11 1980

informatyka

WYDAWNICTWO NOT SIGMA

Proponujemy współpracę

Piszcie!

Środowisko informatyków, do którego przede wszystkim adresowana jest INFORMATYKA, mimo że formalnie dość jednolite, sprawia wrażenie zdezintegrowanego. Postępujący coraz szybciej podział na wąskie specjalizacje i sfery zainteresowań może w rezultacie spowodować zanik kontaktów pomiędzy poszczególnymi grupami. Informatyka jako całość stałaby się wtedy dziedziną abstrakcyjną, w ramach której praktyka ograniczona by została wyłącznie do wycinkowych prac konstruktorów sprzętu i oprogramowania, projektantów lub analityków systemów.

Powolny przepływ informacji i wspomniane trudności wymiany poglądów pomiędzy informatykami mogą doprowadzić nie tylko do znacznych strat gospodarczych i opóźnień technologicznych, ale także do następstw typu psychicznego — obojętności i zniechęcenia. Drogą do dającego satysfakcję współdziałania jest przecież przede wszystkim porozumienie.

INFORMATYKA, jako pismo fachowe, jest szansą dla środowiska. Jej łamy, otwarte dla wszystkich zainteresowanych, powinny zostać w większym niż dotychczas stopniu wykorzystane do szerokiej wymiany doświadczeń i poglądów.

Piszcie do nas o wszystkim, co w ramach informatyki wydaje Wam się ważne. Spróbujmy razem opisać rzeczywisty stan tej dziedziny w Polsce. Ułatwi to niewątpliwie ujawnienie istniejących możliwości i ograniczeń, a także spowoduje, że pojęcie „informatyka" stanie się czymś realnym, a w afekcie — w pełni zrozumiałym dla całego społeczeństwa. Ze swojej strony gwarantujemy każdą, możliwą w warunkach Redakcji pomoc.

W rubryce POGLĄDY będziemy zamieszczać rzeczowe, wnikliwe wypowiedzi, dotyczące najważniejszych problemów informatyki, nowych zastosowań, interesujących pomysłów i koncepcji. Znajdą tu miejsce dyskusje, wywiady, a także omówienia ważkich problemów i osiągnięć zagranicznych.

Jest ponadto rubryka LISTY, którą wykorzystać może każdy dla przekazania swoich racji — niekoniecznie z pozycji wybitnego eksperta. Spodziewamy się tutaj między innymi głosów krytycznych, informujących o wszelkich nieprawidłowościach, z jakimi niestety często stykają się użytkownicy informatyki, a także uwag oraz postulatów w stosunku do treści naszego pisma.

Zatem — czekamy.

REDAKCJA

WYDAWNICTWO NACZELNA ORGANIZACJA TECHNICZNA

SIGMA

CZASOPISMA I KSIĄŻKI TECHNICZNE

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. 401. Papier druk. sat. V kl. 70 g. A1. Obj. 5 ark. druk. Nakład 7000 egz. O-51.

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

Rzeczkowski W., Subieta K.: LINDA — system zarządzania bazą danych. Część 1
INFORMATYKA 1980, nr 11, s. 4

Charakterystyka systemu zarządzania bazą danych przeznaczonego do stosowania na komputerach serii ODRA 1300. System ten, zrealizowany w Instytucie Podstaw Informatyki PAN, oparto o koncepcję lingwistycznego podejścia do teorii baz danych. Dwuletnia eksploatacja użytkowa potwierdziła przydatność systemu do tworzenia baz danych średniej wielkości. W części pierwszej artykułu podano ogólne charakterystyki systemu, języka opisu danych oraz języka manipulacji danymi.

Жечковски В., Субета К.: LINDA — система управления базой данных. Часть I
ИНФОРМАТИКА 1980, № 11, стр. 4

Характеристика системы управления базой данных, предназначенной для применения на вычислительных машинах серии ОДРА 1300. Эта система разработана в Институте Основ Вычислительной Техники Польской Академии Наук. Она основана на концепции лингвистического подхода к теории баз данных. Двулетняя эксплуатация подтвердила пригодность системы для создавания баз данных средней величины. В первой части статьи даются общие характеристики системы, языка описания и языка манипуляции данными.

Drabent W.: Definiowanie języków programowania — Druga Metoda Wiedeńska (semantyka)
INFORMATYKA 1980, nr 11, s. 7

Druga część prezentacji precyzyjnego sposobu definiowania języków programowania w oparciu o Drugę Metodę Wiedeńską (Vienna Development Method), poświęcona definiowaniu semantyki. W zakończeniu autor dokonuje ogólnej oceny walorów użytkowych Metody Wiedeńskiej.

Драбэнт В.: Определение языков программирования — Второй венский метод (семантика)
ИНФОРМАТИКА 1980, № 11 стр. 7

Вторая часть представления точного способа определения языков программирования на основе Второго венского метода (Vienna Development Method), касающаяся определения семантики. В конце статьи ее автор дает общую оценку эксплуатационных достоинств венского метода.

Puchałka T.: Zadania i algorytmy komputerowego sterowania procesami technologicznymi
INFORMATYKA 1980, No 11, p. 10
Ogólna charakterystyka zadań i algorytmów komputerowego sterowania procesami technologicznymi. Omówiono model podstawowy oraz niektóre modele szczegółowe projektowania takich systemów w dostosowaniu do konkretnych procesów.

Пухалка Т.: Задачи и алгоритмы автоматического управления технологическими процессами
ИНФОРМАТИКА 1980, № 11, стр. 10

Общая характеристика задач и алгоритмов автоматического управления технологическими процессами. Обсуждается основная модель и некоторые подробные модели проектирования таких систем в приспособлении к конкретным процессам.

Bujko J., Styczyński Z.: Zasady archiwizacji w systemie projektowania wspomaganego komputerem
INFORMATYKA 1980, nr 11, s. 13

Charakterystyka rozwiązań organizacyjnych i programowych systemu archiwowania w pamięciach zewnętrznych dokumentacji projektowania wspomaganego komputerem.

Буйко Й., Стычиньски З.: Принципы архивизации в системе проектирования вспомогаемой вычислительной машиной
ИНФОРМАТИКА 1980, № 11, стр. 13

Характеристика организационных и программных решений систем архивирования во внешних памятях документации проектирования вспомогаемой вычислительной машиной.

Dobrucki L.: Wybór systemu zarządzania bazą danych dla INFOCHEMU
INFORMATYKA 1980, nr 11, s. 16

Charakterystyka sposobu wyboru systemu zarządzania bazą danych w oparciu o doświadczenia praktyczne zebrane w czasie projektowania wielodziedzinowego systemu informatycznego dla zakładów przemysłu chemicznego. Podano uzasadnienie oraz metodykę oceny prawidłowości takiego wyboru.

Добруцки Л.: Выбор системы управления базой данных для INFOCHEM
ИНФОРМАТИКА 1980, № 11, стр. 16

Характеристика способа выбора системы управления базой данных на основе практического опыта, собранного во время проектирования многоотраслевой вычислительной системы для предприятий химической промышленности. Дается обоснование и методика правильности оценки такого рода выбора.

Bednarz J., Majewski M.: Przygotowanie użytkownika bazy danych
INFORMATYKA 1980, nr 11, s. 20

Charakterystyka zasad oraz przedsięwzięć organizacyjnych jakie należy stosować po podjęciu decyzji o wdrażaniu systemu informatycznego w przedsiębiorstwie w oparciu o wspólną bazę danych.

Бэднаж Й., Маевски М.: Приготовление потребителя базы данных
ИНФОРМАТИКА 1980, № 11, стр. 20

Характеристика принципов и организационных предприятий, какие следует использовать, принимая решения, касающиеся внедрения автоматизированной системы в предприятии на основе общей базы данных.

Kornacki W.: Wybrane problemy analizy niezawodnościowej oprogramowania
INFORMATYKA 1980, nr 11, s. 22

Zdefiniowanie oraz klasyfikacja podstawowych pojęć z zakresu badań nad niezawodnością programów. Podano przegląd oraz istotę stosowanych obecnie metod szacowania niezawodności programów.

Корнацки В.: Избранные проблемы анализа надежности программного обеспечения
ИНФОРМАТИКА 1980, № 11, стр. 22

Определение и классификация основных понятий из области исследования надежности программ. Дается просмотр и указывается на сущность применяемых в настоящее время методов оценки надежности программ.

Rzeczkowski W., Subieta K.: LINDA — a data base management system. Part 1
INFORMATYKA 1980, No 11, p. 4

Characteristics of the data base management system appropriated for application on ODRA 1300 computer series. The system, realised in the Institute for Data Processing Principles of the Polish Academy of Sciences, has been based on linguistic approach to data base theory. Two years of useful operation confirmed the systems ability for creation of the middle class data bases. In the first part of the article presented general characteristics of the system, the data description language and the data manipulation language.

Rzeczkowski W., Subieta K.: LINDA — ein Datenbasisverwaltungssystem. Teil 1
INFORMATYKA 1980, Nr. 11, S. 4

Die Charakteristik eines Datenbasisverwaltungssystems für Anwendung auf der ODRA 1300 Rechnerserie. Das System, realisiert im Informatikgrundlageninstitut der Polnischen Akademie der Wissenschaften wurde auf dem Konzept der linguistischen Einstellung zur Datenbasistheorie gestützt. Der zweijährige Gebrauchsbetrieb bestätigte die Eignung des Systems für die Schaffung von Datenbasen mittlerer Grösse. Im ersten Teil des Artikels wurden die allgemeinen Charakteristiken des Systems, der Datenbeschreibungssprache und der Datenmanipulationssprache angegeben.

Drabent W.: Defining of programming languages — the Vienna Development Method (Semantics)
INFORMATYKA 1980, No 11, p. 7

The second part of the presentation of the way for precise programming languages defining used by the Vienna Development Method and is devoted semantics. In the conclusion the author makes a general appreciation of the method usable advantages.

Drabent W.: Definieren von Programmierungssprachen — die Zweite Wiener Methode (die Semantik)
INFORMATYKA 1980, Nr. 10, S. 7

Der zweite Teil der Vorstellung eines Verfahrens für präzise Definieren der Programmierungssprachen, gestützt auf der Zweiten Wiener Methode (the Vienna Development Method) und ist gewidmet der Semantik. In dem Schlussteil schätzt der Autor den Gebrauchswert der Methode.

Puchałka T.: Tasks and algorithms of computerized technologic process control
INFORMATYKA 1980, No 11, p. 10

General characteristics of tasks and algorithms for computerized technological process control. Discussed the basic and some detailed models of such systems designing in adaptation to specific processes.

Puchałka T.: Aufgaben und Algorithmen der rechnergestützten Prozesssteuerung
INFORMATYKA 1980, Nr. 11, S. 10

Allgemeine Charakteristik der Aufgaben und Algorithmen der rechnergestützten Prozesssteuerung. Es wurden das Basismodell und einige ausführliche Modelle für die Projektierung von solchen Systemen in der Anpassung zu konkreten Prozessen besprochen.

Bujko J., Styczyński Z.: Archiving principles in the computer assisted designing system
INFORMATYKA 1980, No 11, p. 13

Characteristics of organizational and programming solutions for archiving system of the computer assisted designing documentation in external storages.

Bujko J., Styczyński J.: Archivierungsgrundsätze in dem rechnergestützten Projektierungssystem
INFORMATYKA 1980, Nr. 11, S. 13

Die Charakteristik von organisatorischen und programmtechnischen Lösungen des Systems für die Archivierung der rechnergestützten Projektierungsdokumentation in den externen Speichern.

Dobrucki L.: The choice of data base management system for INFOCHEM
INFORMATYKA 1980, No 11, p. 16

Characteristic of the method for data base management system choice based on practical experience gathered while designing the multiarea data processing system for chemical industry plants. Presented motives and estimation methodology for the correct choice.

Dobrucki L.: Die Wahl des Datenbasisverwaltungssystems für INFOCHEM
INFORMATYKA 1980, Nr. 11, S. 16

Die Charakteristik einer Methode für die Wahl eines Datenbasisverwaltungssystems, die auf Grund der praktischen, während der Projektierung eines Mehrbereich-EDV-Systems für die chemischen Industriebe gesammelten Erfahrungen erarbeitet wurde. Es wurden die Begründung und die Schätzungsmethodik einer solchen Wahl angegeben.

Bednarz J., Majewski M.: Preparation of the data base user
INFORMATYKA 1980, No 11, p. 20

Characteristics of principles and organizational ventures which are to apply after taking a decision for data processing system implementation in the enterprise based on common data base concept.

Bednarz J., Majewski M.: Vorbereitung des Datenbasisbenutzers
INFORMATYKA 1980, Nr. 11, S. 20

Die Charakteristik von Grundlagen und organisatorischen Massnahmen, die man nach den Beschluss über den Einsatz des auf der gemeinsamen Datenbasis gestützten EDV-Systems anwenden soll.

Kornacki W.: Selected problems of software reliability analysis
INFORMATYKA 1980, No 11, p. 22

Definition and classification of basic terms from the field of software reliability research. Presented survey and essence of actually applied methods for estimation of programs reliability.

Kornacki W.: Ausgewählte Probleme der Softwarezuverlässigkeitanalyse
INFORMATYKA 1980, Nr. 11, S. 22

Definieren und Klassifizierung der Grundbegriffe aus dem Gebiete der Programmenzuverlässigkeitforschung. Es wurden die Vorschau und das Wesen der heute verwendeten Methoden für die Schätzung der Programmenzurverlässigkeit angegeben.

Informatyka

zastosowania w gospodarce, technice i nauce

Nr 11
MIESIĘCZNIK
1980
ROK XV
Listopad

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

W NUMERZE: Strona

WIKTOR RZECZKOWSKI, KAZIMIERZ SUBIETA
Instytut Podstaw Informatyki PAN
Warszawa

LINDA—system zarządzania bazą danych. Część 1

W ostatnim czasie obserwuje się ogromne zainteresowanie zagadnieniami baz danych. Wychodząc naprzeciw zapotrzebowaniu, w Instytucie Podstaw Informatyki PAN opracowano dla komputerów serii ODRA 1300 system zarządzania bazą danych LINDA [3, 4, 6]. System ten przynajmniej częściowo wypełnia lukę w całokształcie dostępnego oprogramowania komputerów ODRA. Koncepcja systemu została oparta na rezultatach prac w zakresie lingwistycznego podejścia do teorii baz danych [7, 8]. System został zrealizowany w czerwcu 1978 roku i od tego czasu jest poddawany próbnej eksploatacji w kilku instytucjach, m.in. w Centralnym Instytucie Ochrony Pracy w Warszawie [1, 2] oraz w Ośrodku Informacji Naukowej PAN w Warszawie [9, 10]. Eksploatacja w pełni potwierdziła przydatność systemu do tworzenia baz danych średniej wielkości.

OGÓLNA CHARAKTERYSTYKA SYSTEMU

Oprogramowanie systemu LINDA zostało napisane w języku PLAN 4 (22AM). Wykorzystany został system automatyzacji pamięci SAP oraz system obsługi pamięci dynamicznej PADYN [5]. SZBD LINDA działa pod nadzorem ręcznego egzekutora komputera ODRA 1305 typu UDAS. Przy eksploatacji systemu można wykorzystywać niektóre standardowe programy organizacyjne dla pamięci taśmowych i dyskowych, w szczególności program #XKYA do tworzenia i aktualizacji źródłowych zbiorów taśmowych. Zapotrzebowanie na pamięć operacyjną waha się zwykle — w zależności od trybu wykorzystania systemu — w granicach od 48 K do 55 K słów. W czasie pracy systemu używane są następujące urządzenia zewnętrzne:

— czytnik kart i drukarka wierszowa
— pamięć taśmowa (od 0 do 5 jednostek)
— pamięć dyskowa (nieobowiązkowo).

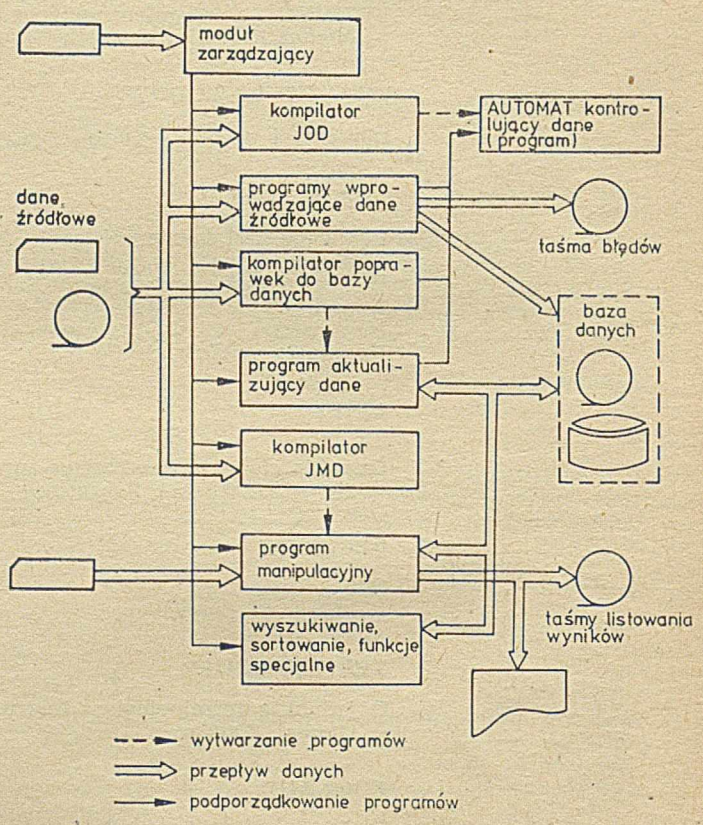
SZBD LINDA jest systemem uniwersalnym, w oparciu o który można zbudować systemy informacyjne dla wielu różnorodnych dziedzin. Przystosowanie systemu do pracy w konkretnej dziedzinie zastosowań tj. budowa konkretnego systemu informacyjnego, polega na:

- opracowaniu wzorców typów rekordów, które będą przechowywane w bazie danych, i zapisanie tych wzorców w specjalnym języku opisu danych (JOD); wyrażenie JOD przygotowuje się jeden raz na dłuższy okres eksploatacji systemu w danej dziedzinie
- przygotowaniu danych i wprowadzeniu ich do systemu; etap ten jest zwykle w eksploatacji systemu najbardziej pracochłonny i kosztowny, z tego też względu opracowano i wdrożono w SZBD LINDA szereg mechanizmów podnoszących efektywność przygotowania i wprowadzania danych
- opracowaniu projektu typowych wydruków, które będą dostarczane przez system informacyjny, i napisaniu odpowiednich programów w specjalnym języku — języku manipulacji danymi (JMD) systemu LINDA; JMD jest językiem progamowania wyższego rzędu, uwzględniającym szczególnie zadania związane z przetwarzaniem danych zawartych w bazie
- opracowaniu trybu i kalendarza wykonania programów manipulacyjnych, aktualizacji bazy danych oraz wykorzystania wyników przetwarzania.

Przebiegi pracy SZBD LINDA na rzecz konkretnego systemu informacyjnego zapisuje się w specjalnym języku sterowania. Wyrażenia tego języka podawane są na wejście modułu zarządzającego systemu LINDA.

Główne funkcje, które wykonuje system, są następujące:
- kompilacja wyrażeń języka opisu danych
- tworzenie bazy danych na nośnikach taśmowych lub dyskowych; podczas wprowadzania danych następuje kontrola ich zgodności z wyrażeniem JOD.
- kompilacja i wykonanie programów aktualizacyjnych
- kompilacja i wykonanie programów manipulacyjnych
- wyszukiwanie danych
- sortowanie danych
- zespół specjalnych funkcji ułatwiających obsługę systemu.

System można zastosować w wielu różnych dziedzinach. W szczególności może on być przydatny w badaniach naukowych, w zastosowaniach techniczno-organizacyjnych, do automatyzacji prac administracyjnych, itp. Wydajność systemu przy średniej złożoności przetwarzania jest rzędu 20 rekordów na sekundę. Budowę jego przedstawiono schematycznie na rysunku 1.



Rys. 1. Schemat budowy SZBD LINDA

DEFINICJE DANYCH I JOD

W systemie LINDA przyjęto, że dana składa się z nazwy i wartości, przy czym wartość danej może być elementarna, tj. niepodzielna lub też może być dowolnym ciągiem danych. Przyjęto zapisywać daną jako ciąg $n(w)$, gdzie n jest nazwą danej, zaś w jest zamkniętą w nawiasy okrągłe wartością danej. Przykładowo, następujące wyrażenie reprezentuje dane standardowe w sensie systemu LINDA:

```
PRACOWNIK (NAZWISKO (NOWAK) IMIE (JAN) IMIE
   (ADAM)
   ZAROBEK (4000) DATA URODZENIA (1944) PLEC (M))
PRACOWNIK (NAZWISKO (KOWALSKA) IMIE (EWA)
   NAZWISKO PANIENSKIE (MALINA)
   ZAROBEK (3500) DATA URODZENIA (1951) PLEC (K))
```

Dane w postaci standardowej wyznaczają logiczną treść w stopniu wystarczającym do manipulacji. W rzeczywistości fizyczna reprezentacja danych w LINDZIE jest inna.

Wyrażenia języka opisu danych (JOD) systemu LINDA umożliwiają zadeklarowanie dowolnej liczby typów rekordów o praktycznie nieograniczonej złożoności. LINDA przetwarza wyrażenie JOD na program, którego zadaniem jest kontrola poprawności danych wprowadzanych do systemu bądź też w nim przechowywanych. Opis typu rekordu jest prosty i naturalny. Dla przykładu — opis rekordu pracownika może mieć postać:

```
#RECORD #OCCUR
   PRACOWNIK (
      NAZWISKO (#NAME)
      #OCCUR 3 IMIE (#NAME) #CO
      [NAZWISKO PANIENSKIE (#NAME)]
      ZAROBEK (#INT)
      DATA URODZENIA (#DATE)
      PLEC (K, M))
#CO #FIN
```

Powyższe wyrażenie opisuje bazę danych zawierającą dowolną liczbę rekordów pracowników (rekordów o nazwie PRACOWNIK), którzy są scharakteryzowani przy pomocy atrybutów: NAZWISKO, IMIE, NAZWISKO PANIENSKIE, ZAROBEK, DATA URODZENIA i PLEC. Atrybut IMIE może wystąpić kilkakrotnie (co jest zadeklarowane przez ograniczenie jego opisu słowami kluczowymi #OCCUR i #CO), lecz nie więcej niż 3 razy. Atrybut NAZWISKO PANIENSKIE może nie wystąpić (co jest zadeklarowane nawiasami []). Dopuszczalne wartości atrybutów są zadeklarowane słowami kluczowymi #NAME (ciąg liter i cyfr), #INT (liczba całkowita) lub #DATE (data w pisowni dzień. miesiąc. rok). Można także używać podobnych deklaracji: #STRING (dowolny ciąg znaków), #REAL (liczba rzeczywista), itd. Dla atrybutu PLEC dopuszczalna jest wartość K lub M.

W systemie przewidziane są dwa tryby wprowadzania danych: w zmiennym i stałym formacie źródłowym. Przy wprowadzaniu w zmiennym formacie poszczególne rekordy źródłowe mogą mieć różną długość, w zależności od liczby i długości zawartych w nim danych elementarnych. Przygotowywane dane są zapisywane w specjalnym języku, który zawiera elementy umożliwiające odseparowanie od siebie poszczególnych danych oraz określenie ich pożądanej struktury. Język ten umożliwia także znaczną kompresję danych, co jest szczególnie ważne przy przygotowaniu ich na nośnikach papierowych.

W przypadku stałego formatu nie stosuje się natomiast żadnego specjalnego języka. Każdy rekord źródłowy danego typu musi mieć jednakową długość. Także pozycje danych elementarnych w rekordach źródłowych muszą być ustalone. Dzięki temu LINDA może akceptować pliki kart lub taśmy magnetyczne z danymi źródłowymi przygotowane dla innych systemów oprogramowania, np. taśmy sporządzone przy użyciu programów w języku COBOL. Aby ustalić związek wyrażenia JOD z tak przygotowanymi danymi, użytkownik musi przygotować specjalne wyrażenie zwane parametrami stałego formatu. Syntaktycznie rzecz biorąc, parametry stałego formatu przypominają wyrażenie JOD, z tą różnicą, że zamiast specyfikacji dla danej elementarnej (takich jak #NAME, #INT) podaje się numery pozycji w rekordzie źródłowym (numery kolumn na kartach perforowanych), na których występuje wartość danej.

Dane źródłowe zapisane w zmiennym lub stałym formacie poddaje się procesowi przekształcania do postaci standardowej, stanowiącej wejście programu kontrolującego, który powstał z wyrażeń JOD. W systemie przewidziano możliwość automatycznego poprawiania niektórych drobnych pomyłek w danych. Jeżeli rekord nie zawiera błędów, wówczas zostaje padany na wejście programu formującego, którego zadaniem jest uformowanie rekordu fizycznego (maszynowego) o organizacji umożliwiającej sprawny dostęp do wszystkich danych. Rekordy fizyczne są odsyłane na nośnik taśmowy lub dyskowy. W systemie LINDA rekordy fizyczne mają zmienną długość (niezależnie od stałego bądź zmiennego formatu rekordów źródłowych) i zajmują tyle miejsca, ile zajmują wartości zawartych w nich danych plus jedno słowo dla każdego wystąpienia nazwy danych.

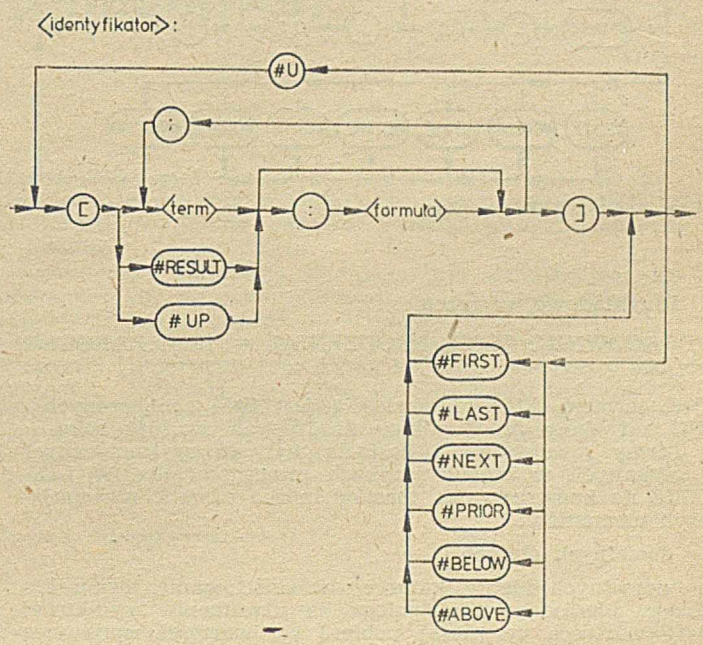
LINDA umożliwia — w trakcie wprowadzania danych źródłowych — wytworzenie taśmy zawierającej wszystkie błędne rekordy w niezmienionej postaci. Dane zawarte na tej taśmie można poprawić przy użyciu standardowych programów edycyjnych i redakcyjnych, a następnie wprowadzić ponownie do systemu, wykorzystując te same, opisane wyżej mechanizmy.

MANIPULACJA DANYMI I JMD

Manipulacja danymi odbywa się przy użyciu programów napisanych specyficznym dla systemu LINDA językiem programowania. Język ten, nazywany w dalszym ciągu językiem manipulacji danymi (JMD), w swej ogólnej konstrukcji nie odbiega zbytnio od takich znanych języków programowania wysokiego rzędu, jak ALGOL, FORTRAN czy PASCAL. Specyfika tego języka polega na tym, że po pierwsze — dostarcza szeregu funkcji umożliwiających operacje na bazie danych, po drugie zaś — że posiada rozbudowane operacje związane z wyprowadzaniem graficznych rezultatów przetwarzania danych. Zaletą języka manipulacji jest duży stopień nieproceduralności, która jest w dużej mierze zrealizowana poprzez wprowadzenie języka identyfikacji danych (JID). Język ten zawiera trzy — wzajemnie z sobą związane — konstrukcje syntaktyczno-semantyczne: identyfikatory, termy i formuły. Niżej je omówimy.

● Identyfikatory

Gramatykę identyfikatorów ilustruje rysunek 2. Z punktu widzenia semantycznego, identyfikatory są to wyrażenia, których wartością jest lista adresów danych (może być pusta). Np. wyrażenie [PRACOWNIK: [ROK URODZENIA #1944] jest identyfikatorem, który przyjmie wartość będącą listą adresów rekordów pracowników, w których pole ROK URODZENIA ma wartość 1944. W identyfikatorach można używać dwóch mechanizmów selekcji danych: poprzez nazwę i poprzez wartość danej. Selekcję poprzez nazwę uzyskujemy używając w wyrażeniu nazwy danej; selekcję poprzez wartość uzyskujemy budując odpowiednią formułę (wyrażenie boolowskie) na wartościach danych.



Rys. 2. Gramatyka identyfikatorów

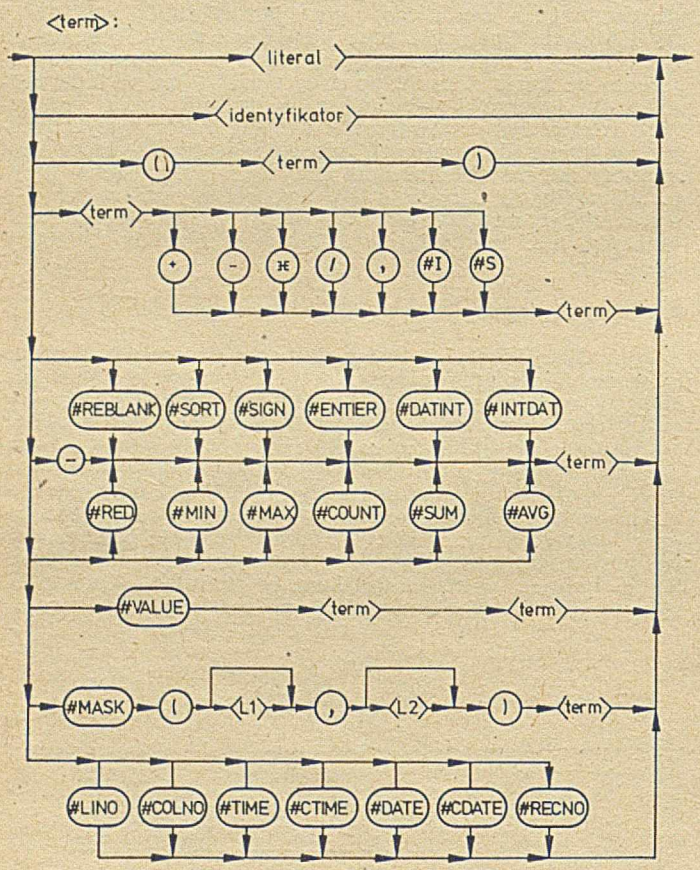
Przykładowo, identyfikator o postaci:

[PRACOWNIK: [ZAROBEK] > 3000; NAZWISKO]

selekcjonuje dane w trzech etapach. W etapie pierwszym następuje wybranie rekordów o nazwie PRACOWNIK. W trakcie etapu drugiego odrzucane są te rekordy, w których wartość pola ZAROBEK jest mniejsza lub równa 3000. W trzecim etapie następuje przejście na niższy poziom hierarchii danych (poziom pól w wybranych do tej pory rekordach PRACOWNIK) i wyselekcjonowanie z tego poziomu wszystkich pól NAZWISKO. W rezultacie utworzy się listę adresów pól NAZWISKO w rekordach pracowników z zarobkiem większym od 3000.

● Termy

Gramatykę termów ilustruje rysunek 3. Term jest wyrażeniem, którego wartością jest lista (może być pusta) wartości elementarnych (w niektórych zastosowaniach dopuszcza się nieelementarne wartości termu, tj. pewne bardziej złożone struktury). Dla przykładu, dowolna stała jest termem, dowolny ciąg termów jest termem, dowolny identyfikator jest termem, dwa termy połączone symbolem operacji tworzą term, itd.



Rys. 3. Gramatyka termów

Przykładowo, wyrażenie:

[PRACOWNIK: [ROK URODZENIA] = 1944; ZAROBEK] + 500

jest termem, którego wartością jest lista zwiększonych o 500 zarobków pracowników urodzonych w 1944 roku.

Termy są uogólnioną postacią wyrażeń arytmetycznych spotykanych w innych językach programowania. W LINDZIE do konstruowania termów można używać następujących elementów:

— dowolnych stałych

— dowolnych identyfikatorów; wartościowanie termu będącego identyfikatorem polega na utworzeniu wynikowej listy wartości, do której pobiera się wartości znajdujące się pod kolejnymi adresami będącymi wartościowaniem identyfikatora

— termów specjalnych, informujących na ogół o stanie środowiska systemu, np. #DATE — bieżąca data, #TIME — bieżący czas, #LINO — numer ostatnio zadrukowanej linii na drukarce, itp.

— symboli operacji arytmetycznych (w dowolnych formach nawiasowych), takich jak + — * /. Ponieważ termy mogą być wieloznaczne, przyjęto, że operacje wykonuje się metodą „każda wartość z każdą". Przykładowo, term (5, 3, 18) + (2, 8) przyjmuje znaczenie będące listą wartości: (7, 13, 5, 11, 20, 26)

— symboli funkcji, np. #SQRT — pierwiastek kwadratowy, #SIGN — znak liczby, #ENTIER — część całkowita liczby, #DATINT — zamiana daty na liczbę dni od pewnego punktu odniesienia, itd.

— symboli funkcji agregowanych (tj. takich, które biorą pod uwagę jednocześnie wszystkie wartości termu), np.:

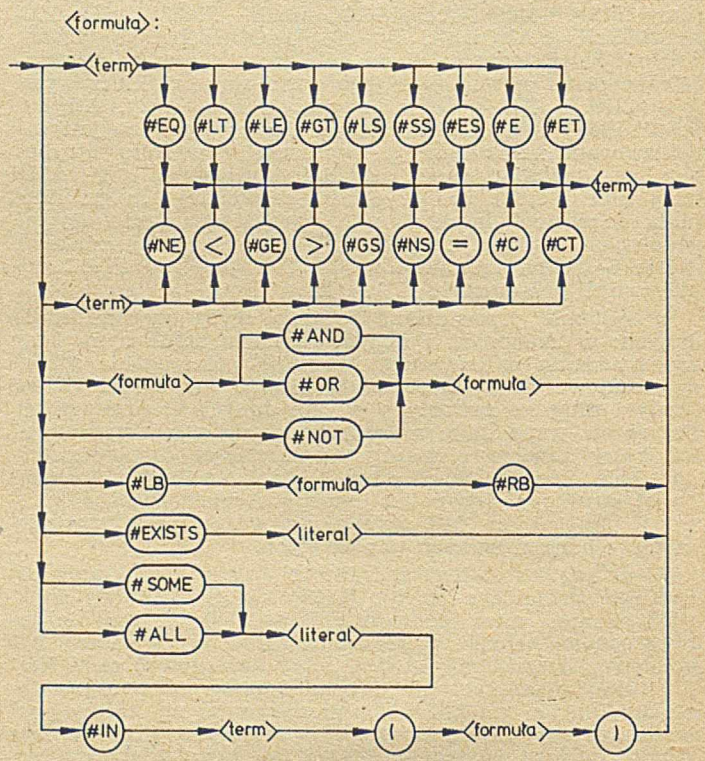
#MAX — element największy spośród wartości termu
#SUM — suma wartości termu
#RED — usunięcie powtórzeń z listy wartości termu, itd.

— symboli operacji mnogościowych:

, — konkatenacja ciągów wartości dwóch termów (w sensie mnogościowym może być rozpatrywana jako suma zbiorów wartości)
#I — przecięcie zbiorów wartości (część wspólna)
#S — różnica zbiorów wartości.

● Formuły

Gramatyka formuł jest przedstawiona na rysunku 4. Wartościowanie dowolnej formuły polega na przypisywaniu jej jednego elementu ze zbioru {prawda, fałsz}. Formuły są uogólnioną postacią wyrażeń boolowskich. Najprostszą formułę uzyskamy, jeśli dwa termy połączymy symbolem relacji, np. [ZAROBEK] > 3000 lub KOWALSKI = = [NAZWISKO].



Rys. 4. Gramatyka formul

W systemie LINDA można używać relacji należących do następujących kategorii:

— porównania liczb, np. #EQ — równość liczb, < — „mniejsze", #LE — „mniejsze równe", itd.

— porównania ciągów znaków, np.: #ES — identyczność ciągów, #LS — zawieranie się ciągu lewego w prawym, #SS — podobieństwo ciągów z dokładnością do jednego znaku, itd.; w przypadku termów wieloznacznych wynik porównania jest prawdziwy, jeżeli chociaż jedna para elementów znajduje się w odpowiedniej relacji

— porównania mnogościowe, np. #E — równość zbiorów, #C — inkluzja zbiorów, #CT — inkluzja ciągów wartości, = — niepuste przecięcie zbiorów.

Formuły można łączyć używając spójników logicznych #AND, #OR, #NOT (oznaczających odpowiednio: konjunkcję, alternatywę, negację) w dowolnych formach nawiasowych. Do budowania formuł można także używać kwantyfikatorów — szczegółowego #SOME i ogólnego #ALL w zwyczajowo przyjętych zapisach matematycznych, a także funkcji #EXISTS służącej do zbadania, czy istnieje w polu widzenia dana o określonej nazwie.

LITERATURA

[1] Karczewski J.: Koncepcja wykorzystania technologii baz danych dla potrzeb ochrony pracy, OCHRONA PRACY nr 1, 1978
[2] Karczewski J., Nałęcz A., Chodorowski J., Rzepecki J.: Projekt systemu informatycznego LAPIS o zagrożeniach zawodowych, Praca planowa CIOP, 1979
[3] Kopeć B., Kuta M., Rzeczkowski W., Subieta K.: Systemy zarządzania bazą danych LINDA, Prace IPI PAN nr 375, 1979
[4] Rzeczkowski W.: Statystyczny moduł systemu LINDA, Prace IPI PAN nr 376, 1979
[5] Rzeczkowski W.: Pamięć dynamiczna dla emc ODRA serii 1300, Prace IPI PAN nr 315, 1978
[6] Subieta K.: Nowe możliwości systemu LINDA, Prace IPI PAN nr 378, 1979
[7] Subieta K.: Linguistic Approach to Database Theory: DDL's for Hierarchical Model, Information Systems, vol. 3, 1978
[8] Subieta K.: Data Indication Language, Opracowanie wewnętrzne IPI PAN (nie publikowane), 1978
[9] Subieta K.: Założenia systemu informacyjnego „Wielka Emigracja", Kwartalnik Historyczny nr 1, 1976
[10] Wyczańska K.: System informacyjny „Wielka Emigracja", Kwartalnik Historyczny nr 1, 1976

Prezentujemy drugą część artykułu o Drugiej Metodzie Wiedeńskiej. Aby ułatwić lekturę powtarzamy na końcu tablicę 2 i Przykład P2, zamieszczone w poprzednim numerze. Sądzimy jednak, że niezależnie od tego warto sobie przypomnieć część pierwszą, tym bardziej, że omawiane w części drugiej przykłady są jej kontynuacją. (Red.)

WŁODZIMIERZ DRABENT
Politechnika Warszawska

Definiowanie języków programowania Druga Metoda Wiedeńska (semantyka)

W drugiej metodzie wiedeńskiej semantyka oparta jest na metodzie denotacyjnej (zwanej też matematyczną), stworzonej przez Ch. Strachey'a i D. Scotta pod koniec lat sześćdziesiątych. W metodzie tej znaczeniami programów są pewne obiekty matematyczne (zwykle funkcje). Semantykę określa się poprzez zdefiniowanie **funkcji semantycznych** przyporządkowujących znaczenia elementom dziedzin składniowych (wartościami tych funkcji mogą być również funkcje). Zasadą porządkującą semantykę jest określanie znaczenia konstrukcji złożonej w oparciu o znaczenie jej składników. Np. w naszej definicji znaczenie instrukcji pętli zależy wyłącznie od znaczenia zawartych w niej: wyrażenia i instrukcji; znaczenie instrukcji złożonej — od znaczenia jej instrukcji składowych itd.. Dzięki prostocie definiowanego języka, znaczeniami niektórych obiektów, np. identyfikatorów, mogą być one same.

Leżąca u podstaw semantyki denotacyjnej teoria matematyczna jest jednym z ciekawszych przykładów zastosowania matematyki w informatyce i stymulującego wpływu informatyki na matematykę. Teoria ta ogranicza klasę stosowanych funkcji i dziedzin tak, aby uniknąć paradoksów związanych z operowaniem funkcjami na równi z innymi obiektami. Teoria ta jest raczej trudna, ale wydaje się, że dla korzystającego z denotacyjnych definicji semantyki znajomość podstaw matematycznych nie jest konieczna.

Krótkie wprowadzenie do semantyki denotacyjnej zawiera artykuł [4]; zainteresowanych jej matematycznymi aspektami można odesłać do artykułów Stoy'a w [1].

Zasadniczą częścią semantyki są definicje funkcji semantycznych. Pomocniczą rolę pełnią definicje dziedzin semantycznych. Określają one, jakie obiekty mogą być wartościami funkcji semantycznych. Ta część opisu języka może być wydedukowana z definicji funkcji semantycznych, ale podanie jej *explicite* istotnie ułatwia tworzenie, a potem rozumienie tych definicji.

Tablica 4. Dziedziny semantyczne

$$Tr = \Sigma \xrightarrow{\sim} \Sigma \qquad (4.1)$$

$$Wart = Int \mid Bool \qquad (4.2)$$

$$\Sigma = Id \xrightarrow[m]{} Wart \qquad (4.3)$$

Dziedziny semantyczne naszego języka podane są w tablicy 4. Dziedzina Σ jest dziedziną stanów — odwzorowań przyporządkowujących zmiennym ich wartości. Stany te można uważać za abstrakcyjny odpowiednik pamięci komputera. Znaczenia wyrażeń należą do dziedziny *Wart* (wartości). Jest ona sumą dwóch dziedzin standardowych. *Bool* jest dziedziną wartości logicznych, a *Int* dziedziną liczb całkowitych. Znaczenia programów i instrukcji należą do dziedziny *Tr* (transformacje, czyli przekształcenia stanów). Symbol $\rightharpoonup$ oznacza operację tworzenia dziedziny funkcji częściowych (tzn. takich, których wartość może być nieokreślona dla pewnych argumentów). W przeciwieństwie do odwzorowań funkcje nie muszą być obiektami skończonymi.

Znaczeniem programu jest przekształcenie stanów, czyli funkcja (częściowa) opisująca efekt jego działania. Przyporządkowuje ona stanowi początkowemu (opisującemu wartości zmiennych przed rozpoczęciem działania programu) stan końcowy (opisujący wartości zmiennych po wykonaniu programu). Dla pewnych stanów początkowych wartość tej funkcji może być nieokreślona. Odpowiada to zapętleniu się lub nienormalnemu zatrzymaniu programu (np. na skutek próby dzielenia przez zero).

Tablica 5. Funkcje semantyczne

```
i-program(mk-program(dek,instrl)) = for i=1 to leninstrl do i-instr(instrl(i))   (5.1)
i-instr : Instr→Tr                                                               (5.2)
i-instr-pusta( ) = I                                                             (5.3)
i-instr-złożona(instrl) = for i=1 to leninstrl do i-instr(instrl(i))             (5.4)
i-instr-podst( mk-instr-podst(id,wyr) )(s) = s + [id→e-wyr(wyr)(s)]              (5.5)
i-instr-war( mk-instr-war(wyr,instr1,instr2))(s)                                 (5.6)
  = if e-wyr(wyr)(s) then i-instr(instr1)(s)
                     else i-instr(instr2)(s)

i-instr-pętli( mk-instr-pętli(wyr,instr))                                        (5.7)
    = let f(s) = if e-wyr(wyr)(s) then f°i-instr(instr)(s) else s
      in f

e-wyr : Wyr → (Σ ⇀ Wart)                                                         (5.8)

e-wyr-2-a( mk-wyr-2-a(wyr1,op,wyr2) )(s)
= let w1 = e-wyr(wyr1)(s) in
  let w2 = e-wyr(wyr2)(s) in
  cases op : (PLUS → w1 + w2,                                                    (5.9)
              EQ → w1 = w2)

e-id(id)(s) = s(id)                                                              (5.10)
```

Tablica 5 zawiera definicje funkcji semantycznych (przykładowego języka, którego składnię abstrakcyjną zawiera tablica 2). Ich nazwy są utworzone od nazw dziedzin syntaktycznych. Dodatkowa litera *i* jest skrótem od „interpret" (interpretuj), a litera *e* — od „evaluate" (oblicz wartość). Użyto tu notacji podobnej jak w tablicy 3 i konwencji opuszczania reguł oczywistych analogicznej do stosowanej tam zasady 1.
Tak więc na przykład opuszczono regułę

e-wyr(w) = **if** w∈Wyr-2-a **then** e-wyr-2-a(w) **else**
if w∈Id **then** e-id(w) **else undefined**

Charakter komentarzy mają (5.2) i (5.8). Występują one na miejscu opuszczonych reguł i wskazują dziedziny, do których należą funkcje *i-instr* i *e-wyr*.

Tak więc np. funkcja *i-instr* przyporządkowuje instrukcji przekształcenie stanów, czyli należy ona do dziedziny $Instr \rightarrow Tr$, co można zapisać inaczej: $Instr \rightarrow (\Sigma \rightharpoonup \Sigma)$ (por. (4.1)). Natomiast *i-instr(instr)* (gdzie *instr* jest jakąś instrukcją) jest wartością funkcji *i-instr* dla argumentu *instr*, czyli przekształceniem stanów będącym znaczeniem instrukcji *instr*, i należy do dziedziny *Tr*.

W regułach (5.1) i (5.4) występuje wyrażenie iteracyjne postaci **for** i = m **to** n **do** e_i.
Jest ono równoważne $e_n \circ e_{n-1} \circ \ldots \circ e_m$ (a dla m > n — funkcji tożsamościowej), gdzie ° oznacza złożenie funkcji:

$$f \circ g\,(x) = f\,(g\,(x)).$$

Operator **len** daje nam długość listy, *instrl(i)* oznacza i-ty element listy *instrl*. Tak więc znaczeniem programu (bądź instrukcji złożonej) jest funkcja będąca złożeniem znaczeń kolejnych jego instrukcji.

Aby się przekonać, że jest to zgodne z intuicją, rozważmy instrukcję złożoną *instrł*, składającą się z dwóch instrukcji *instr1* i *instr2*, których znaczeniem są funkcje *i-instr(instr1)* oraz *i-instr(instr2)*. Wykonanie *instrł* polega na wykonaniu najpierw *instr1*, a następnie *instr2*. Jeżeli rozpoczniemy wykonanie *instrł* w stanie s, to po wykonaniu *instr1* otrzymamy stan będący wartością funkcji *i-instr(i-instr1)* dla argumentu s, czyli *s1=i-instr(instr1)(s)*. Jeżeli teraz wykonamy *instr2*, to otrzymamy stan

i-instr(instr2)(s1) = i-instr(instr2)°i-instr(instr1)(s).

Symbol *I* oznacza funkcję tożsamościową, a więc z reguły (5.3) wynika, że instrukcja pusta nie zmienia stanu. W regule (5.5) występuje operacja uaktualniania odzworowania. *o1 + o2* oznacza odwzorowanie o wartościach takich, jak *o2* dla argumentów należących do dziedziny odwzorowania *o2* oraz o wartościach jak *o1* dla pozostałych argumentów. Tak więc prawa strona reguły (5.5) oznacza stan o wartości zmiennej *id* równej *e-wyr(wyr)(s)* i o wartościach pozostałych zmiennych takich samych jak w stanie s. Zauważmy, że bezpośrednio stąd wynika, że w naszym języku instrukcja podstawienia nie ma efektów ubocznych.

W regułach (5.7) i (5.9) występuje wyrażenie wiążące. Ma ono postać:

let id = e_1 **in** e_2, (ang. *let, in* — niech, w)

gdzie *id* jest metajęzykowym identyfikatorem, a e_1 i e_2 — wyrażeniami. Jest ono równoważne wyrażeniu powstałemu z e_2 przez zastąpienie wszystkich wystąpień identyfikatora *id* przez „wartość stałej *id*" [1]. Wartość ta określona jest w sposób następujący: jeżeli identyfikator *id* nie występuje w e_1 (wersja 1), to jest nią wartość wyrażenia e_1, w przeciwnym wypadku (wersja 2) mamy do czynienia z równaniem postaci id = F(id). Teoria matematyczna semantyki denotacyjnej wprowadza pojęcie rozwiązania takiego równania (najmniejszego punktu stałego funkcji F) oraz ogranicza klasę używanych funkcji i dziedzin tak, by dla każdego równania istniało jednoznaczne rozwiązanie. Wartością stałej *id* jest w tym przypadku rozwiązanie równania $id = e_1$.

A oto kilka prostych przykładów wyrażeń wiążących i wyrażeń im równoważnych:

let a = 5 **in** a + b	5 + b
let c = 3 * 4 **in** c + 1	13
let i = 10-i **in** i	5

W ostatnim przykładzie dla ułatwienia użyto równania algebraicznego. W rzeczywistości tak rozumiane równanie nie może tu wystąpić.

W regule (5.9) występują dwa zagnieżdżone jedno w drugim wyrażenia wiążące (w wersji 1). Ich użycie odpowiada intuicyjnemu rozumieniu obliczania wartości wyrażenia dwuargumentowego — najpierw oblicza się wartości obu argumentów, a następnie wykonuje na nich odpowiednie działanie. Kolejność obliczania wartości argumentów nie ma w naszym przypadku znaczenia, gdyż obie są obliczane dla tego samego stanu (nie ma efektów ubocznych).

W regule (5.7) występuje wersja 2 wyrażenia wiążącego. Dla wygody i przejrzystości zapisano je w postaci „**let** f (s) = ..." zamiast „**let** f = ..." (wartość stałej f jest funkcją).

Zauważmy najpierw, że reguła (5.7) odpowiada intuicyjnemu rozumieniu instrukcji pętli. Znaczeniem tej instrukcji jest takie przekształcenie stanów *f*, że jeżeli dla stanu *s* wartością warunku *wyr* jest FALSE, to stan ten pozostaje nie zmieniony, natomiast w przeciwnym wypadku zostaje on poddany przekształceniu będącemu efektem instrukcji *instr*, a następnie znowu przekształceniu *f*.
Mówiąc inaczej, równanie z reguły (5.7) wyraża fakt, że instrukcja pętli

while WY **do** IN

ma to samo znaczenie, co instrukcja

if WY **then begin** IN; **while** WY **do** IN **end**.

Równoważność ta wystarcza, aby jednoznacznie to znaczenie określić.

[1]) Wyjaśnienie to jest nieścisłe dla pewnych przypadków zagnieżdżonych wyrażeń wiążących

Rozwiązania równania z reguły (5.7) nie da się przedstawić w postaci skończonego wyrażenia. Używając skrótów *e* dla *e-wyr(wyr)* oraz *i* dla *i-instr(instr)* można to rozwiązanie przedstawić jako funkcję przyporządkowującą stanowi *s* stan:

if ¬e(s) **then** s **else**

if ¬e°i(s) **then** i(s) **else**

...

if ¬e°i^n (s) **then** i^n(s) **else**

(i^n oznacza n-krotne złożenie funkcji *i*).

Wartościami funkcji semantycznych są funkcje. Zauważmy, że są możliwe dwa style ich definiowania. Prawe strony reguł (5.5), (5.6), (5.9), (5.10) są wartościami wartości funkcji semantycznych dla określonego stanu s. Prawe strony reguł (5.1), (5.3), (5.4), (5.7) opisują natomiast „całą funkcję naraz".

W zrozumieniu przykładowej definicji może pomóc Czytelnikowi przykład P4 pokazujący, jak znaczenie programu jest uzależnione od znaczeń jego elementów. Rozwlekłość przykładu wynika z chęci szczegółowego przedstawienia wszystkich przekształceń.

Przykład P4. Semantyka programu P2

Oznaczmy wyrażenie dwuargumentowe zawarte w P2 przez *wb* oraz instrukcję podstawienia z tego programu przez *ip*. Z (5.1) i (5.3) mamy

i-program(P2) = i-instr-podst(ip)°i-instr-pusta(mk-instr-pusta(**SKIP**)) =
i-instr-podst(ip)°I = i-instr-podst(ip) = i-instr-podst(mk-instr-podst(A,wb)). (*)
Niech sEΣ będzie stanem. Wtedy wartość wyrażenia *wb* dla tego stanu jest
równa e-wyr(wb)(s) = e-wyr-2-a(wb)(s) = e-wyr-2-a(mk-wyr-2-a(A,**PLUS**,A)) (s)
= let w1 = e-wyr(A)(s) in let w2 = e-wyr(A)(s) in (w1+w2),
a ponieważ e-wyr(A)(s) = e-id(A)(s) = s(A), więc ostatecznie otrzymujemy
e-wyr(wb)(s) = s(A) + s(A) = 2*s(A).
Z reguły (5.5)
i-instr-podst(mk-instr-podst(A,wb))(s)
= s + [A → e-wyr(wb)(s)] = s + [A → 2*s(A)] (**)
Z (*) i (**) otrzymujemy zależność wiążącą stan początkowy ze stanem po wykonaniu programu P2: i-program(P2)(s) = s + [A → 2*s(A)].
i-program(P2) jest funkcją przyporządkowującą stanowi s stan s + [A → 2*s(A)], różniący się od s tylko wartością zmiennej A.

Zaletą metody denotacyjnej jest możliwość ścisłego i nie pozostawiającego wątpliwości zdefiniowania semantyki języka. Dzięki zastosowaniu obiektów matematycznych definicja taka abstrahuje od cech konkretnych komputerów czy też konkretnych implementacji i dlatego jest dogodnym narzędziem do standaryzacji języków.

Szczupłość miejsca nie pozwala zaprezentować zastosowania przedstawionej metody do opisu semantyki struktury blokowej, instrukcji wejścia-wyjścia, tablic, rekordów, plików i innych struktur danych, wskaźników (ang. *pointers*), efektów ubocznych i skoków. Jedynie do opisu procesów współbieżnych konieczne jest wyjście poza ramy semantyki denotacyjnej.

Ograniczono się też do bardzo skromnej części metajęzyka Drugiej Metody Wiedeńskiej. Należy zaznaczyć, że nie jest on systemem zamkniętym i dopuszczalne są wszelkie potrzebne rozszerzenia. W wersji przedstawionej w [5] metajęzyk jest bardzo, może nawet nadmiernie, rozbudowany; zawiera wiele elementów mających charakter makroinstrukcji i dających się sprowadzić do niewielkiego zbioru konstrukcji podstawowych. Szczególnie duże znaczenie mają skróty służące do opisu semantyki efektów ubocznych i skoków. Dzięki nim wzbogacenie języka o efekty uboczne i skoki pociąga za sobą tylko nieznaczny wzrost objętości definicji jego semantyki, chociaż w istocie semantykę tę w zasadniczy sposób komplikuje. Dużą zaletą metajęzyka jest oparcie go na powszechnie znanej symbolice matematycznej i notacji ALGOLU.

Druga Metoda Wiedeńska jest pierwszą metodą o znaczeniu praktycznym, umożliwiającą tworzenie definicji semantyki skomplikowanych obiektów. Wzbogaciła ona semantykę denotacyjną o wygodny metajęzyk umożliwiający czytelny i przejrzysty zapis, o wiele wygodniejszy w praktycznych zastosowaniach od notacji użytej przez twórców semantyki denotacyjnej. W zastosowaniu do definiowania języków programowania podstawową wadą Drugiej Metody Wiedeńskiej jest pominięcie składni konkretnej. Wydaje się jednak, że uzupełnienie tego braku, w sposób zgodny z duchem tej metody i oparty na jej metajęzyku nie byłoby trudne.

Szukających dalszych informacji odesłać można do książki [5] zawierającej między innymi liczne przykłady i wykaz literatury, oraz do [1]. Praca [2] przedstawia zastosowanie Drugiej Metody Wiedeńskiej do systematycznej konstrukcji translatora języka programowania. Przykład użycia Drugiej Metody Wiedeńskiej do opisu bazy danych można znaleźć w [1].

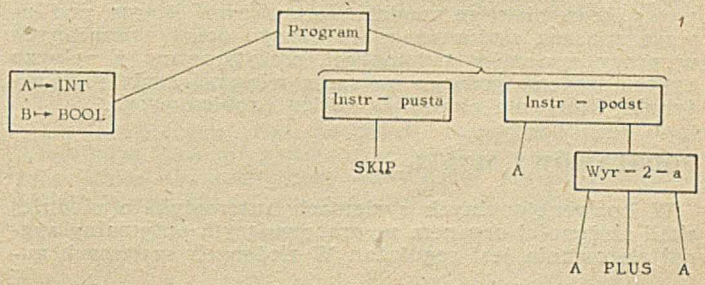
Tabela 2. Składnia abstrakcyjna przykładowego języka (opisana notacją Drugiej Metody Wiedeńskiej)

Program : : Dek Instr*	(2.1)
Dek = Id $\xrightarrow[m]{}$ Typ	(2.2)
Typ = **BOOL** | **INT**	(2.3)
Instr = Instr-pusta | Instr-złożona | Instr-podst | Instr-war | Instr-pętli	(2.4)
Instr-pusta : : **SKIP**	(2.5)
Instr-złożona : : Instr*	(2.6)
Instr-podst : : Id Wyr	(2.7)
Instr-war : : Wyr Instr Instr	(2.8)
Instr-pętli : : Wyr Instr	(2.9)
Wyr = Wyr-2-a | Id	(2.10)
Wyr-2-a : : Wyr Op Wyr	(2.11)
Op = **PLUS** | **EQ**	(2.12)

Wyjaśnienie skrótów:

Dek — deklaracje
Id — identyfikator
Instr — instrukcja
podst — podstawienia
war — warunkowa
Wyr — wyrażenie
2-a — dwuargumentowe
Op — operator
SKIP — ang. pomin.
Nazwami są wyrazy w liczbie pojedynczej, ale dotyczą nie pojedynczych obiektów, lecz ich zbiorów.

Przykład P2. Program abstrakcyjny odpowiadający programowi P1



LITERATURA

[1] „Abstract Software Specifications", Proceedings, 1979, Lecture Notes in Computer Science, Springer-Verlag 1980

[2] Bjørner D.: Programming Languages: Formal Development of Interpreters and Compilers, „International Computing Symposium 77", European ACM, North Holland 1977

[3] Łukaszewicz L.: Outline of the Vienna Method for the formal definition of programming languages, COPAN, Warszawa 1973

[4] Tennent R. D.: The Denotational Semantics of Programming Languages, Comm. ACM, vol. 19, nr 8, August 1976

[5] „The Vienna Development Method: The Meta-Language", edited by D. Bjørner and C. B. Jones, Lecture Notes in Computer Science nr 61, Springer-Verlag 1978

[6] Wegner P.: The Vienna Definition Language, ACM Computing Surveys, vol. 4, nr 1, March 1972.

TADEUSZ PUCHAŁKA
Politechnika Poznańska

Zadania i algorytmy komputerowego sterowania procesami technologicznymi

Automatyka jako samodzielna dyscyplina naukowa narodziła się mniej więcej przed trzydziestu laty. Dziś dysponuje własnym ogólnym aparatem pojęciowym oraz dość dobrze rozwiniętą teorią, zwaną teorią sterowania. W pierwszym okresie rozwoju automatyki dominowały badania dynamiki szczegółowych struktur, w dniu dzisiejszym akcenty przesunęły się w kierunku automatyzacji sterowania złożonymi procesami technologicznymi. Uzasadnienia tego faktu trzeba szukać m.in. w bardzo intensywnym rozwoju środków automatyzacji o coraz większych możliwościach, w szczególności zaś — komputerów.

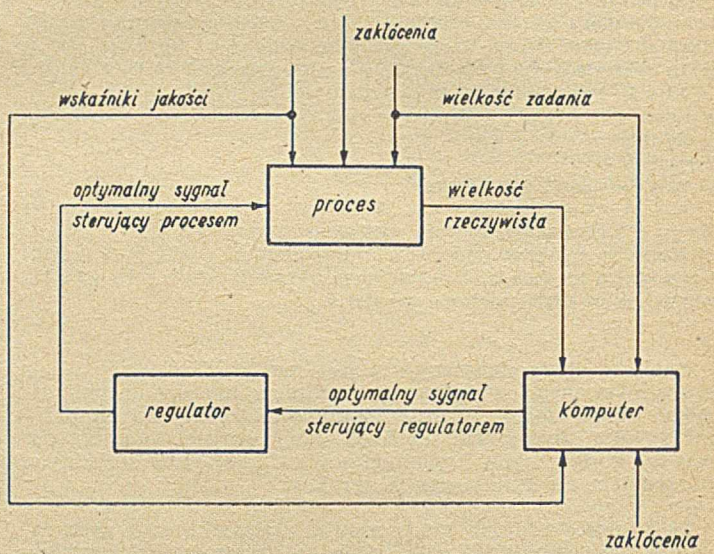
Wprowadzenie automatyzacji procesów technologicznych oraz poziom jej strukturalnej realizacji winien być każdorazowo umotywowany względami natury ekonomicznej. Współczesne systemy automatyki odznaczają się bowiem dość dużą złożonością i dużym zróżnicowaniem struktur, ich wykorzystanie nie w każdym przypadku bywa opłacalne.

Systemy automatyki, jeśli je rozpatrywać w skali rzeczywistej, nierzadko ujawniają jednocześnie trzy właściwości: indeterminizm, nieciągłość i nielinowość. Jest to zresztą przypadek najogólniejszy. Procesy (obiekty) mają charakter wielowymiarowy, zaś systemy sterowania (część sterująca systemu automatyki) wyróżniają się budową wielopoziomową, od warstwy sterowania bezpośredniego począwszy, na warstwie ewolucji kończąc.

W niniejszym artykule podjęto problematykę zadań i algorytmów komputerowego sterowania procesami technologicznymi (KSPT). Punktem wyjścia jest model podstawowy oraz niektóre modele szczegółowe. Biorąc pod uwagę znaczną złożoność analitycznego opisu rozważanych w pracy systemów automatyki, zrezygnowano z pełnego rygoru matematycznego na rzecz większej czytelności artykułu, który ma charakter szkicu problemowego.

PODSTAWOWY MODEL

W konwencjonalnych systemach automatyki o niedużej skali złożoności organem wypracowującym optymalny sygnał sterowania jest regulator. W złożonych systemach automatyki rolę tę przejmuje komputer (ewentualnie mini- lub mikrokomputer), względnie jego odpowiednio zagregowane zespoły. Sytuacja taka zobrazowana jest na rysunku 1. Komputer wypracowuje optymalny sygnał sterujący regulatorem na podstawie szczegółowych programów, stanowiących realizację algorytmu optymalnego sterowania procesem. Ów algorytm winien uwzględniać: informacje o procesie, informacje o zakłóceniach oraz wskaźniki jakości (zadania, funkcje celu — najczęściej postaci funkcjonałowej) jakie winien spełniać proces.



Rys. 1. Schemat blokowy podstawowego modelu komputerowego sterowania procesem technologicznym

Właściwe projektowanie KSPT obejmuje trzy etapy:

- analizę procesu technologicznego, analizę sygnałów użytecznych i zakłóceń, analizę ekonomiczną
- konstrukcję algorytmu optymalnego sterowania, w szczególnych przypadkach algorytmu nadrzędnego i algorytmów szczegółowych
- syntezę struktury systemu sterowania oraz konstrukcję programów realizujących algorytm sterowania optymalnego.

W zakres etapu pierwszego wchodzą w szczególności procedury identyfikacyjne. Konstrukcja algorytmu optymalnego sterowania — etap drugi — może mieć miejsce po określeniu wskaźnika jakości lub jego zbioru (o charakterze bezwarunkowym lub warunkowym). Bezpośrednią konsekwencją algorytmu optymalnego sterowania są określone struktury procedur optymalizacyjnych (optymalizacja parametryczna lub strukturalna). W ramach trzeciego etapu dokonujemy syntezy szczegółowej struktury systemu sterowania i konstytuujących go elementów. W szczególności przesądzony zostaje ustrój i organizacja poszczególnych poziomów systemu.

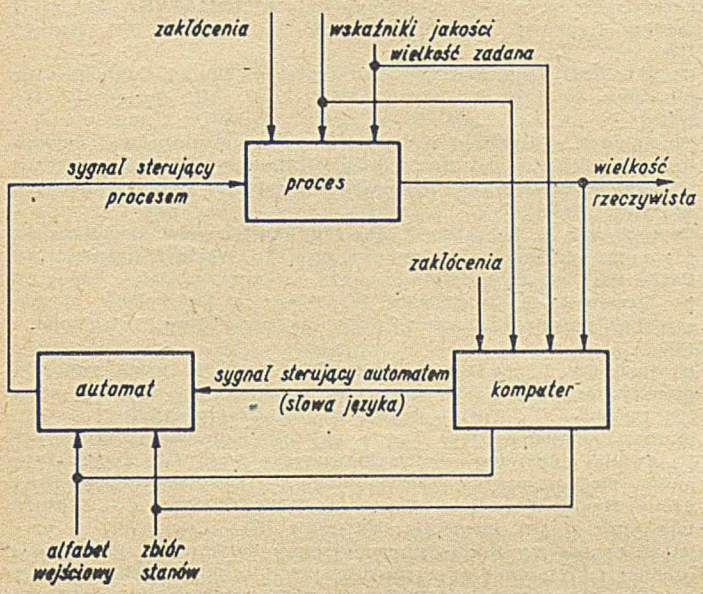
Prof. dr Tadeusz PUCHAŁKA — członek międzynarodowych i krajowych towarzystw i organizacji naukowych — ukończył Wydział Elektromechaniczny Akademii Górniczo-Hutniczej w Krakowie. Jest profesorem zwyczajnym w Politechnice Poznańskiej (Wydział Elektryczny). Prowadzi badania naukowe w dziedzinie teorii i zastosowań systemów sterowania oraz teorii systemów przełączających (98 publikacji). Współpracuje z przemysłem wielkopolskim w zakresie systemów automatyki oraz automatyzacji procesów technologicznych.

MODELE SZCZEGÓŁOWE

Przedmiotem rozważań będą dwa modele KSPT — automatowy oraz typu: kompleks operacji. W pierwszym modelu rolę regulatora pełni automat — deterministyczny bądź stochastyczny, w drugim — proces scharakteryzowany jest przez strukturę typu: kompleks operacji.

Model automatowy

Idea modelu automatowego KSPT zilustrowana jest na rysunku 2. Zakładamy, że automaty pełniące funkcję regulatora należą do klasy automatów Moore'a.



Rys. 2. Schemat blokowy modelu automatowego komputerowego sterowania procesem technologicznym

Automat deterministyczny (A_d) zdefiniowany jest następująco:

$$A_d = \langle S, X, Y, F, \delta, \lambda, s_o \rangle$$

gdzie:

S — niepusty skończony zbiór stanów
X — niepusty skończony alfabet wejściowy
Y — niepusty skończony alfabet wyjściowy
F — zbiór dopuszczalnych stanów końcowych, $F \leqslant S$
δ — funkcja przejść, δ: $(SxX)_t \rightarrow S_{t+1}$
λ — funkcja wyjść, λ: $(S)_t \rightarrow (Y)_t$
s_o — stan początkowy.

Zadanie komputera jest podwójne. W pierwszym rzędzie w oparciu o wszystkie przesłanki wyjściowe ma on wygenerować drogą odpowiedniej procedury algorytmicznej język regularny — $L_r(X)$ — utworzony nad alfabetem wejściowym X automatu. Drugie zadanie wiąże się z syntezą automatu reprezentującego język $L_r(X)$, a dokładniej mówiąc — z konstrukcją funkcji $\bar{\delta}$, w takim sensie, aby:

$$x^* \in L_r \rightarrow \bar{\delta}(s_o, x^*) \in F$$

gdzie:

x^* — słowo utworzone nad alfabetem X
$\bar{\delta}$ — uogólniona funkcja przejść, określona odpowiednio: $\bar{\delta}(s, \Lambda) = s$, przy czym: $s \in S$, Λ — słowo puste, $\bar{\delta}(s, x_1^* x) = \delta(\bar{\delta}(s, x_1^*), x)$.

Szczegółowa procedura konstrukcji automatu reprezentującego język regularny L_r objęta jest tzw. centralnym twierdzeniem syntezy [10].

Odpowiednio automat stochastyczny (A_s) definiujemy jako system algebraiczny:

$$A_s = \langle S, X, \{M(x)\}, F, \pi_o \rangle$$

gdzie:

S, X — oznaczenia jak w poprzedniej definicji
$M(x)$ — zbiór stochastycznych macierzy przejść
F — zbiór dopuszczalnych stanów końcowych
π_o — stochastyczny stan początkowy.

Wiadomo, iż prawdopodobieństwo osiągnięcia któregoś ze stanów należących do zbioru F, określone jest zależnością:

$$m(x^*) = \pi_o M(x^*) \eta_F$$

gdzie:

$$M(x^*) = M(x_1)M(x_2)...M(x_k), \quad dla \quad x^* = x_1 x_2 ... x_k$$

$$F = \begin{Vmatrix} 0 \\ 1 \\ \cdot \\ \cdot \\ 0 \end{Vmatrix}_{nx1}$$

przy czym:

$$\eta_F^i = \begin{cases} 1, & \text{gdy } s_i \in F \\ 0, & \text{gdy } s_i \in F \end{cases} \quad \text{dla } i = 1, \ldots n, \; n = (S).$$

Językiem reprezentowanym przez A_s z poziomem wiarygodności

$$(0 \leqslant \lambda < 1) — L(A_s, \lambda) \text{ — nazywamy zbiór:}$$
$$L(A_s, \lambda) = \{x^* | x^* \in X^*, \; m(x^*) > \lambda\}$$

gdzie: X^* — oznacza słownik.

Głównym zadaniem komputera jest wypracowanie w procedurze heurystycznej algorytmy maksymalizacji prawdopodobieństwa $m(x^*)$. Z punktu widzenia syntezy automatu stochastycznego (regulatora stochastycznego) oznacza to określenie w ramach tej procedury takiego ciągu stochastycznych macierzy przejść:

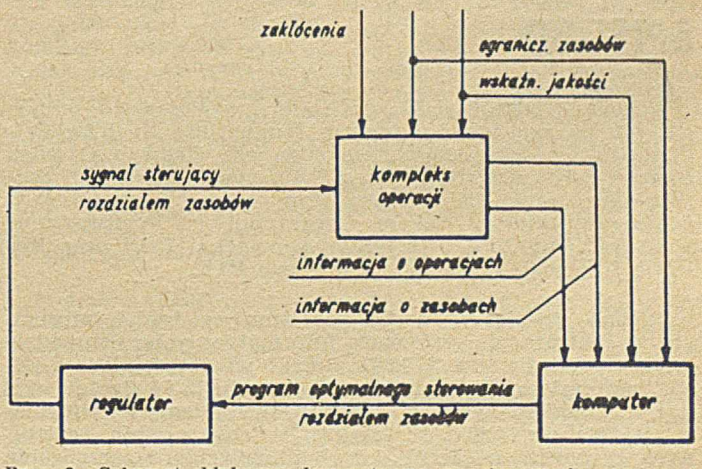
$$(M(x_1), M(x_2)..., M(x_k)),$$

który gwarantuje maksymalizację $m(x^*)$.

Syntezę strukturalną automatu stochastycznego możemy — wykorzystując znane algorytmy [6, 10] — przeprowadzić w oparciu o automat deterministyczny i źródła Markowa rzędu 0.

Model typu: kompleks operacji

Dla niektórych klas procesów technologicznych najwłaściwsze jest przyjęcie modelu typu: kompleks operacji. Idea komputerowego sterowania takimi procesami technologicznymi zobrazowana jest na rysunku 3.



Rys. 3. Schemat blokowy komputerowego sterowania procesem typu: kompleks operacji

Formalnie przez kompleks operacji (*KO*) rozumiemy następujący system algebraiczny:

$$KO = \langle O, Z \rangle$$

gdzie:

O — zbiór operacji
Z — zbiór zasobów.

Zbiór operacji określają: informacja o liczbie operacji oraz informacja o momentach rozpoczynania się operacji. Operację charakteryzują zaś żądania dotyczące zasobów, model matematyczny i ustalenie priorytetu. O zbiorze zasobów decyduje rodzaj jednostek zasobu oraz ich liczba (dla każdego rodzaju). Jednostka zasobu scharakteryzowana jest natomiast przez dopuszczalne obciążenie określające liczbę operacji, które mogą być wykonywane jednocześnie przez tę jednostkę, a także przez informację dotyczącą magazynowania wyprodukowanych wyrobów (w przypadku operacji o charakterze produkcyjnym).

Najbardziej ogólna sytuacja ma miejsce wówczas, gdy model kompleksu operacji ma strukturę indeterministyczną oraz operacje są czasowo uwarunkowane.

Szczególnym przypadkiem rozdziału zasobów jest rozdział zadań (każda operacja może być związana, w każdej chwili, tylko z jedną jednostką zasobu wyłącznie jednego rodzaju).

Optymalne sterowanie rozdziałem zasobów dokonywane jest w oparciu o szczegółowe algorytmy, których konstrukcja wiąże się ściśle z wykorzystaniem zasad i metod programowania matematycznego.

Obszerna klasa zastosowań wiąże się z minimalno-czasowym sterowaniem rozdziałem zasobów [14].

* * *

Opisane ogólnie dwa modele komputerowego sterowania procesami technologicznymi nie wyczerpują oczywiście wszystkich możliwości rozwiązań strukturalnych. Konsekwencją przyjęcia określonego rozwiązania strukturalnego jest zróżnicowanie szczegółowych algorytmów (programów) wypracowywanych przez komputer i wykorzystywanych do sterowania regulatorem mającym konkretną konfigurację.

Model automatowy przykładowo może znaleźć aplikacje w komputerowym sterowaniu niektórymi procesami hutniczymi, np. w walcowniach [4].

Model procesu o charakterze „kompleks operacji" może być wykorzystany w opisie zarówno samego procesu magazynowania, jak i sterowania rozdziałem zasobów, dokładnie z nim związanych. Niektóre idee dotyczące koncepcji modelowych oraz opisu sterowania procesami magazynowania zostały wyłożone m.in. w pracy [12].

LITERATURA

[1] Box G. E.: Evolutionary operation: A method for increasing industrial produktivity, Applied Statistics, vol. VI, no 2, 1957
[2] Burkov V. N.: Raspredelenie resursov kak zadača optimalnego bystrodeistvija, Avtomatika i Telemechanika 27, no 7, 1966
[3] Findeisen W.: Wielopoziomowe układy sterowania, PWN, Warszawa, 1974
[4] Gościński A.: Komputerowe systemy sterowania złożonymi dyskretnymi procesami produkcyjnymi, AUTOMATYKA, zeszyt 14, Zeszyty Naukowe Akademii Górniczo-Hutniczej (nr 575), Kraków 1976
[5] IEAC/IFIP, International Conference on Digital Computer Applications to Process Control, Part I, II, III. Springer-Verlag, 1974
[6] Knast R.: O pewnej możliwości syntezy strukturalnej automatu probabilistycznego, Prace Komisji Budowy Maszyn i Elektrotechniki Poznańskiego Towarzystwa Przyjaciół Nauk, t. I, zeszyt 5, Poznań, 1967
[7] Lefkovitz I., Schoefler J. D.: Multilevel control structures for three discrete manufacturing processes, 5 World Congress IFAC, Paris, Part 3a, 26. 1, 1972
[8] Mesarovič M. D., Macko D., Takahara Y.: Theory of hierarchical multilevel systems, Academic Press, New-York, London, 1970
[9] Puchałka T.: O stanie i kierunkach rozwoju teorii systemów przełączających, INFORMATYKA nr 4/1975
[10] Puchałka T.: Teoria systemów przełączających; część I: Preliminaria matematyczne, Teoria układów kombinacyjnych; część II: Teoria automatów zdeterminowanych; część III: Teoria automatów stochastycznych, Teoria automatów złożonych; Polska Akademia Nauk, Oddział w Poznaniu, PWN, Warszawa — Poznań, 1976 (część I), 1978 (część II), 1980 (część III)
[11] Puchałka T.: Komputerowe systemy automatyki. Podstawowe problemy i kierunki prac rozwojowych i praktycznych, Zbiór referatów: Informatyka w Wielkopolsce, Naczelna Organizacja Techniczna, Oddział Wojewódzki w Poznaniu, Komitet Naukowo-Techniczny ds. Informatyki, Poznań, 1978
[12] Puchałka T., Loręcki M.: Praktyczne możliwości stosowania automatyki w organizacji i technologii procesów magazynowych. Towarzystwo Naukowe Organizacji i Kierownictwa, Oddział w Poznaniu, Instytut Gospodarki Magazynowej w Poznaniu, Poznań, 1979
[13] Shinsky F. G.: Process-Control systems, Mc Graw-Hill Book Company, New York, 1971
[14] Węglarz J.: Minimalno-czasowe sterowanie rozdziałem zadań i zasobów w kompleksie operacji w warunkach deterministycznych, Politechnika Poznańska, Rozprawy nr 78, Poznań, 1976.

SPIS' 80

W dniach 14—16 października br. w Puławach odbyło się już po raz trzeci doroczne seminarium SPIS'80. Inicjatorem tej serii seminariów jest jak wiadomo Ośrodek Badawczo-Rozwojowy Systemu Państwowej Informacji Statystycznej przy GUS, który tym razem zaprosił jako współorganizatorów Sekcję Organizacji i Przetwarzania Danych Komitetu Statystyki i Ekonometrii PAN oraz Komisję Informatyki Zarządu Głównego PTE.

SPIS'80 poświęcony został problematyce źródeł informacji w centralnych systemach informatycznych. Poprzednie seminaria, zwłaszcza SPIS'78 (temat wiodący „SPIS a resortowe systemy informatyczne") oraz SPIS'79 (temat wiodący „Banki danych w centralnych systemach informatycznych") wykazały, że istotną barierą rozwoju centralnych systemów informatycznych planowania i zarządzania oraz statystyki są trudności związane z gromadzeniem zasobów informacyjnych, a zwłaszcza z zasilaniem baz danych tych systemów. Podczas obrad plenarnych tegorocznego seminarium zaprezentowano koncepcję źródeł danych oraz instrumenty ich koordynacji. Obrady oraz towarzyszące im dyskusje w sekcjach podzielono na następujące cztery zespoły tematyczne:

- System informatyczny rachunkowości i ewidencja podstawowa jako źródła zasilania CSI
- Kontrola i korekta danych źródłowych
- Problemy rejestracji danych źródłowych
- Źródła danych w systemach resortowych i regionalnych.

Ze względu na szczególnie interesującą tematykę postaramy się w następnych numerach INFORMATYKI opublikować najciekawsze wystąpienia spośród kilkunastu referatów tego seminarium.

JAN BUJKO, ZBIGNIEW STYCZYŃSKI
Instytut Energetyki
Politechnika Wrocławska

Zasady archiwizacji w systemie projektowania wspomaganego komputerem

Archwizacja dokumentacji projektowej stanowi jeden z istotnych problemów biur i organizacji projektowych. Rosnąca objętość dokumentacji oraz liczba wykonywanych projektów stwarza bariery fizycznego ich gromadzenia. Tworzenie systemów wspomaganego komputerem projektowania stwarza szansę zmiany koncepcji przechowywania dokumentacji, tj. przejścia na archiwizację opartą o pamięci masowe maszyn cyfrowych.

Nie bez znaczenia jest również fakt, że dokumentacja uzyskana z realizacji projektu w systemie komputerowym zawiera zazwyczaj większą liczbę wariantowych (porównywalnych) rozwiązań, a więc jest obszerniejsza od tradycyjnej. Pomijając problem zmiany samej zawartości dokumentacji projetkowej, który nie jest przedmiottem niniejszego artykułu, zagadnienie i tak pozostaje ciekawe z punktu widzenia informatyczno-organizacyjnego.

W przedstawionym artykule rozważone będą następujące problemy archiwizacji dokumentacji projektowej:
- zakres przechowywanej dokumentacji w zależności od stopnia zautomatyzowania procesu projektowego
- forma przechowywania informacji
- miejsce przechowywania dokumentacji w systemie projektowania wspomaganego.

W artykule opisana jest archiwizacja dokumentacji przy projektowaniu wspomaganym komputerem ODRA 1305, pracującym pod kontrolą systemu operacyjnego GEORGE 3. Do obsługi kartoteki proponowane jest wykorzystanie makroinstrukcji języka sterowania pracami systemu GEORGE.

Identyfikacja procesu projektowego ze względu na sposób gromadzenia dokumentacji

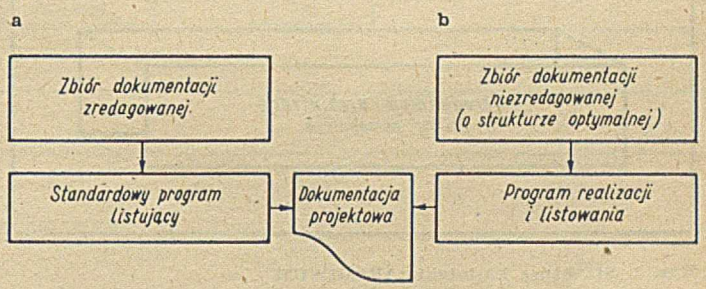
Komputerowy system projektowania inwestycyjnego charakteryzuje się zarówno dużą liczbą algorytmów jak i zbiorów wyników pośredniczących [1, 3]. W zależności od stopnia zautomatyzowania prac projektowych zbiory pośredniczące zawierają rozwiązania problemów projektowych — wymagające przetwarzania przez projektanta do formy wymaganej dokumentacji, rozwiązania cząstkowe projektu — przenoszone w utworzonej formie do dokumentacji, bądź też pełną dokumentację — nie wymagającą dodatkowego przetwarzania (poza maszyną cyfrową). Ta ostatnia forma występuje w systemach charakteryzujących się dużym stopniem automatyzacji prac projektowych. W praktyce systemy takie są trudne do realizacji ze względu na kłopoty w sformalizowaniu niektórych algorytmów heurystycznych oraz kryteriów nieparametrycznych.

W procesie projektowania komputerowego istotne jest określenie zakresu przechowywania informacji. W niektórych przypadkach można przechowywać dane wejściowe (szczególnie w systemach o wysokim stopniu zautomatyzowania) i — w momencie wystąpienia konieczności uzyskania kopii dokumentacji — powtórzyć obliczenia. Sposób ten mógłby być uzasadniony, jeżeli koszt ponownego przetwarzania byłby porównywalny z kosztem przechowywania całej dokumentacji na nośnikach magnetycznych.

Przechowywanie jednak samych danych stanowi niebezpieczeństwo, iż niekiedy nie uda się odtworzyć wyników. Powodem tego może być otwartość, którą charakteryzują się systemy projektowania wspomaganego.

Wprowadzane zmiany w programach obliczeniowych mogą doprowadzić do zmiany formy wyników lub niekompatybilności zbiorów danych z systemem. Dlatego też niezbędne jest przechowywanie (archiwizowanie) zarówno danych jak i wyników obliczeń. Z podobnych względów wydaje się uzasadnione przechowywanie ich w formie zredagowanej, gotowej do natychmiastowego kopiowania.

Formą prowadzącą do mniejszego zapotrzebowania pamięci dla zbiorów archiwizowanych byłoby utworzenie optymalnych struktur zbiorów. Wymagałyby one jednak programów redagujących, które w systemie otwartym również mogą ulegać zmianom, a więc zbiory archiwizowane mogłyby stać się niekompatybilne z systemem. Te dwie formy zbiorów oraz zasady tworzenia kopii dokumentacji w nich zawartych przedstawiono na rys. 1.



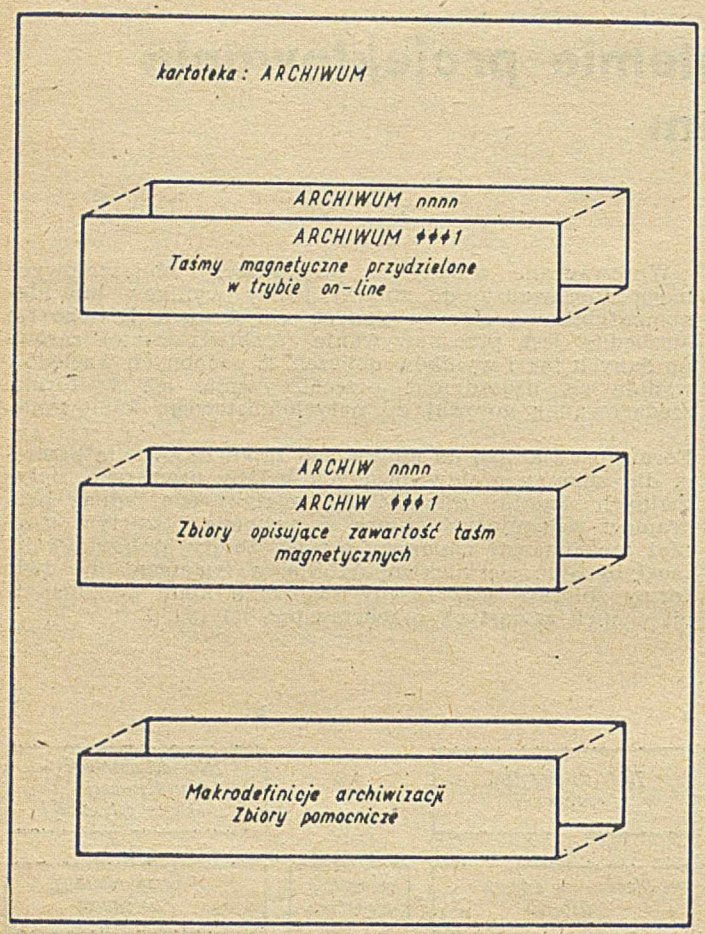
Rys. 1. Zasady tworzenia kopii dokumentacji projektowej w oparciu o zbiory dokumentacji: a) zredagowanej, b) zoptymalizowanej ze względu na wymaganą pamięć

Problemem o dużym znaczeniu organizacyjnym i ekonomicznym jest miejsce przechowywania zbiorów archiwizowanych. Przyjmując strukturę systemu projektowania wspomaganego jak w [2], można by założyć, iż zbiory archiwizowane mogłyby być przechowywane w kartotekach :PROJEKTANTnn. Kartoteki te jednak są tworzone dla określonych problemów i po ich wykonaniu powinny być kasowane. Dlatego też na poziomie kartotek :PROJEKTANTnn tworzy się kartotekę o nazwie :ARCHIWUM. Informacje przeznaczone do zarchiwizowania są przekazywane do niej i przechowywane w postaci zbiorów na taśmach magnetycznych, co znacznie obniża koszty przechowywania w porównaniu ze zbiormi PZS.

Organizacja kartoteki :ARCHIWUM i procesu archiwizacji

Podane założenia prowadzą do organizacji kartoteki :ARCHIWUM. Charakteryzuje się ona wysokim priorytetem SPACEMT, co zapewnia przydzielenie jej dużej puli taśm. Informacje o dokumentacjach przechowywanych na taśmach magnetycznych znajdują się w zbiorach o nazwach ARCHIWnnnn, gdzie nnnn — oznacza numer seryjny taśmy, której zawartość opisuje zbiór. Dodatkowym elementem kartoteki są makrodefinicje archiwizacji i odzyskiwania dokumentacji. Strukturę kartoteki :ARCHIWUM pokazano na rys. 2.

Dokumentacja archiwizowana stanowi zbiory typu GRAPHIC. Zapisuje się ją i odczytuje z taśm magnetycznych, korzystając ze standardowych makrodefinicji, znajdujących się w kartotece :MACROS (COPYOUT i COPYIN), wykorzystujących programy #XKYB i #XKYC.



Rys. 2. Struktura kartoteki :ARCHIWUM

Czynności prowadzące do przekazania zbioru przeznaczonego do kartoteki :ARCHIWUM i jego przechowania na taśmie magnetycznej obejmują:

- utworzenie zbioru wyników końcowych
- scalanie zbioru wyników końcowych ze zbiorem danych
- udostępnienie zbioru przeznaczonego do archiwizacji dla kartoteki :ARCHIWUM
- wypełnienie karty zgłoszenia zbioru do archiwizacji
- przekazanie karty zgłoszenia do dyspozytora kartoteki :ARCHIWUM
- uzupełnienie i wyperforowanie zadania dla kartoteki :ARCHIWUM
- wykonanie zadania archiwizacji w kartotece :ARCHIWUM.

W celu przygotowania zbioru do archiwizacji zbiory zredagowanych danych i wyników scala się poprzez uruchomienie makrodefinicji SCALANIE, której treść jest następująca:

```
MACDEF SCALANIE
CY %A, %B (APPEND)      {%A — zbiór zredagowanych danych
CY %B, %C               {%B — zbiór zredagowanych wyników
ER %A, %B               {%C — zbiór do zarchiwizowania
ENDM
```

Jeżeli zbiór wyników zawiera powtórzone dane wejściowe, nie wykonuje się przebiegu tej makrodefinicji. Następnie udostępnia się zbiory kartoteki :PROJEKTANTnn dla kartoteki :ARCHIWUM przez makrodefinicję DOSTEP, w której otwarte zostaną wszystkie tryby aktywności. Makrodefinicja ta ma postać:

```
MACDEF DOSTEP
TG %A,: ARCHIWUM, ALL          {%A, nazwa do zarchiwizowania
ENDM.
```

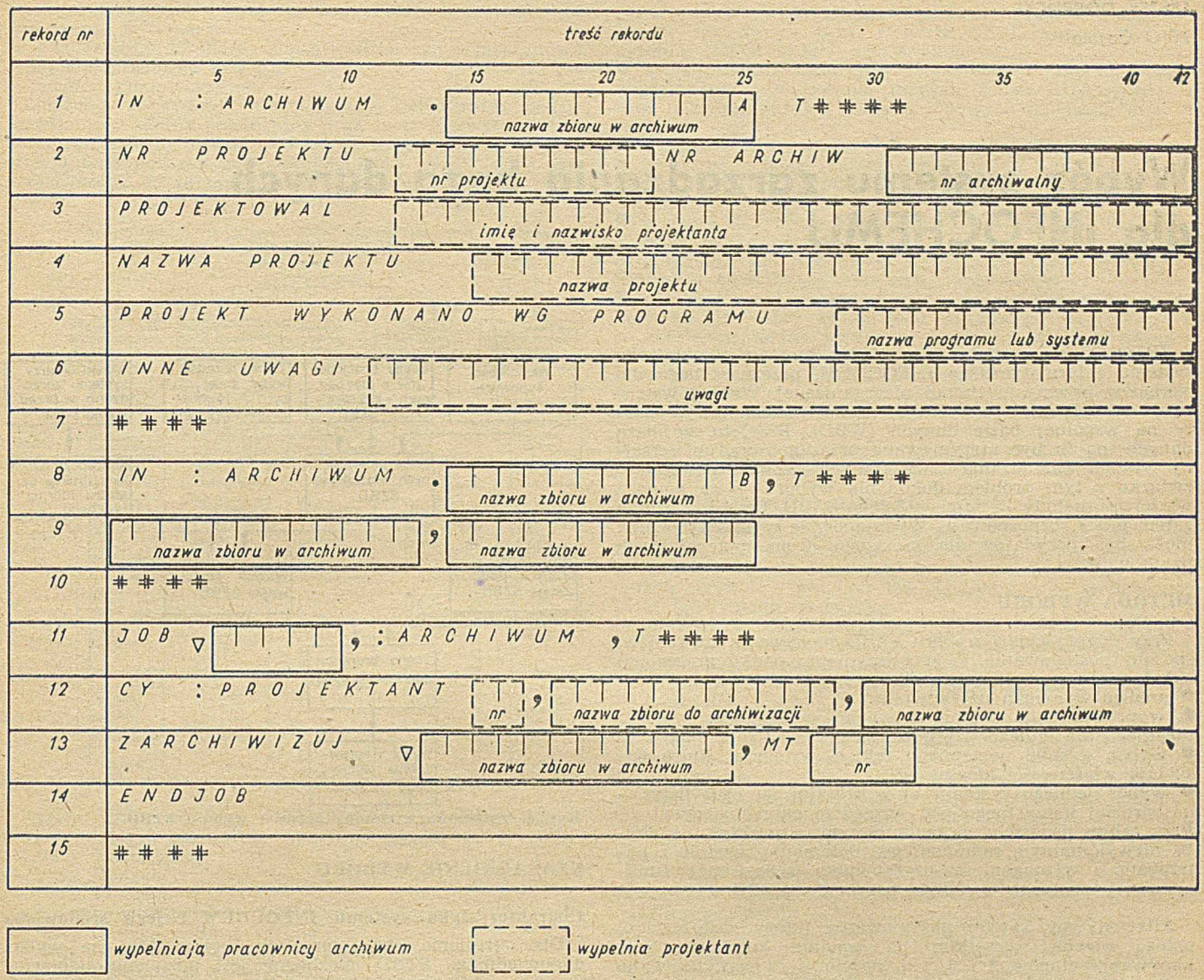
Makrodefinicję projektant wywołuje po zakończeniu obliczeń projektowych w swojej kartotece. Następnie wypełnia on kartę zgłoszenia, którą przedstawiono na rys. 3. Po przekazaniu tej karty do dyspozytora kartoteki :ARCHIWUM oraz przeniesieniu na dziurkowane karty zostaje wykonane zadanie przeniesienia zbioru — przeznaczonego do archiwizacji — z kartoteki :PROJEKTANTnn do kartoteki :ARCHIWUM, i zapisania go na taśmie magnetycznej. Zbiór ten zostanie skasowany w PZS, a informacja o nim zostanie zapisana w zbiorze opisującym zawartości taśm magnetycznych. Proces ten opisuje makrodefinicja ZARCHIWIZUJ, której treść przedstawiono poniżej:

```
MACDEF ZARCHIWIZUJ
DP Z, PROSZE MT % (MT)
CY %AA, ARCHIW % (MT) (APPEND)
# ODCZYT ZAWARTOSCI MT
COPYIN (%(MT)), * CR ZAWMT
IF NOT DELETED (OK), GO TO 1AR
CY ZAWARTOSC, ZAWARTOSC1
CY ZBIORGW, ZAWARTOSC1 (APPEND)
# ODCZYT ZBIOROW Z MT
COPYIN (%(MT)), * CR ZAWARTOSC1
ER ZAWARTOSC1
IF NOT DELETED (OK), GOTO 1AR
SP E, (%AB)
CY %E, ZAWARTOSC (APPEND)
CY ZAWARTOSC, ZAWARTOSC1
CY ZBIORGW, ZAWARTOSC1 (APPEND)
# ZAPIS NA MT
COPYOUT (%(MT)), CR ZAWARTOSC1
LF ZAWARTOSC1, NU
IF NOT DELETED (OK), GO TO 2AR
# KASOWANIE ZBIOROW
ER %AA, %AB
SP C, VALUE (Φ)
1AREP
LF ZAWARTOSC1, FR%C, LI1
SP D, REPLY (1, 12)
IF EXISTS (%D), (ER %D) ELSE (EXIT)
SP C, VALUE (%C + 1)
GO TO 1 AREP
1AR DP Φ, BLAD PRZY ODCZYCIE
DP Φ, POWTORZYC PRZEBIEG
EXIT
ZAR DP Φ, BLAD PRZY ZAPISIE
EXIT
ENDM
```

{%(MT) numer taśmy w :ARCHIWUM
%AA, %AB — zbiory pomocnicze o treści jak na rys. 3
ZAWMT — zbiór pozwalający na skopiowanie z MT podzbioru informacji o jej zawartości
ZBIORGW — zbiór ogranicznika makrodefinicji
COPYIN i COPYOUT

* * *

Archiwizowanie wyników prac projektowych wymaga w systemach wspomaganego komputerowo projektowania odmiennego podejścia. Stosowanie komputera do obliczeń projektowych pociąga za sobą możliwość przekazywania rozwiązań w pamięciach zewnętrznych. W związku z zaproponowaną strukturą informatyczną systemu [2] wydaje się, że stworzenie kartoteki :ARCHIWUM, przy centralnym jej zarządzaniu, stwarza możliwości przechowywania i kopiowania dokumentacji projektowej z pamięci zewnętrznych. Przechowywanie dokumentacji archiwizowanej na taśmach magnetycznych z wykorzystaniem standardowych — dla systemu GEORGE — makroinstrukcji zapewnia łatwość obsługi i niskie koszty przechowywania.



Rys. 3. Karta zgłoszenia zbioru do archiwizacji

LITERATURA

[1] Bujko J., Hejnowicz D., Styczyński Z.: Projektowanie urządzeń elektroenergetycznych. Ćwiczenia w laboratorium komputerowym, Skrypt Politechniki Wrocławskiej, Wrocław 1977

[2] Bujko J., Styczyński Z.: GEORGE 3 w projektowaniu inwestycyjnym, INFORMATYKA 6/1978

[3] Kujszczyk S., Nowoczesne metody obliczeń elektroenergetycznych sieci rozdzielczych, WNT, Warszawa, 1979 r.

Postulaty

Czym jest informatyka w polskich realiach, a czym mogłaby być, gdy nie... Gdyby nie co? Jakie decyzje, jakie fakty z niedawnej historii wpłynęły ujemnie na jej rozwój? Co trzeba by zmienić, co naprawić? Jakimi metodami należałoby działać?

Gorąca prośba! Piszcie do nas. Oczekujemy zwięzłych postulatów dotyczących organizacji informatyki w Polsce. Efekty takiej korespondencyjnej dyskusji przedstawimy na naszych łamach.

REDAKCJA

LESZEK DOBRUCKI
ZETO Gdynia

Wybór systemu zarządzania bazą danych dla INFOCHEMU

ZETO Gdynia podjęło się w 1975 r. budowy obiektowego systemu informatycznego INFOCHEM, przeznaczonego dla zakładów przemysłu chemii nieorganicznej. Według wstępnych założeń miał to być system wielodziedzinowy, oparty na wspólnej bazie danych (WBD). Przyjęto, że baza danych ma tu być utrzymywana przez gotowy, uniwersalny System Zarządzania Bazą Danych (SZBD). Powstał w związku z tym problem dokonania wyboru — za pomocą właściwej metody — odpowiedniego SZBD. Przeprowadzono zatem prace rozpoznawcze. Wydaje się, że wyniki tych prac mogą być przydatne dla szerszego grona potencjalnych użytkowników.

METODA WYBORU

Przy dokonywaniu wyboru SZBD możliwe są dwa różne sposoby postępowania. W pierwszym przypadku proces ten składa się z następujących etapów:

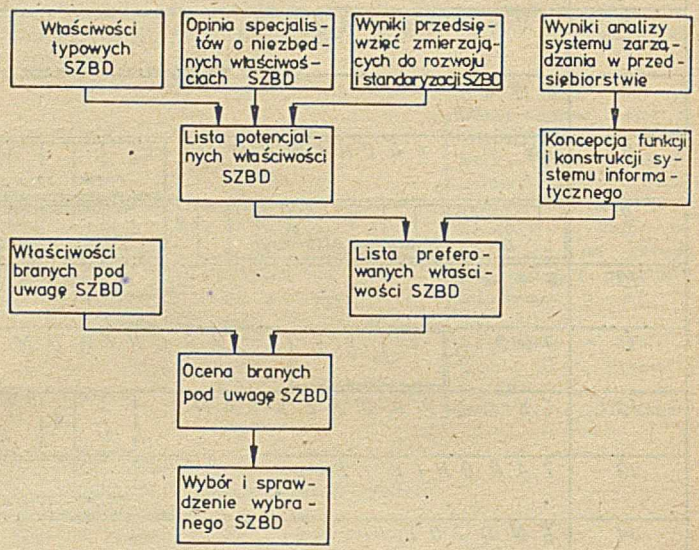
- analiza wymagań użytkownika
- wyprowadzenie z tych wymagań właściwości systemu idealnego dla danego użytkownika
- wybór spośród dostępnych takich systemów, które posiadają właściwości idealne
- wybór jednego systemu — z krótkiej na ogół listy — najbardziej korzystnego pod względem ekonomicznym.

Przy takim podejściu najtrudniejszym, a praktycznie chyba niewykonalnym zadaniem jest dokonanie, zgodnie z potrzebami i wymaganiami użytkownika, specyfikacji funkcjonalnej idealnego w danym przypadku SZBD.

Alternatywny sposób postępowania (który zastosowano) polega przede wszystkim na zmianie kolejności wykonywania etapów 2 i 3, tzn. najpierw są specyfikowane właściwości wybranych wstępnie SZBD, a następnie każda z tych właściwości oceniana jest przez porównanie jej z wymaganiami użytkownika. Przy tym podejściu proces wyboru SZBD składa się z następujących czterech etapów.

- **Określenie potrzeb i wymagań użytkownika,** identyfikowanych w trakcie analizy systemowej, która obejmuje analizę funkcji i analizę danych, oraz uściślanych podczas opracowywania założeń systemu
- **Specyfikacja potencjalnych właściwości SZBD (lista potencjalnych właściwości)** oraz wybór z tej listy tych właściwości SZBD, które są niezbędne lub pożyteczne dla projektanta systemu informatycznego **(lista preferowanych właściwości).** Lista właściwości preferowanych generowana jest przez porównanie listy potencjalnych właściwości SZBD z koncepcją funkcji i konstrukcji systemu informatycznego, wynikającą z analizy potrzeb i wymagań użytkownika.
- **Możliwie dokładne rozpoznanie systemów, przyjętych do szczegółowej oceny.** Etap ten powinien być realizowany równolegle z poprzednim. Stopień rozpoznania wybranych SZBD zależy od szeregu czynników ograniczających, m. in. od przeznaczonych na to nakładów finansowych, możliwości przeprowadzania prób praktycznych itp. Wytypowanie tych SZBD, które mają zostać poddane bardziej szczegółowemu rozpoznaniu, może być dokonane dopiero po częściowym choćby zakończeniu etapu drugiego.
- **Ocena wybranych systemów oraz wybór konkretnego SZBD.** Na tym etapie sprawdza się, czy wybrany system posiada rzeczywiście te właściwości, które przedstawione są w broszurkach reklamowych i dokumentacji. W przypadku niemożliwości dokonania prób praktycznych sprawdzenie to może polegać na zbadaniu (na podstawie dokumentacji), w jaki sposób reklamowane właściwości zostały osiągnięte.

Na rys. 1 przedstawiono w postaci schematu omówiony sposób postępowania.



Rys. 1. Zastosowana metoda oceny i wyboru SZBD

UZASADNIENIE WYBORU

Charakterystyka systemu INFOCHEM i jego środowiska

Dla określenia charakterystyki wpływającej na wybór odpowiedniego SZBD, niezbędne jest udzielenie odpowiedzi na podstawowe pytania dotyczące stanu i funkcji systemu, rodzaju wykorzystywanej bazy danych, stopnia zmienności danych w bazie, rodzaju przetwarzania, rodzaju wykorzystywanego sprzętu oraz systemu operacyjnego, pod kontrolą którego system INFOCHEM będzie eksploatowany.

Przyjęto na wstępie, że INFOCHEM będzie funkcjonował w ramach Zautomatyzowanego Systemu Zarządzania Przedsiębiorstwa (ZSZP), tj. w systemie zarządzania, w którym zastosowano racjonalne metody podejmowania decyzji i wykorzystano automatyczne środki zbierania, przetwarzania i udostępniania danych — dla realizacji podstawowych zadań przedsiębiorstwa. Zadania te łączone są w grupy, nazywane podsystemami funkcjonalnymi ZSZP.

Można zastosować kilka różnych zasad łączenia zadań w podsystemy funkcjonalne. Tutaj przyjęto, że w jeden taki podsystem będą łączone zadania związane z zarządzaniem określonym procesem wykonywanym w przedsiębiorstwie. Założono zatem, że wyodrębnienie podsystemów funkcjonalnych przeznaczonych do zarządzania procesami: produkcji podstawowej, produkcji pomocniczej, remontowym, zaopatrzenia materiałowo-technicznego oraz ekspedycji i zbytu (rys. 2). W stosunku do powyższych podsystemów rolę nadrzędną będzie spełniał podsystem zarządzania techniczno-ekonomicznego, koordynujący ich działalność.

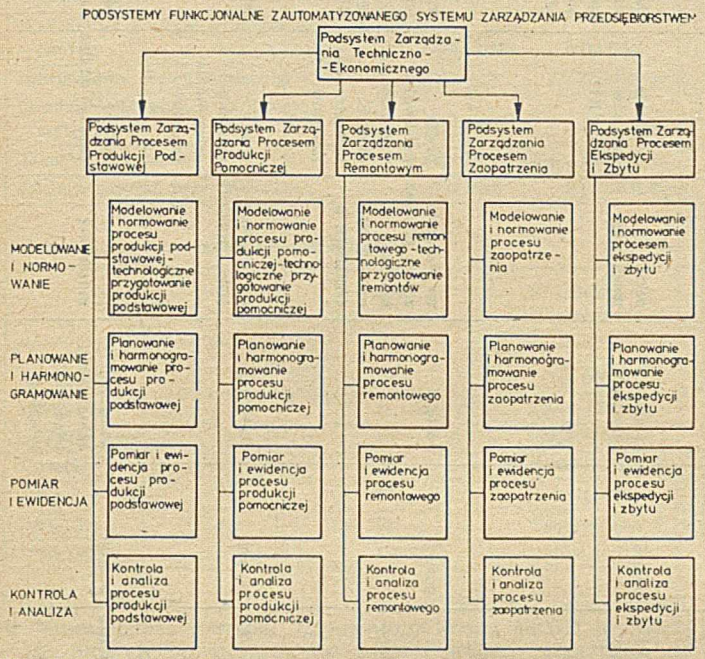
Każdy z podsystemów funkcjonalnych ZSZP dzieli się na dwie części: niezautomatyzowaną i zautomatyzowaną. Tę drugą część będziemy nazywać podsystemem informatycznym danego podsystemu funkcjonalnego. Podsystemy informatyczne wszystkich podsystemów funkcjonalnych ZSZP będą w sumie stanowić system informatyczny INFOCHEM.

W zależności od zakresu zadań realizowanych automatycznie podsystemy informatyczne można podzielić na trzy podstawowe klasy:

— ewidencyjno-informacyjne, nazywane niekiedy systema-

mi informacyjnymi z bankiem danych lub systemami wyszukiwania informacji
— ewidencyjno-doradcze, umożliwiające m.in. maszynową symulację przebiegu procesu
— ewidencyjno-decyzyjne.

Założono, że system informatyczny INFOCHEM będzie rozwijany w sposób ewolucyjny oraz że w pierwszej kolejności realizowany będzie podsystem informatyczny będący częścią tego podsystemu funkcjonalnego ZSZP, który jest przeznaczony do zarządzania procesem remontowym (patrz rys. 2).



Rys. 2. Zadania występujące w podsystemach funkcjonalnych ZSZP

System informatyczny INFOCHEM zbudowany będzie z następujących elementów konstrukcyjnych:
— Bazy Technicznej
— Bazy Danych
— Bazy Programów
oraz z odpowiednich elementów sprzęgających:
— Bazę Danych z Bazą Techniczną
— Bazę Programów z Bazą Techniczną
— Bazę Programów z Bazą Danych.

Na podstawie ustaleń dokonanych ze zleceniodawcą przyjęto, że Bazę Techniczną dla INFOCHEMU będzie stanowił komputer ODRA 1305, w konfiguracji posiadanej przez ZETO Gdynia, rozbudowany o system skanerowy i zainstalowane w przedsiębiorstwie urządzenia końcowe.

Baza Danych będzie wspólna dla podsystemów informatycznych, realizowanych w pierwszej kolejności, co zintegruje je i ograniczy nadmierność danych. Baza Danych będzie zawierać dane i relacje między danymi, niezbędne dla sprzężonych z nią programów, zawartych w Bazie Programów i realizujących zadania odpowiednich podsystemów funkcjonalnych. Baza Danych będzie stopniowo rozbudowywana, w miarę wdrażania nowych programów.

Baza Programów będzie zawierać zapisane w postaci wynikowej programy przetwarzania danych, realizujące zadania odpowiednich podsystemów funkcjonalnych. Złożoność programów będzie wzrastać w miarę przechodzenia od zadań ewidencyjnych do zadań planistycznych i analitycznych.

Funkcję elementów sprzęgających będą spełniać: System Operacyjny i System Zarządzania Bazą Danych. Podstawową funkcją Systemu Operacyjnego jest sprzęganie Bazy Danych i Bazy Programów z Bazą Techniczną oraz uruchamianie i sterowanie wykonaniem zadań — w narzuconej kolejności. System Operacyjny powinien umożliwiać zdalne uruchamianie — z terminali zainstalowanych w przedsiębiorstwie — zadań wykonywanych według programów (zawartych w Bazie Programów i korzystających z Bazy Danych). Pożądana jest również możliwość sterowania realizacją tych zadań (a także przesyłania danych i parametrów do nich) przy wykorzystaniu urządzeń końcowych. Zapewnia to System Operacyjny GEORGE-3 MOP (ang. *Multiple Online Programming*). Przyjęto więc, że w systemie INFOCHEM zostanie zastosowany właśnie ten system operacyjny. Podstawową funkcją Systemu Zarządzania Bazą Danych jest natomiast sprzęganie programów (z Bazy Programów) ze wspólną Bazą Danych.

Na podstawie określonych powyżej założeń można określić decydującą przy wyborze SZBD charakterystykę systemu INFOCHEM i jego środowiska:
- system będzie budowany stopniowo, w sposób ewolucyjny, poprzez dodawanie nowych zadań i podsystemów oraz nowych rodzajów danych i nowych powiązań między nimi
- wszystkie podsystemy będą sprzęgane z WBD
- w systemie będą realizowane zarówno funkcje ewidencyjne, jak i funkcje planowania, harmonogramowania oraz kontroli
- w bazie danych zawarte będą zarówno dane względnie stabilne (np. dane normatywne i modele procesów wykonywalnych), jak i dane o wysokim stopniu zmienności (np. odzwierciedlające zdarzenia i transakcje)
- uwarunkowania czasowe będą w niektórych przypadkach na tyle ostre, że — obok wsadowego przetwarzania — niezbędne będzie zdalne prowadzenie wielu równocześnie wykonywanych zadań, przy wykorzystaniu zainstalowanych w przedsiębiorstwie terminali
- system realizowany będzie na komputrze serii ODRA 1300, należy jednak wziąć pod uwagę możliwość przeniesienia go na inny sprzęt
- system będzie eksploatowany pod kontrolą systemu operacyjnego GEORGE-3 MOP.

Lista preferowanych właściwości SZBD

Listę preferowanych właściwości SZBD, który ma być użyty do budowy systemu INFOCHEM, opracowano porównując listę potencjalnych właściwości idealnego SZBD z listą atrybutów systemu INFOCHEM i jego środowiska. Lista ta wygląda następująco:
— niezależność danych (ze względu na rozwój systemu)
— złożone struktury danych (aby odzwierciedlić dane i skomplikowane związki między nimi)
— nienaruszalność danych (w przypadku awarii sprzętu)
— poufność danych
— eliminacja lub ograniczenie nadmiarowości danych
— wielokrotne strategie wyszukiwania danych (programy zawarte w Bazie Programów wymagają różnych wejść do Bazy Danych i różnych ścieżek dostępu do danych)
— scentralizowana kontrola danych i sposobu ich użytkowania (funkcja Administratora Bazy Danych)
— równoczesny dostęp do danych przez wielu użytkowników
— obsługa zapytań przypadkowych, uprzednio nie przewidzianych (szczególnie przy realizacji funkcji ewidencyjno-informacyjnych)
— zgodność SZBD z wymaganiami CODASYLU
— dostępność wersji SZBD na komputery serii ODRA 1300 oraz możliwość przeniesienia Bazy Danych, Bazy Programów i SZBD na inny sprzęt
— wysoka jakość dostarczanej dokumentacji SZBD oraz dostępność szkoleń i doradztwa w zakresie jego stosowania.

Wybór i ocena wybranego SZBD

Oceny branych pod uwagę SZBD dokonano przez porównanie ich cech (tabela 1) z listą właściwości preferowanych. Ocenę tę przeprowadzono w sposób następujący:
- wzięto pod uwagę kolejno każdą z niezbędnych właściwości SZBD i uporządkowano rozpatrywane systemy w kolejności od najlepszego do najgorszego — z punktu widzenia danej właściwości
- każdy z systemów zarządzania oceniono w skali od siedmiu punktów (dla najlepszego) do jednego punktu, w zależności od miejsca na jakim został on sklasyfikowany z punktu widzenia danej właściwości
- dla każdego systemu zsumowano ilość uzyskanych punktów.

Uporządkowanie rozpatrywanych SZBD dla każdej z niezbędnych właściwości przedstawiono w tabeli 2, a sumaryczną ocenę każdego z SZBD — w tabeli 3. W tej ostatniej przedstawiono także warunki zakupu wybranych systemów. Niemniej trzeba pamiętać, iż zakres produktów i usług, oferowanych przez poszczególnych dostawców SZBD, jest bardzo zróżnicowany.

Biorąc pod uwagę dostępność SZBD dla komputera ICL 1900 (a tym samym komputera serii ODRA 1300), sumaryczną ocenę, warunki zakupu i zgodność systemu z wymaganiami CODASYLU dokonano wyboru SZBD 1900 IDMS, który według wstępnej oceny najlepiej nadaje się do budowy systemu INFOCHEM. Po dokonaniu wyboru sprawdzono, czy system 1900 IDMS rzeczywiście posiada właściwości, które zadecydowały o jego wyborze. Ze względu na niemożliwość przeprowadzenia prób praktycznych, weryfikację przeprowadzono na podstawie posiadanej dokumentacji oraz informacji uzyskanych w firmie DATA-

Tabela 1. Charakterystyka wybranych SZBD

Nazwa	Dostawca	Sprzęt	Okres użytkowania, liczba użytkowników	Język manipulacji danymi	Struktury danych, indeksowanie wtórne	Niezależność danych			Ochrona poufności danych	Język zapytań, generator raportów	Zabezpieczanie i odtwarzanie bazy danych	Pomiary eksploatacyjne i strojenie
						Liczba poziomów opisu danych	Niezależność na poziomie	Wiązanie w czasie				
TOTAL	CICOM	IMB 360/370 H 2000 RCA SPECTRA 70 NCR CENTURY ICL 2900 ICL 1900 DEC PDP 11	7 lat 1200 użytkowników	dobudowany	s. hierarchiczna s. sieciowa	1 (tylko DDL)	elementu	dostępu	na poziomie zbioru	język zapytań i generator raportów SOCRATES A	automatyczne odtwarzanie prac w trybie on-line	organiczne możliwości
ADABAS	SOFTWARE AG	IBM 360/370 SIMENS 4004 UNIVAC 9400	5 lat 70 użytkowników	dobudowany	indeksowanie wtórne s. hierarchiczna	1 (tylko słownik)	pozycji	dostępu	pełne możliwości	język zapytań ADASCRIPT generator raportów ADAWRITER	ślad kontrolny i punkty kontrolne	rejestracja statystyk
SYSTEM 2000	MRI	IBM 360/370 CDC 6000 CYBER 70 UNIVAC 1100	6 lat 70 użytkowników	samowystarczalny i dobudowany	indeksowanie wtórne s. hierarchiczna	1 (dla języka samowystarczalnego 2 (dla języka dobudowanego	pozycji	dostępu	pełne możliwości	język zapytań i generator raportów	odtwarzanie ręczne	brak
1900 IMDS	CULLINAME	IBM 360/370 UNIVAC Seria 90 SIEMENS 4004 RCA SPECTRA ICL 2900 ICL 1900 DEC PDP 11	3 lata 40 użytkowników	dobudowany	hierarchiczna, sieciowa, tablice łączników	2 (schemat i subschemat)	pozycji	kompilacji i dostępu	pełne możliwości	generator raportów CULPRIT	ślad kontrolny, punkty kontrolne, procedury odtwarzania	rejestracja statystyk
DMS-2	ICL	ICL 1900	4 lata 40 użytkowników	samowystarczalny i dobudowany	s. hierarchiczna	2 (zbiór fizyczny i zbiór logiczny)	pozycji	dostępu	na poziomie zbioru poprzez system operacyjny GEORGE 3	pewne możliwości wyszukiwania danych i generowania raportów	jedynie poprzez system operacyjny GEORGE 3	bardzo ograniczone możliwości
ROBOT	SOFTWARE SCIENCE LIMITED	ICL 1900 UNIVAC 9400	3 lata kilku użytkowników	samowystarczalny	zbiory transponowane	1 (tylko słownik)	pozycji	dostępu	na poziomie pola i wektora	pewne możliwości generowania raportów	restarty	brak
1900 DBMS	ICL	ICL 1900	2 lata ograniczone rozpowszechnianie	dobudowany	konwencjonalne	2 (DDL i DBL)	pozycji	kompilacji subschematu i ładowania programu	pełne możliwości	system zapytań on line	odtwarzanie partiowe	rejestracja statystyk

Tabela 2. Ocena wybranych SZBD z punktu widzenia poszczególnych preferowanych właściwości

Preferowana właściwość \ Ocena punktowa	7	6	5	4	3	2	1
Niezależność danych	1900 IDMS	SYSTEM 2000	ADABAS	TOTAL	DMS-2	1900 DBMS	ROBOT
Złożone struktury danych	1900 IDMS	ADABAS	SYSTEM 2000	TOTAL	DMS-2	ROBOT	1900 DBMS
Nienaruszalność danych	1900 IDMS	TOTAL	ADABAS	1900 DBMS	SYSTEM 2000	ROBOT	DMS-2
Poufność danych	ADABAS	SYSTEM 2000	1900 IDMS	1900 DBMS	ROBOT	TOTAL	DMS-2
Wielokrotne strategie wyszukiwania danych	1900 IDMS	ADABAS	SYSTEM 2000	TOTAL	DMS-2	ROBOT	1900 DBMS
Równoczesny dostęp do danych	ADABAS	SYSTEM 2000	TOTAL	1900 IDMS	1900 DBMS	DMS-2	ROBOT
Możliwość eksploatacji na różnym sprzęcie	TOTAL	1900 IDMS	SYSTEM 2000	ADABAS	ROBOT	DMS-2	1900 DBMS
Jakość dokumentacji	SYSTEM 2000	ADABAS	TOTAL	1900 IDMS	ROBOT	DMS-2	1900 DBMS
Obsługa zapytań przypadkowych	ADABAS	SYSTEM 2000	TOTAL	1900 IDMS	DMS-2	ROBOT	1900 DBMS

Tabela 3. Sumaryczna ocena wybranych SZBD

System	Ocena sumaryczna	Opłata podstawowa	Dodatkowa opłata roczna	Dzierżawa miesięczna	Dostępny dla komputera ICL 1900	Zgodny z wymaganiami CODASYL-u
ADABAS	53	120 000 $	—	4500 $	NIE	NIE
1900 IDMS	51	20000—23000 £	2000—2450	—	TAK	TAK
SYSTEM 2000	49	23000—150000 $	—	—	NIE	NIE
TOTAL	42	26000—38250 $	—	750—950 $	TAK	NIE
DMS-2	20	bezpłatnie	—	—	TAK	NIE
ROBOT	19	15000—20000 £	—	—	TAK	NIE
1900 DBMS	18	brak danych	—	—	TAK	NIE

SKILL. Poniżej — wyniki przeprowadzonej weryfikacji.

● **Niezależność danych.** Zapewniona przez SZBD niezależność danych od programów zależy przede wszystkim od liczby, poziomów opisu danych i od fazy wytwarzania programu, w jakiej następuje wiązanie opisu danych z częścią proceduralną programów. Zgodnie z koncepcją CODASYLU dane w SZBD 1900 IDMS opisywane są na dwóch poziomach — Schematu i Subschematu, a wiązanie tych opisów z częścią proceduralną programu odbywa się w fazie konsolidacji programów. Rozwiązanie takie zapewnia wystarczający stopień niezależności danych.

● **Złożone struktury danych.** Dzięki zastosowaniu w SZBD 1900 IDMS koncepcji setu możliwe jest tworzenie złożonych, wielopoziomowych struktur hierarchicznych i sieciowych, co zapewnia pełne możliwości w tym zakresie.

● **Nienaruszalność danych.** SZBD 1900 IDMS umożliwia pełną rekonstrukcję zniszczonych danych, dzięki tworzeniu śladu kontrolnego, okresowemu kopiowaniu całości lub wybranych części bazy danych oraz wykorzystaniu specjalnych programów, umożliwiających odtworzenie bazy danych zarówno w przód (w kierunku zgodnym z porządkiem chronologicznym), jak i wstecz (w kierunku przeciwnym do porządku chronologicznego transakcji, zmieniających zawartość bazy danych).

● **Poufność danych.** W SZBD 1900 IDMS zastosowano dwustopniową ochronę poufności danych: pierwszy stopień kontrolowany jest przez administratora bazy danych, drugi zaś — przez programistę. Administrator bazy danych kontroluje poufność na pierwszym poziomie, poprzez definiowanie Subschematów dostępnych tylko dla wybranych programów i użytkowników, natomiast programista (na drugim poziomie) ma możliwość kontroli sposobów dostępu do wybranych obszarów, setów i zapisów, występujących w przeznaczonym dla niego Subschemacie. Rozwiązanie takie zabezpiecza niezbędną poufność danych.

● **Wielokrotne strategie wyszukiwania danych.** Dzięki temu, że możliwe jest zapewnienie dostępu do każdego zapisu na podstawie wartości wybranych jego pól lub na podstawie wartości przyporządkowanego mu klucza bazowego, istnieje możliwość wchodzenia do bazy w różnych jej miejscach. Z kolei fakt, że każdy zapis może należeć do wielu setów różnych rodzajów, umożliwia „nawigowanie po bazie danych" przy wykorzystaniu odpowiednich łączników adresowych w setach. Dzięki tym dwom rozwiązaniom SZBD 1900 IDMS umożliwia wielokrotne strategie wyszukiwania tego samego zapisu, dostosowane do potrzeb różnych programów.

● **Równoczesny dostęp do danych przez wielu użytkowników.** Właściwość ta, według uzyskanych w firmie DATASKILL informacji, ma być zapewniona w najbliższej wersji SZBD 1900 IDMS. Na temat sposobu rozwiązania tego problemu brak dotychczas szczegółowych informacji.

● **Obsługa zapytań przypadkowych, uprzednio nie przewidzianych.** Właściwość ta ma być uzyskana poprzez sprzęgnięcie SZBD 1900 IDMS z generatorem raportów FILETAB. Generator ten umożliwia użytkownikom tworzenie, obsługę, aktualizację i scalanie zbiorów oraz generowanie raportów przy użyciu kilku prostych parametrów. Zawiera on program SORT, który umożliwia posortowanie danych przed ich wydrukowaniem lub zapisaniem do zbiorów wyjściowych. FILETAB zbudowany jest w postaci procesora tablic decyzyjnych. Umożliwia on użytkownikowi specyfikację jego wymagań, dotyczących wyboru i przetwarzania wybranych rekordów, przy użyciu języka wyższego rzędu. Rozwiązanie takie powinno umożliwiać szybkie uzyskanie nieprzewidzianych wcześniej zestawień i raportów — na podstawie danych zawartych w bazie.

● **Zgodność z wymaganiami CODASYLU.** SZBD 1900 IDMS jest prawie pełną realizacją propozycji i wymagań CODASYLU i z tego punktu widzenia jest na ogół najwyżej oceniany w porównaniu z innymi, aktualnie dostępnymi SZBD, zbudowanymi w oparciu o te wymagania i propozycje.

● **Dostępność systemu dla komputerów serii ODRA 1300 oraz możliwość przeniesienia Bazy Danych, Bazy Programów i SZBD na inny sprzęt.** SZBD 1900 IDMS powstał w wyniku dokonania przez firmę DATASKILL-ICL konwersji systemu IDMS, opracowanego w wersji oryginalnej na komputer IBM 360. W przypadku zastosowania tego SZBD do zarządzania Bazą Danych oraz języka COBOL do wytwarzania programów zawartych w Bazie Programów, możliwe jest stosunkowo łatwe przeniesienie zbudowanego (w oparciu o komputer ODRA 1300) systemu informatycznego na inny sprzęt, np. IBM 360/370, R-32, UNIVAC seria 90, SIEMENS 4004, RCA SPECTRA, ICL 2900, ICL System 4, DEC PDP 11/45, ponieważ na każdym z tych komputerów może być eksploatowana odpowiednia wersja SZBD IDMS.

● **Wysoka jakość dostarczanej dokumentacji systemu oraz dostępność szkoleń i doradztwa w zakresie jego stosowania.** Dostarczana przez firmę DATASKILL-ICL dokumentacja SZBD 1900 IDMS ma charakter *„reference books"*, w związku z czym jest mało przystępna dla osób stykających się po raz pierwszy z problematyką baz danych, nie znających propozycji CODASYLU i nie mających doświadczeń w projektowaniu i realizacji systemów informatycznych z bazą danych. W zagadnienia te projektanci, administratorzy baz danych i programiści są wprowadzani na kursach prowadzonych przez firmę ICL oraz przez konsultantów z firmy DATASKILL.

JERZY BEDNARZ, MAREK MAJEWSKI
ZETO Poznań

Przygotowanie użytkownika bazy danych

Wprowadzanie technologii baz danych jest procesem długotrwałym i z pewnością niełatwym. Ogromną rolę odgrywa w nim element zwany przygotowaniem użytkownika, element niezwykle istotny i jak wykazała dotychczasowa praktyka — newralgiczny. Bez właściwego przygotowania użytkownika trudno się spodziewać znaczących efektów rozwoju zastosowali w nowej technologii.

Decyzja wdrażania systemów informatycznych w przedsiębiorstwie w oparciu o wspólną bazę danych (WBD), podjęta przez dyrekcję, jest z reguły rezultatem zaistnienia określonego układu warunków. Najczęściej ma to miejsce w przypadku, gdy w danym przedsiębiorstwie, na różnych szczeblach zarządzania, wykorzystywane są już od dawna systemy odcinkowe (dziedzinowe). Systemy te, zaprojektowane dla ograniczonych obszarów zastosowań, zorientowane są z reguły na transakcje, operują danymi dotyczącymi poszczególnych zdarzeń procesu gospodarowania i nie mogą ze sobą współpracować. Każdy z nich został wdrożony w innym czasie, operuje danymi na różnych stopniach agregacji. Nazwy danych nie są ujednolicone. Ponadto — co najistotniejsze — niski stopień podatności zbiorów klasycznych na logiczne wiązanie danych jest przyczyną wielokrotnego gromadzenia i przechowywania tych samych informacji w różnych zbiorach, co uniemożliwia koordynację. W efekcie nie można porównywać informacji wychodzących z poszczególnych systemów i wykorzystać ich w systemach nadrzędnych.

Kierowanie przedsiębiorstwem wymaga danych sumarycznych, zbieranych systematycznie, poprawnie zagregowanych i odpowiednio zestawionych, które w odpowiednim czasie przekazywane są dyrekcji. Jak zaznaczono, decyzja o wdrożeniu w przedsiębiorstwie WBD może być podjęta wyłącznie przez dyrekcję i jest podobna do decyzji zainstalowania komputera, z tą różnicą, że wymaga znacznie większego zaangażowania kierownictwa. Jakkolwiek oczekiwane korzyści są w początkowej fazie trudne do zmierzenia, dyrekcja winna możliwie dokładnie przeanalizować koszty przedsięwzięcia. Koszty te można pogrupować w sposób następujący:

- koszty opracowania szczegółowych systemów opartych na WBD
- koszty szkolenia użytkownika tak z działu przetwarzania informacji, jak i komórek prowadzących właściwą działalność gospodarczą
- koszty związane z konwersją istniejących zbiorów i programów oraz koszty przetwarzania równoległego
- koszty nabycia lub dzierżawienia oprogramowania zarządzającego bazą danych oraz pozostałe koszty amortyzacji
- koszty nabycia zestawu komputerowego lub poszerzenia istniejącego.

Niniejszy artykuł jest próbą przedstawienia procesu przygotowania przedsiębiorstwa zdecydowanego na wprowadzenie WBD — z punktu widzenia doświadczeń przedsiębiorstwa usługowego, będącego organizmem zewnętrznym w stosunku do użytkownika. Istotą rozważań będą tutaj posunięcia zmierzające do stworzenia odpowiednich struktur w przedsiębiorstwie, tak aby było ono zdolne przyjąć na siebie w przyszłości ciężar samodzielnego utrzymywania bazy danych oraz poszerzania jej o elementy niezbędne dla kolejnych zastosowań.

Etap opracowywania założeń przyszłego systemu, mający na celu zdefiniowanie jego funkcjonalno-użytkowego zakresu oraz opis struktur danych, niezbędnych do realizacji zastosowań, wymaga intensywnych kontaktów zespołu projektowego z większością komórek organizacyjnych przedsiębiorstwa. Istnieje więc konieczność poinformowania przez dyrekcję wszystkich zainteresowanych kierowników komórek o zasadniczym celu prowadzonych prac, o zakresie i przebiegu przedsięwzięcia oraz o założonych formach współpracy z zespołem projektowym. Konsekwencją takiej informacji jest powołanie w przedsiębiorstwie **koordynatora do spraw wprowadzenia systemu** opartego na WBD oraz wyznaczenie w każdej komórce osób odpowiedzialnych za merytoryczne uzgodnienia. Zadania tej grupy zostały omówione w poz. [5].

Należy też już na wstępie wskazać użytkownikowi konieczność powołania osoby (osób) zdolnej pełnić funkcje **administratora bazy danych.** Wyraźne wydzielenie roli administratora jest konsekwencją rozwoju zastosowań opartych na WBD, prowadzącego do powstania nowych jakościowo problemów w porównaniu do konwencjonalnych metod projektowania i wdrażania systemów informatycznych. Funkcja ta wymaga specjalistycznego przygotowania (w zakresie: zagadnień poszczególnych zastosowań, przygotowania technicznego, pozwalającego posługiwać się mechanizmami opisu i „nawigacji po bazie danych", bieżącej kontroli eksploatacji i sterowania rozwojem bazy oraz prowadzenie dokumentacji). Wymaga to od administratora znajomości takich zagadnień, jak: projektowanie systemów, programowanie systemowe, programowanie użytkowe, organizacja eksploatacji komputera. Jest rzeczą oczywistą, że tak złożone funkcje może pełnić jedynie grupa osób, stopniowo wyodrębniana z dotychczasowych struktur organizacyjnych przedsiębiorstwa.

Na etapie opracowywania założeń systemowych można już wyraźnie wydzielić dwie grupy w ramach wieloosobowej obsady stanowiska administratora bazy danych:
— **administrator zastosowań**
— **administrator danych.**
Celowym wydaje się, aby przyszłe funkcje administratora zastosowań objęły — dla każdej z dziedzin — osoby wyznaczone wcześniej do merytorycznych uzgodnień z wytypowanych komórek organizacyjnych przedsiębiorstwa.

Biorąc pod uwagę sugestie twórców polskiej realizacji SZBD, opartej na specyfikacjach komitetu CODASYL, należy stwierdzić, że administrator zastosowań winien działać w obszarze jednego zastosowania bazy danych i spełniać funkcje, których szczegółowe omówienie zostało zamieszczone w artykule A. Brandta: „Administrator Zastosowań", INFORMATYKA nr 7/1979 [1]. Administrator ten spełnia rolę nadrzędną w stosunku do wszystkich użytkowników danego zastosowania.

Powołanie administratora danych wiąże się natomiast z rolą **słownika danych** w procesie tworzenia i eksploatacji systemów opartych na WBD, która jest obsługiwana przez system zarządzania (SZBD). Potrzeba wprowadzenia w przedsiębiorstwie takiego słownika, rozumianego najogólniej

jako system procedur służący do gromadzenia i operowania występującymi nazwami danych, wynika w części z tych samych przesłanek, dla których użytkownik decyduje się na wdrażanie systemów (opartych na WBD). Szeroko pojęte zarządzanie danymi dotyczy w tym przypadku nie samych danych, lecz ich definicji. Chodzi przede wszystkim o zebranie, uporządkowanie, a tym samym — zminimalizowanie redundacji określeń danych, używanych przez komórki organizacyjne danego przedsiębiorstwa.

Nie analizując głębiej technicznych szczegółów budowy i zawartości słownika danych można stwierdzić, że winien on zawierać kilka podstawowych elementów:

- nazwę każdego typu danych
- definicję danej
- miejsca i sposób jej wykorzystania
- związku z innymi danymi.

Tak zarysowana koncepcja zawartości słownika pozwala dostrzec dwie zasadnicze korzyści wynikające z jego stosowania:
— użytkownik jest władny sprawować kontrolę nad danymi poprzez znormalizowanie nazw danych, ustalenie zasad wprowadzania nowych i zmieniania już istniejących określeń typów danych, ustalenie trybów i priorytetów dostępu do danych

— skupienie w rękach jednego decydenta „konserwacji" nazw danych może wspomóc standardy programowania i projektowania zastosowań, może również ułatwić porozumiewanie się projektanta z użytkownikiem.

Rozważając rolę słownika danych w ramach niniejszego artykułu przyjęto, że użytkownik powinien wykorzystywać metodę ręczną prowadzenia tego narzędzia już w okresie formułowania założeń systemu opartego na WBD. W miarę postępu prac projektowych i stopniowego wdrażania systemu można pewne procedury słownika automatyzować za pomocą odrębnych programów i wówczas wyłonią się dalsze jego zalety, związane przykładowo z automatycznym generowaniem opisów zbiorów.

Należy jednak stwierdzić, że bez względu na fakt, czy zamierza się zaprowadzić u użytkownika słownik ręczny czy zautomatyzowany, powinien być on wprowadzany równolegle z SZBD, a nad prawidłowym spełnianiem przedstawionych wyżej funkcji winien czuwać powołany administrator danych. Jego zadania można ująć następująco:

- niwelowanie redundancji w określeniach danych i usuwanie niezgodności między nimi
- sprawowanie kontroli nad przyjmowaniem nowych nazw danych
- aktualizowanie słownika danych
- określanie form ochrony procedur i prawa dostępu do słownika.

Niezwykle istotne w procesie przygotowania użytkownika jest szkolenie jego służb informatycznych. A że jest to proces niezwykle długi, zatem niezbędnym staje się szybkie i konsekwentne realizowanie wymaganych kursów szkoleniowych. Wymiar szkoleń — dla poszczególnych grup zawodowych — podany został (szacunkowo) w tabeli. Tabela ta została zbudowana w oparciu o szkolenia prowadzone przez COSI ZI (Centralny Ośrodek Szkolenia Informatycznego Zjednoczenia Informatyki), wg konwencji WSSI (Wielopoziomowy System Szkolenia Informatycznego). Ciąg szkoleń, jaki wynika z tabeli, stanowi niezbędne minimum dla służb informatycznych użytkownika.

Należy zdawać sobie sprawę, że proces szkolenia (szczególnie tak złożony) musi być poparty odpowiednią praktyką. Służby informatyczne użytkowników mogą zdobyć niezbędne doświadczenie w przypadku, jeżeli wyspecjalizowane przedsiębiorstwa usługowe — w naszych realiach — ZETO, będą zdecydowanie podejmować i wdrażać zastosowania przy użyciu narzędzi typu SZBD. Współpraca ZETO z użytkownikiem począwszy od wczesnego etapu formułowania założeń, poprzez proces wdrażania i eksploatacji, będzie mieć znamiona długotrwałości. Tym niemniej w pewnym momencie odpowiednio przygotowany użytkownik powinien być zdolny do samodzielnej eksploatacji systemu opartego o WBD.

Osobny problem to eksploatacja gotowych systemów. Tutaj istotnie — zważywszy aktualny stan wiedzy — sprawa jest niezwykle trudna (w przypadku przedsiębiorstwa usługowego) co nie oznacza, że niemożliwa do rozwiązania. Problemem jest przede wszystkim szczupłość istniejących konfiguracji sprzętowych, na których może i musi odbywać się eksploatowanie systemów.

Tabela. Szkolenie użytkownika bazy danych

Temat szkolenia	Orientacyjna liczba dni szkolenia	Służba użytkownika				
		Administrator zastosowań	Programista zastosowań	Administrator danych		Programista systemowy
				SR	SK	
Wprowadzenie w problematykę komputerów Jednolitego Systemu	6	●	●	●	●	●
System operacyjny OS dla programistów i projektantów	11	●	●		●	●
PL-1 — język programowania komputerów JS	11	●	●			●
Programowanie w języku PL-1-OS	12		●			●
Programowanie w języku ASSEMBLER — OS	11					●
System operacyjny OS dla programisty systemowego	11					●
Wprowadzenie w problematykę baz danych	11	●	●	●	●	●
System Zarządzania Bazą Danych	11	●	●		●	●
Projektowanie bazy danych w oparciu o System Zarządzania Bazą Danych	11	●				
Oprogramowanie systemu z bazą danych	11		●			
Administrowanie bazą danych	11		●		●	

SR — dotyczy słownika ręcznego
SK — dotyczy słownika komputerowego

Kompletna baza danych określonej jednostki gospodarczej, to równocześnie konieczność zapewnienia przede wszystkim odpowiednich (niemałych) obszarów pamięci komputera. Wzięcie pod uwagę już kilku jednostek gospodarczych i — co jest dla technologii baz danych wskazane — trybu przetwarzania w trybie bezpośrednim, powoduje znaczne trudności dla przedsiębiorstwa usługowego. Jest to problem nie rozwiązany jeszcze do końca, ale wydaje się, że możliwości rozwiązania istnieją bądź to przez wyposażenie istniejących komputerów w odpowiednio pojemne pamięci, bądź przez tworzenie sieci komputerowych, co zresztą już niebawem ma szanse doczekać się realizacji.

Autorzy zdają sobie sprawę, że zagadnienia zaanonsowane w artykule nie wyczerpują problematyki procesu przygotowania użytkownika do zastosowań technologii baz danych. W okresie gdy brak jeszcze znaczących wdrożeń, proponowane rozwiązania mogą wskazywać jedną z możliwych dróg postępowania.

LITERATURA

[1] Brandt A.: Administrator zastosowań, INFORMATYKA nr 7/1979

[2] Palmer I.: Database Systems: A. Practical Reference Q.E.D. Information Sciences, Inc. 1978

[3] Schussel G.: The tole of data dictionary, DATAMATION nr 6/1977

[4] Wielopoziomowy system szkolenia informatycznego OBRI, Warszawa 1977

[5] Wytyczne do długofalowej strategii realizacji baz danych — Europejski Program Badawczy DIEBOLD-a, zeszyt 104 — ZI, CPIZI, Warszawa 1979.

WITOLD KORNACKI
Instytut Informatyki
Politechnika Warszawska

Wybrane problemy analizy niezawodnościowej oprogramowania

Badania nad niezawodnością programów to dziedzina stosunkowo młoda (o nieustalonej jeszcze terminologii) dlatego też wydaje się koniecznym podanie na wstępie definicji zasadniczych pojęć z tego zakresu. Zamiarem autora było podanie takich określeń, które byłyby zgodne z terminologią stosowaną w nielicznych krajowych pracach w tej dziedzinie oraz z terminologią wykorzystywaną w pracach zagranicznych.

Błąd koncepcyjny programu (ang. *software error*) jest to niezgodność logiczna, pojęciowa czy syntaktyczna, która prowadzi do jednego lub więcej błędów programu.

Błąd programu (ang. *software fault*) jest to pewna nieprawidłowość programu, mogąca wywołać defekty wykonania.

Defekt wykonania (ang. *software failure*) jest to niezgodność wyniku działania programu z jego specyfikacją; powstaje on przy przyjęciu przez zmienne wejściowe wartości powodujących wykonanie części programu zawierających błędy.

Najczęściej defekty wykonania są skutkiem następującego łańcucha przyczyn: programista popełnia błąd koncepcyjny, który powoduje powstanie jednego lub więcej błędów programu, te zaś prowadzą do defektu wykonania. Teoretycznie wiele błędów może przyczyniać się do jednego defektu wykonania, ale bardziej prawdopodobne jest wystąpienie wielu defektów wskutek jednego błędu.

Niezawodność programu jest to prawdopodobieństwo wykonywania programu w pewnym przedziale czasu (w określonych specyfikacjami warunkach) bez wystąpienia defektów.

Przy formułowaniu powyższej definicji przyjęto, że spełnienie wymagań użytkownika jest równoważne nie wystąpieniu defektów wykonania oraz że mimo wystąpienia błędów w programie wykonanie programu jest poprawne, o ile tylko nie wystąpią defekty wykonania.

Dla lepszego scharakteryzowania zdefiniowanego pojęcia niezawodności warto wskazać zasadnicze różnice między niezawodnością programu a niezawodnością układów elektronicznych. Są one związane z następującymi czynnikami:

Mgr inż. WITOLD KORNACKI ukończył w 1977 r. studia na Wydziale Elektroniki Politechniki Warszawskiej. Jest doktorantem na kierunku Informatyka i Automatyka w PW. Zajmuje się zagadnieniami modelowania niezawodnościowego systemów cyfrowych.

- oprogramowanie nie starzeje się wskutek długotrwałego użytkowania (choć może wystąpić tzw. zużycie moralne, czyli nieadekwatność oprogramowania do zmienionych warunków zewnętrznych)
- nie powstają dodatkowe błędy przy powielaniu programów, z wyjątkiem błędów kopiowania
- naprawa (korekcja błędów) programu polega najczęściej na zmianie struktury logicznej, w odróżnieniu od zwykłej zamiany elementów w większości urządzeń elektronicznych
- można zapewnić teoretycznie pełną niezawodność programu poprzez przeprowadzenie dowodu poprawności (ang. *proof of correctness).*

BŁĘDY PROGRAMOWE

Błędy programowe można sklasyfikować ze względu na różne kryteria, a mianowicie:
- ze względu na fazę tworzenia oprogramowania, w czasie której zostały one wykryte (fazy: wstępna, definiowania wymagań projektowania logicznego, kodowania, testowania, eksploatacji)
- ze względu na metodę, za pomocą której zostały one wykryte
- ze względu na ich znaczenie dla pracy systemu: katastrofalne (wyniki programu różnią się znacznie od wartości prawidłowych, określonych w zbiorze wymagań; w związku z czym wykonanie programu zostaje zawieszone), poważne (mniejsze rozbieżności wyników w stosunku do wartości poprawnych, wydłużenie czasu wykonywania, itp.), marginalne (nie naruszające istotnych warunków ze zbioru wymagań).

Na podstawie danych obejmujących wiele systemów oprogramowania [2, 8] szacuje się, że ponad 60% błędów przy tworzeniu tych systemów popełnia się w fazie definiowania wymagań i projektowania logicznego, pozostałe zaś w decydującej części — podczas kodowania. Należy podkreślić, że im wcześniej wykryje się błąd, tym mniejsze są koszty jego usunięcia [8].

Można wymienić następujące praktyczne sposoby ograniczania liczby błędów programowych:
- dokładne testowanie programów
- ograniczenie rozmaitości logicznych struktur programu
- stosowanie określonych metod tworzenia oprogramowania (ang. *top-down, bottom-up- approach)*
- dowodzenie poprawności programu (ang. *proof of correctness).*

Weryfikacja programów jest bardzo pracochłonna. Aczkolwiek istnieją nieliczne systemy automatycznego dowodzenia poprawności, nie akceptują one programów napisanych w tak powszechnie stosowanych językach jak FORTRAN czy COBOL [9]. Ponadto nie można przeprowadzić weryfikacji programów, wykorzystujących dane przekazywane z czujników analogowo-cyfrowych (o nieokreślonych z góry wartościach, przy zmieniającym się stosunku sygnał-szum), ani analizować zmiennych z uwzględnieniem błędów obcięcia i zaokrąglenia [9].

Należy podkreślić, że można udowodnić zgodność programu ze specyfikacjami, a mimo to program może nie być wykonywany poprawnie (przy błędnych specyfikacjach). Ponadto sam proces dowodzenia poprawności programów jest bardzo podatny na błędy [9].

NIEZAWODNOŚĆ PROGRAMÓW

Podstawowym sposobem uzyskania wysokiej niezawodności programu jest dążenie do zmniejszenia liczby błędów. Niemniej, począwszy od lat 1968—1970, w związku z powstaniem koncepcji systemów cyfrowych tolerujących uszkodzenia — również w dziedzinie oprogramowania komputerów powstały nowe techniki, umożliwiające poprawne wykonywanie programu mimo występujących w nim błędów. Wykorzystują one różne postacie redundacji programowej, czasowej oraz mechanizmy protekcji [2, 9]. Ze stosowaniem tych technik związane są wysokie koszty dodatkowych programów, baz danych, czy bloków pamięci, a także problemy określenia ich skuteczności, problemy testowania itp. Skuteczna reakcja na błędy programu wymaga dostatecznie wczesnego ich wykrycia, jeszcze przed ich wystąpieniem pod postacią defektów wykonania.

W szacowaniu niezawodności programów używane są:
- metody statystyczne, polegające na rejestrowaniu liczby błędów występujących w trakcie wykonywania programu oraz czasów ich wystąpienia; na podstawie tej statystyki można otrzymać estymatory szeregu parametrów niezawodnościowych
- estymacja, polegająca na rejestrowaniu błędów w oparciu o statystyczne metody próbkowania, a następnie szacowaniu na tej podstawie niezawodności programu; metodę tę wykorzystano w modelach niezawodnościowych opartych na hipergeometrycznym rozkładzie funkcji prawdopodobieństwa występowania błędów [1]
- przewidywanie, polegające na określaniu niezawodności na podstawie analogii z innymi programami, których niezawodność jest znana.

W celu szacowania niezawodności programów bardzo istotne jest utworzenie dostatecznie dokładnych, zweryfikowanych modeli niezawodnościowych.

Niezbędne dane o błędach programowych uzyskuje się najczęściej w toku długotrwałej, starannej rejestracji błędów, łącznie z czasami ich wystąpienia, podczas różnych faz tworzenia oprogramowania. Związane są z tym duże koszty i nakłady pracy. Z wielu względów niechętnie publikuje się tego rodzaju dane. Autorowi nie są znane żadne statystyki tego typu opracowane w kraju.

Inną metodą uzyskania statystyk błędów programowych jest badanie określonych, specjalnie w tym celu napisanych programów (ang. *benchmark programs)* [8]. Chociaż programy takie są na ogół niewielkie, a więc i niezbyt kosztowne, to jednak wywołują wiele zastrzeżeń ze względu na obserwowaną rozbieżność pomiędzy niezawodnością programów eksperymentalnych i programów tworzonych w warunkach rzeczywistych. Poza tym programów tych nie eksploatuje się dostatecznie długo, aby mieć pewność, że wszystkie błędy zostały zaobserwowane.

KOREKCJA BŁĘDÓW PROGRAMOWYCH

U podstaw wszystkich niezawodnościowych modeli programu leży założenie o przypadkowym charakterze występowania błędów programowych. Wydaje się, że w przypadku programów dużych, wykorzystujących różnorodne strumienie danych wejściowych, określone części wykonywanego programu (w tym fragmenty zawierające błędy) aktywowane są w sposób przypadkowy, zależny od strumienia danych wejściowych.

Prawdopodobieństwo wystąpienia błędu programu w przedziale czasu Δt pod warunkiem bezbłędnego wykonania programu do chwili t wynosi:

$$\Pr\{t < t_f \leqslant t + \Delta t \mid t_f > t\} = \lambda(t)\,\Delta t \qquad \Delta t \to 0$$

gdzie:

$\lambda(t)$ — funkcja intensywności błędów programu
t_f — czas wystąpienia błędu.

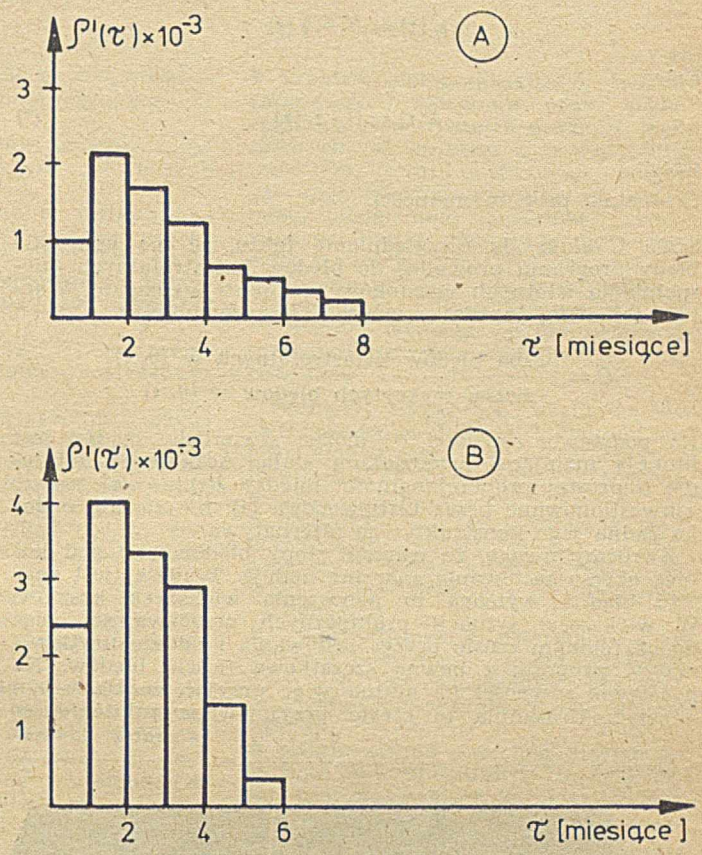
Oznaczmy jako $\varrho(\tau)$ liczbę błędów programu wykrytych w przedziale czasu ΔT (np. w ciągu miesiąca lub tygodnia) po czasie testowania i korekcji τ. Wielkość $\varrho(\tau)$ nazwiemy stopą błędów (ang. *error rate*). Na podstawie danych doświadczalnych, uzyskanych przez Shoomana [7] i potwierdzonych przez innych autorów, można stwierdzić charakterystyczne zmiany stopy błędów w funkcji długości okresu testowania.

Na rys. 1 podano (w oparciu o [7]) wykres **znormalizowanej stopy błędów** $\varrho'(\tau)$

$$\varrho'(\tau) = \frac{\varrho(\tau)}{I}$$

gdzie:

I — liczba instrukcji programu.



Rys. 1. Przebieg zmian $\varrho(\tau)$ w funkcji czasu testowania dla programu systemu operacyjnego (A) i programu użytkowego (B).

Przyjęto $\Delta T = 1$ *miesiąc*. Znormalizowaną stopą błędów posłużono się, aby uniezależnić proces występowania defektów wykonania od wielkości programu. Zarówno w przypadku programów systemu operacyjnego, jak i programów użytkowych widać wyraźnie, że po początkowym wzroście stopy błędów ulega ona następnie stopniowemu zmniejszeniu (w sposób bliski wykładniczemu) [7]. Można podać kilka hipotez tłumaczących taki właśnie przebieg $\varrho'(\tau)$, ale brak jest dostatecznych danych do ich potwierdzenia. Najważniejszy jest wniosek, że stopa błędów po 50 ÷ ÷ 60% okresu testowania ulega stopniowemu zmniejszeniu (niektórzy autorzy przyjmują, że $\varrho(\tau)$ maleje już od chwili $\tau = 0$ [3]). Ponadto, w miarę przedłużania testowania, wzrastają odstępy czasu pomiędzy momentami wystąpienia defektów wykonania, zaś liczba błędów w programie ulega systematycznemu zmniejszeniu (przy założeniu, że intensywność korekcji błędów jest większa od intensywności błędów wprowadzanych przypadkowo w okresie testowania i korekcji).

W związku z tym wydaje się uzasadnione założenie, powszechnie przyjmowane przy tworzeniu modeli niezawodnościowych programu, że funkcja intensywności błędów programu jest wprost proporcjonalna do liczby błędów w programie.

Oznaczając jako $\varepsilon(\tau)$ skumulowaną liczbę błędów wykrytych w okresie czasu $(0, \tau)$ otrzymamy:

$$\varepsilon(\tau) = \int_0^{\tau} \varrho(x)\,dx$$

zaś

$$\varrho(\tau) = \frac{d}{d\tau}\varepsilon(\tau)$$

jest nachyleniem krzywej $\varepsilon(\tau)$.

Jeśli $\varepsilon_r(\tau)$ będzie oznaczać liczbę błędów w programie po czasie τ, to przy założeniu, że początkowa liczba błędów jest równa $N = const$, otrzymamy:

$$\varepsilon_r(\tau) = N - \varepsilon(\tau)$$

zaś

$$\lambda(\tau) = C \cdot \varepsilon_r(\tau) \qquad (*)$$

gdzie:

C — stała proporcjonalności.

Stała C służy do uwzględnienia faktu, że nie wszystkie błędy programu prowadzą do błędów katastrofalnych (niespełnienia istotnych warunków ze zbioru wymagań). Wartość tej stałej można oszacować [7] z równości:

$$C = \frac{\text{liczba błędów katastrofalnych w } (0, \tau)}{\text{liczba wykrytych błędów w } (0, \tau)}$$

Na podstawie równości (*) można stwierdzić, że $\lambda(\tau)$ jest funkcją malejącą, przedziałami stałą. Aczkolwiek założenie o prostej proporcjonalności między $\lambda(\tau)$ a $_r(\tau)$ zostało zakwestionowane przez Littlewoode'a [4], nie została podana żadna jego konstruktywna alternatywa.

Zwróćmy uwagę, że wartość stopy błędów $\varrho(\tau)$ pod koniec okresu testowania znacznie maleje. Trudniej jest znaleźć błąd i wymaga to ponoszenia większych kosztów. W większości sytuacji praktycznych przerywa się więc po określonym czasie proces testowania i oddaje użytkownikowi program z pewną szczątkową ilością błędów. Na podstawie równości (*) można więc przyjąć, że $\lambda(\tau) = \lambda = const$. Uzasadnia to często przyjmowane założenie, że po okresie testowania współczynnik intensywności błędów programu jest stały. Również w normie „Komputery. Niezawodność. Wymagania ogólne" [5] przyjęto to założenie.

Bardzo istotne jest określenie współczynnika λ, oszacowanie niezawodności programu, średniego czasu między błędami, jak też oszacowanie niezbędnego wydłużenia czasu testowania, w celu uzyskania określonych wartości parametrów niezawodnościowych. Znalezieniu rozwiązań tych i podobnych problemów służą modele niezawodnościowe programów, których powstało kilka w ciągu minionych dziesięciu lat. Zostaną one przedstawione w następnym numerze INFORMATYKI

LITERATURA

[1] Basin S. L.: Measuring the Error Content of Software, Science Applications, Inc., Paolo Alto, CA, September 1974; cyt. za: Schick G. J., Wolverton R. W.: An Analysis of Competing Software Reliability Models. IEEE Transactions on Software. Enginnering, nr 2, 1978

[2] Boehm B. W.: Software Engineering, IEEE Transactions on Computer, nr 12, 1976

[3] Goel A. L.: Reliability and Other Performance Measures of Computer Software, II Procedings of Relcomex'79, Zamek Książ, 1979

[4] Littlewood B.: How To Measure Software Reliability and How Not To, IEEE Transactions on Software Engineering, nr 2, 1979

[5] Norma branżowa „Komputery Niezawodność Wymagania ogólne". BN-78,3108-03

[6] Shooman M. L.: Software Reliability. Analysis and Prediction. w: Henley E. J., Lynn J. W.: Generic Techniques in Systems Reliability Assessment, Noordhoff-Leyden, 1976

[7] Shooman M. L.: Probabilistic Models for Software Reliability Prediction. w: Freiberger W.: Statistical Computer Performance Evaluation. Academic Press, New York, London, 1972

[8] Soi I. M., Gopal K.: Error Prediction in Software. Microelectronics and Reliability, vol. 18, nr 5, 1978

[9] Soi I. M., Gopal K.: Some Aspects of Reliable Software Packages. Microelectronics and Reliability, vol. 19, nr 4, 1979

Biblioteka oprogramowania użytkowego

W ramach działań mających na celu zwiększenie stopnia wykorzystania maszyn cyfrowych MERA-ELWRO — producent i zarazem generalny dostawca systemów komputerowych zorganizował w 1977 r. Centralną Bibliotekę Oprogramowania Użytkowego. Jednym z zadań realizowanych przez Bibliotekę jest między innymi gromadzenie i dystrybucja informacji związanych z rozpowszechnianiem systemów informatycznych. Biblioteka prowadzi ewidencję oprogramowania wykonanego i wdrożonego oraz oprogramowania będącego w trakcie opracowywania. Ten drugi rodzaj ewidencji stanowi nowy jakościowo element w ramach procesu rozpowszechniania oprogramowania powtarzalnego. Ewidencja ta powinna być wykorzystywana głównie przez jednostki prowadzące koordynację prac informatycznych.

W 1979 r. w ogólnej liczbie 1523 ośrodków informatyki, 486 ośrodków (31,8%) prowadziło prace koordynacyjne. Średnio na każdy resort gospodarki narodowej przypada około 20 jednostek zajmujących się koordynacją prac z zakresu informatyki. Powoduje to duże zróżnicowanie prowadzonych prac, trudności w opracowywaniu i wdrażaniu systemów resortowych, opóźnienie w realizacji wspólnych przedsięwzięć. Resort przemysłu maszynowego, liczący najwięcej ośrodków informatyki, ma aż 70 jednostek [1]) zajmujących się koordynacją prac. Tworzone systemy nie mogą być w takiej sytuacji — spójne. [2])

Centralna Biblioteka Oprogramowania Użytkowego, mająca na celu uporządkowanie spraw związanych z ewidencją oprogramowania, przeprowadziła w latach 1978—1979, w ośrodkach informatycznych resortu przemysłu maszynowego, odpowiednie badania ankietowe, mające na celu:

— zaewidencjonowanie eksploatowanego w tych ośrodkach oprogramowania użytkowego
— zaewidencjonowanie tematów z informatyki, nad którymi prowadzone są prace.

Informacje otrzymane w wyniku badań ankietowych [3]) pozwoliły na opracowanie:

- Katalogu Oprogramowania, zawierającego opisy ponad 600 systemów informatycznych
- Katalogu Systemów w Opracowaniu, zawierającego opisy około 140 systemów.

Tak znaczna liczba systemów będących w trakcie opracowywania świadczy o zjawisku niekorzystnym, bowiem tworzone są one z reguły dla obsługi klasycznych już niemal dziedzin przetwarzania. Zjawisko to wynika również z braku rzeczywistej koordynacji.

Dotychczasowe doświadczenia pozwalają określić główne problemy Centralnej Biblioteki Oprogramowania Użytkowego:

- **Problem stworzenia mechanizmu zapewniającego stały dopływ informacji o tworzonych systemach informatycznych.** Z uwagi na fakt, że dotychczas podejmowane działania w tej dziedzinie nie przyniosły oczekiwanych rezultatów [4]), prowadzenie ewidencji systemów przez Centralną Bibliotekę Oprogramowania Użytkowego ma znaczenie zasadnicze. Jednak ze względu na brak odpowiednich uprawnień, ewidencja ta ogranicza się w praktyce do przemysłu maszynowego, do dostarczania Bibliotece informacji o eksploatowanych u siebie systemach. Powinny być zatem wydane odpowiednie zalecenia bądź przepisy, zobowiązujące wszystkie ośrodki informatyki do dostarczania Bibliotece informacji o eksploatowanych u siebie systemach. Zagadnienie to powinno być uregulowane odpowiednimi zarządzeniami Komitetu Informatyki.
- **Problem stałej aktualizacji ewidencjonowanych systemów oraz wykorzystania odpowiednich środków technicznych.** W związku z postępującym wzrostem liczby zastosowań, potrzebą uporządkowania ewidencji systemów, udzielaniem informacji coraz większej liczbie użytkowników — następuje konieczność zastosowania w Bibliotece systemu komputerowego oraz techniki mikrofilmowej, co jednocześnie wymaga przeprowadzenia intensywnego przeszkolenia jej pracowników.

Problemy dotyczące użytkowników związane są natomiast z pewnymi nieprawidłowościami ich współpracy z Biblioteką Oprogramowania. Najważniejsze z nich to:

- niedostarczanie w terminie i zgodnie z wymaganiami (w ramach prowadzonych badań ankietowych) informacji o opracowanych i wdrożonych systemach
- nieaktualizowanie informacji o systemach poprzednio zaewidencjonowanych (co jest szczególnie ważne w przypadku zaniechania eksploatacji systemu)
- podejmowanie prac nad nowym systemem, bez uprzedniego sprawdzenia lub zapoznania się z istniejącymi rozwiązaniami
- odmówienie udzielania informacji na temat eksploatowanych u siebie systemów (trzeba przyznać, że są to przypadki sporadyczne).

Stale wzrastająca liczba systemów informatycznych, wzrost ilościowy i rozbudowa instalacji komputerowych, zwiększenie ilości użytkowników, rozwój systemów teleprzetwarzania oraz rosnące wymagania w stosunku do efektywności zastosowań — są czynnikami stymulującymi budowę i rozwój Centralnej Biblioteki Oprogramowania Użytkowego. Biblioteka powinna stale współpracować z odpowiednimi jednostkami centralnymi oraz resortowymi koordynującymi prace informatyczne, dostarczając im informacji zarówno syntetycznych, jak i przekrojowych.

Elżbieta KIERCZUK
CKSAiP MERA-ELWRO
Wrocław

[1]) Źródło: Informatyka i ośrodki informatyki 1979. Materiały Statystyczne, GUS OBR SPIS, Warszawa, czerwiec 1980

[2]) W ciągu ostatnich trzech lat w resorcie przemysłu maszynowego liczba jednotek koordynujących prace informatyczne wzrosła o 24

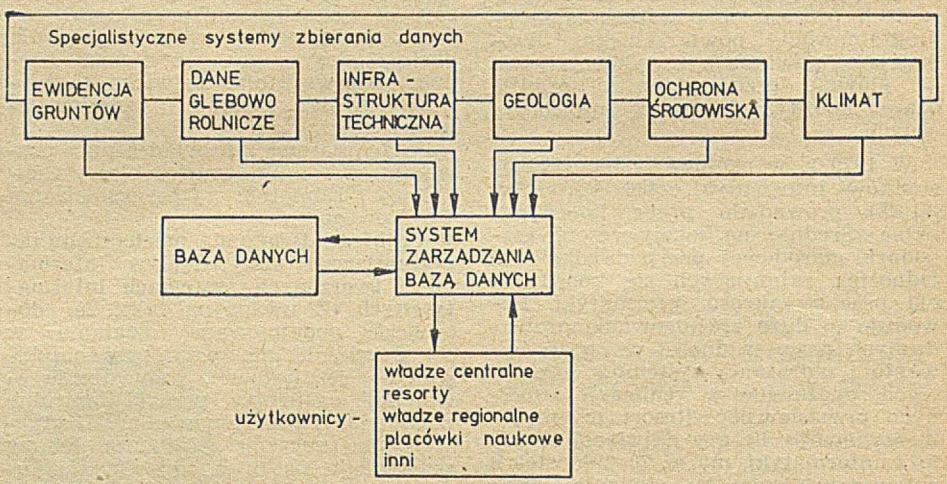
[3]) W trakcie prac wykorzystano pakiet automatyzacji prac bibliotekarskich MARC

[4]) Np. Zarządzenie nr 5/70 Pełnomocnika Rządu ds. Elektronicznej Techniki Obliczeniowej z dnia 17 II 1970 r.

ZETO Lublin wraz z pięcioma ośrodkami naukowymi i innymi zainteresowanymi instytucjami w kraju opracował koncepcję Państwowego Systemu Informatycznego TEREN, który w założeniach pozwala na optymalne wykorzystanie ziemi, służy planowaniu i zarządzaniu gospodarką rolną. Od tej pory (a minęło już dziewięć lat) udało się zrealizować tylko jeden, opisany poniżej podsystem, który ma przede wszystkim charakter ewidencyjny. Nie zostały ponadto, jak dotąd, uporządkowane inne, mniejsze systemy przeznaczone dla gospodarki ziemią. W obecnej sytuacji pilnego poszukiwania sposobów uzdrowienia naszej gospodarki rolnej warto byłoby podjąć sprawę na nowo i rozpocząć działania mające na celu rozwinięcie tego kierunku zastosowań (w takim zakresie, na jaki pozwalają obecne środki techniczne). Doświadczenia EWGRUNU mogą w tym znacznie pomóc. (Red.)

EWGRUN—podsystem ewidencji gruntów

Doświadczenia zagraniczne potwierdzają, iż metody informatyczne dają możliwość osiągnięcia zasadniczych celów, jakie stawiane są przed współczesną ewidencją gruntów, tj. zapewniają uzyskiwanie aktualnych informacji o stanie zagospodarowania terenu [1]). Również w Polsce zaistniała konieczność utworzenia krajowego systemu, który gromadziłby dane geodezyjno-kartograficzne o terenie, jego zasobach, zagospodarowaniu i innych istotnych właściwościach. System taki umożliwiałby waloryzację cech istniejących zasobów terenu (naturalnych i wytworzonych przez człowieka) w celu ich optymalnego wykorzystania oraz określenia potrzeb i możliwości ich przekształcenia. Koncepcję państwowego systemu informatycznego TEREN [2]), mającego spełnić te założenia, opracowano w 1972 r. Koncepcja ta została przedstawiona schematycznie na rysunku.



Schemat ideowy bazy danych w systemie TEREN

W ZETO Lublin od wielu już lat prowadzone są prace nad problemami użytkowania ziemi. Podjęte tu prace koncepcyjne są kontynuowane w Centrum Informatycznym Geodezji i Kartografii w Warszawie, w ramach systemu informatycznego GEOKART. W konstrukcji tego systemu wyróżnić można cztery moduły:

- **zbieranie danych terenowych** w zakresie ewidencji gruntów, dokonywane w procesie zakładania i aktualizacji mapy zasadniczej
- **gromadzenie danych numerycznych** (współrzędne punktów, pola działek, użytków, konturów klasyfikacyjnych, kompleksów glebowo-rolniczych), niezbędnych dla ewidencji gruntów
- **sporządzanie kartograficznej dokumentacji** ewidencji gruntów na podstawie przetworzonych zbiorów danych numerycznych w postaci mapy ewidencyjnej oraz określonych form wyników graficznych, m.in. w postaci szkiców użytkowych
- **przetwarzanie danych związanych z ewidencją gruntów,** danych, które umożliwiają założenie zbiorów pierwotnych, ich aktualizację, emitowanie zespołu tabulogramów wynikowych w postaci rejestrów, zestawień i wykazów gruntów na potrzeby użytkowników resortowych, administracji terenowej oraz użytkowników indywidualnych (część opisowa dokumentacji ewidencji gruntów).

Podstawową częścią systemu GEOKART jest ostatni z wymienionych modułów, który zaprojektowany został jako podsystem ewidencji gruntów — EWGRUN. Powstał on w wyniku ścisłej współpracy ZETO Lublin i Okręgowego Przedsiębiorstwa Geodezyjno-Kartograficznego w Lublinie [3]). Zadaniem podsystemu jest zapewnienie użytkownikom pełnej i aktualnej informacji o stanie zagospodarowania i sposobie użytkowania terenu, a także dostarczenie odpowiednich informacji dla rządowych systemów informatycznych: (SPIS, PESEL), systemów resortowych: BAZROL (Ministerstwa Finansów i ŚRODOWISKO GLEBOWE (Ministerstwa Rolnictwa) oraz systemów obiektowych.

Podsystem EWGRUN tworzy zbiory aktualnych danych z zakresu ewidencji gruntów, ściśle powiązanych z treścią mapy zasadniczej kraju, przetwarza je zgodnie z potrzebami użytkowników (np. dla wybranych jednostek, gmin lub całego województwa) oraz prezentuje wyniki w postaci tabulogramów — okresowo lub na żądanie (po wprowadzeniu do podsystemu danych numerycznych, wyniki otrzymywane są w postaci wykresów bądź map).

Z podsystemu mogą korzystać: jednostki administracji terenowej wszystkich szczebli, jednostki resortów (rolnictwa, finansów, sprawiedliwości, gospodarki terenowej i ochrony środowiska, leśnictwa i przemysłu drzewnego), terenowe komisje planowania

[1]) „Der Kommission 3 FIG" — Landinformationssysteme — Osterreichische Zeitschift fur Vermess, 1979/3

[2]) Wstępne założenia projektowe do państwowego systemu Informatycznego TEREN, ZETO Lublin, 1972

[3]) Nagroda II stopnia Ministra Administracji, Gospodarki Terenowej i Ochrony Środowiska za opracowanie i wdrożenie informatycznego podsystemu ewidencji gruntów dla zespołu autorskiego, Warszawa, 1979

gospodarczego, urzędy statystyki państwowej, rządowe systemy informatyczne (PESEL, SPIS), a także użytkownicy indywidualni.

CHARAKTERYSTYKA PODSYSTEMU

EWGRUN przetwarza zbiory informacji o ziemi, które obecnie zgromadzone są w dokumentach o charakterze ewidencyjnymi i statystycznym. Zawartość tych zbiorów jest następująca:

- **zbiór informacji podstawowych:**

— stosunki własności i władania ziemią
— klasyfikacja gleboznawcza gruntów
— powierzchnia działek i użytków
— powierzchnia gruntów wg gospodarstw
— powierzchnia gruntów wg grup rejestrowych

- **zbiory dodatkowych informacji specjalistycznych:**

— charakterystyka budynków zlokalizowanych na działkach ewidencyjnych
— przydatność glebowo-rolnicza
— erozja gruntów
— rekultywacja gruntów
— stan zmeliorowania
— szczegółowa charakterystyka sytuacji prawnej nieruchomości.

Informacje o gruntach grupowane są wg cech charakteryzujących poszczególne ich rodzaje (np. fizyczno-przyrodnicze, prawne, obszarowe). Identyfikacja i kodowanie danych w podsystemie realizowane są następująco:

- **identyfikacja administracyjno-geodezyjna;** podstawowym identyfikatorem jest numer obrębu [4]) ;wprowadzono zasadę numeracji obrębów według propozycji GUS, dotyczącej Systemu Identyfikacji Miejscowości

- **identyfikacja geodezyjno-kartograficzna;** położenie działki określa się współrzędnymi x, y wyróżnionego punktu wewnątrz działki, zarejestrowanymi z dokładnością 0,1 m; współrzędne te podaje się w obowiązującym układzie współrzędnych geodezyjnych, stosowanym przy sporządzaniu dokumentacji kartograficznej ewidencji gruntów

- **identyfikacja i symbolika danych systemowych;** symbolika stosowana w podsystemie EWGRUN jest tak określona, aby można było wykorzystać zbiory w innych systemach informatycznych, np. ŚRODOWISKO GLEBOWE czy BAZROL (podstawowe identyfikatory podsystemu zastosowano również w obu tych systemach).

Podsystem EWGRUN może być eksploatowany na komputerach serii ODRA 1300 w standardowej konfiguracji taśmowej. Ponadto eksploatacja może odbywać się pod nadzorem systemu operacyjnego GEORGE-2. Oprogramowanie wykonano w języku COBOL. Podsystem umożliwia wprowadzenie danych źródłowych na kartach dziurkowanych 80-kolumnowych, 8-ścieżkowej taśmie dziurkowanej lub taśmie magnetycznej. Podstawowym nośnikiem są karty dziurkowane. Rejestrację danych na taśmie magnetycznej, wraz z kontrolą przy użyciu minikomputera GEO-20, zastosowano aktualnie w Okręgowym Przedsiębiorstwie Geodezyjno-Kartograficznym w Lublinie.

Proces przetwarzania zawiera zakładanie i aktualizację słownika nazw i symboli, zbioru konturów i zbioru podstawowego, przetwarzanie zbioru „Kartoteka osób władających" (z wydrukiem alfabetycznym skorowidza władających), wydruk tabulogramów użytkowych w postaci wykazu działek, rejestrów, zestawień i wykazu gruntów, przetwarzanie i wydruk tabulogramów z wykorzystaniem danych o władaniu poza obrębem (wydruk rejestru zbiorczego miasta lub gminy) oraz zakładanie i aktualizację zbioru z zakresu ochrony użytków rolnych.

W ramach takiej struktury procesu przetwarzania wyodrębniono 18 zadań i 74 przebiegi składające się na całość eksploatacji podsystemu. Zadanie obejmuje funkcję jednego lub kilku programów wykonywanych w logicznej kolejności, natomiast funkcję jednego programu realizuje się za pomocą jednego lub kilku przebiegów. Podział na przebiegi dokonany jest w zależności od postaci zbioru (na wejściu) oraz typu instrukcji operowania zbiorem. Koordynator podsystemu (ze strony użytkownika) określa numery przebiegów, według których odbywa się eksploatacja.

W procesie przetwarzania tworzone są następujące zbiory:

- **zbiory stałe:**

— podstawowy
— słownik
— konturów klasyfikacyjnych
— danych z zakresu ochrony użytków rolnych
— danych do wydruku wykazów gruntów

- **zbiory robocze:**

— dokumentów źródłowych
— działek
— danych do rejestru zbiorczego
— danych do wydruku zestawienia gruntów
— „Kartoteka osób władających" do wydruku skorowidza alfabetycznego władających.

Przetwarzanie prowadzone jest w sposób sekwencyjny.

Najbardziej istotnym przebiegiem w procesie przetwarzania jest zakładanie i aktualizacja zbioru podstawowego, stanowiącego podstawę większości wydawnictw podsystemu. Zbiór ten zawiera aktualny układ informacji, posortowanych i zapisanych w następującej kolejności: nr obrębu, nr jednostki rejestrowej, lp. władającego nr działki. Zbiór podstawowy jest zbiorem wieloszpulowym. Na jednej szpuli taśmy magnetycznej zapisane są przeciętnie cztery gminy.

Zakładanie i aktualizacja zbioru podstawowego prowadzone są przy użyciu tych samych dokumentów źródłowych. Zmienia się jedynie kod operacji. W trakcie przetwarzania wyprowadzana jest taśma sterująca, informująca o rozmieszczeniu jednostek ewidencyjnych w zbiorze oraz dacie ostatniej aktualizacji. Z każdej aktualizacji zbioru tworzony jest raport zawierający sygnalizację błędów (kompletacji, niezgodności powierzchni, przebiegu aktualizacji) i zapisu stanu informacji dotychczasowej oraz zmienianej. W trakcie aktualizacji można kasować, zakładać lub modyfikować działki oraz jednostki rejestrowe.

Wyniki wyprowadzane są za pomocą drukarki wierszowej w postaci następujących tabulogramów:

- **dla pojedynczego obrębu:**

— wykaz działek
— wykaz współrzędnych działek
— skorowidz alfabetyczny władających
— rejestr gruntów
— rejestr gruntów Skarbu Państwa (wg określonych grup rejestrowych)
— zestawienie gruntów z uwzględnieniem dzierżaw
— wykaz gruntów
— rejestr gruntów podlegających rekultywacji i zagospodarowaniu

- **dla gminy lub miasta:**

— łączny rejestr gruntów z uwzględnieniem całkowitych powierzchni gospodarstw na danym obszarze
— skorowidz alfabetyczny władających
— wykaz gruntów
— zestawienie powierzchni terenów osiedlowych
— zestawienie powierzchni kompleksów przydatności glebowo-rolniczej

- **dla województwa:**

— wykaz gruntów z uwzględnieniem sumarycznych powierzchni ewidencyjnych i geodezyjnych.

W rejestrze gruntów obszarów użytkowanych rolniczo uwzględniona jest możliwość określenia wartości szacunkowej gospodarstw dla potrzeb scaleniowych. System umożliwia (w wąskim zakresie) uzyskiwanie dodatkowych informacji o dowolnej jednostce rejestrowej lub działce w relacji pytanie-odpowiedź. Pytania wprowadzane są za pomocą kart parametrycznych. W trakcie bieżącej aktualizacji drukowane są wyłącznie te strony rejestru, które uległy zmianie, oraz sumy dla obrębu.

DOTYCHCZASOWE DOŚWIADCZENIA I ROZWÓJ PODSYSTEMU

Aktualnie zakończono wdrożenie podsystemu na obszarze wybranych 50 miast i czterech gmin z terenu całego kraju. Założone zbiory aktualizowane są przeciętnie dwa razy w roku. Złożoność eksploatacji podsystemu wynika przede wszystkim z terenowego rozproszenia jednostek przygotowujących i aktualizujących dane, jakimi są miejskie pracownie geodezyjne, zakłady terenowe OPGK oraz Wojewódzkie

[4]) Obręb — powierzchnia gruntów, zawarta w granicach wsi, w mieście, dzielnicy lub osiedlu, o obszarze ok. 100 ha; mieści się na jednym arkuszu mapy formatu A1

Biura Geodezji. Zastosowana technologia znacznie usprawnia proces aktualizacji i uzyskiwania danych wynikowych (rocznych bilansów terenowych, zestawień).

Z aktualnie prowadzonych prac wdrożeniowych wynika, że zastosowanie metod informatycznych w procesie zakładania zbiorów ewidencyjnych nie zmniejszy, w porównaniu z techniką tradycyjną, związanych z tym kosztów. Jednak poszerzony zakres informacyjny podsystemu EWGRUN, możliwość emitowania tabulogramów w dowolnym układzie, łatwość aktualizacji zbiorów oraz jednoczesne wyprowadzenie wszystkich bilansów (szczególnie istotne w okresie sprawozdawczości) mają znaczny wpływ na podwyższenie efektywności prac ewidencyjnych. Należy wziąć pod uwagę choćby fakt, że eliminują one takie mankamenty systemu tradycyjnego, jak: brak jednolitości materiałów, mała czytelność operatów (szczególnie po kilku latach bieżącej aktualizacji) oraz stosunkowo liczne błędy rachunkowe.

Podsystem EWGRUN będzie dla następnych wielodziedzinowych systemów związanych z gospodarką ziemią pierwszym etapem warunkującym ich wdrożenie i eksploatację. Przykładem tego jest system BAZROL, który będzie bezpośrednio zasilany z podsystemu EWGRUN. Aktualnie trwają w ZETO Lublin prace nad dostosowaniem podsystemu EWGRUN do bezpośredniego zasilania zintegrowanej bazy danych BAZROL. Szacuje się, że rozwiązanie takie da oszczędności rzędu dziesiątków milionów złotych w skali województwa.

Jak już wspomniano, konstrukcja podsystemu EWGRUN i zastosowane w nim identyfikatory pozwalają na współpracę z innymi systemami resortowymi czy obiektowymi. Między innymi przewiduje się w perspektywie wykorzystanie szczegółowych informacji przestrzennych podsystemu EWGRUN w systemach: SRODOWISKO GLEBOWE, FIZGLEB i CHEMGLEB, projektowanych w ZETO Lublin na zlecenie Zakładu Agrofizyki PAN w Lublinie. Systemy te zawierają szczegółowe informacje dotyczące cech jakościowych i właściwości fizykochemicznych gleb Polski.

Integracja stwarza olbrzymie możliwości wszechstronnego wykorzystania tych systemów — poczynając od doradztwa nawozowego, poprzez optymalizację struktury zasiewów i upraw (na szczeblu wsi), aż do planowania i prognozowania produkcji rolnej (na szczeblu centralnym). Znaczne usprawnienie procesu eksploatacji oraz zwiększenie możliwości technologicznych podsystemu EWGRUN uzyska się po zastosowaniu Systemu Zarządzania Bazą Danych RODAN i transmisji danych. Prace w tym kierunku zostały już podjęte w ZETO Lublin.

mgr inż. Michał Kosiński
ZETO Lublin
mgr inż. Stanisław Zaremba
Okręgowe Przedsiębiorstwo Geodezyjno-Kartograficzne w Lublinie

ZWIĄZKI ZAWODOWE

Jak już informowaliśmy w poprzednim numerze,, 12 października br.' odbył się I Walny Zjazd NSZZ Pracowników Informatyki. Zjazd wybrał władze oraz podjął uchwałę, którą bez zmian publikujemy poniżej. (Red.)

Uchwała I Walnego Zjazdu

Niezależnego Samorządnego Związku Zawodowego Pracowników Informatyki

Informatyka jest branżą specyficznie różną od innych spotykanych w gospodarce narodowej i w życiu społecznym kraju. Istnienie tej branży, jako warunku sprawnego działania systemu społeczno-gospodarczego, nie dla każdego jest oczywiste. Nasz zawód czasami uważany jest za mało ważny.

Nawet na wysokie stanowiska kierownicze przenikają często ludzie bez odpowiedniego przygotowania, których działalność utrudnia nam życie. Przeważa u nas biurokratyczno-towarzyski styl i klimat zarządzania, wysoce niewłaściwy w naszej branży. Związek musi domagać się stosowania nowoczesnych, partnerskich metod zarządzania. Chcemy rozwijać siebie i swoją branżę, tak aby praca była bardziej wartościowa i ciekawa.

Związek ocenia stopień Umów Społecznych, zawartych pomiędzy Międzyzakładowymi Komitetami Strajkowymi a Komisjami Rządowymi, za niepełny. Jesteśmy zdania, że w zakresie naszej branży nie zrealizowano następujących uzgodnień:

— nie stworzono możliwości rozwoju nowego ruchu związkowego
— nie zrealizowano postanowień płacowych
— nie stworzono realnej możliwości publicznego opiniowania przez związki zawodowe kluczowych decyzji determinujących warunki życia ludzi pracy, a w szczególności podstawowych zasad wynagradzania i kierunków polityki płac w informatyce, zasady automatycznej korekty płac w warunkach inflacji, wieloletnich planów gospodarczych, kierunków inwestycji i zmian cen w informatyce
— nie podano do publicznej wiadomości (w środowisku informatycznym) pełnej informacji o sytuacji społeczno-gospodarczej w informatyce
— nie umożliwiono dyskusji nad programem reform gospodarczych w zakresie informatyki
— nie przeprowadzono weryfikacji kadry kierowniczej pod kątem kwalifikacji i kompetencji
— nie uregulowano spraw związanych z czterobrygadowym systemem pracy
— nie wprowadzono w systemie zmianowym zwiększenia dodatku za pracę na drugiej i trzeciej zmianie.

Jesteśmy zdania, że nasza ocena jest zgodna ze stanowiskiem Niezależnego Samorządnego Związku Zawodowego „Solidarność".

Zjazd zatwierdził następujący ramowy program działania:
1) bieżąca kontrola realizacji Porozumień Gdańskich
2) przygotowanie porozumień z innymi Niezależnymi Związkami Zawodowymi w zakresie:
— współpracy w ramach Nowego Ruchu Związkowego
— ochrony interesów informatyków należących do innych związków
3) powołanie komisji:
— badania dochodów środowiskowych i płac
— kwalifikacyjnej, ustalającej kwalifikacje pracowników na poszczególnych stanowiskach pracy
— do spraw organizacji i warunków pracy w przedsiębiorstwach informatycznych
4) realizacja wniosków przedstawionych na Zjeździe (spis wniosków stanowi załącznik do niniejszej uchwały).

Za zadanie pierwszoplanowe Zjazd uznał podwyżki płac dla wszystkich informatyków w kraju. Zakończenie wprowadzania podwyżek, zgodnie z Porozumieniem Gdańskim, winno nastąpić do dnia 30.09.1980. Związek stoi na stanowisku, że wszyscy informatycy w kraju winni być traktowani na tych samych warunkach.

Zjazd uchwalił składki członkowskie w wysokości minimalnej 1% od wynagrodzenia. Kwota powyżej minimalnej pozostaje do dyspozycji Rady Zakładowej, której członkowie uchwalili wyższą składkę. Jeden procent jest dzielony pomiędzy Radę Zakładową Zarząd Związku w relacji 75% i 25%.

SPIS WNIOSKÓW:

1. Doprowadzić do stanu aby czasopismo INFORMATYKA było współredagowane przez NSZZPI [1]). Jako pierwsze stadium już od dnia 01.11.1980 r. należy wydzielić jedną kolumnę dla aktualnych problemów NSZZPI.

2. Opracować Kartę Zawodu Informatyka (np. na wzór Karty Nauczyciela, Górnika, Hutnika itp.), która w sposób jednoznaczny określił specyfikę zawodu i wynikające z niej uprawnienia.

3. Opracować w terminie do dnia 01.12. 1980 r. układ zbiorowy dla pracowników informatyki.

4. Za pośrednictwem NSZZPI doprowadzić do integracji informatycznego środowiska naukowego i badawczo-rozwojowego. W tym celu proponuje się utworzenie towarzystwa wyższej użyteczności pod nazwą Stowarzyszenie Informatyków Polskich i nadanie mu odpowiedniej rangi.

5. Powołać społecznego inspektora pracy.

6. Wystąpić do Prezydenta m. st. Warszawy o przyznanie lokalu biurowego.

7. Wyegzekwować od dotychczasowych związków zawodowych środki finansowe dla prowadzenia własnej działalności (np. od energetyków).

8. Spowodować posiadanie własnego teleksu, telefonu i kopiarki.

9. Przeprowadzić podział majątku związkowego będącego w gestii dotychczasowych związków.

10. Rozpoznać możliwości i przedstawić propozycje porozumień z innymi NSZZ. Porozumienia powinny być zawarte na takich samych zasadach jak umowa federacyjna.

11. Nawiązać współpracę z innymi grupami informatycznymi działającymi poza związkiem.

12. Rady Zakładowe na terenie swoich instytucji, wraz z administracją przedsiębiorstw, winny opracować projekty reform przedsiębiorstw, zgodnie z duchem punktu szóstego Porozumienia Gdańskiego.

13. Związek nasz winien ustosunkować się do branży jako całości pod względem:

— istniejących metod zarządzania i kierowania tą branżą
— wypracowania modelu przedsiębiorstwa informatycznego
— poprawy jakości i wzmocnienia potencjału sprzętu informatycznego oraz jego ujednolicenie

[1]) INFORMATYKA chętnie przeznaczy część swoich łamów na materiały dotyczące ruchu związkowego. Sądzimy jednak, że redagowaniem pisma nadal powinno się zajmować Kolegium Redakcyjne (przyp. red.)

14. Zorganizować Nadzwyczajny Zjazd Związku w terminie nie później niż w dwa tygodnie po wejściu w życie ustawy o związkach zawodowych i nie później niż 31.01.1981.

15. NSZZPI powinien wziąć aktywny udział w pracach nad ustawą o związkach zawodowych aż do jej wprowadzenia.

16. Należy przyjąć statut w formie jaka została zarejestrowana w Sądzie Wojewódzkim w Warszawie, natomiast w ciągu 14 dni opracować szczegółową interpretację poszczególnych punktów statutu.

17. Żądamy opracowania nowego projektu podwyżek wynagrodzeń zgodnie z postanowieniami Porozumienia Gdańskiego dla wszystkich pracowników informatyki.

18. Związek w swojej działalności powinien dążyć do aktywnego rozróżniania potrzeb informacyjnych społeczeństwa od tzw. „standardu życiowego".

19. Związek powinien jak najszybciej określić zasady i rozpocząć działalność informacyjną, prawną, finansową i ewidencyjną.

20. W ramach prowadzenia polityki zatrudnieniowo-płacowej Związek powinien:

— dążyć do tego, aby zatrudniano pracowników zgodnie z ich kwalifikacjami

— dążyć do zlikwidowania szkodliwej praktyki ustalania planów zatrudnienia na poziomie roku ubiegłego

— dążyć do tego, aby wygospodarowane fundusze roku bieżącego były przenoszone na rok następny

— czynnie uczestniczyć przy opracowywaniu układów zbiorowych i innych aktów normatywnych z zakresu polityki płacowej i kadrowej

— dążyć do tego, aby płaca była wynagrodzeniem za rzeczywisty wkład pracy

— mając na uwadze średnie płace dążyć do zlikwidowania różnic płacowych pomiędzy pracownikami informatyki we wszystkich jednostkach gospodarczych w kraju.

21. W ramach usprawnienia gospodarki powinniśmy:

— dążyć do opracowywania takich zadań dla przedsiębiorstw naszej branży, które wynikają z potrzeb i specyfiki regionu

— dążyć do upowszechniania systemów dobrych, sprawdzonych, odpowiadających rzeczywistym potrzebom i warunkom użytkowników (systemy nabywane przez zjednoczenia i resorty powinny spełniać wymogi wszystkich przedsiębiorstw, a ich sprzedaż winna odbywać się systemem konkursowym przy współudziale jednostek autorskich)

— rozważyć możliwość utworzenia funduszu informatycznego na podobnych zasadach jak fundusz postępu technicznego

— dążyć do zapewnienia odpowiedniej bazy sprzętowej umożliwiającej efektywne wykorzystanie informatyki (nowoczesny sprzęt informatyczny powinien trafić przede wszystkim do wyspecjalizowanych w informatyce jednostek)

— dążyć do nowego, racjonalnego cennika usług informatycznych.

22. W ramach polepszania warunków pracy i warunków socjalno-bytowych Związek powinien:

— zmierzać do zapewnienia podstawowych warunków pracy poprzez niedopuszczanie do zaniechania budowy ośrodków już rozpoczętych bądź rozpoczynania budowy nowych

— dokładnie przeanalizować istniejące warunki pracy i przedstawić konkretne wnioski dotyczące ich poprawy oraz przestrzegać ich realizacji

— wypracować wytyczne co do zasad wcześniejszego przechodzenia na emeryturę

— spowodować zbadanie czynników wpływających szkodliwie na zdrowie, powodujących nerwicę, utratę wzroku i słuchu oraz spowodować przyznanie dodatków z tego tytułu.

23. Należy przeprowadzić normalizację produktów programowych pod względem ich wartości i autorstwa.

24. Należy wypracować zasady prawa regulujące działalność wynalazczą i racjonalizatorską w informatyce.

25. Związek powinien podnosić rangę Rad Zakładowych zgodnie z duchem statutu.

26. Związek powinien bronić praw i interesów pracowników informatyki nie należących do Związku, o ile wystąpią z taką prośbą.

27. Należy uwzględnić w układzie zbiorowym pracy istnienie innych grup pracowniczych.

28. Należy przeprowadzić weryfikację szkoleń zawodowych i wpłynąć, aby kierunki i poziom były ustalane właściwie.

29. Uchwała winna być przedstawiona:

— PRiTV
— NSZZ „Solidarność"
— Ministerstwu PPiSS
— Komitetowi Informatyki
— redakcjom „Życia Warszawy" i „Głosu Pracy".

* * *

W tekście uzupełniającym, przekazanym MNSzWiT, mającemu pieczę nad rozwojem informatyki w Polsce, czytamy między innymi:

Związek oczekuje wyjaśnienia poniższych problemów:

● Sytuacji społeczno-gospodarczej w informatyce:
— jaką rolę pełni informatyka w życiu społeczno-gospodarczym?
— dlaczego informatyka nie spełniła swojej roli w życiu społeczno-gospodarczym w Polsce?
— prosimy o ocenę stanu i analizę przyczyn w następujących aspektach: sprzęt, zastosowania (systemy krajowe, systemy obiektowe), oprogramowanie, ludzie a kwalifikacje, oragnizacja branży i jej elementów składowych, współpraca międzynarodowa (Polska a świat)

● Programu reform w informatyce:

— kto i w jakim terminie jest odpowiedzialny za przygotowanie takiego programu?

● Weryfikacji kadry kierowniczej pod kątem kwalifikacji i kompetencji:

— kto i do kiedy ustali kryteria kwalifikacyjne?

● Sprawy czterobrygadowego systemu pracy

— kto i w jakim terminie przeprowadzi analizę tego systemu pracy w branży informatycznej?

● Sprawy dodatku za pracę na drugiej i trzeciej zmianie

— w jakim terminie zostanie wprowadzony dodatek?

Zasady organizacji komórek związkowych

1. Utworzyć Komitet Założycielski
2. W oparciu o statut przeprowadzić nabór członków.
3. Zorganizować zebranie członków i przeprowadzić wybory Rady Zakładowej lub delegata związkowego zgodnie ze statutem.
4. Zgłosić się do Zarządu Związku w celu:
a) uzyskania materiałów programowych i organizacyjnych (deklaracje, sposób założenia konta bankowego, rodzaje pieczątek itp.)
b) podanie aktualnego stanu organizacyjnego (ilość członków oraz wysokość wpływów ze składek członkowskich).

Skład Zarządu Związku:

Przewodniczący — Bogdan Fiutowski
Zastępca przewodniczącego — Paweł Zieliński
Sekretarz — Marek Golanko
Skarbnik — Bogusław Piętka
Członkowie — Antoni Garbacik, Anatol Stefaniuk, Mariusz Stefanowski

---

Tymczasowy adres Związku:

00-608 Warszawa, al. Niepodległości 190
tel. 25 80 61 w. 60, telex 81 34 92 CPZI pl

KOMUNIKAT

W dniu 15.10.1980 r. odbyło się w Warszawie Kolegium Dyrektorów Zjednoczenia Informatyki z udziałem przedstawicieli załóg zakładów ZETO. W trakcie zebrania nastąpiły pertraktacje w sprawie podwyżek płac wynikających z POROZUMIEŃ GDAŃSKICH dla pracowników Zakładów ZETO. Ustalono podwyżkę z tytułu wzrostu kosztów utrzymania na nie mniej jak 500 zł. Termin wprowadzenia jest przedmiotem dalszych negocjacji. Pertraktacje wykazały, że jedyną drogą uzyskania rzeczywistej realizacji Porozumień w branży jest natychmiastowe zorganizowanie branżowego ruchu związkowego.

Postulaty kształcenia informatyków w Austrii

Problem przygotowania kadr dla informatyki nie jest właściwie rozwiązany nie tylko u nas, ale również w krajach bardziej zaawansowanych w komputeryzacji. Poznanie istniejących rozwiązań i propozycji zagranicznych może przyczynić się z pewnością do dalszego usprawnienia oraz zwiększenia efektywności kształcenia kadr dla informatyki polskiej. Poniżej zostanie przedstawiona propozycja austriacka opublikowana przez Martina Weissenboeka w nr 28 Komunikatów ÖCG (Austriackiego Towarzystwa Komputerowego) z czerwca br. [1]) **Wynika z niej, że w tej dziedzinie jesteśmy bardziej zaawansowani (szkoły średnie i pomaturalne o kierunku informatycznym uruchomiono w Polsce już 15 lat temu), natomiast zbyt mało wiemy o rezultatach realizowanej treści nauczania. Zachęcamy do wypowiedzi na ten temat (Red.).**

Na dzień 1.1.1979 wartość zainstalowanego w Austrii sprzętu komputerowego osiągnęła kwotę 16,2 mld szylingów (ok. 1,3 mld $), a do 1985 r. przewiduje się dalsze nakłady na jego zakup w kwocie 57,8 mld szylingów (ok. 4,5 mld $). W wyniku gwałtownego rozwoju mikroelektroniki oraz tak dużego zapotrzebowania sprzętu, rozpatruje się coraz częściej możliwość uruchomienia krajowej produkcji, zwłaszcza urządzeń peryferyjnych i mikroprocesorów. Wymaga to oczywiście odpowiednich kadr, których niestety nie przygotowuje aktualny austriacki system szkolnictwa.

AUSTRIACKI SYSTEM SZKOLNICTWA A INFORMATYKA

Przeprowadzona przez autora analiza wykazała, że aktualny system szkolnictwa jest całkowicie nieprzygotowany do zaspokojenia potrzeb tak szybkiego rozwoju informatyki. Mimo, że formalnie większość szkół wszystkich szczebli ma w swych programach nauczania zagadnienia informatyki, to jednak udział procentowy tego przedmiotu jest niewielki: w szkołach średnich ogólnokształcących na łączną liczbę 136 godzin tygodniowo (w skali całego okresu nauczania), przewiduje się na te zagadnienia wprawdzie osiem godzin, lecz tylko w ramach przedmiotów fakultatywnych. Gorzej jeszcze przedstawia się sytuacja w średnich szkołach zawodowych, gdzie z łącznej liczby 205 godzin zaledwie dwie godziny, a więc mniej niż 1%, przewidziano na informatykę. Nieco lepiej przedstawia się sprawa w akademiach handlowych [2]), gdzie na 159 godzin informatyce poświęca się osiem godzin (a więc ok. 5%). Przyjmując nawet, że liczby te są nieco większe dzięki przedmiotom fakultatywnym lub tzw. aktualnym dziedzinom specjalizacji zawodowej, powyższy zakres nauczania znacznie odbiega od potrzeb, jakie coraz szerzej zgłasza gospodarka narodowa.

Jeśli chodzi o kształcenie ściśle informatyczne, to o ile w szkołach wyższych można uznać go za zadowalające — na 16 uczelni kierunki informatyczne prowadzą dwa uniwersytety (Wiedeń i Linz) oraz dwie politechniki (Wiedeń i Graz) — to na szczeblu szkolnictwa średniego istnieją jedynie sporadyczne kursy przystosowania do zawodu (Wiedeń i Dolna Austria) oraz kursy dla maturzystów (Wiedeń). Uderza więc zupełny brak kształcenia informatycznego na poziomie szkoły średniej. Ponadto w strukturze treści nauczania panuje tu wielki konserwatyzm. Wynika to głównie z obowiązującego pięcioletniego cyklu modyfikacji programów (najbliższych zmian można oczekiwać dopiero w 1982 r.). Przy pięcioletnim cyklu kształcenia w średnich szkołach zawodowych, w najlepszym przypadku można więc spodziewać się absolwentów nowego, uwzględniającego lepiej potrzeby informatyki programu dopiero w 1987 r.! W tej sytuacji należy wątpić, czy realne są zamiary rozwoju krajowego przemysłu komputerowego, zdolnego do konkurowania z producentami zagranicznymi.

Autor proponuje więc możliwie szybkie podjęcie następujących działań zaradczych:

- wprowadzenia do różnych przedmiotów nauczania w programach zarówno średnich szkół ogólnokształcących, jak i zawodowych, tzw. ogólnych aspektów informatyki, uwzględniających metody planowania i realizacji oraz operowania złożonymi systemami. W treści nauczania oprócz aspektów gospodarczych należy uwzględnić skutki społeczne rozwoju techniki informacyjnej oraz mikroelektroniki;
- włączenie w średnich szkołach zawodowych do obowiązującego materiału dydaktycznego poszczególnych przedmiotów problematyki informatycznej. Postulat ten dotyczy wszystkich bez wyjątku kierunków kształcenia zawodowego;
- stworzenie w oparciu o istniejące wzorce kształcenia informatycznego wyższych uczelni nowych specjalistycznych placówek i kierunków kształcenia zawodowego, zarówno w zakresie zastosowań gospodarczych jak i naukowo-technicznych.

SZCZEGÓŁOWE PROPOZYCJE

Według niepotwierdzonych informacji od jesieni 1980 zostanie uruchomiona w Wiedniu średnia szkoła techniczna ukierunkowana na zastosowania gospodarcze informatyki. Niestety plany dydaktyczne nie mogły być przedyskutowane w ÖCG, ponieważ ich twórcy zasłonili się pretekstem tajemnicy służbowej.

Zgodnie z przeprowadzonymi przez ÖCG badaniami roczne zapotrzebowanie na informatyków ze średnim wykształceniem technicznym w dziedzinie zastosowań naukowo-technicznych wynosi co najmniej 200 absolwentów. Autor proponuje następujące cztery modelowe rozwiązania problemu na poziomie szkoły średniej oraz szkoły pomaturalnej, przygotowujące w zakresie specjalizacji nazwanej przez autora umownie „Elektrotechnika — informatyka" [3]):

1. Odrębne technikum lub wydział w istniejącej średniej szkole technicznej z pięcioletnim programem nauczania. Nauka rozpoczynać się będzie od klasy IX (uczniowie 14—15 letni) i kończyć maturą zawodową.

2. Specjalistyczna szkoła pomaturalna z programem pięciosemestralnym dla absolwentów średnich szkół ogólnokształcących lub zawodowych innych kierunków. Nauka kończy się drugą maturą i — podobnie jak inne tego rodzaju szkoły — po trzyletniej praktyce zawodowej daje uprawnienia do uzyskania tytułu inżyniera. Szkoły takie, prowadzone już od kilku lat dla specjalizacji elektrotechnika, budowa maszyn oraz budownictwo, uznano za bardzo udane rozwiązanie, chociaż jest ono jeszcze mało spopularyzowane (zrealizowano jedynie w formie państwowej szkoły eksperymentalnej w Wiedniu).

3. Technikum dla osób pracujących zawodowo. Przewiduje się tu roczny nieobowiązkowy kurs przygotowawczy

[1]) Weissenboeck M.: Informatik in Österreich — Vorschläge für berufsbildende Schulen (Informatyka w Austrii — propozycje w zakresie szkolnictwa zawodowego)

[2]) Akademie handlowe w Austrii nie zaliczają się do kategorii szkół wyższych

[3]) niem. „Elektrotechnik — Informatik", zapewnie dla łatwiejszej adaptacji przez urzędowy aparat szkolnictwa w oparciu o istniejący obecnie schemat nauczania w szkołach zawodowych elektronicznych (przyp. tłum.)

oraz czteroletni program nauczania. Zajęcia odbywają się w godzinach wieczorowych, a naukę kończy się egzaminem dojrzałości, dającym identyczne uprawnienia jak poprzednie rodzaje szkół.

4. Roczny kurs dla pracujących już zawodowo techników, mający na celu uzupełnienie posiadanej wiedzy o nowości postępu technicznego np. z zakresu mikroelektroniki. Wykłady, podobnie jak w rozwiązaniu scharakteryzowanym w poprzednim punkcie, odbywają się w godzinach wieczorowych.

Co należy rozumieć pod pojęciem specjalizacji „elektrotechnika — informatyka"? Istotą tej specjalizacji jest odpowiednio wyważony program nauczania, uwzględniający zarówno problematykę sprzętu, jak i oprogramowania i to nie tylko od strony teoretycznej, ale i praktycznej. Wybór właściwej specjalizacji, wzorem istniejących już rozwiązań w szkołach mechanicznych i budowlanych, powinien następować w sposób świadomy dopiero w trzecim roku nauczania, a więc dopiero przez uczniów 16—17-letnich (pierwsze dwa lata należy poświęcić wyłącznie przedmiotom ogólnokształcącym i ogólnotechnicznym). Łączny tygodniowy wymiar godzin należy ustalić na poziomie obowiązującym w innych średnich szkołach technicznych, a więc średnio 41 godzin tygodniowo (w ciągu pięciu lat łącznie 205 godzin). Ponieważ matura takiego technikum uprawnia do ubiegania się o przyjęcie na studia wyższe, należy uwzględnić szeroki zakres przedmiotów szkoły ogólnokształcącej (łącznie 87 godzin), a także przedmioty zajęć praktycznych, charakterystyczne dla średnich szkół technicznych (zajęcia warsztatowe i laboratoryjne — łącznie 42 godziny). Tak więc na specjalistyczne przedmioty kierunkowe pozostaje łącznie 76 godzin. Aby zachować spójność kierunku informatycznego z kierunkiem „elektrotechnika", na przedmioty kierunkowe w pierwszym i drugim roku nauczania należy zarezerwować łącznie 20 godzin, natomiast w pozostałych trzech latach nauczania — pozostałe 56 godzin. Te 56 godzin należy następująco podzielić na przedmioty o charakterze ogólnotechnicznym i ściśle informatycznym:

- 22 godziny — sprzęt komputerowy oraz mikroelektronika
- 22 godziny — technika oprogramowania oraz problemy zastosowań EPD i funkcje programowania
- 12 godzin — przedmioty ogólnotechniczne.

Wprowadzenie mikroelektroniki i techniki przetwarzania informacji powoduje oczywiście konieczność opracowania treści nauczania całkowicie nowych przedmiotów. Przykładowo autor proponuje w zakresie problematyki sprzętu następujący podział:

- elektronika i telekomunikacja (6 godzin)
- technika cyfrowa i technika połączeń (7 godzin)
- konstruowanie z ćwiczeniami (9 godzin).

W zakresie problematyki programowania propozycja obejmuje:

- programowanie z ćwiczeniami (10 godzin)
- zastosowanie oraz organizacja EPD z ćwiczeniami (10 godzin)
- translatory i systemy operacyjne (2 godziny).

Celem uzupełnienia treści przedmiotów ogólnotechnicznych oraz przedmiotów z zakresu elektrotechniki, nauczanych w ciągu pierwszych dwóch lat należy w ciągu pozostałych trzech lat uwzględnić:

- mechanikę jako kontynuację programu drugiego roku nauczania (1 godzina)
- podstawy elektrotechniki jako kontynuację programu pierwszych dwóch lat nauczania (3 godziny)
- miernictwo elektryczne ze szczególnym uwzględnieniem miernictwa z zakresu elektroniki oraz automatycznej rejestracji danych pomiarowych (2 godziny)
- maszyny i urządzenia elektryczne w ujęciu encyklopedycznym wzorowane na programach technikum telekomunikacyjnego i elektronicznego (6 godzin).

Propozycja programu zakłada obowiązkową, co najmniej czterotygodniową praktykę po ukończeniu drugiego i czwartego roku nauki. Ponadto przewiduje się jako przedmioty nadobowiązkowe: stenotypię (po dwie godziny w pierwszych dwóch latach) oraz „aktualne dziedziny specjalizacji zawodowej" (po dwie godziny w trzech ostatnich latach), a także ćwiczenia wf. (po dwie godziny w trzech pierwszych oraz po trzy godziny w dwóch ostatnich latach nauczania). Zajęcia nadobowiązkowe obejmują wreszcie nauczanie wspomagające z zakresu języka niemieckiego i angielskiego, matematyki i teoretycznych przedmiotów zawodowych (po jednej godzinie dla każdego z czterech przedmiotów w ciągu wszystkich lat nauczania).

REALIZACJA PLANÓW I AKTUALNA SYTUACJA

Pierwszy projekt powyższego programu średniej szkoły technicznej o kierunku „Elektrotechnika — informatyka" został w kwietniu 1979 przesłany szeregu firmom i zainteresowanym instytucjom z prośbą o opinię. Ankieta wykazała pilną konieczność utworzenia szkoły o proponowanym profilu zawodowym. W październiku i listopadzie 1979 r. przedstawiciele głównych koncernów przemysłu elektrotechnicznego wypowiedzieli się za pilną realizacją tych planów. W styczniu br. pozytywnie ustosunkowały się kompetentne komórki austriackiego ministerstwa oświaty i sztuki (niestety tylko w zakresie utworzenia wyodrębnionego kierunku specjalizacji). Niezależnie od tych propozycji kilka średnich szkół technicznych uzyska już formalne prawo zorganizowania w roku szkolnym 1980/81 tego rodzaju kierunków specjalizacji zawodowej o nieco zmodyfikowanej nazwie w formie kursów dziennych lub wieczorowych. W ciągu kilku tygodni od daty ogłoszenia odpowiedniej informacji, do wspomnianego już państwowego technikum eksperymentalnego w Wiedniu zgłosiło się 44 kandydatów co świadczy o dużym zainteresowaniu nowym kierunkiem nauczania, a także o istniejących w społeczeństwie dążeniach do rozszerzania kwalifikacji zawodowych.

Został więc dokonany pierwszy krok w dziedzinie informatycznego szkolnictwa zawodowego. Należy ufać, że zrealizowane zostaną również pozostałe propozycje. Realizację tę mogłoby przyśpieszyć zgłoszenie konkretnego zapotrzebowania przez zainteresowane firmy, co lepiej przygotowałoby gospodarkę narodową do wspomnianego zorganizowania krajowego przemysłu komputerowego.

Tłumaczył i opracował W. K.
na podstawie Mitteilungsblatt ÖCG
Nr 28 (czerwiec 1980 r.)

Analizator mowy bez „przyuczenia"

Postulat zbudowania głosowego wejścia do komputera (zastąpienia klawiatury czy taśm — mikrofonem) od dawna podniecał wyobraźnię elektroników. Nie ulega wątpliwości, że komputer rozumiejący mowę ludzką byłby nieprzeciętnym ułatwieniem w sferze kontaktu człowiek — maszyna. Do niedawna jeszcze wszystko co na tym polu osiągnięto, to urządzenia rozpoznające pewną, niewielką zresztą, liczbę pojedynczych, oddzielnie wypowiadanych wyrazów i to w zasadzie tylko przez jedną osobę, do głosu której aparaturę „przyuczono". Ostatnio wszakże dokonano istotnego postępu i w tej dziedzinie.

Na posiedzeniu National Computer Conference w Nowym Jorku zaprezentowano w ubiegłym roku nowo opracowane przez Dialog Systems Incorporated wejście głosowe pn. MODEL 1800. Jest ono pomyślane jako ogniwo pośredniczące między aparatem telefonicznym i komputerem; jest wyposażone w kilka łączy mogących pracować równolegle. Przewidziane jest w pierwszym rzędzie do zastosowania w bankowości i przedsiębiorstwach handlowo-transportowych.

Nowe urządzenie charakteryzuje się zdolnością „rozumienia" określonego zasobu wyrazów bez względu na osobę mówiącego, rozpoznaje prawidłowo ok. 95% wyrazów tego zasobu, niezależnie od różnic ich wymowy w rozmaitych regionach USA, wypowiedzianych przez kobietę lub mężczyznę. Zasób rozpoznawanych wyrazów na razie jeszcze przedstawia się dość skromnie, obejmuje liczebniki od *0* do *9*, potwierdzenie *yes* i przeczenie *no* oraz ok. 30 dalszych terminów bankowych, jak: *konto, suma, razem, depozyt* itp. Osoba mówiąca ma możność potwierdzenia albo zakwestionowania i poprawienia wypowiedzianego wyrazu, w razie wątpliwości co do prawidłowego zrozumienia go przez urządzenie.

Proces rozpoznawania wyrazów polega na porównaniu wyrazu wypowiedzianego z zapisem utrwalonym w pamięci urządzenia. Poprzednio zapisu takiego dokonywała jedna osoba (w procesie tzw. przyuczenia) — ta, która urządzenie miała obsługiwać. Głosy innych osób — wobec indywidualnych różnic wysokości, barwy głosu i różnic wymowy — mogły być dla urządzenia niezrozumiałe. Chcąc tego uniknąć, dokonano fonetycznej analizy występujących w USA regionalnych odmian wymowy, oddzielnie dla głosów męskich i kobiecych, wybierając w tym celu pięciuset reprezentantów i nagrywając ich głosy. Wypowiadane przez nich pojedyncze wyrazy sklasyfikowano dźwiękowo, dzieląc najpierw każdy wyraz na maksimum 12 charakterystycznych cząstek (tzw. segmentów), a następnie każdą cząstkę wg 32 klas, zależnie od częstotliwości dźwięku, obejmując skalę od 300 do 3300 Hz. W ten sposób stworzono $12 \times 32 = 384$ wzorców rozpoznawczych, uwzględniających średnie wartości „materiału" dźwiękowego. Wzorce te wprowadzono do pamięci urządzenia.

Wyrazy wypowiadane do urządzenia przez dowolne osoby są w ten sam (powyższy) sposób dzielone, klasyfikowane i porównywane z wzorcami. Jeżeli występuje zgodność w granicach założonych (różnych dla różnych dźwięków) pól tolerancji, poszczególne części wyrazu i cały wyraz zostaje zidentyfikowany. Proces rozbioru, porównania i identyfikacji poszczególnych cząstek nieznanego wyrazu przebiega bardzo szybko, trwa 30 μs. Identyfikacja wyrazu jako całości trwa dłużej i zależy od obszerności „słownika" zakodowanego w urządzeniu. Jeżeli np. słownik ten liczy 1000 wyrazów, czas identyfikacji wyrazu wyniesie $1000 \times 30\ \mu s = 30$ ms.

Jednostką sterującą i koordynującą pracę urządzenia jest minikomputer PDP 11/04. Jego oprogramowanie i słownik zapisane są na elastycznych dyskach magnetycznych. Urządzenie może współpracować z komputerami zewnętrznymi poprzez asynchroniczny system RS-232, pracujący w układzie „duplex" z szybkością przenoszenia 4800 bodów.

Zainteresowanych pełniejszym opisem urządzenia odsyłamy do artykułów w prasie fachowej — Ruske G.: Automatische Erkennung gesprochener Worter mit einem Funktionsmodell des Gehörs. Nachrichten Elektronik nr 6/1979; Link W.: Die Erkennung von Sprache durch elektronische Systeme. Nachrichten Elektronik, nr 9/1978.

Opracował A. R. na podstawie Nachrichten Elektronik, nr 8/1979

Wrażliwość skomputeryzowanego społeczeństwa na zagrożenia

Specjalna szwedzka komisja rządowa opracowała na zlecenie szwedzkiego Ministerstwa Obrony raport na temat „APD a wrażliwość społeczeństwa na zagrożenia".

Raport odróżnia zewnętrzne i wewnętrzne czynniki zagrożenia i omawia je kolejno, zaczynając od zagrożeń zewnętrznych.

Działania przestępcze, które mogą mieć znaczenie z punktu widzenia bezpieczeństwa państwa, to sabotaż i szpiegostwo, a także niektóre przestępstwa przeciw własności, które w warunkach komputeryzacji mogą przybrać zupełnie nowy wymiar, np. w związku z komputeryzacją obrotu płatniczego.

Szczególną uwagę poświęcono w raporcie terroryzmowi politycznemu, który zdaniem autorów raportu może stać się środkiem agresji międzypaństwowej. Wskazano w związku z tym na szereg zamachów terrorystycznych na ośrodki obliczeniowe w ostatnich latach.

Duże zbiory danych, szybkość ich przetwarzania, transmisja danych itd. otwierają możliwość nowych form szpiegostwa.

Dalszą formą zagrożenia zewnętrznego mogą być sankcje gospodarcze w postaci embarga na sprzęt informatyczny lub części zamienne wobec krajów zależnych od importu tego sprzętu.

W przypadku konfliktu zbrojnego należy też liczyć się z infiltracją agentów do systemów EPD spełniających żywotnie ważne funkcje; celem takiej infiltracji będzie sabotaż i wywołanie zamętu w kraju.

Sytuację wojenną raport ocenia w następujący sposób: zastosowanie informatyki w wojsku jest zaplanowane z uwzględnieniem sytuacji zarówno pokojowej jak i wojennej. Jeśli natomiast idzie o zastosowania cywilne w czasie wojny, rząd szwedzki wydał specjalne instrukcje dla władz cywilnych i gospodarki prywatnej, przewidujące znaczne ograniczenie użytkowania EPD w czasie wojny.

W przypadku wojny zachodzi możliwość okupacji części kraju przez siły nieprzyjaciela. Ze względu na koncentrację mocy obliczeniowych na obszarach wielkomiejskich, nawet okupacja ograniczonej części kraju może sparaliżować dużą część krajowych systemów EPD. Koncentracja ta powoduje też większą wrażliwość na bombardowania i sabotaż.

Specjalne ustawodawstwo z przepisami wykonawczymi reguluje w Szwecji sprawę niszczenia różnych obiektów, by nie wpadły w razie okupacji w ręce nieprzyjaciela. Brak jednak specjalnych przepisów dotyczących komputerów i zbiorów danych w analogicznej sytuacji. Raport zaleca wydanie takich przepisów.

Dalszym ważnym czynnikiem jest problem kadr. W czasie wojny duża część personelu informatycznego potrzebnego do obsługi systemów wojskowych rekrutować się będzie spośród obsługi systemów cywilnych.

Przy zastosowaniu broni jądrowej należy się liczyć z tzw. efektem uderzenia magnetycznego, które zniszczy lub uszkodzi komputery i systemy transmisji danych. Ochrona przed tym zjawiskiem jest ogromnie kosztowna.

Raport zajmuje się też skutkami katastrof żywiołowych i innych, które mogą bezpośrednio lub pośrednio wpłynąć na EPD. Może nastąpić zniszczenie samej instalacji komputerowej; może też nastąpić przerwa w zasilaniu energią, pociągająca za sobą poważne konsekwencje dla użytkowników komputerów.

Wśród wewnętrznych czynników wrażliwości raport zwraca uwagę na zbiory danych zawierające tajne lub poufne informacje. Raport zaleca ograniczenie liczby i zakresu rejestrów ludności do niezbędnego minimum i wskazuje na niebezpieczeństwo grożące takim rejestrom. W różnych sytuacjach wojennych i kryzysowych takie zbiory danych mogą ułatwić agresorowi uzyskanie kontroli nad ludnością i wyselekcjonowanie osób, które chce wyeliminować lub użyć do swoich celów.

Raport zwraca też uwagę, że niektóre zbiory danych zawierają ogromnie czułe informacje osobiste i w razie dostania się w niepowołane ręce mogą stać się narzędziem różnego rodzaju wymuszeń.

Dane o poufnym charakterze zawarte są także w zbiorach danych o przedsiębiorstwie. Innymi danymi, które mogą mieć znaczenie w razie dostania się w niepowołane ręce, są np. szczegóły sieci komunikacyjnej kraju, a także przechowywane w komunalnych systemach EPD szczegółowe dane techniczne o sieciach elektrycznych, gazowych, wodociągowych itd.

W dalszym ciągu raport zajmuje się systemami funkcjonalnie wrażliwymi. Duża część wrażliwych funkcji komunalnych i społecznych jest skomputeryzowana (np. dane zakładu ubezpieczeń społecznych, systemy gospodarki kredytowej, systemy handlu, składowania i dystrybucji). Wrażliwe są też systemy sterowania produkcją. Zakłócenia takich systemów mogą spowodować opóźnienia wypłat, trudności w wykonywaniu zamówień, utratę kontroli nad gospodarką magazynową i dystrybucją, a w konsekwencji braki w zaopatrzeniu, przerwy w produkcji, zakłócenia komunikacji itd.

Gdy mowa o koncentracji działalności EPD, raport rozróżnia koncentrację geograficzną i funkcjonalną. Koncentracja geograficzna oznacza duże skupienie mocy obliczeniowych na pewnych obszarach. Koncentracja funkcjonalna oznacza scentralizowane przetwarzanie danych wielu klientów przez wielkie centralne systemy lub biura usługowe. Obie te koncentracje zwiększają wrażliwość na zakłócenia.

Dalszym czynnikiem zwiększającym wrażliwość jest wzajemne powiązanie i uzależnienie wielu systemów informatycznych. Istotnym czynnikiem wrażliwości jest zależność systemów od konkretnych osób. W braku pełnej dokumentacji systemu użytkownik jest całkowicie uzależniony od twórcy systemu. To samo dotyczy w ogóle zależności systemów od pracowników.

Służby wywiadowcze już dawno stosują metodę masowego zbierania informacji, które same przez się nie są tajne, lecz z ich powiązania i analizy można wyciągnąć wnioski o tajemnicach wojskowych i gospodarczych. Zastosowanie komputerów znacznie ułatwia i przyspiesza taką analizę danych.

Szczególne niebezpieczeństwo kryje w sobie międzynarodowy przepływ danych. Ochrona danych przy ich przesyłaniu poprzez granice państwowe jest oczywiście trudniejsza niż na terenie kraju.

Raport kończy się następującymi ogólnymi wnioskami:

- Wrażliwość skomputeryzowanego społeczeństwa na zagrożenia jest niedopuszczalnie wysoka i jeśli nie zostaną podjęte kroki zaradcze, będzie nadal rosła.
- Do głównych czynników zwiększających wrażliwość należą:
  — zależność od zagranicznych źródeł dostaw sprzętu
  — koncentracja systemów
  — zależność od konkretnych osób
  — ryzyko powiązania określonych typów zbiorów.

  Wszystkie te czynniki ułatwiają różne formy agresji zewnętrznej.
- Dla zapobieżenia zagrożeniom potrzebne jest staranne zaplanowanie środków ochrony.
- Przeniesienie wielkich komputerów w bezpieczniejsze miejsca jest praktycznie niewykonalne. Natomiast zastosowanie mniejszych komputerów i rozproszenie mocy obliczeniowych zmniejsza wrażliwość. Trzeba opracować metody niszczenia komputerów i nagromadzonych informacji jako ochronę przed zawładnięciem przez nieprzyjaciela.
- Trzeba wzmocnić powszechną świadomość wrażliwości systemów informatycznych na zagrożenia.
- Konieczne jest zmniejszenie zależności od zagranicznych dostawców sprzętu i od zagranicznych usług serwisowych.
- Niezbędny jest staranny dobór zaufanego personelu.
- Istnieje potrzeba fachowej konsultacji przy planowaniu środków bezpieczeństwa i postępowania na wypadek sytuacji wyjątkowych. Dotyczy to także czasu pokoju.

Opracował: A. M.
DOI — CPiZI
na podstawie
„Computer als Friedensstifter?" Diebold Management Report 1979 nr 6 s. 1—3, nr 7/8 s. 14—15

Komputeryzacja szklarni

W Hillscheid k. Koblencji (RFN) jedno z wielkich gospodarstw ogrodniczych zastosowało ostatnio minikomputer do nadzoru i sterowania produkcji kwiatów. Olbrzymie rozmiary produkcji ogrodniczej, a także coraz bardziej złożone i rygorystyczne wymagania klimatyczne — przy intensywnej hodowli roślinnej spowodowały, że tradycyjne metody nadzoru i interwencji ogrodników ręcznie regulujących temperaturę, wilgotność czy nasłonecznienie upraw stają się bardzo uciążliwe, a coraz częściej praktycznie niewykonalne, prowadząc do znacznych strat produkcyjnych.

Wspomniane gospodarstwo ogrodnicze obejmuje 30 szklarni po ok. 1000 m² powierzchni (a więc łącznie ok. 3 ha), które obecnie obsługuje w sposób ciągły minikomputer, do którego przyłączono ok. 500 różnych stanowisk pomiarowych. Dane z tych stanowisk są na bieżąco przetwarzane, powodując zwrotne przesyłanie odpowiednich sygnałów sterujących do punktów regulacji klimatycznej. Każda szklarnia jest nadzorowana i sterowana odrębnie w zależności od ustalonych dla danej rośliny i jej wzrostu parametrów klimatycznych. Oprócz tradycyjnych elementów pomiaru i regulacji warunków produkcji szklarniowej (temperatura, wilgotność, nasłonecznienie) system automatyczny uwzględnia również szereg dodatkowych czynników, takich jak np. temperatura gleby, zawartość dwutlenku węgla w powietrzu, a także wpływ zewnętrznych zmian klimatycznych, wprowadzanych przez operatora na podstawie bieżących komunikatów meteorologicznych.

Oprac. WK na podstawie czasopisma DATA REPORT nr 2/79

Nowoczesne metody programowania dla niewidomych

Fundacja Popierania Rehabilitacji oraz Federalne Biuro Pracy w RFN podejmują od wielu lat starania zmierzające do udostępnienia niewidomym zawodów informatycznych. Dzięki zastosowaniu wielu specjalistycznych przyrządów i urządzeń udało się już na początku lat siedemdziesiątych stworzyć w RFN warunki do zatrudniania niewidomych na stanowiskach programistów. Rozwój techniczny sprzętu komputerowego, a zwłaszcza programowania w trybie konwersacyjnym, wymagał ostatnio dostosowania nowych metod i narzędzi pracy do możliwości niewidomych programistów.

Cel ten ostatnio osiągnięto z pomocą pewnej firmy w Stuttgarcie, która skonstruowała specjalną przystawkę do monitora ekranowego typu SIEMENS 8161, dzięki czemu uzyskano wyspecjalizowany monitor ekranowy dla niewidomych typu BD 80 w cenie ok. 15 tys. marek (ok. 8300 dol. USA). Przystawka ta umożliwia przekształcenie dowolnego, wyświetlonego na ekranie, tekstu w ciągu 0,5—2,0 s na postać przystowaną do odczytu przez niewidomego. Podstawowym elementem przystawki, która jest wyposażona w mikroprocesor, jest tzw. moduł wiersza brajlowskiego. Moduł ten umożliwia konwersację wyświetlonego na ekranie monitora 80-znakowego wiersza na tekst brajlowski oparty na zapisie 8-punktowym. Dwa dodatkowe pola brajlowskie przeznaczone są do wykazywania numeru wyświetlanego wiersza oraz aktualnej pozycji kursora (znacznika) na ekranie. Wywołanie określonego wiersza ekranu (1—24) odbywa się za pomocą klawiatury dziesiętnej o układzie analogicznym jak w kalkulatorach kieszonkowych. Inne klawisze pozwalają na skasowanie lub powtórzenie wiersza brajlowskiego oraz wywołanie poprzedniego lub następnego wiersza z ekranu monitora. Przystawka ma ok. 50 cm szerokości 8 cm wysokości i 25 cm głębokości. Obsługa monitora ekranowego odbywa się za pomocą normalnej — nie przystosowanej do potrzeb niewidomego — klawiatury.

Pierwsze doświadczenia wykazały, że pożądane było by wprowadzenie sygnalizacji dźwiękowej informującej o gotowości monitora ekranowego do wprowadzania nowych danych, ponieważ niektóre procedury przetwarzania powodują znaczne czasy oczekiwania. Sytuacje te są trudno rozpoznawane przez niewidomego, który nie wie, w którym momencie może kontynuować pracę. Dzięki wbudowaniu do wspomnianego monitora typu 8161 dodatkowego mikroukładu oraz głośnika, ten istotny z punktu widzenia potrzeb niewidomego mankament zdołano usunąć, powodując generalną poprawę niezawodności i szybkości operowania tym urządzeniem. Dzięki opisanym konstrukcjom niewidomi programiści mogą już w pełnym zakresie korzystać z nowoczesnych metod programowania.

Oprac. WK na podstawie czasopisma DATA REPORT nr 6/79

Komputer w Kościele

W końcu marca br. firma SPERRY UNIVAC w swym międzynarodowym ośrodku szkolenia kadr kierowniczych St. Paul de Vence (Francja) zorganizowała nietypowe sympozjum „Komputer w Kościele". W ciągu trzech dni duchowni (w tym dwóch biskupów austriackich) oraz osoby świeckie zatrudnione w administracji kościelnej wygłaszali referaty i dyskutowali na temat komputeryzacji działalności kościelnej. Referaty dotyczyły podstawowego problemu, jakim jest miejsce oraz sposób usprawnienia współczesnych i przyszłych zadań Kościoła w wyniku zastosowania informatyki.

Największa część uczestników sympozjum pochodziła z Austrii, gdzie era komputerowa w Kościele katolickim rozpoczęła się już 10 lat temu z chwilą zakupu przez Wydział Finansowy diecezji St. Pölten własnego systemu komputerowego UNIVAC 9300.

Informatyka okazała się dziś niezbędną również dla działalności kościelnej, jakkolwiek spełnia tu głównie funkcje usługowe przy realizacji zadań podstawowych. Jeden z głównych referatów wygłosił austriacki biskup dr Krätzl. Przedstawił on na wstępie następujące przyczyny tak szybkiego przyjęcia przez Kościół nowej technologii przetwarzania danych:

- zadania administracyjne współczesnego Kościoła nieustannie się rozrastają, a jednocześnie coraz dotkliwiej odczuwany jest deficyt duchownych. Ponadto na Kościół spadły szczególnie trudne zadania administracyjne związane z pobieraniem daniny kościelnej
- nowoczesna służba duszpasterska stawia całkowicie nowe wymagania, zwłaszcza w zakresie tzw. duszpasterstwa aktywnego (niem. *nachgehende Seelensorge)* wynikającego z konieczności odszukania i nawiązania kontaktu ze współczesnym człowiekiem również w jego środowisku pracy oraz w czasie wolnym
- II Sobór Watykański na nowo zdefiniował stosunek Kościoła do problemów światowych, popierając jego szersze otwarcie dla osiągnięć drugiej rewolucji technologicznej; wkroczenie nowych technologii w życie Kościoła wiąże się oczywiście z nowymi problemami, zarówno w zakresie nowych możliwości, jak i nowych ograniczeń.

Biskup dr Krätzl przedstawił problematykę zastosowania komputerów w Kościele z następujących trzech punktów widzenia:

- usprawnienia administracji
- nowych możliwości duszpasterstwa
- lepiej administrowanego Kościoła i jego podstawowych zadań.

Poprawa sprawności działań administracyjnych w wyniku zastosowania komputera sprowadzała się początkowo do dziedziny pobierania daniny kościelnej. Wkrótce informatyką objęto również dziedzinę rachunkowości kościelnej, obliczania wynagrodzeń dla duchownych i pozostałych pracowników diecezji, a także ewiedencji księgowej i realizacji budżetów.

Dodatkowe wykorzystanie komputera osiągnięto przez obsługę rachunkowości różnych instytucji kościelnych, takich jak Caritas, przedszkola parafialne, szkoły i szpitale.

Od samego początku informatykę włączono również do bezpośredniej działalności duszpasterskiej. Obecnie proboszczom oferuje się usługi informatyczne w zakresie wykazów parafian, wykazów akt urodzenia, usług adresowych. Od 1978 r. w archidiecezji wiedeńskiej istnieje grupa robocza „Duszpasterstwo i informatyka", która ma opracować tzw. pakiet duszpasterski. Informatyka pomaga duchownym lepiej poznać środowisko parafialne oraz odciąża ich od pracy biurowej, dzięki czemu mogą poświęcić więcej czasu i sił dla działalności duszpasterskiej.

Biskup Krätzl podkreślił fakt, że informatyka całkowicie spełni oczekiwania duszpasterstwa wtedy, gdy zostaną w pełni poznane jej funkcje usługowe. „Komputer daje do rąk administracji kościelnej wiele możliwości, ale wymaga również wielu badań".

Oprac. W. K. na podstawie SPERRY UNIVAC. Informationen 3/1980

Czy uda się ożywić wymianę myśli na łamach INFORMATYKI?

W ostatnich tygodniach powróciła na łamy prasy prawdziwa dyskusja. Można w niej także znaleźć coraz pełniejsze informacje — co powoduje, że dzienniki i tygodniki są rozchwytywane. Wypada mieć nadzieję, że i czasopisma naukowo-techniczne otrząsną się z letargu. Może wtedy zainteresowanie nimi znacznie się podniesie? Dziś INFORMATYKA, jako bodajże pierwsze z czasopism SIGMY, próbuje poszerzyć autentyczny kontakt z Czytelnikami. To bardzo cieszy, zwłaszcza że w ostatnich latach były na jej łamach widoczne cienie różnych bolesnych zjawisk; m.in. swoistej „propagandy sukcesu" [1]).

Zastanówmy się nad tym, jakie warunki powinny być spełnione, aby „sprzężenie zwrotne" z Czytelnikami rzeczywiście ożywiało czasopismo. Najogólniej biorąc — chodzi o to, aby Czytelnicy uważali miesięcznik „za swój", za coś, co jest wspólne dla całego środowiska informatyków. Sukcesem byłaby zatem sytuacja, w której INFORMATYKA będzie się opierała bardziej na wiernych Czytelnikach, niż na Autorach, którzy muszą pisać ze względu na prestiż czy awans. Wszyscy jednak muszą być przekonani, że zaprzyjaźnienie się z INFORMATYKĄ nie przyniesie im zawodu, że Redakcja ma zawsze coś ważnego do powiedzenia i że zawsze życzliwie wysłucha wszystkich głosów. Podkreślamy: nie chcemy przez to sugerować, że Redakcja „wykorzystuje chwilę"! Chodzi nam o to, że wzajemne zaufanie i dobry klimat do współpracy przy wydawaniu czasopisma fachowego — szczególnie o tak szerokim zasięgu i tak rozległej, nowoczesnej tematyce — tworzy się całymi latami.

Zdajemy sobie sprawę z wagi spraw poruszonych przez Redakcję w numerze 9/80. Podzielamy wiele krytycznych ocen i rozumiemy wątpliwości. Domyślamy się, że napisane słowa są tylko wywołaniem pewnych problemów. Niepowtarzalny czas ogólnonarodowej dyskusji daje jedyną szansę do szerokiego omówienia spraw zawodowych. Analiza taka wymaga jednak nie tylko apelu, lecz przede wszystkim — odważnego programu. Jeżeli takiej mądrej wizji zabraknie w rozpoczynającej się dyskusji, to (o ile ta dyskusja się rozszerzy) utoniemy w szczegółach i widmach. Z powyższym łączy się ściśle sprawa oceny dotychczasowego „modelu" INFORMATYKI oraz zaproponowania stosownych zmian w jej kształcie. Na ten temat mamy kilka uwag i zamierzamy wkrótce je przedstawić.

Oprócz kilku spraw ogólnych, naszkicowanych wyżej, warto również zwrócić uwagę i na szczegóły, tak ważne dla stworzenia właściwej atmosfery w kontaktach z Czytelnikami. Sądzimy, że otrzymanie każdego listu Redakcja powinna potwierdzić w niezdawkowy sposób — tak, aby przez bezpośredni kontakt podtrzymać nawiązaną łączność. Niby to drobiazg, ale jakże ważny dla psychicznego nastawienia piszących!. Podstawowym warunkiem dyskusji jest szczera wymiana poglądów co najmniej dwóch osób. Warto się zastanowić, czy odpowiedź Redakcji powinna być drukowana obok listu? W wielu przypadkach konieczne będzie szersze naświetlenie sprawy przez Redakcję, uściślenie danych, czy komentarz — nie powinno to chyba sprawiać większego kłopotu.

Listy nie powinny być jedyną drogą zdobywania opinii Czytelników. Wiele osób ma tak serdecznie dość panoszącej się wszędzie biurokracji, że nawet do pisania listów czuje wstręt. Nie każdy zresztą ma dar ciekawego pisania o codziennych sprawach zawodowych. Sądzimy więc, że warto co pewien czas zrobić w serdecznej, ciepłej atmosferze spotkania dyskusyjne z udziałem Redakcji, Rady Programowej, no i oczywiście — Czytelników. Jednym z niezłych pomysłów wydają się konferencje naukowo-techniczne czy inne spotkania środowiskowe, stanowiące niekłopotliwą okazję do wyłonienia „ciekawych postaci" i „pogadania" z nimi przy okrągłym stole. Każda taka idea wymaga jednak pewnej ofensywności ze strony Redakcji. Nie można zaprzepaścić tej szansy, bo dziś wielu fachowców chce mówić — najwyższa więc pora ich wysłuchać. Wypowiedzi takie mogłyby wtedy znaleźć prostszą drogę na łamy pisma.

Zwracacie się do Czytelników z pytaniem, jakie decyzje wpłynęły na obecną sytuację informatyki. Dyskutując na ten temat można postawić setki trudnych pytań. Oto niektóre:

— kto jest odpowiedzialny za skandaliczne opóźnienia w dziedzinie krajowego systemu mikroprocesorowego?

— kto polecił podjąć produkcję nieudanego minikomputera MERA 300?

— czyje zaniedbania doprowadziły do tego, że brak jest krajowego programowanego kalkulatora stołowego o dużej mocy obliczeniowej?

— jakie czynniki powodują rozproszenie wysiłków w dziedzinie oprogramowania i wypaczenia w jego obrocie?

— kto przeforsował (i dlaczego) szwedzką licencję na monitory ekranowe MERA 7950, zamiast np. szerokiej współpracy z VIDEOTONEM? Aby jednak osąd sprawy był obiektywny, odpowiedź na takie pytania nie może rozpływać się w „ogólnych tendencjach" i „decyzjach odgórnych".

Pytanie o nazwiska decydentów i ich doradców nie jest szukaniem kozłów ofiarnych do rozrachunku z przeszłością. Co najmniej dziesięć ostatnich lat, w których uprawianie zawodu informatyka (lub bezpośredni kontakt z tą dziedziną) przynosiło więcej rozczarowań niż zadowolenia, upoważnia Czytelników do żądania konkretnych wyjaśnień oraz do sformułowania twardej, ale sprawiedliwej oceny ludzi. Zwłaszcza tych, którzy doprowadzili sytuację w informatyce na pogranicze absurdu. Nie wszyscy przecież jesteśmy winni.

Powstaje myśl, aby jedyne czasopismo informatyczne w kraju było jednym z ważniejszych ogniw, które powinny łączyć całe środowisko z decydentami. Można marzyć, że stanie się ono jednym ze skuteczniejszych zabezpieczeń przed dalszymi błędami w naszej branży. Tworząca się szersza niż dotąd więź z Czytelnikami (dwukierunkowa!) mogłaby przyczynić się do integracji środowiska wokół programu poprawy sytuacji w informatyce. Z jednej strony — czasopismo mogłoby się stać środkiem rzetelnej informacji o bieżących pracach i zamierzeniach w branży, z drugiej — umożliwiłoby krytykę i zgłaszanie postulatów. Oby ten kanał informacyjny był stale drożny!

Redakcji zaś wypada życzyć, aby twórczej dyskusji nigdy nie było dość.

Krystyna i Jacek ŻEBROWSCY

Łódź, 26 X 1980

[1]) Dobrym tego przykładem jest optymistyczna informacja o reorganizacji Komitetu Informatyki. Oczywiście — było to typowe działanie pozorne. „Niesprawność" Komitetu była tylko objawem, przyczyny trudności leżą zaś głębiej. Może zobaczymy je teraz?

Mini... mikro...

Nie bez wzruszenia trzymam w ręku „żółtą książeczkę" o mikroelektronicznych maszynach cyfrowych [1]). Pierwszy z jej Autorów jest nestorem informatyków polskich i polonizatorem jakże użytecznego pojęcia francuskiego *informatique* (a w zaraniu lat pięćdziesiątych był nie tylko moim Mistrzem Wtajemniczenia).

Trójka autorska reprezentująca Instytut Podstaw Informatyki PAN, skutecznie wyminęła rafy wspominek narodowych oraz kontrowersji historycznych pomiędzy zwolennikami mikroprocesorów a protagonistami tworzenia minisystemów cyfrowych, jak niekiedy nazywane są minikomputery, omawiane w osobnej książce WNT-owskiej [2]). Obie prace powstały zresztą poza tradycyjną serią *Informatyka* i można je traktować jako prototypy monografii, tyle że świadomie adresowanych do Czytelników, którzy dopiero zaczynają poznawać omiawiane dziedziny.

Wróćmy jednak do „żółtej książeczki". Można się spierać, czy Autorzy nie potraktowali historii układów scalonych i mikroprocesorów aż nazbyt zwięźle. Przykładowo — pominięto stosowane w niektórych typach kalkulatorków kieszonkowych układy typu EXTRA-LSI. Niemniej, taka minimonografia mikroprocesorowa nie może traktować, niczym katalog, o wszystkich produkowanych na świecie elementach konstrukcyjnych (a zwłaszcza kalkulatorów, które dla Autorów świadomie stanowią marginesowy obszar zainteresowania). Zresztą świat mikroelektroniki nie kończy się na technice cyfrowej.

Omawiając *Mikroprocesory* trzeba zdawać sobie sprawę z faktu, że jest to pierwsze ogólnodostępne polskie opracowanie na ten temat. Dotychczas ukazywały się tylko wewnętrzne skrypty CEMI (technologia) czy też ZETO (programowanie), w dodatku powielane zaledwie w 100—200 egz. Mówienie o dostępności może w oczach naszych Czytelników zakrawać na ironię, bowiem nawet nakład 100-tysięczny rozszedłby się bardzo szybko. Za granicą książki tego typu ukazują się w formie wydawnictw marketingowych, nakręcających koniunkturę mikroprocesorową, często rozsyłanych gratisowo. Jedyną wadą takich wydawnictw jest ich firmowa monotematyczność.

Mikroprocesory pod tym względem próbują stworzyć warunki pewnego obiektywizmu. Wprawdzie Autorzy podkreślają, że przyszłość krajowej mikroinformatyki to odpowiednik linii rozwojowej INTEL-8080, ale omawiają także możliwie obszernie wyroby firmy MOTOROLA oraz TEXAS INSTRUMENTS. Szczególnie obszernie potraktowano mikroprocesory INTEL-8080 oraz TMS-9900, zamieszczając pełne listy instrukcji, formaty, oznaczenia końcówek i sformalizowane opisy funkcji. Znaczną część objętości przeznaczono na systematyczne omówienie podstawowych technologii cyfrowych układów scalonych. *Mikroprocesory* będą jednak stanowiły raczej podstawowe źródło informacji o anonimowym jeszcze odpowiedniku INTEL-8080, przygotowywanym do masowej produkcji w ITE UNITRA-CEMI. To ostatnie podkreślenie jest o tyle istotne, że za kilkanaście miesięcy prawdopodobnie łatwiej będzie w naszym kraju o kupno mikroprocesora, aniżeli książki na jego temat!

Autorzy *Mikroprocesorów* napotkali duże trudności próbując wprowadzić jednolitą terminologię polską (w miejsce określeń żargonowych, fonetycznie lub ortograficznie zapożyczanych z angielskiego). Ucierpiała w efekcie strona dokumentacyjna: czytelnik musi zadać sobie nieco trudu i sam poprawić podstępne błędy zecerskie w oznaczeniach symbolicznych, niestety, nie stanowiące rzadkości.

Osobiście za największą wadę *Mikroprocesorów* w ich obecnej formie uważam bynajmniej nie różne usterki — redakcyjne, drukarskie czy terminologiczne — ale brak szczegółowo omówionego przykładu zastosowań, odpowiedniego do poziomu technika-elektronika.

Druga z omawianych książek może być na zasadzie inwersji nazwana mikromonografią o minikomputerach. W swym ujęciu monograficznym książka ta jest też pionierska w literaturze rodzimej, chociaż minikomputery były już opisywane. Już choćby w serii *Informatyka* (WNT) ukazały się w ub.r. książki Weitzmana o strukturze i zastosowaniach systemów minikomputerowych oraz Eckhouse'a o organizacji i oprogramowaniu takich systemów. Otóż owa pionierskość polega na odejściu od dyskusji „mini czy nie-mini" i zajęciu się wspomaganiem eksperymentatorów.

Książka Ożarowskiego i Kornackiego jest więc adresowana do zupełnie innego kręgu czytelników. Aż połowa jej objętości poświęcona jest na różnego rodzaju wprowadzenia informatyczne. Autorzy starają się nie wychodzić poza kontekst przetwarzania sygnałów pomiarowych środkami minikomputerowymi. W tym aspekcie nawet „połowa wprowadzająca" staje się ciekawa i dla informatyków, zwłaszcza interesujących się systemem CAMAC czy też telemetryczną zmianą skali czasu. Odczuwam jednak pewien niedosyt w wytłumaczeniu systemu CAMAC jako standardu światowego i jego związków ze standardami URS.

W swym subiektywnym odczuciu za szczególnie cenne w *Minikomputerach* uważam obszerne omówienie przykładowych stanowisk badawczych wspomaganych komputerem:

— cewnikowania serca podczas operacji

— wydzielania sygnału EKG z zakłóceń telekomunikacyjnych

— przetwarzania mikroskopowego obrazu komórki.

Ostatni z wymienionych przykładów odnosi się do systemu zbudownego i uruchomionego w Instyucie Biocybernetyki i Inżynierii Biomedycznej PAN (na bazie minikomputera K-202). Wydaje mi się jednak, że przy omawianiu minikomputerowych analizatorów składu można było nadmienić o możliwościach wbudowywania układów mikroprocesorowych np. w chromatografy.

Niewątpliwą zaletą książki Ożarowskiego i Kornackiego jest dbałość redakcyjna o tekst i poprawność rysunków. *Minikomputery* mogą więc chyba liczyć na zainteresowanie szersze niż pierwotnie oczekiwał wydawca. A z pewnością po upływie kilku lat zaistnieje konieczność opracowania następnego wydania, rozszerzonego o dokonane w tym czasie w kraju nowe systemy minikomputerowe.

I tak w podsumowaniu obu recenzji dochodzimy do problemu polityki popierania postępu technicznego. Truizmem byłoby tu przypominanie, że w historii postępu technicznego o powodzeniu szeregu wynalazków decydowało wywołanie pozytywnego zainteresowania u potencjalnych subsydiodawców. Albo inaczej mówiąc, autorzy pomysłu nie mogą liczyć na sukcesy dokonywane wbrew ludziom przemysłu. Otóż obie omawiane tu książki łączy to, że mogą liczyć zarówno na zainteresowanie konstruktorów, jak i działaczy. Ba, będą stymulować tak pożyteczną społecznie falę zainteresowania nowoczesnymi systemami „mini" i „mikro", uzmysłowiając wyraźnie, że w skali masowej można się wspaniale wyżywać twórczo bez konieczności opracowania na nowo „swojego" pomysłu mikroprocesora czy też miniprocesora. Obie książki uzmysłowiają bowiem atrakcyjność szeroko rozumianej „architektury zastosowań", zarówno dla elektroników, jak i programistów.

Adam B. EMPACHER

[1]) Romuald Marczyński, Przemysław Bąkowski, Janusz Sochacki: Mikroprocesory. Wiadomości wstępne. WNT, Warszawa, 1979; Wyd. I; nakład 10 tys., str. 230, 23 tab. + 137 rys., cena 50 zł

[2]) Ryszard Szczepan Ożarowski i Stanisław Kornacki — Minikomputery w pracach eksperymentalnych. WNT, Warszawa 1980, wyd. I, nakład 4 tys., str. 244, 149 rys. + 7 tab., cena 48 zł

O jednolitą terminologię

„Instrukcja" czy „rozkaz"?

Odpowiedź na pytanie, którego z tych określeń używać, nie jest oczywista nawet dla specjalistów. W wielu publikacjach termin instrukcja jest używany wymiennie z terminem rozkaz. Zakresy znaczeniowe obu terminów z pewnością zachodzą na siebie. Sytuację pogarsza fakt, że w informatyce wyraz instrukcja nie jest dokładnym odpowiednikiem angielskiego określenia *instruction*.

Według normy PN-71/T-01016 instrukcja (ang. *statement*) jest to wyrażenie języka źródłowego określające pewną operację, a rozkaz (ang. *instruction*) jest to ciąg znaków inicjujących i określających częściowo lub całkowicie operację. Można stąd wnioskować, że termin instrukcja dotyczy języka, a rozkaz nie; lecz czy wyrażenie języka nie musi być ciągiem znaków w tym języku?

Wątpliwość wyjaśnia norma ISO-2382 (obowiązująca także w Polsce), która nie różnicuje pod tym względem obu terminów. Według tej normy instrukcja (ang. *statement*) jest to w języku programowania wyrażenie znaczące, które może opisywać lub określać operacje i jest zwykle zupełne w kontekście tego języka, a rozkaz (ang. *instruction*) oznacza w języku programowania wyrażenie znaczące, które określa pojedynczą operację i — jeżeli istnieją — jej argumenty.

Z podanych określeń wynika, że istotna różnica między instrukcją a rozkazem polega na większej ogólności instrukcji. Według normy ISO-2382 instrukcja dotyczy wszystkich operacji wobec pojedynczej operacji określanej przez rozkaz, a według normy PN-71/T-01016 — pełnej operacji wobec częściowego określenia operacji przez rozkaz. Zauważalna niezgodność wynika najprawdopodobniej stąd, że operacja jest rozumiana nieco inaczej w każdej z norm.

Jeżeli nadal istnieją wątpliwości, warto zajrzeć do „Leksykonu informatyki" (G. Löbel, P. Müller, H. Schmid, WNT, Warszawa, 1977), który choć jest pracą popularną, spełnia w naszych warunkach pożyteczną rolę wyjaśniając wiele kwestii szczegółowych. Choć nie podaje on precyzyjnej definicji instrukcji, jednak pod tym hasłem stwierdza się wyraźnie, że instrukcja może składać się sama z wielu instrukcji cząstkowych, a instrukcje, których nie można rozłożyć na cząstkowe, nazywają się rozkazami.

Zatem każdy rozkaz jest instrukcją, lecz nie każda instrukcja rozkazem. Nie jest więc błędem, jeżeli wyrażenie MOV A, B języka symbolicznego nazwiemy instrukcją, choć precyzyjniej należałoby je nazwać rozkazem. Natomiast wyrażenie GO TO 100 języka FORTRAN można nazwać tylko instrukcją.

Przez listę rozkazów (taki odpowiednik ang. określenia *instruction set* podaje norma PN-71/T-01016, a więc nie *zbiór rozkazów* ani repertuar rozkazów) rozumie się według normy ISO zbiór wszystkich rozkazów komputera, języka lub języków programowania. Określenie podane w normie polskiej jest węższe, ponieważ dotyczy tylko języka programowania, natomiast treść haseł, lista rozkazów i repertuar rozkazów w „Leksykonie informatyki" jest zgodna z definicją ISO.

Wymieniona norma międzynarodowa precyzuje dalsze określenia związane z terminem rozkaz, które warto przytoczyć.

Makrorozkaz (makroinstrukcję) określa się jako rozkaz języka źródłowego, który ma zostać zastąpiony przez określony ciąg rozkazów tego samego języka źródłowego. Zdecydowanie nie zaleca się używania terminu pseudorozkaz (pseudoinstrukcja), lecz dyrektywa (ang. *directive*), który oznacza wyrażenie znaczące w języku programowania, wpływające na interpretację innych wyrażeń tego języka. Terminem bliskim dyrektywie jest komenda (ang. *command*), zwana także poleceniem, którą można określić jako wyrażenie znaczące w języku porozumiewania się z innymi programami systemowymi, np. z programem redagującym.

Skoro mowa o rozkazach, warto poświęcić trochę miejsca nazwom poszczególnych rodzajów rozkazów. Choć trudno przypuszczać, aby w praktyce udało się ujednolicić nazwy rozkazów, przede wszystkim dlatego, że nie dbają o to producenci komputerów i mikroprocesorów, to warto podać pewne wskazówki — jak postępować w sytuacjach wątpliwych. Istotną rolę może tu odegrać norma ISO-2382 która definiuje niektóre rodzaje operacji, a więc pośrednio odpowiadające im rozkazy.

Przykładowo, zmiana kolejności wykonywania rozkazów może nastąpić w wyniku wykonania rozkazów skoku (ang. *jump*) lub rozgałęzienia (wg normy PN-71/T-01016 odpowiednik angielskiego terminu *branch*). Według normy ISO skok podczas wykonywania programu jest to zmiana niejawnej lub zadeklarowanej kolejności wykonywania rozkazów. Rozgałęzienie podczas wykonywania programu jest to wybór jednego z kilku alternatywnych zbiorów rozkazów. Żadna z norm nie definiuje jednak rozkazu skip, który można określić polskim odpowiednikiem przeskok, ponieważ zmiana kolejności polega w tym wypadku zazwyczaj na pominięciu jednego lub najwyżej kilku rozkazów.

Nie jest powszechnie wiadomym, że najczęściej wykonywane operacje przepisywania lub kopiowania danych (ang. *copy*) są także dokładnie zdefiniowane w normie ISO. Operacja wprowadzania danych do rejestru nazywa się ładowaniem (ang. *load*). Wprowadzanie danych z rejestru do pamięci wewnętrznej nazywa się zapamiętywaniem (ang. *store*), a przepisywanie z jednej komórki pamięci wewnętrznej do innej komórki tej pamięci — przesyłaniem (ang. *move*). Wprowadzanie danych do pamięci zewnętrznej nazywa się zapisywaniem (ang. *write*), a przepisywanie z pamięci zewnętrznej — odczytywaniem (ang. read). Przepisywanie danych między dwoma komórkami pamięci zewnętrznej nazywa się również przesyłaniem (ang. *transfer*). Wynika stąd, że odpowiednie rozkazy powinny nazywać się po polsku rozkazami ładowania, zapamiętywania, przesyłania, zapisu i odczytu, co jest na ogół praktykowane.

Pozostaje jeszcze do wyjaśnienia kwestia przyjmowania danych przez system cyfrowy jako całość lub dostarczania danych przez ten system. Nazwy odpowiednich rozkazów są w zasadzie ustalone przez normy jako rozkaz wejścia (ang. *input*) i wyjścia (ang. *output*), natomiast odpowiadające tym rozkazom czynności określane są jako wprowadzanie i wyprowadzanie (ang. *input process, output process*). Dlatego wydaje się, że obu ostatnich określeń nie należałoby używać w innym znaczeniu.

Choć istnieje wiele innych rodzajów rozkazów, niebezpieczeństwo niejednoznaczności jest niewielkie, dlatego na ogół nie są definiowane w normach.

Janusz ZALEWSKI

Bibliografia wydawnictw polskich z dziedziny informatyki

● **Oprogramowanie teleprzetwarzania maszyn Jednolitego Systemu — NIEMCZYCKI L., WNT, Warszawa 1979, s. 179, cena 54 zł**

Wprowadzenie do transmisji danych. Oprogramowanie teleprzetwarzania. Podstawowa metoda dostępu telekomunikacyjnego BTAM. Telekomunikacyjna metoda dostępu — TCAM. Wirtualna metoda dostępu telekomunikacyjnego — VTAM.
Książka przeznaczona jest dla programistów i projektantów systemów teleprzetwarzania danych. Wymaga od czytelnika odpowiedniego przygotowania. Niezbędna jest znajomość programowania w ASEMBLERZE uzupełniona makrorozkazami zarządzania zbiorami danych dla trzech metod dostępu: BSAM, QSAM i VSAM. W mniejszym stopniu konieczna jest wiedza o makroznakach usługowych programu organizacyjnego systemu operacyjnego, zdaniach języka kontroli zadań i metodyce generowania systemu operacyjnego.

● **Sieci teleinformatyczne — DAVIES D. W., BARBER L. A., Tłum. wyd. ang. z 1973 r., WNT, Warszawa 1979, s. 661, cena 165 zł**

Komputery i telekomunikacja. Przekazywanie danych a sieć telefoniczna. Współpraca z komputerem. Sieci teledacyjne wydzielone. Teledacja. Sterowanie przepływem informacji, pamiętanie i kodowanie. Zwielokrotnianie cyfrowe. Systemy komutacji wiadomości. Zasady komutacji danych. Struktura sieci z komutacją pakietów. Protokoły, terminale i nadzorowanie sieci. Geografia sieci, jej niezawodność i trasowanie. Oprogramowanie systemów komutacji pakietów. Przegląd zasad projektowania sieci teledacyjnych.
Celem książki jest bliższe zapoznanie czytelników z nową dziedziną techniki, powstałą ze zbliżenia komputerów i telekomunikacji. Przeznaczona jest przede wszystkim dla teleinformatyków. Przydatna także studentom wydziału elektroniki, na kierunkach telekomunikacji i informatyki.

● **SIMPSCRIPT jest do symulacji systemów — WEREWKA J., Wyd. Akademii Górniczo-Hutniczej im. S. Staszica w Krakowie, Kraków 1979, s. 178, cena 23 zł, skrypty uczelniane nr 690**

Podstawowe instrukcje języka SIMPSCRIPT. Zmienne trwałe systemu, tablice nieprostokątne, generowanie liczb pseudolosowych. Tworzenie tabulogramów wyjściowych. Obiekty chwilowe i ich atrybuty. Stosowanie podprogramów. Zbiory obiektów. Stosowanie instrukcji FIND, COMPUTE. Organizacja programu symulacyjnego. Budowa programu symulacyjnego.
Opracowanie to jest materiałem pomocniczym, przeznaczonym dla studentów informatyki (przedmiot — Języki symulacyjne), a także dla studentów innych kierunków.

● **Język symulacyjny MIMIC — BUDZIASZEK J., Wyd. Akademii Górniczo-Hutniczej im. S. Staszica w Krakowie, Kraków 1979, s. 94, cena 7 zł, skrypty uczelniane nr 710**

Metodyka i możliwość modelowania układów dynamicznych za pomocą języka MIMIC. Pojęcia podstawowe. Opisy instrukcji i funkcji MIMIC. Wybrane zagadnienia numeryczne i prace procesora. Przykłady programów.
Skrypt przeznaczony jest nie tylko dla studentów, może być również przydatny dla inżynierów i pracowników naukowych przy modelowaniu i projektowaniu złożonych systemów dynamicznych.

● **Obliczenia statystyczne doświadczeń. Instrukcja dla użytkowników programów na EMC Zakładu Metodyki Badań i Informatyki — FILIPIAK K., KRZYMUSKI J., WILKOS S. Wyd. Instytutu Uprawy Nawożenia i Gleboznawstwa, Puławy 1979, s. 68**

Cel i zakres instrukcji. Podstawowe informacje o obliczeniach statystycznych. Rodzaje doświadczeń. Podstawowe testy statystyczne stosowane w programach ZMB I. Arkusze danych. Tabulogramy wyników obliczeń. Programy obliczeń. Współpraca użytkowników z Zakładem Badań i Informatyki.
Instrukcja ma na celu ułatwienie pracownikom Instytutu Uprawy Nawożenia i Gleboznawstwa (i innych instytucji), korzystania z usług metodyczno-obliczeniowych Zakładu Metodyki Badań i Informatyki (ZMBT) oraz ich rozszerzenie i pogłębienie w badaniach rolniczych.

● **System HYDRO. Regionalne banki danych hydrogeologicznych — STENZEL P., BERESTKA A., Wyd. Geologiczne, Warszawa 1979**

Cz. 1. Wiadomości podstawowe. Cele, zadania, ogólna koncepcja systemu HYDRO. Organizacja biblioteki programów systemu HYDRO. Opis techniczny warunków eksploatacji systemu, przykłady eksploatacji, uwagi dotyczące rozwoju. Arkusze kodowe.
Cz. 2. Instrukcja kodowania bazy danych. Cele i zadania karty syntetycznej. Struktura informacji w wierceniu (ujęciu) hydrogeologicznym. Opis karty syntetycznej. Instrukcja kodowania. Arkusze kodowe.
Cz. 3. Słowniki.
Materiały przeznaczone są dla inżynierów hydrogeologów.

● **Współczesne półprzewodnikowe układy pamięciowe — KOSLACZ S., RZYMKOWSKI K., Wyd. ROINTE Energetyki i Energii Atomowej, Warszawa 1979, s. 208. Biblioteka postępów techniki jądrowej**

Wprowadzenie do zagadnień przechowywania informacji. Technologia wykonywania scalonych układów półprzewodnikowych. Podstawowe elementy układów scalonych i układów pamięciowych. Pamięci z dostępem swobodnym RAM. Pamięci skojarzeniowe CAM. Pamięci stałe ROM. Mikroprogramowanie. Słownik wybranych terminów i skrótów.
Książka przeznaczona jest dla inżynierów elektroników specjalizujących się w układach cyfrowych.

● **Programowanie w ALGOLU — WAJS W., Wyd. Akademii Górniczo-Hutniczej im. S. Staszica w Krakowie, Kraków 1979, s. 333, cena 27 zł, skrypty uczelniane nr 685**

Opis języka wzorcowego, symbole podstawowe, łańcuchy, wielkości klasy i obszary działania. Wyrażenia: arytmetyczne, boolowskie, mianujące. Instrukcja. Opisy. Procedury we/wy w ALGOLU 1900. ALGOL i GEORGE. Testowanie programu w języku ALGOL 1900.
Skrypt przeznaczony jest dla studentów kierunków informatycznych wyższych uczelni technicznych.

● **FORTRAN i ALGOL dla maszyn ODRA serii 1300 w układzie porównawczym — JANKOWSKI A., Wyd. Politechniki Gdańskiej, Gdańsk 1979 r., s. 306, cena 29 zł**

Wiadomości ogólne, pojęcia i oznaczenia podstawowe. Instrukcje podstawienia, warunkowe logiczne, sterujące. Rezerwacja miejsc w pamięci maszyny. Instruktory programów. Podprogramy. Instrukcje we/wy. Kompilacja programów. Dodatki.
Podręcznik przeznaczony jest dla studentów wyższych uczelni technicznych i programistów. Przydatny przy nauczaniu przedmiotu „Elektroniczna technika obliczeniowa", wykładanego na wszystkich wydziałach Politechniki Gdańskiej.

● **Zasady generacji pobudzeń testowych na podstawie algorytmu dla uszkodzeń logicznych w układach cyfrowych — PIECHA J., Wyd. Uniwersytetu Śląskiego, Katowice 1979, s. 132, cena 45 zł**

Wstęp. Pojęcia podstawowe. Γ — algorytm generacji pobudzeń testowych dla układów cyfrowych małej skali integracji. Zasady generacji pobudzeń testowych dla układów modułowych. Organizacja obliczeń numerycznych. Licznik binarny typ 401. CAMAC — przykład zastosowania algorytmu Γ. Podsumowanie. Wnioski. Dodatek: Opis programu automatycznej generacji pubudzeń testowych.
Materiały przeznaczone są dla projektantów, operatorów i programistów systemów cyfrowych.

● **Minikomputer MERA 303 w pracowni fizycznej — PUCH A., Wyd. Wyższej Szkoły Pedagogicznej w Rzeszowie, Rzeszów 1979, s. 160, cena 13 zł**

Organizacja minikomputera MERA 303. Język konwersacyjny MINI-MERA-BASIC. Błędy obliczeń i pomiarów. Metody statystyczne opracowania wyników pomiarów wartości danej wielkości fizycznej i zależności pomiędzy dwoma wielkościami fizycznymi. Metody numeryczne. Dodatek. Skrypt przeznaczony jest dla studentów fizyki odbywających przewidziane programem ćwiczenia laboratoryjne w różnego typu pracowniach.

Rachunkowość nie doinformatyzowana

Panuje zgodna opinia, że istotnym bodźcem dla rozwoju informatyki i jej społecznej recepcji, przynajmniej w pierwszym okresie po jej powstaniu, są wyniki jej zastosowań gospodarczych, jako najbardziej widoczne i powszechnie odczuwalne. Ponieważ kryzys naszej gospodarki, narastający stopniowo od kilku lat i ujawniony z całą ostrością latem 1980 r., wyznaczył w Polsce oczywistą dla wszystkich cezurę trendu rozwojowego informatyki i jej zastosowań, warto się może zastanowić, jakie rzeczywiste rezultaty w zastosowanich gospodarczych już osiągnęliśmy oraz jakie są w związku z tym wnioski na przyszłość.

Uważa się najczęściej, że okres ten, który z punktu widzenia zastosowań gospodarczych rozpoczął się praktycznie w połowie lat sześćdziesiątych, a więc mniej więcej z dziesięcioletnim opóźnieniem w stosunku do sytuacji światowej, zamyka się aktualnie opóźnieniami, które trzeba szacować na lat więcej niż dziesięć. Pomimo pewności, że wszędzie tam, gdzie zastosowania informatyki pojawiły się w praktyce gospodarczej nie może już być od nich odwrotu, a zatem — że stały się one trwałym elementem procesu gospodarowania, stan obecny wykazuje poważne mankamenty, a oceny społeczne dotychczasowych wyników nie są i nie mogą być korzystne.

Sądzę, że uwzględniając nawet istotne ograniczenia i specyficzne warunki, w których informatyka polska znajdowała się dotąd, za jeden z głównych czynników niepowodzeń i opóźnień przyjdzie uznać powierzchowny i niechętny, naznaczony nieznajomością rzeczy stosunek informatyków profesjonalnych do rachunkowości, informatyków — dodajmy — zajmujących się zastosowaniami gospodarczymi. Chciałbym skoncentrować się na tym właśnie problemie.

Wygodnie będzie uzasadnić powyższą tezę na dwóch poziomach: systemów przetwarzania danych i informatycznych systemów zarządzania (w skali obiektowej i na szczeblach wyższych), mimo iż w istocie oba te poziomy tworzą jednolitą, spójną całość.

Gdy chodzi o pierwszy z nich, warto sobie przypomnieć, że pojęcie przetwarzania danych uformowane zostało w Stanach Zjednoczonych w połowie lat pięćdziesiątych, pierwotnie jako przeciwstawienie w stosunku do obliczeń numerycznych. Wynikało to z przewidywanych bądź faktycznych zastosowań środków informatyki w rachunkowości, tradycyjnej gałęzi wiedzy ekonomicznej, realizowanej systemowo przez stulecia techniką zapisów ręcznych, współcześnie zaś — za pomocą różnych urządzeń mechanicznych i elektromechanicznych (maszyn do księgowania, maszyn licząco-analitycznych). Rachunkowość dała pierwszy impuls w kierunku automatyzacji prowadzenia kartotek, tj. manipulowania wielkimi zbiorami danych, a także — technologii zbierania masowych i stale napływających danych źródłowych.

W połączeniu z prostymi procedurami obliczeniowymi, które opierają się z reguły na trywialnych algorytmach arytmetyki elementarnej, rachunkowość stała się wyraźnie odrębną dziedziną zastosowań. Dlatego też czytelnicy INFORMATYKI (nr 6—8/1972) mogli dowiedzieć się z artykułu pt. „Co to jest informatyka?", że rachunkowość jest jednym z ważnych źródeł informatyki, na równi z metodami numerycznymi, elektroniką i teorią obwodów elektrycznych.

Jeśli tak, to jest oczywiste, że potrzeby i uwarunkowania rachunkowości, jako jedynego faktycznie działającego i stosowanego na wielką skalę systemu przetwarzania danych, pochodzącego z epoki przedinformatycznej, powinny być przy zastosowaniach gospodarczych nowej techniki uwzględnione w pierwszej kolejności (jako stosunkowo proste i łatwe do wdrożenia) — w sposób kompleksowy. Taką właśnie linię rozwojową można było zaobserwować w krajach przodujących w informatyce.

Nie obyło się i w tych krajach bez kontrowersji między „starym" i „nowym", jednakże dziś — nie mówiąc o masowości, poprawności technologicznej i formalnej oraz opłacalności zastosowań (przy różnorodności sprzętu, od wyspecjalizowanych minikomputerów do równie wyspecjalizowanych sieci, obsługujących dziesiątki tysięcy klientów) — świadectwem właściwego podejścia są dwa istotne fakty: udział teorii rachunkowości i jej centralnego narzędzia myślowego (metody bilansowej) w analizie i opisie struktury informacyjnej i struktury danych (przy kompleksowej budowie bazy danych dla celów gospodarczych) oraz automatyzacja procedur kontrolno-rewizyjnych rachunkowości, objęta odrębną już dzisiaj dyscypliną zawodową i teoretyczną, tzw. ADP-Auditing.

U nas natomiast w omawianym okresie odczuwało się brak związków zastosowań gospodarczych informatyki z rachunkowością i jej aparatem wykonawczym, a także — lekceważący stosunek informatyków do rachunkowości, potwierdzony dziwną doktryną o niepotrzebnym, a nawet szkodliwym „przetwarzactwie", niegodnym prawdziwej informatyki. Wywoływało to często najzupełniej zbyteczne napięcia pomiędzy zespołami informatyków a służbami finansowo-księgowymi, przejawiającymi skłonności do konserwatywnego traktowania swych obowiązków i reguł działania i — bywało — niechętnie witającymi komputerowe nowinki. Choć dzisiaj sytuacja uległa zmianie, ówczesne zlekceważenie spraw rachunkowości przyniosło przykre następstwa. A mianowicie:

- nie dysponujemy w skali masowej żadnym sprzętem, który umożliwiałby bądź zdecentralizowane prowadzenie rachunkowości (elektroniczne maszyny do księgowania lub wyspecjalizowane minikomputery), bądź też scentralizowane, gdzie przygotowywanie masowych danych realizowane jest w wyodrębnionych ośrodkach obliczeniowych.
- poziom usług informatycznych dla rachunkowości nie jest zbyt wysoki (prymitywne na ogół i niestandaryzowane systemy F-K o niskim stopniu powielarności i dużej wrażliwości na zmiany); technologie informatyczne są tu najczęściej przestarzałe, istnieje bowiem stały hamulec w postaci wspomnianego wyżej braku urządzeń do wpro-

wadzania danych, a także odczuwa się jeszcze (choć w ostatnich latach jakby w mniejszym stopniu) iż informatycy traktują zadania z tego zakresu jako drugorzędne.

- w związku z powyższym służby finansowo-księgowe znajdują się w bardzo trudnej sytuacji, w której — na skutek wykruszającego się sprzętu elektromechanicznego i (przy stałym niedoborze kadr) niemożności powrotu do technik ręcznych — może za kilka lat dojść realnie do groźby wstrzymania ich pracy
- podobne niebezpieczeństwo zaczyna zarysowywać się przed aparatem bankowym, mimo iż dysponuje on własną siecią ośrodków obliczeniowych
- prace nad automatyzacją procedur kontrolno-rewizyjnych rachunkowości nie zostały jeszcze rozpoczęte, biegli księgowi i rewidenci stronią często od informatycznych rozwiązań systemowych, a nawet niekiedy nie lubią czytać tabulogramów, zaś samo pojęcie ADP — Auditing nie jest jeszcze w ogóle znane i rozumiane (tutaj opóźnienie sięga na pewno 20 lat).

Istnieją też poważne trudności na drugim poziomie — informatycznych systemów zarządzania, tak obiektowych, jak wyższego rzędu.

W skali obiektowej trudności te powoduje koncepcja wielodziedzinowego systemu zarządzania, przyjęta w naszym kraju jako jedyna, gdy w istocie wykazuje ona poważne braki metodologiczne. Nie zapewnia bowiem — jeśli już nie w teorii, to w praktyce — spójności podsystemów składowych (nie można dziś nawet o jednym krajowym przedsięwzięciu tego typu powiedzieć, że osiągnęło ono rzeczywiście pełną kompleksowość i uzyskało całkowitą spójność między uruchomionymi podsystemami; zwykle funkcjonują tylko poszczególne podsystemy lub ich wycinki) i w efekcie — nadaje się tylko do zastosowań przemysłowych.

Ponadto, o czym informatycy zdają się w ogóle nie wiedzieć, koncepcja ta powoduje rozerwanie na oddzielne kawałki kompleksowego systemu rachunkowości, który w naszym kraju od trzydziestu lat obejmuje organizacyjnie takie podsystemy analityczne, jak: ewidencja ilościowo-wartościowa materiałów, ewidencja środków trwałych, rachuba płac itd. W rezultacie, podsystemy F-K, nawet gdyby były najstaranniej realizowane, są zawsze tylko kaleką namiastką dawnego systemu rachunkowości, a służby finansowe mają częstokroć trudności nie tylko z zapewnieniem obowiązujących standardów jakościowych całej ewidencji, ale nawet ze sporządzeniem bilansu, który z samego podsystemu F-K nie wynika.

Nie chodzi mi — oczywiście — o cofnięcie rozwiązań, które się już przyjęły i uzyskały po okresie doświadczeń właściwy standard, ani o negację rzeczywistych osiągnięć tam, gdzie mają one miejsce. Myślę natomiast że gdyby dopuszczono tu — jako realną alternatywę koncepcji systemu wielodziedzinowego — ideę kompleksowego automatyzowania systemów rachunkowości, koszty tych przedsięwzięć (pieniężne i społeczne) byłyby niższe, a funkcjonujące dziś wycinkowe moduły czy podsystemy (na czele z ukochanymi przez ośrodki obliczeniowe ewidencjami ilościowo-wartościowymi materiałów, które występują zwykle pod szumną nazwą systemów gospodarki materiałowej, gdy tak naprawdę są tylko cząstką systemu). zastąpione by zostały przez dobrze funkcjonujące, pełne systemy rachunkowości, stanowiące podstawę tzw. automatyzacji zarządzania.

Warto tu jednak zauważyć, choć częściowo wykracza to już poza temat, że sama idea automatyzacji zarządzania, interpretowana w początkowej fazie przez informatyków zbyt dosłownie, spowodowała znaczne szkody. Nawet ujęta właściwie, jako komputerowe wspomaganie procesów decyzyjnych, wymaga ona dobrze określonego modelu gospodarowania, tak na szczeblu obiektowym, jak w skali gospodarki narodowej. Przenoszenie koncepcji zachodnich, opartych bez wyjątku na rynkowym modelu gospodarki, nie mogło przynieść efektu tam, gdzie ten model nie był wprowadzony. Fakt, że wspomniane wyżej systemy gospodarki materiałowej nie wspomagają u nas zarządzania tą gospodarką i są praktycznie sprowadzone do roli narzędzi ewidencji, jest właśnie przykładem prawdziwości tego stwierdzenia.

Informatycy często nie uświadamiali sobie tego; gorzej jeszcze, bo w pewnym okresie usiłowali wyręczać ekonomistów i rozwiązywać za nich zagadnienia modelowe. Niepowodzenia tych działań uznać można za szczęście dla informatyki, jeśli spojrzeć na nie z perspektywy ostatnich wydarzeń.

Na tej samej wreszcie linii znajduje się sprawa makrosystemów zarządzania gospodarką. Po pierwszym okresie nadmiernie ambitnych i szerokich koncepcji, niemożliwych do zrealizowania w dającej się przewidzieć przyszłości, skoncentrowano się wprawdzie na dwóch takich systemach (CENPLAN i SPIS), jednak — rzecz charakterystyczna i ważna z punktu widzenia tez, które są tu prezentowane — i w tym wypadku nikt nie pomyślał o systemie bilansów gospodarki narodowej, naturalnej nadbudowie jednostkowych systemów rachunkowości i emitowanej przez nie sprawozdawczości finansowej.

Byłby to makrosystem stosunkowo łatwy do zaprojektowania i wdrożenia, przynoszący jednocześnie znaczne efekty informacyjne i porządkujące, a również tani i nie wymagający wyrafinowanych środków technicznych. Był on do pomyślenia jako podsystem SPISU lub CENPLANU, bądź jako najistotniejsza część SEIFU, resortowego systemu Ministerstwa Finansów, który ciągle jeszcze pozostaje w sferze mglistych koncepcji. Jego brak jest jeszcze jednym aspektem niedoceniania lub nieznajomości rachunkowości (i związanej z nią metody bilansowej) w środowisku informatycznym.

Myślę, że w ten sposób można naczelną tezę mojej wypowiedzi uznać za udowodnioną. Wnioski nasuwają się same. W sferze filozoficznej jest to myśl o niewłaściwym działaniu, bez rozeznania rzeczywistych problemów i prawdziwej, pogłębionej znajomości rzeczy, nawet jeśli dysponuje się narzędziem o wielkich możliwościach.

Aktualny jest zatem postulat kompleksowego rozwiązania powstałych zaniedbań (sprzęt, oprogramowanie, systemy, koordynacja) tak szybko, jak będzie to możliwe, spraw tych bowiem nie można przeskoczyć, nawet jeśli uzyskano określone efekty na innych, zdawałoby się bardziej nowoczesnym polach. Załamanie się systemu rachunkowości (z aparatem bankowym włącznie) jest teraz realnym niebezpieczeństwem. Nie sposób nawet wyobrazić sobie możliwych konsekwencji.

Tadeusz PECHE

Cena 30 zł.