do działania na strukturze i na jej elementach.

Efektywność. Wszechstronność i elastyczność ADT powodują znaczne kłopoty implementacyjne. Skala problemów BD praktycznie uniemożliwia zastosowanie do nich ADT. By skorzystać z zalet ADT, musiałoby ulec zmianie podejście do metodologii systemów zarządzania BD — ze „zorientowanego na schemat" na „zorientowanie na translację" [10].

* * *

ADT są świetnym narzędziem inżynierii oprogramowania. Służą zwiększeniu niezawodności programów. Wymóg niezawodności będzie prawdopodobnie odgrywał coraz większą rolę również w dziedzinie BD. Dojdzie zatem być może do głębokich zmian w metodologii systemów zarządzania BD. Inne problemy, jak współbieżność i kontrola dostępu, są wspólne dla BD i systemów operacyjnych. Im większą rolę będą one odgrywać w dziedzinie BD, tym bardziej zmniejszać się będzie metodologiczna różnica między tymi dziedzinami.

LITERATURA

[1] Brodie M. L.: Specification and verification of database semantic integrity. Tech. Rep. CSRG-91, Univ. of Toronto, April 1978

[2] Goguen J., Thatcher J., Wagner E.: An initial algebra approach to the specification, correctness and implementation of abstract data types; w: Yeh R. (ed): Current Trends in Programming Methodology. Data Structuring (Vol. IV), Prentice-Hall, 1978

[3] Guttag J.: Abstract data types and the development of data structures. Comm. ACM 20, 6, June 1977

[4] Guttag J., Horowitz E., Musser D.: Abstract data types and software validation. Comm. ACM 21, 12, December 1978

[5] Hebalkar P. G.: Application specification for distributed data base systems; w: Yao S. B. (ed): Proc. 4th Int. Conf. on Very Large Data Bases, Berlin, September 1978

[6] Jones A. K., Liskov B. H.: A language extension for expressing constraints on data acces. Comm. ACM 21, 5, 1978

[7] Liskov B., Snyder A., Atkinson R., Shaffert C.: Abstraction Mechanisms in CLU. Comm. ACM 20, 8, August 1977

[8] Lockemann P., Wohlleber W.: Semantics of a binary relational model and its data definition for the conceptual level. IB II/79, Fakultät für Informatik, Universität Karlsruhe

[9] Paolini P., Pelegatti G.: Formal definition of data models. Rap. int. 77—2. Laboratorio di Calcolatori, Politecnico di Milano

[10] Paolini P., Pelegatti G.: Formal definition of mappings in a data base. Proc. ACM SIGMOD 1977

[11] Smith J. M.: Comments on the papers by A. I. Wassermann and by H. Weber; w: Yao S. B. (ed): Proc. 4th Int. Conf. on Very Large Data Bases, Berlin, September 1978

[12] Tompa F.: A practical example of the specification of abstract data types, MCC 5/78, Departamento de Informatica, Pontificia Universidade Catolica do Rio de Janeiro

[13] Turski W. M.: Metodologia programowania. WNT, Warszawa 1978

[14] Veloso F., Pequeno T.: Don't write more axioms than you have to. Int. Computing Symposium 1978

[15] Weber H.: A Software Engineering View of Data Base Systems: w: Yao S. B. (ed): Proc. 4th Int. Conf. of Very Large Data Bases, Berlin, September 1978

COMNET'81

Węgierskie Towarzystwo Informatyczne im. Johna v. Neumana organizuje w dniach 11—15 maja br. w Budapeszcie międzynarodową konferencję na temat sieci komputerowych i systemów teleprzetwarzania. Konferencji patronują UNESCO i Międzynarodowa Federacja Przetwarzania Informacji (IFIP) oraz zainteresowane tą problematyką czołowe węgierskie instytucje i stowarzyszenia naukowe. Głównym celem imprezy jest wymiana doświadczeń na temat eksploatacji istniejących sieci i systemów jakie zgromadzono od czasu ostatniej konferencji COMNET w 1977 r. Organizatorzy pragną położyć szczególny akcent na analizę zagadnień z punktu widzenia użytkownika oraz wynikających z niej wniosków dla projektowania nowych rozwiązań w tych dziedzinach.

Program konferencji przewiduje następującą tematykę obrad:

- **korzystanie z sieci komputerowych z punktu widzenia użytkownika**
- **projektowanie i wdrażanie sieci komputerowych**
- **doświadczenia z wdrożenia i eksploatacji sieci specjalnych**
- **analiza przykładów zastosowań**
- **normalizacja i próby ujednolicania rozwiązań**
- **ogólnodostępne sieci i usługi**
- **pomiary i ocena wydajności sieci i systemów**
- **analiza architektury sieci**
- **nowe rodzaje usług w dziedzinie skomputeryzowanego przekazywania informacji tekstowych.**

Oprócz światowej czołówki naukowców, udział w konferencji zgłosiły również: Międzynarodowa Organizacja Normalizacyjna (ISO), zarządy telekomunikacji krajów przodujących w rozwoju teleinformatyki, główni producenci sprzętu teleinformatycznego oraz organizacje eksploatujące międzynarodowe sieci teleinformatyczne (EURONET, SWIFT, TRANSPAC).

Programowi konferencji towarzyszyć będą liczne pokazy praktycznego działania węgierskich i zagranicznych systemów teleinformatycznych oraz sieci komputerowych.

Oficjalnymi językami obrad konferencji będą angielski i rosyjski. Opłata uczestnictwa dla osób z krajów RWPG, obejmująca komplet materiałów konferencyjnych, wynosi 2000 forintów, jeżeli zostanie ona wniesiona w terminie do 1 kwietnia 1981 r. Po tym terminie opłata będzie wyższa o 15%. Dalsze informacje na temat konferencji można uzyskać w Sekretariacie COMNET'81 pod adresem: **P.O.B. 240, H-1368 Budapest, Hungary, tel. (361) 112-027, teleks 22-5369.**

Formularze zgłoszeń uczestnictwa oraz rezerwacji hotelu otrzymać można w naszej Redakcji (tel. 27-71-40).

JÓZEF B. LEWOC
Centrum Obliczeniowe
Politechnika Wrocławska

O zawodzie inżyniera informatyka

Rozwój środków technicznych elektroniki, a w szczególności układów scalonych (w tym mikroprocesorowych) pozwala na coraz szersze wykorzystanie informatyki w różnych dziedzinach życia. W warunkach polskich, gdzie jeszcze nie dysponujemy zbyt bogatym zestawem środków technicznych, szczególnie istotną sprawą jest umiejętność dobierania i tworzenia nowych zestawów sprzętowo-programowych według potrzeb użytkownika. Dla nowych zastosowań informatyki potrzebny jest więc zawód inżyniera, który potrafi zrozumieć potrzeby użytkownika, zaprojektować architekturę ogólną systemu cyfrowego oraz skonstruować potrzebne, a niedostępne elementy takiego systemu, dobrać i opracować niezbędne oprogramowanie podstawowe (testy i systemy operacyjne) oraz oprogramowanie użytkowe.

Oczywiście, nierealne jest postulowanie wykształcenia inżynierów informatyków, którzy mogliby konkurować z zawodowymi programistami czy konstruktorami sprzętu. Istnieje natomiast potrzeba wykształcenia inżynierów, którzy dobrze rozumieją problemy wszystkich grup specjalistów zaangażowanych w nowe zastosowania informatyki — od użytkownika, poprzez serwis, konstruktorów sprzętu, programistów, aż do matematyka opracowującego metody potrzebne dla takich zastosowań.

I tu trzeba stwierdzić, że profil kształcenia inżynierów informatyków w znanym mi dobrze środowisku wrocławskim niestety nie odpowiadał tej potrzebie. Początkowo (lata sześćdziesiąte) inżynierów o specjalności „Maszyny matematyczne" kształcono na Wydziale Elektroniki (wcześniej Wydział Łączności). Obsolwenci otrzymali dobre podstawy w zakresie projektowania sprzętu, ale nie uzyskali dostatecznie szerokiego zakresu wiadomości w dziedzinie oprogramowania. Po reorganizacji Uczelni program nauczania na Wydziale Elektroniki uległ zmianie i obecnie — na przykład — nie obejmuje w ogóle zagadnień projektowania urządzeń i maszyn cyfrowych! [1]. Na nowym Wydziale Informatyki i Zarządzania główny nacisk był położony na zagadnienia programowania. Dopiero niedawno opracowano program, który obejmuje zagadnienia projektowania sprzętu i programowania [2].

Dotychczas nie kształcono więc inżynierów, którzy mogliby dobrze rozumieć zagadnienia sprzętowe i programowe oraz merytorycznie kierować wdrażaniem informatyki w różnych dziedzinach życia. Skutki braku kompetentnego, merytorycznego kierownictwa różnych pilotowych zastosowań zostaną przedstawione na konkretnych przykładach w dalszej części niniejszego artykułu.

Dr inż. JÓZEF B. LEWOC dwukrotnie uzyskał tytuł magistra: w 1967 r. na Wydziale Elektroniki Politechniki Wrocławskiej oraz w 1971 r. na Uniwersytecie Wrocławskim. W 1973 r. uzyskał w Politechnice Wrocławskiej tytuł doktora. W 1966 r. podjął pracę w WZE ELWRO. W latach 1971—1979 był zatrudniony w Instytucie Automatyki Systemów Energetycznych. Obecnie pracuje w Centrum Obliczeniowym Politechniki Wrocławskiej, gdzie kieruje pracami nad przygotowaniem podsieci komunikacyjnej dla Międzyuczelnianej Sieci Komputerowej MSK. Od marca ub. r. jest sekretarzem Kolegium Sekcji Maszyn i Systemów Cyfrowych przy Oddziale Wrocławskim SEP.

SKUTKI BRAKU INŻYNIERÓW INFORMATYKÓW

Liczne nieudane wdrożenia informatyki w przemyśle, medycynie, energetyce itp. spowodowane były — moim zdaniem — brakiem kompetentnej kontroli merytorycznej tego rodzaju przedsięwzięć. Miałem bezpośrednią styczność z wieloma takimi nieudanymi przedsięwzięciami, z których najbardziej drastyczne postaram się poniżej scharakteryzować.

Pierwsze z nich — to **uniwersalny system komputerowy do zastosowań przemysłowych, opracowany na maszynę ODRA 1204 [3]; brak głównego inżyniera informatyka o odpowiednich kwalifikacjach spowodował błędną, a w konsekwencji nieudaną realizację zadania.** Przy braku jakichkolwiek doświadczeń z nowym komputerem (1967 r.), a także doświadczeń w zakresie zastosowań komputerowych systemów czasu rzeczywistego, próbowano zbudować system dla komputera, a nie dla faktycznych potrzeb praktyki przemysłowej. Dlatego też system ten nie wyszedł nigdy poza deski kreślarskie.

Drugie przedsięwzięcie — to **system śledzenia materiału w walcowni prętów [4]; podczas realizacji tego pilotowego w skali kraju zastosowania polskiego sprzętu komputerowego w przemyśle popełniono szereg błędów technicznych.** Po pierwsze — część projektantów walcowni nie wierzyła w sensowność i możliwość skutecznego wdrożenia komputerowego systemu śledzenia przepływu materiału. W wyniku tego nie zwrócili oni dostatecznie dużej uwagi na zapewnienie możliwości automatycznego wykrywania przejścia materiału z jednego stanowiska na następne. Po drugie — na obiekt wysłano system faktycznie nie uruchomiony, nie mówiąc już o pełniejszym jego przebadaniu. A było to przecież pierwsze zastosowanie przemysłowe nowego wówczas komputera ODRA 1204, do którego trzeba było opracować pełny zestaw urządzeń do współpracy z obiektem oraz pełne oprogramowanie. W wyniku tego początkowe prace uruchomieniowe, które mogły — moim zdaniem — trwać około trzech miesięcy w laboratorium producenta, trwały na obiekcie około roku. Przedłużające się prace uruchomieniowe wywołały uzasadnione obawy użytkowników dotyczące kompetencji zespołu wdrażającego system śledzenia, a więc również przydatności tego systemu.

Dalsze sprawy były prostą konsekwencją tych obaw. Opierając się na argumencie, że warunki pracy w walcowni są różne od projektowanych, kilkakrotnie wprowadzono zmiany podstawowych założeń systemu. Każda taka zmiana prowadziła do około półrocznej pracy, po czym nie dokonywano prób odbioru, a wprowadzano nowe założenia. Pracownicy, którzy mieli przejąć serwis techniczny, nie zadali sobie trudu poznania systemu, a szukali formalnych pretekstów do jego odrzucenia. Doszło nawet do tego, że nie poinformowali o własnej produkcji dobrych czujników fotoelektrycznych — w sytuacji, gdy fotoprzekaźniki używane w systemie śledzenia były jego najmniej pewnie działającym elementem. Z tych powodów system nie został wdrożony do eksploatacji mimo ostatecznego pozytywnego wyniku prób odbiorowych.

Inny symptomatyczny przykład to przygotowywane od 1972 r. **systemy komputerowe dla okręgowych dyspozycji mocy (SAPI ODM) [5, 6]; od początku do końca brak**

było merytorycznego kierownika o odpowiednich kwalifikacjach i kompetencjach. Nastąpiła dekompozycja zadania na autonomiczne zadania cząstkowe: przygotowanie uzupełnień sprzętu, kompletacja sprzętu, przygotowanie systemu operacyjnego oraz przygotowanie oprogramowania użytkowego, przy czym całym przedsięwzięciem nie kierował merytorycznie inżynier informatyk o odpowiednich kwalifikacjach.

W efekcie ośrodek badań automatyki systemowej (OBAS), utworzony w celu laboratoryjnego uruchomienia i badania systemów komputerowych dla energetyki, stał się od początku ośrodkiem obliczeniowym pracującym głównie na potrzeby instytutu prowadzącego prace projektowe. Ściągnięty do tego instytutu główny konstruktor urządzeń przemysłowych SMA (co znacznie utrudniło uruchomienie produkcji tych urządzeń) został skierowany do projektowania konfiguracji ogólnych (na czym się nie znał), a pracami nad przygotowaniem prototypowego zestawu BAZI (urządzenia przemysłowe wzorowane na SMA) kierował inny pracownik, nie posiadający doświadczenia w tej dziedzinie. Zestaw BAZI, mimo że został wykonany wcześniej niż model SMA, nigdy nie został wykorzystany.

Uruchomiony zestaw SMA był dostępny na miejscu dla programistów przez jeden tydzień. Argumentem za jego przewiezieniem na obiekt było to, że należy sprawdzić kanał przemysłowy w zainstalowanej tam maszynie cyfrowej ODRA 1325. Zatem — w celu sprawdzenia dwóch płytek o wartości kilkunastu tysięcy zł — wysłano do pierwszego ODM zestaw sprzętu o wartości kilku milionów zł, uniemożliwiając skuteczne uruchomienie systemu operacyjnego oraz oprogramowania użytkowego, a także zbadanie prototypowego zestawu sprzętowego. Sprzęt stał bezużytecznie pół roku. Po tym czasie wysłano ekipę do uruchomienia oprogramowania. Ekipa ta wróciła z niczym, bo okazało się, że nie podłączono wcześniej układów obiektu z SMA. Takie postępowanie musiało wzbudzić nieufność użytkowników. W rezultacie system w pierwszym ODM zdemontowano w 1975 r. po rocznych nieudanych próbach wdrożeniowych. Dopiero w końcu 1979 r. rozpoczęła się eksploatacja systemu zainstalowanego w drugim ODM (również przekazanego bez odpowiednich badań).

Jedyną udaną instalacją był system trzeci, zainstalowany w ODM Warszawa. Tam dopiero — dzięki splotowi sprzyjających okoliczności, głównie zaangażowaniu przedstawiciela użytkownika, a nie planowemu działaniu instytutu przygotowującego SAPI — udało się osiągnąć zamierzony cel: od 1978 r. system pracuje i okazał się bardzo cennym narzędziem wspomagającym dyspozytorów w czasie zimowego niedoboru energii na przełomie lat 1978//1979. Sukces ten nie może przesłaniać faktu, że z planowanych i — moim zdaniem — realnych sześciu systemów wdrożono tylko dwa i to dopiero w kilka lat po zaplanowanym terminie.

Dalszy przykład to **uniwersalny system dla maszyny cyfrowej ODRA 1325 z urządzeniami wg norm CAMAC** [7]. Jako przykład zastosowania miał być realizowany centralny rejestrator dla bloku energetycznego 200 MW. **Brak rozsądnej oceny potrzeb i możliwości sprzętowo-programowych doprowadził do tego, że wieloosobowy zespół prowadził przez ponad rok prace, by stwierdzić w końcu, że rozwiązanie systemu ODRA 1325/CAMAC są znacznie gorsze dla planowanego zastosowania, niż opanowanego wcześniej systemu ODRA 1325/SMA.** Notabene system ODRA 1325/SMA [8] został zainstalowany w elektrowni ze znacznym opóźnieniem, wynikającym m. in. z braku wykwalifikowanych programistów.

Kolejne przykłady — to **cyfrowy symulator bloku energetycznego [9] i symulator sieciowy [10] — oba jako urządzenia ćwiczebne dla celów szkolenia, odpowiednio: obsługi bloku i dyspozytorów systemu elektroenergetycznego.** W pierwszym przypadku utworzono zespół ludzi z wielu instytucji, każdej odpowiedzialnej za odpowiedni wycinek zadania (sprzęt, system operacyjny, programy koordynacyjne, modele części bloku, programy szkolenia). **W wyniku braku kompetentnego kierownika całego zadania (mógłby nim być tylko inżynier informatyk) wycinki zadania stały się zadaniami samymi w sobie, a wieloletnia praca kilkunastoosobowego zespołu poszła na marne.** Drugi system (stymulator sieciowy) jest realizowany na innych zasadach. Całość prac związanych z realizacją wykonuje jedna instytucja. Jeśli uda się tam doprowadzić do rozsądnego merytorycznego kierowania całym zadaniem przez jedną, kompetentną osobę (inżyniera informatyka), to istnieje szansa uzyskania bardzo użytecznego narzędzia do szkolenia dyspozytorów.

Kolejnym przykładem może być **emulator jednostki sterującej monitorów ekranowych systemu ODRA 1300, zrealizowany na minikomputerze systemu MERA-300 [11]. Techniczny sukces, jakim było wykonanie i uruchomienie emulatora nie może przesłaniać faktu, że nie jest on w pełni wykorzystywany przez użytkowników systemów ODRA 1300.** Błąd polegał w tym przypadku na utracie z pola widzenia bezpośredniego użytkownika i na zbyt słabej kontroli najważniejszego dla użytkownika elementu: obrazu na ekranie monitora.

Proszę zauważyć, że nie próbuję tu zarzucić własnych błędów na inne osoby. W każdym z tych nieudanych przedsięwzięć brałem udział, niektórymi kierowałem merytorycznie, lub mogłem i powinienem był kierować z lepszym skutkiem. A wszędzie popełniłem błędy prowadzące w znacznej mierze do negatywnych wyników wdrożeniowych. Sądzę, że powyższe przykłady dobrze ilustrują potrzebę istnienia kompetentnego kierownika merytorycznego dla przygotowania, uruchamiania i wdrażania systemów komputerowych czasu rzeczywistego. W dalszej części spróbuję scharakteryzować kwalifikacje, jakie powinien posiadać taki kierownik.

KWALIFIKACJE KIEROWNIKA

Spróbujmy naszkicować przykład zespołu wdrażania techniki komputerowej w walcowni prętów (tabela). Zespół taki powinien wspólnie określić cel, dla jakiego ma się wdrażać system komputerowy. W walcowni prętów (szczególnie wytwarzającej dużą liczbę małych partii z różnych gatunków stali) sprawą istotną jest śledzenie przepływu materiału i odpowiednie kierowanie kolejnymi fazami procesu technologicznego.

Tabela. Zespół do wdrażania techniki komputerowej w walcowni prętów

Osoba	Funkcja w zespole
Dyspozytor walcowni	ocena funkcjonalności układu śledzenia
Przedstawiciel kierownictwa walcowni	określenie celów, jakie ma spełniać układ śledzenia
Przedstawiciel serwisu układu śledzenia	zaznajomienie się z budową i działaniem układu śledzenia oraz jego ocena pod kątem konserwacji i remontów
Projektant walcowni	przystosowanie projektu walcowni do zastosowania układu śledzenia oraz zalgorytmizowanie procesu śledzenia materiału
Projektant systemu komputerowego	zaprojektowanie sprzętu i oprogramowanie na podstawie celów, ocen i algorytmów określonych przez pozostałych członków zespołu. Uruchomienie i wdrożenie układu śledzenia. **Uwaga. Projektant ma prawo rozszerzać cele i korygować algorytmy procesu śledzenia.**

Zauważmy, że wdrożenie systemu komputerowego jest celem głównym działania tylko dla projektanta tego systemu, tak jak np. podstawowym celem działania dyspozytora jest prawidłowe prowadzenie procesu technologicznego. Zatem projektant systemu komputerowego powinien być tu kierownikiem merytorycznym (głównym konstruktorem), który wykonuje nie tylko swoje zadanie techniczne, ale również kontroluje i kieruje pracą całego zespołu. Kwalifikacje głównego konstruktora powinny wynikać z zadań, jakie ma on spełniać we wdrażaniu systemu komputerowego.

W zakresie współpracy z wymienionymi w tabeli członkami zespołu wdrożeniowego, główny konstruktor powinien zdobyć potrzebne kwalifikacje drogą poznania: procesu technologicznego walcowni i zasad kierowania tym procesem, problematyki serwisu technicznego oraz szczegółów projektu walcowni.

Oprócz dobrego opanowania wiadomości dotyczących podstawowych elementów komputeryzowanego obiektu, główny konstruktor musi dysponować szerokim zakresem umiejętności warunkujących prawidłowe zaprojektowanie, uruchomienie, zbadanie i wdrożenie systemu w walcowni. Dotyczy to zwłaszcza umiejętności:

- projektowania i uruchamiania sprzętu komputerowego
- przygotowania systemów operacyjnych oraz testów kontrolnych (konieczna znajomość programowania w języku typu asembler)
- przygotowania oprogramowania użytkowego (konieczna znajomość programowania w językach wyższego poziomu)
- dobierania lub opracowywania metod matematycznych służących do rozwiązywania problemów sterowania procesami technologicznymi (konieczne pogłębienie wiadomości z zakresu matematyki, a zwłaszcza zdobycie umiejętności abstrakcyjnego myślenia)
- wdrażania techniki komputerowej w walcowni.

* * *

Mimo przedstawienia wielu negatywnych doświadczeń z zakresu wdrożeń informatyki — jestem optymistą. Uważam, że nowy program kształcenia na kierunku Informatyki i Zarządzania wychodzi częściowo naprzeciw potrzebom chwili. Wierzę, że jeśli ten program zostanie nieco zmodyfikowany, to przyszli inżynierowie informatycy będą mogli skutecznie wdrażać informatykę we wszystkich dziedzinach życia.

Na tle powyższych rozważań nasuwają się następujące wnioski:

1. Merytorycznymi kierownikami nowych wdrożeń informatyki powinni być inżynierowie dobrze rozumiejący zagadnienia sprzętowe i programowe.
2. Należy jak najszybciej zapewnić odpowiednie kształcenie inżynierów informatyków przez:

- rozszerzenie programu studiów o przedmioty metrologii i ochrony pracy oraz rozszerzenie zakresu podstawowych przedmiotów inżynierskich, np. podstaw elektrotechniki (nawet kosztem przedłużenia studiów o jeden semestr)
- tworzenie laboratoriów dydaktycznych systemów minikomputerowych i mikrokomputerowych
- przeniesienie studium specjalistycznego „Systemy sterowania" na Wydział Elektroniki i zwiększenia liczby studentów kształconych wg zmodyfikowanego programu „Inżynierii programowania"
- wprowadzenie kursów specjalizacyjnych dla merytorycznych kierowników wdrożeń informatyki.

LITERATURA

[1] Informator dydaktyczny dla studiów specjalizujących, kierunek elektrotechnika i elektronika. Politechnika Wrocławska, Wrocław, 1978
[2] Program kształcenia na subkierunku informatyki na Wydziale Informatyki i Zarządzania, Wrocław, grudzień 1979
[3] Projekt wstępny uniwersalnego systemu sterowania na mc ODRA 1204. WZE ELWRO, Wrocław, 1968
[4] Wojsznis W., Lewoc J. B.: Sistema uprawlenija prodwiżeniem metalla w prokatnom cehe. IV Międzynarodowa Konferencja RWPG na temat automatyzacji procesów produkcyjnych w czarnej metalurgii, Zaporoże, 1971
[5] Sawicki J., Kowalski A. J., Lewoc J. B.: Implementation of the Automatic Data processing for the Power System Control in Poland. Data Processing Conference, Madryt, 1974
[6] Kowalski A. i in.: System automatycznego przetwarzania informacji dla okręgowych dyspozycji mocy w energetyce, PN ICT Politechniki Wrocławskiej, nr 8, Wrocław, 1973
[7] Jakubek K.: System operacyjny dla zestawu ODRA 1325/CAMAC (informacja prywatna). Wrocław, 1977
[8] Lewoc J. B. i in.: Primary Data Processing Methods Applied in Power Generating Unit Monitoring System. Międzynarodowa konferencja na temat systemów elektrotechniki, Praga, 1978
[9] Założenie organizacji systemu operacyjnego dla symulatora bloku energetycznego 200 MW. IASE, oprac. nr 63340, Wrocław, 1975
[10] Rozent M.: Oprogramowanie symulatora systemu elektroenergetycznego (informacja prywatna). Wrocław, 1979
[11] Lewoc J. B., Łanowska B., Pogorzelski A.: Wykorzystanie minikomputera MOMIK do rozszerzenia możliwości zastosowania maszyn cyfrowych serii ODRA 1300 w systemach wielodostępnych. PN ICT Politechniki Wrocławskiej, nr 43, Wrocław, 1977

Projektowanie języka użytkownika

Programy i systemy programów oferujące użytkownikom wiele dowolnych funkcji (wariantów i kombinacji funkcji), wymagają podania dla każdego przebiegu programu warunków, które wywołują zarówno żądane funkcje jak i przeznaczone do przetwarzania dane. Tego rodzaju informacje sterujące zapewniają programom odpowiednie instrukcje sterujące. Programy parametryczne stwarzają ponadto możliwość opracowywania różnych struktur danych (pod względem budowy zbiorów, formatów danych itp.) albo ich generowania (np. zmiennych postaci list). Oprócz tych możliwości użytkownik dysponuje również instrukcjami do definiowania aktualnej struktury danych.

Aby zagwarantować prawidłowe wykonanie przez program określonych funkcji na opisanych strukturach danych, instrukcje te muszą zostać najpierw sprawdzone pod względem formalnym i merytorycznym.

Sprawdzenie instrukcji pod kątem prawidłowości formalnej oraz kompletności nazywane jest kontrolą syntaktyczną. W ramach kontroli syntaktycznej należy stwierdzić, czy elementy danej instrukcji są uporządkowane w przepisanej lub dopuszczalnej kolejności. Oprócz tego należy sprawdzić, czy elementy te składają się z odpowiednich znaków (liter, cyfr, znaków specjalnych). Merytoryczna prawidłowość instrukcji ustalana jest za pomocą procedur kontroli semantycznej.

Pracownik naukowy wydziału ekonomiki i organizacji firmy SIEMENS, p. Claus Weichselbaumer, podaje zalecenia i wskazówki dotyczące efektywnego projektowania wygodnych w użyciu rozwiązań w zakresie systemów informatycznych sterowanych przez użytkownika. Autor opiera się na przykładzie rozwiązań opracowanych na użytek firmy i niedostępnych dotąd jako gotowe produkty programowe.

KLASYFIKACJA JĘZYKÓW UŻYTKOWNIKA

Podstawowymi narzędziami informatyki w komunikacji człowiek—maszyna są tzw. języki sformalizowane (rys. 1). W zależności od obszaru ich zastosowań można je podzielić na następujące grupy:

- języki uniwersalne (ang. general purpose languages)
- języki specjalizowane (ang. special purpose languages).

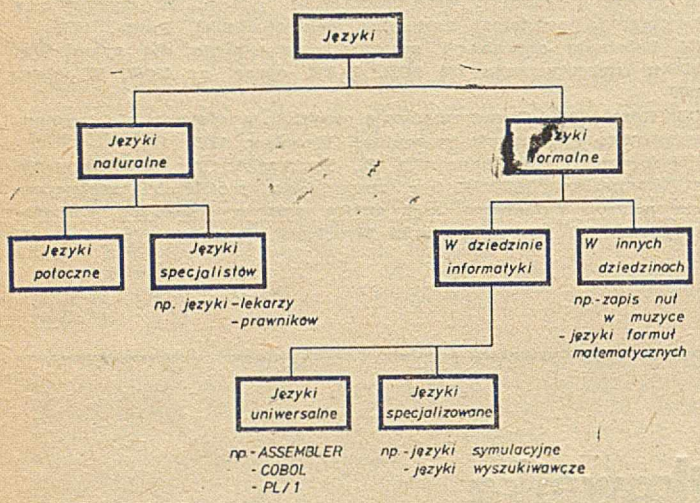
Do pierwszej grupy należą przykładowo języki COBOL, FORTRAN, PL/I oraz ASSEMBLER, natomiast do drugiej — języki symulacyjne, języki dostępu do banków danych oraz języki komend systemu operacyjnego. Języki specjalizowane, opracowane dla określonej dziedziny działalności, nazywane są językami użytkownika. Są to języki programowania ukierunkowane problemowo na potrzeby specyficznych zastosowań. Od ogólnych problemowo ukierunkowanych języków programowania (jak np. PL/I) odróżniają je następujące cechy:

- języki użytkownika są opracowane do rozwiązywania bardzo specyficznych problemów określonej dziedziny działalności
- liczba elementów języka jest stosunkowo niewielka (formułowanie problemów specyficznych wymaga najczęściej tylko kilku takich elementów)

- format instrukcji oraz elementy języka są ukierunkowane na język fachowy danej dziedziny działalności, dzięki czemu są łatwo dostępne również dla nie informatyków
- dzięki odwzorowaniu w instrukcjach lub elementach języka zadań cząstkowych można oferować gotowe rozwiązania ułatwiające nie tylko formułowanie, ale również znajdywanie rozwiązań problemów
- instrukcje języków użytkownika są wykonywane w sposób interpretacyjny i zwykle bezpośrednio po ich kontroli formalnej
- kontrolę formalną (syntaktyczną) można przeprowadzać za pomocą uniwersalnych programów (podprogramów) kontroli syntaktycznej.

Z punktu widzenia pracochłonności zastosowania języki użytkownika można podzielić na następujące trzy grupy (wg kryterium rosnącej łatwości korzystania):

- języki z instrukcjami o stałym formacie (format kart dziurkowanych)
- języki użytkownika z instrukcjami o formacie wywołań makrorozkazowych z argumentami typu pozycyjnego lub słów kluczowych (tzw. format komend)
- języki z instrukcjami o dowolnym formacie.



Rys. 1. Miejsce języków sformalizowanych jako podstawowego narzędzia w komunikacji człowiek — maszyna

Dążenie do poprawy komunikacji „człowiek—maszyna" zakłada, że przyszłe zastosowania informatyki, udostępniające użytkownikowi obszar działań bezpośrednio zależnych od problemu, zapewnią dysponowanie językami z instrukcjami o dowolnym formacie, ponieważ z jednej strony gwarantują one wystarczającą elastyczność wprowadzania danych, a z drugiej umożliwiają logiczne ukierunkowanie elementów takiego języka na język fachowy użytkownika. Ażeby maksymalnie zmniejszyć nakłady na opracowanie tego rodzaju języków można korzystać z metod i narzędzi, jakie zostały stworzone do opracowywania kompilatorów.

Opracowanie języka użytkownika rozpoczyna się od projektu języka. W fazie tej funkcjom przyjętego za podstawę systemu komputerowego zostają przyporządkowane językowe środki formułowania oraz ustalony alfabet, zasób słownictwa i lista instrukcji języka programowania. Wygodny język użytkownika powinien być ukierunkowany na zadania, potrzeby oraz mentalność przyszłego użytkownika. Wychodząc z takiego założenia w fazie projektowania powinien odbywać się ciągły dialog pomiędzy projektantami i późniejszymi użytkownikami języka. Projekt języka powinny sprawdzić obie zainteresowane strony wg określonych kryteriów. Aby zapewnić łatwą komunikację pomiędzy wszystkimi zainteresowanymi, każdy język użytkownika od samego początku powinien być opisywany w odpowiedniej metanotacji.

OPIS FORMALNY JĘZYKÓW UŻYTKOWNIKA W FAZIE PROJEKTOWANIA

Do formalnego opisu języków użytkownika istnieje wiele konwencji, za pomocą których w sposób przejrzysty można przedstawić strukturę instrukcji języka. Dla fazy projektowania tych języków zaleca się opisywanie poszczególnych instrukcji w tzw. „formacie ogólnym" (rys. 2). Jest to najpowszechniejsza postać opisu różnych formatów instrukcji systemów komputerowych (COBOL, język komend systemów operacyjnych). Opis instrukcji w „formacie ogólnym" ustala strukturę instrukcji złożoną z dozwolonych elementów języka. Reguły budowy różnych elementów języka z poszczególnych znaków (liter, cyfr, znaków specjalnych) zostają opisane również w sposób werbalny. Opis formalny projektu języka staje się podstawą jego oceny, ewentualnego korygowania i późniejszej implementacji.

Cecha	Znaczenie	Przykład
Słowa zarezerwowane	pisane są dużymi literami i mają stałe znaczenie	
• słowa kluczowe	muszą być bezwzględnie podane oraz podkreślone	DIVIDE
• słowa fakultatywne	mogą być zapisane lub opuszczone — ich zadaniem jest poprawa czytelności	ON
Słowa użytkownika	pisane są małymi literami (ogólne wyrażenia dla zadań formułowanych przez użytkownika)	literał numeryczny — 1
nawiasy prostokątne	wskazują, że wyrażenie zawarte w nawiasach jest fakultatywne	[ROUNDED]
nawiasy klamrowe	zamykają wyrażenia umieszczone piętrowo oraz informują, że jedno z tych wyrażeń jest obligatoryjne	{BY / INTO}
kropki kontynuacji	część wyrażenia bezpośrednio poprzedzająca kropki kontynuacji może być dowolnie powtarzana	...

Przykład:

DIVIDE {literał numeryczny — 1 / literał zmiennego przecinka — 1 / określnik — 1}
{BY / INTO} określnik — 2 [ROUNDED]
[ON SIZE ERROR instrukcja bezwarunkowa ...]

Rys. 2. Zapis instrukcji w „formacie ogólnym", zalecany w fazie projektowania języka

KRYTERIA OCENY PROJEKTU JĘZYKA

Atrakcyjność języka użytkownika wynika z gotowości dostosowania się do tego języka przez osoby z niego korzystające. Gotowość tę można wzmocnić przez lepsze dostosowanie języka do postawionych zadań oraz potrzeb i wymagań kręgu potencjalnych użytkowników.

Jako kryteria oceny projektu języka mogą być użyte następujące, istotne z punktu widzenia zastosowania oraz implementacji języka, własności i udogodnienia.

1. Kryteria oceny ukierunkowane na zastosowanie:
   - zasięg stosowania
   - komunikatywność
   - prostota
   - redundancja
   - łatwość nauczenia się
   - wygoda użytkowania.
2. Kryteria oceny ukierunkowane na implementację:
   - dogodność implementacji
   - łatwość konserwacji
   - kompatybilność „w górę".

Zasady stosowania języka określa szereg jego własności, które każdy użytkownik ocenia w sposób subiektywny. Dojrzały do implementacji projekt języka będzie więc zawsze pewnym kompromisem. Celem oceny projektu języka pod kątem jego użytkowania stosuje się m. in. kryteria oceny ukierunkowane na zastosowanie.

Zasięg zastosowania języka jest obszarem rozwiązań problemów, jaki w sensowny sposób można opisać środkami tego języka. Zasięg jest zależny od zakresu funkcji odpowiedniego systemu komputerowego, zakresu i efektywności instrukcji oraz przyjętych konwencji języka. Zwykle rozszerza się on wraz z zasobem słownictwa oraz liczbą instrukcji. Jednocześnie jednak rozszerzenie zasięgu powoduje zmniejszenie się prostoty języka.

Miarą komunikatywności języka użytkownika jest stopień trudności zastosowania takiego języka w kontaktach międzyludzkich. Osoba, której trzeba wyjaśniać rozwiązanie problemu w języku użytkownika (metody i technologię rozwiązań określonej dziedziny specjalizacji), lecz nie zna tego języka, oceni go jako bardziej komunikatywny wtedy, im bardziej upodabnia się on do języka fachowego danej dziedziny.

Prostota języka użytkownika wyraża się w:
- stosunkowo ograniczonym repertuarze znaków (eliminacja mało używanych znaków specjalnych)
- ograniczonym regertuarze słów kluczowych (słów zarezerwowanych)
- ograniczonej liczbie obowiązujących instrukcji i konwencji
- nieskomplikowanych formatach instrukcji
- eliminowaniu instrukcji lub elementów instrukcji, nie odpowiadających podstawowemu formatowi instrukcji (różnych wariantów jednej instrukcji).

Kryterium prostoty jest niestety sprzeczne z kilkoma innymi, również pożądanymi kryteriami, takimi jak np. szeroki zakres stosowania oraz elastyczność języka.
Podobnie jak w przypadku typowych języków programowania, również w językach użytkownika pożądany jest pewien stopień redundacji przy formułowaniu instrukcji. Stopień „rozwlekłości" języka użytkownika zależy od liczby elementów języka, jakie mogą ale nie muszą być użyte do jednoznacznego wyrażenia określonej treści zgodnie z regułami syntaktycznymi. Ograniczenie redundacji języka użytkownika powoduje na ogół zmniejszenie jego komunikatywności.

Łatwość nauczenia się języka jest w znacznym stopniu powiązana dotychczas wymienionymi kryteriami. Zakres stosowania języka, wielkość obszaru rozwiązywanych problemów oraz prostota reguł formułowania instrukcji warunkują stopień trudności jego opanowania przez użytkownika. Oprócz tego na łatwość opanowania takiego języka wpływają jasność zastosowanego słownictwa oraz instrukcji.

W czasie opracowania projektu języka należy dokonać ustaleń, określających rodzaj i zakres wykrywania oraz podawania informacji o błędach, co decyduje o komforcie zastosowania. W przypadku instrukcji błędnych pod względem formalnym procedura opracowania błędów powinna sygnalizować pozycję błędnego znaku lub elementu języka w meldunku o błędzie. Meldunki te powinny maksymalnie dokładnie precyzować informację o wykrytym błędzie. Wskazanie w meldunku dozwolonych znaków lub elementów języka pozwala wyeliminować konieczność sięgania do instrukcji operatorskiej. Oprócz tekstu, każdy meldunek o błędach powinien zawierać kod błędu, który w niezbędnym przypadku umożliwia sięgnięcie do bardziej szczegółowych wyjaśnień w instrukcji operatorskiej. Meldunki o błędach powinny być w maksymalnym stopniu ukierunkowane na problem. Należy unikać w meldunkach takich sformułowań, jak „błąd syntaktyczny" lub „błędna instrukcja", jeżeli możliwe jest bezpośrednie nawiązanie do znaczenia instrukcji lub jej argumentu. Należy preferować takie sformułowania jak „5000 — niedopuszczalny nr wyrobu" lub „8001 — błędny kod pocztowy".

W systemach konwersacyjnych zaleca się zaprojektowanie funkcji umożliwiającej zapytywanie o format lub znaczenie poszczególnych instrukcji lub ich argumentów. Oprócz tego uczestnik konwersacji powinien mieć możliwość decydowania o tym, aby meldunki o błędach były wyprowadzane w formie skróconej (tylko kod błędu), niezależnie od możliwości wywołania pełnego tekstu tego rodzaju informacji.

Już przy ustalaniu założeń języka pożądane jest uwzględnienie również kryteriów oceny ukierunkowanych na implementację. Projektanci języka muszą dbać o to, aby nakłady na implementację, a także tego, aby późniejsze nakłady na konserwację i dalszy rozwój języka były możliwie najmniejsze.

Skomplikowane konstrukcje języka (np. liczne alternatywne formaty instrukcji, nadmiar znaków specjalnych, nieujednolicone formaty instrukcji) świadczą o niedogodności implementacyjnej. Z punktu widzenia programu kontroli składni są to bardzo wrażliwe konstrukcje, które mogą łatwo prowadzić do błędnej interpretacji. Wynika z tego bezpośredni związek łatwości opracowania języka z kosztem jego implementacji. Koszt implementacji modułów kontroli syntaktycznej języka użytkownika można znacznie zmniejszyć przez zastosowanie programów standardowych, używanych do takiej kontroli podczas projektowania kompilatorów.

Oprócz wymienionych, istnieją narzędzia opracowane specjalnie na potrzeby implementacji języka użytkownika. Podczas opracowania języka należy stosunkowo wcześnie dokonać prób zbadania ich praktycznej przydatności, ponieważ w zależności od sposobu działania programy kontroli syntaktycznej wspomagają różne elementy konstrukcji języka (formaty instrukcji), a więc powodują konieczność uwzględnienia określonych ograniczeń. Zasadniczo należy unikać opracowania indywidualnych programów kontroli syntaktycznej dla każdego z języków ze względu na pracochłonność implementacji oraz związane z tym duże koszty.

Jeżeli istota lub repertuar funkcji systemu komputerowego ulegają zmianie, to w większości przypadków należy również zaktualizować związany z nim język użytkownika. Języki opracowane w oparciu o program kontroli syntaktycznej można stosunkowo łatwo zmienić, ponieważ ręcznie trzeba zmienić jedynie opis składni. Jeżeli język użytkownika został zrealizowany za pomocą indywidualnego programowania, to wprowadzenie zmian powoduje nieporównywalnie większe nakłady, ponieważ w większości przypadków trzeba wtedy opracować nowe lub znacznie zmienić istniejące podprogramy kontroli syntaktycznej, co oznacza istotne pogorszenie możliwości konserwacji.

W przypadku rozwoju i rozszerzania języka użytkownika należy uwzględnić kompatybilność z jego wersją początkową. Powinna istnieć możliwość włączenia do języka nowych instrukcji bez konieczności zmian jego koncepcji. Również ustalone na początku reguły języka w zakresie formatu instrukcji nie mogą stać się niekompatybilnymi w wyniku rozwoju lub zmian tego języka. Dla zagwarantowania możliwości rozszerzania języka należy więc zagwarantować pełną jego kompatybilność „w górę".

FORMALNY OPIS JĘZYKA Z UWZGLĘDNIENIEM WYMAGAŃ IMPLEMENTACJI

Instrukcje języka użytkownika składają się z argumentów (słów, liczb dziesiętnych, symboli), złożonych z kolei ze znaków (liter, cyfr, znaków specjalnych). Ustalone z góry reguły pozwalają tworzyć z argumentów oraz znaków prawidłowe instrukcje i elementy języka. Reguły te to gramatyka lub składnia języka. Składnię języka dzieli się na dwie części: składnię wysokiego poziomu (ang. high-level-syntax) oraz składnię niskiego poziomu (ang. low-level-syntax).

Składnia wysokiego poziomu reguluje wiązanie elementów języka z instrukcjami, natomiast składnia niskiego poziomu obejmuje zasady tworzenia różnych elementów języka ze znaków (alfabet języka). Kontrola syntaktyczna instrukcji dzieli się — odpowiednio do obu wymienionych części składni — na fazy analizy leksykalnej oraz analizy syntaktycznej.

Zależnie od sposobu działania programów kontroli syntaktycznej obie składnie opisywane są za pomocą różnych lub identycznych konwencji.

Jeżeli program kontroli syntaktycznej działa wg zasady „od dołu do góry" (ang. bottom-up), to składa on pojedyncze znaki w elementy języka, a te w instrukcje. Gdy w ten sposób zostanie stwierdzona prawidłowa dla instrukcji kombinacja elementów języka, następuje jej uznanie za dopuszczalną. Tego rodzaju reguły składni zgodnie z istotą swego działania nazywane są systemem generującym.

Do przedstawienia reguł składni niskiego poziomu — w przypadku programów kontroli syntaktycznej realizowanych metodą „od dołu" — stosuje się w zasadzie tabele stanu. Składnia wysokiego poziomu zapisywana jest w metajęzyku, np. w notacji standardowej Backusa lub notacji Backusa—Naura (BNF).

W przypadku programów kontroli syntaktycznej działających wg zasady „od góry do dołu" (ang. top-down), a więc rozkładających instrukcje na elementy języka, a te z kolei na tworzące je elementarne znaki, składnie zarówno niskiego, jak i wysokiego poziomu opisywane są w notacji standardowej Backusa lub w podobnych notacjach. Reguły składni dla programu kontroli syntaktycznej typu „odgórnego" nazywane są — zgodnie z istotą przetwarzania — systemem redukującym. W przypadku tej zasady instrukcje są uznawane za prawidłowe wtedy, gdy ich rozłożenie na elementy języka oraz znaki elementarne przebiega bez zakłóceń.

Systemy generujące i redukujące określają nie tylko budowę instrukcji poprawnych formalnie, ale również podstawowy algorytm programu kontroli syntaktycznej.

Ponieważ notacje składni różnych narzędzi implementacji wykazują istotne różnice pod względem formatu oraz wygody stosowania, poniżej zostanie omówiony dokładniej tylko najbardziej znany metajęzyk, jakim jest niewątpliwie notacja standardowa Backusa.

Aby język użytkownika opisać za pomocą metajęzyka, niezbędne są z jednej strony alfabet i słownictwo metajęzyka, a z drugiej strony — identyczne elementy opisywanego języka. Oprócz tego potrzebny jest zestaw zapisanych w metajęzyku reguł systemu (generującego lub redukującego).

Symbole syntaktyczne, stanowiące słownictwo metajęzyka, nazywane są w literaturze metazmiennymi, zmiennymi notacji, symbolami pośrednimi albo nieterminalami (ang. non-terminals). Metazmienne reprezentują tzw. klasy gramatyczne (słowa kluczowe, słowa wyszukiwawcze, stałe znakowe) instrukcji albo ciągu klas gramatycznych. Oznacza to, że elementy języka o identycznej strukturze formalnej należą do jednej klasy gramatycznej.

Od chwili swego zaprojektowania pierwotna notacja metajęzyka jest w ograniczonym stopniu uzupełniana w kierunku „rozszerzonej" notacji BNF (rys. 3).

<x> zapis metazmiennej x metajęzyka

: : = rozdziela prawą i lewą stronę wzoru generowania i oznacza „składa się z" lub „zostało wytworzone z". Metazmienne są więc tworzone stopniowo — aż do symboli ostatecznych (słów lub znaków opisywanego języka), które nie są już zawarte w ostrych nawiasach

<x> / <y> } są notacjami ekwiwalentnymi dla alternatyw
{<x>, <y>} } o znaczeniu „lub"

[x] notacja nie obligatoryjnych części instrukcji

. . . podane przed kropkami metazmienne lub klasa gramatyczna mogą występować dowolnie często

[x] max, min ustala liczbę dopuszczalnych powtórzeń metazmiennej lub klasy gramatycznej. Jeżeli podane jest min = 0, to zdefiniowana część instrukcji nie jest obligatoryjna

przykład: < liczba > : : = < cyfra > / < cyfra > < liczba >
< cyfra > : : = 0/1/2/3/4/5/6/7/8/9

Rys. 3. Formalizacja gramatyki języka użytkownika za pomocą rozszerzonej notacji Backusa

Formalne ustalenie gramatyki języka użytkownika daje następujące korzyści:

- podstawowa struktura języka (instrukcje i słownictwo) jest przedstawiona w przejrzystej, a zarazem oszczędnej postaci
- zapewnia dysponowanie opisem pełnego zestawu reguł tworzenia prawidłowych instrukcji
- pozwala rozróżniać niejednoznaczne zasady generowania lub redukowania
- w przypadku wprowadzenia nowych elementów języka, projektanci oraz realizatorzy języka mogą szybko i bezbłędnie stwierdzić, czy nowe elementy dadzą się włączyć do istniejącej struktury języka albo czy będzie można rozszerzyć przyjęte reguły tego języka
- skoncentrowanie w systemie generującym lub redukującym reguł językowych pozwala ocenić stopień trudności implementacji języka
- zapewnienie przez system generujący lub redukujący możliwości konkretyzacji zadań implementacji.

Algorytm kontroli syntaktycznej określa w znacznym stopniu ścisły i jasny podział systemu reguł oraz dokładne przedstawienie procesu rozkładania języka aż do poziomu elementarnych symboli.

Prezentacja systemu reguł języka stanowi ponadto istotny punkt wyjścia dla podprogramów kontroli semantycznej. Z systemu generującego lub redukującego można wyprowadzać dowolne układy danych, mające duże znaczenie dla semantycznej kontroli instrukcji.

NARZĘDZIA IMPLEMENTACJI JĘZYKÓW UŻYTKOWNIKA

Zadania translatorów języków programowania, takich jak COBOL lub FORTRAN, polegają między innymi na analizie leksykalnej i syntaktycznej. Właśnie dla kompleksu tych zadań, już na początku lat sześćdziesiątych projektanci translatorów rozpoczęli opracowanie metod i narzędzi automatycznego wytwarzania modułów kontroli syntaktycznej. Podstawą tego rozwoju było poznanie możliwości formalnego opisu składni języków programowania. Dzisiaj istnieje już wiele sterowanych tabelami modułów kontroli syntaktycznej i generatorów programów, pozwalających tworzyć dla języków formalnych indywidualne moduły takiej kontroli. Kilka spośród nich opracowano specjalnie na potrzeby implementacji języków użytkownika.

Dzięki zastosowaniu tych narzędzi udało się zmniejszyć koszty implementacji języków użytkownika o 80% w porównaniu do rozwiązań kodowanych metodą tradycyjną. Do chwili obecnej istnieje jednak niewiele przypadków użycia tych narzędzi w dziedzinie zastosowań gospodarczych.

Dokładny opis wszystkich istniejących programów kontroli syntaktycznej znacznie przekroczyłyby ramy niniejszego artykułu. W trzech przykładach, które z jednej strony odzwierciedlają zastosowanie w różnych dziedzinach, z drugiej zaś zróżnicowanie form implementacji, zostaną dlatego wskazane jedynie najbardziej znaczące własności tego rodzaju narzędzi.

CDL 2 (Compiler Description Language)

CDL 2 jest językiem programowania przeznaczonym do implementacji kompilatorów i kompilatoropodobnych programów (interpreterów, edytorów itp.). Oprócz składni języka, za pomocą CDL 2 można również opisywać lub zestawiać parametryczne odwoływanie się do podprogramów semantycznych lub wywoływanie samych podprogramów tego typu. Wynikiem translacji CDL 2 jest asemblerowy program źródłowy w systemie operacyjnym BS 2000. Po translacji programu, połączonej z przebiegiem łączenia, powodującym m.in. dowiązanie systemu czasu rzeczywistego CDL 2, uzyskuje się gotowy do wykonania program opracowania instrukcji pierwotnie użytego języka. Język programowania CDL 2, który opracowano w ścisłej współpracy z Politechniką Berlińską, jest dostosowany do wymagań systemu operacyjnego BS 2000. Jest użytkowany głównie do rozwiązywania problemów badawczo-rozwojowych.

ISP (Interpretive Scanner and Processor)

System kontroli składni ISP składa się z zależnego od systemu operacyjnego modułu oraz zestawu sześciu makrorozkazów, pozwalających przeprowadzić analizę leksykalną (za pomocą tabel stanu) oraz analizę syntaktyczną instrukcji języka użytkownika. ISP stanowi część składową systemu operacyjnego BS 2000 i jest stosowany do kontroli języka komend tego systemu, wzorów matematycznych ujętych w funkcjach kalkulatorów stołowych, a także kontroli instrukcji większości programów usługowych.

Użytkownik ISP definiuje repertuar znaków, słownictwo i składnię implementowanego języka użytkownika posługując się parametryzacją makrorozkazów ISP. Korzysta-

jąc z nich może on ponadto żądać wyprowadzenia określonych elementów języka do sformatyzowanego obszaru wyjścia. ISP umożliwia dodatkowo definiowanie przekształceń formatu (np. z dziesiętnego na upakowany). ISP stosować można zarówno w systemie operacyjnym BS 1000 jak i BS 2000.

LINGUS (procesor języków użytkownika)

LINGUS jest wygodnym w użyciu procesorem, tworzącym z opisu języka w notacji typu BNF indywidualne programy kontroli syntaktycznej. Włączone w system użytkowy, moduły LINGUSA realizują następujące zadania:

- kontrolę składni (metodą „od góry do dołu") założonych instrukcji języka
- przenoszenie rozpoznanych elementów języka do obszaru wyjściowego
- przekształcanie postaci rozpoznanych elementów języka (np. dziesiętnej na binarną)
- przekodowywanie rozpoznanych elementów języka (np. nazwy Monachium na liczbę 8000, będącej kodem pocztowym tego miasta)
- wywoływanie specjalnych podprogramów użytkownika, służących do dodatkowego opracowania elementów języka, części instrukcji lub kompletnych instrukcji (np. sprawdzanie prawdopodobieństwa)
- udostępnianie w obszarze komunikowania zdefiniowanych tekstów wspomagania dialogu (dynamiczna funkcja HELP)
- udostępnianie w obszarze komunikowania zdefiniowanych opisów instrukcji za pomocą standardowej instrukcji HELP (statyczna funkcja HELP)
- przenoszenie do obszaru błędów zdefiniowanych, ukierunkowanych na metody komunikatów o błędach syntaktycznych.

Moduły LINGUSA mogą być wywoływane z programów napisanych w językach COBOL, FORTRAN i ASSEMBLER. Procesor LINGUS działa w systemie operacyjnym BS 2000, natomiast jego moduły są niezależne od systemu operacyjnego, ponieważ przekazują one wejście i wyjście nadrzędnemu systemowi informatycznemu. Oprócz zastosowania w problemach użytkowych — LINGUS jest wykorzystywany do opracowania kompilatorów oraz jako część składowa systemu operacyjnego BS 2000.

OPRACOWANIE INSTRUKCJI JĘZYKA UŻYTKOWNIKA

Opinia, czy sprawdzona instrukcja jest prawidłowa czy też formalnie błędna, nie jest jedynym wynikiem kontroli syntaktycznej. Dalsze zadanie programu tego rodzaju kontroli polega na takim przygotowaniu sprawdzonej instrukcją, aby podczas jej opracowania można było bezpośrednio dotrzeć do elementów języka o istotnym znaczeniu.

Po zakończeniu kontroli syntaktycznej elementy te muszą zostać zbadane (indywidualnie oraz łącznie) pod względem merytorycznym celem ustalenia znaczeń poszczególnych instrukcji. Operacje te realizowane są przez podprogramy kontroli semantycznej.

Wynikiem kontroli semantycznej jest merytoryczna ocena prawidłowości instrukcji, przy czym wyprowadzane są wyłącznie instrukcje prawidłowe.

Podprogramy kontroli semantycznej realizują specyficzne dla problemu badania prawdopodobieństwa oraz kompletności.

Podprogramy kontroli semantycznej — podobnie jak pozostałe podprogramy systemu informatycznego — opracowuje programista oprogramowania użytkowego. Pracochłonność opracowania tego rodzaju podprogramów jest w większości przypadków stosunkowo niewielka.

* * *

Osoby zainteresowane zastosowaniami informatyki życzą sobie na przyszłość możliwości dysponowania znaczną liczbą specjalizowanych języków użytkownika, dobrze przystosowanych do zakresu ich działalności. Języki te powinny charakteryzować się prostotą budowy oraz łatwością nauczenia się. W ten sposób każdy użytkownik będzie mógł w bardzo krótkim czasie opanować i zastosować taki język bez uciążliwego szkolenia, nawet gdy jest on całkowitym laikiem w zakresie informatyki. W praktyce oznacza to, że np. księgowy będzie się mógł posługiwać językiem finansowym, a nawet tak bliskim mu językiem rachunkowości. Będzie on więc zdolny samodzielnie rozwiązywać problemy programowania sprzętu komputerowego bez konieczności korzystania z pomocy programistów.

Opracował Władysław KLEPACZ na podstawie artykułu Clausa Weichselbaumera „Methoden und Hilfsmittel zur Entwicklung von Benutzersprachen" (DATA REPORT nr 3/79)

Wielkie bazy danych

Francuski Państwowy Instytut Badawczy Informatyki i Automatyzacji INRIA organizuje w dniach 9—11 września 1981 r. w Cannes (Francja) międzynarodową konferencję na temat bardzo wielkich baz danych (7th International Conference on Very Large Data Bases).

Konferencje tego cyklu poświęcone są prezentacji prac badawczo-rozwojowych oraz praktycznych zastosowań baz danych. Głównym celem tegorocznej konferencji jest przedstawienie stanu aktualnych badań oraz rozszerzenie wymiany informacji i doświadczeń z projektowania, budowy i eksploatacji baz danych, a także stworzenie forum do dyskusji na temat przyszłych badań i rozwoju. Główne akcenty zostaną położone na projektowanie, wdrażanie i eksploatację rozproszonych baz danych, a także maszyn baz danych. Organizatorzy proponują następującą szczegółową tematykę konferencji:

- rozproszone bazy danych (ocena i optymalizacja zapytań, równoczesność oraz niezawodność działania, zabezpieczenia, lokalne systemy rozproszone, schematy rozproszenia i lokalizowanie baz),
- teoria baz danych (logika bazy danych, teoretyczne koncepcje systemów, modelowanie niekompletnej struktury oraz problem niedostatecznej precyzji danych)
- maszyny bazy danych (pamięć asocjacyjna, architektura wieloprocesorowa, interfejsy sieci i komputera głównego, ocena wydajności)
- projektowanie bazy danych (modelowanie przedsięwzięcia, schemat koncepcyjny i szczegółowy projektu, narzędzia i pomocnicze środki projektowania, szacunek wydajności)
- automatyzacja biur (widograficzne przetwarzanie tekstów, systemy elektronicznej poczty, modele przepływu informacji w biurze, bazy danych kadrowych)
- wdrażanie systemu bazy danych (architektura systemu, słowniki danych, systemy eksploatacji bazy danych, struktury pamięci)
- interfejsy użytkownika (przetwarzanie formularzy, języki zapytań, interfejsy foniczne i wideograficzne, języki naturalne)
- inne tematy (języki programowania a bazy danych, konwersja baz danych i programów, analiza przykładów oraz zastosowania specjalne, sztuczna inteligencja).

Dalsze informacje można uzyskać pod adresem: **INRIA — Public Relations Department, B.P. 105-78150 LE CHESNAY, France, tel. 954 20 20 ext. 600, teleks: IRIA 697 033 F.**

SOFSEM'80

Dzięki uprzejmości organizatorów mgr Łukasz Kawecki i ja mieliśmy okazję uczestniczyć w czechosłowackim seminarium SOFSEM'80, odbywającym się w dniach 23.11 — 5.12. ub.r. w Białym Krzyżu. Malownicza lokalizacja (szczyt Biały Krzyż — 1200 m npm, w sercu Beskidów), dogodny termin imprezy (gwarantowany śnieg i dobre warunki narciarskie) oraz rozkład dnia (codziennie kilka wolnych godzin przed zmrokiem) znacznie uatrakcyjniły spotkanie. Nie należy przy tym przypuszczać, że dbałość o aspekty socjalno-bytowe obniżyła poziom naukowy imprezy.

SOFSEM'80 zaoferował uczestnikom wiele form zdobywania informacji. Systematyzacji wiadomości oraz metod i technik informatycznych służyły cykle sześciogodzinnych wykładów, których tematyka objęła nie tylko teoretyczne podstawy informatyki, ale i nowinki, takie jak: język ADA, minikomputer VAX-11 czy ostatnie ciekawostki ze świata mikroprocesorów. Krótsze referaty (30 min) prezentowały przegląd najnowszych osiągnięć i prowadzonych prac — zarówno teoretycznych, jak i praktycznych (implementacja oprogramowania, automatyzacja prac programistycznych, sieci komputerowe). O ładunku rzetelnej informacji niech zaświadczy sama objętość materiałów konferencyjnych — 425 stron formatu A4! W trakcie seminarium znalazło się także miejsce na prezentację błyskawicznych (3—10 min) doniesień, zgłaszanych w ramach tzw. wolnej trybuny.

Dyskusje panelowe dotyczyły zastosowań minikomputerów oraz oblicza przyszłych SOFSEMów (ostatni był siódmy z kolei). Wystąpienia ministra informatyki i szefa organizacji odpowiadającej naszemu Zjednoczeniu Informatyki przyniosły aktualną informację o warunkach i potrzebach rozwoju informatyki w CSRS. Podczas konferencji czynna była wystawa najnowszych publikacji informatycznych, do której uczestnicy dokładali przywiezione przez siebie ciekawostki; można je było przejrzeć lub wypożyczyć do przeczytania. Pojawiło się także mnóstwo ogłoszeń o rozpoczynających się niebawem innych zjazdach, konferencjach i sympozjach, także międzynarodowych. Uderzające w tym wszystkim jest realne spojrzenie na dystans dzielący nas od krajów zachodnich oraz brak zakłamania w ocenie zaopatrzenia w sprzęt i stanu informatyki.

Mimo długiego relatywnie czasu trwania imprezy (dwa tygodnie) brak było pustych miejsc i „dziur" w programie seminarium. Wszystko zaprogramowane było dokładnie i zapięte na ostatni guzik, a szybka obsługa techniczna gwarantowała sprawny przebieg SOFSEMu. Wymianie informacji między uczestnikami sprzyjały imprezy integrujące, choćby tradycyjny już bal maskowy. Oprócz zwyczajnych na tego typu zjazdach dwu- i wielostronnych dyskusji kuluarowych regularnie odbywały się zebrania kółka samokształceniowego. Zadbano również o rozrywkę, np. tradycyjny ANTISOFSEM, w którym zobaczyliśmy, jak u naszych sąsiadów jedni uznani informatycy wyśmiewają się z innych uznanych informatyków (tak w sprawach naukowych, jak i z osobistych wad i słabostek) i nikogo to nie boli, nikt się nie obraża... Poczuciu wspólnoty sprzyjało całkowite zrezygnowanie z tytułów naukowych i zawodowych, których mnogość zastąpiły uniwersalne „sofsemistka" i „sofsemista". Przyklasnąć też należy dobrej selekcji uczestników. Od „tubylców" dowiedzieliśmy się, że nie tak łatwo dostać się na SOFSEM i każdorazowa obecność na nim jest rodzajem wyróżnienia.

Mgr Jerzy J. KLACZAK
Instytut Informatyki
Uniwersytet Wrocławski

Polskie Towarzystwo Informatyczne

Przygotowania do Zjazdu

Trwają prace organizacyjne nad utworzeniem Polskiego Towarzystwa Informatycznego, o których zainicjowaniu donosiliśmy w dwóch poprzednich numerach. Komitet Założycielski ukończył prace nad statutem Towarzystwa i statut ten został przesłany na początku lutego do Urzędu Miasta St. Warszawy w celu rejestracji. Obecnie dyskusje dotyczą problemów struktury Towarzystwa i jego przyszłego programu. Przewidywane są między innymi następujące struktury organizacyjne:

- **komitety naukowe** — np. do spraw metod translacji
- **sekcje tematyczne i techniczne** — np. klub użytkowników maszyn ODRA
- **komisje do spraw specjalnych** — np. opracowania kodeksu etycznego.

Uwagi i propozycje dotyczące programu struktury Towarzystwa mogą być składane na ręce **doc. Stanisława Waligórskiego, Instytut Informatyki UW, PKiN, p. 850, tel. 26 82 50 lub 20 17 20.**

Systemy operacyjne komputerów Jednolitego Systemu

Efektywność pracy komputera w znacznym stopniu uzależniona jest od jakości stosowanego systemu operacyjnego. Z chwilą wprowadzenia na rynek krajowy komputerów serii RIAD powstało szereg problemów z doborem systemu operacyjnego. Niepełna kompatybilność systemów DOS i OS zwiększa trudności związane z podejmowaniem właściwych decyzji w przypadku inicjowania nowych zastosowań, a w szczególności komplikuje proces eksploatacji zastosowań pod kontrolą różnych systemów operacyjnych. Dodatkowe utrudnienie w doborze odpowiedniego systemu powoduje rozwój konfiguracji sprzętu, a także wprowadzenie teletransmisji.

W krajowych ośrodkach informatycznych znajdują zastosowanie różne (najczęściej nie kompatybilne) systemy operacyjne: systemy dostarczane przez producentów maszyn JS, systemy IBM, technologiczne wersje systemów opracowane i upowszechniane w sieci ZETO, system DOS3 autorstwa BOOiI POLMO. Ponadto szereg systemów operacyjnych uzupełnianych jest o takie dodatkowe pakiety, jak POWER, HASP, pakiety rozliczania zasobów itp.

Wyżej opisana sytuacja zainspirowała do zorganizowania konferencji, której postawiono następujące cele:

- **zinwentaryzowanie opracowań systemów operacyjnych maszyn JS**
- **wymiana doświadczeń, zdobytych przy eksploatacji maszyn JS pod różnymi systemami operacyjnymi**
- **dokonanie próby określenia optymalnego doboru systemu operacyjnego, w zależności od dostępnego sprzętu oraz konkretnych zadań**
- **zdefiniowanie zaleceń dla krajowego programu prac nad rozwojem systemów operacyjnych.**

Miejsce i termin konferencji: **Augustów, 25—27 maja 1981 r. Zgłoszenia uczestnictwa przyjmuje Towarzystwo Naukowe Organizacji i Kierownictwa, Oddział w Białymstoku, ul. Dąbrowskiego 1.** Zgłoszenia można nadsyłać na podany adres do 20 kwietnia br.

Mgr inż. Aleksy BRECZKO
Sekretarz Organizacyjny
Konferencji

zjednoczenie informatyki ZETO

System rezerwacji miejsc sypialnych i kuszetek

Informatyczny system rezerwacji miejsc sypialnych i kuszetek opracowany został przez ZETO Wrocław na potrzeby i zlecenie Polskiego Biura Podróży ORBIS. Prace, w ramach których zrealizowany został pełny cykl budowy tego systemu, trwały od grudnia 1977 r. do czerwca 1979 r.

Z metodycznego punktu widzenia założono, że w odniesieniu do rozwiązań projektowo-programowych oraz zakresu realizowanych funkcji, opracowany zostanie najpierw system pilotowy, a następnie — na bazie doświadczeń i analiz zebranych w trakcie jego eksploatacji — przystąpi się do budowy jego wersji użytkowej. Zgodnie z tym założeniem prace nad systemem przebiegały według następujących etapów:

- wstępne rozpoznanie problemu oraz opracowanie założeń systemu
- opracowanie pilotowej wersji projektu technicznego i oprogramowania systemu
- wdrożenie wersji pilotowej systemu w obiekcie zarządzania (PBP ORBIS w Warszawie) oraz jego próbna eksploatacja w warunkach rzeczywistych
- opracowanie użytkowej wersji projektu technicznego oraz oprogramowania systemu
- wdrożenie wersji użytkowej oraz eksploatacja próbna systemu (na danych rzeczywistych) w PBP ORBIS w Warszawie
- eksploatacja systemu
- modernizacja i optymalizacja rozwiązań systemowych.

Wersja użytkowa systemu została rozszerzona — w porównaniu do wersji pilotowej — o zapewniające ciągłą pracę systemu procedury zabezpieczeń awaryjnych.

Okres eksploatacji systemu ustalono na 7 lat, tj. do 1985 roku. Przewiduje się, że w tym terminie zostanie opracowany i wdrożony przez przewoźnika, tj. PKP, nowy system rezerwacji miejsc, a jednostki organizacyjne PBP ORBIS — realizujące przedsprzedaż miejsc w pociągach — stanowić będą jedno z ogniw jego eksploatacji.

ZAKRES I FUNKCJE SYSTEMU

System obejmuje prowadzenie rezerwacji miejsc sypialnych i kuszetek w pociągach relacji krajowych i zagranicznych, wychodzących i przychodzących z (do) danego węzła komunikacyjnego (system zaprojektowany został dla węzła warszawskiego). Przyjęto, podobnie jak w tradycyjnym systemie rezerwacji miejsc, że okres wyprzedzenia, w którym można rezerwować miejsca, wynosi 90 dni dla relacji krajowych oraz 60 dni dla relacji zagranicznych.

W zdefiniowanym obszarze zastosowań (obsługa jednego węzła komunikacyjnego) realizowane są podstawowe funkcje rezerwacji miejsc oraz emisji dokumentacji użytkowej, a także funkcje zabezpieczające utrzymywanie zbiorów i systemu, zapewniające ciągłą, prawidłową i sprawną jego eksploatację, a w razie awarii systemu — możliwość odtworzenia jego poprzedniego stanu pracy.

STRUKTURA I ORGANIZACJA PROCESU PRZETWARZANIA

Z punktu widzenia organizacji i technologii procesu przetwarzania oraz realizowanych funkcji, w systemie wyodrębniono następujące główne części:

- zakładanie i utrzymywanie zbiorów danych
- aktualizacja zbiorów danych
- rezerwacja i bieżące bilansowanie miejsc
- zbiorcze bilansowanie miejsc
- emisja dokumentacji użytkowej
- nadzorowanie i sterowanie pracą systemu
- odzyskiwanie systemu.

System rezerwacji jest systemem o działaniu bezpośrednim. Dzięki temu użytkownik ma możliwość prowadzenia konwersacji z systemem przy obsłudze podstawowych procedur użytkowych, jak również bieżącego aktualizowania zbiorów danych. Przetwarzanie jest w całości realizowane w oparciu o system minikomputerowy DATAPOINT [1], przy wykorzystaniu dyskowego systemu operacyjnego oraz pakietu podziału czasu DATASHARE.

Komunikacja pomiędzy użytkownikiem (peronel PBP ORBIS) a systemem odbywa się w trybie konwersacyjnym, poprzez monitory ekranowe z klawiaturą, zainstalowane w siedzibie użytkownika.

Podział monitorów z punktu widzenia organizacji pracy systemu i realizowanych funkcji jest następujący:

- **terminale kasowe, tzw. operacyjne,** zainstalowane w kasach PBP ORBIS prowadzących przedsprzedaż miejsc w pociągach, wykorzystywane do realizacji procedur: rezerwacji miejsc indywidualnych, odwołania (kasowania) rezerwacji oraz komunikacji z operatorem głównym systemu
- **terminal (terminale) kasowy specjalny** [2], wykorzystywany do realizacji procedur: rezerwacji miejsc grupowych i specjalnych (np. poselskich), a także miejsc indywidualnych, odwołania (kasowania) rezerwacji ww. miejsc, komunikacji z operatorem głównym systemu
- **terminal główny,** zainstalowany przy jednostce centralnej, wykorzystywany do realizacji wszystkich przewidzianych w systemie procedur, a zwłaszcza procedur związanych z zakładaniem i umiejscowieniem zbiorów danych, nadzorowaniem, sterowaniem i zabezpieczeniem systemu przed dostępem osób niepowołanych do jego obsługi oraz procedur związanych z utrzymywaniem oprogramowania użytkowego rezerwacji miejsc w stanie aktualnym.

Sterowanie pracą terminali przeznaczonych do obsługi systemu odbywa się poprzez przydzielone do każdego z nich programy ANSWER i MASTER, które są każdorazowo uaktywniane po uruchomieniu dyskowego systemu operacyjnego oraz pakietu DATASHARE. Program ANSWER jest uruchamiany automatycznie po włączeniu terminala.

Program MASTER jest również uruchamiany automatycznie — po wykonaniu programu ANSWER.

W ramach programu ANSWER następuje zainicjowanie pracy terminala, w wyniku czego należy wprowadzić do systemu hasło identyfikujące dane urządzenia końcowego. Program MASTER natomiast jest to podstawowy program sterujący pracą terminala. W ramach tego programu realizo-

[1] O wyborze systemu minikomputerowego DATAPOINT zadecydował wynik analizy porównawczej systemów: NCR 8200, MERA 400 i DP 6600 (lub 5500), przeprowadzonej przez autorów systemu w trakcie opracowywania założeń projektowych.

[2] Terminale specjalne przewidziane są głównie do obsługi procedur rezerwacji miejsc grupowych i specjalnych, co z organizacyjnego punktu widzenia znacznie usprawnia działalność biura przedsprzedaży.

wane są procedury pozwalające na uruchomienie dowolnego programu użytkowego systemu rezerwacji, uruchomienie systemu komunikacji między terminalami oraz zakończenie pracy terminala. Po zrealizowaniu programu użytkowego następuje każdorazowo powrót do programu sterującego MASTER.

DZIAŁANIE SYSTEMU

Inicjując pracę systemu rezerwacji należy wprowadzić aktualną datę, w oparciu o którą wyliczona zostaje ostatnia data przyjętego okresu rezerwacji oraz dane o typie i liczbie dostępnych w konfiguracji dysków. System przechodzi następnie do modułu „Zakładanie i utrzymywanie zbiorów danych", w ramach którego wykonywane są funkcje związane z tworzeniem:

- zbiorów systemowych: „katalogu relacji", zawierającego wykaz wszystkich relacji objętych aktualnie przetwarzaniem oraz „zbioru specjalnego", służącego m.in. do realizacji procedur restartów awaryjnych
- zbiorów użytkowych, zakładanych dla każdej relacji objętej przetwarzaniem, zawierających zapisy o aktualnym stanie rezerwacji i dostępności miejsc w składzie pociągu.

Przy inicjowaniu codziennej pracy należy wprowadzić hasło, a następnie aktualną datę. Data ta może być równa dacie uprzednio wprowadzonej i wtedy oznacza wznowienie pracy systemu w tym samym dniu (po przerwie). Jeśli natomiast wprowadzona do systemu data jest następną po ostatnio wprowadzonej i przechowywanej w systemie dacie aktualnej — oznacza to pierwsze inicjowanie pracy systemu w danym dniu. W wyniku tego następuje automatyczne uruchomienie procedury codziennego dopisywania i kasowania zapisów w zbiorach, co gwarantuje utrzymanie aktualnego ich stanu z punktu widzenia przyjętego okresu rezerwacji.

Po zainicjowaniu lub wznowieniu codziennej pracy na ekranach terminali (głównym i kasowych), zostaje wyświetlony komunikat, podający wykaz głównych funkcji dostępnych do realizacji na danym terminalu. Wybór określonej funkcji przez użytkownika (operatora terminala) możliwy jest po wprowadzeniu jej identyfikatora, w wyniku czego wywołany zostaje odpowiedni program (podprogram) realizujący daną procedurę.

W przypadku, jeśli w danym dniu mają być włączone do przetwarzania nowe relacje połączeń kolejowych to przed rozpoczęciem realizacji procedur rezerwacji miejsc należy uruchomić moduł „Zakładanie i utrzymywanie zbiorów danych", w ramach którego założone zostaną podstawowe zbiory użytkowe dla tych relacji. W przypadku kontynuacji pracy systemu w oparciu o przyjęte poprzednio relacje, przetwarzanie w danym dniu opiera się na realizacji pozostałych modułów systemu.

Utrzymywanie zbiorów informacji w aktualnym stanie, zapewniające prawidłową eksploatację systemu, realizuje również moduł „Aktualizacja zbiorów danych". W ramach tego modułu przeprowadzana jest bieżąca aktualizacja zbiorów, polegająca na zmianie stanu zapisów (zawartości pól zapisów) w zbiorach, kasowaniu zbiorów (relacji) użytkowych itp. Prawidłowość i terminowość wykonania aktualizacji rzutuje w zasadniczy sposób na poprawność funkcjonowania pozostałych procedur systemu, a w szczególności rezerwacji, dokonywanej w ramach modułu „Rezerwacja i bieżące bilansowanie miejsc" — codziennie w trybie konwersacyjnym w oparciu o transakcje wprowadzane z klawiatur terminali zainstalowanych w kasach przedsprzedaży.

Realizacja procedury rezerwacji rozpoczyna się od wprowadzenia (w odpowiedzi na pojawiającą się sekwencję pytań) danych identyfikujących relację oraz datę odjazdu pociągu. Powoduje to pojawienie się na ekranie komunikatu o dostępnej w danym dniu liczbie wolnych miejsc, wg ich rodzaju oraz klasy w całym składzie pociągu. Operator terminala ma możliwość wyboru trybu dokonywania rezerwacji, a w przypadku braku w danym dniu miejsc (co sygnalizowane jest odpowiednim komunikatem) wycofanie się z dalszej drogi realizacji procedury i ewentualną zmianę warunków rezerwacji.

Rezerwacja miejsc może być wykonywana w dwojakim trybie (alternatywnie):

- **rezerwacji systemowej,** w ramach której przydział rezerwowanych miejsc odbywa się automatycznie, tzn. bez udziału operatora. Realizacja tego rodzaju rezerwacji przebiegu wg następujących etapów:

— deklaracja liczby miejsc przeznaczonych do zarezerwowania wg rodzaju (damskie, męskie, rodzinne) i klasy (sypialne 1 i 2 klasa, kuszetki)

— rezerwacja wstępna (tzw. symulacja rezerwacji), polegająca na przeszukiwaniu składu pociągu, wg zadeklarowanej liczby i rodzaju miejsc oraz przyjętych priorytetów, w wyniku czego na ekranie pojawia się komunikat z „propozycją rezerwacji". Zaakceptowanie (po uzgodnieniu z klientem) i potwierdzenie przyjęcia tej propozycji powoduje przejście do etapu następnego. Nie zaakceptowanie podanej „propozycji" powoduje powrót do początku procedury

— rezerwacja właściwa, polegająca na zaktualizowaniu zawartości zbiorów w oparciu o dane poprzedniego etapu

- **rezerwacji z udziałem operatora,** w ramach której o przydziale wybranego miejsca decyduje operator w oparciu o wyświelony na ekranie diagram (grafik) odpowiedniego wagonu (ze względu na żądany typ, rodzaj i liczbę miejsc). W ramach procedury rezerwacji operatorskiej istnieje możliwość przeszukiwania dowolnego składu pociągu odjeżdżającego w danym dniu, a w przypadku braku miejsc żądanego rodzaju — możliwość zmiany warunków rezerwacji. Po wyszukaniu zadeklarowanych na wstępie miejsc, realizacja procedury przebiega tak samo jak w rezerwacji systemowej.

W trakcie wykonywania procedury rezerwacji następuje automatyczne bilansowanie miejsc wg rodzaju i klasy. Brak jakiegokolwiek z rodzajów miejsc jest natychmiast sygnalizowany wyświetleniem odpowiedniego komunikatu. Potwierdzenie rezerwacji dla klienta odbywa się w sposób tradycyjny — przez ręczne wypisanie biletu przez kasjerkę.

W module „Rezerwacja" realizowana jest również procedura odwołania (skasowania) rezerwacji. W module „Zbiorcze bilansowanie miejsc sypialnych i kuszetek" — w dowolnym momencie pracy systemu — może być na żądanie użytkownika sporządzone i wyprowadzone na drukarce wierszowej zestawienie z wykazem relacji, obrazujące stan dokonanej rezerwacji miejsc wg rodzaju i klasy w poszczególnych dniach przyjętego okresu rezerwacji. Moduł „Emisja dokumentacji użytkowej" zamyka cykl codziennej pracy systemu. W ramach tego modułu tworzone i wyprowadzane są na drukarce wierszowej diagramy konduktorskie (w postaci akceptowanej przez PKP), będące podstawową dokumentacją przewozową w wagonach sypialnych i kuszetkowych.

Powyżej opisano moduły systemu związane z realizacją jego podstawowej funkcji, jaką jest rezerwacja miejsc. Całość oprogramowania użytkowego jest zarządzana przez moduł „Nadzorowanie i sterowanie pracą systemu". Po zrealizowaniu określonego programu użytkowego lub jego części (wyznaczonych przewidzianymi w programie punktami przerwania) następuje każdorazowo przejście do programu sterującego dla danego terminala i sprawdzenie czy nie wystąpił błąd w systemie, a także przeglądanie zbioru depesz (czy nie ma wśród nich komunikatu przeznaczonego dla określonego terminala). Następnie system wysyła zapytanie do operatora terminala: czy chce on nadać komunikat do głównego operatora systemu.

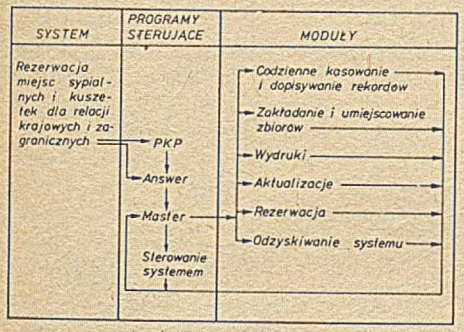
W przypadku stwierdzenia na terminalu błędu w systemie (procedurze) lub w programie, następuje przejście do procedur awaryjnego odzyskiwania systemu. Usunięcie błędu w systemie może być wykonywane:

- na poziomie programu użytkowego — polega ono na obsłudze wyznaczonych w programie punktów przerwań, dla których w zbiorze specjalnym systemu zapisane są wszystkie niezbędne informacje pozwalające na wznowienie jego pracy po awarii; odtworzenie stanu pracy programu (od punktu przerwania) realizowane jest przy wykorzystaniu zapisu „śladu" wprowadzonych transakcji, do momentu kiedy wystąpiło przerwanie; na podstawie odczytanych informacji ze zbioru specjalnego, system doprowadza zapis w (wykorzystywanych przez program) zbiorach danych do zgodności, a na ekranie wyświetlane

są odpowiednie komunikaty oraz zestawienie wprowadzonych dotąd danych transakcyjnych,

● na poziomie systemu — polega ono np. na usunięciu błędów w zbiorach danych systemu; w tym przypadku następuje blokada całego systemu (wszystkich zidentyfikowanych w konfiguracji sprzętowej terminali) i przejście do procedur awaryjnego odzyskiwania systemu, które mogą być wykonywane automatycznie lub przy współudziale operatora głównego; po usunięciu błędu w systemie następuje jego odblokowanie — z terminala głównego przesłany zostaje odpowiedni komunikat do pozostałych terminali, informujący o wznowieniu pracy systemu rezerwacji. Kontynuacja pracy terminali kasowych realizowana jest od punktu, w którym nastąpiło przerwanie pracy systemu, procedury lub części programu.

Schemat realizacji modułów systemu rezerwacji miejsc przedstawiono na rysunku 1.



Rys. 1. Schemat realizacji modułów systemu rezerwacji miejsc

ŚRODKI TECHNICZNE

System rezerwacji miejsc realizowany jest na sprzęcie minikomputerowym DATAPOINT model 6600 o następującej zróżnicowanej dla poszczególnych etapów konfiguracji sprzętowej:

● konfiguracja dla etapu wdrożenia i eksploatacji próbnej (do połowy 1980 r.):

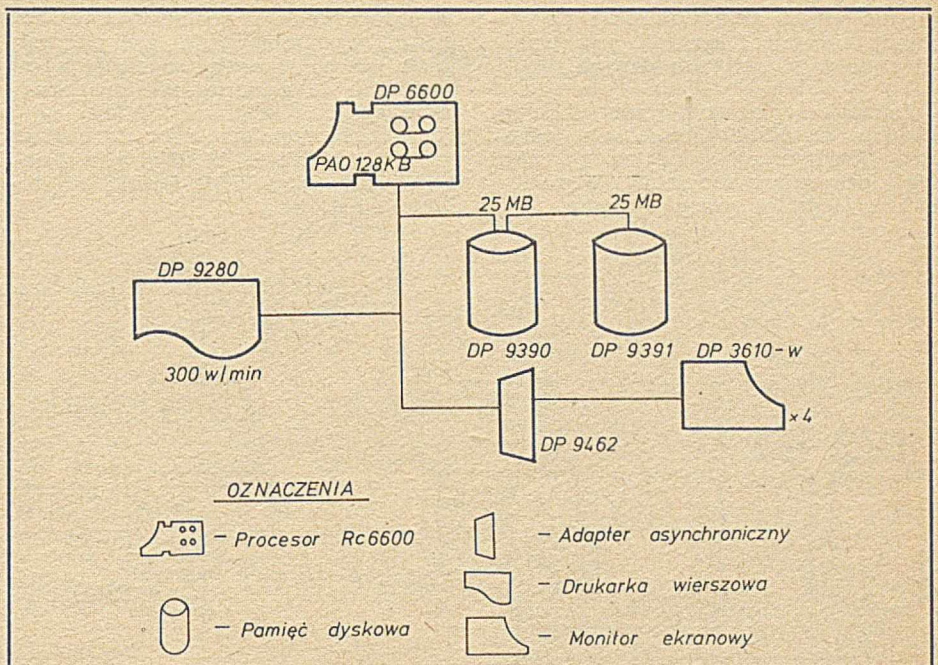
— jednostka centralna DP 6600 z pamięcią operacyjną o pojemności 128 KB

— 2 jednostki pamięci dyskowej o pojemności 25 MB każda (po jednej dla przechowywania oryginałów i kopii zbiorów danych)

— 1 drukarka wierszowa

— 8-kanałowy asynchroniczny adapter komunikacyjny

— 4 monitory ekranowe z klawiaturą (pojemność ekranu 24 wiersze po 80 znaków); przetwarzanie dla tego etapu realizowane było dla 10 wytypowanych relacji krajowych i 5 zagranicznych; schemat konfiguracji sprzętowej dla tego etapu podano na rysunku 2;



Rys. 2. Konfiguracja sprzętu dla wdrożenia i próbnej eksploatacji systemu

● konfiguracja dla etapu eksploatacji bieżącej (od III kw. 1980):
— 3 jednostki centralne DP-6600 o parametrach jak poprzednio
— 2 drukarki wierszowe
— 10 jednostek pamięci dyskowej o pojemności 25 MB każda lub 4 jednostki o pojemności 62,5 MB każda (zlokalizowane przy dwóch jednostkach centralnych z równym podziałem dla przechowywania oryginałów i kopii zbiorów danych)
— 12 terminali konwersacyjnych (dla terminali przyłączonych na odległość większą niż 10 km wymagane jest łącze telefoniczne z uwzględnieniem pomocy modemów).
Schemat konfiguracji sprzętowej dla etapu eksploatacji bieżącej przedstawiono na rysunku 3.

Przewiduje się, że docelowo systemem rezerwacji miejsc objętych będzie 110 relacji (66 krajowych i 44 zagraniczne).

System wymaga następującego oprogramowania podstawowego:
● dyskowego systemu operacyjnego DOS.D (wersja 2.4 lub kolejna)
● systemu podziału czasu DATASHARE (wersja IV)
● pakietu edycji tekstowej DSTEXT
● systemu ARC [3]) dla konfiguracji sprzętowej etapu eksploatacji bieżącej,

a także użycia następujących języków programowania:

● DATABUS — dla programów wykonywanych pod nadzorem systemu DATASHARE

● ASSEMBLER — dla programów odzyskiwania systemu oraz kontrolno-testujących, wykonywanych pod nadzorem systemu DOS.

ZABEZPIECZENIE INFORMACJI

Systemy rezerwacyjne, realizowane w trybie bezpośrednim i w sposób ciągły, wymagają maksymalnego zabezpieczenia danych oraz stworzenia takich narzędzi, które częściowo będą zapobiegać awariom systemu, a w razie ich powstania pozwolą złagodzić skutki, a następnie umożliwią szybkie odtworzenie poprzedniego stanu pracy.

W systemie przewidziano zabezpieczenia typu programowego, technicznego i organizacyjno-eksploatacyjnego.

Zabezpieczenia programowe chronią przed:

● niepożądanym dostępem — przez zastosowanie systemu haseł dla osób obsługujących terminale operacyjne (kasowe) i terminal główny oraz przez przypisanie poszczególnym terminalom określonym funkcji systemu użytkowego zgodnie z ich przeznaczeniem (kasowy, główny, specjalny) i kwalifikacjami obsługującego personelu

● przed awariami — przez zastosowanie metody dublowania zbiorów danych oraz wszystkich programów sterujących i użytkowych, a także systemu DATASHARE, prowadzenie zapisu „śladu" pracy zamkniętej lo-

[3]) ARC — Attached Resource Computer (komputer z dołączanymi zasobami). Wieloprocesorowy system minikomputerów DATAPOINT połączonych ze sobą szyną międzyprocesorową w postaci łącza mikrofalowego. W systemie tym istnieje możliwość specjalizacji (np. obsługa zbiorów, terminali, drukarek) i współpracy poszczególnych minikomputerów, a także łączenia systemów i współpracy z dużymi komputerami.

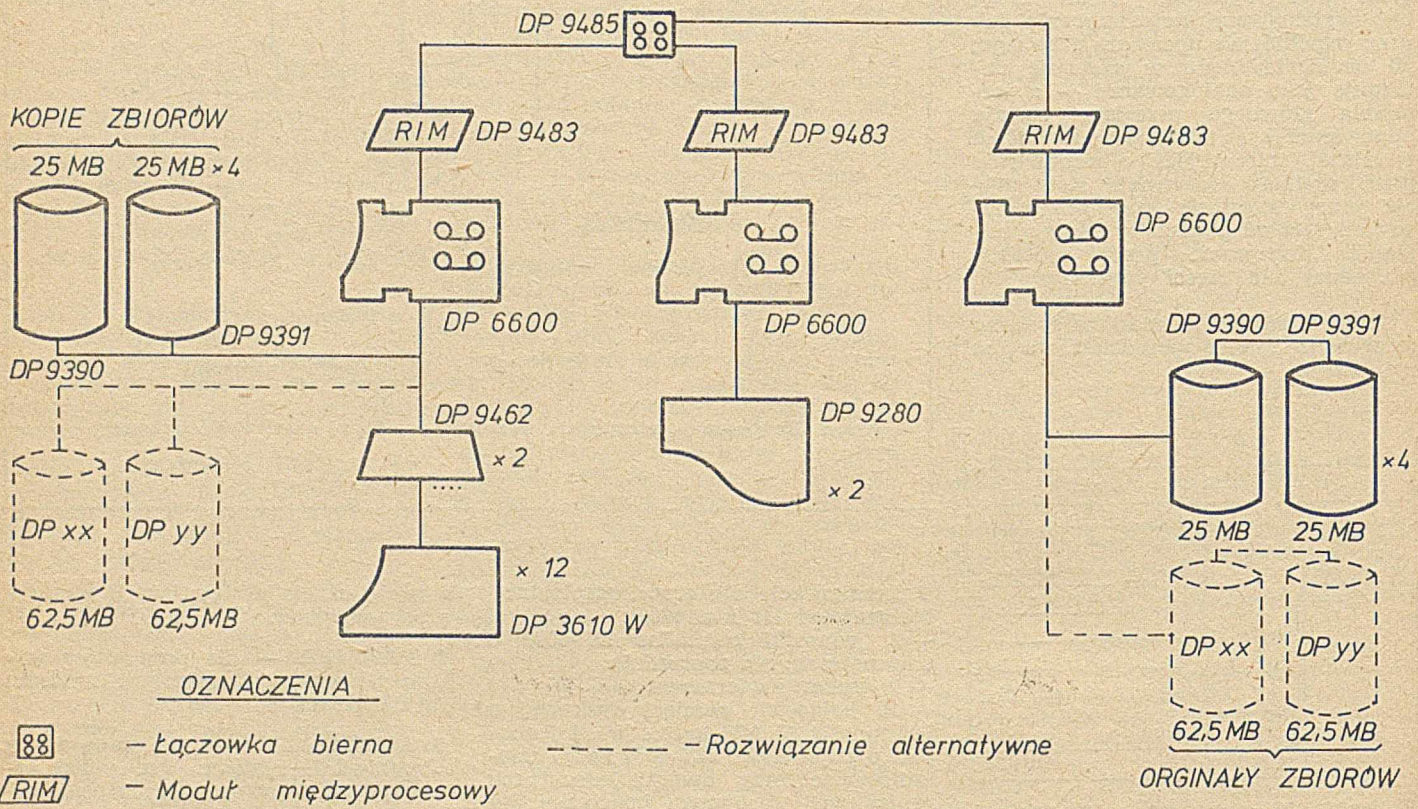
gicznie części programu oraz prowadzenie okresowej kontroli poprawności zapisów w zbiorach danych.

Zabezpieczenia techniczne polegają głównie na zdublowaniu tych jednostek konfiguracji sprzętowej, które mają decydujące znaczenie dla zapewnienia ciągłości pracy systemu.

- usprawnienie obsługi klientów dzięki wyeliminowaniu dotychczasowych czasochłonnych czynności
- możliwość zastosowania opracowanego systemu rezerwacji miejsc jako rozwiązania powtarzalnego do obsługi kas przedsprzedaży biletów w pozostałych placówkach PBP ORBIS na terenie całego kraju

tego rodzaju działalność na terenie kraju. Od III kw. 1980 r. rozpoczęto prace w kierunku modernizacji rozwiązań systemowych polegającej na:

- opracowaniu modułu bezpośredniego drukowania biletów (potwierdzeń rezerwacji) oraz rozbudowę terminali o drukarki



Rys. 3. Schemat konfiguracji ARC dla obsługi 110 relacji ze zdublowanymi zbiorami danych

Zabezpieczenia organizacyjno-eksploatacyjne polegają na systematycznym stosowaniu przez użytkownika działań zapobiegawczych (planowane konserwacje i bieżące przeglądy sprzętu komputerowego, okresowe uruchamianie procedury kontroli poprawności stanu zbiorów danych, zabezpieczania pomieszczeń przed dostępem osób nieupoważnionych).

EFEKTY ZASTOSOWANIA SYSTEMU

W wyniku zastosowania systemu osiągnięto już lub są spodziewane następujące efekty:

- wyeliminowanie pomyłek, w większości polegających na dublowaniu rezerwacji miejsca.

* * *

Opisane rozwiązanie, chociaż opracowane dla węzła warszawskiego jest systemem powtarzalnym, uwzględnia ono bowiem zasady przedsprzedaży i rezerwacji miejsc obowiązujące we wszystkich Oddziałach PBP ORBIS. W konsekwencji tego system może być zastosowany i wdrożony we wszystkich placówkach prowadzących

- przyłączeniu do systemu modułu obsługi dalekopisów, przez które również będzie można wprowadzać do systemu informacje dotyczące rezerwacji miejsc
- sprzężenie systemu rezerwacji miejsc sypialnych i kuszetek z systemem rezerwacji promowej (opracowanym na analogicznym sprzęcie komputerowym dla Polskiej Żeglugi Bałtyckiej).

Małgorzata ŚWITALSKA -JELEŃKOWSKA
ZETO Wrocław

Środowisko informatyków zelektryzował opublikowany w „Życiu Warszawy" list (z 9 lutego br.) prof. prof. A. Bliklego, L. Łukaszewicza, A. Mazurkiewicza i W. M. Turskiego, wymierzony przeciw książce Andrzeja Targowskiego „Informatyka. Modele systemów i rozwoju". W odpowiedzi na to Wydawca książki (dyrektor PWE) zwrócił się do autorów listu o merytoryczne argumenty, dowodzące niskiej wartości publikacji. W następnym numerze przedstawimy cztery recenzje, które uzasadniają ostre słowa listu profesorów. (Red.)

RURY - system obliczania samokompensacji rurociągów

Rurociągi stosowane są w wielu obiektach przemysłowych. Przy założeniu, że temperatura przenoszonego czynnika nie różni się znacznie od temperatury otoczenia, obliczenia wytrzymałościowe rurociągów są stosunkowo mało skomplikowane. Przy projektowaniu układów pracujących w wysokich temperaturach pojawia się natomiast problem kompensacji wydłużeń cieplnych. W przypadku rurociągów średnio- i wysokoprężnych najczęściej nie ma możliwości stosowania specjalnych elementów kompensujących dylatacje termiczne i dlatego rurociągi takie muszą być tak zaprojektowane, by sam układ rurociągu mógł skompensować wydłużenia cieplne bez pojawienia się odkształceń plastycznych.

Projektowanie rurociągu odbywa się następującymi etapami:

- ustalana jest trasa rurociągu lub układu rurociągów, wynikająca z topografii obiektu
- określane są wymiary elementów rurociągu, wynikające z warunków przepływowych
- przeprowadzane są obliczenia wytrzymałościowe, sprawdzające zdolności samokompensacyjne rurociągu.

Obliczanie zdolności samokompensacji rurociągu jest problemem bardzo skomplikowanym rachunkowo, ponieważ rurociąg jest układem wielokrotnie statycznie niewyznaczalnym, co prowadzi do konieczności rozwiązania układu równań liniowych. Istnieje szereg metod przybliżonych, wykreślno-analitycznych i wykreślnych, służących wykonywaniu tych obliczeń. Należy zaznaczyć, że za rzetelne można uważać jedynie wyniki uzyskane metodą ściśle analityczną lub modelową, zwłaszcza gdy obliczany jest rurociąg o skomplikowanych kształtach.

Zastosowanie metod przybliżonych obliczeń samokompensacji rurociągów zmniejszyło się z chwilą pojawienia się elektronicznych maszyn cyfrowych. Dzięki nim obliczenia rurociągów mogą być przeprowadzane z zachowaniem dużej dokładności rachunkowej i dużej zgodności modelu z obiektem rzeczywistym.

Obliczenia komputerowe ogromnie skracają czas obliczeń, eliminują możliwość powstania błędów rachunkowych i pozwalają projektantowi na skoncentrowanie się na działalności twórczej. Zadanie projektanta ogranicza się w tym przypadku do sformułowania zadania dla komputera, a następnie do analizy otrzymanych wyników. Opisany dalej system ma za zadanie służyć jako narzędzie dla projektanta tych rurociągów, w których nie może być pominięta samokompensacja termiczna.

Rurociąg jest skomplikowanym tworem powłokowym. Obliczenia wytrzymałościowe można by prowadzić traktując powierzchnię jego ścianek jako powłokę. Taki sposób modelowania byłby najbardziej zgodny z rzeczywistością, nie wymagałby wprowadzania uproszczeń oraz dawałby możliwość wiernego odwzorowywania wszelkich kształtów rurociągu. Ma on jednak poważną niedogodność. Rozwiązanie nawet stosunkowo prostego układu rurociągów urasta do poważnego problemu obliczeniowego i czyni obliczenia bardzo kosztownymi w przypadku bardziej skomplikowanych układów.

Obecnie do obliczeń wytrzymałościowych rurociągów powszechnie stosowane są metody uproszczone. Odcinki rurociągu modelowane są za pomocą elementów prętowych o kształcie określonym przez linię środkową odcinka rurociągu. Przy takim uproszczeniu przekroje rurociągu sprowadzane są do punktów, którym przypisanych jest sześć stopni swobody (przemieszczenia w trzech kierunkach i obroty wokół trzech osi układu współrzędnych). W tym ujęciu układ rurociągów może być rozwiązywany metodami stosowanymi do obliczania ram przestrzennych. Istnieje wiele programów przeznaczonych do obliczeń wytrzymałościowych konstrukcji prętowych; nie uwzględniają one jednak specyfiki rurociągu, polegającej na tym, że rurociąg może być obciążony ciśnieniem wewnętrznym i że dla otrzymania poprawnych wyników należy uwzględnić wpływ owalizacji przekroju poprzecznego rury na jej sztywność.

Prezentowany system obliczeń samokompensacji rurociągów opiera się na następujących założeniach:

- rurociąg modelowany jest trzema rodzajami elementów: odcinkiem prostym, łukiem lub odcinkiem nieodkształcalnym
- szereg kolejno następujących po sobie elementów tworzy gałąź
- poszczególne gałęzie łączą się z sobą w punktach zwanych węzłami rurociągu; ze względu na sposób organizacji obliczeń przyjęto, że w punkcie węzłowym nie mogą się zbiegać więcej niż trzy gałęzie
- końce układu rurociągu są sztywno utwierdzone, mogą jednak podlegać wymuszonym przemieszczeniom i obrotom wywołanym przemieszczeniami i obrotami króćców urządzeń, do których są przytwierdzone
- rurociąg jest wykonany z materiału elastoplastycznego, odpowiadającego wymaganiom stawianym przez prawo Hook'a i pracuje w zakresie naprężeń, w którym prawo to obowiązuje
- owalizacja przekroju poprzecznego rury jest uwzględniana tylko w elementach zakrzywionych — poprzez wprowadzenie współczynnika zmniejszającego sztywność łuku na zginanie.

Rurociąg może być obciążony:

- siłami skupionymi
- obciążeniem ciągłym
- przyrostem temperatury
- ciśnieniem wewnętrznym

Uwzględniane są:

- naciąg wstępny
- wymuszone przemieszczenia końców gałęzi
- następujące rodzaje podparć: zamocowanie całkowite, zamocowanie częściowe oraz podparcie sprężyste z możliwością wprowadzenia wstępnego naciągu sprężyny.

W wyniku obliczeń otrzymuje się: przemieszczenia, siły wewnętrzne, maksymalne naprężenia zredukowane (we wszystkich punktach charakterystycznych rurociągu), a także sumaryczne siły działające na punkty zbiegu gałęzi.

Za pomocą systemu mogą być wykonywane obliczenia wytrzymałościowe rurociągu w stanie gorącym i zimnym. System umożliwia również projektantowi zastosowanie właściwego rozmieszczenia podparć i wprowadzenie odpowiednich napięć wstępnych zawieszeń sprężystych. Umożliwia on również dobranie najkorzystniejszego naciągu wstępnego rurociągu — zarówno co do jego wielkości, jak też i miejsca jego wykonania.

ROZWIĄZANIE SYSTEMU

System obliczeń wytrzymałościowych rurociągów RURY został napisany w języku FORTRAN-IV poziom G oraz w języku ASSEMBLER i może być wykonywany na komputerach Jednolitego Systemu oraz komputerach IBM serii 360 i 370, wyposażonych w system operacyjny OS. Do wykonania obliczeń potrzebna jest następująca konfiguracja:

- pamięć operacyjna — nie mniej niż 256 KB
- czytnik kart dziurkowanych
- drukarka wierszowa
- jednostka pamięci dyskowej.

System składa się z czterech niezależnych programów: programu sprawdzania danych DANE oraz trzech wersji programu obliczeniowego STAT20, STAT50 i STAT100. Wersje te różnią się między sobą możliwościami obliczeniowymi (wielkością problemu, który może być rozwiązany, oraz pojemnością pamięci operacyjnej, potrzebnej do wykonania programu). Wielkość problemu określona jest przez liczbę punktów zbiegu rurociągu (węzłów rurociągu).

Możliwości obliczeniowe systemu oraz pojemność pamięci operacyjnej, potrzebnej dla poszczególnych programów, kształtują się następująco:

- STAT20 — 20 węzłów oraz 110 KB
- STAT50 — 40 węzłów oraz 162 KB
- STAT100 — 100 węzłów oraz 240 KB
- DANE — pamięć 50 KB (dla wszystkich wersji).

System działa pod kontrolą procedury napisanej w Języku Sterowania Zadaniami, która może być skatalogowana w systemowej bibliotece procedur. Procedura wyposażona jest w parametr umożliwiający użytkownikowi dobór odpowiedniej do jego potrzeb wersji programu obliczeniowego. Tego typu rozwiązanie ma znaczenie przy wykonywaniu obliczeń na komputerach pracujących w wieloprogramowości i umożliwia dostosowanie wielkości strefy pamięci (a tym samym kosztów obliczeń) do wielkości problemu obliczeniowego.

Dane wejściowe wprowadzane są na kartach dziurkowanych. Stosowany jest zapis formatowy. Rodzaj informacji identyfikowany jest przez typ karty. Program DANE podaje diagnostykę ewentualnych błędów, informacje odnośnie wielkości problemu, a także generuje zbiór danych będący wejściem dla programu pisaka X-Y, umożliwiającego otrzymywanie rysunków aksonometrycznych rurociągu wraz z wprowadzonymi oznaczeniami. W przypadku nie wykrycia błędów w danych, wykonywany jest jeden z wybranych przez użytkownika programów obliczeniowych.

Obliczenia są prowadzone analogicznie jak w analizie wytrzymałościowej konstrukcji, za pomocą metody elementów skończonych. Dane do programu definiują ciąg następujących po sobie elementów trzech typów (odcinek prosty, łuk i element nieodkształcalny). Poprzez eliminację punktów wewnętrznych gałęzi tworzona jest macierz sztywności gałęzi, zredukowana do jej końców. Macierze sztywności gałęzi zbierane są w macierz sztywności rurociągu, zredukowaną do punktów połączeń gałęzi. Macierz sztywności rurociągu tworzona jest w pamięci operacyjnej w zapisie klatkowym. Liczba klatek w wierszu nie przekracza czterech. Po utworzeniu macierzy sztywności całej struktury rozwiązywany jest układ równań opisujący równowagę rurociągu.

W wyniku rozwiązania układu otrzymuje się przemieszczenia punktów zbiegu gałęzi (węzłów). Na podstawie przemieszczeń końców gałęzi obliczane są przemieszczenia punktów wewnętrznych gałęzi, które po pomnożeniu przez odpowiednie macierze sztywności elementów dają siły we wszystkich punktach charakterystycznych rurociągu.

Znając wartości sił można obliczyć naprężenia zredukowane we wszystkich punktach charakterystycznych rurociągu. Program obliczeniowy generuje również zbiór zawierający przemieszczenia i będący wejściem do programu umożliwiającego narysowanie rurociągu odkształconego na tle rurociągu nieodkształconego.

WDROŻENIA I EKSPLOATACJA

Prace nad systemem obliczeń rurociągów energetycznych zostały podjęte w b. ZETO-ZOWAR w 1973 r. W pierwotnej wersji program ten umożliwiał obliczanie rurociągów składających się maksymalnie z 30 gałęzi i 15 węzłów. Rurociąg mógł być obciążony przyrostem temperatury, ciśnieniem wewnętrznym i siłami skupionymi. Dodatkowo program uwzględniał przemieszczenia króćców i zawieszenie rurociągu na podporach stałych i sprężystych. Eksploatacja programu wykazała, że ma on zbyt małe możliwości obliczeniowe. Ograniczenie do 15 węzłów powodowało, że obliczanie bardziej skomplikowanych układów było niemożliwe bez wprowadzania znacznych uproszczeń.

Podjęto więc prace nad nowym programem, który umożliwiał obliczanie rurociągów zawierających do $N = 100$ węzłów i $2N + 1$ gałęzi. Ponadto zwiększono możliwości obliczeniowe programu poprzez uwzględnienie w obliczeniach obciążenia rozłożonego i wprowadzanego naciągu wstępnego. Obecna postać programu została ukształtowana w wyniku prac prowadzonych przy tworzeniu systemu obliczeń dynamicznych rurociągu, uwzględniających zjawiska sejsmiczne.

Program obliczeń samokompensacji rurociągów został włączony do systemu obliczeń dynamicznych z głównym zadaniem utworzenia macierzy sztywności, wykorzystywanej w dalszej analizie. Modyfikacje polegały na zastosowaniu innej metody obliczania macierzy sztywności gałęzi, co umożliwiło zastąpienie całkowania numerycznego — całkowaniem analitycznym oraz zwiększenie efektywności obliczeń przez wprowadzenie zbiorów roboczych, przechowujących wyniki pośrednie. Zmiany te spowodowały, że czas potrzebny do wykonania obliczeń samokompensacji rurociągów zmniejszył się prawie dwukrotnie. Sposób przygotowania danych dla programu obliczeniowego oraz postać wyprowadzanych wyników pozostały — poza niewielkimi zmianami — takie jak w wersji początkowej.

W ciągu kilkuletnich prac nad omawianym systemem zespół autorski nawiązał kontakty z wieloma przedsiębiorstwami, projektującymi rurociągi na różne potrzeby. Obliczano rurociągi zaprojektowane m.in. w biurach projektowych MEGADEX, ENERGOPROJEKT, PROZEMAK, CEWOK, dla odbiorców krajowych i zagranicznych. System został zainstalowany w ZETO Katowice, w Głównym Biurze Studiów i Projektów Przeróbki Węgla SEPARATOR oraz w SOETO Warszawa. W wyniku przeprowadzonych obliczeń sugerowano projektantom rurociągów wprowadzenie zmian do projektów, np. zastosowanie naciągu wstępnego lub zmianę jego wielkości, zmianę rozmieszczenia podpór i zawieszeń sprężystych itp.

W grudniu 1979 r. Centrum Projektowania i Zastosowań Informatyki zorganizowało seminarium poświęcone systemowi obliczeń wytrzymałościowych rurociągów. Obecni byli przedstawiciele przedsiębiorstw z całego kraju, zainteresowanych omawianym problemem. Zgłoszono szereg postulatów dotyczących sposobu wprowadzania danych, konieczności weryfikacji programu, kosztów obliczeń itd., które są uwzględniane przez CPiZI w modernizacji systemu RURY.

Jak już zaznaczono, sposób przygotowania danych opisujących rurociąg pozostał praktycznie nie zmieniony od chwili rozpoczęcia prac nad systemem. W chwili obecnej wielu użytkowników sugeruje jego uproszczenie, gdyż jest zbyt pracochłonny. Szereg wartości wpisywanych na kartach danych jest niepotrzebnie powtarzanych kilkakrotnie, co stwarza możliwość pomyłek i znacznie wydłuża czas opisu rurociągu. Rozważana jest możliwość wykorzystania minikomputerów i monitora graficznego na etapie sprawdzania danych. Wprowadzanie danych z klawiatury wyeliminowałoby konieczność pisania wielu kart dziurkowanych, a także umożliwiłoby projektantowi natychmiastowe sprawdzenie poprawności definicji obiektu.

Na podstawie wykonanych w CPiZI na komputerach R-22, R-32 oraz IBM-360/50 obliczeń kilkudziesięciu rurociągów o różnej liczbie węzłów i elementów stwierdzono, że czas obliczeń waha się w granicach 3—5 s CPU/element. Czas obliczeń rurociągu składającego się z 20 węzłów wynosi ok. 15 min (w zależności od typu komputera). W miarę wzrostu liczby węzłów rośnie oczywiście czas i koszt obliczeń. Jest on jednak zdecydowanie mniejszy od kosztów ponoszonych przez przedsiębiorstwa projektujące rurociągi na wykonanie obliczeń sprawdzających w specjalistycznych ośrodkach obliczeniowych na Zachodzie.

W obecnej wersji system umożliwia otrzymywanie aksonometrycznych rysunków rurociągu nieodkształconego, a także odkształconego. Prace nad urozmaiceniem form graficznych wyprowadzanych wyników będą kontynuowane. Pozwoli to na otrzymywanie wyników obliczeń rurociągów w postaci czytelnej dla inżyniera nie zajmującego się techniką komputerową.

Sądzimy, że w obecnej postaci system obliczeń wytrzymałościowych rurociągów, opracowany i w dalszym ciągu rozwijany w CPiZI, ma szanse konkurować z odpowiednimi systemami zagranicznymi. Powinno to przynieść gospodarce narodowej, a konkretnie — zainteresowanym przedsiębiorstwom specjalistycznym, wymierne efekty w postaci zmniejszenia wydatków dewizowych oraz usprawnienia procesu projektowania.

Lech GAWRYS
Jan WISZNIEWSKI
Centrum Projektowania i Zastosowań Informatyki Warszawa

ZWIĄZKI ZAWODOWE

„Solidarność" w ZETO

W dniach 14—15 stycznia br. odbyło się trzecie plenarne posiedzenie Tymczasowego Komitetu Porozumiewawczego NSZZ „Solidarność" Zakładów Elektronicznej Techniki Obliczeniowej. Komitet zmienił nazwę na Tymczasową Komisję Porozumiewawczą. Komisja określiła następująco cel i formy swej pracy:

- podstawowym celem Komisji jest umożliwienie wymiany informacji o działalności poszczególnych organizacji związkowych w sieci ZETO oraz ustalenie wspólnego stanowiska i podejmowanie wspólnych działań dotyczących przedsiębiorstw i załóg
- Komisja uznaje za celowe podjęcie współpracy z komisjami reprezentującymi informatyków z innych środowisk, zrzeszonych w NSZZ „Solidarność", widząc potrzebę utworzenia Sekcji Zawodowej Informatyków na szczeblu krajowym
- Tymczasowa Komisja Porozumiewawcza składa się z przewodniczących Komisji Zakładowych (w razie nieobecności przewodniczącego — w pracy Komisji może uczestniczyć członek Komisji Zakładowej)
- formą pracy Komisji jest plenarne zebranie jej członków
- dla prowadzenia bieżącej administracji, korespondencji, w celu inicjowania i zwoływania zebrań, przekazywania materiałów informacyjnych — powołuje się Sekretariat TKP w składzie:

Włodzimierz Lipiński (Warszawa) — tel. 22 67 02
Zofia Winawer (Warszawa) — tel. 23 74 04
Zbigniew Kaśmierczak (Wrocław) — tel. 44 54 31 w. 143
Krzysztof Markowski (Gdańsk) — tel. 21 66 46 w. 209

- w razie potrzeby Komisja może spośród swego grona oraz towarzyszących ekspertów powoływać zespoły problemowe dla przygotowania projektów uchwał oraz propozycji rozwiązania problemów.

Jednocześnie powołane zostały zespoły problemowe, których zadaniem jest opracowywanie wspólnego stanowiska wobec istotnych spraw dotyczących członków Związku. Komisje problemowe rozpoczęły prace przygotowawcze, które będą przedmiotem obrad następnych posiedzeń plenarnych. Uzgodnione stanowiska Tymczasowej Komisji Porozumiewawczej będą przedstawiane w formie komunikatów, oświadczeń i uchwał.

Pracę rozpoczęły następujące Komisje:

- komisja ds. układu zbiorowego dla pracowników informatyki w sieci ZETO; pracami nad układem kierują przedstawiciele Centrum Projektowania i Zastosowań Informatyki i ZETO Gdańsk
- komisja ds. nowego taryfikatora dla pracowników informatyki, której zadaniem jest ujednolicenie stanowiska wszystkich członków „Solidarności" w ZETO; prace te koordynują przedstawiciele Związku z ZETO Poznań
- komisja ds. organizacji samorządu w Zakładach Elektronicznej Techniki Obliczeniowej; pracami tymi kierują członkowie „Solidarności" z ZETO Łódź
- komisja zbierająca i kompletująca wszelkie wnioski pod adresem władz zwierzchnich; pracami tej komisji kieruje Komisja Zakładowa ZETO Wrocław.

Powołana została ponadto grupa problemowa do spraw scentralizowanego funduszu socjalnego oraz gospodarki funduszem socjalnym na szczeblu przedsiębiorstwa (zorganizowana przez „Solidarność" w ZETO Kraków), a także grupa specjalistów ekonomicznych do spraw reformy gospodarczej (ustala opinię związkową na temat zmian w funkcjonowaniu przedsiębiorstw informatycznych).

Włodzimierz LIPIŃSKI

NSZZPI o problemach płacowych

Właściwa regulacja płac dla informatyków jest jednym z pierwszoplanowych zadań naszego Związku. Pragnęliśmy rozwiązać to zagadnienie generalnie już w październiku ub.r., interweniując w Komitecie Informatyki oraz Ministerstwie Pracy, Płac i Spraw Socjalnych. Interwencje te nie dały jednak bezpośrednich pozytywnych rezultatów. Wiceminister Dudziński stwierdził, że informatycy jako branża nie byli brani pod uwagę i że jesteśmy pierwszą ich reprezentacją. Pozostała więc jedynie droga interwencji poprzez resorty. MPPiSS zobowiązało się wziąć pod uwagę istnienie naszej grupy zawodowej w przyszłości. Mamy nadzieję, że zmiany personalne w ministerstwie nie zmienią tego stanowiska.

Wprowadzone podwyżki płac w sieci ZETO (wynegocjowane 15 października ub.r.), w wysokości nie niższej od 500 zł dla każdego pracownika — były niższe od przeciętnej w kraju, miały charakter zaliczkowy oraz zostały wprowadzone w ramach tymczasowego, bezsensownego taryfikatora płac. Taka sama sytuacja wystąpiła w GUS Radom. To dyskryminujące podejście do pracowników naszej grupy zawodowej spowodowało interwencje NSZZPI w oparciu o treść porozumienia, zawartego pomiędzy ministrem NSzWiT a NSZZ „Solidarność" w Gdańsku. Żądaliśmy zwiększenia płac o dodatkowe 300 zł i wydzielenia kwoty podwyżki na liście płac. (Analogiczne działania w odniesieniu do sieci ZETO prowadził ZZE). Interwencja ta dała pozytywne rezultaty; w chwili oddania tego tekstu do druku załogi przedsiębiorstw ZETO i GUS Radom zastanawiają się nad zasadami rozdziału przyznanych środków finansowych.

Zdaniem NSZZPI otrzymane kwoty w wysokości 300 zł powinny być wypłacane w formie dodatku, jako odrębna pozycja listy płac. Fakt ten znalazł potwierdzenie w ustaleniach między Zjednoczeniem a Związkami Zawodowymi. Sposób wypłaty podwyżki z października może, ale nie musi ulec zmianie. Sygnalizujemy, że w chwili obecnej załogi poszczególnych przedsiębiorstw powinny doprowadzić do porozumień ze swoimi dyrekcjami co do formy wypłaty kwot przyznanych zarówno w październiku, jak i w styczniu. Zweryfikowanie ustaleń z października i wprowadzenie całej przyznanej kwoty 800 zł jako dodatku do płacy stwarza możliwość awansu tym wszystkim pracownikom, którzy są obecnie na tak zwanym górnym pułapie. Decyzja ta jednak obniży praktycznie kwotę podwyżki z obecnie uzyskanej (w ZETO) 846 zł średnio na pracownika — do 800 zł.

Bogdan FIUTOWSKI

Związki zawodowe informatyków — NSZZ „Solidarność" i NSZZ Pracowników Informatyki, po wstępnym etapie organizowania swych szeregów, rozpoczynają już normalną działalność. (Informują o tym poniższe wypowiedzi). Pojawiają się też pierwsze oznaki współpracy; dotychczasowa nieufność „Solidarności" w stosunku do NSZZPI, która znalazła wyraz w artykule W. Grudzińskiego i R. Kasprzyka „Niezależni i samorządni" (INFORMATYKA, nr 12/80), zdaje się maleć. Ponadto argumenty autorów wspomnianego tekstu zostały podważone przez NSZZPI. Nie jesteśmy w tym przypadku w stanie sami rozstrzygać o prawdzie, dlatego też zaprosiliśmy do Redakcji przedstawicieli obu związków. Zapis dyskusji zostanie zamieszczony w numerze 5—6/81.

REDAKCJA

Prace nad Układem Zbiorowym

Głos „Solidarności"

Próba przygotowania układu zbiorowego dla pracowników informatyki nasuwa wiele wątpliwości natury interpretacyjnej. Układ zbiorowy pracy jest formą umowy społecznej między grupą pracowników reprezentujących zawód lub branżę, a pracodawcą, reprezentowanym najczęściej przez odpowiedniego ministra. Czy jednak informatycy są pracownikami jednej branży? Z całą pewnością nie. Może więc są przedstawicielami jednego zawodu? Na to pytanie również nie można dać jednoznacznie twierdzącej odpowiedzi.

Patrząc od strony zdobytych kwalifikacji, trudno o bardziej różnorodną „zbieraninę" ludzi wykonujących tę samą, w końcu, robotę. Są wśród nas matematycy i ekonomiści, inżynierowie różnych specjalności, psycholodzy, filolodzy i wielu, wielu innych. Pracujemy właściwie wszędzie: w przedsiębiorstwach przemysłowych, instytutach naukowo-badawczych, spółdzielczości, uczelniach, w urzędach administracji centralnej i terenowej oraz w wyspecjalizowanych firmach informatycznych, działających na zasadach pełnego rozrachunku gospodarczego.

Cóż więc byłoby przedmiotem układu zbiorowego dla pracowników informatyki? Oczywiście — sprawy finansowe. Ale to za mało. Do zagwarantowania odpowiedniego wynagrodzenia nie potrzeba zawierać układu zbiorowego, wystarczy taryfowy system płac. Układ zbiorowy dla informatyków powinien określać specyfikę zawodu, warunki pracy w informatyce, zagrożenia zdrowia (nie tylko fizycznego) oraz — wynikające z tych zagrożeń — przywileje dla narażonych na nie pracowników (dodatkowe świadczenia pieniężne, skracanie czasu pracy itp.).

Informatyka nie stwarza, tak jak np. przemysł wydobywczy, ewidentnych zagrożeń zdrowia pracowników, ale z drugiej strony — nikt do tej pory autorytatywnie nie stwierdził, że zagrożenia w ogóle nie występują (choćby promieniowanie rentgenowskie monitorów ekranowych). Następstwa mogą się ujawniać dopiero po wielu latach lub nawet — w następnych pokoleniach. Nie zapominajmy przy tym, że na najbardziej zagrożonych stanowiskach (operatorzy emc i upd) pracują głównie kobiety. Uważamy więc, że najważniejszą częścią układu zbiorowego powinny być postanowienia dotyczące bezpieczeństwa i higieny pracy.

Zawarcie układu zbiorowego dla wszystkich pracowników informatyki w kraju nie jest proste z innego jeszcze powodu. Jest nim wspomniane już wcześniej, bezprecedensowe w historii układów zbiorowych rozproszenie informatyków w gałęziowej strukturze gospodarki narodowej. Dlatego też, gdy w Zjednoczeniu Informatyki (Ministerstwo NSzWiT) podjęto wstępne prace nad projektem Układu Zbiorowego Pracy dla Pracowników Informatyki, musiano ograniczyć zakres podmiotowy przyszłego układu do obszaru zjednoczenia. Pocieszająca jest możliwość rozciągnięcia postanowień układu — za zgodą odpowiednich związków zawodowych i ministrów — na informatyków zatrudnionych poza Zjednoczeniem Informatyki.

W pracach nad projektem układu wzięli udział przedstawiciele trzech związków zawodowych, działających w ośrodkach sieci ZETO i Centrali ZI: NSZZ „Solidarność", NSZZ Pracowników Informatyki i ZZ Energetyków. Po wstępnej ocenie — przez załogi ośrodków ZETO — pierwszej wersji układu, przygotowane zostały wersje druga i trzecia. Ta ostatnia zostanie ponownie przedstawiona do akceptacji pracownikom, tym razem z dołączonymi opiniami MNSzWiT i MPPiSS. Dopiero pozytywna opinia większości członków Związku będzie równoznaczna z udzieleniem mandatu upoważnionym przez KKP NSZZ „Solidarność" przedstawicielom do podpisania układu.

Ostateczna wersja układu zbiorowego musi uwzględnić posierpniowe status quo w polskim życiu społeczno-gospodarczym, w związku z czym nie może być opracowana przed uchwaleniem tak ważnych aktów normatywnych, jak ustawa o związkach, o przedsiębiorstwie, o samorządzie pracowniczym oraz znowelizowany kodeks pracy. Dodatkowo, w części dotyczącej spraw socjalno-bytowych, układ nie może być sprzeczny z opracowywanym obecnie programem socjalnym NSZZ „Solidarność". Oceniamy zatem, że Układ Zbiorowy Pracy dla Pracowników Informatyki może zostać podpisany dopiero w końcu trzeciego lub na początku czwartego kwartału br.

Włodzimierz MROZEK

Głos NSZZPI

Układ Zbiorowy powinien być podstawą ochrony interesów pracownika, nie może to być tylko zbiór przepisów finansowych. Układ musi zagwarantować ochronę zdrowia oraz uznanie efektów wieloletniej pracy, musi regulować specyficzne warunki pracy informatyków.

Zjednoczenie Informatyki zaproponowało w końcu grudnia ub.r. pierwszą wersję projektu. Po konsultacjach ze Związkami Zawodowymi opracowano dwie kolejne wersje. Uważamy, że nadszedł czas na następną turę konsultacji. Tym razem — wraz z ośrodkami spoza sieci ZETO, w których brakowało dotąd informacji o pracach nad Układem Zbiorowym dla Pracowników Informatyki.

Do rozstrzygnięcia są obecnie następujące kwestie:

- **jakich istotnych problemów nie ujmuje projekt Układu?**
- **czy Układ powinien zawierać Kartę Informatyka?**
- **jak doprowadzić do objęcia Układem wszystkich informatyków, bez uszczuplenia ich praw resortowych?**

Uważamy, że Układ powinien umożliwić przyjęcie go poza siecią ZETO (czyli — musi zapewnić korzystanie z resortowych uprawnień, musi określić status pracownika informatyki oraz informatyka, wreszcie — musi regulować specyficzne warunki pracy informatyków (np. wprowadzić rozliczanie programistów z wykonanej pracy, a nie z godzin spędzonych przy biurku).

Nie rozstrzygniętym problemem są dodatki. Czy za znajomość języków obcych płacić tym, którzy systematycznie używają języka obcego (czego wymaga ich stanowisko pracy), czy też wszystkim, którzy zdali egzamin państwowy?

Sądzimy i będziemy bronić tego zdania, że niezmiernie istotnym problemem jest wynegocjowanie zespołu prewencyjnych norm bhp i ochrony pracy kobiet. Niezrozumiałe jest uparte stanowisko administracji, że 85 dB hałasu w salach komputerowych jest właściwe i przed zakończeniem badań nie można tej normy (wziętej z sufitu) zmienić. Trudno też zrozumieć stwierdzenie, że praca przed monitorem ekranowym jest nieszkodliwa, gdyż w Polsce nie prowadzono badań. Możliwe, że kilka μrtg/sek. czyli 0,1—0,5 rtg miesięcznie jest nieszkodliwe dla kobiety ciężarnej i przyszłego dziecka, ale w końcu po co ryzykować? Lepiej uchylić zbyt ostre przepisy, niż obciążać swoje sumienie.

Proponujemy więc ostrzejsze normy dla stanowisk pracy, dopuszczając pewne opóźnienia przy ich wprowadzaniu w życie — tak, aby administracja i przemysł mogły się do nich dostosować. Podstawowa zasada musi być zaś skracanie dziennego czasu pracy w warunkach uciążliwych. Głuchemu pieniądze słuchu nie przywrócą.

Przypominamy, iż zgodnie z informacją w numerze lutowym INFORMATYKI z pełnym tekstem projektu Układu można się zapoznać w sekretariacie NSZZPI, Warszawa, Al. Niepodległości 190 p. 110.

Wojciech MADEJSKI

MIKROELEKTRONIKA – rewolucja nie dokonana

Wizja zapowiadanej rewolucji mikroelektronicznej została najwyraźniej sztucznie wyolbrzymiona. Prawda, lata osiemdziesiąte będą w krajach rozwiniętych gospodarczo złotym okresem elektroniki, niemniej rozpowszechnienie mikroukładów nie osiągnie spodziewanego poziomu, a ich wpływ na dalsze życie będzie mieć bardziej charakter ewolucji niż rewolucji.

Czegóż to nie obiecywano sobie po mikroelektronice na najbliższe lata: telekopiarki, pracujące na łączach telefonicznych (znakomicie uzupełniające telefony i wyręczające tym samym pocztę), miały być nieomal w każdym domu; końcówki komputerowe miały dawać każdemu momentalny dostęp do powszechnych zbiorów informacji i bibliotek; ucząca się dziatwa miała chłonąć wiedzę u siebie w mieszkaniach, zamiast się tłoczyć w szkolnych klasach; panie domu miały zamawiać dostawy produktów spożywczych na drodze teleelektronicznej i sterować automatycznymi kuchniami na odległość, przyciskając odpowiednie guziki w samochodach. A wszystkim tym i wielu, wielu innym dobrodziejstwom miało towarzyszyć uwolnienie milionów fabrycznych robotników od trudu pracy, którą miały za nich wykonywać wydajne i zdyscyplinowane roboty.

Dowodzono przy tym, że nie ma ani technicznych, ani ekonomicznych przeszkód do urzeczywistnienia tej wspaniałej wizji w ciągu najbliższych dziesięciu lat, a to w oparciu o cały przyszły i ekstrapolowany na przyszłość rozwój techniki półprzewodnikowej, w szczególności zaś mikroelektroniki i najcudowniejszego jej wynalazku — mikroprocesora. Postępująca miniaturyzacja sprzętu elektronicznego i nieustanna do tej pory obniżka jego ceny, średnio o 28% rocznie, dały asumpt entuzjastom i popularyzatorom postępu techniki do głoszenia nastania ery „swobodnej" inteligencji, kiedy to prawie każde urządzenie techniczne — uzbrojone bardzo małym kosztem w mikroprocesor — zyska cechy rozumnej istoty.

Rzeczywistość okazuje się jednak inna. Po pierwsze — nie nastąpił zapowiadany zalew nie tylko nowymi, ale nawet starymi typami mikroukładów elektronicznych. Opracowanie ich konstrukcji i technologii, pomimo wykorzystania komputerów, wymaga czasu. Szkolenie fachowców, którzy mają zdobyć jedną z najtrudniejszych umiejętności: budowania (z mikroukładów) systemów dla konkretnych zastosowań praktycznych też wymaga czasu. Do tego dochodzi nie mniej ważny czynnik nakładów kapitałowych, niezbędnych na wszystkich szczeblach rozwoju informatyki, a specjalnie — na nowe, masowe inwestycje w przemyśle elektronicznym. Po drugie — nie wszystko co jest technicznie możliwe musi być zaraz urzeczywistniane w praktyce. Ludzie po prostu nie akceptują, a przynajmniej nie od razu, wielu z proponowanych im zmian, i to tak z pozycji użytkownika, jak twórcy nowej techniki. Konserwatyzm myślenia i działania jest wielką siłą społeczną. Wprawdzie apokaliptyczna perspektywa powszechnej klęski bezrobocia, spowodowanego informatyką, okazała się generalnie fałszywa, jednak obawy pozostały.

A wpływ informatyki na przeobrażenie naszego życia codziennego? I tu sprawę wyraźnie przejaskrawiono. Bardziej trzeźwe analizy nie potwierdzają, aby komputeryzacja miała odegrać choćby w przybliżeniu taką rolę, jaka przypadła w udziale telewizji. Jeszcze parę lat temu przeciętny mieszkaniec Europy pozostawał w błogiej niewiedzy co do znaczenia i konsekwencji wynalazku mikroprocesora. Potem wprawdzie zaczął mu przypisywać cechy niemal diaboliczne, dziś jednak wahadło biegnie z powrotem w drugą stronę — być może znów za daleko.

Wydaje się, że przy całej obfitości bardzo ostatnio modnych rozważań na temat przyszłości informatyki, niektóre elementy sytuacji, rzutujące na jej rozwój w najbliższych latach — zwyczajnie przeoczono. Nie doceniono mianowicie, że na drodze wielkiej, tzw. drugiej rewolucji technicznej, pojawiły się hamulce mające swe źródła w przemyśle, i to w przemyśle elektronicznym, a więc tym, który właśnie tej rewolucji miał przewodzić.

Przemysł półprzewodników powstał przy końcu lat pięćdziesiętych w okręgu Santa Clara w Kalifornii, nazwanym później „krzemowym zagłębiem", i do chwili obecnej jest zdominowany przez Stany Zjednoczone. Trzeba przyznać, że tempo jego rozwoju było naprawdę imponujące. Ale jednocześnie, zgodnie z prawami dialektyki, osiągnął poziom, kiedy jego wielkość (ok. 10 mld $ wartość rocznej produkcji) i coraz bardziej skomplikowana technologia stały się czynnikami hamującymi jego dalszy rozwój.

Począwszy od 1960 r. przemysł półprzewodników nieustannie zwiększał skalę integracji zespołów elektronicznych, średnio podwajając co roku liczbę elementów w pojedynczym układzie scalonym, niewiele zmieniając ich gabaryty — dzięki niewiarygodnej miniaturyzacji i jednocześnie obniżając ich ceny o ok. 28% rocznie. W 1963 r. gęstość „upchania" tych elementów w jednej „kostce" wynosiła 16, w 1967 r. — 256, natomiast w 1970 r. osiągnęła już ok. 4000. W 1976 r. potrafiono ich pomieścić ponad 60 tys., a poziom 1 miliona — uwzględniając pewne zwolnienie tempa wzrostu — ma być osiągnięty ok. 1990 r. Gdy w roku 1960, dla przykładu, cena pojedynczego tranzystora kształtowała się na poziomie 10$, to dziś — po 20 latach — kosztuje on w układzie scalonym poniżej 1 centa, czyli ponad 1000 razy mniej.

Równolegle z tym postępem rosły nakłady inwestycyjne. W końcu lat sześćdziesiątych można było zbudować kosztem 2 mln $ zakład produkcji podstawowych typów układów scalonych, spełniających ówczesne wymagania. Przy dzisiejszych wymaganiach na tego rodzaju zakład trzeba wyłożyć minimum 50 mln $. Nic dziwnego, że w 1970 r. odnotowano powstanie tylko 3 nowych firm tej branży. W sytuacji, gdy w wartości sprzedaży półprzewodników udział kosztów inwestycyjnych osiągnął 16%, a zyski producentów — po raz pierwszy od wielu lat — wykazały tendencję spadkową, kapitał finansowy mniej chętnie angażuje się w rozwój mikroelektroniki szukając sobie bardziej atrakcyjnych pól działania; ostatnio np. kieruje swe zainteresowania w stronę biotechniki. Tendencja ta ujawniła się już w 1978 i 1979 r., kiedy to na skutek nienadążania rozbudowy bazy produkcyjnej za popytem, 10 głównych amerykańskich wytwórców półprzewodników pozwoliło sobie narzucać swym odbiorcom odległe terminy odbioru zamówionego towaru, nierzadko dochodzące do 10 miesięcy. Koszty inwestycyjne w nowoczesnym przemyśle półprzewodników osiągnęły bowiem poziom w istotny sposób ograniczający przyrost nowych mocy produkcyjnych.

Jest rzeczą mało prawdopodobną, aby amerykański przemysł półprzewodników, zaspokajający ok. 2/3 światowego zapotrzebowania, mógł w najbliższych latach dotrzymać kroku popytowi. Niedobór układów scalonych bije głównie w wielkich klientów przemysłu półprzewodników, tzn. w firmy, które z tych układów budują gotowy produkt. Narażeni są oni na przestoje, muszą ograniczać, a nieraz i zatrzymywać produkcję niektórych wyrobów, odkładać wprowadzanie na rynek nowych modeli sprzętu i — ogólnie rzecz biorąc — nie wykorzystują swych mocy przerobowych. Prowadzi to do załamania planów produkcji, zysków i rozwoju.

W ostatnim roku, dwie z najpoważniejszych dziesięciu firm komputerowych: NCR i DATA GENERAL ogłosiły spadek obrotów i zysków na skutek niedostatku układów scalonych.

Oba te przedsiębiorstwa należą do największych stałych konsumentów półprzewodników. Jeżeli tej rangi odbiorcy nie zdołali sobie zapewnić dostaw, jeżeli potężny przemysł motoryzacyjny musi wypraszać półprzewodniki na swe potrzeby, łatwo sobie wyobrazić jaki jest los innych, słabszych i nie stałych klientów.

Tymczasem popyt na półprzewodniki, nie dość że już bardzo duży, ma tendencję dalszego wzrostu. Firma marketingowa DATAQUEST, operująca we wspomnianym już „zagłębiu krzemowym", doliczyła się, że liczba przedsiębiorstw kupujących układy scalone za więcej niż 100 mln $ rocznie, wzrosła od 1976 r. siedmiokrotnie, a do 1981 r. wyrazi się wzrostem 17-krotnym. Skąd się wziął ten zadziwiający popyt? Niewątpliwie ma w tym swój udział największe (w dwudziestoletnim okresie rozwoju elektroniki) osiągnięcie: wynalezienie w 1972 roku mikroprocesora. Najistotniejszy walor tego mikroukładu polega na tym, że daje się programować, tzn. że ciąg rozkazów, który wykonuje, nie musi być sztywny, dzięki czemu ten sam mikroprocesor nadaje się do wykorzystania przy rozwiązywaniu bardzo zróżnicowanych zadań, a więc jest to cecha właściwa komputerowi, czyniąca zeń element przydatny we wszelkich, zwłaszcza bardziej wysublimowanych układach automatyki.

Deficyt na rynku półprzewodników powinien — teoretycznie rzecz biorąc — prowadzić do angażowania się wolnych kapitałów, do tworzenia nowych i powiększania starych mocy w sferze produkcji półprzewodników w poszczególnych przedsiębiorstwach — na własne ich potrzeby. Nie jest to jednak takie proste. Poprzednio wskazaliśmy na koszty inwestycji jako czynnik hamujący rozwój tej branży. Żeby przyciągnąć kapitał, muszą wzrosnąć ceny półprzewodników. I rzeczywiście — era nieustającej obniżki cen mikroukładów ma się wyraźnie ku końcowi. Już w ubiegłym roku ceny niektórych ich typów wzrosły o 15—20%. Ale sama podwyżka cen nie rozwiązuje sprawy. Nie tylko dlatego, że występuje deficyt wolnych kapitałów, ale głównie dlatego, że równolegle występuje deficyt specjalistów w tej dziedzinie. Rosnącej złożoności i doskonałości układów scalonych towarzyszy nie tylko wzrost niezbędnych nakładów inwestycyjnych, ale także zwiększone zapotrzebowanie na wysokokwalifikowaną kadrę fachowców.

Dla przykładu — w ubiegłym roku producenci półprzewodników koncentrowali wysiłki nad zbudowaniem kolejnej generacji układów pamięciowych, oznaczony 64k RAM (Random Access Memory). Wymagało to pomieszczenia, praktycznie w tej samej objętości mikroukładu, czterokrotnie większej liczby obwodów, w porównaniu z układem 16k. Okazało się to zadaniem ogromnie trudnym. Firma INTEL wyliczyła, że na opracowanie nowego mikroprocesora oznaczonego w katalogu numerem 8086 wydała ogółem 200 mln $. Żeby opracować i wdrożyć do produkcji nowy typ mikroukładu, zwłaszcza mikroprocesora, trzeba zaprojektować wiele dziesiątków układów cząstkowych i przebadać tysiące możliwych kombinacji ich skojarzenia, trzeba wybrać z nich najlepsze i stworzyć ustrój spełniający różnorodne wymagania użytkowników. Daje to pojęcie nie tylko o kosztach, ale i o trudności tego rodzaju przedsięwzięć, trudności, którym mogą podołać tylko najwyższej rangi specjaliści. Tych zaś — brak.

Brak kadry specjalistów występuje nie tylko w sferze produkcji półprzewodników, ale i w sferze tworzenia z nich jednostek wyżej zorganizowanych oraz systemów bezpośrednio użytecznych, w sferze ich oprogramowania i obsługi. Tego niedoboru nie da się zlikwidować z miesiąca na miesiąc. Trzeba lat nauki w zakresie wiedzy podstawowej i dalszych lat nieprzerwanego doskonalenia praktycznego. Jeżeli więc chodzi o „drugą" rewolucję techniczną, „nowy" styl życia, erę „swobodnej" inteligencji itp. istnieje wiele przesłanek, aby traktować je w kategorii zjawisk łagodniejszych i nieco bardziej odległych, niż to głoszą katastrofiści oraz entuzjaści postępu.

Opracował A.R. na podstawie THE ECONOMIST z 1 marca 1980 r.

UNIVAC zwiększa produkcję półprzewodników

W ub.r. firma UNIVAC utworzyła wyodrębnioną organizację ds. produkcji półprzewodników, zwiększając jednocześnie dotychczasowy kapitał zakładowy swych zakładów wytwórczych w dziedzinie układów VLSI o 50 mln dolarów. Oprócz tego przy nowym zakładzie w Eagan (stan Minnesota) utworzono ośrodek naukowo-badawczy, zatrudniający ponad 300 pracowników.

Dzięki tym posunięciom organizacyjnym oraz inwestycjom firma UNIVAC znacznie wzmocni swą dotychczasową pozycję w obszarze działalności decydującym o rozwoju sprzętu komputerowego. Szczególnie duże nadzieje wiąże firma z efektami ekonomicznymi wynikającymi z własnych opracowań technologicznych.

Nowa organizacja położy główne akcenty na przyspieszenie rozwoju i unowocześnienie technologii wytwarzania oraz na rozwinięcie produkcji elementów zaprojektowanych z uwzględnieniem indywidualnych potrzeb użytkownika.

Oprócz działalności naukowo-badawczej i produkcyjnej organizacja ta jest odpowiedzialna za zaopatrzenie wszystkich zakładów produkcyjnych UNIVAC — w pełny asortyment części i elementów handlowych. Wzrost zakresu i rozmiarów własnej produkcji półprzewodników nie wpłynie jednak w zasadniczy sposób na dotychczasową pozycję firmy jako jednego z największych na świecie odbiorców elementów elektronicznych.

Oprac. W. K. na podstawie biuletynu informacyjnego firmy SPERRY UNIVAC nr 5/80

Współpraca UNIVAC z ChRL

W połowie sierpnia ub.r. firma UNIVAC podpisała umowę kooperacyjną z Chińską Republiką Ludową. Umowa ta obejmuje organizowanie szkolenia informatycznego, a także stałą pomoc w realizacji tego rodzaju szkolenia w ramach dostaw sprzętu komputerowego. Ze strony chińskiej sygnatariuszami umowy jest zjednoczenie państwowego przemysłu komputerowego oraz centralna organizacja kooperacji międzynarodowej w dziedzinie inwestycji.

Umowa zapewni realizację programów szkolenia informatycznego oraz obsługi technicznej użytkowników sprzętu firmy UNIVAC na terenie Chin, jak również — możliwość samodzielnego opracowywania i adaptacji oprogramowania dla tego sprzętu.

Umowa wymaga jeszcze formalnej akceptacji odpowiednich organów administracji państwowej USA oraz ChRL.

Oprac. W. K. na podstawie biuletynu informacyjnego firmy SPERRY UNIVAC nr 5/80.

O realizacji komputerowych układów automatyki

Od czasu, gdy komputera użyto po raz pierwszy do sterowania procesem przemysłowym, minęło właśnie 20 lat. Jednak dopiero w ostatnim dziesięcioleciu sformułowano ogólne zasady stosowania komputerów do automatyzacji procesów produkcyjnych. Zagadnieniom tym poświęcona jest ksią ka Henryka Orłowskiego — „Komputerowe układy automatyki" [1]). Autor omawia w niej środki realizacji komputerowych układów sterowania w trybie bezpośrednim, a więc zagadnienia praktyczne, pomijając metody matematyczne automatyki (teorię sterowania), które stanowią oddzielną grupę problemów i mają już bogatą literaturę w języku polskim.

Po obszernym wprowadzeniu (w którym znalazło się m.in.uzasadnienie celowości stosowania komputerów do sterowania) i gruntownym przedstawieniu zasad sprzęgania komputerów z obiektami instalacji przemysłowych, autor omówił kolejno sprzęt komputerowy oraz urządzenia sprzęgające poszczególne elementy systemu, jakimi są obiekt, komputer i operator (poświęcając najwięcej uwagi przedstawieniu sprzęgu PI), by przejść następnie do zagadnień oprogramowania i układów rozłożonych przestrzennie. Książkę kończy omówienie niezawodności komputerowych układów automatyki oraz krótki opis przykładowych zastosowań.

Generalnie rzec można, że autor — zajmujący się od dość dawna zagadnieniami komputerowych układów automatyki — ujął całą problematykę w sposób bardzo nowoczesny. W książce, podobnie jak w praktyce, centralne miejsce zajmują znormalizowane układy sprzęgające komputer z obiektem sterowanym, co jest bardzo charakterystyczne dla nowoczesnego ujęcia.

Autor bardzo obszernie omówił podstawowe i najbardziej rozpowszechnione układy sprzęgające, poczynając od magistrali wewnętrznej komputera, jaką jest UNIBUS, przez właściwe sprzęgi objęte normami międzynarodowymi, jak system CAMAC i magistrala IEC-625 (a nie 488, jak podano w książce) oraz krajowy system PI, aż do typowego sprzęgu komunikacyjnego, objętego zaleceniem V.24 CCITT.

W ten sposób czytelnicy otrzymali po raz pierwszy pełny i w miarę aktualny opis możliwości układów sprzęgających, co z pewnością ułatwi im orientację i pomoże w konfigurowaniu zestawów. Należy jednak pamiętać, że wiadomości te trzeba ciągłę uaktualniać, gdyż wszystkie systemy międzynarodowe są nieustannie rozwijane i unowocześniane. Dotyczy to szczególnie systemu CAMAC i norm serii V (równoważnym normom amerykańskim EIA RS — Electronic Industries Association Recommended Standards).

Wydaje mi się, że w tej części książki warto by wyraźniej określić miejsce poszczególnych sprzęgów w systemie komputerowym jako całości, gdyż nie są one wzajemnie niezależne. Nie chodzi tu o porównanie (to nie miałoby sensu), lecz ustrukturalizowanie ich w taki sposób, aby czytelnik wiedział, co do czego używać. Choć wszystkie wymienione sprzęgi mogą współpracować w jednym systemie komputerowym, to w zasadzie każdy będzie spełniać inną funkcję.

Przechodząc do przedstawienia zasadniczych cech sprzętu komputerowego stosowanego do sterowania, autor przyjął sygnalizowaną wcześniej zasadę (która nie ma chyba konkurencji), aby ilustrować poszczególne koncepcje przykładami konkretnych urządzeń, które są produkowane i stosowane w praktyce. Omówienie cech komputerów do sterowania na przykładzie jednostek PDP-11 oraz INTEL 8080 jest zabiegiem bardziej szczęśliwym, niż wprowadzenie jeszcze jednego komputera „przykładowego". Warto jedynie dodać, że w dziedzinie systemów mikroprocesorowych koncepcja sprzęgów standardowych ma także wielu zwolenników i pierwsze prace normalizacyjne są bliskie zakończenia. Natomiast rozdział o urządzeniach peryferyjnych można by z powodzeniem umieścić w dodatku, jako informację o stanie produkcji krajowej.

Szczegółowe rozważania dotyczące układów sprzęgających mają bardzo praktyczny charakter. Choć podstawowym elementem sprzężenia obiektu z komputerem jest modularny system cyfrowy, omówiony tu na przykładzie sprzęgu PI, to autor poświęcił także wiele uwagi wyposażeniu obiektu w źródła i odbiorniki sygnałów, układom sprzęgającym człowiek—maszyna (odgrywającym coraz większą rolę w nowoczesnych systemach sterowania), a także — właściwościom samego toru pomiarowego.

W tej części książki znalazła najpełniejszy wyraz deklaracja autora zaprezentowania przykładów poszczególnych rozwiązań. Jednak nie wszystkie możliwości zostały tu wyczerpane. Praca zyskałaby na spójności, a czytelnik — na pełności opisu, gdyby wraz z dokładnym omówieniem modułów sprzęgu PI podano konsekwentnie szczegóły architektury i organizacji któregoś z minikomputerów produkcji krajowej, lub gdyby — omówiwszy dokładnie minikomputer PDP-11 — przedstawiono realizację różnych układów sprzęgających dla tego komputera. Nikt tego bowiem dotąd w Polsce nie uczynił. Zademonstrowanie inżynierom w sposób zwarty całości systemu komputerowego, a jednocześnie uzmysłowienie im, jak bardzo można uniezależnić się od konkretnego komputera stosując sprzęg standardowy, wydaje mi się bardzo pożądane. Jednak trudno wymagać od autora spełnienia zbyt wielu warunków równocześnie.

Jednym z atutów książki jest z pewnością pełne przedstawienie zagadnień oprogramowania — tak, jak je widzi użytkownik, tzn. z wyraźnym dążeniem do uproszczeń w programowaniu, z czego wynikają także tendencje normalizacyjne. Pod tym kątem autor omówił zasadnicze metody programowania komputerów do sterowania, począwszy od języków wysokiego poziomu (na przykładzie języka FORTRAN), przez języki blankietowe i języki schematów blokowych, aż do pakietów programów, charakteryzujących się największą łatwością użycia; przedstawił też w skrócie rolę systemów operacyjnych. Fragment dotyczący języka FORTRAN zawiera bardzo ważne informacje o trzech normach rozszerzających FORTRAN standardowy do celów programowania systemów czasu rzeczywistego. Wypada jednak dodać, że propozycje do czwartej normy, najważniejszej, jako że dotyczącej zarządzania zadaniami, są już nieaktualne i spośród podanych nazw podprogramów zaledwie trzy wchodzą w skład obecnie zalecanych (European Workshop on Industrial Computer Systems, Technical Committee 1, January 1980).

Metoda ilustrowania wykładu przykładami wziętymi z praktyki zaowocowała tu omówieniem systemu blankietowego SZPAK dla komputera ODRA 1325 i systemu operacyjnego PSOT dla zestawu MERA-300/PI. Również przy omawianiu układów wielokomputerowych podano przykład realizacji systemu o rozłożonej inteligencji, składającego się z minikomputerów MERA i sprzęgu PI. Należy dodać, że ostatnio zagadnienia sieci komputerowych nabierają coraz większego znaczenia. Każdy z kilku poziomów organizacji sieci jest przedmiotem odrębnych prac normalizacyjnych, co prawdopodobnie nie było jeszcze widoczne w czasie powstawania książki.

W wymiarze ogólnym treść książki stanowi odpowiedź na pytanie, jak realizować systemy komputerowe do sterowania, a nie — jak je eksploatować, choć w wielu

[1]) Henryk Orłowski: Komputerowe układy automatyki. Wydawnictwa Naukowo-Techniczne, Warszawa, 1980. Wyd. I, nakł. 5000 + 260 egz., str. 600, cena 152 zł.

miejscach autor daje po temu wyraźne wskazówki. Jednak dopiero w końcowych rozdziałach, które dotyczą niezawodności i zawierają przykłady zastosowań, przedstawiono w sposób systematyczny doświadczenia z eksploatacji systemów pod względem niezawodności, nie pomijając także zagadnień niezawodności oprogramowania. Jest to cenna zaleta książki, gdyż sprawy te — w istocie podstawowe — są w monografiach prawie zawsze pomijane. Można je wyraźniej podkreślić już w rozdziałach wstępnych. Przykładowo, wyodrębniając poszczególne etapy powstawania komputerowego układu automatyki, tj. projektowanie, kompletowanie i zainstalowanie zestawu (rys. 1.2), warto byłoby zwrócić uwagę na konserwację sprzętu, co ma zasadniczy wpływ na utrzymanie bezawaryjnej pracy (o tym zaś często się u nas zapomina).

W całej książce wiele jest stwierdzeń będących wynikiem osobistych doświadczeń i przemyśleń autora, sprawdzonych przy tym w praktyce. Powiększa to znacznie wartość pracy i wzmacnia zaufanie czytelnika do przedstawionych rezultatów.

Pod względem terminologii książka napisana jest z dużą starannością, choć z uwagi na brak w tej dziedzinie jednoznacznych ustaleń łatwo o błędy. Osobiście boleję tylko nad jednym. Użycie niepoprawnego określenia **interfejs** na oznaczenie angielskiego *interface* znów odsunie o jakiś czas wprowadzenie terminu **sprzęg (układ sprzęgający)**. Choć uważam, że ze względów językowych używanie wyrazu **interfejs** jest niedopuszczalne, autor jest formalnie w porządku, bo słowo to figuruje — o zgrozo — jako termin w Polskiej Normie.

W chwili obecnej książka spełnia wymagania kompletności opisu, nowoczesności ujęcia i poziomu przedstawienia opisywanych zagadnień. Choć z natury rzeczy rozwiązania szczegółowe będą się zmieniać, to zaprezentowane koncepcje pozostaną aktualne, nawet w przypadku dających się dziś przewidzieć zmian technologicznych. Dlatego warto polecić tę pracę, jako lekturę nieodzowną, wszystkim, którzy wprowadzają komputery do automatyzacji sterowania procesami technologicznymi oraz eksperymentami badawczymi.

Jeżeli umiejętność stosowania komputerów w automatyce nie idzie w parze z możliwościami (a jest to po części wina braku odpowiedniego przygotowania potencjalnych użytkowników), to książka Henryka Orłowskiego daje szansę uzupełnienia lub wręcz nabycia odpowiedniej wiedzy. Najlepsza jednak książka niczego nie zmieni, dopóki nie będzie można zdobyć odpowiedniego sprzętu.

Janusz ZALEWSKI

Jak koordynować ?

„Sprawa informatyki", artykuł Zbigniewa Gluzy (INFORMATYKA nr 9/80) porusza bardzo istotne zagadnienia dotyczące rozwoju polskiej informatyki, a jednocześnie skłania do pewnych refleksji natury ogólnej.

Wielokrotnie już podkreślano rolę informatyki w gospodarce narodowej, wskazywano na rosnącą liczbę zastosowań, coraz liczniejszą i lepiej wykwalifikowaną kadrę informatyków, nowocześniejsze rozwiązania sprzętowe. Oczywiste jest jednak, że obecny stan sprzętu informatycznego (m.in. malejąca z roku na rok produkcja komputerów krajowych, ograniczone możliwości wzbogacania konfiguracji, znaczna awaryjność niektórych typów urządzeń, a także permanentny niedobór materiałów eksploatacyjnych i fatalny poziom sieci łączności) hamuje rozwój branży.

Jedną z przyczyn niskiej efektywności dotychczasowych przedsięwzięć informatyki w kraju jest „brak skutecznej koordynacji jej rozwoju". Na pytanie kto zajmuje się koordynacją informatyki i jej rozwojem nie można dać dzisiaj jednoznacznej odpowiedzi. Chyba, że powiemy „wszyscy, czyli nikt". Potwierdzeniem tego jest fakt, że z 1523 ośrodków informatycznych w 1979 r. — 486 (31,9%) wykazało, że w ramach swojej działalności zajmuje się koordynacją z zakresu informatyki. Stan taki budzi zaniepokojenie i uzasadnia potrzebę scentralizowania działań koordynacyjnych.

Wymóg racjonalnego wykorzystania zainstalowanego sprzętu komputerowego, a także tendencja zmniejszania nakładów na prace informatyczne (głównie typu inwestycyjnego) zmuszają do rzeczywistej koordynacji działań między poszczególnymi uczestnikami procesu informatyzowania kraju, to jest producentami i dostawcami sprzętu, autorami oprogramowania oraz użytkownikami, niezależnie od miejsca zajmowanego przez nich w tym procesie. Eksploatowane systemy informatyczne powinny stanowić element wiążący działalność różnych jednostek w obszarze wymiany informacji. W rzeczywistości jednak większość jednostek organizacyjnych wdraża systemy na własną rękę, rzadko spójne z systemami centralnymi.

Trzeba przyznać, że polska informatyka nie miała szczęścia do zmian organizacyjnych, w wyniku których zarówno Biuro Pełnomocnika Rządu ds. Elektronicznej Techniki Obliczeniowej, Krajowe Biuro Informatyki i ostatnio Komitet Informatyki nie spełniły w pełni swoich zadań koordynacyjnych. Powołanie odpowiedniego organu i postawienie przed nim zadań nie rozwiązuje problemu. Brakuje bowiem nadal skutecznych metod i bodźców ekonomicznych pozwalających na efektywne korzystanie z informatyki w poszczególnych dziedzinach zastosowań. Realizowana na szczeblu centralnym koordynacja dokonywana jest z dużą dozą przypadkowości, co powoduje stopniowe i niestety skuteczne zmniejszanie roli głównego koordynatora, jakim powinien być Komitet Informatyki.

W zakresie funkcjonowania i rozwoju informatyki wskazane są obecnie — moim zdaniem — następujące działania:

- *rozwijanie, w większym niż dotychczas stopniu, oprogramowania typu narzędziowego*
- *wprowadzenie ujednoliconej dokumentacji systemów informatycznych, terminologii, odpowiednich klasyfikacji, itp.*
- *rozwijanie konsultacji z przyszłymi użytkownikami na etapie podejmowania decyzji o wdrożeniu systemu informatycznego*
- *wprowadzenie określonych mechanizmów oceniających jakość tworzonego oprogramowania*
- *zwiększenie wykorzystania gotowych i sprawdzonych („powtarzalnych") systemów informatycznych*
- *wprowadzenie obiektywnej oceny dotyczącej przygotowania użytkownika do wdrożenia systemów informatycznych oraz stopnia wykorzystania zainstalowanego sprzętu.*

Osobne zagadnienie stanowi problem dostaw sprzętu informatycznego oraz części zamiennych. I w tym zakresie niezbędne są działania o charakterze koordynacyjnym, pozwalające na bardziej efektywne wykorzystanie obecnego potencjału.

Elżbieta KIERCZUK

Sugestie praktyka

„Czym jest informatyka w polskich realiach, a czym mogłaby być, gdyby nie...?", „Co trzeba by zmienić, co naprawić...?" oraz „czy uda się ożywić wymianę myśli na łamach INFORMATYKI?" — te trzy pytania, postawione w numerze 11/80, które uzupełniłbym zawartym w tym samym numerze przekonaniem Redakcji o trudnościach w wymianie poglądów pomiędzy informatykami w kraju, niezrozumieniem informatyki w społeczeństwie i o wynikającymi z tego stratami gospodarczymi — już dają pewien obraz sytuacji. Sytuacji niedobrej, gdyż w istocie: informatyka nie spełnia oczekiwań społeczeństwa (które zresztą niewiele o niej wie). Wewnątrz samej informatyki nie ma zaś dostatecznej współpracy pomiędzy teorią a praktyką, które izolują się od siebie i zamykają we własnych sprawach.

Chciałbym ustosunkować się do tych i innych negatywnych zjawisk w naszej współczesnej informatyce — z pozycji praktyka (projektanta systemów przetwarzania danych) z 17-letnim stażem w ośrodku obliczeniowym wielkiego przedsiębiorstwa przemysłowego — Huta im. Lenina). Pragnąłbym także zaproponować pewne kierunki działań, które — jak mi się zdaje — mogłyby wpłynąć na poprawę sytuacji przez wyeliminowanie lub osłabienie zjawisk negatywnych.

Jakie to zjawiska? Sądzę, że są to przede wszystkim:

- dysproporcja pomiędzy nowoczesnością rozwiązań konstrukcyjnych i możliwościami komputerów a zacofaną organizacją pracy przedsiębiorstw i instytucji
- fatalna informacja o informatyce, zwłaszcza o systemach przetwarzania danych
- błędy w szkoleniu informatyków, głównie — projektantów systemów
- brak dobrej łączności pomiędzy teorią i praktyką
- nieuporządkowana terminologia informatyczna.

Można oczywiście powiedzieć, że to wszystko jest uzasadnione, że informatyka jest nauką młodą, dopiero 40-letnią — nie może się więc równać z tysiącami lat „stażu" matematyki czy historii. Trudno się jednak przy tym zgodzić z olbrzymim marnotrawstwem, jakie powstaje wszędzie tam, gdzie toleruje się wymienione tu skrzywienia w informatyce, powodujące błędne decyzje w sferach: organizacji ośrodków, inwestycji, szkolenia kadr, wydawnictw itd. Temu trzeba przeciwdziałać.

Dysproporcje pomiędzy nowoczesnością i możliwościami dobrych komputerów a zacofaną w wielu miejscach organizacją pracy — jest widoczna zwłaszcza tam, gdzie uwierzono w mit, że kupiony za miliony komputer sam wymusi dobrą organizację produkcji, sprzedaży, gospodarki materiałowej czy kadrowo-płacowej. Znana jest bowiem zasada: najpierw organizacja potem komputeryzacja, nigdy odwrotnie. Wszędzie tam, gdzie nie przygotowano najpierw systemów informacyjnych (porządkujących i upraszczających obieg informacji), gdzie nadal nikt nie wie, że organizacja nie jest darem natury, lecz wiedzą, którą trzeba posiąść, a która składa się z setek elementów (nie zaś z trzech czy czterech, o których się stale mówi) i że organizacja nie ma nic wspólnego z improwizacją (jak się to przeważnie mniema) — otóż wszędzie tam komputery wprowadzone w chaos organizacyjny, chaos ten powiększają. Te zaś, które stoją od lat bezczynnie — także służą w tym przypadku za dowód.

O organizacji informatyki w przedsiębiorstwie, instytucji czy w państwie decydują nie ilość i jakość sprzętu komputerowego, lecz na pierwszym miejscu — ludzie. Byłem nieraz naocznym obserwatorem załamywania się projektów informatycznych (po wielu miesiącach pracy), a także innych przedsięwzięć poza informatyką — podejmowanych na zebraniach i naradach — tylko dlatego, że błędnie ustalono cel, powierzono sprawę niewłaściwym ludziom czy wreszcie — oparto je na błędnych informacjach.

Myślę, że informatycy (a rzecz jasna — nie tylko oni) muszą lepiej poznać podstawy organizacji pracy i to niekoniecznie z dużych opracowań, ale przede wszystkim — z niewielkich broszurek. Dodajmy, że w krajach rozwiniętych gospodarczo, np. w Wielkiej Brytanii, wymaga się od informatyków kursu ogólnej organizacji pracy (100 godzin), zakończonego egzaminem.

Fatalna informacja o sprawach informatyki to druga, bolesna ułomność. Liczne prace na temat informatyki są niespójne i bardzo nieczytelne — nawet dla informatyka, a już całkiem nieprzydatne dla szerszego ogółu. Powinny być one pisane językiem komunikatywnym, zwłaszcza zaś — publikacje popularne. Istotnym też błędem w publikacjach o informatyce jest personifikowanie komputerów. Jest to bezsensowna i szkodliwa maniera. Chodzi o takie stwierdzenia, jak: komputer powiedział, komputer nie jest zainteresowany, zobaczymy co komputer o tym sądzi itp. Niegdyś przeczytałem, że we wrocławskim DOLMEDZIE — komputer potrafi odróżniać nowotwór wątroby od innych jej schorzeń. Ależ nic podobnego! Komputer nie mówi, nie sądzi, nie zastanawia się i niczego nie potrafi zrobić samodzielnie. Jest on wprawdzie wysokozorganizowanym, lecz zupełnie bezmyślnym narzędziem człowieka. Odmienne przeświadczenie, popierane bezustannie w publikacjach, wytwarza wśród szerszego ogółu szkodliwe przekonanie o wszechmocy komputerów.

Co można zrobić w tym zakresie? Informować o komputerach rzetelnie. Nie unikać prawdy o 2—3-letnim i dłuższym okresie przygotowywania systemów oraz o ręcznym przygotowaniu każdej wprowadzanej cyfry lub litery, a przede wszystkim o tym, że komputer podaje tylko te informacje (przetworzone), które przedtem zostały doń wprowadzone. Konieczne jest również wydanie popularnej broszurki, zawierającej podstawowe pojęcia, zarys historyczny, przegląd sytuacji w informatyce krajowej i światowej, omówienie sprzętu, warunków organizacyjnych funkcjonowania informatyki w przedsiębiorstwie oraz opisy zastosowań w zarządzaniu, sterowaniu produkcją oraz w obliczeniach naukowo-technicznych. W jej przygotowaniu muszą wziąć udział również praktycy, którzy nasycą jej treść realiami gospodarki.

Błędy w szkoleniu informatyków dotyczą zwłaszcza szkolenia projektantów, których zadaniem jest również opracowanie organizacji nowego obiegu informacji u użytkownika. Szkolenie takie ma u nas charakter marginesowy. W szkołach różnych szczebli uczy się głównie programistów, natomiast projektanci szkoleni są albo przez zagranicznych producentów sprzętu (nieliczni), albo na sporadycznie organizowanych kursach krajowych; najczęściej jednak uczą się dopiero w miejscu pracy od starszych kolegów, co bynajmniej nie jest wystarczające.

Należałoby zatem opracować ujednolicone programy nauczania w zakresie projektowania systemów (ze specjalizacją „projektant systemów" w szkołach) i odrębnie — programy dla kursów zawodowych. Dla porównania podaję, że wg norm brytyjskich NCC projektant systemów musi szkolić się ok. 700 godzin (w tym 100 godzin problematyki organizacyjnej) oraz zaliczyć od jednego do dwóch lat praktyki.

Brak łączności między teorią i praktyką uwidocznia się m.in. w separacji teoretyków od ośrodków informatycznych, pracujących na rzecz gospodarki i odwrotnie — w nader rzadkim korzystaniu przez praktyków z dorobku teoretyków. Widać to także w publicystyce. Teoretycy często piszą o bazach danych, bardzo rzadko natomiast wypowiadają się na ten temat praktycy. A właśnie taka konfrontacja poglądów byłaby ciekawa i pożyteczna. Za mało jest praktyków w szkołach i na łamach czasopism! Teoretycy powinni wzbogacać swe wiadomości przez częstszy kontakt z ośrodkami, a praktycy przez dyskusje z przedstawicielami nauki. Ogół zaś, dzięki tej współpracy, uzyska lepsze publikacje i poprawę efektów szkolenia.

Osobna i ostatnia sprawa to terminologia stosowana w informatyce. Nie ma jednolitości w nazwach, a wszelkie działania w tym kierunku przyjmowano z dużymi oporami. Stąd też: **APD, EPD** i **API, zapis** i **rekord, plik**

i **zbiór, dane zmienne** i **transakcje, operacje** i **procedury** itd. Wśród kilkudziesięciu autorów książek trudno znaleźć choćby kilku, którzy podawaliby takie same definicje na określenie takich nawet terminów podstawowych, jak: **przetwarzanie danych, system informatyczny** czy **informatyka**, nie mówiąc już o **projektowaniu systemów**.

Źle jest gdy informatyk nie może zrozumieć tekstów pisanych przez niektórych autorów. Co znaczy np.: *„Wykorzystanie integrujących funkcji danej dziedziny przy budowie systemu ewidencyjno-informatycznego drogą zastosowania metody koncentracji (dedukcji) lub dekoncentracji (indukcji)"*. Nie wiem. U niektórych autorów **pojęcia bazy danych** i **banku danych** używane są zamiennie, u innych — — nie. Itp. Itd.

Trzeba chyba życzyć wytrwałości tym, którzy pracują nad unifikacją słownictwa informatycznego, ale także i tego, by zebrali z ośrodków wiadomości, jakich nazw tam używają. Język jest bowiem materią żywą i pewne nazwy utrwalają się, inne zanikają. Trzeba proponować te określenia, których używa większość.

Jerzy DROBISZEWSKI

Co dalej ze skomputeryzowaną rachunkowością?

W rubryce POGLĄDY opublikowany został artykuł prof. Tadeusza Peche „Rachunkowość nie doinformatyzowana" (INFORMATYKA nr 11/80). Uważam, że słusznie wyeksponowano w nim rachunkowość jako tę sferę działania przedsiębiorstw, dla której techniką komputerowa powinna świadczyć, a w nielicznych przypadkach — już świadczy, olbrzymie usługi.

Służba finansowo-księgowa naszego, nowego i dużego przedsiębiorstwa, jej bezpośrednie kierownictwo oraz władze zwierzchnie od początku, czyli — od połowy lat sześćdziesiątych, nie miały żadnych wątpliwości co do potrzeby zastosowania techniki komputerowej. Obecnie sytuacja jest taka, że rezultaty informatycznych wdrożeń w dziedzinie rachunkowości oraz w zakresie różnych prac ekonomicznych i administracyjnych są odczuwalne w skali całego przedsiębiorstwa. Służą one do różnych analiz, badania i kontroli wyników produkcyjnych, ekonomicznych i finansowych na szczeblach: przedsiębiorstwa i zjednoczenia.

Informatyzacją objęte zostały następujące zagadnienia:
- obliczanie wynagrodzeń pracowników
- gospodarka materiałowa i przedmioty nietrwałe
- gospodarka środkami trwałymi
- realizacja produkcji i zbytu wraz z bieżącym fakturowaniem sprzedaży i wystawianiem żądań zapłaty (listy inkasowe).

Wszystkie wyżej wymienione podsystemy mają charakter dziedzinowy, a jednocześnie w ramach ich eksploatacji uzyskuje się odpowiednie zbiory danych (rozdzielniki kosztów, zestawienia, specyfikacje), które są automatycznie wczytywane do podsystemu nadrzędnego FINKA (finanse-koszty). Podsystem ten realizuje: ewidencję dokumentów bieżących, transakcyjnych w przekroju kont syntetycznych i analitycznych, rozliczanie i kalkulację kosztów, analizę należności i zobowiązań pozainkasowych (łącznie z wystawianiem potwierdzeń sald na żądanie użytkownika) oraz szeroko pojętą analizę kosztów.

Można zatem śmiało stwierdzić, że jest to już prawie kompleksowy system przetwarzania danych w ramach rachunkowości oraz dziedzin z nią związanych, służący do zarządzania przedsiębiorstwem. A przecież prof. Peche wyraźnie stwierdził brak takiego systemu...

Emitowane tabulogramy zawierają — oprócz zwykłych danych ewidencyjnych — duży wachlarz informacji analitycznych. Ponadto sumy globalne (wyrażające ilość, wartość, wskaźniki itp.) przenoszone są z otrzymywanych tabulogramów na formularze zawierające miesięczne wyniki działalności przedsiębiorstwa. Odbiorcami tych informacji są wszyscy członkowie dyrekcji, główni specjaliści, kierownicy niektórych komórek funkcjonalnych, Komitet Zakładowy PZPR, Rada Zakładowa, Komisja Związkowa NSZZ „Solidarność" oraz Główny Księgowy i Główny Ekonomista w naszym Zjednoczeniu.

Wymienione podsystemy są powielarne oraz stale doskonalone w zakresie rozwiązań organizacyjnych i procedur przetwarzania. Nad tymi zagadnieniami stale czuwa zespół specjalistów, koordynujący eksploatację programów w jednostkach organizacyjnych zgrupowanych w Zjednoczeniu. W czym jak w czym, lecz w tej dziedzinie zasada centralnej koordynacji projektowania rozwoju informatyki, programowania systemów, inspirowania wdrożeń, etc. z d a ł a i z d a j e e g z a m i n. W związku z tym ani ja, ani inni zainteresowani postępem komputeryzacji rachunkowości przedsiębiorstw w naszej branży nie mamy żadnych wątpliwości, że to jest to, na co nasz kraj było dotychczas stać i sądzimy, że tego właśnie oczekiwano.

Pragnąłbym ponadto wypunktować te problemy, które — podobnie jak wielu praktyków — widzę w „czarnych kolorach". A mianowicie:

- roztrwoniono wiele pieniędzy i sił społecznych na realizowanie partykularnych oprogramowań, różnych pseudo-systemów i — podsystemów z zakresu rachunkowości, określając te przedsięwzięcia nierzadko nazwą pojedynczych przedsiębiorstw, branż czy zjednoczeń; ma to swoje potwierdzenie w tym, że istnieje ponoć w Polsce z górą 400 różnych oprogramowań (dumnie zwanych „pakietami programów") przeznaczonych dla ewidencji obrotu materiałowego
- nadano nieuzasadniony priorytet licznym przedsiębiorstwom (np. z resortu przemysłu maszynowego czy elektrotechnicznego) do wyposażenia zakładowych ośrodków informatyki w kosztowny sprzęt komputerowy, uszczuplając w ten sposób możliwości komputeryzacji w innych resortach lub zjednoczeniach
- ośrodki informatyczne (nie dotyczy to regionalnych ośrodków ZETO) wyposażono w różnorodny sprzęt komputerowy i peryferyjny, często bez zapewnienia odpowiedniego zaplecza konserwacyjno-naprawczego oraz zapasów części zamiennych
- istniejące od kilku lat ograniczenia dewizowe na import z drugiego obszaru płatniczego uniemożliwiają nie tylko powiększenie potencjału sprzętowego tej samej marki i klasy, ale także wymianę zużytego sprzętu na nowy
- szczególnie negatywnym zjawiskiem jest dokuczliwy brak papieru lub jego właściwego asortymentu do drukowania wyników przetwarzania
- wyżej wymienione czynniki wpłynęły hamująco na rozwój wdrażania wspomnianego efektywnego i wypróbowanego podsystemu FINKA w naszej branży.

Wyrażam osobiście nadzieję, że do kryzysu na dużą skalę w dziedzinie skomputeryzowanej rachunkowości dojść nie może. To co zostało dotychczas zrealizowane w naszym kraju winno przekonać decydentów, że skomputeryzowana rachunkowość w wielu jednostkach organizacyjnych „zarobiła" już z naddatkiem na siebie i że trzeba dać jej szansę kontynuacji oraz dalszego rozwoju.

Zdzisław CHĄDZYŃSKI
Główny Księgowy
Zakładów Azotowych
„PUŁAWY"

„Adresowanie"

Niektóre terminy związane z adresowaniem omówiliśmy w numerze 1/81. Ponieważ istnieją różne poglądy na tematy dotyczące ich używania, nie należy uważać, że sprawa jest już zamknięta. Przedmiotem sporu może być sam termin adresowanie. Musi on wzbudzać wątpliwości niektórych autorów, skoro próbują, i to w najnowszych książkach, używać innego — adresacja. Niestety, jest to błąd językowy.

Przypomnijmy, że termin adres pochodzi z języka potocznego, w którym samo określenie miejsca przeznaczenia przesyłki (określenie adresu) nazywa się adresowaniem. Nie ma więc powodu, aby w informatyce mówić adresacja. Jest to czynność, a czynność trudno wyrazić inaczej jak rzeczownikiem odsłownym, tym bardziej, że nie istnieje słowo angielskie „addressation". Tego rodzaju błędy językowe, np. rozpowszechnienie (także w informatyce) słowa kompletacja zamiast kompletowanie, są piętnowane przez językoznawców (por. W. Cienkowski: Język dla wszystkich; cz. II, Książka i Wiedza, Warszawa, 1980, str. 147). Wypada więc uznać używanie słowa adresacja za nieuzasadnione i traktować jako błąd językowy.

Znaczne, choć może nieuzasadnione wątpliwości budzi we mnie sposób definiowania poszczególnych trybów adresowania, spotykany w różnych słownikach polskich, a także w Polskich Normach, przejęty zresztą z norm ISO. Polega on na tym, że adresowanie A-te (np. pośrednie) określa się jako metodę adresowania za pomocą adresu A-tego (w tym przypadku — pośredniego). Tymczasem najważniejszą cechą adresu komórki jest jego jednoznaczność. Adres jest jeden, a więc różne jego postacie (nie tylko — końcowa, tzn. po modyfikacjach) nie powinny określać kilku różnych trybów adresowania.

Wydaje mi się, że w rzeczywistości jest odwrotnie, niż to wynika z definicji słownikowych. Nazwy metod adresowania pochodzą od sposobów wykonywania tej czynności, a nie od rodzajów adresów, natomiast nazwy adresów biorą się od trybów adresowania, ponieważ są względem nich wtórne.

Oczywiście, istnieją rodzaje adresów, które nie są związane z żadnym trybem adresowania. W tych przypadkach nazwy pochodzą od innych cech charakterystycznych. Przykładowo, adresem symbolicznym nazywa się adres wyrażony w symbolice konkretnego języka programowania (PN-71/T-01016). Nieco mniej banalna jest definicja adresu efektywnego (ang. *effective address*). Według normy polskiej adres efektywny jest to adres uzyskany przez przekształcenie adresu pierwotnego w sposób jednoznacznie określony dla danego komputera i języka programowania, i który — dodajmy za normą ISO-2382 — nie wymaga żadnej modyfikacji przed wykonywaniem rozkazu.

W rzeczywistości podane określenie dotyczy tej samej komórki pamięci, co terminy: adres bezpośredni, bezwzględny i rzeczywisty, a więc komórki, w której znajduje się argument. Wielość nazw jest związana z kontekstem, w jakim o tym argumencie się mówi. Zwróćmy uwagę, że takie terminy, jak adres pośredni i adres względny mają także rację bytu, ponieważ odnoszą się do komórek pośredniczących w określeniu argumentu, choć same adresy nie wystarczają do takiego określenia.

Z adresowaniem względnym związane jest pojęcie adresu podstawowego, inaczej — bazowego (ang. *base address*), którego określenie raczej nie powinno prowadzić do kontrowersji. Jest to liczba używana jako odniesienie przy obliczaniu adresu podczas wykonywania programu (norma polska i międzynarodowa).

Do obliczeń wykorzystuje się tu tzw. przemieszczenie (ang. *displacement, offset*), tj. liczbę, jaką należy dodać do adresu podstawowego, aby otrzymać — zmodyfikowany. Na ogół przemieszczenie dotyczy obliczania nie tylko adresów danych, lecz także adresów rozkazów. Bardzo popularnym rozkazem jest tzw. skok krótki, w którego słowie rozkazowym znajduje się informacja o odległości skoku. Jest to kilkubitowe przemieszczenie dodawane do adresu komórki zawierającej ten rozkaz, w celu wyznaczenia adresu rozkazu, do którego należy wykonać skok.

Zwróćmy uwagę, że w rzeczywistości następuje tu przemieszczenie wzdłuż obszaru zajmowanego przez program. Często zachodzi potrzeba przemieszczenia całego programu w określonym obszarze pamięci. Może tak być np. w wypadku wprowadzania programu do pamięci (ładowania) lub — łączenia modułów programu już załadowanego. Wtedy należałoby mówić raczej o relokacji (ang. *relocation*), utrzymując termin przemieszczenia na oznaczenie pojęcia omówionego powyżej.

Jednakże program poddający się zmianie położenia bez konieczności przeadresowania, tj. bez zmian adresów, a więc program niezależny od położenia (ang. *position independent*) można by nazywać przesuwnym, a niekoniecznie relokowalnym (ang. *relocatable*), mimo że termin przesunięcie jest zarezerwowany na oznaczenie odpowiednich rozkazów.

Wracając do adresowania warto podać regułę, wg której można jednoznacznie określić, jaki tryb jest podstawowy. Przy złożonych metodach adresowania za tryb podstawowy wypada uznać ten, który określa właściwy dostęp do danych lub rozkazów (do komórki zawierającej je). Przykładowo, gdy na adresowanie składają się dwa tryby: indeksowy i pośredni (por. INFORMATYKA, nr 1/1981), należy powiedzieć — indeksowe adresowanie pośrednie, jeżeli dane są zaadresowane pośrednio, natomiast — (pośrednie) adresowanie indeksowe, jeśli są zaadresowane przez indeksowanie. Zatem przy kilku trybach adresowania zastosowanych łącznie o nazwie podstawowej powinna decydować końcowa faza dostępu do adresowanej komórki.

Często zdarza się, że rozkazy nie zawierają żadnej informacji adresowej, tzn. ani adresu lub innego sposobu dostępu, ani argumentu, a mimo to następuje wykonanie operacji na danych. Ten rodzaj adresowania proponujemy określić mianem adresowanie niejawne, a nie — implikowane (ang. *implied addressing*), jak się spotyka. Przykładem, choć rzadko spotykanym, adresowania niejawnego jest tzw. adresowanie powtarzalne (ang. *repetitive addressing*), które polega na tym, że operacja jet wykonywana na argumentach poprzedniego rozkazu.

Na zakończenie podkreślmy wyraźnie, że choć słowo adres pochodzi z języka potocznego, to w informatyce adres jest na ogół samą liczbą. Stąd też często spotykane połączenie wyrazowe duży adres, większy adres, itd., należałoby uznać za poprawne, podobnie jak — adres parzysty i nieparzysty. Niekiedy spotyka się też określenie adresy górne i dolne, co wynika ze sposobu organizacji pamięci. Natomiast, nie ma uzasadnienia sformułowanie wysoki (niski) adres.

Janusz ZALEWSKI

Bibliografia wydawnictw polskich z dziedziny informatyki

● **Teoria szeregowania zadań — COFFMAN E. G., tłum. wyd. ang. z 1976 r., WNT, Warszawa 1980, s. 317, cena 65 zł**

Wprowadzenie do deterministycznej teorii szeregowania zadań. Algorytmy minimalizujące długość uszeregowania. Algorytmy szeregowania minimalizujące średni ważony czas przepływu. Złożoność obliczeniowa problemów szeregowania. Oszacowanie dokładności algorytmów szeregowania. Przeglądowe oraz iteracyjne metody obliczenia.
Książka przeznaczona jest dla informatyków i specjalistów badań operacyjnych.

● **Metody komputerowe w mechanice — SZMELTER J., PWN, Warszawa 1980, s. 293, cena 75 zł**

Elementarne wiadomości o technice komputerowej. Elementarny opis języka FORTRAN. Algebra macierzowa. Dyskretne liniowe układy mechaniczne. Wielomiany. Interpolacje funkcji. Pierwiastki algebraicznych równań nieliniowych. Całka oznaczona. Równania różniczkowe zwyczajne. Rozwiązywanie dużych układów równań liniowych. Metoda różnic skończonych. Metoda elementów skończonych. Opis podprogramów w języku FORTRAN 1900.
Książka przeznaczona jest dla inżynierów zajmujących się projektowaniem konstrukcji mechanicznych oraz studentów i pracowników naukowych wydziałów mechanicznych wyższych uczelni technicznych.

● **Komputery Jednolitego Systemu — JAGIELSKI R., PWE, Warszawa 1980, s. 361, cena 53 zł**
(Z serii „Informatyka w praktyce")

Architektura maszyn cyfrowych Jednolitego Systemu. Organizacja wejścia-wyjścia. Charakterystyki techniczne komputerów Jednolitego Systemu. Urządzenia zewnętrzne. Dyskowy System Operacyjny. Assembler PL/I. Organizacja i przetwarzanie kartotek. Aneks. Załączniki: lista rozkazów maszyn cyfrowych Jednolitego Systemu. Kody używane w Jednolitym Systemie.
Książka przeznaczona jest dla użytkowników maszyn cyfrowych Jednolitego Systemu, informatyków pragnących uzyskać podstawowe wiadomości o tego typu komputerach oraz studentów.

● **Modelowanie analogowe i cyfrowe — ZIELIŃSKI J. S., Wyd. Politechniki Łódzkiej, Łódź 1989, s. 427, cena 35 zł**

Cz. 1. Elementy teorii podobieństwa: Rozważania wstępne. Matematyczne podstawy podobieństwa. Twierdzenie o podobieństwie. Praktyczne kryteria podobieństwa różnych zjawisk elektromagnetycznych. Uwagi o dokładności modelowania i sposobach opracowania wyników badań.
Cz. 2. Modelowanie analogowe pól: Teoretyczne podstawy analogowego modelowania pól. Modele z przewodzących materiałów. Modele elektrolityczne i siatkowe. Siatki R. C. Nieelektryczne modele pól.
Cz. 3. Elektroniczne maszyny analogowe i hybrydowe: Ogólne wiadomości o maszynach analogowych. Zasada działania elementów operacyjnych EMA. Programowanie EMA. Wybrane zagadnienia z maszyn hybrydowych.
Cz. 4. Elektroniczne maszyny cyfrowe: Ogólne wiadomości o EMC. Uwagi o modelowaniu z wykorzystaniem EMC.
Skrypt stanowi pomoc do przedmiotu „Modelowanie analogowe i cyfrowe" wprowadzonego do nowego programu studiów wydziałów elektrycznych uczelni technicznych.

● **O metodologii projektowania systemów informatycznych — KOLBUSZ E., Wyd. Towarzystwa Naukowego Organizacji i Kierownictwa, Szczecin 1979, s. 175**

Systemy informatyczne jako przedmiot projektowania. Węzłowe problemy procesu projektowania. Podstawy metodologii projektowania. Metoda wzorca idealnego w projektowaniu ISZ. Metoda wzorca idealnego — zarys metodyki projektowania. Zarys projektowania strukturalnego.
Materiały przeznaczone są dla ekonomistów zajmujących się projektowaniem systemów informatycznych dla potrzeb zarządzania organizacjami gospodarczymi.

● **Komputerowe układy automatyki — ORŁOWSKI H., WNT Warszawa 1980, s. 660, cena 152 zł**

Struktury i przeznaczenia komputerowych układów automatyki. Pojęcia podstawowe. Interfejsy. Minikomputery. Mikroprocesory i mikrokomputery. Urządzenia zewnętrzne. Sygnały analogowe. Urządzenia obiektowe, operatorskie oraz sprzęgające z obiektem. Oprogramowanie. Układy wielokomputerowe i rozłożone przestrzennie. Niezawodność działania. Przykłady zastosowań. Tablice. Książka jest przeznaczona dla projektantów komputerowych układów automatyki oraz studentów wyższych szkół technicznych o specjalności automatyka przemysłowa.

● **Metody programowania nieliniowego — GOLIŃSKI J., Wyd. Zjednoczenia Informatyki — Centrum Projektowania i Zastosowań Informatyki, Warszawa 1980, s. 122, cena 82 zł**

Problemy informatyki
Pojęcia podstawowe. Metody poszukiwania ekstremum. Klasyfikacja metod optymalizacyjnych. Realizacja systemów optymalizacyjnych na EMC (SOPT, YSOP, CYBORG, CYNIUS).
Materiały przeznaczone są dla ekonomistów i inżynierów zajmujących się rozwiązywaniem różnych zadań technicznych i ekonomicznych oraz programistów.

● **Nowa rola systemu informacji kierownictwa w przedsiębiorstwie (tłum. wyd. ang. z 1978 r.) — Wyd. Zjednoczenia Informatyki — Centrum Projektowania i Zastosowań Informatyki, Warszawa 1980, s. 50 cena 126 zł**

Europejski Program Badawczy Diebolda. Zeszyt 115 (E 165)
SIK w okresie przejściowym. Kierownik SIK w przedsiębiorstwie przyszłości. Dokąd zmierzasz kierowniku SIK.
Materiały przeznaczone są dla pracowników SIK.

● **Wybrane zagadnienia programowania modularnego na komputer ODRA 1300 — BYC T., Wyd. Zjednoczenia Informatyki — Centrum Projektowania i Zastosowań Informatyki, Warszawa 1980, s. 108, cena 82 zł**

Problemy Informatyki

Opis metodyki: Programowanie modularne. Rodzaje programów. Definicja programowania modularnego. Korzyści programowania modularnego. Odwzorowanie struktury modularnej. Zasady kodowania modułów. Sposób kopilagi ODRY 1300. Łączenie języków programowania. Testowanie modułów. Schemat działania pakietu STEM. Fazy programowania techniką modularną.
Materiały przeznaczone są dla programistów.

● **System MERA 9150. Vademecum superoperatora — Wyd. Przemysłu Maszynowego WEMA, Warszawa 1980, s. 70**

Informacje ogólne. Modus superoperatora. Procedury operowania: paczkami, we/wy, jednostkami taśmowymi. Biblioteki. Sekwencja rozkazów. Operacje pomocnicze. Wprowadzenie formatów (programów). Zestawienie komunikatów błędów w systemie MERA 9150.
Podręcznik przeznaczony jest dla operatorów.

● **Algorytmy i programy sterowania — GÓRECKI H. (Kier. pr. zb.), WNT, Warszawa 1980, s. 534, cena 125 zł**

Charakterystyki statystyczne systemu. Opis dynamiki systemów liniowych o parametrach stałych i skupionych. Badanie stabilności. Obliczanie pierwiastków równania charakterystycznego. Obliczanie rozwiązania równań stanu dla systemów dynamicznych. Synteza układów regulacji. W pracy podano gotowe programy w języku FORTRAN napisane na maszynę ODRA 1304.
Książka przeznaczona jest dla inżynierów i studentów starszych lat studiów, specjalizujących się w automatyce.

● **Programowanie strukturalne — Wyd.Towarzystwa Naukowego Organizacji i Kierownictwa, Szczecin 1980, s. 100**

Oprogramowanie systemów informatycznych. Podejście strukturalne. Programowanie modularne. Strukturoprogramy. Przykład zastosowania.
Materiały przeznaczone są dla programistów.

Cena 30 zł