TOWARZYSTWO NAUKOWE ORGANIZACJI
I KIEROWNICTWA
Oddział Szczecinie

VII Kołobrzeskie Dni Informatyki

INFOGRYF'84

INFORMATYKA W ZARZĄDZANIU PRZEDSIĘBIORSTWEM

Zbiór referatów programowych

TOM 1

Szczecin—Kołobrzeg 1984

TOWARZYSTWO NAUKOWE ORGANIZACJI
I KIEROWNICTWA
Oddział Szczecinie

VII Kołobrzeskie Dni Informatyki

INFOGRYF'84

INFORMATYKA W ZARZĄDZANIU PRZEDSIĘBIORSTWEM

Zbiór referatów programowych

TOM 1

Szczecin—Kołobrzeg 1984

Sekretarz naukowy konferencji i koordynacja pracy:

dr WOJCIECH OLEJNICZAK

Opracowanie redakcyjne:

mgr HELENA CZEPNIAKIEWICZ

Spis treści

str.

Andrzej Blikle — DENOTACYJNA METODA SPECYFIKACJI OPROGRAMOWANIA 5

Wiesław Flakiewicz — SYSTEMY INFORMATYCZNE ZARZĄDZANIA KIERUNKI I BARIERY ROZWOJU 31

Hermann Krallmann — COMPUTERUNTERSTÜTZTE UNTERNEHMENS-PLANUNG 51

Wojciech Olejniczak — ZASADY PROJEKTOWANIA SYSTEMÓW INFORMATYCZNYCH PODŁUG STRUKTUR .. 71

Jan Pełech — NIEKTÓRE PROBLEMY ZASTOSOWAŃ W KRAJU SYMULACYJNYCH GIER DECYZYJNYCH 89

Celette Relland — AN EXTENSION OF THE THREE LEVEL ARCHITECTURE FOR INFORMATION SYSTEM DESIGN. 129

Leszek Sankowski — PRZEGLĄD ZASTOSOWAŃ GOSPODARCZYCH MINIKOMPUTERÓW 149

Jan Sztajer — BARIERY INFORMATYZACJI ZARZĄDZANIA PRZEDSIĘBIORSTWEM PRZEMYSŁOWYM 161

Zdzisław Szyjewski — METODA PROJEKTOWANIA I REALIZACJI OPROGRAMOWANIA ZŁOŻONEGO SYSTEMU INFORMATYCZNEGO NA PRZYKŁADZIE SYSTEMU TRANSTER 183

W.M. Turski — MIEJSCE SPECYFIKACJI W STRUKTURZE OPROGRAMOWANIA 197

Stanisław Waligórski — MIĘDZY PRAKTYKĄ A TEORIĄ: O NOWYCH METODACH INFORMATYKI ... 211

Andrzej Blikle [1]

DENOTACYJNA METODA SPECYFIKACJI OPROGRAMOWANIA [2]

1. Wstęp

W ostatnich latach dokonuje się w świecie przemysłu software'owego swoista rewolucja intelektualno-technologiczna. Polega ona na coraz szerszym wprowadzaniu do praktyki formalnych metod projektowania, wytwarzania i pielęgnacji oprogramowania. Metody znane jeszcze kilka lat temu wyłącznie w wąskich środowiskach akademickich zaczynają propagować do przemysłu. Dokonuje się to dwoma kanałami. Po pierwsze, mury wielu uczelni na świecie opuszczają dziś absolwenci przygotowani do stosowania tych metod na skalę przemysłową. Po drugie, wiele firm software'owych inicjuje szeroko zakrojone programy badawczo-szkoleniowe na własny użytek.

Wśród metod zdobywających uznanie praktyków najpowszechniej stosowaną jest tzw. metoda denotacyjna wraz z jej przemysłową wersją VDM /ang. Vienna Development Method/. Celem niniejszego artykułu jest przekazanie czytelnikowi polskiemu kilku sygnalnych wiadomości dotyczących tej metody, a także warunków i przyczyn jej powstania, powodów jej sukcesu, zakresu zastosowań, dostępnej w kraju i na świecie literatury. Oczywiście niewielki raport konferencyjny nie może pretendować do roli chociażby wprowadzenia w metodę i jej zastosowania. Wymagałoby to tekstu o

---

1/ Instytut Podstaw Informatyki Polskiej Akademii Nauk.

2/ Praca jest kopią oryginału, który zamieszczony jest w materiałach konferencji Polskiego Towarzystwa Informatycznego "Współczesne kierunki rozwoju informatyki" organizowanej w ramach Jesiennej Szkoły PTI, Rydzyna, Październik 1984.

objętości sporej książki oraz co najmniej pełnosemestralnego wykładu. Niżej opisany ma jednak nadzieję, że ta krótka informacja przyczyni się do wzrostu zainteresowania metodą dekompozycyjną w naszym kraju i spowoduje podjęcie działań zmierzających do jej upowszechnienia.

2. Trochę historii i stan obecny

Inżynieria oprogramowania jest jedyną dziś na świecie dyscypliną techniczną, w której projektowanie dużych, złożonych i spełniających nieraz bardzo odpowiedzialne funkcje systemów przemysłowych prowadzone jest najczęściej metodami rzemieślniczymi. Tam, gdzie w elektronice, mechanice, budownictwie, przeprowadza się obliczenia oparte na dobrze ugruntowanej teorii, tam w inżynierii oprogramowania twórcy systemu opierają się /najczęściej/ jedynie na własnym doświadczeniu i intuicji. Gwarancję jakości i niezawodności systemu uzyskuje się właściwie wyłącznie na drodze testowania i wstępnej eksploatacji. Mimo że te zabiegi pochłaniają nierzadko połowę /i więcej !/ kosztu całego projektu, uzyskiwane wyniki bywają dalekie od zadowalających. Jest powszechnie znanym faktem, że większość poważnych katastrof w programie kosmicznym USA, a także szereg innych spektakularnych wypadków - np. w elektrowniach atomowych - spowodowanych zostało błędami w systemach oprogramowania.

Opisane wyżej problemy są od lat przedmiotem badań wielu naukowców. Badania takie rozpoczęto właściwie jednocześnie z narodzinami informatyki. Już w roku 1949 znakomity matematyk angielski, pionier informatyki teoretycznej, Alan Turing /twórca znanego pojęcia maszyny Turinga/ ogłosił pracę /16/ poświęconą matematycznej metodzie weryfikacji zgodności programu za specyfikacją. Niestety, praca Turinga - zapewne zbyt wybiegająca w przyszłość - nie została przez współczesnych zauważona. Dopiero w 18 lat później R.W. Floyd /8/ odkrywszy i opisawszy tę metodę ponownie doprowadził do jej wielkiej popularności. Dziś wchodzi ona do klasyki matematycznej teorii programowania.

Poważny rozwój badań matematycznych w teorii programowania datuje się mniej więcej od połowy lat sześćdziesiątych. Jako pierwsze zostały opisane i weszły na stałe do repertuaru narzędzi praktyków metody związane z definiowaniem składni języków programowania /BNF/ oraz ich rozbiorem syntaktycznym /parsing/. Dziś nie sposób wyobrazić już sobie, aby przy projektowaniu i implementacji jakiegokolwiek języka programowania nie posłużono się tymi metodami.

Opis składni języka jest oczywiście tylko jednym spośród wielu problemów, jakie należy rozwiązać przy projektowaniu języków i systemów. Drugą, bardzo ważną grupę stanowią zagadnienia związane z opisem semantyki oraz badania poprawności tej semantyki ze względu na różne kryteria. W tej grupie zagadnień natrafiono na szereg trudnych problemów teoretycznych i praktycznych. Okazało się, że do opisu i analizy semantyki stworzyć trzeba nowe narzędzia matematyczne. Mimo że w badania nad tymi problemami zaangażowanych było wiele ośrodków naukowych na świecie, pierwsze wyniki mogące mieć znaczenie praktyczne zaczęto uzyskiwać dopiero w drugiej połowie lat siedemdziesiątych. I wtedy jednak były to - mówiąc językiem przemysłowym - wyniki prototypowe, nie wsparte jeszcze ani dostatecznym doświadczeniem eksperymentalnym, ani tym bardziej żadnymi narzędziami wspomagania komputerowego. Nie ułatwiało to oczywiście ich adaptacji w praktyce. Do trudności tych dochodziły jeszcze poważne opory natury psychologicznej. Długi okres oczekiwania na praktycznie użyteczne formalne metody projektowania software'u doprowadził w wielu środowiskach do przekonania, że tzw. zdrowy rozsądek, intuicja, doświadczenie i odpowiedni budżet na koszty testowania są jedyną rozsądną metodą zapewniającą powodzenie w praktyce.

Tymczasem ta sama praktyka stwarza nowe fakty. Burzliwy rozwój zastosowań informatyki prowadzi do oczywistej dziś konieczności radykalnego polepszenia niezawodności systemów software'owych. Związane jest to z dwoma czynnikami. Po pierwsze, współczesnym systemom powierza się coraz bardziej odpowiedzialne zadania, od których właściwego wykonania zależy nierzadko

gospodarcza, społeczna czy polityczna równowaga świata. Po drugie, rozwój mikroinformatyki prowadzi do powstawania systemów, których kopie sprzedawane są nie w tysiącach - jak to bywało dotychczas - ale w milionach egzemplarzy. Wszystko to, jak wskazuje McKeeman /12/, doprowadziło w ostatnim dziesięcioleciu do dramatycznego wzrostu wysokości strat, jakie spowodować może błąd w oprogramowaniu. Z drugiej strony wiadomo już, że jakości systemów oprogramowania nie da się racjonalnie poprawić metodami tradycyjnymi.

Opisana wyżej sytuacja jest oczywiście dobrze znana producentom oprogramowania na całym świecie. Stąd też szereg poważnych firm przemysłowych wykazuje coraz większe zainteresowanie możliwością wprowadzenia metod formalnych do stałego repertuaru narzędzi w swoich fabrykach software'u. Wymaga to oczywiście zorganizowania odpowiednio szeroko zakrojonej akcji szkolenia. Wśród dużych firm jako pierwsza zdecydowała się na taką inicjatywę IBM. Od szeregu już lat organizuje ona regularne kursy dla swoich pracowników w tym zakresie /por. C. Jones /11//. Niezależnie od tego wiedeńskie laboratorium IBM opracowało wspomnianą już wcześniej przemysłową wersję metody denotacyjnej zwaną VDM wraz z metajęzykiem specyfikacji software'u META-IV. Metody VDM nie należy mylić z opracowanym wcześniej w tym samym laboratorium i mniej udanym metajęzykiem VDL /Vienne Definition Language/.

VDM wraz z metodami pokrewnymi oraz niezbędną podbudową teoretyczną wchodzą dziś do regularnych programów nauczania w wielu uniwersytetach na świecie. Dla przykładu, wykłada się je w Anglii na jedenastu uniwersytetach, w RFN na dziewięciu, we Francji i we Włoszech na czterech, w Danii i Szwecji na trzech, w Belgii na dwóch /dane pochodzą z roku 1983/. Liczba uniwersytetów nauczających podstaw teoretycznych tych i podobnych metod formalnych jest jeszcze większa. Niezależnie od tego wiele firm przemysłowych prowadzi własne szkolenia i badania w tym zakresie. Obok wspomnianego już wcześniej IBM wymienić tu można ICL, Standard Telecomunication Laboratories Limited, AEG Telefunken, Siemens, Olivatti, Philips, VEB Robotron, Xerox, General Electric, Ford Aerospace, Honeywell, Bell Laboratories i wiele in-

nych. Ostatnio rządy państw Wspólnoty Europejskiej postanowiły uruchomić duży projekt badawczo-wdrożeniowy, poświęcony rozwojowi i przemysłowym zastosowaniom formalnych metod projektowania i rozwoju software'u. Jeden z głównych kierunków prowadzonych tam prac dotyczyć będzie metody VDM.

3. Dokładniej o przyczynach, dla których metody formalne stosowane są w inżynierii oprogramowania

Wytworzenie dużego oprogramowania czy to systemowego, czy użytkowego jest złożonym procesem przemysłowym konsumującym wiele dziesiątek osobo/lat pracy i prowadzącym do poważnych problemów technologicznych i organizacyjnych. Od systemów oprogramowania wymaga się, aby spełniały szereg wymogów konstrukcyjnych i użytkowych, wśród których do najważniejszych należą:

1/ Zgodność z założoną specyfikacją zwana również niekiedy po prostu poprawnością.

2/ Niezawodność obejmująca odporność systemu na błędy własne, środowiska, sprzętu i użytkownika.

3/ Wydajność w sensie oszczędnego korzystania z zasobów takich, jak: czas, pamięć, urządzenia zewnętrzne itp.

4/ Rozszerzalność, tzn. możliwość rozszerzenia systemu w przyszłości bez konieczności dokonywania zbyt wielu zmian w istniejącej części.

5/ Przenaszalność, tzn. możliwość łatwej adaptacji systemu do różnych konfiguracji systemowo-sprzętowych.

6/ Łatwość pielęgnacji, tzn. podatność systemu na wprowadzanie niewielkich zmian bez nadmiernego ryzyka wprowadzenia błędów.

7/ Odzyskiwalność /ang. reusability/, tzn. możliwość ponownego użycia modułów systemu w innych systemach lub w nowych generacjach tego samego systemu.

8/ Odporność na włamanie /ang. security/, tzn. zabezpieczenie systemu przed nieporządanym użytkownikiem.

9/ Zgodność ze standardami, np. dotyczącymi protokołów komunikacyjnych, reprezentacji informacji, szybkość reakcji itp.

10/ Łatwość użycia, tzn. prostota koncepcyjna systemu, umożliwiająca szybkie jego opanowanie oraz pomoc udzielana użytkownikowi przez system w trakcie pracy.

11/ Dobra dokumentacja będąca podstawą do dalszej obsługi systemu przez producenta /pielęgnacja, modyfikacja itp./, a także do napisania odpowiednich podręczników dla różnych grup użytkowników.

Wymienione wyżej wymagania dotyczące jakości systemu oprogramowania powinny znaleźć odzwierciedlenie i być realizowane we wszystkich etapach projektowania i implementacji systemu. Wymaga to posługiwania się we wszystkich tych etapach w miarę jednolitym i dostatecznie precyzyjnym językiem tak, aby przechodzenie z poziomu na poziom - od założeń poprzez wszystkie fazy projektowe do kodu wynikowego - gwarantowało zgodność treści pomiędzy poszczególnymi poziomami oraz możliwość formalnej weryfikacji tej zgodności. Konieczne jest również, aby opis każdego modułu systemu na każdym etapie jego projektowania i tworzenia był kompletny i jednoznaczny w ramach danego etapu.

Tradycyjna metoda wytwarzania software'u stosowana do dziś w wielu ośrodkach na świecie przyjmuje w zasadzie trzy poziomy językowe dla prac projektowych nad systemem:

1/ całkowicie nieformalny poziom języka potocznego,

2/ nieco bardziej formalny poziom różnego rodzaju schematów, np. sieci działań,

3/ poziom języka programowania.

Poważną wadą tej metody jest całkowita intuicyjność poziomu pierwszego, niemal całkowita intuicyjność poziomu drugiego oraz brak systematycznych metod przechodzenia z poziomu na poziom oraz sprawdzenia poprawności tych przejść. Aby uprzytomnić sobie rozmiar związanych z tym problemów, zacytujemy za O. Herzogiem /10/ dosyć typowy przykład rozwoju dokumentacji projek-

towej dużego systemu softwareowego. System ten realizować miał mechanizm komunikacji pomiędzy niezależnie wykonującymi się procesami.

etap projektu	język opisu
założenia wstępne - 1 system - 61 stron opisu	j. potoczny
wstępna specyfikacja projektowa - 60 modułów - 589 stron opisu	j. potoczny
pełna specyfikacja projektowa - 544 funkcje - 2100 stron opisu	j. potoczny
modułowa specyfikacja logiczna - 4385 modułów - 12000 stron opisu	j. sieci działań
kod wynikowy - 3800000 linii kodu	j. programowania

Posługiwanie się językiem potocznym przy specyfikowaniu i projektowaniu softwareu prowadzi z reguły do licznych błędów. Przytoczmy tu ponownie za Herzogiem /10/ następującą statystykę typów błędów wykonaną dla dużej grupy projektów przemysłowych, prowadzonych metodą tradycyjną:

- błędy w projekcie	45 %
- błędy kodu	25 %
- błędy wprowadzane przy usuwaniu innych błędów	23 %
- pozostałe błędy	7 %

Zauważmy, że błędy trzeciej grupy zawdzięczają swoje istnienie w dużej mierze błędom grupy pierwszej. Tak więc udział błędów projektowych w ogólnym ich bilansie jest jeszcze większy niż to wynika bezpośrednio z przytoczonej statystyki.

Do podstawowych wad specyfikacji software'u pisanych w języku potocznym zaliczyć należy ich niejednoznaczność, niekompletność oraz niepoddawanie się ani testowaniu ani analizie logicznej, tzn. dowodzeniu poprawności. W tej sytuacji błędy powstające w fazie projektu są wykrywalne w większości dopiero na etapie testowania kodu. Prowadzi to do wysokich, związanych z tym kosztów, gdyż, jak to wynika z powyższej tabelki, koszt wykrycia i usunięcia błędu rośnie eksponencjalnie wraz z etapem, na którym błąd wykrywamy /10/:

etap	relatywny koszt wykrycia i usunięcia błędu
- projektowanie	1
- testowanie modułów	4
- testowanie funkcji systemu	9
- testowanie całego systemu	15
- eksploatacja	30

Te i liczne inne jeszcze problemy związane z wytwarzaniem i eksploatacją systemów oprogramowania skłaniają coraz więcej środowisk przemysłowych na świecie do stosowania metod formalnych w inżynierii oprogramowania. Oznacza to z jednej strony sięgnięcie do odpowiednich podstaw teoretycznych przy wyborze i opisie mechanizmów projektowanego systemu, a z drugiej strony użycie we wszystkich etapach projektowania języka formalnego w miejsce potocznego. Język taki przypomina zazwyczaj język programowania bardzo wysokiego poziomu - lub notację matematyczną - i ma z zasady precyzyjnie zdefiniowaną semantykę. Posługiwanie się językiem formalnym pozwala na precyzyjne i pełne definiowanie wszystkich elementów systemu na dowolnym etapie projektu. Pozwala również na sprawdzenie poprawności każdego etapu z osobna, a także sprawdzenie zgodności pomiędzy etapami. Oczywiście definicje formalne są takstowo nie mniejsze od nieformalnych, zatem proces ich tworzenia i weryfikacji powinien być wspomagany komputerowo. Odpowiednie systemy wspomagania znajdują się już w fazie projektowo-prototypowej, nie będą jednak dos-

tępne wcześniej jak za kilka lat. Okazuje się jednak, że nawet "ręczne" stosowanie metod formalnych pozwala na kilkakrotne /do dziesięciokrotnego!/ /10/ zmniejszenia ilości błędów w systemie. Wynika to stąd, że metody formalne posługują się precyzyjnym językiem opisu oraz narzucają określoną dyscyplinę pojęciową, opartą na dobrze zdefiniowanym modelu matematycznym. Precyzyjny język i właściwie dobrany model pozwalają na uniknięcie wielu błędów typu niekompletność lub wewnętrzna sprzeczność opisu oraz na zauważeniu wielu podobnych i innych błędów na długo przed przystąpieniem do etapu kodowania, testowania i eksploatacji.

4. Idea metody denotacyjnej i jej metajęzyk

Metoda denotacyjna powstała jako matematyczna technika definiowania semantyk języków programowania. Obecnie stosowana jest ona w znacznie szerszym kontekście i służy do definiowania /projektowania/ wielu rodzajów oprogramowania systemowego i użytkowego takiego, jak kompilatory, systemy operacyjne, systemy obsługi baz danych, architektura sprzętu itd. W tym i następnym rozdziale omówimy pokrótce tę metodę w kontekście jej pierwotnych /ale nadal aktualnych!/ zastosowań, jakimi były definicje języków programowania.

Definicję każdego języka programowania podzielić można na dwie podstawowe części: składnię, zwaną również <u>syntaksą</u> oraz <u>semantykę</u>. Definicja składni jest na ogół generacyjna, tzn. wyjaśnia jak z podstawowych elementów języka, takich jak np.: zmienne i symbole operacji, budować elementy złożone, takie jak wyrażenia, deklaracje, instrukcje itp. Definicję składni formułuje się najczęściej jako gramatykę bezkontekstową, zapisaną w notacji BNF.

Zdefiniowanie semantyki jest problemem znacznie subtelniejszym i trudniejszym. Przede wszystkim należy wyjaśnić, co to jest semantyka języka programowania. W metodzie denotacyjnej przyjmuje się, że semantyka jest funkcją

$$S : L \longrightarrow D$$

gdzie L oznacza język programowania traktowany jako zbiór programów, instrukcji, wyrażeń itp., natomiast D oznacza zbiór odpowiadających im znaczeń, zwanych denotacjami. Denotacje definiowane są najczęściej jako funkcje operujące na stanach pewnej abstrakcyjnej maszyny. Denotacjami wyrażeń są funkcje przekształcające stany na wartości /np. boolowskie, całkowite itp./. Denotacjami instrukcji i deklaracji są funkcje przekształcające stany na stany. Funkcję semantyki definiuje się przez tzw. indukcję strukturalną. Oznacza to, że denotację każdego złożonego obiektu syntaktycznego opisuje się jako złożenie denotacji jego składowych. Pozwala to na bardzo czytelną strukturalizację definicji języka i stanowi istotę metody denotacyjnej.

Aby posługiwać się metodą denotacyjną w praktyce dysponować należy odpowiednią do tego celu notacją. Przy jej ustalaniu trzeba wziąć pod uwagę, że typowe definicje denotacyjne liczą od kilkudziesięciu do kilkuset stron tekstu oraz operują tego samego rzędu liczbą pojęć i oznaczeń. Używana notacja powinna więc być możliwie przejrzysta, sugerować jednolitą metodę mnemotechnicznie czytelnych oznaczeń oraz narzucać pewne ogólne zasady dotyczące struktury definicji denotacyjnych. Powinna być również na tyle sformalizowana, aby poddawała się wspomaganiu odpowiednim systemem komputerowym. Mówiąc krótko, powinna przypominać język programowania strukturalnego wysokiego poziomu.

W obecnej chwili istnieje na świecie kilka notacji, lub jak niektórzy mówią metajęzyków, dostosowanych do metody denotacyjnej. Najpowszechniejszym z nich jest wspomniany już język META-IV /1/, /2/. Ma on z kolei kilka dialektów. Poniżej przedstawiamy - w wielkim skrócie oczywiście - dialekt nazwany META /5/. Jego zastosowanie zilustrujemy na prostym przykładzie w następnym rozdziale.

Niech A i B oznaczają dowolne zbiory,

A | B oznacza sumę teoriomnogościową A i B,

A × B oznacza iloczyn kartezjański A i B,

A^* oznacza zb. wszystkich skończonych ciągów nad A wraz z ciągiem pustym,

$A \to B$ oznacza zb. wszystkich funkcji całkowitych z A do B,

$A \xrightarrow{\sim} B$ oznacza zb. wszystkich funkcji częściowych z A do B.

Przyjmujemy, że $|$, $\times$ oraz * wiążą mocniej niż $\to$ i $\xrightarrow{\sim}$ co pozwoli na opuszczanie wielu nawiasów w wyrażeniach definiujących zbiory.

Zbiory w semantyce denotacyjnej nazywamy tradycyjnie dziedzinami. Ponieważ dziedzin w typowej definicji denotacyjnej jest od kilkudziesięciu do kilkuset wprowadzamy dla nich zawsze nazwy mnemotechniczne oraz definiując dziedzinę wprowadzamy jednocześnie oznaczenie dla jej elementów. To ostatnie jest najczęściej skrótem nazwy dziedziny. Na przykład dziedzinę stanów zdefiniować możemy formułą

sta : Stan = Identyfikator $\xrightarrow{\sim}$ Wartość

którą odczytamy jak następuje: stany są funkcjami częściowymi ze zbioru identyfikatorów w zbiór wartości. Elementy zbioru Stan, tzn. stany, oznaczamy przez sta być może z indeksami.

Wprowadzimy teraz kilka oznaczeń przydatnych przy definiowaniu funkcji. Przez $f : A \to B$ oraz $f : A \xrightarrow{\sim} B$, zapisujemy fakt, że f jest funkcją całkowitą, lub odpowiednio częściową, z A do B. Jeżeli $f : A \xrightarrow{\sim} B$, $a \in A$ oraz $b \in B$ to

$$P[b/a] : A \xrightarrow{\sim} B$$

jest modyfikacją funkcji f taką, że

$$f[b/a](x) = \underline{\text{if}}\ x = a\ \underline{\text{then}}\ b\ \underline{\text{else}}\ f(x)$$

Tę notację uogólnia się w naturalny sposób na przypadek $f[b_1/a_1 \ldots\ldots b_n/a_n]$, gdzie $a_1 \ldots\ldots a_n$ muszą być wzajemnie różne.

Na użytek definicji przez przypadki wprowadzamy notację

$$b \to c,d$$

na oznaczenie $\underline{\text{if}}$ b $\underline{\text{then}}$ c $\underline{\text{else}}$ d gdzie b reprezentuje oczywiście wartość boolowską. Używamy również złożonych formuł warun-

kowych postaci

$$b_1 \to c_1, (b_2 \to c_2 \ldots\ldots (b_n \to c_n, d)\ldots)$$

które najczęściej piszemy w kolumnie oddzielając jej wiersze przecinkami:

$$b_1 \to c_1,$$
$$\ldots$$
$$b_n \to c_n,$$
$$\mathrm{TRUE} \to d$$

Funkcje więcej niż jednego argumentu definiujemy często w tzw. postaci Curry'ego. Np. postacią Curry'ego funkcji $f : A \times B \times C \to B$ jest funkcja

$$f : A \to (B \to (C \to D))$$

gdzie $((f^*(a))(b))(c) = f(a,b,c)$. Funkcje tej postaci są bardzo wygodne w definicjach denotacyjnych. Aby uniknąć nadmiaru nawiasów przy korzystaniu z takich funkcji, przyjmujemy następującą notację kropkową: $f^*.a$ oznacza $f^*(a)$ oraz $f^*.a.b.c$ oznacza $((f^*.a).b).c$. Zwróćmy uwagę, że jest ta notacja podobna do przyjmowanej w wielu językach programowania /np. w Pascalu/ dla rekordów.

Dla polepszenia przejrzystości definicji dziedzin funkcji w postaci Curry'ego opuszczamy w takich definicjach nawiasy o ile ich zagnieżdżenie rośnie od lewej do prawej.
Np. $A \to B \to C \to D$ oznacza $A \to (B \to (C \to D))$. W definicjach takich dziedzin dopuszczamy również używanie operatora $\simeq\!\!\to$ na równi i wraz z operatorem $\to$. Np. możemy zdefiniować dziedzinę $A \to B \simeq\!\!\to C \to D$.

Ostatnia konwencja notacyjna dotyczy tzw. dziedzin syntaktycznych, czyli języków formalnych. Jeżeli A i B są takimi językami, to AB oznaczamy ich konkatenację. Jeżeli $a \in A$ to będziemy również pisali aB zamiast $\{a\}B$ oraz a|B zamiast $\{a\} \mid B$. Ta ostatnia konwencja dotyczy również dziedzin niesyntaktycznych, tzn. dowolnych zbiorów.

5. Prosty przykład definicji denotacyjnej języka programowania

Nasz przykład rozpoczniemy od zdefiniowania trzech dziedzin syntaktycznych uproszczonego języka programowania:

ide(ntyfikatory), wyr(ażenia) oraz ins(trukcje).

ide : Ide = a|b|...

wyr : Wyr = Ide | net Wyr | (Wyr less Wyr)
(Wyr plus Wyr)

ins : Ins = read Ide | print Ide | Ide := Wyr |
Ins; Ins |
if Wyr then Ins else Ins fi |
while Wyr do Ins od

Definicje tych dziedzin są oczywiście indukcyjne. Formalnie mamy tu do czynienia z równaniami stałopunktowymi, ale czytelnik nie znający tej techniki może czytać powyższą definicję tak, jak się czyta definicje napisane w BNF.

W dalszej kolejności definiujemy dziedziny typów danych dostępnych w naszym języku:

b : Bool = {prawda, fałsz} wartości boolowskie

c : Cał = {i | m ⩽ i ⩽ M}. liczby całkowite

Oczywiście liczby całkowite należą do pewnego skończonego przedziału wyznaczonego przez rozmiar słowa w maszynie.

Dla pełnego zdefiniowania mechanizmu typów danych określić należy poza ich dziedzinami, również operacje, jakie na tych danych możemy wykonywać. Określamy trzy takie operacje:

net : Bool → Bool

less : Cał × Cał → Bool | NADMIAR

plus : Cał × Cał → Cał | NADMIAR

Ich definicje są następujące:

net.prawda = fałsz

net.fałsz = prawda

less.

less.(c_1, c_2) =

$c_1-c_2 < m$ lub $c_1-c_2 > M$ → NADMIAR

$c_1-c_2 < 0$ → prawda

TRUE → fałsz

plus.(c_1, c_2) =

$m \leq (c_1+c_2) \leq M$ → c_1+c_2,

TRUE → NADMIAR

W definicjach tych wystąpił niezdefiniowany wcześniej element NADMIAR. Element ten reprezentuje sygnał błędu jaki powinien pojawić się w momencie, gdy zastosowanie operacji less lub plus prowadzi do powstania nadmiaru. Nazywamy go błędem abstrakcyjnym. Posługiwanie się błędami abstrakcyjnymi na poziomie semantyki denotacyjnej pozwala na opisanie mechanizmu wykrywania i sygnalizacji błędów przez implementację języka.

Po opisaniu mechanizmu typów danych przystępujemy do opisania tzw. typów wykonawczych. Definiujemy tu dziedziny i operacje odnoszące się do pojęcia stanu hipotetycznej maszyny widzianej z poziomu języka. W naszym przypadku wprowadzimy trzy dziedziny wykonawcze:

war : Wartość = Bool|Cał

stan : Stan = Ide|IN|OUT → Wartość|Wartość*|PUSTY

błąd : Błąd = NADMIAR|PUSTY|BŁĄD-TYPU|PUSTE-WEJŚCIE

Zdefiniowane powyżej stany są funkcjami opisującymi zawartość pamięci maszyny oraz zawartości urządzeń wejścia i wyjścia. W uzupełnieniu powyższej definicji zakładamy, że IN, OUT ∈ Ide oraz, że każdy stan spełnia następujące trzy warunki, których koniunkcję nazywamy niezmiennikiem stanu:

stan.ide ∈ Wartość|PUSTY

stan.IN ∈ Wartość*

stan.OUT ∈ Wartość*

Intuicyjnie oznacza to, że każdy identyfikator języka jest nazwą komórki pamięci, która albo przechowuje jakąś wartość, albo jest pusta oraz że IN i OUT reprezentują bufory wejścia i wyjścia przechowujące ciągi /pliki/ wartości. Zauważmy, że zarówno plik wejściowy jak i wyjściowy mogą być puste.

Mając zdefiniowane dziedziny wykonawcze języka możemy przystąpić do definiowania funkcji semantyki. Zwykle dla każdej kategorii syntaktycznej języka definiujemy odrębną funkcję semantyki. W naszym przypadku zdefiniujemy dwie takie funkcje:

W : Wyr → Stan → Wartość|Błąd

I : Ins → Stan|Błąd ≃ Stan|Błąd

Funkcje semantyki są funkcjami całkowitymi przypisującymi wyrażeniom i instrukcjom ich denotacje. Denotacje wyrażeń są funkcjami całkowitymi, które biorą jako argument stan i oddają albo wartość albo odpowiedni sygnał błędu. Opisują one mechanizm wyliczania wyrażeń. Denotacje instrukcji są funkcjami częściowymi, które biorą jako argument stan lub sygnał błędu i oddają jako wynik również stan /na ogół zmieniony/ lub sygnał błędu. Opisują one mechanizm wykonywania instrukcji. Przyjmujemy dodatkowo, że ilekroć denotacja instrukcji otrzyma sygnał błędu jako argument, to odda ona ten sam sygnał błędu jako wynik. W ten sposób opisujemy mechanizm propagacji błędu. Fakt, że denotacje instrukcji są funkcjami częściowymi wynika stąd, że nasz język zawiera konstrukcję pętli /instrukcja while/, a więc niektóre wykonania instrukcji mogą być nieskończone nie dając tym samym żadnego wyniku końcowego.

Możemy przystąpić teraz do zdefiniowania funkcji W oraz I. Jak mówiliśmy już o tym wcześniej, funkcje semantyk definiuje się metodą indukcji strukturalnej, tzn. przez przypadki odnoszące się do definicji składni języka

(W.1) Identyfikator jako wyrażenie

W[ide].stan = stan.ide

Wartością wyrażenia ide w aktualnym stanie jest wartość przechowywana pod tym identyfikatorem w tym stanie. Zauważmy, że może to być wartość PUSTY.

(W.2) Zanegowane wyrażenie boolowskie

W[net wyr].stan =
  let war = W[wyr].stan in
  war ∈ Błąd → war,
  war ∉ Bool → BŁĄD-TYPU,
  TRUE → net.war

Obliczając wartość wyrażenia net wyr najpierw obliczamy wartość podwyrażenia wyr. Jeżeli wartość ta jest błędem abstrakcyjnym, to staje się ona wartością całego wyrażenia. W przeciwnym przypadku sprawdzamy, czy jest to wartość boolowska. Jeżeli tak nie jest, to generujemy odpowiedni sygnał błędu. W przeciwnym przypadku wartością całego wyrażenia jest zanegowana wartość podwyrażenia wyr.

W powyższej definicji posłużyliśmy się nieopisaną jeszcze konstrukcją notacyjną let ... in. Jest to jedna z typowych konstrukcji w META-IV oraz w META. Pozwala ona na wprowadzenie oznaczenia lokalnego dla danej definicji. Gdyby nasz metajęzyk był interpretowany maszynowo, konstrukcja ta byłaby deklaracją stałej lokalnej.

(W.3) Wyrażenie boolowskie "less"

W[(wyr$_1$ less wyr$_2$)].stan =
  let war$_1$ = W[wyr$_1$].stan in
  war$_1$ ∈ Błąd → war$_1$,
  war$_1$ ∉ Cał → BŁĄD-TYPU,
  let war$_2$ = W wyr$_2$ .stan in
  war$_2$ ∈ Błąd → war$_2$,
  war$_2$ ∉ Cał → BŁĄD-TYPU,
  TRUE → less.(War$_1$, war$_2$).

Interpretacja intuicyjna tej definicji jest podobna do poprzedniej. Zauważmy, że w obu przypadkach odwołujemy się w definicji semantyki wyrażeń do funkcji zdefiniowanych uprzednio w dziedzinach danych. Przyjęliśmy przytym mnemotechniczną odpowiedniość pomiędzy symbolami języka not, less i plus oraz odpowiadającymi im metasymbolami not, less i plus.

(W.4) Wyrażenie arytmetyczne "plus"

$$W[(wyr_1 \text{ plus } wyr_2)].sta =$$
$$\ldots (\text{jak w (W.3)})$$
$$TRUE \rightarrow plus.(war_1, war_2)$$

Przechodzimy teraz do zdefiniowania semantyki instrukcji. Przypominamy, że w myśl przyjętego wcześniej założenia, dla każdej instrukcji $ins \in Ins$ oraz każdego błędu abstrakcyjnego $błąd \in Błąd$ zachodzi

$$I[ins].błąd = błąd$$

tzn., że każda instrukcja przekazuje odziedziczony od innej instrukcji sygnał błędu bez zmiany. Intuicyjnie oznacza to, że mechanizm obsługi błędów ograniczony jest w naszym języku do informowania o każdym błędzie użytkownika. Oczywiście na gruncie metody denotacyjnej opisać można również bardziej zaawansowane mechanizmy tego typu.

Podobnie jak funkcja semantyki dla wyrażeń również funkcja semantyki dla instrukcji zdefiniowana jest metodą indukcji strukturalnej.

(I.1) Konstrukcja wejścia

$$I[\underline{read}\ ide].sta =$$
$$\underline{let}\ (war_1, \ldots, war_n) = sta.IN\ \underline{in}$$
$$n > 0 \rightarrow sta[war_1/ide, (war_2, \ldots, war_n)/IN]$$
$$TRUE \rightarrow PUSTE\text{-}WEJŚCIE$$

Najpierw sprawdzamy, czy plik wejściowy jest niepusty. Jeżeli tak jest, to pierwszy element z pliku wejściowego przypisujemy identyfikatorowi ide, a następnie usuwamy ten element z

pliku. W przeciwnym przypadku generujemy odpowiedni sygnał błędu.

(I.2) Instrukcja wyjścia

$I[\underline{\text{print}}\ wyr].sta =$

$\underline{\text{let}}\ war = W[wyr].sta\ \underline{\text{in}}$

$war \in$ Błąd $\rightarrow war,$

$\underline{\text{let}}(war_1,\ldots,war_n) = sta.OUT\ \underline{\text{in}}$

$\text{TRUE} \rightarrow sta[(war_1,\ldots,war_n,war)/OUT]$

Najpierw obliczamy wartość wyrażenia wyr. Jeżeli ta wartość jest sygnałem błędu, to pozwalamy temu sygnałowi prepagować dalej. W przeciwnym przypadku umieszczamy tę wartość na końcu pliku wyjściowego.

(I.3) Instrukcja przypisania

$I[ide := wyr].sta =$

$\underline{\text{let}}\ war = W[war].sta\ \underline{\text{in}}$

$war \in$ Błąd $\rightarrow war,$

$\text{TRUE} \rightarrow sta[war/ide]$

Wyliczamy wartość wyrażenia wyr i jeżeli jest to sygnał błędu to pozwalamy mu propagować. W przeciwnym przypadku przypisujemy tę wartość identyfikatorowi ide.

(I.4) Sekwencyjne złożenie instrukcji

$I[ins_1 : ins_2].sta =$

$I[ins_2].(I[ins_1].sta)$

Najpierw wykonujemy ins_1 a potem ins_2.

(I.5) Instrukcja warunkowa

$I[\underline{\text{if}}\ wyr\ \underline{\text{then}}\ ins_1\ \underline{\text{else}}\ ins_2\ \underline{\text{fi}}].sta =$

$\underline{\text{let}}\ war = W[wyr].sta\ \underline{\text{in}}$

$war \in$ Błąd $\rightarrow war,$

$war \notin$ Bool $\rightarrow$ BŁĄD-TYPU,

$war = \text{prawda} \rightarrow T[ins_1].sta,$

$\text{TRUE} \rightarrow I[ins_2].sta$

Najpierw obliczamy wartość wyrażenia wyr i jeżeli prowadzi to do sygnału błędu, to pozwalamy mu propagować dalej. W przeciwnym przypadku sprawdzamy, czy jest to wartość boolowska. Jeżeli tak nie jest, to generujemy odpowiedni sygnał błędu. Jeżeli tak jest, to wykonujemy odpowiednią gałąź naszej instrukcji warunkowej.

(I.6) Instrukcja pętli

I[while wyr do ins od].sta =
  let war = W[wyr].sta in
  war ∈ Błąd →war,
  war ∉ Bool →BŁĄD-TYPU,
  war = prawda →I[while wyr do ins od].(I[ins].sta),
  TRUE →sta

Najpierw obliczamy wartość wyrażenia wyr i jeżeli jest to sygnał błędu, to pozwalamy mu propagować dalej. W przeciwnym przypadku sprawdzamy, czy jest to wartość boolowska i jeżeli tak nie jest, to generujemy sygnał błędu. Jeżeli jest to wartość boolowska, to mamy jeszcze dwie możliwości. Jeżeli wartością tą jest "prawda", to wykonujemy instrukcję ins będącą ciałem pętli, a następnie przesyłamy stan wynikowy ponownie do instrukcji pętli. Jeżeli wartością tą jest "fałsz", to stan wejściowy instrukcji pętli przesyłamy dalej niezmieniony. Zauważmy, że przypadek war = prawda prowadzi pozornie do błędnego koła w definicji. W rzeczywistości jednak definicja nasza jest poprawna, co wynika z teorii punktu stałego w zbiorach częściowo uporządkowanych. Oczywiście rozmiar niniejszego opracowania nie pozwala na pobieżne chociażby wyjaśnienie podstaw tej teorii. Zainteresowany czytelnik znajdzie odnośniki do odpowiedniej literatury w rozdz. 7.

6. Kilka słów o zastosowaniach metody denotacyjnej

Jak już wspomniano w rozdz. 2 denotacyjna metoda definiowania i projektowania software'u stosowana jest w projektach

przemysłowych od kilku już lat, przy czym zarówno zakres jej zastosowań, jak i liczba użytkowników stale rośnie. Dotyczy to szczególnie jej przemysłowej wersji VDM. Wśród ciekawszych zastosowań tej metody /dane z roku 1982/ przytoczyć można definicje formalne następujących języków i systemów:

Basic	Prolog
Fortran	Ada
Algol 60	CHILL
Pascal	PL/1
Pascal Plus	System/R
Modula 2	CODASYL/DBTG

Niektóre z tych definicji - np. dla Ady i CHILL - stały się punktem wyjścia do zbudowania już istniejących kompilatorów. Inne - np. dla System/R /IBM-owski system baz danych/ - pozwoliły na wykrycie i usunięcie wielu błędów w istniejącej implementacji.

Definicje języków i systemów tworzone metodą VDM reprezentują oczywiście dosyć spore obiekty tekstowe. Dla zorientowania czytelnika o ich rozmiarach podajemy poniższą tabelkę zawierającą kilka typowych przykładów /4/:

język	liczba linii definicji napisanej w META-IV
Algol-60	1 000
Pascal	2 500
PL/1	4 000
Pascal Plus	4 500
Ada	14 000

Oczywiście przy tworzeniu takich tekstów i posługiwaniu się nimi na szeroką skalę konieczne są komputerowe systemy wspomagające. W kilku ośrodkach na świecie, m.in. na Uniwersytecie w Manchester i na Politechnice w Kopenhadze prowadzone są już prace zmierzające do zbudowania takich systemów. Mają to być bogate środowiska komputerowe wspomagające pracę projektanta, analityka i programisty systemowego. Przewiduje się, że będą one miały na-

stępującą strukturę:

1. Edytor źródłowy

   1.1. Sterowany składnią edytor metajęzyka, np. META-IV

   1.2. Analizator systaktycznej pełności i niesprzeczności definicji

   1.3. Analizator poprawności definicji ze względu na użyte w niej meta-typy meta-obiektów

2. Meta-interpreter

   2.1. Interpreter metajęzyka pozwalający na eksperymentalną eksploatację zdefiniowanego systemu

   2.2. Zestaw narzędzi pozwalający na przeprowadzenie dynamicznej analizy systemu, tzn. analizy przepływu danych, testowania itp.

3. Procesor transformacji

   3.1. Interaktywny system wspomagający transformowanie definicji źródłowych w metajęzyku na definicje innego poziomu, ale też w metajęzyku, np. transformowanie definicji języka na definicję kompilatora tegoż języka

   3.2. Translatory z metajęzyka na poziom języków programowania

4. Weryfikator poprawności

   4.1. Systemy wspomagające dowodzenie poprawności projektowanego software'u ze względu na różne kryteria

   4.2. Systemy wspomagające dowodzenie zgodności pomiędzy kolejnymi etapami projektu.

Brak, w obecnej chwili, systemów komputerowego wspomagania metody denotacyjnej nie powinien prowadzić do biernego oczekiwania na takie systemy. Aby stosować metodę denotacyjną w przyszłości należy zacząć uczyć się jej już dziś. Na metodę tę skła-

dają się bowiem - poza metajęzykiem - określona teoria matematyczna, metodologia pracy, a nawet filozofia. Opanowanie teorii, przyzwyczajenie się do nowej metodologii, zaakceptowanie nowej filozofii, wymaga czasu i nakładu pracy. Wymaga śledzenia literatury przedmiotu oraz jednoczesnego ręcznego eksperymentowania z metodą. Poświęcony na to czas nie będzie z pewnością stracony. Zauważmy przecież, że wszystkie dotychczasowe zastosowania tej metody - również kompilatory dla języków Ada i CHILL - wykonane zostały całkowicie bez pomocy systemów wspomagania.

7. Uwagi bibliograficzne

Czytelnik, którego niniejszy artykuł zdołał zainteresować metodą denotacyjną zechce zapewne zapytać o dostępną literaturę przedmiotu. Niestety, odpowiedź na to pytanie nie jest zbyt zachęcająca. W literaturze światowej, a tym bardziej krajowej, brak jest dobrego podręcznika zastosowań metody denotacyjnej. Na pocieszenie i zachętę można dodać, że co najmniej dwa takie podręczniki są w trakcie powstawania. Ukażą się one jednak nie wcześniej niż za dwa lata. Na razie więc należy ograniczyć się do tego, co jest dostępne.

Zacznijmy od literatury w języku polskim. Podstawową pozycją jest tu książka M. Gordena /9/ /tłumaczenie z j. angielskiego/ wyjaśniająca zasady stosowania semantyki denotacyjnej. Zaletą tej książki jest jej wielka jasność i przystępność. Wadą - że przedstawia ona tzw. model kontynuacyjny, który ze względu na swoją złożoność nigdy nie był stosowany w praktyce. Warto może jednak wspomnieć, że grupa studentów Uniwersytetu Warszawskiego po wysłuchaniu jednosemestralnego wykładu na podstawie właśnie tej książki napisała z własnej inicjatywy pełną definicję denotacyjną dla Pascala. Do polskiego wydania książki Gordena dołączony jest dodatek napisany przez autora niniejszego artykułu, a poświęcony podstawowej dla semantyki denotacyjnej teorii zbiorów częściowo uporządkowanych i dziedzin Scotta.

Czytelnika, który chciałby uzyskać szersze spojrzenie na metodę denotacyjną w kontekście innych modeli dla matematycznych semantyk języków programowania oraz innych formalnych metod inżynierii oprogramowania odesłać można do książki P. Dembińskiego i J. Małuszyńskiego /7/. Osoby zainteresowane formalnymi metodami konstruowania i analizy poprawnościowej programów użytkowych polecić można przekład książki C.Jonesa /11/ która ukaże się już niebawem nakładem Wydawnictw Naukowo-Technicznych w serii Biblioteka Inżynierii Oprogramowania.Czytelnikom, którzy czują potrzebę podniesienia swoich kwalifikacji ogólno-matematycznych w zakresie przydatnym do lepszego zrozumienia metody denotacyjnej polecić można książkę H. Rasiowej /13/.

W literaturze obcojęzycznej /tzn. anglosaskiej/ najlepszą pozycją przedstawiającą teorię metody denotacyjnej jest książka J. Stoy'a /14/. Nie jest to jednak, z pewnością, książka dla praktyków. Przedstawia ona, podobnie jak książka M. Gordona, model kontynuacyjny, a ponadto posługuje się bardzo niepraktyczną notacją. Jest to książka napisana wyraźnie dla teoretyków. Niemniej jednak z niej właśnie oraz z jeszcze trudniejszych tekstów korzystali twórcy VDM budując swoją metodę i próbując jej pierwszych zastosowań.

W zakresie VDM najpełniejszym dziś i najaktualniejszym tekstem jest zbiór rapertów technicznych /1/ wydany i częściowo napisany przez dwóch współtwórców i pionierów tej metody:D. Bjørnera i C. Jones'a. Niestety, jest on w kraju trudno dostępny. Tekstem mniej aktualnym i mniej czytelnym, ale dostępniejszym, jest wydany przez tych samych autorów zbiór /2/. Matematyczne podstawy VDM - będące w zasadzie skrótem matematycznych podstaw dla semantyki denotacyjnej - przedstawione są w artykule J. Stoy'a /15/. Ukazał się on w materiałach konferencji poświęconej podstawom sementyki denotacyjnej, które w całości polecić można jako dobrą uzupełniającą literaturę przedmiotu . Aspekty inżynierskie VDM opisano w pracy /4/ D. Bjørnera i S. Prehna, natomiast najobszerniejszy przykład zastosowań - deno-

tacyjna semantyka dla języka ADA - przedstawiona jest w zbiorze wydanym przez D. Bjórnera i O. Oesta.

Klasyczna semantyka denotacyjna - w tym również i VDM - oparta jest na dosyć trudnym matematycznie modelu tzw. dziedzin zwrotnych stworzonym przez D. Scotta. Ostatnio A. Blikle i A. Tarlecki /6/ zaproponowali zastąpienie tego modelu przez prostszy oparty na elementarnej teorii mnogości. Wstępna propozycja modyfikacji języka META-IV przystosowana do tego nowego modelu opisana została w pracy /5/.

Bibliografia

/1/ Bjórner, D., Jones, C.B. /eds./ The Vienna Development Method: The Meta Language, Lecture Notes in CS, vol. 61 Springer - Verlag 1978

/2/ Bjórner, D. Jones, C.B. Formal Specification and Software Development, Prentice-Hall International 1982

/3/ Bjórner, D,. Oest, O.N. /eds/. Towards a Formal Description of ADA, LNCS 98, Springer-Verlag 1980

/4/ Bjorner, D., Prehn, S. Software Engineering Aspects of VDM: the Vienna Development Method. in: Theory and Practice of Software Technology /D. Ferrari, M. Bolognani, J. Goguen , eds/, North-Holland 1983

/5/ Blikle, A. A metalanguage for naive denotational semantics, Progetto Finalizzato Informatica, C.N.R. Progetto P. Onet 104, Piza 1983

/6/ Blikle, A., Tarlecki, A. Naive Denotational Semantics, in: Information Processing 93 /R.E.A. Manson ed./, Proc. IFIP 1983, North-Holland

/7/ Dembiński, P., Małuszyński, J., Matematyczne metody definiowania języków programowania, WNT, Warszawa 1981

/8/ Floyd, R.W., Assingning Meanings to Programs, in: Mathematical Aspects of Computer Science /J.T. Schwartz, ed./, Proc. of a Symp. in Appl. Math., American Mathematical Society Providence 1967

/9/ Gordon, M.J.C., Denotacyjny opis języków programowania, WNT, Warszawa 1983

/10/ Herzog, O. Formal Software Development Methods in Industrial Environments, in: Formal Software Development: Combining Specification Methods, proc. workshop, Nyborg 1984

/11/ Jones, D. Software Development, Rigorous Approach, Prentice Hall, Englewood Cliffs 1979

/12/ McKeeman, W.M., The Role of Software Engineering in the Microcomputer Revolution: An Overview, Proc. 4th Int.Conf. on Soft. Eng., Munich 1979, IEEE Cat. no 79CH1479-5C

/13/ Rasiowa, H., Wstęp do matematyki współczesnej, PWN, Warszawa 1973

/14/ Stoy, J.E., Denotational Semantics: The Scott-Strachey Approach to Programming Leaguage Theory, The MIT Press, Cambridge Mass. 1977

/15/ Stoy, J.E., Foundations of Denotational Semantics, in: Abstract Software Specification, 1979 Copenhagen Winter School Proc. /D. Bjorner, ed./, Lacture Notes in CS, vol.86, Springer-Verlag 1980

/16/ Turing, A.M. On Checking a Large Routine, Report of a Conference on High Speed Automatic Calculating Machines, pp. 67-69, University Mathematical Laboratory, Cambridge 1949

Wiesław Flakiewicz[1/]

SYSTEMY INFORMATYCZNE ZARZĄDZANIA
KIERUNKI I BARIERY ROZWOJU

1. Założenia wyjściowe

U podstaw rozważań stanowiących treść referatu, leżą następujące założenia początkowe:

a. Analiza sytuacji bieżącej i główne wnioski, mogą być odnoszone w zasadzie do każdej organizacji gospodarczej, bez względu na jej usytuowanie w strukturze organizacyjnej gospodarki narodowej. Tym niemniej główny nurt rozważań kierowany jest na szczebel przedsiębiorstw /jednostek równorzędnych/, gdyż w pozostałych przypadkach, tzn. gdy podmiotem rozważań stają się urzędy i instytucje /tj. organizacje gospodarcze nie będące przedsiębiorstwami/, należy uwzględnić dodatkowe elementy istotne, które musiałyby zwiększyć znacznie złożoność wywodów /m.in. chodzi tu o konieczność uwzględnienia zróżnicowanych relacji wiążących urzędy i instytucje między sobą oraz z przedsiębiorstwami, co komplikuje obraz i wymaga wprowadzenia pewnych klas przypadków/.

b. Zakłada się, że dana organizacja gospodarcza posiada /rozwija/ informatyczne systemy dziedzinowe /agendowe/, względnie może je zbudować i eksploatować dla swoich własnych celów, albo w oparciu o własny potencjał informatyczny, albo poprzez zlecenie wdrożenia takich systemów wyspecjalizowanym ośrodkom ob-cym /np. poprzez sieć ZETO/. Założenie to jest równoznaczne stwierdzeniu, że krajowa służba informatyczna dysponuje wystarczającym doświadczeniem praktycznym i umiejętnościami fachowymi, aby systemy takie zaprojektować, oprogramować i wdrożyć do eksploatacji przy uwzględnieniu potrzeb użytkownika i w dostosowaniu do posiadanych przez niego możliwości techniczno-technologiczno-organizacyjnych.

1/ Szkoła Główna Planowania i Statystyki w Warszawie.

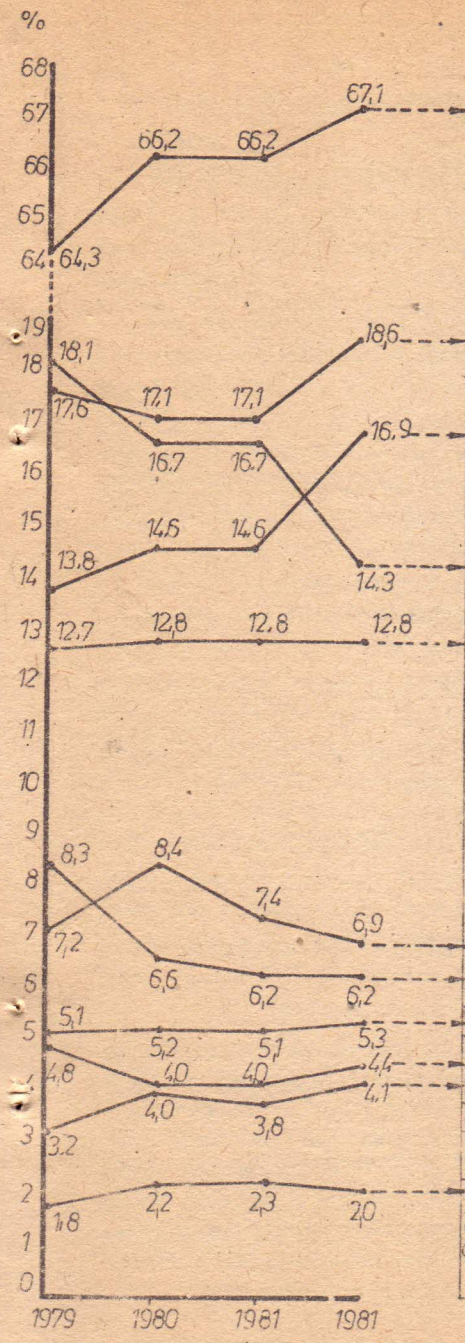
c. Mówiąc o systemach agendowych /dziedzinowych/ mamy na myśli informatyczne systemy budowane dla potrzeb zarządzania, które charakteryzują się następującymi, głównymi własnościami:

- wejścia informacyjne są powtarzalne, programowane i zrutynizowane co do treści, formy oraz periodyzacji pojawiania się; to samo dotyczy wyjść użytkowych,
- strumień danych wejściowych ma charakter masowy,
- generowane informacje wyjściowe mają charakter ewidencyjno-powiadamiająco-kontrolny,
- transformacja wejść na wyjściu oparta jest o "płytkie" algorytmy przetworzeń /cztery działania, procent, średnie, proste relacje logiczne/,
- systemy te są budowane niezależnie od siebie, wykazują wzajemnie duży stopień semantycznej redundancji, orientowane są na strukturę organizacyjną użytkownika,
- systemy te obsługują co najwyżej tzw. decyzje standardowe /wg nomenklatury Kornai'a/ tzn. decyzje związane z codziennym tokiem zajęć użytkownika, a więc o charakterze zrutynizowanym, powtarzalnym, dotyczącym alokacji zasobów, sterowania bieżącego oraz działań interwencyjno-kontrolnych.

d. Dziedzinowe systemy informatyczne stanowią obecnie zdecydowaną większość. Mogą o tym świadczyć pewne wskaźniki, które uwidocznione są w tablicy 1 "Tematyka systemów informatycznych zarządzania /SIZ/ w latach 1979 - 1982. Tablica ta stanowi syntetyczny obraz ilościowy sytuacji bieżącej traktowanej jako moment wyjściowy analizy.[1/]

1/ Podstawą opracowania tablicy jest publikacja GUS-OBRI SPIS "Informatyka i ośrodki informatyki w 1982r. W-wa. Czerwiec 1983 r. Oznaczenia i symbole występujące w tablicy są następujące:

"+" - przyrost %-wy udziału danej tematyki w 1982r. w stosunku do 1979 r.
"-" - spadek %-wy udziału danej tematyki w 1982 r. w stosunku do 1979 r.
"$\bar{x}$" - przeciętny udział /w %/ za lata 1979 - 1982
"s" - odchylenie standardowe /w %/ za lata 1979 - 1982
"v" - współczynnik zmienności /w %/ za lata 1979 - 1982
"x" - tematyka wchodząca w skład ogólnej pozycji "zarządzanie"



TEMATYKA	+	−	$\bar{X}$	σ	V
Zarządzanie	2,8		65,9	1,02	1,5
Autom. prac zawodow.	1,0		17,6	0,61	3,4
Rozliczenia finansowe *	3,1		14,9	1,16	7,7
Autom. proc. techno l.		3,8	16,41	1,37	8,3
Gosp. mate-riałowa *	0,1		12,8	0	0
Stat. i analiza ekonom. *		0,3	7,5	0,56	7,5
Przygot. plan. i kontrola produ-kcji *		2,1	6,8	0,86	12,7
Gosp. wyrobami *	0,2		5,2	0,08	1,5
Kadry zatrud. i płace *		0,4	4,3	0,10	2,3
Syst. wielodzie-dzinowe *	0,9		3,7	0,36	9,9
Gosp. śr. trwa-łymi i narzę-dziami	0,2		2,1	0,2	9,5

PARAMETRY OKRESU 1971-1982 (w %)

TABLICA 1 TEMATYKA SIZ W LATACH 1979 – 1982

Przytoczone w tablicy wskaźniki i proporcje dają podstawę do następujących wniosków:

a/ Zastosowania komputerów w Polsce skierowane są na trzy główne dziedziny:

- zarządzanie /65,9 % udziału/
- automatyzacja prac zawodowych /17,6 %/
- automatyzację procesów technologicznych /16,5 %/.

b/ W ramach zarządzania najwyższy udział wykazują:

- rozliczenia finansowe /14,9 %/
- gospodarka materiałowa /12,8 %/

co daje łącznie udział tych dwóch grup tematycznych w granicach 28 %. Jeśli jednak udział ten odnieść do zarządzania traktowanego za 100, udział ten wzrasta do 44 %.

c/ Zwraca uwagę bardzo niski udział tematyki związanej z gospodarką środkami trwałymi /2,1 %/ oraz statystyki i analiz ekonomicznych /7,5 %/

d/ W latach 1979 - 1982 tematyczną strukturę SIZ można uznać za stabilną. Świadczy o tym niska wartość /nie przekraczająca 12 %/ współczynnika zmienności mierzącego stosunek odchylenia standardowego od średniej arytmetycznej.

e/ W pewnych grupach tematycznych można zaobserwować pewne tendencje dotyczące:

- wzrostu udziału rozliczeń finansowych i problemów kadrowo-płacowych, co jest odbiciem potrzeb informatycznych związanych z nową regulacją prawną tej problematyki w ramach reformy gospodarczej,
- spadku udziału planowania i kontroli procesów produkcji, co może wiązać się z niestabilną sytuacją produkcyjną związaną ze zjawiskami kryzysowymi,
- obniżania się i tak małego dotąd zainteresowania się problematyką analiz gospodarczych, które przez wiele lat miały charakter sprawozdawczy i były narzędziem manipulacja dla uzasadnienia z góry założonych celów /wykazania albo sukcesów albo uzasadnienia tzw. "obiektywnych trudności"/,

- braku większego zainteresowania problematyką majątku trwałego.

Podkreślmy wreszcie, że struktura tematyczna SIZ odpowiada w pełni koncepcji systemów agendowych /dziedzinowych/, stanowiących główną bazę informacyjną w scentralizowanym systemie zarządzania.

Na tle tak zarysowanej sytuacji wydaje się zasadne zwrócenie uwagi na następujący problem. Jeśli założyć obecny status quo na przyszłość, informatyka dla celów zarządzania nie wyjdzie poza zaklęty rewir systemów dziedzinowych. Nie negując potrzeby ich istnienia, rodzi się jednak pytanie czy tematyka, sposób budowy i wykorzystania tych systemów wystarczyć może dla obsługi tych potrzeb informacyjnych, które są immanentnie związane z wdrażaniem i rozwijaniem reformy gospodarczej, zwłaszcza, że już od pewnego czasu agendowe systemy są krytykowane zarówno od strony ich struktury informacyjnej jak i przestrzennej, oraz techniczno-technologicznej. Krótko mówiąc rodzi się problem kierunków rozwoju krajowej informatyki dla potrzeb zarządzania. W referacie chcemy zająć się wycinkiem tej problematyki a mianowicie rozwojem struktury informacyjnej tych systemów. Starać się więc będziemy odpowiedzieć na lapidarne pytania: co? po co? dlaczego? i dla kogo? przetwarzać dane i informacje, aby wyjść naprzeciw unowocześnieniu zarządzania organizacjami gospodarczymi zgodnie z duchem reformy.

2. Kilka uwag metodologicznych

1. W praktyce informatycznej przyjęło się traktować jako synonimy dwa pojęcia: system informatyczny zarządzania /SIZ/ oraz zbiór istniejących /i/lub rozwijanych/ systemów dziedzinowych /agendowych/ na rzecz określonego użytkownika. Ta synonimia jest nieprawdziwa, błędna nawet wtedy, gdyby systemy dziedzinowe /rozumiane tak jak ma to miejsce w pkt I referatu/ pokryły swym zasięgiem 100 % aktualnych potrzeb informacyjnych użytkownika we wszystkich obszarach agendowych. Systemy dzie-

dzinowe stanowią sobą część i tylko część informatycznego systemu zarządzania, i choć część ta jest istotną, to jednak nie wyczerpuje zadań i funkcji SIZ, które zasadniczo przekraczają tematykę typową dla ujęcia agendowego.

2. Dla wyjaśnienia zaprezentowanego w referacie poglądu czym jest SIZ przyjmiemy następujące przesłanki:

- Wyróżnimy, jak to już uczyniliśmy wcześniej, dwa rodzaje organizacji gospodarczych: przedsiębiorstwa /i jednostki im równorzędne/ oraz urzędy i instytucje,
- W przedsiębiorstwach wyróżnimy dwie główne sfery działalności: procesów realnych /produkcyjnych, usługowych, obrotu, budownictwa, transportu itp. /oraz sferę regulacji /zarząd/. W instytucjach i urzędach wyróżnimy tylko sferę regulacji,
- W obu typach organizacji gospodarczych wyróżnimy ponadto trzy szczeble zarządzania: operacyjny, taktyczny i strategiczny,
- W/w trzy szczeble zarządzania zdefiniujemy w sposób następujący:
  - Każde świadome oraz racjonalne działanie człowieka powinno uwzględnić trzy czynniki składowe: stan początkowy od którego działanie się zaczyna, cel który chce się osiągnąć oraz drogę dojścia do celu. Implicite zakłada się tu wpływ czwartego czynnika - czasu /interwału czasu między momentem rozpoczęcia działania a planowanym momentem realizacji celu/, który powoduje, że im interwał ten staje się dłuższy, tym zachodzi konieczność etapowania realizacji celu - czyli wyznaczania celów pośrednich,
  - Zarządzanie operacyjne zachodzi wtedy, jeśli zdetermino - wane zostały charakterystyki stanu początkowego, cel oraz droga dojścia, zaś zadaniem tak pojętego zarządzania, jest likwidacja, minimalizacja lub blokada tych zjawisk, które odchylają tor realizacji zadania od toru wytyczonego, poza pasmo dopuszczalnych odchyleń,
  - Zarządzanie taktyczne zachodzi wtedy, gdy zdeterminowane zostały charakterystyki stanu początkowego, cel główny oraz kryteria wyboru torów działań, zaś zadaniem zarządzania

staje się wybór drogi /dróg/ dojścia do celu głównego i celów cząstkowych, z możliwością korekt tych ostatnich,

- Zarządzanie strategiczne zachodzi wtedy, gdy zdeterminowane zostały charakterystyki stanu początkowego, zaś zadaniem zarządzania jest wyznaczenie celu głównego, kryteriów wyznaczania celów cząstkowych oraz określenie możliwych dróg realizacji celu głównego.

3. Przyjmując powyższe założenia, systemy informatyczne funkcjonujące w danej organizacji gospodarczej /a w szczególności w przedsiębiorstwie, mogą być sklasyfikowane w ramach pięciu głównych klas /vide rys. 1/.

Klasa I - Systemy informatyczne, sterujące bezpośrednio procesami techniczno-technologicznymi w ramach sfery procesów realnych /np. obrabiarki sterowane numerycznie, komputerowe sterowanie linią produkcyjną, robotyzacja procesów technologicznych itp/. Systemy te instalowane są wyłącznie na szczeblu operacyjnym /tzw. poziom "ruchu"/

Klasa II - Dziedzinowe /agendowe/ systemy informatyczne. Pobierają one informacje wejściowe ze sfery realnej, obsługują głównie operacyjny szczebel zarządzania, dostarczając jednocześnie /wybiórczo/ pewnych syntetycznych informacji dla szczebla taktycznego oraz - w miarę konieczności - również i dla szczebla strategicznego. Bezpośrednie oddziaływanie tych systemów maleje w miarę przechodzenia na wyższe poziomy organizacji zarządzania.
Lokalizacja: głównie komórki liniowe, oraz częściowo w komórkach sztabowych /np. koszty, księgowość, płace itp./

Klasa III - Stanowi ją wieloelementowy zbiór systemów informatycznych o wyspecjalizowanych funkcjach orientowanych głównie na: kompleksowe i odcinkowe /problemowe/ analizy techniczno-ekonomiczne o charakterze przyczynowo-skutkowym, odcinkowe i kompleksowe planowanie krótko i średnio-okresowe /w horyzoncie do 2 lat/, badania tendencji i trendów w otoczeniu organizacji gospodarczej, wspomaganie decyzji dotyczących tworzenia warunków prawidłowej realizacji zadań rocznych, włączając w to korekty i zmiany, bieżącą ocenę stanu

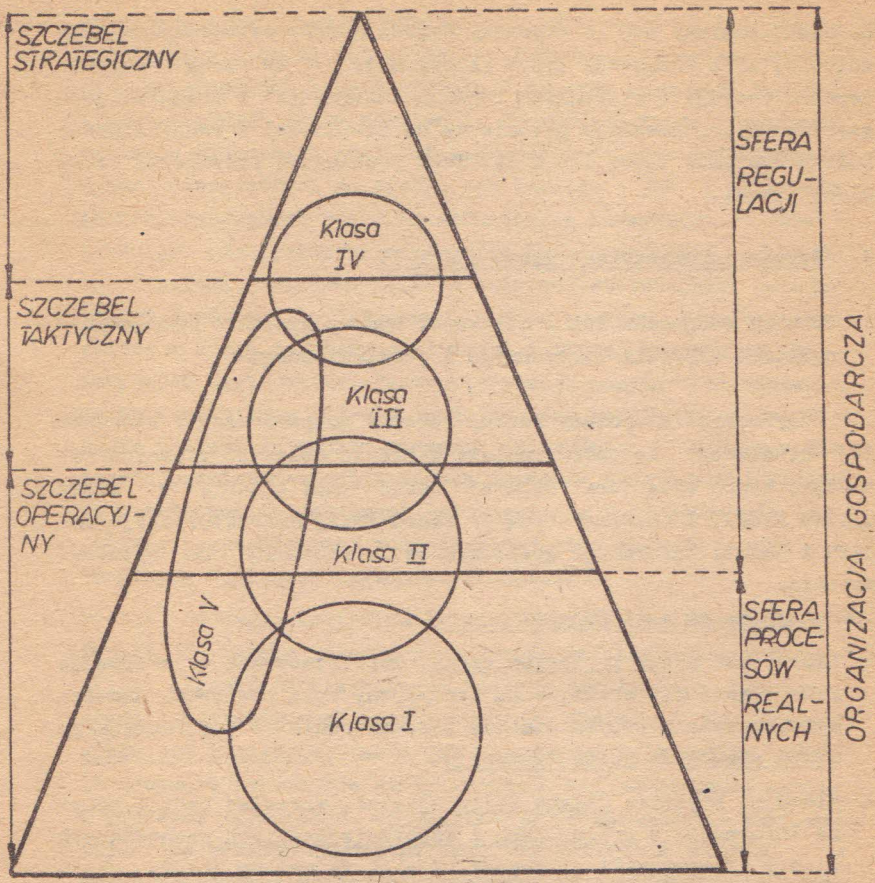
realizacji procesów realnych /w formie syntetycznych wskaźników i proporcji/ oraz przygotowywanie raportów i informacji dla szczebla strategicznego w zakresie realizacji zadań taktycznych.
Lokalizacja: główne komórki sztabowe /ze szczególnym uwzględnieniem komórek analitycznych, ekonomicznych i logistycznych/.

Klasa IV - Stanowi ją wieloelementowy zbiór systemów informatycznych generujących warianty decyzji dotyczących wyboru określonej polityki gospodarczej dla całej organizacji gospodarczej lub jej istotnych fragmentów /techniki, technologia, inwestycje, finanse, efektywność w aspekcie rozwoju w czasie/, ze szczególnym uwzględnieniem prognozowania, programowania i wyznaczania celów /planowania/ działalności w zasadzie w horyzoncie czasu ponad 2 lata, jak też oceny i kontroli działalności organizacji traktowanej in toto.
Lokalizacja: głównie na szczeblu taktycznym, przygotowującym warianty strategii; obsługa ścisłego kierownictwa oraz niektórych narad i konferencji dyrekcji i sztabów.

Klasa V - Stanowią ją systemy wyszukiwawcze /biblioteczne/, spełniające funkcje obsługi wszystkich poziomów zarządzania, dostarczające informacji naukowo-technicznej i ekonomicznej, odpowiedzialne za tworzenie wspólnej i spójnej bazy normatywnej. Mogą być tworzone jako systemy satelitarne np. w stosunku do systemów SINTE, o różnym powiązaniu techniczno-organizacyjnym z otoczeniem.

4. System informatyczny zarządzania /SIZ/ stanowi sumę logiczną klas II - V /SIZ sensu largo/ lub klas II - IV /SIZ sensu stricto/.

5. Tak przedstawiona koncepcja ma charakter docelowy tj. przy uwzględnieniu długiego horyzontu czasu.



Rys. 1 Lokalizacja czterech klas systemów informatycznych wg sfer procesów i szczebli zarządzania

6. Jeśli pominąć typ II /systemy dziedzinowe/, co założyliśmy na wstępie, oraz /czasowo/ typ V /który może być tworzony po wypracowaniu decyzji o kierunkach rozwoju informatyki w organizacjach gospodarczych/ powstaje pytanie co do zasadności i racjonalności budowy systemów klasy III i IV. Temu problemowi poświęcimy teraz więcej uwagi.

3. Systemy informatyczne klasy III i IV

A. Problem racjonalności i celowości budowy systemów informatycznych dla szczebli taktycznego i strategicznego

Zakładając stopniowy rozwój reformy gospodarczej w kierunku jej docelowego kształtu, należy wymienić dziewięć głównych tendencji, które będą coraz silniej dawać znać o sobie w miarę upływu czasu, i zmierzać będą do zasadniczej restrukturyzacji popytu i podaży informacji gospodarczej, w stosunku do sytuacji obecnej.
Tendencje te są następujące:

1/ Biorąc pod uwagę kryterium czasu, zapotrzebowane informacje będą stawać się wrażliwe na dynamiczny opis procesów. Stąd restrukturyzacja popytu oznacza przechodzenie z informacyjnego opisu stanów do opisu tendencji.

2/ Stosując kryteria języka opisu zjawisk, wzrastać będzie popyt na informacje o proporcjach i współzależnościach zachodzących między zjawiskami i procesami. Wymaga to zmiany języka opisu z dominującego jeszcze powszechnie języka obiektowego /opis "rzeczy"/ Na metajęzyki i generowane przez nie metainformacje charakteryzujące relacje między "rzeczami" i /lub ich atrybutami. Klasycznym przykładem może tu być rozpowszechnianie się rachunku ekonomicznego, jako podstawy decyzji gospodarczych, które nie mogą być należycie przygotowane informacyjnie przy użyciu wyłącznie języka obiektowego.

3/ Przy założeniu kryteriów organizacyjnych, restrukturyzacja popytu informacyjnego oznaczać będzie przejście z orientacji na

strukturę organizacyjną użytkownika informacji na orientację problemową. Oznacza to, że zbiory informacji tworzone dotychczas programowane były w oparciu o zakresy czynności komórek organizacyjnych sfery regulacji. Prowadziło to do tworzenia systemów agendowych /tak informacyjnych jak i informatycznych/, niezależnych od siebie tak logicznie jak i fizycznie, charakteryzujących się dużą redundancją informacji /nawet treściowo identycznych/, niespójną bazą normatywną /słowniki, katalogi, tezaurusy/, a co może najistotniejsze, pomijających problematykę "stykową", a więc procesy, zjawiska itp. leżące w kręgu zainteresowań dwóch i więcej komórek organizacyjnych użytkownika. Restrukturyzacja popytu orientowana na opis problemów /ujęcie funkcjonalne/ będzie oznaczała poddanie rewizji /z punktu widzenia celowości/ utrzymywania często nadmiernie rozbudowanych systemów agendowych i wzrost zainteresowania problemami istotnymi dla funkcjonowania organizacji gospodarczej rozpatrywanej albo in toto albo w przekroju węzłowych zagadnień /funkcji/.

4/ Biorąc pod uwagę kryterium "algorytmizacji przetwarzania informacji", należy założyć przesuwanie zainteresowania z "płytkich" algorytmów klasycznych /cztery działania, procent, średnia arytmetyczna, proste operacje logiczne/ na złożone algorytmy klasyczne, takie jak np. włączenie w orbitę zainteresowania parametrów statystycznych czy modelowania zjawisk i procesów /symulacja, optymalizacja, programowanie sieciowe itp./. Te właśnie algorytmy będą "wzrastały w cenie" w miarę właściwego rozumienia rachunku ekonomicznego, nie tylko jako elementu kalkulacji, ale i stwarzania podstaw do kreowania właściwej polityki gospodarczej w różnych horyzontach czasu i o różnym zakresie tematycznym.

5/ Jeśli za kryterium przyjąć charakter strumieni informacji, restrukturyzacja popytu oznaczać będzie reorientację czyli przechodzenie z bazy informacyjnej o charakterze ewidencyjno-powiadamiająco-kontrolnej, na strumienie informacji analitycznych /przyczynowe-skutkowych/ oraz zadaniowych tj. przy-

gotowujących projekty - warianty zadań i celów, w różnych horyzontach czasowych i tematycznych. Oznaczać to będzie relatywny spadek udziału informacji ewidencyjnych na rzecz - ogólnie mówiąc - informacji zarządczej. Warto przy tym podkreślić, że ten trend oznacza również odwrócenie proporcji między informacją "rzeczową" a informacją finansowo-kosztową. Ta orientacja nabierać będzie w coraz większym stopniu wiodącego charakteru.

6/ Biorąc pod uwagę formalne atrybuty informacji, restrukturyzacja popytu oznacza przejście z masowych standardowych i zrutynizowanych strumieni informacji na wzrost udziału informacji specjalistycznych, charakteryzujących się na ogół wąskim strumieniem danych wejściowych często stanowiących wynik "pośredniego" przetwarzania, "głębokim" algorytmem transformacji oraz różnorodnością wyjść użytkowych o dużym udziale wyjść "na życzenie".

7/ Przyjmując za kryterium lokalizację zjawisk jako miejsca powstania oryginału odwzorowywanego przez model informacyjny, restrukturyzacja oznacza wzrost udziału tych informacji, które będą opisywały zjawiska zachodzące w otoczeniu organizacji gospodarczej, co prowadzi do relatywnego spadku informacji o zjawiskach zachodzących "wewnątrz jej". Wymagać to będzie zmian również w strukturze oraz kierunku informacji przepływających między przedsiębiorstwami /i jednostkami równorzędnymi/ a szeroko pojętym Centrum, co łączy się ze zmianami funkcji urzędów centralnych. O ile dotąd przepływy informacji miały wyraźny kierunek "w górę" /Centrum odgrywało tu rolę pompy ssącej/, o tyle teraz należy założyć przepływ dwustronny - "w górę" i "w dół". /Centrum odgrywać będzie rolę pompy ssące-tłoczącej/. Związane jest to z obowiązkiem centrum informowania o własnych założeniach, koncepcjach i wariantach polityki gospodarczej - niezależnie od informacji obligatoryjnych /parametry sterujące, sposób funkcjonowania mechanizmów sterowania itp./.

8/ Jeśli za kryterium przyjąć kierunek przepływu informacji w ramach organizacji gospodarczej i między nimi, restrukturyzacja popytu oznaczać będzie tendencję ograniczania do niezbędnego minimum przepływów pionowych przy intensyfikacji powiązań informacyjnych poziomych, skośnych i krzyżowych.

9/ Jeśli wreszcie za kryterium przyjmiemy poziom pewności i jednoznaczności informacji restrukturyzacja popytu oznaczać będzie wzrost udziału informacji obarczonych ryzykiem i niepewnością - w postaci np. tworzenia "supozycji", "hipotez", "przewidywań", "predykcji" itp. przy relatywnym spadku istniejącej dziś tendencji do wyłącznie dokładnego odwzorowywania badanych atrybutów zjawisk w sensie generowania informacji opartych na dokładnej i szczegółowej dokumentacji źródłowej. Ten typ restrukturyzacji będzie obserwowany we wszystkich poziomach struktury organizacyjnej gospodarki narodowej.

Jak się wydaje, w świetle tych tendencji zasadność i racjonalność budowy systemów informatycznych klasy III i IV można uznać za uwidocznioną biorąc pod uwagę prezentowany stopień ogólności naszych wywodów.

B. Problem budowy systemów klasy III i IV

1. Stawiamy tezę, iż metodyka budowy informatycznych systemów klasy III i IV, w stosunku do metodyki właściwej dla systemów agendowych różni się zasadniczo conajmniej w zakresie trzech struktur: informacyjnej, przestrzennej oraz organizacyjnej. W strukturze informacyjnej systemu istotą jest odpowiedź na pytania: w jakim celu dla jakich funkcji, co, po co, dlaczego i dla kogo i jak często przetwarzać informacje. Mówiąc o strukturze przestrzennej mamy na myśli lokalizację źródeł informacyjnych /WE/ oraz lokalizację odbiorców informacji wraz z analizą możliwości wykorzystania WE/WY przez wielu użytkowników. Jeśli chodzi o strukturę organizacyjną chodzi

tu o szeroką paletę koniecznych uwarunkowań które muszą być spełnione aby omawiane dwie klasy systemów mogły prawidłowo funkcjonować /problemy struktur organizacyjnych, problemy szkoleniowe, barier hamujących rozwój tych systemów: metodologicznej, psychologicznej itp./.

2. Odmienności budowy systemów klasy III i IV - dają się wprowadzić do trzech zasadniczych grup, a mianowicie:

a/ przejście z podejścia retrospektywnego na prospektywne. Istota tej zmiany polega na przejściu od analizy "post factum" na antycypację zjawisk. Informacje generowane przez systemy agendowe dostarczały informacje o zdarzeniach, które zaistniały gdy informacje generowane przez systemy klasy III i IV mają orientację przyszłościową /w różnym horyzoncie czasu/. Przejście z myślenia retrospektywnego na prospektywne wymaga jednak ogniwa pośredniego którym jest analiza ekonomiczna, która wiąże badanie przyczyn /w przeszłości/ ze skutkami /bieżącymi i/ lub przyszłymi/.

b/ Przejście ze statyki na dynamiczny opis zjawisk, a więc z oceny "punktowej" /na dany moment czasu/ na ocenę tendencji /trendów/. Wiąże się to m.in. z nadaniem "prawa obywatelstwa" informacjom z gatunku: hipotez, szacunków itp. przybliżeń, co wprowadza w orbitę zainteresowania czynnik ryzyka /niepewności/ w generowanych informacjach, czyli myślenie o zjawiskach gospodarczych w kategoriach probabilizmu /opartego o obliczenia albo nawet o heurezę/. Jednocześnie oznacza to przejście na metajęzyki opisu zdarzeń co automatycznie pociąga za sobą wzrost straty informacyjnej /odejście od dokładności i szczegółowości informacji/ ale przy równoczesnym zysku otrzymania złożonych informacji użytkowych o dużej informacyjnej nośności /której w inny sposób osiągnąć nie można/.

c/ Przejście z szczegółów na agregaty i na koordynację działań współzależnych, co prowadzi do zmodelowania zjawisk z punktu widzenia wielu kryteriów, które można synchronizować w spójne całości ze względu na cel modelu.

3. Można wskazać dwie główne możliwości budowy struktury informacyjnej systemów klasy III i IV: podejście problemowe i podejście poprzez typizację procedur.
Pierwsza możliwość zasadza się na następującej procedurze o charakterze zindywidualizowanej, tzn. stosowanej osobno dla każdorazowego użytkownika /instytucji/:

a/ Postrzeżenie i określenie istoty problemu /oryginału/, który staje się przedmiotem rozstrzygnięcia,

b/ Wybór kryteriów dla budowy modelu /cel, ograniczenia, zmienne, relacje/,

c/ Budowa modelu,

d/ Analiza wystarczalności informacji dla uruchomienia modelu /wyznaczenie zakresu luki informacyjnej/,

e/ Określenie sposobu likwidacji luki informacyjnej / sposobu uzyskania informacji uznanej za istotną i brakującą/,

f/ Realizacja modelu /wraz z dalszym jego rozwojem/.

Syntetyzując, ten sposób doprowadza do sekwencji - problem - model - algorytm - informacja końcowa /użytkowa/. Do zalet tego ujęcia można zaliczyć: uwzględnienie specyfiki użytkownika, maksymalne uwzględnienie aktualnych potrzeb - a zatem maksymalny sposób dostosowania rozwiązań do rzeczywistości przy uwzględnieniu warunków lokalnych. Wady takiego ujęcia to głównie: rozproszenie sił i środków na rozwiązania, incydentalne uzależnienie stopnia poprawności rozważań od posiadanych środków i potencjału intelektualnego, straty czasowe.
Możliwość druga wychodzi z założenia, że problematyka funkcjonowania przedsiębiorstwa niesie za sobą duży stopień podobieństwa problemów, które mogą być rozwiązywane w podobny sposób

przy uwzględnieniu pewnych typowych klas uwarunkowań /np. technicznych, organizacyjnych itp./. Pozwala to na typizowanie problemów, przy czym typizacja ta może przebiegać dwojako:

- problem standaryzowany - mutacje - algorytm /mutacje/ - informacja użytkowa,
- typ algorytmu - klasy problemów dające się rozwiązać - informacja użytkowa.

Zalety tego podejścia można ująć następująco: typizacja rozwiązań klas problemów w oparciu o problemowe pakiety standardowe, opracowywanie pakietów przez wyspecjalizowane ośrodki dysponujące odpowiednim zapleczem fachowym, możliwość uwzględniania w pakietach rozwiązań dostosowanych zarówno do tematyki problemów jak i do typowych konfiguracji komputerowych, rozprapogowywanie pakietów w drodze sprzedaży /dzierżawy/, co ma wpływ na koszty i ceny takich przedsięwzięć, możliwość modułowej konstrukcji pakietów /w dostosowaniu do klas założonych uwarunkowań/, możliwość adaptacji pakietów do szczegółowych warunków lokalnych - co radykalnie zmniejsza koszty własne i umożliwia szybką instalację użytkową pakietu u konkretnego użytkownika, zgodność z trendami światowymi. Jak się wydaje, podejście to nie wykazuje zdecydowanych wad, natomiast realizacja może napotkać istotne trudności, zwłaszcza w początkowym okresie, związane z wyborem problematyki i opracowaniem ich mutacji - jako podstawy do projektowania /występowanie barier/.

4. Przeprowadzone wyżej rozważania pozwalają na pewną syntezę struktury informacyjnej systemów klasy III i IV. Przedstawia ją tablica 2.

Tablica 2

Cel systemu / Zakres /funkcje/ systemu	Analiza, ocena kontrola	Wyznaczenie celów i kryteriów dróg dojścia	Ustalenie polityki realizacji celów
Generowanie informacji przyczynowo-skutkowych	T	T	T
Wariantowanie decyzji dopuszczalnych	N	T	T
Informowanie kierownictwa	T	N	T

T - tak; N - nie.

Tablica 2 ilustruje tezę, że systemy klasy III i IV mogą wystąpić w trzech głównych formach:

a/ systemy generujące informacje przyczynowo-skutkowe. Mają one głównie charakter analityczny, oparte są o dane historyczne, wskazują na związki przyczynowo-skutkowe występujące między obserwowanymi zmiennymi, informują o związkach, współzależnościach, sile i kierunku powiązań między zmiennymi, mogą mieć charakter odcinkowy /np. majątek trwały , problematyka oceny stanu ekonomiczno-finansowego itp./ lub całościowy - tj. dotyczący przedsiębiorstwa traktowanego in toto.

b/ Wariantowanie decyzji dopuszczalnych. Są to systemy rodzaju "Decision Support Systems" orientowane na wybrane problemy złożone, związane z kierowaniem polityki gospodarczej całego przedsiębiorstwa lub jego pewnych fragmentów - w różnych horyzontach czasu i różnej strukturze tematycznej.

c/ Systemy Informowania Kierownictwa. Są to systemy, które widzimy jako te, które generują syntetyczną informację o różnej konfiguracji elementów kontroli, analizy i oceny. Systemy te przejawić się mogą głównie w trzech układach, a to jako:

- informacja odchyleniowa - zarówno od wskaźników i parametrów ekonomiczno-technicznych zadanych jawnie tj. pochodzących z istniejących zapisów w dokumentach, jak też od wskaźników i parametrów uwikłanych, tj. stanowiących relacje na danych jawnych,
- informacja alarmowa - o zagrożeniach w realizacji obowiązujących norm i parametrów /jawnych - uwikłanych/, z komentarzem co do ewentualnych przyczyn - skutków, źródła i miejsca wystąpienia,
- informacja strukturalna - mówiąca o strukturze danego zjawiska, z położeniem nacisku na ocenę stopnia stabilności lub zmian dostrzeganych w tej strukturze, wraz z podaniem omówienia tendencji /przyczyny - skutki/.

5. W tworzeniu systemów klas III i IV występują określone hamulce. Do głównych zaliczyć należy /jeśli świadomie pominiemy aspekty techniczno - technologiczne/:

a/ bariery instytucjonalne. Mamy tu na myśli brak wystarczająco silnych presji wywieranych przez aktualny model wdrażanej reformy do tworzenia popytu /a tym samym i podaży/na informacje generowane przez systemy klasy III i IV. Jest to związane głównie z:

- tendencjami do utrzymywania istotnych obszarów działalności przedsiębiorstwa pod kontrolą działań dyrektywno-rozdzielczych /co wynika ze zjawisk kryzysowych/,
- preferowania działań operacyjnych, doraźnych, wymagających szczegółowej i masowej informacji operacyjnej,

b/ bariery psychologiczne. Myślenie w kategoriach rachunku ekonomicznego, wariantowania rozwiązań itp. nie zostało w latach ubiegłych zaszczepione przez system zarządzania. Powrót do tych kategorii myślenia ujawnia brak przygotowania /wiedza/, oraz konieczność ponoszenia ryzyka za wyłączną własną odpowiedzialność, co budzi opór powiązany z neofobią.

c/ bariery organizacyjne. Wiążą się głównie z przejściem na analizę tzw. problemów stykowych tj. takich, które leżą w orbicie zainteresowania dwóch i więcej komórek organizacyjnych przedsiębiorstwa. Tego typu podejście poddaje zasadniczej krytyce obecny sposób budowy struktur organizacyjnych, które wykazują wyłączną orientację agendową a nie problemową.

4. Uwagi końcowe

1. Rozwój informatyki dla celów zarządzania wymaga podjęcia istotnej decyzji o znaczeniu strategicznym dla kierunków dalszych działań. Podstawowy problem generalnego charakteru można przedstawić w pytaniu, czy agendowe systemy, sposób ich

budowy i funkcjonowania są wystarczające dla zarządzania w dłuższym horyzoncie czasu. Przyjmujemy, że odpowiedź na to pytanie będzie przecząca. W przeciwnym razie wszystko to co napisane zostało wyżej uznać należy za hobbystyczny punkt widzenia, o charakterze science fiction.

2. Jeśli opowiadamy się za rozwojem, a zatem za wprowadzeniem w orbitę zainteresowań systemów klasy III i IV, powstają problemy realizacyjne. A oto kilka propozycji w tym zakresie:

a/ rozwój systemów klasy III i IV powinien być zakładany w dłuższym horyzoncie czasu i powinien być etapowany,

b/ etapowany rozwój tych systemów powinien objąć sobą trzy równoległe podejścia:

- praktyczne /aplikacyjne/ - na poziomie przedsiębiorstwa,
- normatywne,
- metodologiczne, w tym:
  - teoretyczno-szkoleniowe,
  - projektanckie,

c/ Aplikacyjne podejście powinno rozważyć możliwości budowy systemów informowania kierownictwa, jako łącznika między systemami agendowymi a systemami decyzyjnymi: SIK w zaproponowanym rozumieniu tkwi bowiem w filozofii systemów agendowych, natomiast - niezależnie od już stosowanych - generuje wyjścia użytkowe o bardziej złożonym efekcie informacyjnym. Niezależnie od użytecznych informacji spełnić on może funkcję popytotwórczą.

d/ Podejście normatywne ma na celu nadanie właściwej rangi problematyce informacji w reformie gospodarczej. Wydaje się celowe postulować, aby odpowiednie organy centralne podjęły w tej sprawie stosowne postanowienia o charakterze normatywno-zalecającym. Jednocześnie rozwiązania normatywne reformy /np. dotyczące oceny działalności podmiotów gospodarczych, wymagań rachunku ekonomicznego itp/ powinny być ustalone w ten sposób, aby wymuszały /czyli stanowiły czynnik popytotwórczy/ infor-

macje o wyższym stopniu przetworzenia, przynajmniej "punktowo" - w etapie wstępnym. Ten problem wymaga osobnego naświetlenia w sposób bardziej szczegółowy.

e/ Podejście metodyczne o charakterze teoretyczno - szkoleniowym orientowane być powinno na uwzględnianie zasad budowy systemów typu III i IV w programach wyższych szkół ekonomicznych. W tej sprawie poczyniono już pewne kroki w SGPiS poprzez włączenie tej problematyki do programu znowelizowanego programu studiów na Wydziale Finansów i Statystyki /specjalność: projektowanie SIZ/. Niezależnie od tego, problematyka budowy informacyjnych i informatycznych systemów zarządzania powinna być wprowadzona do programów doskonalenia zarówno kadr kierowniczych przedsiębiorstw i instytucji, jak i projektantów SIZ.

f/ Podejście metodyczne o charakterze projektanckim dotyczyć powinno rozpoczęcia prac wstępnych nad możliwością, oraz sposobem opracowywania i zakresem tematycznym pakietów standardowych związanych z problematyką Systemów typu III i IV.

Z powyższego wynika, że rozwiązanie budowy systemów "ponadagendowych" oznacza jakościowo nowy etap rozwoju informatyki dla celów zarządzania. Zdaniem autora, etap ten - aczkolwiek możliwy do realizacji w dłuższym horyzoncie czasu - jest niezbędny dla prawidłowego sterowania przedsiębiorstwem, w kierunkach zgodnych z duchem reformy gospodarczej.

Hermann Krallmann 1/

COMPUTERUNTERSTÜTZTE UNTERNEHMENSPLANUNG

1. EINFÜHRUNG

Wachsende Komplexität und Dynamik sowie abrupte Veränderungen der Unternehmensumwelt erschweren das Festlegen strategischer Unternehmensziele, das Gestalten alternativer Strategien und Programme zu deren Realisierung. Neuartige Phänomene wie Ressourcenknappheit aus politischen Gründen und ökologische Bewußtseinsänderungen produzieren zusätzliche Konfliktherde. Demographische Trends, technologische Durchbrüche in Verbindung mit hohen Diffusionsgeschwindigkeiten bereiten unvorhergesehene Strukturprobleme. Unkontrolliertes Wachstum, schwerfällige Organisationsstrukturen in der Führungsspitze, mangelhafte Kontroll- und Berichtssysteme sowie Finanzierungsfehler sind in ihrer gegenseitigen Abhängigkeit Indikatoren für einen nicht zielgerichteten Wandel des Unternehmens in einer ständigen Veränderungen unterworfenen Umwelt. Um so mehr ergeht die Suche nach potentiellen, rechnergestützten Instrumentarien zur Erfassung, Erweiterung und Bewältigung des Informationsangebotes als Grundlage für die Früherkennung krisenträchtiger Prozesse und Handhabung komplexer Entscheidungssituationen in Unternehmen. Rechnergestützte Entscheidungsunterstützungssysteme werden in der heutigen Zeit den oben genannten Anforderungen, die durch unternehmensexterne und -interne Charakteristika formuliert werden, in der Weise gerecht, daß sie

- Maßnahmen zur rechtzeitigen Erfassung von bedrohlichen, aber auch erfolgversprechenden Entwicklungen

- Maßnahmen zur beschleunigten Alternativenfindung, -bewertung und -auswahl in komplexen Entscheidungssituationen

bereitstellen (Bennet 1983; Keen 1980; Krallmann 1981).

1/ Technische Universität Berlin

Im folgenden werden zwei Entscheidungsunterstützungssysteme vorgestellt, die mit großem Erfolg in namhaften deutschen Unternehmen implementiert wurden.

DECISION SUPPORT SYSTEM ZUR ENTWICKLUNG KOORDINIERTER PRODUKTIONS- UND ABSATZSTRATEGIEN

1. PROBLEMSTELLUNG

Die strategische Unternehmensplanung wird durch eine Vielzahl von Faktoren geprägt, deren simultane Berücksichtigung infolge von Zeitvarianz, hoher gegenseitiger Abhängigkeit und stochastischem Charakter die Grenzen menschlicher Fähigkeiten übersteigt. Zur Beseitigung dieser menschlichen Schwachstellen sowie zur Unterstützung der kreativen Komponente haben sich computergestützte Planungsmodelle bewährt. Ihr breiter Einsatz insbesondere direkt durch den Entscheidungsträger wird jedoch durch das Fehlen adäquater, sprachlich angepaßter Kommunikationsschnittstellen behindert. Meist ist noch die Zwischenschaltung von EDV-Spezialisten zur Handhabung verschiedener Soft- und Hardware-Komponenten erforderlich, wodurch der kreative Prozess der Strategieformulierung, -bewertung und -auswahl unterbrochen wird. Zur Erhöhung der Akzeptanz und Diffusion komplexer Methoden in die Managementebene bedarf es einer einheitlichen Sprachumgebung, in die alternative Methoden und evtl. Standardsoftwarepakete zur Modellformulierung, Alternativengenerierung und entscheidungsorientierten Ergebnisaufbereitung integriert sind bzw. in flexibler Erweiterung entsprechend sich ändernden Bedürfnissen integriert werden können.

2. KONZEPTION DES DECISION SUPPORT SYSTEMS

Die Gestaltung des realisierten Decision Support Systems (DSS) - im deutschen: Entscheidungs-Unterstützungs-System (EUS) - wurde dementsprechend auf die Erfüllung folgender Anforderungen ausgerichtet:

- interaktive Modellformulierung mit alternativen Simulations- und Planungssprachen, vornehmlich deskriptiver Natur (z.B. DYNAMO)
- Parametrisierung und Kopplung von vorgefertigten Teilmodellen
- Alternativengenerierung durch interaktive Definition und Variation von Modellvariablen als Repräsentanten des unternehmenspolitischen Instrumentariums
- interaktiver Anstoß und automatische Steuerung von Simulationsläufen alternativer Strategien mit Speicherung frei wählbarer Ergebnisgrößen
- interaktive Auswahl alternativer graphischer Ergebnisaufbereitungsmethoden (z.B. SAS) mit Definition der zu vergleichenden Ergebnisgrößen und Strategien.
- Sensitivitätsanalyse singulärer Strategien
- synergetische Kombination singulärer Strategien

Das entwickelte, erweiterbare Rahmensystem besteht demzufolge aus den folgenden Komponenten:

1. Kommunikationsteil
2. Steuerteil
3. Modellbank
4. Methodenbank
5. Datenbank

3. DIALOGSTRUKTUR DES KOMMUNIKATIONSTEILS

In Abb. 1 ist der Selektorbaum des Dialogs für den Teil das Entscheidungs-Unterstützungs-Systems dargestellt, der die Generierung, Bewertung und Auswahl unternehmenspolitischer Strategien auf der Basis zuvor definierter Simulationsmodelle betrifft.

Nach der Auswahl eines Modells, z.B. zur Produktions- und Absatzplanung stehen folgende Funktionen und Unterfunktionen zur Verfügung:

1. Variablenmodifikation zur Definition von Unternehmensstrategien bzw. Umweltszenarien

   1.1 Simulationsvariablen
   Hierunter sind Steuerparameter zusammengefaßt wie z.B. die Laufzeit des Modells, die Zeitintervalle zur Speicherung von Ergebnissen etc.

   1.2 Modellvariablen
   Hier findet die eigentliche Definition von Unternehmensstrategien bzw. Umweltszenarien statt, indem die Modellgrößen des definierten unternehmenspolitischen Instrumentariums, z.B. Einstellpolitik, Werbebudget, Kapazitätsinvestitionen etc. durch Parameter (u.U. zeitvariant) festgelegt werden.

   1.3 Simulationslauf
   Dieser Parameter dient zum Abschluß einer Strategie-Alternative und zu deren Benennung, unter der später in der Ergebnisaufbereitung auf sie zugegriffen werden kann.

2. Auswahl Ergebnisaufbereitung

Auf der Basis einer Übersicht über gespeicherte Variablenverläufe unterschiedlicher Alternativen können beliebige Kombinationen von zu vergleichenden Zielgrößen verschiedener Strategien zusammengestellt und mit einer Aufbereitungsmethode aus dem angeschlossenen Methodenpool vornehmlich graphisch dargestellt werden.

3. Modelldokumentation

Die parallel zur Modellerstellung produzierte Dokumentation der Modelle kann jederzeit auf verschiedenen Aggregationsniveaus abgefragt werden, z.B. die allgemeine Beschreibung in der Phase der Modellbildung, -auswahl und -verknüpfung oder die einzelnen Variablenbeschreibungen in der Phase der Definition und Variation von Unternehmensstrategien.

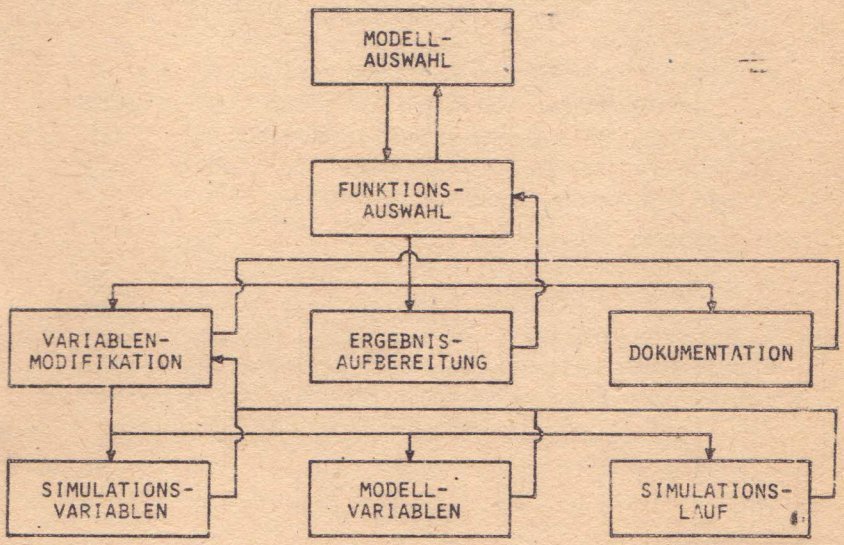


Abb. 1: Ausschnitt des Selektorbaumes

4. DEMONSTRATIONSMODELL

Als Demonstrationsbeispiel wird ein Simulationsmodell zur Entwicklung koordinierter Produktions- und Absatzstrategien für Großmotorenhersteller vorgeführt. Es besteht aus drei eigenständig lauffähigen Teilmodellen, die in der in Abb. 2 dargestellten Weise gekoppelt wurden:

1. Teilmodell Schiffsmarkt

Das Teilmodell Schiffsmarkt dient der langfristigen Prognose der Nachfrage-Entwicklung nach Großmotoren in Abhängigkeit von alternativen Umweltszenarien, z.B. Umfang und Verteilung des Transportvolumens, Treibstoffkosten etc.. Hierzu wird die Strukturdynamik der Weltschiffstonnage nach Regional- und Größenklassen abgebildet.

2. Teilmodell Marketing

Das Teilmodell Marketing dient der Prognose der Unternehmensentwicklung in Abhängigkeit vom Einsatz des unternehmenspolitischen Instrumentariums, z.B. Investitionen in puncto Produktionskapazität, Umfang und Distributionsgrad das Außendienstes, Werbebudget, Preiskonditionen etc.. Als Bewertungsgrößen für die Unternehmensentwicklung sind Absatzmenge, Marktanteil und monatliches Betriebsergebnis vorgesehen.

3. Teilmodell Produktion

Das Teilmodell Produktion bildet die Entwicklung des dreistufigen Produktionsprozesses incl. Materialdisposition in Abhängigkeit vom Einsatz des unternehmenspolitischen Instrumentariums ab. Es ist parametrisierbar z.B. bezüglich des Halbfertigungsgrades, der Bestellpolitik etc..

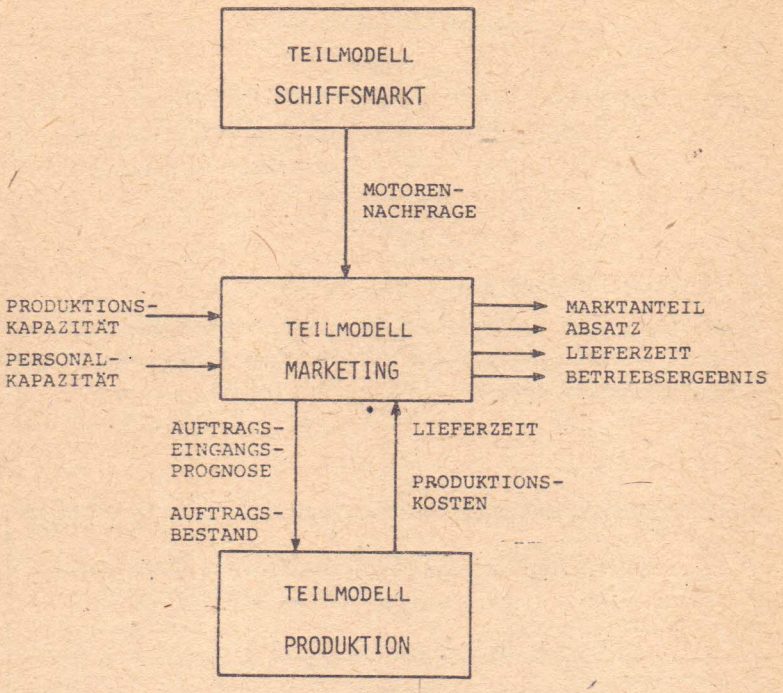


Abb. 2: Kopplungsstruktur des Demonstrationsmodells

5. ANWENDUNCSBEISPIELE

Nachfolgend sind exemplarische für den Einsatz des Entscheidungs-Unterstützungs-Systems auszugsweise einige wesentliche Bildschirmmasken dargestellt.

```
[illegible] - BERLIN                                          FG - SYSTEMANALYSE/EDV

*****************************************************************************

  ***  M O D E L L   V A R I A B L E N        ***  BSM 1.1.2  ***

          BITTE WAEHLEN SIE ===> mva

            MVA - MODELL VARIABLEN AENDERN

            END - ENDE

    BASISKAPAZITAET          ===>  89  (MOTOREN/MONAT)
    (BRAZ)

    EINSTELLUNGEN IM
    AUSSENDIENST             ===>  83  (PERSONEN/6 MONATE)

    (PERANZ)

-----------------------------------------------------------------------------

  WEITER MIT (ENTER)-TASTE
```

Abb. 3: Modellvariablenmodifikation

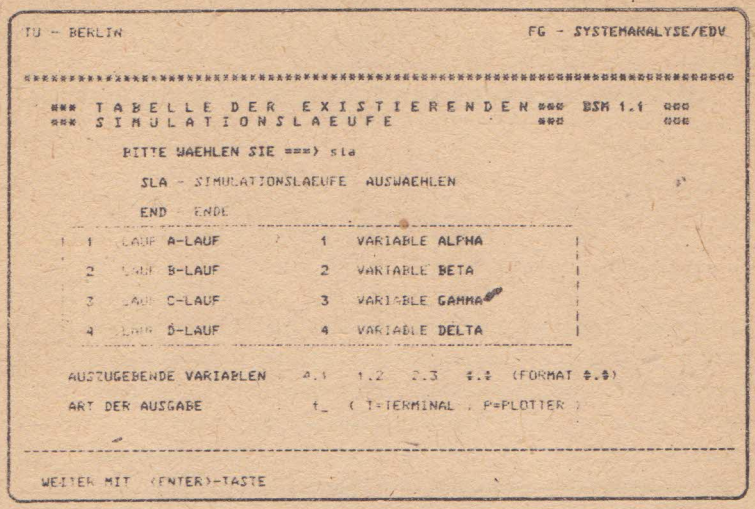


Abb. 4: Zusammenstellung zu vergleichender Zielgrößen alternativer Strategien

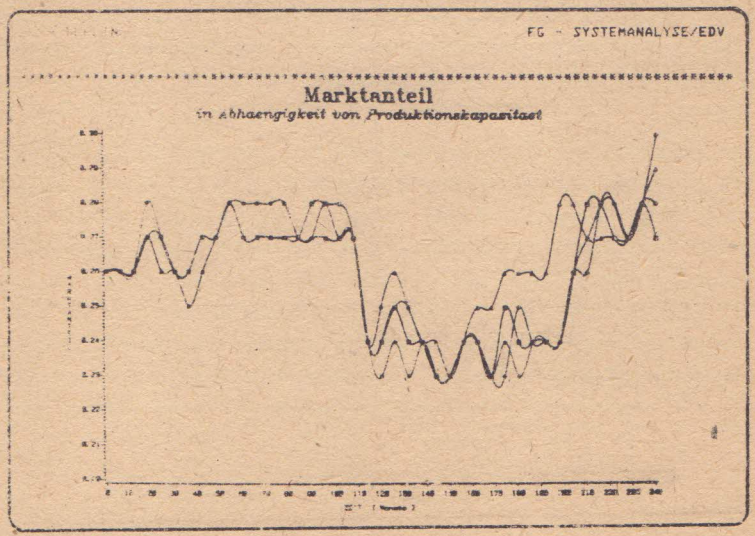


Abb. 5: Graphische Darstellung alterna

DECISION SUPPORT SYSTEM ZUR PREISGESTALTUNG VON MEHRPRODUKTUNTERNEHMEN

1. PROBLEMSTELLUNG

Die Preisgestaltung von Mehrproduktunternehmen gegenüber Großkunden, die eine breite Produktpalette in großen Stückzahlen abnehmen, wird häufig von individuellen, mehrtägigen Verhandlungsrunden über Nettopreise und Mengenanteile geprägt. Hierbei stellt die Technik des preispolitischen Ausgleichs ein wichtiges Verhandlungsinstrument dar. Sein optimaler Einsatz wird dabei wesentlich von der Qualität der vor und während Verhandlungen verfügbaren Informationen sowie Bewertungs- und Entscheidungshilfen bestimmt. Von besonderer Bedeutung sind Flexibilität, Reaktionsschnelligkeit und Entscheidungsrelevanz, um einerseits Zielauswirkungen von Preis- und Mengenabweichungen direkt bewerten und andererseits über Art und Ausmaß geeigneter Strategie-Anpassungen entscheiden zu können.

Die bekannten Ansätze zur Preispolitik in den Wirtschaftswissenschaften bieten in dieser Hinsicht keine adäquate Unterstützung, da sie mit starken Vereinfachungen wie z.B. Einproduktunternehmen, idealtypischen Marktformen oder ausschließlicher Gewinnmaximierung arbeiten bzw. sich in einer wenig operationalen Beschreibung von preispolitischen Alternativen wie Rabattformen erschöpfen. Insbesondere die Erzeugung direkter Entscheidungshilfen für die Verhandlungssituation findet keinerlei Berücksichtigung.

2. KONZEPTION DES DECISION SUPPORT SYSTEMS

Die Gestaltung des realisierten Decision Support Systems (DSS) - im deutschen: Entscheidungs-Unterstützungs-System (EUS) - wurde dementsprechend auf die Erfüllung folgender Anforderungen ausgerichtet:

- iterativer, verhandlungsbegleitender Einsatz
- Generierung operationaler Verhandlungsziele
- Generierung flexibler Verhandlungsspielräume
- Berücksichtigung mehrdimensionaler Zielsysteme
- Einbezug des Verkäufer-Know-how
- parametrische Modellbildung mit interaktiver Konsistenzprüfung
- einfache, sichere und benutzerfreundliche Dialogschnittstelle
- transparente, entscheidungsorientierte Ergebnisaufbereitung

Zur Gewährleistung einer durchgehend rechnergestützten Verhandlungsführung wurde das in Abb. 1 dargestellte Gesamtkonzept mit folgenden Komponenten gewählt:

1. Kommunikationsteil
Der maskengesteuerte Kommunikationsteil wickelt alle Dialogfunktionen zwischen dem Verhandlungsführer und dem EUS in den Phasen der Verhandlungsvor- und -nachbereitung ab. Seine klare Abkopplung vom übrigen System gestattet eine flexible Anpassung an die spezifischen Bedürfnisse des Top- und Middle-Management.

2. Steuerteil
Der Steuerteil hat die Aufgabe, alle Systemfunktionen und das Zusammenspiel der verschiedenen Datenbasen zu koordinieren. Dazu gehören beispielsweise Parameterverwaltung, Modellgenerierung, Ergebnisaufbereitung, Sensitivitätsanalyse, Datenfernverarbeitung etc.

3. Methodenbank

Die Methodenbank enthält eine erweiterbare Sammlung von Programmbausteinen zur Aufbereitung von Produkt-, Kunden- und Ergebnisdaten sowie zur Modellbildung, Konsistenzprüfung und Generierung von Entscheidungshilfen (z.B. Lineare Optimierung etc.).

4. Produktdatenbank

In ihr sind alle Produktstammdaten zusammengefaßt, z.B. Listenpreise, verschiedene Kostenwerte oder Klassifizierungskriterien.

5. Kundendatenbank

Sie enthält Informationen über die Kundenvorgeschichte, z.B. Umsatz- und Auftragsbestandsdaten auf Produktebene oder bisher gewährte Rabatte.

6. Ergebnisdatenbank

In der Ergebnisdatenbank werden die generierten Verhandlungsziele sowie erreichte Teilergebnisse als Basis für verschiedenste Ergebnisaufbereitungen gespeichert.

7. Modellbank

Die Modellbank enthält vorgefertigte und erweiterbare Modellbausteine, aus denen der Steuerteil entsprechend der parametrischen Methodenwahl und Modelldefinition des Benutzers automatisch ablauffähige Modelle erzeugen kann. Beispiele für solche Modellbausteine sind Preisintervalle für Einzelprodukte und Produktgruppen, Preis-Mengen-Reaktions-Funktionen etc. Ferner werden in einem eigenen Segment alle Parameter für verschiedene Modellversionen des Benutzers gespeichert.

8. Portable Mikrocomputer

Zur direkten Unterstützung der Preisverhandlungen beim Kunden ist der Anschluß eines portablen Mikrocomputers vorgesehen, um Verhandlungsziele abfragen, Teilergebnisse erfassen und Zielabweichungen bewerten zu können. Mittels Datenfernverarbeitung werden Verhandlungsergebnisse an die Zentrale übermittelt bzw. angepaßte Verhandlungsziele zum Verhandlungsort überspielt.

9. Verhandlungsdaten

Hierunter fallen sowohl die generierten und überspielten Verhandlungshilfen wie z.B. optimale Preise, Bandbreiten etc. als auch die erzielten Verhandlungs-(teil-)ergebnisse.

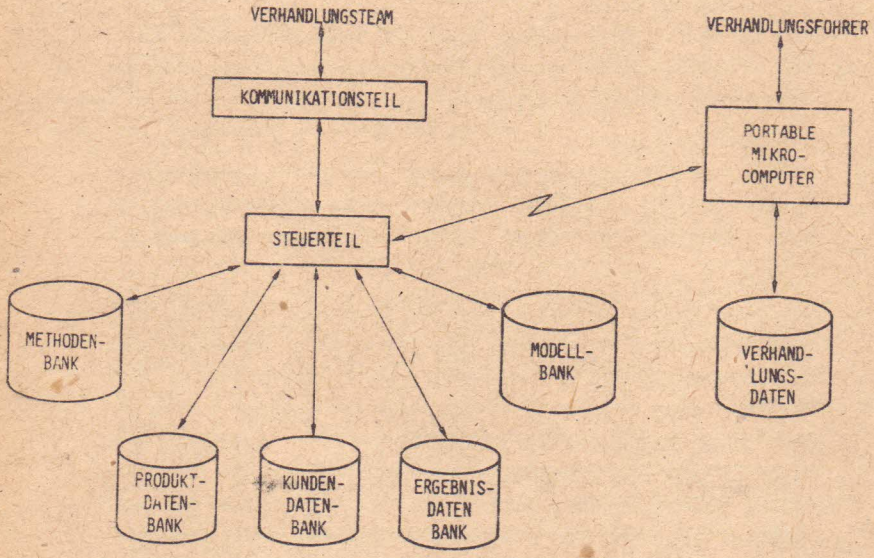


Abb. 1: Gesamtkonzept des DSS zur Preisgestaltung von Mehrproduktunternehmen

3. LEISTUNGSSPEKTRUM

Im einzelnen umfaßt das DSS zur Preisgestaltung von Mehrproduktunternehmen folgende Funktionen:

- interaktive Definition des Verhandlungspakets

In dieser ersten Phase werden durch Parameter die zu selektierenden Stammdaten für eine Simulation festgelegt. Hierzu gehören z.B. die Wahl der Simulationsebene (Kunde, Konzern oder Branche), die Wahl der Mengenbasis (Auftragsbestand, alter oder laufender Umsatz) und die Wahl der Kostenbasis (Herstellkosten, Selbstkosten).

- interaktive Definition bzw. Variation des Verhandlungsszenario

In dieser zweiten Phase können durch Parameter verschiedene Versionen von Verhandlungsszenarien für eine Simulation definiert oder verändert werden. Die Szenarien werden charakterisiert durch z.B. zulässige Preisintervalle, Mindestforderungen bezüglich Mengen oder Kennzahlen und Hypothesen über das Preis-Mengen-Reaktionsverhalten des Kunden etc..
Die Berücksichtigung von Teilergebnissen wird mittels automatischer Variation der Szenarien durch Überspielung der Parameter über Datenfernverarbeitung realisiert.

- interaktive Konsistenzprüfung

Unmittelbar vor der Modellgenerierung werden die Szenario-Parameter mehreren Konsistenzprüfungen unterzogen, um unlösbare Widersprüche aufzudecken und die Lösbarkeit des erzeugten Modells sicherzustellen. Ein Beispiel hierfür ist etwa ein geforderter Mindestquotient zwischen Preis und Kosten, der wegen der Preisobergrenze

nicht realisierbar ist. Die Parameterkorrektur wird durch Hinweise zur Inkonsistenzbeseitigung unterstützt.

- automatische Modellgenerierung

Der Steuerteil erstellt entsprechend der gewählten Entscheidungshilfe (Methode) aus den Simulations- und Szenarioparametern automatisch ein ablauffähiges Modell. Hierzu werden alle notwendigen Modellbausteine selektiert, mit Daten versorgt und anschließend verknüpft.

- automatische Generierung von optimalen Verhandlungszielen

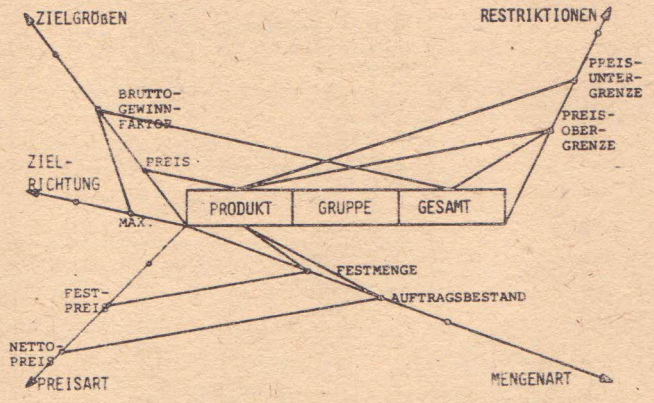
Mit Hilfe des erzeugten, konsistenten Modells können dann automatisch optimale Verhandlungsziele auf Einzelproduktebene berechnet werden. Dabei kann zwischen verschiedenen Zielkriterien gewählt werden. Die Ergebnisdaten werden für anschließende, spezielle Ergebnisaufbereitungen in einer Ergebnisdatenbank zwischengespeichert.

- automatische Generierung von Verhandlungshilfen

Neben der Überspielung der Ergebnisdaten auf einen portablen Mikrocomputer können je nach Ausbaustufe der Methodenbank zusätzliche Verhandlungshilfen online oder in Listenform bereitgestellt werden. Angereichert mit Stammdaten handelt es sich hierbei prinzipiell um

- optimale Preisziele
- Zielauswirkungen bei marginalen Preisabweichungen
- Bandbreiten von Preisabweichungen mit Angabe von von Kompensationsmöglichkeiten bezüglich anderer Produkte sowie Kompensationsdifferentiale

Abb. 2 verdeutlicht am Beispiel einer speziellen Fragestellung den Parameterraum der Modellierungskomponente, indem die gewählten Simulations- und Szenarioparameter als Vernetzung der Ausprägungen auf den Parameterachsen aufgetragen sind.



FRAGESTELLUNG: Wie sind die Preise der verhandelten Produkte zu wählen?

WENN der Bruttogewinnfaktor gesamt zu maximieren ist!

BEI verhandelter Preisart Nettopreis auf Mengenbasis Auftragsbestand,

festen Preisen und Mengen bereits verhandelter Produkte,

Preisober- und -untergrenzen je verhandeltem Produkt und

maximaler Preisänderung des gesamten Verhandlungspakets

Abb. 2: Beispiel für den Parameterraum der Modellierungskomponente

4. DIALOGSTRUKTUR DES KOMMUNIKATIONSTEILS

In Abb. 3 ist die Dialogstruktur des DSS zur Preisgestaltung von Mehrproduktunternehmen mit den normalen und optionalen Sprungmöglichkeiten dargestellt.

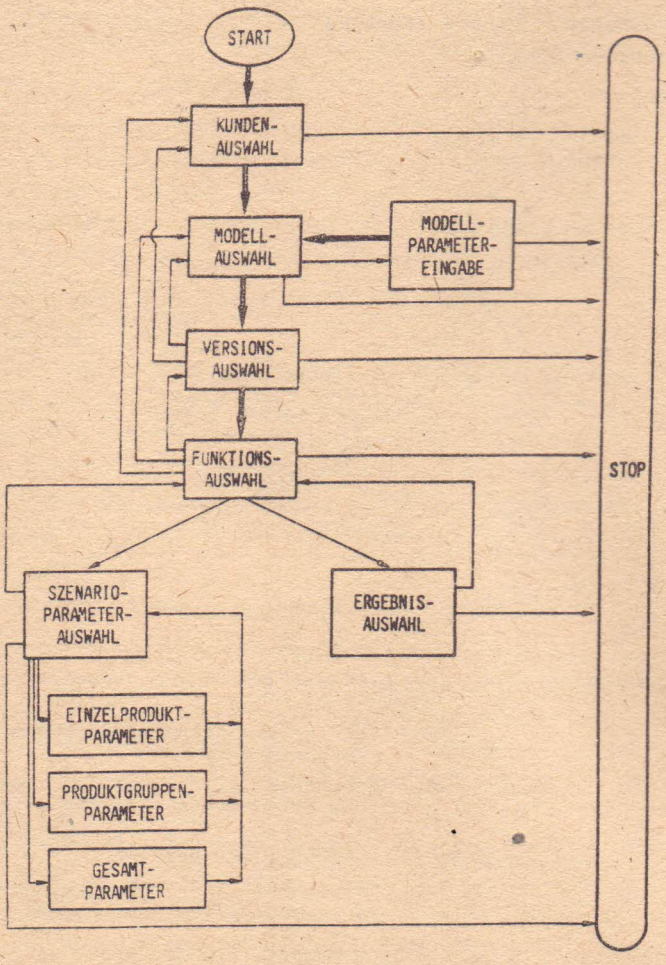


Abb. 3: Dialogstruktur des Kommunikationsteils

5. ANWENDUNGSBEISPIELE

Nachfolgend sind exemplarisch für den Einsatz des Entscheidungs-Unterstützungs-Systems auszugsweise wesentliche schirmmasken als Hardcopy dargestellt:

```
TU BERLIN,  P R E I S S I M U L A T I O N

*******************************************************************************

      M O D E L L A U S W A H L
      =========================

KUNDE      111111  FIRMA 1

SIMULATION-NR.  ===>  1          (BEI N -> KOPIE)
MODIFIKATION    ===>  _          N NEU, U UPDATE

GESPEICHERTE MODELLE
SIMULATION  MENGENBASIS  PREISARTEN   KOSTENBASIS  GRUPP.KRIT.
      1     Auftr-Bst.   Nett-Preis   HK/3         Bauart
      2     Ums-lfd-J.   Nett-Preis   HK/2         Preis-Kl.
      3     Ums-Vorj.    Li/N-Preis   HK/3         Bauart
      5     Auftr-Bst.   Nett-Preis   HK/3         Bauart
      6     Ums-lfd-J.   Li/N-Preis   HK/1         Preis-Kl.
      7     Ums-Vorj.    List-Preis   HK/2         Bauart
      8     Ums-Vorj.    Nett-Preis   HK/1         Preis-Kl.
      9     Ums-lfd-J.   List-Preis   HK/2         Preis-Kl.


KOMMANDO  ===>

KOMMANDOS      2 ENDE   3 KUNDENAUSWAHL   5 MODELLDATENEINGABE

PF1 FUER HELP AN BELIEBIGER STELLE
```

Abb. 4: Modellauswahl

```
TU BERLIN,  P R E I S S I M U L A T I O N

))) PARAMETER-INKONSISTENZEN ENTSPR. HINWEISEN KORRIGIEREN (((
********************************************************************************

      T Y P E N - P A R A M E T E R - E I N G A B E
      =============================================

KUNDE        111111          FIRMA 1
SIMULATION:  1               MENGENBASIS: AUFTR.-BST      PREISARTEN : NETT-PREIS
VERSION      1               KOSTENBASIS: HK/1            GRUPP.KRIT.: BAUART
STATUS       INKONSIST.


TYPE     10   PRODUKT 10                 KALK.MENGE  :  5000
GRUPPE    3   BAUART 3                  DECK.BEITRAG:  10000.00 DM   12.0 V.H.
                                        UMSATZ      :  40000.00 DM   16.0 V.H.

                    PREIS - GRENZEN                          BRUTTO-GEWINN-FAKTOR
LISTE / NETTO-ALT     M I N     M A X                        ALT          M I N
10.00    8.00     =) 7.50   =) 8.50  (ABSOLUT DM )          1.333    =) 1.500
 3.00             =)        =)       (AEND.  V.H.)
       20.00      =)                 (RABATT V.H.)

KONSISTENZGRENZEN              >= 9.00                                   (= 1.416
                               >=

KOMMANDO  ===>

KOMMANDOS       2 ENDE   6 FUNKTIONSWAHL   7 PARAMETERAUSWAHL

PF1 FUER HELP AN BELIEBIGER STELLE
```

Abb. 5: interaktive Konsistenzprüfung

```
TU BERLIN,  P R E I S S I M U L A T I O N

********************************************************************************

      E I N Z E L - T Y P E N - B A N D B R E I T E N
      ===============================================

KUNDE        111111          FIRMA 1
SIMULATION   1               MENGENBASIS: AUFTR.-BST      PREISARTEN : NETT-PREIS
VERSION      1               KOSTENBASIS: HK/1            GRUPP.KRIT.: BAUART
STATUS       OPTIMIERT


TYPE     10   PRODUKT 10

PREIS   ALT   8.00 DM    MENGE   ALT   5000
        OPT   8.50 DM            OPT   5000

BEI KOMPENSATION   VON OPT     JE 10 PFG   BIS MAX     B A N D B R E I T E N
DURCH              PREIS       SENKUNG     PREIS     UNTERGRENZE  OBERGRENZE
 5  PRODUKT 5      11.00 DM     4 PFG      11.40 DM    7.50 DM     8.80 DM
12  PRODUKT 12      9.00 DM     9 PFG       9.45 DM    8.00 DM     9.00 DM
15  PRODUKT 15      4.50 DM    14 PFG       5.06 DM    8.10 DM     9.00 DM

KOMMANDO  ===>

KOMMANDOS       2 ENDE   6 FUNKTIONSWAHL   8 ERGEBNISAUSWAHL

PF1 FUER HELP AN BELIEBIGER STELLE
```

Abb. 6: Verhandlungsziel mit Bandbreiten für Einzelprodukt

LITERATUR

Bennet, J.L. (Hrsg.):

Building Decision Support Systems.
Reading, Mass., USA, 1983

Keen, P.G.W.:

Decision Support System: Trasnlating analytic
Techniques into useful Tools.
In: Sloan Management Review, Vol. 21, No. 3, 1980

Krallmann, H.:

Strategische Entscheidungsunterstützungssysteme
im Unternehmen.
In: Bergner, H. (Hrsg.): Planung und Rechnungswesen
in der Betriebswirtschaftslehre. Duncker & Humblot,
Berlin, 1981

Wojciech Olejniczak [1/]

ZASADY PROJEKTOWANIA SYSTEMÓW INFORMATYCZNYCH PODŁUG STRUKTUR

1. Wprowadzenie metodologiczne

Że system informatyczny zarządzania można dzielić na cztery struktury:

- funkcjonalną,
- informacyjną,
- techniczną,
- przestrzenną,

to już wiemy. Jednak nie zawsze ten podział jest w sposób uświadomiony uwzględniany w praktyce projektowania. Zwykle, uwzględniając raczej interesy systemu liczącego niż użytkownika, cały wysiłek projektanta koncentruje się na projektowaniu tego fragmentu struktury informacyjnej, który ma decydujące znaczenie w styku z maszyną niźli z człowiekiem. Nie od rzeczy zatem będzie poparcie w niniejszym referacie zasady harmonii w projektowaniu systemów informatycznych, przejawiającej się w sprawiedliwym traktowaniu różnych, nie tylko informacyjnych aspektów systemu. Wiele zależy w realizacji zasady harmonii od postawy przyjętej przez projektanta. W zasadzie projektant postrzega pewne całości istniejące w świecie zewnętrznym i odczytuje je jako pewne systemy - etap analizy i realizuje je na wzór obmyślonej konstrukcji systemowej, aczkolwiek odpowiedniość tej konstrukcji jest niezupełna.

Możliwym jest w zasadzie przyjęcie przez projektanta dwojakiej postawy systemowej, w której 1° obmyśla system o zidenty-

---

1/ Instytut Informatyki i Zarządzania Politechniki Szczecińskiej

fikowanych powiązaniach tworzących określoną strukturę, a o nie zidentyfikowanych / a zatem dowolnych/ składnikach, bądź też 2° system posiada zdefiniowaną strukturę i składniki. Wyrażamy pogląd, że w przypadku projektowania systemu konkretnego, dedykowanego konkretnemu użytkownikowi właściwa jest postawa druga. Stąd też wprowadzamy definicję uznającą, że " projektowanie to łączenie znanych elementów w nowy, niepowtarzalny sposób". Ta definicja, przyjmuje za punkt wyjścia określenie systemu jako "zbioru obiektów i relacji między nimi lub ich właściwościami" /Hall/, co daje jej zgodność z inżynierią systemów,którą to dyscyplinę można uznać za nadrzędną w stosunku do projektowania SIZ.

Preferowanie zaś postawy pierwszej wymagałoby przyjęcia innej definicji systemu, bardziej ogólnej np. /Mesarovic/. Tą drogą jednak, przynajmniej w niniejszym referacie, nie pójdziemy.

Projektowanie SIZ wiąże się także z ogólną metodologią projektowania. O ile jednak nowoczesna koncepcja projektowania dąży do wyizolowania samej istoty projektowania i sformułowania jej jako pewnego standartu, który byłby na tyle ogólny, aby odnieść go do wszystkich sytuacji projektowych, to projektowanie SIZ niesie z sobą pewną specyfikę. Wynika to z faktu, że przedmiotem są otwarte, socjo - techniczne - informacyjne systemy, oczywiście sztuczne i o dużym stopniu złożoności. Systemy takie wykazują tendencję do antyintuicyjnego zachowania. Projektowanie SIZ wymaga zaangażowania dużych zespołów, dysponujących określoną metodologią projektowania, bazą techniczną - w rzeczywistości projektowane są przez system projektujący, który wspomaga też swoją działalność komputerem.

Obecnie wśród metod szczegółowych projektowania SIZ najczęściej stosuje się projektowanie strukturalne. Projektowanie strukturalne wprowadza podział strukturalny przedsięwzięcia na podsystemy. Redukuje się w ten sposób złożoność problemu i równocześnie zwiększa spójność systemu. W projektowaniu obszernych

i kompleksowych systemów informatycznych jest więc tworzenie podsystemów środkiem do redukcji skomplikowania ich struktury. Jednak teza, że tworzenie podsystemów zmniejsza skomplikowanie systemów, nie daje jeszcze żadnych praktycznych wskazówe. Pytanie jakie kryteria podziału przyjąć pozostaje bez odpowiedzi, podobnie zresztą jak związek między rodzajem struktury a jej efektywnością. Warto nadmienić, że efektywność systemów informatycznych mierzona np. jakością generowanych zbiorów informacji czy stałością i zmiennością tych systemów, jest uwarunkowana nie tylko technologicznymi cechami zastosowanych środków rzeczowych czy jakością informacji, ale również czynnikami natury psychologicznej, socjologicznej i organizacyjnej stworzonej przez użytkowników. Stąd problem doboru kryteriów podziału systemów na podsystemy nie daje się jednoznacznie rozwiązać. Uboczną konsekwencją tego są też trudności rachunku ekonomicznego informacji. Podział systemu na podsystemy nie jest dowolny. Podział taki powinien być przeprowadzany ze względu na spełnione funkcje, a nie np. ze względu na oddzielność przestrzenną. Rzecz jasna, że podziału dokonuje się wielokrotnie dopóty, dopóki dalszy okaże się niemożliwy lub niecelowy z trzech powodów:

- ze względów "pryncypialnych" - to znaczy osiągnięcia dolnej granicy podziału,[1/]
- ze względów metodologicznych - to znaczy, że nie może już mieć sensu /w projektowaniu/ podział danych na znaki lub dalej bity,
- ze względów ekonomicznych - to znaczy, że zysk z dokonania jeszcze jednego podziału byłby mniejszy od dodatkowych kosztów, które trzebaby w związku z tym ponieść.

Krańcowe obiekty, uznane dalej za atomiczne, stanowić będą podstawowe cegiełki konstrukcji systemu i przyrównać je można do systemów cybernetycznych - kombinacyjnych, to jest takich, dla których wartości sygnałów wyjściowych uzależnione są tylko wartościami sygnałów wejściowych /w czasie Δ t/. Ponadto postępowanie podziałowe następuje hierarchicznie "w dół", co ma tą zaletę, że

---

1/ Ta granica może być wyznaczana przez zdolność opisywania danego podsystemu przez przyjęte języki programowania.

unika się pętli /sprzężeń zwrotnych/. Ułatwia to identyfikację systemu. Bowiem w przypadku występowania sprzężeń zwrotnych, obowiązują odmienne kryteria identyfikacji. Efekt oddziaływania identyfikacyjnego wraca z powrotem do elementu określającego, skutkiem czego zatraca się możliwość wyróżnienia stanów elementów sprzężonych. W takich przypadkach muszą istnieć dodatkowe wejścia zewnętrzne,które umożliwiłyby pełną identyfikację.

Metody strukturalne są też z powodzeniem stosowane w programowaniu, ich stosowanie w projektowaniu systemów informatycznych podaje E. Yeurdan. Zna je także ogólna metodologia projektowania.Jest to proces identyfikacji systemu, który należy rozumieć jako proces doboru elementów systemu z wielu potencjalnych elementów. Strukturalizacja problemu pozwala na określenie zakresu i kształtu systemu, czyli przekształcenie koncepcji systemu w określone struktury, umożliwiające realizację koncecji. Strukturę problemu projektowania można uważać za hierarchiczny układ działalności projektowej,w którym działalność na wyższym poziomie mieści w sobie działalność na poziomie niższym. Tedy w metodach strukturalnych projektowania zawiera się także podejście hierarchiczne /"z góry do dołu"mczej niż odwrotnie, choć i to jest możliwe/.

Proces projektowania strukturalnego systemów informatycznych zarządzania wyróżnić może trzy etapy:

- pierwszy z nich polega na sformułowaniu problemu projektowego w postaci zbioru zadań reprezentujących cel /wiązkę celów/ projektowanego systemu - nazwać go można specyfikacją zadań,
- etap drugi to strukturalizacja właściwa problemu projektowanego ; obejmuje organizację /działanie/ polegającą na grupowaniu zadań dotyczących różnych aspektów projektowanego systemu, tak aby tworzyły one podgrupy /podsystemy/ dogodne dla dalszego projektowania, programowania itd. - nazwiemy go organizacją zadań,
- trzeci etap jest procesem syntezy, w którym problem projektowy jest najlepiej spełniony i poszczególne zadania projektowe przedstawione w postaci rozwiązania - nazwiemy go spełnieniem rozwiązania.

Pozostaje otwartym pytanie jak pogodzić projektowanie strukturalne SIZ z projektowaniem podług struktur. Częste utożsamianie tych dwóch podejść może prowadzić do nieporozumienia.

Aby uniknąć tego mankamentu, przyjmujemy, że system informatyczny zarządzania dla celów projektowania można podzielić na cztery podstawowe struktury:

- funkcjonalną,
- informacyjną,
- techniczną,
- przestrzenną,

ich projektowanie zaś jest zawsze innym spojrzeniem /uwzględnianiem odmiennego aspektu/ na ten sam system. I choć wyróżnienie tych struktur można uznać za pierwszy podział przedsięwzięcia w podejściu strukturalnym, to projektowanie podług struktur takim wcale być nie musi. Można projektować SIZ podług struktur zupełnie tradycyjnie. Wydaje się więc, że projektowanie podług struktur jest metaformułą początkującą proces projektowania SIZ i stanowi jeden z możliwych /a jak się wydaje i systemowych/ sposobów logicznego projektowania założonych informatycznych systemów zarządzania. Zaś rozmieszczenie problemów projektowych wedle jakiegoś określonego schematu, wybranego przez projektanta w celu rozwiązania stojącego przed nim zadania projektowego, świadczy o przyjęciu określonej perspektywy, która umożliwia najważniejsze dostrzeżenie tego, co uprzednio pozostawało w cieniu. Ponadto ma to też znaczenie dla rozwoju samej metodologii projektowania SIZ, już choćby poprzez wynajdywanie i doskonalenie szczegółowych sposobów projektowania dla poszczególnych struktur SIZ.

Przyjmiemy dalej następujące definicje obowiązujące dla projektowania poszczególnych struktur; i tak:

- projektowanie struktury funkcjonalnej SIZ, to organizowanie systemu zadań przetwarzania danych,
- projektowanie struktury informacyjnej SIZ, to organizowanie systemu danych,

- projektowanie struktury technicznej SIZ, to organizowanie systemu liczącego,
- projektowanie struktury przestrzennej SIZ, to organizowanie systemu komunikacyjnego.

Definicje te w możliwie najkrótszej formie wyrażają istotę problemów występujących w projektowaniu struktur systemu.

2. Projektowanie struktury funkcjonalnej SIZ

W projektowaniu struktury funkcjonalnej rozwiązuje się kwestię konceptualizacji systemu informatycznego. Elementami tworzącymi strukturę są zadania przetwarzania danych - wynikiem zaś sieci podsystemów funkcjonalnych. Istnieje tu ponadto zbiór wymagań stawianych przed systemem jak i zbiór form /wyników/ , które mogą być zaakceptowane przez użytkowników. Procedury projektowe operują tymi zbiorami w celu spełnienia wymagań i znalezienia formy, która przedstawiałaby zadawalające rozwiązanie problemu projektowego. Proces poszukiwania właściwej formy systemu informatycznego spełniającej wymagania jest celem projektowania struktury funkcjonalnej. W swej istocie jest to proces heurystyczny.

Z tak sformułowanego celu projektowania struktury funkcjonalnej wynika, że zbiór wymagań jest dany i reprezentuje cel systemu informatycznego zarządzania. Istnieją zasadnicze dwa źródła wymagań stawianych przed systemem, uzależnione tylko sposobem podejścia. Pierwszym źródłem wymagań są sponsorzy, drugim może być sam system projektujący. Sposób dochodzenia zaś do spełnienia zbioru wymagań może być diagnostyczny lub prognostyczny i jest to niezależne od źródeł ich pochodzenia. Zbiór wymagań ogólnych nazywać będziemy zbiorem zadań. Zbiór wymagań specjalnych, jak się przekonamy, zmniejsza liczbę wymiarów przestrzeni rezultatów, antycypując decyzje dotyczące formy zakończonego projektu. System projektujący posiada pełną swobodę poruszania się w tej części przestrzeni rezultatów, która nie jest

ograniczona wymaganiami specjalnymi, co pozwala na indywidualizację rozwiązań systemu, o ile nie preferuje się i powiela szablonów. Ograniczenia nałożone na formy można sklasyfikować następujące:

- wartości progowe, określające granice między akceptowanymi i nieakceptowanymi przez użytkownika czy np. system liczący formami,
- wartości szczegółowe, definiujące wąską klasę rezultatów do których należą akceptowane formy,
- wartości kluczowe, które określają maksymalne lub minimalne wybrane charakterystyki akceptowanych form.

Organizacja wymagań w projektowaniu struktury funkcjonalnej sprowadza się do odpowiedzi na pytanie: jak połączyć poszczególne zadania przetwarzania danych w większe segmenty zadaniowe - w podsystemy funkcjonalne? Powstaje tu złożony zakres problemowy o strukturze podobnej do labiryntu. Związane z tym procesy wyszukiwania, oceny i selekcji są realizowane w bardzo pojemnych i wielostronnych obszarach problemowych, przeplecionych różnorodnymi decyzjami, zmierzającymi do ustalenia:

- stopnia segmentacji - ile podsystemów funkcjonalnych utworzyć?
- struktury podziału - ile zadań do podsystemów przydzielić?
- formy realizacji - jakie zadania przydzielić poszczególnym podsystemom?.

Tworzenie podsystemów funkcjonalnych może przebiegać drogą:

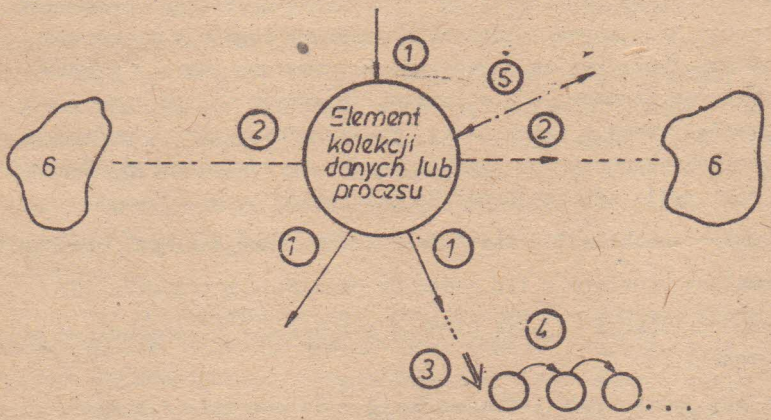
- syntezy zadań opartej na uwzględnieniu cech,
- syntezy zadań opartej na uwzględnieniu współzależności.

3. Projektowanie struktury informacyjnej SIZ

Projektowany system zadań przetwarzania danych /o którym pisano w poprzednim punkcie/ zapewnia identyfikację zadań przetwarzania danych w następującym układzie;

- podstawowym elementem jest zadanie opisane jak niżej:
  - nazwa zadania,
  - cechy zadania,
  - algorytm,
- zadania są powiązane kanałami, w których określone:
  - nazwę kanału,
  - kolekcję danych przysłaną z zadania do zadania.

W projektowaniu struktury informacyjnej zasadniczą rolę odgrywa kolekcja danych - organizowana w system danych. Element kolekcji danych jest tu podstawowym elementem strukturalnym, widzianym przez projektowanie strukturalne jako:



Poszczególne linie oznaczają typ związków /relacji/ występujących podczas tworzenia systemu danych:

1. oznacza związek hierarchiczny, z jakich elementów, w postępowaniu "z góry do dołu" dana kolekcja się składa i/lub w skład jakich elementów wchodzi /"z dołu do góry"/,
2. oznacza przepływ, przechodzenie jednej kolekcji w drugą w wyniku działania procesów przetwarzania danych, z tym że

elementami skrajnymi /6/ są użytkownicy tj. dawcy i biorcy informacji,

3. oznacza pojedyncze dane związane logicznie ze sobą /4/ w jakiś sposób /określony mianem struktur danych/,

5. oznacza związek logiczno-funkcjonalny pomiędzy pewnymi elementami /danymi/ kolekcji danych jednego rodzaju.

Podział kolekcji danych przedstawia się następująco:

- kolekcja danych wejściowych z dalszym podziałem na grupy danych i dane elementowe,
- kolekcja danych wyjściowych z dalszym podziałem na grupy danych i dane elementowe,
- kolekcja danych stałych /zbiory i podzbiory/ z dalszym podziałem na:
  - rekordy logiczne,
  - grupy danych,
  - dane elementarne.

Przepływ informacji w strukturze informacyjnej realizowany jest poprzez procesy przetwarzania danych działając na podzbiorach kolekcji danych. To zagadnienie ujęte jest w projekcie technologii przetwarzania, który można powiedzieć jest relacją " kalendarza " systemu, procesów przetwarzania, kolekcji danych, zdarzeń i warunków.

Kalendarz zawiera przedziały czasowe [1/] istotne dla odzwierciedlenia zdarzeń zachodzących w SIZ; wyróżnić można trzy typy zdarzeń:

- zdarzenia zewnętrzne zachodzące w systemie rzeczywistych, a oddziaływujące na SIZ poprzez napływ danych źródłowych do systemu, zapytań itp.,
- zdarzenia przypadkowe związane jednak ze zdarzeniami zewnętrznymi, jak np. wystąpienie błędów w kolekcji danych,
- zdarzenia ze stałym krokiem, których powodem wystąpienia jest

---

1/ Przedziały czasowe dyktowane są poziomem rozdzielczości w jakim odzwierciedlany jest przez SIZ system rzeczywisty.

sam upływ czasu, np. sporządzanie przez SIZ bilansu raz w tygodniu, czy też emisja rachunków telefonicznych dnia 25-go każdego miesiąca.

Zdarzenia występują przy spełnieniu określonych warunków, powodując w danym przedziale czasowym uruchomienie procesów przetwarzania danych na pewnych podzbiorach kolekcji danych.

W projektowaniu struktury informacyjnej główna uwaga zwrócona jest ku organizowaniu kolekcji danych, tak aby ich treść i forma informacji spełniała oczekiwania użytkownika. Równocześnie obowiązuje w projektowaniu struktury informacyjnej większa dbałość o szczegóły - głębszy poziom szczegółowości i precyzji.

Specyficznymi obiektami opisywanymi w strukturze informacyjnej są:

1° Należące do otoczenia komórki organizacyjne użytkownika systemu informatycznego zarządzania lub też inne systemy:
   - dawcy informacji /zespołu danych elementarnych/,
   - biercy informacji - użytkownicy sensu stricto.

2° Kolekcja danych wejściowych /wejścia/ - informacje wprowadzane /dostarczane/ do SIZ przez dawców.

3° Kolekcja danych wyjściowych /wyjścia/ - informacje wyprowadzane przez SIZ do użytkowników - bierców.

4° Kolekcja danych stałych - informacja stale utrzymane w SIZ,

5° Technologia przetwarzania danych - czynności, procesy, algorytmy,

6° Przedziały czasowe w jakich realizowana jest technologia przetwarzania danych,

7° Relacje - powiązania /stosunki/ wszystkich danych i informacji.

8° Parametry ilościowe struktury informacyjnej,

9° Sytuacje decyzyjne.

10° Warunki zdarzeń i zdarzenia.

Bliższa charakterystyka przedstawionych terminów może być przedstawiona jak poniżej.

D a w c a m i i n f o r m a c j i są komórki organizacyjne użytkownika SIZ, które emitują d a n e ź r ó d ł o w e /lub elementarne/ dotyczące zdarzeń występujących w obiekcie. Dawcami informacji mogą być także obiekty leżące poza obiektem, dla którego projektuje się system. Przykładem może być rynek, centrum, ogólnie inne systemy, które w jakikolwiek sposób wiążą się z interesującym nas obiektem. Dane te mogą być dostarczone w sposób jawny lub też pozyskiwane w jakikolwiek dostępny sposób. Innymi słowy dane źródłowe napływające do SIZ są d o s t a r c z a n e lub p o b i e r a n e [1]. Dane mogą być wreszcie "tworzone", gdy w żadem sposób nie mogą być dostarczone lub pobrane. W takim przypadku są one hipotetyczne - wynikające z wiedzy ogólnej, pewnych założeń czy nawet domysłów.

B i o r c a m i i n f o r m a c j i są komórki organizacyjne wyrażające zapotrzebowanie na określonego rodzaju informacje - p r z e t w o r z o n e. Podobnie informacje mogą być dostarczone na zewnątrz do innych systemów w sposób rozmyślny lub też "wyciekać" z systemu w sposób niezamierzony.

Istnieją trzy rodzaje k o l e k c j i d a n y c h. Podział ten wynika z ich umiejscowienia w strukturze informacyjnej. Stąd wpierw definicja ogólna. "Kolekcją danych" jest słownik wszystkich występujących w systemie danych, które reprezentują istotne cechy zdarzeń. Odróżniamy kolekcje danych od ich wystąpień, gdzie dane są już reprezentowane przez nazwy i wartości - - jakie przebierają. Wyróżniamy:

- kolekcję danych wejściowych,
- kolekcję danych stałych przechowywanych i przetwarzanych /dalej/ w systemie,
- kolekcję danych wyjściowych.

1/ Pobieranie danych może się odbywać bez wiedzy dawcy lub nawet w pewnych przypadkach wbrew dawcy drogą wywiadu gospodarczego. Sposoby takiego zdobywania danych nie leżą jednak w naturze pracy, która nie jest scenariuszem filmu sensacyjnego. Biorąc pod uwagę, że SIZ może być przedmiotem penetracji - istnieje problem projektowy systemowego zabezpieczenia informacji.

Same nazwy kolekcji dalsze objaśniają ich charakter i przeznaczenie. Każdą zaś kolekcję możemy scharakteryzować podając /projektując/: strukturę, zawartość, reprezentację /nośnik fizyczny danych/ częstotliwość powstawania, sposób obliczania czy wykorzystywania. Choć wartość użytkową mają informacje zebrane w kolekcji danych wyjściowych - to jednak centralnym problemem, z punktu widzenia projektowania jest kolekcja danych przechowywanych i przetwarzanych w systemie.
Istnienie gradacji kolekcji danych /zbiory, podzbiory, układy, grupy .../ między danymi lub ich zbiorami związki strukturalne lub funkcjonalne, w których odwzorowuje się wszelkie powiązania i stosunki między danymi.

T e c h n o l o g i a p r z e t w a r z a n i a d a n y c h to wszelkie czynności, procesy i algorytmy dzięki którym wykorzystując dane wejściowe i przechowywane w systemie uzyskuje się akceptowane informacje wynikowe. Technologia jest realizowana w pewnych p r z e d z i a ł a c h c z a s o w y c h i powtarzana cyklicznie według przyjętej w systemie skali czasu /kalendarz systemu/.

Parametry, sytuacje, warunki dotyczą pewnych stanów jakie przejmują kolekcje danych w wyniku procesów.

Opisanie przez projektanta poszczególnych elementów składowych struktury informacyjnej, choć zawiera zagadnienia powiązań, nie wyczerpuje całego jej obrazu. Istnieje tu konieczność uwzględnienia:

- głębokości opisu /szczegółowości/,
- punktu widzenia.

G ł ę b o k o ś ć opisu wynika z niemożności percepcji w tym samym momencie i we wszystkich szczegółach całości opisu. Stąd konieczność stopniowania opisu na kilku poziomach szczegółowości. W zależności od predyspozycji i jakości problemu można tu stosować podejście top-down lub botton np.
Dobrą techniką, za pomocą której można swobodnie dokumentować

strukturę informacyjną jest HIPO. W szczególności można powiedzieć, że technika HIPO jest narzędziem niezbędnym w projektowaniu strukturalnym oraz zapewnia ona dyscyplinę dokumentowania projektu w sposób niemal idealny.

P u n k t w i d z e n i a na projektowanie struktury informacyjnej wynika z przyjęcia określonej teorii systemowej /ogólnej/ za podstawową. Jeżeli przyjmie się za punkt widzenia związany z językiem PSL można uzyskać porządek opisu wg następujących zakresów:

- przepływu informacji,
- konstrukcji struktury informacyjnej,
- struktur danych,
- przetwarzania danych,
- charakterystyk ilościowo-czasowych,
- sytuacji decyzyjnych,
- własności systemu.

Jest to w rzeczywistości propozycja konspektu dokumentacji projektowej struktury informacyjnej systemu.

Przyjmując określoną głębokość opisu struktury informacyjnej i punkt widzenia tworzy się łącznie jej d o k u m e n t a c j ę p r o j e k t o w ą, która ujmuje c a ł o ś ć zaprojektowanej struktury informacyjnej. Z tego właśnie względu dokumentacja projektowa struktury informacyjnej musi spełniać określone warunki, które pozwalają na łatwość jej analizowania. Są to:

- zrozumiałość dokumentacji,
- precyzyjność dokumentacji,
- spójność dokumentacji,
- zupełność dokumentacji,
- poprawność dokumentacji,
- łatwność modyfikacji dokumentacji.

Zrozumiałość dokumentacji oznacza, że zastosowano do czytania łatwy format, materiał jest przedstawiony w sposób zro-

zumiały. Zastosowano standardy i szablony. Ustalono pojęcia podstawowe.

Precyzyjność dokumentacji oznacza, że zawiera ona jasno zdefiniowaną terminologię, tak że przedstawione określenia są interpretowane w sposób jednoznaczny.

Dokumentacja spójna nie zawiera zdań wzajemnie sprzecznych lub niekonsekwentnych. W szczególności kontekst w jakim występują poszczególne pojęcia i obiekty struktury jest trwały i definiowany przez projektanta.

Zupełność dokumentacji oznacza, że opis jest przedstawiony w sposób wystarczająco szczegółowy, tak że nie jest koniecznym zasięganie dodatkowych informacji dla pełnego jej zrozumienia. Wszystkie konieczne szczegóły są dostępne w dokumentacji.

Dokumentacją poprawną jest taka, w której wszystkie określenia pojęć oraz związków między nimi są prawdziwe i aktualne. Wiąże się to też z postulatem łatwości modyfikacji dokumentacji.

Łatwość modyfikacji dokumentacji wiąże się ponadto z koniecznością zapewnienia r o z w o j u systemu. Łatwą do modyfikacji jest taka dokumentacja, w której każda pozycja musi być dostępna, jeśli konieczna jest jej zmiana i to w różnych układach. Ponadto dokumentacja musi zapewniać możliwość dodawania narastająco informacji w miarę ich powstawania przez różnych projektantów. Pozwala to poznawać system na kolejnych poziomach projektowania i analizę opisu w różnych przekrojach. Każdy projektant powinien mieć dostęp do całego aktualnego opisu, aby mógł wnieść poprawki do swojej części i zgłaszać swoje uwagi do pozostałych części. Pozwala to sprawdzać zgodność opisu części z całością projektu.

4. Projektowanie struktury technicznej SIZ

W projektowaniu struktury technicznej SIZ tworzy się system liczący, który jest zbiorem wzajemnie połączonych urządzeń przetwarzania informacji, mogących w oparciu o wyposażenie programowe realizować zadania systemu, poprzez wykonywanie logicznych i arytmetycznych operacji na danych. Wśród operacji na danych wyróżniać się będzie:

- przetwarzanie, jako zmiana jednego rodzaju danych na inny, zgodnie z określonymi regułami. Reguły przetwarzania są zapisane w postaci odpowiednich algorytmów przetwarzania,
- przesyłanie, jako zmiana współrzędnych przestrzennych danych poprzez transmisję danych,
- przechowywanie, jako zmiana współrzędnych czasowych danych,
- kodowanie i dekodowanie, jako zmiana znaków na sygnały i odwrotnie,
- identyfikacja, jako określenie danych na podstawie analizy znaków.

W wyborze systemu liczącego, niezależnie od stosowanej metody szczegółowej możliwe są w zasadzie dwa podejścia:

- podejście syntetyczne,
- podejście analizujące.

W podejściu syntetycznym projektant dobiera do pewnego schematu systemu liczącego elementy go tworzące, a każdy element podlega ocenie, w sensie spełniania przypisanych mu zadań.

W podejściu analizującym ocenie podlegają całe konfiguracje/schematy/ systemu liczącego, które wariantowo dopuszcza się do realizacji zadań rozpatrywanego SIZ.

W istocie rzeczy powinniśmy mieć do czynienia z jednym podejściem, acz dwustopniowym, w którym najpierw dobieramy elementy do różnych konfiguracji, a potem rozpatrujemy warianty biorąc pod uwagę takie czynnika jak:

- sprawność systemu liczącego,
- skuteczność systemu liczącego,
- niezawodność systemu liczącego,
- trwałość systemu liczącego,
- wydajność systemu liczącego,
- koszty instalacji i eksploatacji.

Reasumując, można stwierdzić, że projektowanie systemu liczącego daje się sprowadzić do problemu przyporządkowania zasobów systemu liczącego do zadań SIZ.

5. Projektowanie struktury przestrzennej SIZ.

Projektowanie struktury przestrzennej SIZ nazwano tworzeniem systemu komunikacyjnego, gdyż w istocie chodzi o optymalne rozmieszczenie elementów SIZ w środowisku i zapewnienie połączeń między nimi, które są niczym innym jak drogami przesyłania danych. Samą zaś w sobie struktura przestrzenna jest grafem, którego węzły mają być:

- źródłami danych,
- ujęciami danych,
- punktami koncentracji danych,
- punktami przechowywania danych,
- punktami przetwarzania danych,
- punktami dyspozycji danych,
- ujęciami danych,
- punktami wykorzystywania danych.

Punkty te są połączone kanałmi przesyłania danych. Punktami, w których system informatyczny, poprzez strukturę przestrzenną, styka się z obiektem - miejscami styku, są dwa rodzaje punktów:

- źródła danych,
- punkty wykorzystywania danych.

Kojarzymy je z komórkami organizacyjnymi, które zostały opisane w strukturze informacyjnej. Źródła danych są stanowiskami

emisji danych źródłowych dla SIZ przez obiekt lub jego otoczenie, które następnie będą w określony sposób, kanałami przesyłania danych dostarczane do SIZ. Punkty wykorzystywania danych są natomiast punktami do których dane w postaci przetwarzanej mają być dostarczone. Zarówno źródła danych, jak i punkty wykorzystywania danych nie wchodzą w skład SIZ i w projektowaniu struktury przestrzennej są ustalone. W projektowaniu struktury przestrzennej działanie systemu projektującego może przebiegać według następujących etapów:

- identyfikacja źródeł danych i punktów wykorzystywania,
- wyodrębnienie pozostałych punktów,
- ustalenie połączeń,
- optymalizacja struktury.

Jest to w zasadzie problem wyznaczania układu komunikacyjnego dla danego systemu informacji.

6. Podsumowanie

Zmienność projektowania poszczególnych struktur jako funkcji czasu jest podstawową cechą procesu projektowania. Należy tu też zauważyć, że niezmienność kolejności w czasie jest szczególnym przypadkiem zmienności w czasie.
Procedura projektowa z biegiem czasu przechodzi z jednej struktury do innej /pozwala określić w jaki sposób projektowanie struktur zmienia się w czasie/. Wprowadzenie tego objaśnienia, choć mąci uporządkowany obraz projektowania systemu jest konieczne dla uzmysłowienia niezdeterminowanego charakteru kolejności projektowania struktur systemu. W takim projektowaniu powstaje istotny problem kierowania i formalnego,a szczególnie merytorycznego projektem. Modele sieciowe, za pomocą których można opisywać kierowanie projektami mają alternatywne struktury, uwzględniają ryzyko rozumiane jako ekonomiczny skutek niepewności. Studium metodyczne planowania i kontroli złożonych przedsięwzięć projektowych przedstawia E. Ignasiak w pracy

"Optymalne struktury projektów." Jednak warto tu wzrócić uwagę, że wspomagane komputerowo projektowanie w niebagatelnej swej części obejmuje zagadnienia postępu prac projektowych, nadzorując w ten sposób pracę systemu projektującego. Zasygnalizowanie spraw kierowania projektem pozwala zwrócić w końcu uwagę na zagadnienie, które z rozmysłem i konsekwentnie pominięto. Jest to problem podziału prac projektowych na etapy typu: założenia projektowe, projekt merytoryczny, projekt techniczny itd., ze szczególnym rozważaniem zawartości wyodrębnionych etapów głównych. W integralnym projektowaniu, podział na etapy, ich szczegółowa zawartość, może być tylko pomocniczy i pozostawiony do swobodnej decyzji systemu projektującego.

Andrzej Pełech [1]

NIEKTÓRE PROBLEMY ZASTOSOWAŃ W KRAJU SYMULACYJNYCH GIER DECYZYJNYCH

Wstęp

Adresatami tej pracy są informatycy zainteresowani problemami zarządzania bądź kierownicy zainteresowani informatyką i symulacją zarazem - w zakresie wychodzącym poza bieżące sprawy zawodowe.

Przedmiotem tu rozważanym są symulacyjne gry decyzyjne zwłaszcza zaś kierownicze gry komputerowe. Nie da się jednak sensownie omówić zastosować gier nie zajmując się symulacją w ogóle, a także symulacją często algorytmiczną /w praktyce: komputerową/ oraz symulacją "ręczną". Wyboru problemów i ich naświetlenia starałem się dokonywać z punktu widzenia użyteczności społecznej /przyjmując, że i gospodarcza użyteczność się w tym mieści, jako podrzędna/. Do poważnych ograniczeń wyboru należało słabe dotąd rozpoznanie wielu problemów /znajomość licznych sprowadza się do zaledwie zarysowanych pytań/.

Tekst ma układ niejednolity. Jest to w zasadzie lista problemów i faktów, opatrzonych komentarzami o nader różnej obszerności, zgrupowanych według trzech tematów: /1/ możliwości /i spodziewane korzyści/, /2/ bariery, /3/ zagrożenia. Dużo miejsca zajmuje rozważenie tego czym właściwie jest symulacja - niezbędne dla rzetelnego rozpatrzenia jej aktualnego i potencjalnego czy przyszłego znaczenia /także i tylko

1/ Instytut Organizacji i Zarządzania Politechniki Wrocławskiej

t a k ż e dla nas, w kraju/. Część tekstu zawiera bardzo szkicowy i fragmentaryczny opis "stanu sztuki" symulacji w Polsce.

Całość traktuję jako punkt zaczepienia do dyskusji lub jako punkt wyjścia do dokładniejszych studiów.

Praca zawiera sporo opinii /poglądów/ zarówno specjalistów jak i generalistów, w tym wielu z krajów, gdzie symulacja jest bardziej niż u nas rozwinięta. Źródła starałem się jednak ograniczyć do polskojęzycznych, odstępując od tego w kilku przypadkach. Źródła oryginalne podane są w bibliografiach zawartych w większości powołanych pozycji.

To co zostało tu przedstawione w niewielkim stopniu należy do wyników pracy autora niniejszego opracowania, natomiast w ogromnym pochodzi z prac ludzi zajmujących się symulacją w świecie zwłaszcza w Polsce. Praca ta powstała pod przemożnym wpływem dyskusji w Szkole Symulacji Systemów Gospodarczych, skupiającej corocznie krajowych "symulogów". Oczywiście Kolegów ze Szkoły nie obciążają w żadnym stopniu wady tej pracy. Nie zamierzam Im też przypisywać tych poglądów - - tu wyrażonych - których by nie podzielali.

Popularność i realia

Symulacja staje się w kraju ostatnio modna. Jest ona eksponowana - jako warte pokazania narzędzie - w prasie, w tym w ściśle profesjonalnych czasopismach ekonomicznych. Oto niektóre przykłady: 1/

- Reklama usług świadczonych za pomocą komputerowego s y m u l a t o r a systemu finansowego ukazała się na początku 1983 roku w "Zarządzaniu" /przypuszczalnie pierwsza tego rodzaju reklama w Polsce/.

1/ Przykłady te zostały w większości uprzednio powołane w przedmowie do prac VI Szkoły Symulacji Systemów Gospodarczych - Miłków '83, którą napisałem wspólnie z Krzysztofem Nowakiem.

- Wypowiedział się prof. Czesław Borowski, przewodniczący Konsultacyjnej Rady Gospodarczej: "... należy mieć świadomość i stale pamiętać, że w toku dotychczasowego wdrażania reformy popełniono liczne i często poważne błędy wynikające z wadliwego rozpoznania rzeczywistości i zaniechania rachunków s y m u l a c y j n y c h oraz niecierpliwości decydentów ...". /"Życie Gospodarcze, nr 27 z 3.07.1983 r., p.1; podkr. moje/.

- Pomocał się minister rolnictwa i gospodarki żywnościowej /ówczesny/ Stanisław Zięba: "... Przeprowadziliśmy rachunek s y m u l a c y j n y, uwzględniający różne warianty projekcji. Na tej podstawie dokonaliśmy wyboru ..." /"Życie Gospodarcze", nr 30 z 24.07.1983 r., p. 3; podkr. moje/.

- Symulacja pojawiła się - jako główny instrument - na pierwszej stronie "Życia Gospodarczego" /w artykule "Wariant refinansowania" Joanny Kotowicz, Zdzisława Sadowskiego i Andrzeja Szeworskiego; "ŻG", nr 32, z 7.08.1983 r./: "Badania prognostyczne przeprowadzone za pomocą modelu s y m u l a c y j n e g o MODO zapoczątkowane zostały w roku 1978 ..." /podkr. moje/.

- W Ministerstwie Pracy, Płac i Spraw Socjalnych odbyła się kierownicza gra symulacyjna, której tematem było wprowadzenie obowiązkowego pośrednictwa pracy /"Ster-83"/. /Kowalska 1983/.

- W "Przeglądzie Technicznym", a więc piśmie o tematyce szerszej niż "Życia Gospodarczego" i - jak się zdaje - bardziej odeń rozpowszechnionym, zaczęły ukazywać się wyniki badań symulacyjnych nad przedsiębiorstwem krajowym "w stylu reformy" sygnowane przez spółkę autorską Jan Szumiłak i Bogusław Wąsik[1] /np. "PT", nr 17 z 24.04.1983 r., pp. 16-17/.

1/ Drugi - po długiej przerwie - numer "Wektorów" - pisma Polskiego Towarzystwa Ekonomicznego - zawiera artykuł tejże spółki autorskiej na temat symulacyjnych badań związku zatrudnienia i zakłóceń w dostawach materiałów. /"W", nr 2, listopad 1983, pp. 53-56/.

- W popularnym "Przekroju" ukazała się dość obszerna notatka pt. "Referent w asteroidzie" /"P", nr 1921, 4.04.1982, p.12/ o masowych symulacyjnych grach maszynowych /"automatach"/.

Przytoczone przypadki dotyczą - z jednym wyjątkiem - symulacji związywanej z gospodarką, a tylko dwa gier decyzyjnych /zresztą skrajnie różnych/. Jeden przypadek dotyczy społecznego problemu symulacji. W niektórych z przytoczonych przypadków słowo "symulacja" ma - być może - tak szerokie znaczenie, że zaciera się jego sens jako wyróżnika metody /w każdym razie w oczach pewnej grupy ludzi zajmujących się symulacją/. Nie jest to w tym miejscu istotne, ponieważ przytoczyłem symptomy mody: rozpowszechnienia w kraju pojęcia "symulacja" nie zaś przykłady do dyskusji terminologicznej czy metodologicznej [1].

Całkowite pominięcie sygnałów o symulacją /bądź używaniu pojęcia "symulacja"/ w technice i w biologii wynika z odmienności spraw [illegible] dziedzin od spraw społeczno-gospodarczych. Pominąłem też - z innych względów - zastosowania wojskowe.

Obecne rozpowszechnienie słów "symulacja" czy "gra" / w sensie "... symulacyjna"/ [2] oceniane na podstawie prasy, a zwłaszcza koncentracja tych słów w przykładach powyżej, mogą stwarzać wrażenie intensywności zainteresowań symulacją w kraju i poniekąd sugerować rozległość używania u nas symulacji. W rzeczywistości bardzo dalego od tego. Spostrzeżenia prof. Bobrowskiego /cokolwiek rozumiałby On pod słowem "symulacyjny"/: "... błędy wynikające z /.../ zaniechania ..." dominują

1/ Rozpowszechnione w środowisku ekonomistów znaczenie słowa "symulacja" /np. wg Klcina /1982, p.108// jest dalekie od przyjmowanego w pracach "symulogów" /np. /Evans et al 1973, pp. 12-29/, /Świtalski 1982//.

2/ Odmiennym od znaczenia słowa "gra" wprowadzonego do teorii organizacji przez Crozier'a i Friedberg'a /1982, pp.90-122/ /oryg.: 1977/, /rozpowszechniane w Kraju przez Koźmińskiego i Zawiślaka, np. /1979, pp. 65-78//.

nad pozostałymi przykładami. Można też to spostrzeżenie bez ryzyka omyłki uogólnić na sprawy nie związane bezpośrednio z próbami reformowania gospodarki, a i tak nie wyczerpuje to niewykorzystanych u nas dotąd okazji użycia symulacji. Osiem lat temu Jerzy Żukrowski zakończył swój felieton w "Innowacjach" następująco: "Skoro więc, jak z tego wszystkiego wynika, dzięki zabiegom symulacyjnym można określać skutki rzeczywistych działań, oszczędzić wiele środków materialnych, a nawet uniknąć śmierci, należałoby chyba upowszechnić stosowanie symulacji we wszystkich dziedzinach życia." /"PT-I", nr 51-52, 1976, p. 45/ [1] Tytuł felietonu brzmiał: "O potrzebie symulacji" - co pozostaje aktualne mimo wielkiego wysiłku i wartościowych wyników sporej liczby ludzi w Polsce, którzy dostrzegli i docenili możliwości symulacji, zaczynając gdzieś od drugiej połowy lat sześćdziesiątych.

Korzenie i luk

Warto zwrócić uwagę na następujące fakty:

- Pierwszą g o s p o d a r c z ą grą była najprawdopodobniej "Top Management Decision Simulation" z 1956 roku.[2] Pierwszą podobną grę rozegrano u nas w początku lat siedemdziesiątych. Była to "GRAKAP".[3] /Nb. jej historia, a zwłaszcza jej związek z tzw. "reformą WOG-owską", są warte skrupulatnego odtworzenia/.

1/ Dodając, w ostatnim zdaniu: "Oczywiście symulacji w dobrym znaczeniu tego słowa".

2/ Zamówiona przez American Management Association. Wiadomości o wcześniejszych grach c y w i l n y c h /ale nie gospodarczych/ w /Nowak 1983a/.

3/ W ówczesnym Instytucie Doskonalenia Kadr Kierowniczych Administracji Państwowej.

- W latach 1769 - 76 w Szkole Rycerskiej Stanisława Augusta Poniatowskiego stosowane gry wojenne / operacyjne wg obecnych pojęć/.[1/] W latach 1765 - 68 w Szkole Inżynierii Wojskowej w Mézières we Francji stosowano symulację w nauczaniu /używane tam nazwy "simulacre"/.[2/]

- Jay W. Forrester opublikował w roku 1958 swą koncepcję "Industrial Dynamics"[3/] - z której wyrosła szeroko rozpowszechniona później metoda badań symulacyjnych o wielkiej sile intelektualnego oddziaływania /"System Dynamics"/. W latach 1973-74 wykonano wg tej metody badania dynamiki zapasów materiałów w gospodarce remontowej dla Zakładów Chemicznych w Oświęcimiu.[4/]

Przytoczone powyżej fakty to wybrane a po części izolowane epizody z historii symulacji. Ale epizody ważne. Przyjrzenie się ich datom wyda się, być może, interesujące. Same w sobie mogą one - ewentualnie - posłużyć krzepiącemu wsparciu się na tradycji. Ślady prowadzą tu w starożytność i to - między innymi - poprzez Polskę: w książce Jana Bakałowicza z 1771 roku /"Czynności wojenne"/ jest opis szkolenia za pomocą gry wojennej /wg obecnej terminologii/ przypisany Filopojmenowi, strategowi Związku Achajskiego /253-183 p.n.e./ - 2 200 lat temu![5/] Szczegół ten przytaczam także dlatego, że świeżo upieczeni - na fali mody - propagatorzy symulacji reklamują ją jako Nowoczesność, przeoczając to, że starożytna trwałość może być znacznie lepszym argumentem /zwłaszcza u nas, obecnie, po bolesnym wypadnię-

---

1/ Przypadek "Oelsnitz" wg /Nolewajka 1983/. Jest zachowana dokumentacja gry z roku szkolnego 1769/70.

2/ Przypadek "Bousmard" wg /Nolewajka 1983/. Symulacja w Mézières mogła być używana zarówno wcześniej jak też później i wiele wskazuje, że tak było.

3/ W: Harvard Busines Review, July-August 1958. Podręcznik w trzy lata później.

4/ O badaniach dla Oświęcimia relacja jednego z autorów: /Wąsik 1977, pp. 87-88, 139/. Artykuł zapoznawczy o "SD" innego z autorów tych badań, R. Łukaszewicza, ukazał się w nr 3 Problemów Organizacji w 1971 r. Trzeci autor: M. Żebrowski.

5/ Szczegóły patrz /Bratkowski 1977/, gdzie także inne epizody z historii symulacji w wojskowości z XVIII - XIX wieku /pp. 75-76, 103, 122-135, 370-374, 407/.

ciu z infostrady/. Nie mówiąc już, że ta Nowoczesność to nieprawda /choć zapewne symulacja, gry zwłaszcza bywają, zdarza się - - osobistymi odkryciami/.

Czym jest symulacja?

Nie chodzi tu o definicję lecz o uchwycenie idei /"istoty rzeczy"/ symulacji - rozpatrywanej ze względu na jej społeczne skutki /użyteczność - jakkolwiek by była rozumiana - jest tylko jednym z ich ujęć/ - bez zatrzymywania się nad warsztatowymi problemami symulegów.

Proponuję tu dokonanie kilku spostrzeżeń na podstawie poglądów z rozmaitych punktów widzenia, zaczerpniętych z nader różnych źródeł. Dalej spróbuję wydobyć z nich materiał do zarysowania pewnego uogólnienia.

A. Harvey M. Wagner w przedostatnim rozdziale swej opasłej książki o badaniach operacyjnych /1980/ omawia symulację. Pierwszy podrozdział tego rozdziału tytułuje: "G d y w s z y s t k i e m e t o d y z a w o d z ą ...". A dalej w tekście: "...obecnie i w dającej się przewidzieć przyszłości, podejścia, które rozważaliśmy w poprzednich rozdziałach nie będą w stanie zapewnić kompletnej analizy problemów decyzyjnych z zakresu zarządzania dotyczących:

I. Wyboru polityki inwestycyjnej w planowaniu strategicznym /.../.

II. Określenia warunków działania w planowaniu operacyjnym /.../.

III. Ustalenia reguł funkcjonowania i sprzężeń zwrotnych między systemem informatycyjnym a operatywnym podejmowaniem decyzji." /pp. 956-958/.

B. Lawrence R. Klein na początku rozdziału o symulacji w "Wykładach z ekonometrii" /1982/: "Symulację rozumiemy jako rozwiązanie układu równań reprezentujących model /.../ W języku równań różniczkowych symulacja jest całką z systemu, liczoną

przy ustalonych warunkach początkowych." /p.108/ /L.R.K. pisze to w kontekście modeli gospodarki narodowej/.

C. Robert Jungk w obszernym eseju o człowieku i o społeczeństwach współczesnego przełomu epok odwołuje się do następującego przykładu:[1/]

Dla rozwoju amerykańskiej polityki zagranicznej może nie bez znaczenia był fakt, że młody profesor Harvardu Henry K i s s i n g e r /podkr. moje/ w latach sześćdziesiątych bardzo aktywnie uczestniczył w wynajdywaniu i przeprowadzaniu politycznych gier futurologicznych. Nagłe zmiany kursu, jak jego późniejsza polityka chińska, były nie do pomyślenia w politycznych realiach okresu zimnej wojny, natomiast w symulowanej rzeczywistości gier planistycznych, w których Kissinger brał udział, była to zupełnie normalna rzecz nawet w szczytowym momencie wojującego antykomunizmu. Szczególnie owocne i pobudzające fantazję bywają gry, w których traktuje się historię jako labolatorium, a znanych faktów historycznych używa się jako materiału doświadczalnego. W grze odtwarza się więc pewne wydarzenia, z tym że w punkcie decydującym zmienia się je, a nawet całkowicie stawia na głowie: Napoleon wygrywa bitwę pod Waterloo, Islam podbija Europę, wynalazek sztuki drukarskiej pojawia się sto lat wcześniej czy sto lat później, Lenin żyje trzydzieści lat dłużej /.../... Na pierwszy rzut oka tego rodzaju tematy mogą zdawać się bezsensowne, nawet wariackie, zresztą często tak o nich mówiono.

Ale nie ulega wątpliwości, że uczestnicy tych ćwiczeń z czasem nabierają niesłychanego wyczucia przeoczonych bądź alternatywnych szans. Ogarniają szeroką skalę politycznych możliwości, rozwijają ostrzejsze spojrzenie na czynniki normalnie nie rozważane, a mające swoje znaczenie dla przyszłości, wyrabiają sobie znaczną elastyczność duchową, wyrażającą się w rosnącej zdolności do rozwiązywania problemów i stawiania konstruktywnych propozycji."

1/ [Jungk 1981, pp. 147-148], Oryg.: 1973. Tamże inne o symulacji: pp. 137-156, 350-356.

Tytuł podrozdziału z którego ten cytat: "Gdyby Lenin żył trzydzieści lat dłużej". A tytuł poprzedniego: "Fabryka myśli p r o b l e m ó w s p o ł e c z n y c h". /podkr. moje/.

D. Tytuły, które nadaje Joël de Rosnay częściom swej "próby wizji globalnej" 1/ są doskonałymi wręcz ich systezami:

"Modele i symulacja: podstawowe narzędzia" /rozdział w części II: "Rewolucja systemowa - nowa kultura"/.

"Podstawowe metody: autoinstrukcja i symulacja" /rozdział w części VI: "Wartości i edukacja"/. Skąd fragment:

"G r y s y m u l a c y j n e. Gdy pedagogiczne, wspomagane lub nie przez komputer, stanowią jedną z podstawowych metod edukacji systemowej /.../ od gier wykorzystujących karty aż do gier w przedsiębiorstwa".

E. Z notatki w "Przekroju" /wspomnianej na początku/: "Wielkie salony gier były do niedawna odwiedzane głównie przez uczniów szkół średnich /.../ Ostatnio zaczęły się tam pojawiać tłumy urzędników z okolicznych biur i instytucji, którzy przerwę obiadową spędzają przed /.../ automatami o nazwie Asteroid. Za jedyne 25 centów automat zmienia przeciętnego kasjera w banku w nieustraszonego kapitana kosmicznego pojazdu, prowadzącego własną bohaterską wojnę gwiazd.

Asteroidy zrobiły błyskawiczną karierę /.../ ponad 7 000 automatów zainstalowano pod arkadami ulic, w barach, na lotniskach i w holach hoteli w wielu krajach świata. Pierwsze duże maszyny zainstalowane właśnie w Los Angeles przed kilku miesiącami, kosztowały po 2,5 tysiąca dolarów - koszt ten zwrócił mu się przed upływem tygodnia! /.../

Miłośnicy asteroidów twierdzą, że gra pozwala im powrócić do pracy z uczuciem satysfakcji. Jak powiedział jeden z urzędników Nareszcie człowiek ma poczucie, że zależą od niego jakieś

1/ /Rosnay 1982, pp. 121-126, 324-327/,/Oryg.: 1975/. Tamże inne o symulacji: pp. 101-136 m.in. znakomity wykład idei "dynamiki systemów" Forrestera.

decyzje . /et/ [1]

F. Stanisław Lem: "... wędrować bez skafandra i maski tlenowej po Księżycu, zdobywać na czele oddanej drużyny w dzwoniącej zbroi średniowieczne grody /.../ z rąk króla szwedzkiego przyjmować nagrodę Nobla, kochać ze wzajemnością Mme de Pompadour /.../ czuć jak wyrastają mu olbrzymie orle skrzydła, latać albo znów stać się rybą i pędzić życie wśród raf koralowych; jako ogromny żarłacz mknąć z otwartą paszczą na stada ofiar /.../ zjadać ich ze smakiem i trawić w spokojnym zakątku podwodnej swojej pieczary."

To fragment odpowiedzi na uprzednie pytanie: "Co może przeżywać człowiek podłączony do fantomatycznego generatora?".

Dalej Lem szkicuje ogólną ideę działania nieistniejącej /"narazie"/ "maszyny fantomatycznej" podłączanej do mózgu/ i przeprasza za "melodramatyczny ton wykładu"/.

I dalej o fantomatyce: "Można przy jej pomocy stwarzać sytuacje treningowe i szkolące najwyższej próby /.../. Będzie fantomatyka prawdziwym błogosławieństwem dla niwidomych /.../ chorych /.../ Powstaną grupy zapiekłych jej przeciwników, wielbicieli autentyczności /.../ może się też oczywiście stać prawdziwą groźbą, plagą społeczną." [2]

G. "O problematyce symulacji jako narzędzia władzy człowieka nad człowiekiem pisał Robert Bogusław /.../ ukazuje on uzależnienie /.../ od: - klienta zlecającego symulację /.../; - projektantów systemów i programistów /.../; - producentów komputerów /.../." [3]

1/ /Referent 1982/. Moje własne obserwacje z sesji Szkoły Symulacji Systemów Gospodarczych - Miłków'83 i z nieco późniejszego seminarium firmy Hewlett-Packard w Warszawie potwierdzają wiadomość /et/ i pozwalają poszerzyć tezę o uleganiu namiętności do gier poza populację urzędników - na "symulogów" /przeważnie naukowców/ i na informatyków.

2/ /Lem 1967, pp. 259-260, 277-278/, Rozdział "Fantomologia" /pp. 254-307/.

3/ Wg /Jungk 1981, p. 353/.

H. Odwołam się teraz do życiowego doświadczenia Czytelnika: Proszę zacytować siebie samego na swój własny użytek: proszę przypomnieć sobie przeżywane lub /i/ obserwowane:

1° Zabawy dziecięce: chłopców w Indian lub w wojnę; dziewcząt w układanie lalki do snu etc., w aktualnie modnych idoli sportu czy rozrywki.[1/]

2° Wszystkie te czynności, które robione są na próbę - aby się ich nauczyć, przyzwyczaić do nich.

3° Marzenia. Sny.

Czym zatem jest symulacja?

- Ostatnią deską ratunku w badaniach operacyjnych /Wagner;, A/
- Sposobem na rozwiązanie układu równań różniczkowych /Klein; B/
- Szansą - jedną z najważniejszych - zrozumienie świata, w którym żyjemy, przygotowania zmian a także przygotowania się na zmiany /Jungk, Rosnay; C,D/
- Metodą nauczania i - równocześnie - kształtowania osobowości. Metodą w pewnym zakresie jedyną. /Jungk, Rosnay, Lem; C.D.F/
- Metodą treningu i /zwłaszcza/ sprawdzania przydatności do określonych zadań /Jungk, Rosnay, Lem; C.D.F/.
- Zastępczym działaniem, ostatnią szansą na pozorowaną choćby samorealizację człowieka /Lem, /et/; E,F/.
- Narkotykiem, plagą odciągającą człowieka od realnego świata, od aktywnego, świadomego wpływu na własne i losy innych /Lem, et; E.F/.
- Instrumentem wprowadzania w błąd, narzucania woli / Bogusław, Lem; G,F/.
- Zabawą, odprężeniem.

1/ W bardziej zdyscyplinowanej terminologii i z odwołaniem się do źródeł, przedstawiono tę sprawę na szerszym tle antropologii kulturowej w /Metera et al 1983, p. 5-7/ - pierwszym - jak się wydaje polskim podręczniku symulacyjnych gier kierowniczych.

Wszystko powyższe jest, lub wygląda na to, że może stać się prawdą. Dotyczy to symulacji w ogóle, ale zwłaszcza gier symulacyjnych /a w tym i kierowniczych/.

Wydaje się, że w poszukiwaniu "istoty symulacji" warto zatrzymać się nad czterema sprawami, które sformułuję w postaci tez:

1. Symulacja jest naturalnym zachowaniem człowieka /dawno ukształtowanym/. Jako gra jest nim dosłownie. Jest częścią sposobu życia: wyobrażenia, przeżywania, uczenia się i myślenia refleksyjnego. Symulacja ma wiele wspólnego z powieścią, sztuką teatralną, filmem. Z symulacją jest jak z mówieniem prozą. Używają jej wszyscy choć nieliczni ją nazywają a jeszcze mniej liczni zastanawiają się nad nią.

2. Symulacja jest niezastąpionym narzędziem przedstawiania świata, wtedy gdy nie potrafimy sprowadzić go do niewielu bardzo prostych zależności dobrze wyodrębnionych obiektów /jak np. w klasycznej astronomii Newtona/ lub do statystycznego ujęcia wielkiej liczby jednakowych obiektów /np. w termodynamice/ - a tak jest wtedy gdy rozważyć się chce np. przedsiębiorstwo czy nawet gospodarkę narodową. 1/

3. Symulacja jako zjawisko społeczne może być zarówno pożytkiem /dobrodziejstwem nawet/ jak i zagrożeniem. Wizje odległej przyszłości znajdują tu potwierdzenie w symptomach już teraz obserwalnych /"asteroidy" - fakt; 1982, "fantomat" - przewidywany w 1967/. 2/

Pojmowanie symulacji jest bardzo różne. Od rozpatrywania jej jako zjawiska społecznego o dużym znaczeniu i metody o rozległych zastosowaniach - do zawężenia w ciasne ramy usprawnień modelowania matematycznego czy nawet obliczeń numerycznych.

1/ Patrz /Weinberg 1979, pp. 25-43/. Ponadto tamże podrozdział: "Symulacja - biała skrzynka" /pp.175-185/.

2/ Oczywiście "Asteroid" nie jest realizacją "Fantomatu". Ale jest podobnym narzędziem, którego używanie wywołuje niektóre ze skutków przewidywanych dla fantomatu. Podobieństwo "A" - "F" obydwa są symulatorami.

Być może najważniejsze jest tu połączenie naturalności symulacji - jej zgodności z właściwościami człowieka ujawniającymi się od dzieciństwa,[1] z jej dostosowaniem do odzorowywania "systemów średnich" [2] - które nastręczają ludziom największych kłopotów. W tym też połączeniu zawarte są - wydaje się, po równo - sposobności pożytków i zagrożeń jakie niesie używanie symulacji. Źródła tych pożytków i zagrożeń tkwią - oczywiście - w ludziach i w ich systemach społecznych. Symulacja jest tylko tym co każde ostre narzędzie. Gier dotyczy to - można sądzić - w większym stopniu niż symulacji algorytmicznej.

Pominąłem tu - zgodnie z wcześniejszym założeniem - sprawy warsztatowe "symulogów", a także pominąłem gospodarczą stronę symulacji, która jest na Zachodzie swoistym biznesem - przez który bez wątpienia wiodą ważne kanały kształtujące jej zastosowania /vide opinia Bogusława powyżej/. Pominięcie tych spraw warsztatowych, które bezpośrednio wiążą się dostępnym zakresem sensownych zastosowań /m.in. ale na czele: problem podobieństwa modeli symulacyjnych do tego co mają one przedstawiać i związany z nim problem tego co przy pomocy modeli symulacyjnych możemy się właściwie dowiedzieć/ jest na pewno poważną wadą. Nie jest jednak możliwe przedstawienie tego tutaj ze względu na potrzebną ilość miejsca. Zainteresowanych odsyłam do artykułów /Rzońca 1980/, /Rzońca 1981/, /Radosiński 1983/, /Wander 1983a/ i do książki /Metera et al 1983/.

Elementy krajowych możliwości

Założenia

Przedstawię tu pewien pogląd na krajowe możliwości używania symulacyjnych gier kierowniczych. Tłem będą możliwości świadomego, zorganizowanego używania w kraju symulacji. Będę dążył do przedstawienia możliwości obecnych, tak aby uwzględnić ich znaczenie dla przyszłości.

1/ I nie tylko człowieka: Proszę zwrócić uwagę na zabawy małych kociąt czy psich szczeniaków /i na to co sądzą o tym etolodzy/.

2/ /Weinberg 1979, p. 25-43/ używa tu określenia "system o średniej liczności".

Przyjmuję, że najważniejszym elementem możliwości są umiejętności, doświadczenie ludzi oraz to co je wiąże i uruchamia, a co w sposób niedoskonały bywa nazywane motywacjami. Ponadto: wiadomości utrwalone na piśmie, zwłaszcza ogólnie dostępne. Wybór takiego punktu widzenia przesądza o traktowaniu środków materialnych i instytucji jako ograniczeń - nie oznaczając bynajmniej umniejszenia wielkiej mocy ich znaczenia w rozpatrywanej sprawie. Traktowane jako ograniczenia znajdą się jednak poza obszarem w tym miejscu omawianym.

Bardzo ważna grupa właściwości nazwana umownie "motywacjami" jest trudno uchwytna, nie było - o ile wiem - systematycznie u nas badana /u "symulogów" i podobnych/, toteż nie będę się w tej sprawie bezpośrednio wypowiadał. Poprzestanę na uwadze, że w pewnym stopniu - aczkolwiek niezadowalającym - można motywację ocenić na podstawie dokonań /zwłaszcza biorąc pod uwagę rozporządzalne środki/. Tak więc w końcu możliwości pokażę wyłącznie poprzez prace i ich wyniki. Proponuję tu przyjęcie domniemania /choćby ograniczonego/, że jeżeli coś dało się komuś zrobić raz to uda się to powtórzyć a nawet zrobić lepiej. Jednym z argumentów na rzecz przyjęcia tego domniemania teraz jest względna krótkość historii używania symulacji do celów "cywilnych" u nas "ok. 25 lat/ i fakt, że wszyscy/w każdym razie mnie osobiście znani/ specjaliści krajowi w tej dziedzinie żyją aktualnie, a większość może być aktywna zawodowo jeszcze przez nie mniej niż 20 lat.

Przyjmuję, że mam odpowiedzieć na pytanie o następującej postaci: "Trzeba przeprowadzić symulacyjne badania obiektu klasy X za pomocą gry /nauczać o nim lub posługiwania się nim, przewidzieć jego zachowanie - w tym zakładając też: jeżeli zrobione zostanie A, to ... /. Czy są w kraju ludzie, którzy umieją to zrobić? Nauczyć się tego? Rozumieć taką pracę /i jej wyniki/?" - Pytań tych nie będę dalej powtarzał. Podam wyłącznie odpowiedzi. W pierwszej części będą one zawierały nazwy wykonanych gier, niekiedy, z krótkimi komentarzami, wskazane źródła informacji i roku opracowania. Jeżeli w wyróżnionej klasie

"X" brak wykonanych u nas /dla nas/ gier lub są bardzo nieliczne a istnieją komputerowe /ściśle algorytmiczne/ modele, które przypuszczalnie mogą stanowić punkt zaczepienia do sporządzenia gier to podam nazwy tych modeli. W drugiej części podam te dokonania, które dotyczą wszystkich lub większości klasy "X" obiektów, a więc: wykształcenie ludzi i opublikowanie wiadomości: książek /zwłaszcza podręczników/ i artykułów przeglądowo-zapoznawczych.

Nie ma to być katalog krajowych gier mający ambicje kompletności. Jest to pewna reprezentacja - zniekształcona tym, że zapewne nie mam w niektórych klasach gier pełnego rozeznania oraz pominięciami: - W klasach, w których gry są nieliczne podaję wszystkie sobie znane gry - choćbym wiadomości o nich miał niewiele, natomiast w klasach obficiej obsadzonych pomijam gry, o których nie mam pewnych wiadomości oraz te, dla których nie mogę wskazać źródła w miarę dostępnego dla Czytelnika, a także gry będące powtórzeniami czy niewielkimi modyfikacjami innych gier[1/]. Zainteresowanych dokładniejszymi wiadomościami odsyłam do odpowiednich, podanych tu źródeł [2/].

Przyjmuję następujące klasy obiektów /"X"/ mogącymi być przedmiotem symulacji, grami:

- środowisko kraju,
- kraj /społeczeństwo i środowisko gospodarcze i przyrodnicze/,
- gospodarka narodowa,
- gałęzie i przekroje gospodarki,
- regiony,

1/ Nie uwzględniłem też gier odtwarzających problemy "zasilaniowe", typowe dla organizacji produkcji, dyspozycji ruchem itp. oraz tych obiektów nazywanych przez autorów "grami", które nie mają cech symulacji w sensie na ogół przyjętym wśród zajmujących się nią /patrz np. cyt. wyżej /Świtalski 1982//.

2/ Katalog gier kierowniczych krajowych /i państw sąsiednich/ znajduje się w rozdziałach 6, 7 i 8 pracy /Kunicka et al 1980/. Liczne informacje o grach są w pracach Szkół Symulacji Systemów Gospodarczych z lat 1978-83 /lista zbiorów tych prac jest na końcu spisu literatury/. Ponadto można uzyskać wiadomości w Centrum Informacyjnym Gier Ekonomicznych w Jenie /adres na końcu/.

- społeczności lokalne, mikroregiony,
- grupy przedsiębiorstw,
- przedsiębiorstwa i ich przekroje,
- rodzina - gospodarstwo domowe; grupa robocza.

Klasyfikacja ta ma liczne wady wśród których brak jednego kryterium podziału może się rzucać w oczy najbardziej. Jej jedyną rolą jest skrótowe przedstawienie wyników modelarstwa za pomocą gier - co też jest jedynym usprawiedliwieniem.

Wyniki: gry

- Zrobiono G r y odtwarzające
  - Środowisko kraju
    Brak.
    Zaczątki: komputerowe modele symulacyjne krajów RWPG /Rybicki 1983a/.
  - Kraj /społeczeństwo i środowisko materialne/
    Dwie gry, raczej społeczno-polityczne:
    - rozegrana latem 1981 r. i wtedy relacjonowana w "Przeglądzie Technicznym" 1/
    - projekt gry "Jak Polak z Polakiem" przedstawiony przez A. Meterę na sesji szkoły symulacji Trzebieszowice'81 /brak publ., wzmianka w /Trzebieszowice '81 - S, p. 37/.
  - Gospodarka Narodowa
    Trzy gry:
    - Duża komputerowa /Pawilno-Pacewicz 1980b/, opr. 1979, ujmująca gospodarkę w 27 gałęziach. Służyła do badań nad gospodarką Polski lat 1976-85 /m.in. prognozy na 1980-85/. Wyniki były wielokrotnie referowane i demonstrowane /m.in. w Komisji Planowania/. Projekt wersji dydaktycznej /Pawilno-Pacewicz 1980a/.

1/ Nie posiadam, niestety, dokładnych danych bibl.

- Ręczna formularzowa, dydaktyczna "Planowanie w gospodarce zdecentralizowanej" /Jurek,Sledziński 1977/, opr. ok. 1976 r.
- Ręczna - klockowa dydaktyczno-oświatowa "GRAEK" Wąsik 1983.

Koncepcje i projekty: budowy i korzystania z modeli gospodarki narodowej w kraju Pailno-Pacewicz 1980c.

Zaplecze i tło: komputerowe modele:

- gospodarki DMR - 1 Rybicki 1980, opr. 1979
- wahań tempa wzrostu gospodarki Pawilno-Pacewicz 1982.

• Gałęzie i przekroje gospodarki

Brak.

Potencjalne zaplecze:

- komputerowy model dynamiki procesów rynkowych /Niemyjska 1981/,
- komputerowy model wpływu systemu paliwowo-energetycznego na rozwój gospodarczy /Rybicki 1983/,
- komputerowy model systemu obrotu materiałowo-technicznego /Południk, Sołtan 1979/.

• Regiony /i ich przekroje/

Województwo:

- Gra odtwarzająca rozwój województwa Jeleniogórskiego oparta na CROSS-IMPACT ANALYSIS./Mackowiak 1980/ /opr.: ?/. M.in. wykonane prognozę do 2 000 r.

Przekroje grupy województw:

- "Praca-82" gra odtwarzająca zatrudnienie na tle gospodarki w grupie hipotetycznych województw /Maj 1982/ /opr.: ?/.
- "Ster-83" gra odtwarzająca obowiązkowe pośrednictwo pracy na tle gospodarki w grupie hipotetycznych województw /Kowalska 1983/ /opr.: ?/

Całościowo region: brak. Jest projekt, koncepcja /Świtalski 1982c/.

- Społeczności lokalne, mikroregiony

Brak.

Zaczątki: Projekt komputerowego modelu gminy /Dąbkowski, Nastaj 1980/ /opr.: ?/.

- Grupy przedsiębiorstw

- "GRAKAP", komputerowa gra związana z koncepcją WOG-ów /systemu finansowego głównie/, oparta na grze "NYU Management Game" /Uniwersytetu Nowojorskiego , zaprojektowana przez specjalistów stamtąd /częściowo?/, wg krajowych założeń, opr. nie później niż w 1974 r. [1] przyp. 1972 ?. Używana w dydaktyce. /Planistyczna?/.

- "METRAKT", wersja ręczna i komputerowa, dydaktyczna /Kisiel, Metera 1976/, /Kisiel 1977/.

- Rodzina komputerowych, dydaktycznych gier: "GRAJSIL" /modyfikacja gry ICL [2]/, "GRASIL II" / opr. 1974 /Wołk 1974/, GRAPLAN" /opr. 1975/ /Wołk 1976/, "GRAFES" /Wołk 1975/ /"modyfikacja WOG"/, "GRAFES 3" /Wołk 1983/ /wg zasad reformy gospodarczej na rok 1983/ /opr. 1982/, wersja "GRAJSIL""z uwzględnieniem elementów reformy" Santarek 1983.

- Gry ręczne, formularzowe: "Bitwa o plan" /pokazana 1979 r. w TV/ i "Producenci - Hurtownicy" /Nowak 1980/, /Nowak 1982/. Używana w dydaktyce.

- Gra ręczna, planszowa "VAN D'OR-4" i "VAN D'OR-K" /Wander 1984a/ /oprac. "4":1983/ [3] odtwarzająca rozwój produkcji i sprzedaży wyrobów na ograniczonym rynku. Używane w dydaktyce.

1/ "W roku akademickim 1974-1975 gra ta była rozgrywana z powodzeniem ze studentami /.../ IOiK UW i PAN." /Kisiel 1976/.

2/ Autorzy: J. Empacher, A. Kisiel, A. Kubacka, W. Sokołowski "Modyfikacja gry ICL stosowanej w IDKKAP od 1969 r." /Wołk 1976/.

3/ Pierwotna wersja oprac. 1982.

- Gra ręczna, formularzowa "DOSTAWCA-ODBIORCA" /Pełech 1978/. /opr. 1977/. Używana w dydaktyce.

- Gry planszowe ręczne pokrewne "MONOPOLY" /lub wzorowana na niej/, "EUROBUSINESS" /w sprzedaży zauważyłem w 1983/, gry "Przeglądu Technicznego - Innowacji": "SUKCES" /1979/ i "DYREKTORZY" /1977/.

- "ASKO", gra komputerowa odtwarzająca klasyczny oligopol z Zachodu, wzorowana na grze "Melnikeff" z Central London Polytechnic" /Skoczylas 1982/. /nie eksploatowana/.

- Handlowa gra komputerowa w sprzedaży samochodów w gospodarce o cechach zachodniej /Ziółkowski et al 1983/.

- Gra planszowa "Legat" odtwarzająca pewne elementy działalności przedsiębiorstw naftowych /Górecki 1983/. /opr. 1982/. Doświadczalna.

- Gra planszowa, ręczna, A-BDL" odtwarzająca sytuację producentów wielkich samolotów pasażerskich na przełomie lat 1982-83/. Doświadczalna. /Pełech 1983/.

- Przedsiębiorstwa i ich przekroje

- "SYMORG", komputerowa gra odtwarzająca działanie przedsiębiorstwa przemysłowego uwzględniająca samodzielność i samefinansowanie wg pewnych wstępnych koncepcji reformy gospodarczej /Wander 1983b/ /opr. 1981/82/. Raz została użyta w konkursie na dyrektora, głównie używana do celów badawczych.

- "KGF-1", dydaktyczna komputerowa gra finansowa odtwarzająca głównie system finansowy przedsiębiorstwa, nowa wersja wg zasad systemu finansowania na rok 1984 /Radosiński, Szczurowski 1983/. /opr. 1982-84/. Używana w dydaktyce 1984.

- Gra ręczna, formularzowa "VAN D'OR-R-84"odtwarzająca przedsiębiorstwo /produkcja, zaopatrywanie materiałowe, system finansowy/ wg zasad na rok 1984 /Wandor 1984 b/ /opr. 1983/ /84/. Obecnie przenoszona na komputer ZX-80. Używana w dydaktyce 1983/84.

- Gra ręczna, formularzowa "Kombinacje Pana Dorobka" odtwarzająca prowadzenie gospodarstwa rolnego /Garbicz et al 1981/. Używana w dydaktyce. W roku akademickim 1980/81 odbyła się rozgrywka tej gry, w której współzawodniczyli studenci SGPiS i Politechniki Wrocławskiej. 1/

- Gry ręczne, formularzowe: "PL-E-1" oraz "Rozmieszczenie" odtwarzające działalność produkcyjną /Nowak 1980/,/Nowak 1982/. Używane w dydaktyce. /opr. 1979?/.

- Gra ręczna, planszowo-formularzowa "Grosz do Grosza" /"GdG"/ odtwarzająca realizację przedsięwzięcia /Pieczykolan A.,Pieczykolan P., 1982/. Używana w dydaktyce 2/. /opr. 1982/.

- Gra ręczna, formularzowa "Dźdżownica" odtwarzająca realizację przedsięwzięcia /Kwiatkowski 1983/ /opr. 1982/. Doświadczalna.

- "OSTROW TUMSKI" - komputerowa dydaktyczna gra odtwarzająca realizację przedsięwzięcia. Projekt /opr. 1981/82/ 3/

• Rodzina - gospodarstwo domowe; grupa robocza:

Brak. Nie mam wiadomości o jakiejkolwiek próbie.

Wyniki: Wykształcenie

Większość osób zajmujących się w kraju symulacją nauczyła się tego sama lub drogą przyuczania. Do pewnych celów - należących do najważniejszych - droga ta wydaje się skuteczna, być może najbardziej. W ten sposób uformowani specjaliści są dziś autorami gier czy też symulatorów czysto komputerowych bądź też autorami rozmaitych prac z dziedziny symulacji. Ich listę można spo-

---

1/ Rozgrywka ta ma obszerną literaturę cytowaną w całości w /Pieczykolan 1983/.

2/ Analiza strategii w tej grze w /Balcerak 1984/ oraz w /Jaworek, Jaśkiewicz 1984/ - ta ostatnia przy pomocy komputerowego symulatora rozgrywki ręcznej gry "GdG". Prototyp "GdG": "Pajęczyna" /oprac. 1982/.

3/ W 1984 wykonane moduły. Jeden z nich opisany w Kwiatkowski 1984. Komunikat o projekcie w /Pełech 1982/.

rządzić bez większych trudności z zadowalającą dokładnością.[1/] Jest to nie mniej niż 121 osób.

Drugą drogą dochodzenia do zajmowania się symulacją i rozumienia symulacji jest systematyczne zorganizowane kształcenie. Mam przy tym na myśli tylko takie kształcenie, w którym "gry"lub "symulacje" /zawierająca gry/ występują w sposób wyodrębniony organizacyjnie jako "przedmiot". O ile wiem, w kraju tylko w Politechnice Wrocławskiej [2/] prowadzony jest przedmiot "modelowanie symulacyjne" /a także przedmiot "gry decyzyjne"/. Pierwszy od roku akademickiego 1977/78 drugi od 1976/77. Przedmiot "modelowanie symulacyjne" studiowało dotąd łącznie około 350 osób. Około 150 osób przebyło "gry decyzyjne" - prowadzone z przerwami. /Z tym, że częściowo te same osoby studiowały obydwa przedmioty/. Co najmniej sześć osób z tych ok. 400 zajmuje się aktualnie zawodowo symulacją lub używa jej jako jednego z głównych narzędzi w zawodowej pracy /trzy prowadzą zajęcia z symulacji i gier/. Z pozostałych co najmniej 9 osób opublikowało po jednej lub więcej pracy z symulacji [3/]. Przypuszczam, że przynajmniej część z tych około 400 osób, które zaznajomiły się tą drogą z symulacją może przez kilka lub kilkanaście lat w przyszłości nadawać się na partnerów /choćby pośrednich/ do współpracy w badaniach symulacyjnych. Z tego znów część może podjąć samą pracę za pomocą symulacji byle w niezbyt odległym czasie, zapewne do kilku lat.

Wyniki: Wiadomości utrwalone na piśmie i dostępne

Przedstawię tu tylko wiadomości utrwalone po polsku. Trudnej i niezwykle ważne sprawy dyfuzji wiadomości zza granicy /obcojęzycznych/ nie będę tu poruszał. Nikłą namiastką będą wzmianki o tłumaczeniach i przeglądach.

1/ Na przykład: korzystając ze spisów uczestników kolejnych Szkół Symulacji Systemów Gospodarczych lub z wyboru publikacji w /Pełech 1984/ można sporządzić listę niepełną ale jednak zdecydowanej większości osób zajmujących się w kraju symulacją i gospodarką zarazem.

2/ W Wydziale Informatyki i Zarządzania.

3/ Niektóre z tych osób są cytowanymi tu autorami.

- Podręczniki gier są dwa krajowe: /Metera et al 1983/ i /Putkiewicz, Ruszczyńska 1983/. Tłumaczeń podręczników brak. Są wzmianki lub rozdziały w niektórych z tłumaczonych książek ale ogólnikowe. Najwięcej w cyt. tu /Jungk 1981/ i /Rosnay 1982/ a ponadto w /Barton 1974/, /Evans et al 1973/ i /Nayler 1975/.

- Artykuły. Zbiory prac Szkół Symulacji Systemów Gospodarczych zawierają ich ponad 20. Prócz tego ukazały się nieliczne w innych pismach. Wydaje się, że ich łączna liczba nie przekracza 40.

- Dokumentacja gier: Na wymienione tu [1] 35 gier istnieje niewątpliwie i może być /przy pewnym wysiłku/ dostępna dokumentacja 26 gier.

- Bibliografia prac po polsku: Przypuszczam, że ta którą zebrałem ostatnio dla swych studentów /Pełech 1984/ jest najobszerniejsza i dość aktualna /do 1983 r./ choć niekompletna.

- Przeglądy typu stan "sztuki": dotyczące symulacji generalnie, wszystkie publikowane w pracach Szkoły Symulacji Systemów Gospodarczych: /Rzeńca 1980/; /Rzeńca 1981/; /Wander 1982/, /Nowakowski 1983/. Wszystkie oparte na bardzo obszernych, trafnie dobranych /reprezentatywnych/ i aktualnych /w czasie opracowywania/ publikacjach zagranicznych.

Pewne uwagi zmierzające do syntezy

W modelowaniu czego mamy w kraju najwięcej doświadczeń ? Proszę spojrzeń na poniższe zestawienie, sporządzone na podstawie wykazu gier powyżej /przy wszystkich wadach tego wykazu o których wspomniałem - nie wydaje się aby przedstawiony obraz zmienił się jakościowo po doprowadzeniu "wykazu" do doskonałej kompletności/:

1/ Nie wątpię, że liczba gier omawianych tu klas jest nieco większa. Być może też osiągalna jest dokumentacja więcej niż 26 gier z 35 wymienionych.

Obiekt	Obraz liczności	Liczba przypadków
- Środowisko kraju :	.	0
- Kraj :	gg	2
- Gospodarka narodowa :	Ggg	3
- Gałąź gospodarki :	...	0
- Region :	ggg	3
- Mikroregion :	.	0
- Grupa przedsiębiorstw :	GGGGGGGGggggggggg	18
- Przedsiębiorstwo :	GGgggggg	9
- Rodzina, gosp. dom. etc.:		0
	G - gra komputerowa g - gra ręczna	35

Jak widać symulacyjne modele growe przedsiębiorstw i grup przedsiębiorstw to około 2/3 ogółu modeli omawianej tu kategorii. Z pozostałych tylko modele gospodarki narodowej mieszczą się bez większych zastrzeżeń w swej klasie. W każdym razie takie i tylko takie modele umiemy robić na pewno. Tworzenie hipotez dlaczego tak jest pozostawię Czytelnikom wraz z ich dyskusją. Zauważę tylko z obowiązku, że powodem na pewno nie są rozmiary modelu czy inaczej rozumiany stopień jego komplikacji [1]. "GRAEK" odtwarzająca gospodarkę jako całość wytrzyma bez wątpienia porównanie co do prostoty z każdą z gier odtwarzających przedsiębiorstwo czy tylko jego przekrój. A przykładowo gra "SYMORG" odtwarzająca tylko przedsiębiorstwo jest, mierząc liczbą zmiennych, nie mniej skomplikowana niż gra Pawilno-Pacewicza odtwarzająca gospodarkę.

Czy są ludzie umiejący sporządzić gry
i posługiwać się nimi?

- Tak! I te spory nadmiar. Ten nadmiar ludzi, którzy otrzymali podstawowe wykształcenie w modelowaniu symulacyjnym sprzyja

1/ Ani też rozmiary obiektu modelowanego.
Ani związek rozmiarów obiektu z rozmiarami modelu.

wprawdzie rozpowszechnianiu się tegoż, ale na te znów liczba tak wykształconych jest za mała /licząc nawet z dyplomantami, doktorantami itp., którzy zajmowali się w swych pracach lub posługiwali symulacją jest to kilkaset osób w kraju/ - szansa na trafienie na takiego "zamrożonego" sprzymierzeńca /przepraszam za zapożyczone, z niemiłego skądinąd, pojęcie/ jest raczej mizerna.

Trzeba też pamiętać, że nieużywane wiadomości i umiejętności zanikają, a brak wyraźnych wyników - choćby tylko obserwowanych ale w swym bezpośrednim otoczeniu - mogą spowodować osłabienie pozytywnego nastawienia a nawet frustrację. Prowadząc zajęcia z "modelowania symulacyjnego" i gry każdego roku spotykam się z kilkorgiem na ogół młodych ludzi, których zainteresowania wykraczają znacznie poza zaliczenie /np. w ostatnich dwóch latach 14 studentów uczestniczyło w nadobowiązkowych seminariach a pięcioro z nich brało udział w Szkołach Symulacji/. Możliwości wykorzystania i rozwijania tych zainteresowań po studiach są obecnie dla nich nikłe. Tym bardziej dotyczy to dalszych kilku osób rocznie, które uzyskują całkiem dobre rozeznanie w symulacji a często i niezłe, choć oczywiście dość surowe jeszcze, umiejętności.

Czy wyniki /doświadczenia itp./ są utrwalone pismem?

- Same modele /gry na ogół tak. Ale już na przykład doświadczenia z eksploatacji gier są dokumentowane i publikowane w znikomym stopniu /znam tylko 5 prac tego rodzaju - wszystkie cytowane w /Pieczykolan 1983/ też należącej do tej grupy/. Dowiedzieć się łatwo "co" ale trudno "jak" i "dlaczego" i "po co".
- Przypuszczalnie dość kompletnie dokumentowane są wyniki teoretyczne w symulacji.

Niektóre bariery

Ograniczę się tu do dwóch przeszkód w czerpaniu pożytków z gier symulacyjnych- nie przypisując zresztą tym przeszkodom wyłącznego związku z symulacją. Przypadki, którymi posłużę się do przedstawienia tych przeszkód zawierają zastosowania decyzyjne.

Dotyczą jednak i innych zastosowań - zwłaszcza dydaktycznych. Potraktuję je ilustracyjnie - mają one tylko zwrócić uwagę na pewne problemy barier nie mniej ważnych niż niewątpliwie dotkliwe przez nas w tej chwili odczuwane przeszkody /też niepomijalnej ważności/: ogólna bieda: brak sprzętu, brak informacji /zagranicznych/, brak pieniędzy. Wnioski pozostawię w większości Czytelnikom.

Przypadek "Mieriežkow" 1/

W styczniu 1941 roku marszałek Mieriežkow /ówczesny szef Sztabu Generalnego ZSRR/ przeprowadził grę wojenną na temat hipotetycznej wojny z Niemcami. Zachodnim Napastnikiem był generał Żukow, a generał Pawłow dowodził Armią Czerwoną. Pawłow bronił się według założeń ówczesnej oficjalnej doktryny wojny z Niemcami /m.in. w ufortyfikowanym rejonie bagien Prypeci/. Żukow przeprowadził trzy zbieżne uderzenia na koncentrację Pawłowa w Grodnie i w Białymstoku niszcząc jego siły i kontynuując uderzenie na Lidę. Według założeń oficjalnej doktryny /przyjętej przez Stalina/ uderzenie niemieckie miało być spodziewane i zatrzymane w pobliżu granicy a dalsza wojna miała się toczyć na terytorium wroga /tj. na ówczesnych ziemiach polskich/. Stalin uznał wyniki gry za całkowicie nie do przyjęcia. Marszałek Mieriežkow został usunięty ze stanowiska szefa Sztabu Generalnego a na jego miejsce został mianowany Żukow.

Wzmiankowana gra była jedną z co najmniej kilku gier na temat wojny z Niemcami, przeprowadzonych w ZSRR w latach 1936-1941. Zapoczątkował je generał Tuchaczewski w Akademii Sztabu Generalnego - m.in. zakładając zaskakujące uderzenie Niemiec - i prowadził w 1937 roku jako gry - cienie gier oficjalnych rozgrywanych w Sztabie Generalnym. Rezultaty gier - cieni były podobne do rezultatów gry Mieriežkowa.

1/ Przypadek ten opisuję na podstawie Brackena /1977/ i Nowaka /1983b/ - który zresztą też jako źródła użył m.in. Brackena.

Niemcy swą ostatnią grę przed wojną z ZSRR rozegrali w grudniu 1940 roku, jako sprawdzenie planu "Barbarossa". Grę prowadził generał Paulus. Z jego sprawozdania z rozgrywki: "Czołowe elementy trzeciej armii pancernej wyszły w rejon na wschód od Lidy" /Bracken 1977, p. 298/. Wcześniej wymienia też Białystok.

Wyniki gier radzieckich i niemieckich pokrywały się w znacznym stopniu. Pokrywały się też z początkowym przebiegiem wojny rozpoczętej w czerwcu 1941 roku.

Ważne i interesujące jest tu pytanie: Dlaczego Stalin zignorował wyniki gier, które jednoznacznie wykazywały wady przyjętej doktryny i opartego na niej planu wojny? Ważniejsze jest jednak stwierdzenie, że tak się właśnie stało. Przy tym należy zauważyć, że nie można tego tłumaczyć /a przynajmniej nie można w całości/ cechami charakteru Stalina czy osobliwością jego pozycji./Bracken 1977 pp. 299-300/ pokazuje inny przypadek /nazywa go "Midway"/, w którym admirał Ugaki ingeruje w przebieg rozgrywki, zmniejszając arbitralnie poniesione w bitwie straty /z dwóch do jednego lotniskowca/ w wyniku czego bitwa została uznana za wygraną. Japończycy bitwę tę - jak wiadomo - przegrali, tracąc cztery lotniskowce.

Przypadek "Commitee of Imperial Defence" 1/

W 1937 roku została rozegrana w Wielkiej Brytamii gra, w której odtworzono lotnicze uderzenie bombowe Niemiec i własną obronę /także "cywilną"/. Był to jeden z 9 etapów prac na ten temat prowadzonych od 1922 do 1939 roku.

Prognoza strat w ciągu pierwszych 60 dni bombardowań była następująca:

- 600 tys. zabitych,
- 1 200 tys. rannych,
- ok. 120 mld funtów strat materialnych.

1/ Jak poprzedni przypadek i ten opisuję na podstawie Brackena /1977/ i Nowaka /1983b/.

Gra oparta była na danych ze Sztabu Sił Powietrznych /Air Staff/, który oceniał niemiecką wydajność bombardowania na 644 ton na dzień. W rzeczywistości szczytowa wydajność niemiecka w Bitwie o Anglię, w 1940 roku, wynosiła 150 t na dzień /przy znacznie mniejszej Luftwaffe w 1937 r. niż w 1940/.

Wyciągnięte z tej gry wnioski spowodowały m.in. następujące skutki /wymieniam kontrastowo/:

- plany masowych pogrzebów w wapnie /Ministerstwo Spraw Wewnętrznych/,
- wydanie dodatkowo jednego miliona formularzy, zgonów przez Ministerstwo Zdrowia,
- panika w okresie kryzysu monachijskiego /150 tys. ludzi ewakuowało się do Walii/.

W rzeczywistości straty choć dotkliwe były w Bitwie o Anglię wielokrotnie niższe niż oszacowane na początku gry.

Istotne jest tu pytanie: jak doszło do tak błędnych wyników? Bracken i Nowak wymieniają m.in. następujące przyczyny:

- Było tylko jedno źródło informacji - dalej już nie kwestionowanych /był nim Sztab Sił Powietrznych/,
- Oceny "śmiertelności" jednostkowej dokonano ekstrapolując wyniki najbardziej niszczycielskich ataków na Londyn w Pierwszej Wojnie Światowej. 1/

Najważniejszym jest tu stwierdzenie zaistnienia błędnej prognozy.

Przypadek "Sher-83"

"Najlepszymi wynikami w tym czasie - i tak już będzie do końca - może poszczycić się województwo północne. Ku zaskocze-

---

1/ Być może siła brytyjskiej obrony powietrznej w Bitwie o Anglię w 1940 r. /myśliwce, artyleria przeciwlotnicza, system dowodzenia zwłaszcza - korzystający z radaru/ została rozbudowana do swych rozmiarów dzięki apokaliptycznej prognozie z omawianej gry?

niu wszystkich pojawiła się tam groźba ... bezrobocia. Do końca gry ekipa ta nie korzystała z możliwości wprowadzenia obowiązkowego pośrednictwa pracy, a mimo to stąd do stanowiska z napisem GUS napłynęły rewelacyjne dane. Dopatrzone się jednak nieprawidłowości statystycznych i sukces stanął pod znakiem zapytania. Jeden z dyrektorów z Katowic, któremu przypadła rola wojewody północnego, powie później, że przyczyną ich sukcesu mogło stać się nie kretactwo, ale założenia przyjęte w trakcie gry, że fabryki zaczną racjonalizować zatrudnienie i oszczędzać energię, a to pozwoli im zwiększyć produkcję szybciej niż przyrastać będzie liczba zatrudnionych. Z większym rozmachem przystąpiono więc do gry, zakładając intensywne sposoby przyrostu produkcji, odrzucając natomiast model może prostszy, ale nie efektywny. Zaskrzeczała rzeczywistość - takich założeń autorzy "Steru" nie uwzględnili, a nadmierny /jak oceniono/ wzrost produkcji "GUS" uznał za przejaw skłonności do naginania statystyki. I tak niekwestionowany faworyt spadł na drugą pozycję ..." /Kowalska 1983/.

Pytanie: dlaczego "niekwestionowany faworyt spadł na drugą pozycję?

- Sądzę, że warto tu rozważyć - z całym respektem dla niewspółmierności wagi spraw - czy czasem Przypadek "Ster -83" nie wykazuje jakichś cech podobieństwa z Przypadkiem "Mierieżkow"? /i wspomnianym przy okazji "Midway"?/

I: - Czy wnioski z obu Przypadków "M" nie mogą być ważne dla gospodarki - mimo niewątpliwych i oczywistych różnic gospodarowania i wojowania?

Pocieszająca będzie w pewnym stopniu każda odpowiedź. Jeżeli byłaby ona wypadkiem: "tak" - to w każdym razie:

- mamy po każdej stronie, coś wspólnego z Japończykami /"GUS" - - Ugaki/,
- istnieje Postęp: bądź co bądź "wojewoda północny" zajął jednak drugie miejsce a za to - mam nadzieję - Odpowiedzialny Za Grę nie podzielił losów Mierieżkowa /nie mówiąc już o Tucha -

czewskim/ - nawet, jak już zastrzegłem, w proporcji rangi spraw.

Uwagi końcowe

Nie chciałbym tym co tu napisałem stwarzać choćby przelotnego wrażenia zajmowania miejsca po sprzedawcach różnych cudownych metod, które kolejno miały nas zbawić. Pisałem oczywiście cały czas o symulacyjnych grach i o symulacji bo taki jest temat tej pracy a nie dlatego, że to pępek świata. Starałem się po prostu pokazać charakter ich właściwości ocenionych według użyteczności. Pisałem także dlatego, że gry uważam za silne narzędzie, które jednak może być użyte zarówno do dobrego jak i do złego. Starałem się też zwrócić uwagę na to, że gra użyta jako pomoc w decydowaniu może dawać bardzo różne wyniki zależne od kompetencji, wysiłku i charakteru ludzi, którzy się nią posługują /i od cech społeczności w której tkwią/. Decydowanie skupia i wyjaskrawia wszystkie cechy gry symulacyjnej jako narzędzia. Wystąpią one jednak również w nauczaniu i w rozrywce.

Rzeczowa wiedza o tym czym są naprawdę gry nie jest wielka, a to co jest znane jest bardzo mało rozpowszechnione. Stąd kształcenie w rozumieniu tego czym jest symulacja i do czego może być użyta jest w perspektywie ważniejsze niż kształcenie specjalistów, którzy będą ją profesjonalnie robić - a w każdym razie nie mniej ważne. Pomyślny symptom daje tu zakiełkowanie tej wiedzy w szkolnictwie podstawowym i średnim /Putkiewicz, Ruszczyńska 1983/. Proszę też jeszcze raz posłuchać człowieka, który miał z symulacją do czynienia na skalę u nas nieznaną. Pisze Jungk /1981, p. 353/:"Boguslaw domaga się zdemokratyzowania symulacji: opinia publiczna powinna uzyskać wpływ na stawiane zadania a ogół powinien nauczyć się rozumienia "zawoalowanej" dotąd techniki symulacji".

Pisząc dla informatyków i dla kierowników zainteresowanych informatyką niewiele miejsca poświęciłem komputerom, językom programowania /"symulacyjnym"/, kłopotom ze sprzętem itp. A także przeciwnie: perspektywom rozwoju komputerów w świecie /świetlanym/ i naszemu zacofaniu /ogromnemu/. Nie uważam, że są to sprawy małej

wagi w ogóle i dla używania symulacji. Uważam nawet wprost przeciwnie. Sądzę jednak, że: 1° większość tego Czytelnicy wiedzą /na pewno lepiej niż ja/, oraz: 2° istotą rzeczy jest wiedzieć "co" i "po co" przed "jak"i "czym". Łatwiej będzie wtedy wykorzystać dobrze najskromniejsze choćby środki.

Literatura

Uwaga: Oznaczenia L'80, T'81, T'82, M'83, S'78 objaśniono na końcu.

Balcerak, A. 1984; Strategia gry "GdG" - propozycja. /w:/ M'83-S, /w druku/.

Barton, R.F. 1974; Wprowadzenie do symulacji i gier. WNT, Warszawa 1974. /oryg. 1970/

Bracken, P. 1977; Unintended Consequences of Strategic Gaming. Simulation and Games, vol 8, Nr 3, 1977, pp. 283-318.

Bratkowski, S. 1977; Z czym do nieśmiertelności. Wyd. "Śląsk", Katowice 1977.

Crozier, M. Friedberg, E. 1982; Człowiek i system - ograniczenia działania zespołowego. PWE, Warszawa 1982.

Dąbkowski, A. Nastaj, E. 1980; Modelowanie systemowe rozwoju społeczno-gospodarczego układów mikroprzestrzennych. /w:/ L'80, pp. 3-21.

Dyrektorzy 1977; Dyrektorzy. /gra: plansza, rekwizyty, instrukcja/ Przegląd Techniczny - Innowacje nr 50/51, 1977 /wkładka/.

Evans, G.E.
Wallace, C.F.
Sutherland, G.L. 1973; Symulacja na maszynach cyfrowych. WNT, Warszawa 1973 /oryg. 1967/

Garbicz, M.
Kaliszyk, J.
Kuszewski, T.
Pawilno-Pacewicz, J. 1981; Nowa gra decyzyjna w nauczaniu programowania matematycznego. /w:/ L'80-S. pp. 11-14.

Górecki, K. 1983; Gra planszowa "Legat" - pozyski - wanie źródeł ropy naftowej. /w:/ T'82-S. pp. 17-20.

Jaworek, A.
Jaśkiewicz, J. 1984; Symulator komputerowy gry "GdG". /w:/ M'83-S. /w druku/.

Jungk, R. 1981; Człowiek tysiąclecia. PIW, Warszawa 1981. /oryg. 1973/.

Jurek, W.
Śledziński, W. 1977; Gra dydaktyczna "Planowanie w gospodarce zdecentralizowanej". Zeszyty Naukowe Akademii Ekonomicznej w Poznaniu, Tom. 29, Poznań 1977. pp. 54-83.

Kisiel, A.W. 1976; Gra kierownicza "GRSKAP" - Instrukcja dla uczestnika gry. Wyd. drugie popr. i uzup. Wyd. IDKKAP, Warszawa, styczeń 1976.

Kisiel, A.W. 1976; Gra kierownicza "METRAKT" - wersja niekomputerowa - Instrukcja dla uczestnika. Wyd. IDKKAP, Warszawa, styczeń 1976,

Kisiel, A.W. 1977; "METRAKT-1" Komputerowa gra kierownicza - Instrukcja dla uczestnika gry. Wyd. IDKKAP, Warszawa, styczeń 1977.

Klein, L.R. 1982; Wykłady z ekonometrii. PWE, Warszawa 1982. /oryg.: maszynopis/

Kowalska, M. 1983; Minus za pośrednictwo. Życie gospodarcze, Vol 38, nr 41, p. 12.

Koźmiński, A.K. Zawiślak, A.M. 1979; Pewność i gra - wstęp do teorii zachowań organizacyjnych. PWE, Warszawa 1979.

Kunicka, A. Metera, A. Pańków, J. Sikorski, A. Wach, T. Widomska, K. 1980; Wykorzystanie gier symulacyjnych w doskonaleniu kadr kierowniczych. IOZ i DK, Warszawa 1980 /maszynopis/

Kwiatkowski, J. 1984; /Symulator zasobów w grze kierowniczej "Ostrów Tumski" - odtwarzającej przedsięwzięcie/. /w:/ M'83-S. /w druku/.

Kwiatkowski, J. 1983; Opis gry kierowniczej "Dżdżownica". /w:/ T'82-S. p. 21-22.

Lem, St. 1967; Summa technologiae. Wyd. 2 poszerzone. Wyd. Literackie, Kraków 1967.

Maćkowiak, A. 1980; Zastosowanie metody prognostycznej - analizy wpływów krzyżowych - w grach decyzyjnych. /w:/ L'80. pp. 53-58.

Maj, D. 1982; Gra w rzeczywistość. Polityka, nr 8/1301/, 10 kwietnia 1982. pp. 1,6

Metera, A. Pańków, J. Wach, T. 1983; Teoretyczne i metodyczne zagadnienia symulacyjnych gier kierowniczych. IOZiDK, Materiały nr 38, Warszawa 1983.

Naylor, T.H. 1975; Modelowanie cyfrowe systemów ekonomicznych.
PWN, Warszawa 1975 /oryg. 1971/.

Niemyjska, E.Z. 1981; Dynamika procesów rynkowych - na przykładzie modelu DMR-PWR.
/w:/ T'81, pp. 49-63.

Nolewajko, R.K. 1982; Rozwój symulacyjnych gier decyzyjnych na przykładach zastosowań wojskowych od połowy wieku XVII do roku 1956.
/w:/ T'82, pp. 235-263.

Nowak, J. 1980; Gry symulacyjne w dydaktyce badań operacyjnych.
/w:/ L'80, pp. 67-88.

Nowak, J. 1982; Ogólna charakterystyka systemów przedmiotowych wybranych gier symulacyjnych do nauczania badań operacyjnych.
/w:/ T'81-S. pp. 23-33.

Nowak, J. 1983a; Pierwsze cywilne gry symulacyjne.
/w:/ T'82-S. pp. 51-53.

Nowak, J. 1983b; O efektywności w praktyce gier symulacyjnych - rozważania wstępne.
/w:/ T'82-S. pp. 55-64.

Nowakowski, M. 1983; Nieokreśloność - rozmyte modele - symulacja.
/w:/ M'83-1, pp. 79-102.

Pawilno-Pacewicz, J. 1980a; Wariantowanie ścieżek wzrostu gospodarczego przy pomocy symulacyjnego modelu makroproporcji /gra kierownicza/. Saminarium: "Symulacja i ekonomiczne gry decyzyjne". Akademia Ekonomiczna w Poznaniu, maj 1980. /maszynopis powiel./.

Pawilno-Pacewicz, J. 1980b; Symulacja wariantów wzrostu gospodarczego Polski w latach 1976 - - 1985.
/w:/ L'80. pp. 109-113.

Pawilno-Pacewicz, J. 1980c; Uwagi w sprawie budowy i wykorzystania modeli gospodarki narodowej.
/w:/ L'80. pp. 89-108.

Pawilno-Pacewicz, J. 1981; Symulacja wahań tempa wzrostu w gospodarce centralnie planowanej.
/w:/ T'81. pp. 73-84.

Pełech, A. 1982; "Ostrów Tumski" - gra kierownicza odtwarzająca realizację przedsięwzięcia - koncepcja i stan prac nad projektem.
/w:/ T'81-S. pp. 77-80.

Pełech, A. 1983; Projekt gry "A-BDL" w rozwój, produkcję i sprzedaż samolotów komunikacyjnych.
/w:/ T'82-S. pp. 87-97.

Pełech, A. 1984; Przewodnik po literaturze o symulacji systemów gospodarczych.
Wrocław 1984. /maszynopis powiel./.

Pieczykolan, A. 1983; Ryzykować znaczy wygrać! /?/ - kolejny fragment analizy pewnej rozrywki gry "Kombinacje Pana Dorobka".
/w:/ T'82-S. pp. 99-108.

Pieczykolan, A.
Pieczykolan, P. 1982; "Grosz do Grosza" - planszowa, symulacyjna gra decyzyjna odwzorowująca realizację przedsięwzięcia - - Instrukcja dla gracza. /wyk. na zlec. SGPiS/
Wrocław 1982. /masz. powiel. do użytku w M'83/.

Południk, K. Sołtan, T. 1979; Symulacja komputerowa systemu obrotu materiałowo-technicznego Centr. Ośr. Bad. Rozw. Gosp. Mat., Warszawa 1979. /dalsze inf. /w:/L'80, pp 145-153/

Putkiewicz, E. Ruszczyński-Schiller, M. 1983; Gry symulacyjne w szkole. Wyd. Szkolne i Pedagogiczne. Warszawa 1983.

Radosiński, E. 1983; Weryfikacja komputerowych modeli symulacyjnych na przykładzie analizy modelu DEF-II. /w:/ T'82-S. pp. 113-140.

Radosiński, E. Szczurowski, L. 1983; Komputerpwa gra finansowa "KGF-1". /w:/ M'83-1. pp. 117-142.

Referent 1982; Referent w Asteroidzie. Przekrój, nr 1921, 4 kwietnia 1982 p. 12.

Rosnay, de J. 1982; Makroskop - próba wizji globalnej. PIW, Warszawa 1975. /oryg. 1975/.

Rybicki, M. 1980; Zastosowanie makroekonomicznego modelu dynamiki systemów /DMR-1/ do symulacji założeń planu na lata 1981-85. /w:/ L'80, pp. 161-168.

Rybicki, M. 1983a; Model symulacyjny ograniczeń rozwoju krajów RWPG /I etap/. /w:/ M'83-1. pp. 143-154.

Rybicki, M. 1983b; Komunikat o realizacji modelu symulacyjnego współzależności między rozwojem gospodarczym a systemem paliwowo-energetycznym. /w:/ M'83-1. pp. 155-157.

Rzeńca, W. 1980; Komputerowa symulacja systemów gospodarczych - dotychczasowy rozwój i stan aktualny.
/w:/ L'80, pp. 185-219.

Rzeńca, W. 1981; Uwagi o standaryzacji symulacji,
/w:/T'81, pp. 135-161.

Santarek, K. 1983; Gra kierownicza "GRAJSIL" i jej realizacja na maszynie ODRA-1305.
/w:/ M'83-1, pp. 175-181.

Skoczylas, A. 1982; Symulacyjna gra decyzyjna "ASKO".
/w:/ T'82, pp. 157-159.

Sukces. 1979; Sukces.
/gra: plansza, rekwizyty, instrukcja/.
Przegląd Techniczny-Innowacje nr 51-52. 1979.

Switalski, W. 1982a; Gry symulacyjne.
/w:/ T'82, pp. 187-205.

Switalski, W. 1982b; Dynamiczne aspekty modelowania.
/w:/ T'82, pp. 207-210.

Switalski, W. 1982c; Założenia symulacyjnego modelu planowania i realizacji polityki regionalnej.
/w:/ T'81-S, pp. 85-106.

Wagner, H.M. 1980; Badania operacyjne - zastosowania w zarządzaniu.
PWE, Warszawa 1980. /oryg. 1969, 1975/.

Wander, B.W. 1982; Metody symulacyjne w teorii organizacji i zarządzania.
/w:/ T'82, pp. 213-234.

Wander, B.W. 1983a; Weryfikacja modeli - problem ciągle aktualny.
/w:/ M'83-2, pp. 207-231.

Wander, B.W. 1983b; Gra decyzyjna "SYMORG" - struktura i możliwości eksplerymentowania. /w:/ T'82-S, pp. 149-175.

Wander, B.W. 1984a; Symulacyjna gra kierownicza "VAN D'OR-K" - instrukcja dla uczestników gry. Wyd. PTE - Dyrekcja Szkolenia Ekonomicznego w Katowicach, Katowice 1984.

Wander, B.W. 1984b; Symulacyjna gra kierownicza "VAN D'OR-R-84" - instrukcja dla uczestników gry. Wyd. PTE-Dyr. Szk. Ekon., Katowice 1984.

Wąsik, B. 1977; Modelowanie dynamiki systemów gospodarczych. Wyd. Akademia Ekonomiczna w Krakowie - Monografie nr 36, Kraków 1977.

Wąsik, B. 1983; "GRAEK" - gra symulująca działalność gospodarczą /wersja podstawowa/. Akademia Ekonomiczna w Krakowie - - Studium Doskonalenia Kadr Kierowniczych, Kraków 1983.

Weinberg, 1979; Myślenie systemowe. WNT, Warszawa 1979.

Wołk, R. 1974; Komputerowa gra kierownicza "GRAJ-SIL II" - instrukcja dla uczestników. IDKKAP, Warszawa 1974.

Wołk, R. 1975; Komputerowa gra kierownicza "GRA-FES" - instrukcja dla uczestników. IDKKAP, Warszawa 1975.

Wołk, R. 1976; Komputerowa gra kierownicza "GRA-PLAN" - instrukcja dla uczestników w wersji 1976. IDKKAP, Warszawa 1976.

Wołk, R. 1983; Gra kierownicza "GRAFES 3",3 wersja na 1983 r. - skrócona instrukcja dla uczestników gry. IOZiDK /Warszawa/ kwiecień 1982.

Ziółkowski, Iwański, C. Stefański, J. 1983; Symulacyjny model konkurencji rynkowej w postaci wieloosobowej gry komputerowej. /w:/ M'83-2. pp. 295-300.

Zbiory prac Szkoły Symulacji Systemów Gospodarczych. Tytuły zbiorów z lat 1980-84 są jednakowe:

"Symulacja systemów gospodarczych"

z dodaniem miejsca sesji Szkoły i dwóch ostatnich cyfr roku. Poniżej nazwy tych miejsc wraz z wydawcami, miejscami i latami wydania wg skrótów stosowanych jako odwołania do źródeł w spisie powyżej. Litera "S" po cyfrach roku oznacza Suplement. Cyfry 1 i 2 po cyfrach roku oznaczają nr-y tomów.

L'80 ; /.../ - Lubachów'80.
Wyd. TNOiK - Wrocław. Wrocław - Gliwice.1980.

L'80-S ; /.../ - Lubachów'80 - Suplement.
Wyd. TNOiK - Wrocław. Gliwice - Wrocław 1981.

T'81 ; /.../ - Trzebieszowice'81.
Wyd. TNOiK - Wrocław, Wrocław - Gliwice 1981.

T'81-S ; /.../ - Trzebieszowice'81 - Suplement.
Wyd. TNOiK - Wrocław, Gliwice - Wrocław 1982.

T'82 ; /.../ - Trzebieszowice'82.
Wyd. TNOiK - Wrocław, Wrocław - Gliwice 1982.

T'82-S ; /.../ - Trzebieszowice'82 - Suplement,
Wyd. TNOiK - Wrocław, Wrocław 1983.

M'83-1 -2 ; /.../ - Miłków'83.
Wyd. TNOiK - Wrocław. Wrocław 1983.

M'83-S ; /.../ - Miłków'83 - Suplement.
Wyd. PTE - Katowice, Katowice /1984/. /w druku/.

Dodatek A: Adres Centrum Informacyjnego Gier Ekonomicznych w Jenie:

Informationszentrale für ökonomishe Spiele Friedrich-Schiller-Universitat Jena
Sektion Wirschftswissenschaften-Okonomishes Labor
DDR-6900 Jena, Schillerstra e 8.

Colette ROLLAND [1/]

AN EXTENSION OF THE THREE LEVEL ARCHITECTURE FOR INFORMATION SYSTEM DESIGN

INTRODUCTION

11. An extension of the ANSI architecture

The three level architecture (conceptual, internal, external) introduced by the ANSI/X3/SPARC group (standards planning and requirements) is now widely accepted as a framework for data base design.

Let us recall the definitions :

"A conceptual schema comprises a unique central description of the various information contents that may be in a data base.

The data base itself might be implemented in anyone of a number of possible ways. Its physical storage structure would be described by an internal schema. Applications programs and end users might access the data in a variety of ways, each described by an external schema."

This architecture is only related to the data analysis. We think that it can be efficiently extended to the data processes. Our scope in this introduction is to generalize the three level modeling to information system design.

Generalized Conceptual Schema

An information system has both structural and behavioural properties. Structure concerns state and static properties. Behaviour refers to state transitions and dynamic properties.

Static aspects correspond to the IS data ; dynamic aspects correspond to IS transactions and programs.

The conceptual data schema is a high level description of the IS static aspects. It defines entities and relationships that are pertinent for the enterprise and have to be represented in the information system.

The data conceptual schema defines the future data base semantic content in a structured way. The data structure is an image of the Universe of discourse component structure. The data values at a point t of time characterize the IS state at this time as an image of the real world state at t.

1/ Professeur, Université PARIS I Sorbonne

The generalized conceptual schema that we propose consists of adding to the data conceptual schema a description of the IS dynamic properties. Not only entities and relationships must be represented but also their time dependent behaviour. Including behavioural aspects in IS conceptual design means to explain why entities and relationships change in time and how they change. Defining the IS behaviour is to define how and why the system state transitions are performed.

"The generalized conceptual schema is a unique global description of the various information contents that may be in an information system and of all the rules that explain their behaviour in time".

The rules complete the data description and explain their time dependent behaviour. The systematic use of the rules guarantees the data consistency.

Generalized internal schema

The data might be implemented in different ways ; the rules themselves might be computerized in different ways :

"the generalized internal schema includes a description of the data physical storage structure and a set of programs allowing the rules performing".

The most current implementation of rules consists of application programs or transactions based on the use of data manipulation languages available on DBMS. But the recent emphasis on artificial intelligence and expert systems suggest other solutions based on logic, semantic networks, languages such as LIPS, PROLOG ...

Let us note also that the programs included in such generalized internal schemas are a sub-set of the application programs usually developed around a data base. They are restricted to data updating processes. Their role is to perform, at given points of time, the IS transitions from one consistent state to a new consistent one.

The three level architecture extension leads to a dichotomy of the set of processes usually programmed on a data base. Part of the processes belongs to the conceptual and internal schemas. They are a necessary complement of the data structure in order to obtain a complete description of the information system static and dynamic invariants. Another part corresponds to end-user manipulations and belongs to external models.

Generalized external schemas

End-user external views might also be generalized.

"An external schema describes the particular view of an IS end-user in terms of data he wants to use as well as in terms of application programs he wants to develop for his specific usage."

The external schema processes are queries, calculations, reports made for a specific purpose. They correspond to particular end-user requirements either systematic or occasional.

They might be statistics, data analysis, aggregative calculations ... and might use editing tools, report writers, graphic tools... But in any case theses processes might not update the global data shared by all the end-users community.

The conceptual and internal schemas have few variations in time. In the contrary external schemas are permanently evolving according to new end-users requirements.

12. Extension consequences

The conceptual schema extension implies an extension of data models. They must provide not only concepts for data structuring but also for rules definitions. They must be able to allow a complete modeling of IS structure and behaviour.

Recent papers introduce concepts, tools and methods for behaviour modeling and integration with their structural counterparts for structure modeling.

The purpose of this paper is to present the state of art on modeling behavioural semantics of information systems.

2 - THE STATE OF ART : A REVIEW OF BEHAVIOURAL MODELING APPROACHES

Researchers concerned with behaviour modeling point out the special role that time seems to play. They point to the need for a realistic real world model to include representations not only for snapshot descriptions of the real world but also for histories, or the evolution of such descriptions over time. With respect to handling of the temporal dimension we can suggest to group modeling methods in three categories :

1. Methods which more or less ignore the time aspect

They are typically most of data modeling methods developed during the seventies. The data base is seen as a finite collection of data which represent the current state of real world entities and relationships. Of course, time can be, in a restricted sense included in the modeling. This is, however, done in an ad-hoc and arbitrary way by the designer by introducing intrinsic time domains in relations or by introducing entity-time associations in binary models.

2. Methods which include the time dimension as a fundamental concept.

They have elaborated modeling concepts for dealing with the definition and referencing of time varying entities and relationships. The data base gives the opportunity to understand the history of real facts. It provides a view of the successive states in the life of entities and relationships.

3. Methods which include concepts to model the state transitions of entities and relationships in time.

Not only they consider the definition and referencing of time varying entities and relationships but they explain the reason of the state transitions and the transitions themselves. The data base must be able not only to give the state of an entity or relationship at any point of time but also to explain why and how the entity or relationship has reached this state. These methods fully address the problem of modeling behavioural semantics of data bases.

The present survey concentrates on the last two categories of approaches.

21. Concepts for time modeling

Methods which include the time dimension as a fundamental concept address two kinds of problems :

1) to define a calendar i.e a set of time references

2) to propose notations for dealing with the referencing or time stamping of time varying entities and relationships.

1. Calendars are often defined in a similar way.
B. Breutmann et al [4] use a language called "CSL" for defining calendar systems. The salient features of this language are as follows : first a sequence of domains called time segment types can be defined which correspond to increasingly coarse time periods such as days, months and years. If there are n domains in the sequence then a sector $(t_k,\ldots t_n)$, $1 \leq k \leq n$, in which t_j belongs to the jth domain $k \leq j \leq n$, represents a generalized point in time e.g june, 1984. Each such point has a beginning and an end, which are elements of the next finer domain. The language provides facilities for defining the beginning and the end of such time points by means of specific procedures for each k.

In M.R. Gustafsson et al [10] a time model consists of an infinite set T of time points and a set TI of time intervals, where $TI = \{ TI_1,\ldots TI_N \}$. It is partially ordered according to a component function, which is total and has the semantics of "part of". If $TI = \{S, MIN, H, D, M, Y\}$ then second : T --> S, minute S --> MIN, ... etc .... The "part of" function holds an inheritence property i.e if X and Y are two time interval types and "part of : X --> Y" holds, then X "inherits" all functions defined for Y. For each application at hand one defines the appropriate time intervals.

S. Leifert [12] introduces in the RFMORA methodology [17] a calendar defined as part of the cartesian product of domains V_i. $V_i = \{ d_i, \mu(D_i) \}$ $1 \leq i \leq n$, where D is a set of integer domains D_i, $1 \leq i \leq n$, M a set of time units such as card (M) = n and μ a mapping from D onto M ($\mu : D \rightarrow M$). A date with order i is the projection $\pi \{ V_i, V_{i+1} \ldots V_n \}$ of the calendar on domains $\{ V_i, V_{i+1} \ldots V_n \}$. Complementarily the language provides operations on the calendar such as date comparison , date decomposition (month (19800504) = 05), date creation and date composition.

A. Olivé [15] uses in the DADES method the life span L of an information system, i.e the time interval in which the system operates or exists. References to a given instant or set of instants of L must be done in a time scale using some time units. Dades defines a set of time units like TIME which refers to a year/month/day/hour/minute/second, DATE which refers to year/month/day...etc. Dades includes a set of functions to convert references from a time unit to another one. For example the function MONTH gives the month of a time or a month of a date. In order to specify some particular instants of L, Dades allows the definition of user functions and specific points or intervals of time. For example, the end-user can define the function New-Price-Time (date) which gives the point in time in which changes of prices of a date are entered to the system.

2. Time stamping

J. Bubenko in [6] proposes to make the distinction between "time-unrestricted" and "time-restricted" modeling methods. Time-unrestricted methods consider the time dimension in its full generality, leading to an "infinite" model. A machine of potentially unlimited memory size (for storing an ever increasing body of historical information) and infinitesimally small processing speed (so that the time of a user request is the time of the satisfaction of the request) is assumed.

Time-restricted methods are more concerned with data processing (storage, updating, deletion) issues. Conceptual modeling refers to a finite collection of time-stamped n-tuples of entities or messages. End-users have a limited amount of information available in the data base at any given time and there is always a delay between a request and its processing.

CIAM [10], INFOLOG [23] and [7] are probably time-unrestricted approaches instead of REMORA [17], DADES [15], CSL [4],[11],[24],[3],[25] which can be considered as time-restricted modeling approaches.

By using the CIAM [10] time model previously introduced and mathematical notations, it is possible to define time dependent functions such as :

$$qoh : ARTICLE, T \dashrightarrow VQOH$$

where qoh is the name of the function and ARTICLE, T and VQOH denotes domains. qoh(x,t) denotes a particular article's x quantity-on-hand at a time point and equation such as $qoh(x,t) = \sum_{\forall t_i < t} QDEL(x,t_i) - \sum_{\forall t_j < t} QHIP(x,t_j)$,

(QDEL (x,t_i) resp. QSHIP (x,t_j) denote quantities delivered resp. shipped at time points t_i, t_j). Furthermore, it is possible to "reason about" the quantity-on-hand in a time perspective without bringing on question about processing, storage, etc... For instance, it is possible to introduce the constraint that bounds the function's value as $0 \leq qoh(x,t) \leq 5\,000$ or to define a new three argument function avgqoh (x, t_1, t_2) which gives the average qoh in the interval (t_1, t_2) as $avgqoh\ (x,t_1,t_2) = \frac{1}{t_2-t_1}\int_{t_1}^{t_2} qoh\ (x, t)\ dt$.

Of course the mapping from between such time-unrestricted view and the time restricted view has to be made explicit. Some aspects of this mapping are discussed in [10].

Similar ideas are developed in "temporal aspects of logical procedure definition" [22] and are integrated in the INFOLOG [23] methodology. A model or a specification of an information system is said to be "memory independent" if it is not concerned with what the system is to memorize or remember from its past and what it should be forget. A system designer using such a model need not worry about what facts to keep around for future use ; such decisions are relegated to a lower level of design. A memory independent specification language for describing information systems at the message/process level. This is to be constrasted with an object level specification where the application world itself is modeled. In describing a process the designer may need to refer to the past history of the information system. Extensive tools based on modal logic are provided for such reference. They include a variety of operators for time reference. Examples are operators for "always" and "sometimes", "before" and "after" and combinations of these such as "always after". Also included are time generation operators. Among these is a way of specifying the set of times at which some facts is true, of specifying a time interval by its end point and constants such as "THISWEEK" and "1984".

Other authors propose more "information processing" oriented time models. Olivé [15] assumes that the informations of the UOD can be modelled by means of the relational model. Using this model the IS schema is a set of relations schemes which satisfy two conditions : first they must be in third normal form, second, they should be elementary. A relation scheme is elementary if there exists at most one non-key attribute. For example PRODUCT (PRODUCT, NAME) or PRICES (PRODUCT, DATE, PRICE) or STOCK (PRODUCT, TIME, QUANTITY). However, the designer may define a non elementary relation scheme if he considers that it is a "natural" or practically non decomposable unit.

Similar way was already suggested by Langefor's [11] and Sundgren's approaches [24] where an information schema at the conceptual level is seen as an "ever-growing" set of e-messages with different time versions. An elementary message (e-message) is a 3-tuple $\langle o, a, t \rangle$ which informs about an elementar situation (e-situation) in the object system. In a e-message, "o" is a reference to a non-empty set of objects (entities), "a" is a reference to a relationship type or to a property/value pair and "t" is a reference to a point in time or a time period when this relationship or property holds or is measured (observed).

The relational model is proposed by Rolland in the REMORA method [8] to build conceptual schemas. A conceptual schema is (partly) a collection of c-object relation schemes. A relation scheme is a c-object if two conditions are satisfied : first, it must be a third normal form relation, second the dependencies must be permanent. A time varying function pfd : $D_X \longrightarrow D_Y$ between two attribute domains D_X and D_Y is a permanent functional dependency denoted $X \longrightarrow Y$ if, at every point in time, for a given value $x \in D_X$ there will be a unique value $y \in D_Y$. A pfd is a stronger dependence than a functional dependency : if $X \longrightarrow Y$, at every point in time, the value y is the same.

PRODUCT (<u>PRODUCT</u>, NAME, CREATIONDATE), STOCK (<u>PRODUCT, STOCKDATE</u>, QUANTITY), PRICE (<u>PRODUCT, PRICEDATE</u>, PRICE) are c-object relation schemes.

A c-object relation represents a time consistent aspect of an UOD object or relationship class. Th notion of c-class is introduced as the set of c-objects corresponding to the same UOD object or relationship class.

Bradley [3] remarks that some attributes of entities are time-varying and claims that the simplest way to represent a time-varying attribute is via a distinct record type that is related by an n : 1 relation to the entity record type involved and that has one instance for each period over which the attribute is constant.

At last, Wiederhold et all [25] suggest that information must be represented in what may be called a "discrete temporal relation" in which each tuple has a time attribute designating the time at which a set of properties are observed for an entity.

22. Concepts for modeling behavioural semantics of data bases

The scope of behaviour oriented methods is more general than to provide concepts for time modeling and time stamping the states of entities and relationships represented in a data base. It includes also the representation of the mechanisms that generate the state transitions.

The focus of behaviour oriented methods is to provide modeling concepts that support designers in reasoning about all aspects of the UOD and force them to explicitly and formally define and explain statics as well as dynamics and relationships.

The purpose of this paragraph is to review some concepts which have been added in behaviour oriented methods in order to satisfy such objective.

Of course, the inspection of literature shows a great variety of concepts. However two concepts seems to emerge : these are : event and transition rule (or operation, procedure,...).

Let us examine their definitions in different methods.

Benci et al [2] discuss in an early work (1976) concepts for the design of conceptual schema. Their assumption is that the real-world can be completely described with objects, properties and associations between these-objects. The n-ary relational formalism is considered adequate to describe a conceptual data base which is seen as a set of indexed relations $B^i : \{ R_1^i, R_2^i, \ldots R_q^i \}$. A conceptual data base B^{i+1} is obtained by applying to B^i operations, which conform to defined evolution rules and are triggered by events. An event is "any decision which triggers the processing of an operation on the data base. An event is either at rest or activated. The activation of an event can be due either to the host organization (entry event) or to a particular state of the base. In this latter case, one will discriminate between the events which trigger the operations, being internal to the base (internal events) and those which are known by the organization (result events).

Events are seen as decisions which cause changes by the execution of operations i.e evolution rules. They describe the dynamics of a data base and in particular the dynamics of the integrity constraints expressed by event-action sequences.

In Grossman [9] 1976, again the notion of event is taken into account to express constraints. An event "corresponds to some change in the real world which might have to be reflected as a change to the data base". Grossman claims that an event can be represented either as the difference between a "before" state and a "after" state or by specifying the before state and the transition. We find here the two possible approaches for representing dynamic facts : the static approach where pairs (before state, after state) are described and the dynamic approach where the after state just follows as a consequence of doing the transition to the before state.

Rolland presents in [16], 1978, a dynamics model that has later been followed up by Foucaut [8] and completed in [17,18,19,10]. Rolland claims that an information system specification is an integrated representation of collections of object classes, operation classes, event classes and associations between these categories. Three concepts so called c-object, c-operation and c-event are defined to provide a formal representation of the three classes of phenomena and relationships. The three concepts are defined using the relational formalism in a normal form called permanent normal form (see p. 7) in order to provide a not redundant conceptual schema. A c-object represents an atomic state of the information system. A c-operation represents an atomic action operating on a unique object. A c-event represents the atomic type of state change of a unique c-object which triggers one or several c-operations. The state change type is described by a predicate. For example the c-event "stock-breaking" is defined by a predicate

$$pred_{stock\text{-}breaking}(s,s') : \exists pro \in PROD \quad s(pro) > limit \ \underline{and} \ s'(pro) \leq limit$$

(PROD is the name of the c-object PRODUCT, s and s' refers to two successive states of the system).
Event occurs when the predicate is true.

The conceptual schema describes both the structure of the information system through the c-objects collection and its behaviour in a causal way by interrelations between c-objects, c-operations and c-events. (An event triggers operations which induce state changes on objects that can be events ...).

Remora provides graphical notations in order to describe both the static sub-schema and the dynamic sub-schema, and a specification language called ISDEL to describe c-object, c-operation and c-event relations and constraints.

An application of the dynamics modeling allows one to represent reality as a set of interdependent and interacting processes which together work on the objects of the information system. Events act as invocation conditions of processes. From the conceptual schema it is possible to define a synchronization graph of processes associated to a chronological graph of c-events : two processes (and two c-events) can be permanently chronologically ordered (an event type EV2 always follows an event type EV1), conditionally chronologically ordered (an event type EV2 may follow an event type EV1) or independent. In addition concurrency between processes can be specified through a graph of process execution dependencies.

Succedence and precedence of events are also introduced by Codd [6b], 1979, to be used by insertion and update programs for a data base to check the validity of a requested transaction :

> An event type E1 is an unconditional successor of an event type E2 if an event of type E1 always succeeds an event of type E2, but there is no intermediate event type E such that E is an unconditional successor of E2 and E1 is an unconditional successor of E.
> The alternate successor relation represents an exclusive-or of successors. An event of a given type must be followed by an event that belongs to one of its alternate successor types (if any).
> Unconditional and alternate precedence are defined in a similar way.

Breutmann et al 1979 [4] claims that a conceptual schema should specify both the static and the dynamics aspects of the data base. Statics is defined in terms of "atomic state, which means a single, elementary fact or association". Dynamics is defined in terms of atomic events representing the beginning, ending or change of an atomic state. All events corresponding to either the insertion, the deletion or the modification of association instances of a particular type, set up an event procedure. Again in this approach event and action are connected. Events occur at particular points in time and temporal notions (introduced p.4) are used to keep track of the history of events and to specify constraints on events, for example, constraints that limit the internal between two consecutive events of specified types or that limit the frequency of a given type of event within an interval.

In Sernadas [22], 1980, an information system is modelled by a network of concurrent, autonomous processes which exchange messages amongst themselves and with the environment through the IS temporal data base. Two kinds of messages are manipulated by processes and described by relations : state messages and event messages. States are facts that are turn over intervals of time. State relations are composed of state assertions that are usually in the system for a while. Events happen at discrete times. Events are seen as atomic state changes of the system. The state change is defined by an alteration rule in terms of pre-and-post conditions. The event-message is composed by references to the objects modified and details of the modification. For example, the event-message "stock-decreasing" gives informations about the occurrences of products that are modified and the amounts of the diminutions of quantities-in-stock. There are three atomic operations that processes can perform : signal the beginning of a state message, signal the end of a state message an generate an event message. Events can trigger others events and evolution rules describe the possible sequences of events. The relationship between two events a and b in the sequence is made by a process called generation rule which generates the event-message b using information of the event-message a. Processes can be executed in sequence or in parallel.

In order to develop a data base modeling language, Roussopoulos 1980, emphasized the need of taking into account events which occur in the real world. An event "occurs when the value of a predicate on some object of the real world changes from true to false and vice versa. We call the predicates that characterize some aspects of the real world, conditions. When an event occurs in the real world, actions may follows. An event denotes the moment when a state variable changes values. Therefore an event is an instantaneous thing, it lasts for a short period".

Antonellis et al [0]1981, again base the behaviour modeling on events. "An event expresses a change of conditions from the conditions which hold before (pre-conditions) to those which hold after (post-conditions). An event is described in terms of its pre-conditions and of the actions making the change (post-conditions just follow as a consequence of doing these actions)". Antonellis claims that the analysis of the UOD dynamic part is an event analysis activity leading to an events glossary, a specification form for each event and an events architecture. Events architecture is a formal description obtained by means of Petri Nets of causal dependencies, concurrencies and conflicts between events.

In Zisman [26], and in Barron [1] "Petri Nets are also used to model the temporal relationships between events and condition-action pairs to model the knowledge associated with individual events. Only when all the conditions associated with an enabled transition are true and the actions of the condition-action pair have been successfully carried out, will the transitions fire. The enabling and firing of transitions represents the occurence of events".

In a similar perpective Leonard et al [14], 1981, try to make the connection between the modeling of transactions and the modeling of data combining the relational model for data structuring and free-choice Petri nets to model the transactions.

Also in Mc Leod [13], 1982, a data base application is modelled by objects and application events described by classes. An application event can be seen as a transaction on the data base. It is composed by a collection of actions organized according to classical control structure : sequence, conditional statement and iteration. An action might be a call to primitive actions on the data base (insertion, deletion, modification) or a call to another application event. Event are external if they are fired by end-users and internal if they are triggered by another event.

Brodie [5], 1982, presents concepts, tools and techniques for an integrated design of data base and transactions. The methodology called ACM/PCM is mainly based on the principle of abstractions. For the design and specification of structural properties of data base applications SHM_+ provides one structural concept, the object and four forms of abstraction for relating objects. Three of them (classification - aggregation - generalization) come from the semantic hierarchy model, SHM [24] ; the fourth called association is added. It is a form of abstraction in which a relationship between member objects is considered as a higher level set object.

For the design and specification of behavioural properties of data base applications ACM/PCM provides primitive data base operations on objects and three forms of control abstraction with which to compose application-oriented operations and two forms of procedural abstraction : actions for conceptual modeling and transactions for transaction modeling. The basic operations are INSERT, DELETE and UPDATE and the three forms of abstraction are sequence, choice and repetition. An action is an application-oriented operation designed for one object to ensure that all properties of the object are modified.

A transaction is an application-oriented operation which alters one or more objects and can be decomposed into action. A transaction includes pre-conditions, action invocations, post-conditions and exception-handling.ACM/PCM provides both graphical notations for object scheme and behaviour scheme and a formal language for specification called BETA.

Gustafsson et al [10], 1982, proposes to develop a model of the application reality called the "conceptual information model" (CIM) that is free of traditional data processing concepts. There are three basic kinds of objects in the CIM : data, entities and events which are grouped into types. It is assumed that the UOD is consisting of a time varying set of entities. Entities are characterized by attributes specified by attribute functions. Attribute functions are often time-varying and have defined derivation-rules that may depend on events that affect their values. For example, a person's address may vary from day to day and can be defined in terms of the latest "address change event" that happened for that person. As another example, the monthly salary of an employee may be defined as the sum of all his earnings on jobs according to "job report" events for that month. Examples show that the event concept is another fundamental notion. Events are used to define the existence of entities. An entity "exists" for an application over specific time intervals. Each entity type has an associated existence rule defined in terms of event. For example, a "hiring event" signals the beginning of the existence of an employee. To each event is associated an occurence condition which specifies when it is to occur. Events are further categorized as external or internal. An internal event has a formal occurence rule specified. External events do not have such rules and their occurence is dependent on conditions outside the formal scope of the system.

3. CONCLUSIVE COMMENTS

By the inspection of the above review we may conclude that the increasing number of modeling approaches which try to include behavioural modeling in data base application design clearly indicates that authors emphasize the importance of behaviour. However, the variety of propositions also indicates that cumulative results have not been reported to a significant degree, that much work remains to be done and lead us to state the following comments :

31. A set of concepts necessary and sufficient

Reasonably we must admit that there is no general consensus concerning what an integrated structure and behaviour modeling is. Authors do not agree on which basic concepts and notions to employ. So many different ways of modeling left one bewildered. It seems that little modeling researchers have used the results of the previous work of their colleagues. They have frequently produced new modeling approaches with some fashionable features but whose modeling power has at best been marginally better than that of the previous models. It is also astonishing how researchers have proposed models with equivalent power and with so different numbers of concepts. INFOLOG for example uses about thirty concepts instead of four basic constructs in CIAM and three in REMORA. Models are growing even more enormous, but not stronger. May be the subject is mature to try a synthesis's work with the scope of defining a set of necessary and sufficient concepts. From the whole set of propositions it is not evident to decide what is the set of basic concepts but some convergence could be found (for the behavioural part) on the two concepts of EVENT and X-RULE (X means deduction or derivation or existence .... X-rule might be operation or procedure or action . .). From a semantic point of view an event appears as a phenomenon that occurs only once and that is instantaneous. It signals that something has happened in the real world. When an event occured, it exists for ever : nothing can delete or modify it. From a modeling point of view an event is described as a particular state change of the data base using predicates or pre and post conditions. Events explain why transitions on the data base occur, X-rules explain how the transition is made. Time dependent relationships between events are significant and explain the successive state-transitions of the data base.

32. Appropriate specification languages

We must admit that many modeling approaches have not been precisely and formally defined and have often only consisted of a simple graphical notation. Many of these graphical modeling approaches lack a well defined language for expressing complex static and dynamic constraints and rules of behaviour.

Researches in behaviour modeling probably suffer form the lack of a firm and formal mathematical foundation. The area of relational data bases has clearly illustrated the importance of such foundation particulary to achieve cumulative research results. Efforts in researching theoretical foundations of behaviour modeling and formal ways of specification have to been done.

Most "modern" approaches explore the way of extending logic-based views on data bases introduced by several researchers in the field of Artificial Intelligence. The information system would be monitored by a knowledge base including a syntax part, a rule part and a fact part. The syntax part would contain the syntax of the formal language to be used when communicating with the system ; the rule part would contain a set of formal definitions of assumptions made and rules established concerning the structure and behaviour of the UOD ; the last part would contain a set of facts observed in the UOD and all the facts derived from the observed facts using rules in the rule part .

This requires considerable work on logic-based knowledge representation and efficient deduction techniques for modeling of knowledge and behaviour in dynamic organizations.

33. A complete methodology of the use and application of models

We must also comment on the fact that authors have developed models and that little has been published on the methodology of the use and application on models to practical situations in data base applications development. This is comprehensive. Research in behaviour modeling is nevertheless a recent one and the definition of a model is logically the first focus point of the researchers. But it is clear that it is an intermediate step.

Efforts are needed to define methods able to integrate the use of models in the production of a high-level conceptual description of structure and behaviour and to map such description into a data processing system.

Complementarily efficient languages and data base processors to support execution of high level schemas are needed as well as development tools which aid the user in the modeling process.

34. The direction of work

The direction which emphasizes practical applicability of behavioural modeling approaches would include research issues such as :

- definition of a "standard" set of concepts for structure and behaviour modeling,

- development of a methodology of the systematic and active use of models in data base application development,

- definition of tools for analysis of correctness and consistency,

- development of efficient languages and data processors to support execution of a high level description of data base structure and behaviour,

- development of computer aided design tools.

The other direction would be to concentrate on research topics which could contribute to the definition of theoretical foundations of behaviour modeling and of formal languages for specification.

REFERENCES

[0] DE ANTONELLIS B, ZONTA : "Modeling events in data base applications design" in Proc. of the 7th VLDB Conf., Cannes, 1981

[1] DARRON J.L. : "Dialogue organization and structure for interactive information systems", in TR-CSRG-108, Computer research group, University of Toronto, 1980

[2] BENCI E, BODART F., BOGAERT H., CABANES A. : "Concepts for the design of a conceptual schema, in "Modeling in DBMS", Nijssen (ed), North Holland (pub), 1976

[3] BRADLEY J. : "Operations on data bases" in Proc. of the 4th int. VLDB Conf., Berlin, 1978

[4] BREUTMANN B., FALKENBERG E., and MAUER R. : "A language for defining conceptual schemas" in "Data Base Architecture", G.M. Nijssen (ed), North Holland (pub), 1979

[5] BRODIE M., SILVA E. : "Active and passive component modeling : ACM/PCM"
in Proc of IFIP-TC8-WC on "Comparative review of information systems design methodologies", TW Olle, HG Sol, AA Verrijn Stuart (eds), North Holland (pub), 1982

[6] BUBENKO J. : "the temporal dimension in information modeling"
in Proc of IFIP-TC2-WC in "Architecture and models on DBMS", Nijssen (ed) North Holland (pub), 1977

[6b] CODD EF. : "A relational model of data for large shared data banks"
Comm. ACM, vol 13, n°6, 1970

[7] Mc DERMOTT D. : "A temporal logic for reasoning about processes and plans"
in "research report" ≠ 196, Dpt of computer science, Yale University, 1981

[8] FOUCAUT O. : "modèle et outil pour la conception des systèmes d'information dans les organisations : Projet Remora"
In Thèse de docteur es sciences mathématiques, Nancy 1982

[9] GROSSMAN R.W. : "Some data base applications of constraint expressions"
in MIT-TR-158, 1976

[10] GUSTAFSSON MR, KARLSSON T., BUBENKO J : "A declarative approach to conceptual information modeling"
in IFIP-TC8-WC on "Comparative review of information systems design methodologies", TW Olle, HG Sol, AA Verrijn Stuart (eds), North Holland (pub) 1982

[11] LANGEFORS B. : "Theoretical analysis of information systems"
4th ed, Studentlitetaratus/Auerbach, Lund, Sweden, 1973

[12] LEIFERT S. : "Un langage de spécification des systèmes d'information, un outil pour leur gestion"
Thèse de docteur ingénieur en informatique, Nancy 1980

[13] Mc LEOD, KING R : "The event data base specification model"
In Proc. of the 2nd Int. Conf. on data bases : "improving data base usability and responsivness", Jerusalem, 1982

[14] LEONARD M., LUONG BT. : "Information systems design approach integrating data and transactions"
in Proc. of 7th int. VLDB Conference, Cannes. 1981

[15] OLIVE A. : "DADES : A methodology for specification and design of information systems"
in Proc. of IFIP-TC8-WC on "comparative review of information systems design methodologies", TW Olle, HG Sol, AA Verrijn Stuart (eds), North Holland (pub), 1982

[16] ROLLAND C, FOUCAUT O. : "Concepts for the design of an information conceptual schema and its utilization in the Remora Project"
In Proc. of the 4th int. VLDB Conference, Berlin, 1978

[17] ROLLAND C. : "A methodology for information systems design"
in Proc of the int. NCC'81, Chicago, 1981, AFIPS (pub)

[18] ROLLAND C. : "Active Data bases"
in ACM DATA BASE review, 1983

[19] ROLLAND C, RICHARD C. : "The Remora methodology for information systems design and management"
in Proc. of IFIP-TC8-WC on "Comparative review on information systems design methodologies", TW Olle, HG Sol, AA Verrijn Stuart (eds), North Holland (pub), 1982

[20] ROLLAND C., FOUCAUT O., THIERY O. : "Active Information Systems"
In Proc. of the 4th Jerusalem Conference on Information Technology, Jerusalem, 1984

[21] SMITH JM, SMITH DCP : "Data base abstraction : aggregation and generalization
in ACM TODS 2, 2, 1977

[22] SERNADAS A : "temporal aspects of logical procedure definition"
In Information systems, Vol 5, n°3, 1980

[23] SERNADAS A., SERNADAS C. : INFOLOG
In Proc. of open IFIP CRIS-Conference, Lisbonne, 1983

[24] SUNDGREN B : "theory of data bases"
New York, 1975
or "conceptual foundation of the infological approach to data bases"
in Proc. of IFIP Conference on Data Base management, 1974

[25] WIEDERHOLD G., FRIES JF, WEYL S. : "Structured organization of clinical data base"
in Proc. of AFIPS NCC'75, 1975

[26] ZISMAN MD. : "Use of production systems for modeling asynchronous concurrent processes"
In Pattern directed inference systems, Academic Press., 1978

Leszek Sankowski [1/]

PRZEGLĄD ZASTOSOWAŃ GOSPODARCZYCH MINIKOMPUTERÓW

1. Wstęp

Tematyka zawarta w tytule jest niezwykle obszerna. Wydaje się celowe, aby świadomie ograniczyć prowadzone tu rozważania. Proponuję skupić uwagę na zdefiniowaniu i krótkim scharakteryzowaniu tej klasy sprzętu komputerowego, którą często intuicyjnie określa się jako minikomputery. Pozwoli to w sposób pośredni wykazać obszary zastosowania tej klasy maszyn. Artykuł kończy przegląd typowych, realizowanych w oparciu o gotowe dostępne na rynku pakiety aplikacyjne, dostarczane przez producentów sprzętu bądź niezależne firmy softwarowe. Kilka uwag dotyczy także stanu i przyszłości zastosowań minikomputerów w Polsce.

2. Przegląd typowych rozwiązań

Zważywszy, że termin minikomputer, choć używany już od 1960 roku może być różnie rozumiany, przypomnijmy sobie o jakiej klasie sprzętu komputerowego mówimy i jaka jest charakterystyka urządzeń należących do tej klasy.
Przyjmijmy za /2/ skrótowo, że do interesującej klasy należą rozwiązania bazujące na 16-32 bitowym procesorze o wielkości PAO rzędu 256 Kb-2Mb.
Za ogólne kryterium przynależności dla klasy minikomputerów przyjmuje się cenę instalacji, która waha się w granicach 50-300 tys. dol. USA. Przyjmuje się także, że instalacje minikompute-

---

1/ Centrum Projektowania i Zastosowań Informatyki, Warszawa

rowe powinny transakcyjnie obsługiwać od 4 do 32 użytkowników równocześnie. Za typowych reprezentantów klasy minikomputerów można przyjąć przykładowo urządzenia firmy MICRODATA, WANG, DEC -PDP 11, SM-4, HP 1000/3009, IBM serii 30, IBM serii 1,7 i dalszych ok. 200 typów oferowanych na światowych rynkach. Dla porównania komputery duże oferuje zaledwie ok. 12 dostawców, minikomputery ok. 500 dostawców. Tak więc minikomputery to bardzo szerokie spektrum urządzeń o dość szerokim zakresie zastosowań i mocy obliczeniowej instalacji. Wszystko wskazuje, że jest to przejściowa klasa urządzeń wydzielona wg. płynnych kryteriów dla celów komercyjnych.
W zależności od przyjętych kryteriów wiele typów urządzeń można byłoby zaliczać do różnych klas /np. IBM 38, DEC VAX 11/780 czy np. MERA-60, które mimo wszystko należy zaliczyć do klasy minikomputerów/. Minikomputery reprezentują szerokie spectrum zastosowań i pracują jako instalacje:

- ogólnego przeznaczenia, uniwersalne /general purpose/, a więc realizujących dowolne zaprogramowane zadania,
- specjalnego przeznaczenia /special purpose/ realizujących wybrane, pokrewne klasy zastosowań,
- dedykowane do realizacji wybranych zadań.

I tak przykładowo minikomputery DEC PDP-11 zaliczane są do urządzeń uniwersalnych, IBM serii 30 wykorzystuje się wyłącznie dla przetwarzania danych dla celów zarządzania / dostępne języki programowania - mało efektywny kompilator języka Cobol oraz RPG/. Zastosowanie dedykowane przejmują obecnie minikomputery.

Dzisiaj w latach 80 po wprowadzeniu na rynek takich maszyn jak przykładowo IBM-38 czy ICL DRS 20 pojęcie minikomputer staje się historyczne. W przypadku IBM 38 mamy do czynienia z wieloprogramową, wirtualną maszyną bazy danych pracującą w trybie interaktywnym z możliwością fizycznej rozbudowy PAO do 2 MB, kompatybilną z serią 3 i rodziną 30 maszyn IBM. DRS-20 określić można jako zestaw inteligentnych terminali i procesorów prze-

znaczonych do tworzenia rozproszonych systemów przetwarzania danych.

Krótkie charakterystyki wybranych typów minikomputerów zawiera załącznik 1 i 2.

3. Minikomputery przetwarzania danych /Small Business Minicomputer/.

Przeznaczone są do interaktywnej współpracy użytkownika z komputerem. Takie funkcje jak wprowadzanie, przetwarzanie, aktualizacja danych odbywa się głównie przez terminale w trybie interaktywnym. Bazując na modułowej strukturze i odpowiednim oprogramowaniu komunikacyjnym pozwalają na przechodzenie od zastosowań prostych, inicjujących komputeryzacje do coraz bardziej skomplikowanych systemów opartych o rozproszone bazy danych. Z tego wynika, że aby być prawdziwie efektywnym system taki musi być wieloprogramowy, jednocześnie zarządzać siecią swoich terminali i być zdolnym do komunikowania się z pozostałymi elementami sieci komputerowej, w szczególności z komputerem głównym /host computer/.

4. Przetwarzanie rozproszone /Distributed processing/

Najczęściej stosuje się minikomputery jako elementy zdecentralizowanej sieci komputerowej. W ramach sieci minikomputery muszą realizować cztery podstawowe funkcje:

- być efektywnym narzędziem wprowadzania danych,
- przetwarzać dane dla lokalnego użytkownika,
- tworzyć i utrzymywać lokalne bazy danych,
- zapewniać współpracę i komunikację z pozostałymi częściami sieci komputerowej.

Takie właśnie usytuowanie minikomputerów było i jest podstawą ich sukcesów w zastosowaniach. Rozproszone przetwarzanie i możliwości budowy hierarchicznych systemów komputerowych oto podstawowe przeznaczenie małych maszyn.

5. Oprogramowanie

Oprogramowanie podstawowe dostarczane jest w zasadzie przez producentów sprzętu. W skład tego oprogramowania wchodzą systemy operacyjne, edytory tekstu, assembler, programy załadowcze i pomocnicze /utility/. Większość producentów sprzętu dostarcza także kompilatory języków wyższego poziomu /Cobol, Fortran, PL/I/ oraz oprogramowanie narzędziowe bazami danych, obsługi zbiorów, formatowania ekranów, generatory wydruków itp. Pakiety oprogramowania aplikacyjnego są w zasadzie dostarczane przez organizacje współpracujące z producentami sprzętu.

6. Zastosowanie. Wybrane przykłady z rynków zachodnich

Zastosowanie minikomputerów da się ująć w pięć dość szerokich grup.

A. Przetwarzanie i obliczenia.

- zastosowania w przetwarzaniu informacji ekonomicznej obejmujące, typowe dla działalności organizacji gospodarczych sfery księgowości i rachunkowości /accounting, sales analisis itp./,
- planowanie i sterowanie produkcją /inventory, bill of materials itp./
- obliczenia inżynieryjno-naukowe,
- projektowanie wspomagane komputerem.

B. Przetwarzanie tekstów.

- edytowanie i składanie tekstów.

C. Komunikacja - przykładowo.

- koncentratur linie,
- front-end processor.

D. Wprowadzanie danych.

E. Sterowanie procesami fizycznymi.

Na problem zastosowań minikomputerów i ich miejsca spojrzeć można także z punktu widzenia ich roli sprawowanej w systemach hierarchicznych. Wyróżnić tu można:

- komputery zaplecza /BACK-END/, których funkcje polegają na organizacji współpracy komputera głównego /HOST/ z peryferiami, zwłaszcza pamięcią masową,
- teleprocesory /FRONT-END PROCESSOR/, organizacja sesji i transmisji danych,
- komputer komunikacyjny /COMMUNICATION CONTROLLER, REMOTE COMMUNICATION CONTROLLER/, zwany węzłem sieci komputerowej, wykorzystywany często do budowy podsieci o bardzo szybkiej transmisji,
- komputer lokalny - lokalna obsługa użytkownika.

W załączniku 3 zawarto krótkie opisy 10-ciu przykładowych zastosowań bazujących na różnych typach minikomputerów. Ceny dostawy systemu wraz z ich wdrożeniem wahają się od 30 do 100 tys. dol. USA i w zasadzie nie przekraczają tej kwoty. Większość pakietów aplikacyjnych powstaje w niezależnych grupach software'owych współpracujących z producentami sprzętu.

7. Zastosowanie minikomputerów w Polsce

Sprawy wyglądają bardzo źle. Brak sprzętu i jego jakość to podstawowy determinat obecnego stanu, a zapewne i następnych lat. Kilkadziesiąt sprowadzonych w latach 70-tych zestawów i maszyny z serii MERA-400 jest już dzisiaj fizycznie i moralnie zużytych. Z produkowanych obecnie w Polsce minikomputerów, mam na myśli SM-4 produkowane przez FMiK "ERA" i MERE-60 produkowaną przez "MERE-STER" tylko pojedyńcze egzemplarze trafiają do rąk krajowego użytkownika i to w konfiguracjach dalece odbiegających od

potrzeb masowego przetwarzania danych gospodarczych /podstawowe ograniczenie - małe pamięci dyskowe/. Import na poziomie kilkunastu dalszych minikomputerów z krajów RWPG nie zmienia perspektywy. A popyt na dobry, tani, minikomputer istnieje, mimo zachowawczości i nieinowacyjnego trendu, który opanował polskie organizacje gospodarcze. Interesujące, choć cząstkowe, badania popytu na zastosowanie minikomputerów w gospodarce oraz próbę zdefiniowania typowych konfiguracji dostosowanych do struktury produkcyjno-organizacyjnej państwa Polskiego zostały przeprowadzone na zlecenie MERY-STER przez CPiZI "ZETO-ZOWAR" /3/. Badania nt. możliwości zastosowań MERY-60 przeprowadzono w ponad 90 obiektach przemysłowych i usługowych. Badania pozwoliły na zdefiniowanie typowych konfiguracji sprzętowo-software'owych dostosowanych do badanych obiektów /klas tych obiektów/. Wnioski z badań potwierdzają potrzebę rozbudowy konfiguracji o pamięci masowe, przy czym łączna pojemność dysków do 60 MB wydaje się maksymalna dla zaspokojenia nawet największych potrzeb.W przedsiębiorstwach o większej liczbie komputeryzowanych obiektów należy przewidywać budowę sieci komputerowej. W połowie badanych obiektów należy przewidywać dwuzadaniowy typ pracy MERY-60 tzn. foreground/background. Dalszej rozbudowie powinny podlegać możliwości podłączenia większej liczby terminali i specjalizowanych końcówek /np. plotter/. W zakresie software aplikacyjnego niezbędne jest oprogramowanie prostych operacji w zakresie zakładanie i obsługi zbiorów, co jest szczególnie ważne dla użytkowników wchodzących w komputeryzację.

Literatura

/1/ AUERBACH Report. Bussines Minicomputers, Auerbach Publishers Inc. 1980 r.

/2/ WENIG R.P. PARDOE T.D. Introduction to Micro and Personal Computers, International Management, Framingham, Massachusetts.

/3/ WROBLEWSKI A. Studium możliwości zastosowań hierarchicznych systemów komputerowych z wykorzystaniem minikomputera MERA-60 dla potrzeb zarządzania w przedsiębiorstwach przemysłowych i handlowych, CPiZI "ZETO-ZOWAR" 1983 r.

/4/ Materiały informacyjno-reklamowe producentów sprzętu.

Zał. 1

Minikomputery uniwersalne. Wybrane przykłady.

/General - Purpose Minikomputers/.

Producent / Model	Procesor: Długość słowa /bity/			Procesor: Maksymalna pojemność PAO /bajty/				Pamięci pomocnicze						oprogramowanie: System operacyjny				oprogramowanie: Języki programowania								oprogramowanie: Oprogramowanie specjalizowane				
	12	16	32	128-256 K	256-512 K	512-1 M	powyżej 1 M	Diskette	dyski wymienne	dyski stałe	taśma magnet.	kasety magnet.	inne	pamięć wirtualna	wieloprogram.	Time-Sharing	Real-Time	Fortran	APL	Basic	Cobol	RPG	inne	SZBD	Oprogr. sieciowe	oprogram. przetwarzania plików/ rekordów	RJE/HASP	Obsługa wprowadzania/wyprowadzania danych
CONTROL DATA																													
-Cyber 18-20		x		x				x	x		x		x				x					x	x					x
DATA GENERAL																													
Nova 3/D		x		x				x	x	x	x		x		x		x	x		x	x		x				x	
DEC																													
PDP-8	x			x				x	x	x	x		x				x	x		x			x					
PDP-11/35		x		x				x	x	x	x	x	x		x		x	x		x	x		x		x			x	
VAX-11			x				x	x	x	x	x		x	x	x	x		x		x	x		x	x	x	x			x
HELWETT-PACKARD																													
HP-1000		x				x		x	x	x	x		x		x	x	x	x		x			x	x	x		x		x
HP-3000		x			x				x	x	x		x	x	x	x		x	x	x	x	x	x	x	x		x		x
HONEYWELL																													
Level Six/23		x		x					x	x	x		x			x	x	x		x	x	x	x	x		x	x		x
IBM																													
System 7		x		x					x	x		x	x		x		x	x					x						x
PRIME COMPUTER																													
500		x					x		x	x	x		x	x		x	x	x		x	x	x	x	x	x		x		x
UNIVAC																													
V77-400		x					x		x	x	x		x		x	x	x	x		x	x	x	x	x	x		x		x

Minikomputery specjalizowane. Wybrane przykłady. /Business Minicomputers/

Producent	Procesor							Pamięci pomocnicze						Oprogramowanie																	
	Długość słowa /bity/			Maksymalna pojemność PAO /bajty/										Systemy operacyjne				Języki programowania						Oprogramowanie specjalizowane							
Model	12	16	32	128-256 K	256-512 K	512-1 M	powyżej 1 M	Diskette	dyski wymienne	dyski stałe	taśma magnet.	kasety magnet.	Inne	Foreground/ Background	Wieloprogram.	Interaktywny	Batch	Fortran	Basic	Cobol	RPG	APL	Inne	Bazy danych	Oprogr. sieciowe	Przetwarz. plików/rekordów	RJE/HASP	Obsługa wprowadzania/wyprowadzania danych			
BURROUGHS																															
B 1830		x			x			x	x	x	x					x			x	x	x			x	x	x		x		
DATAPOINT																															
6600		x		x				x	x	x	x					x			x	x	x				x	x		x		
HELWETT PACKARD																															
HP 250		x		x				x	x	x						x			x	x				x		x		x		
IBM																															
System 3		x		x				x	x	x	x			x						x	x					x	x	x		
System 34				x				x			x				x	x	x			x	x					x	x	x		
System 38							x	x	x	x	x				x	x				x	x			x	x	x	x	x		
ICL																															
System Ten		x		x					x		x				x	x				x	x					x		x		
NCR																															
NCR-8000		x			x			x	x	x				x	x	x				x						x		x		

Zał. 3

Przegląd wybranych zastosowań bazujących na minikomputerach. Źródło /1/.

1. NAZWA SYSTEMU	PLEAS Insurance	Management System
2. DOSTAWCA	AMERICAN HEALTH	ACS
3. TYP EMC	DEC PDP 11/70	DEC PDP seria 11
4. ZASTOSOWANIE	Ubezpieczenia -rozliczenia świadczeń, utrzymywanie karteteki ubezpieczeń	Przetwarzanie inform. dla celów zarządzania w organizacjach badawczo-rozwojowych.
5. MAX. PAO	-	1 Mb
6. PAMIĘCI ZEWNĘTRZ.	-	taśma magnet., dyski
7. SYSTEM OPERACYJNY	Interaktywny RSX-11M	RSX-11M
8. CENA ZAKUPU	100 tys. dol. USA	75-100 tys. dol. USA
9. PIERWSZA INSTAL.	1978	197

1. NAZWA SYSTEMU	BARRISTER	Compudata System
2. DOSTAWCA	Compek	Compudata
3. TYP EMC	Nova	IMB seria 1
4. ZASTOSOWANIE	utrzymywanie bazy danych tekstowych oraz rozliczenia biur prawniczych.	Planowanie i sterowanie produkcją
5. MAX. PAO	192 Kb	64-356 Kb
6. PAMIĘCI ZEWNĘTRZ.	taśmy magnet + dyski ok. 200 Mb	diskette + dysk
7. SYSTEM OPERACYJNY	Interakt.,wieloprogram	Interakt.,wieloprogram.
8. CENA ZAKUPU	25-100 tys. dol. USA	25-50 tys. dol. USA
9. PIERWSZE INSTAL.	1972	1978

1. NAZWA SYSTEMU	BBPS	Bookmaker System
2. DOSTAWCA	Binary Data System	Bristol Systems
3. TYP EMC	ECLIPSE S/130	DATAPOINT
4. ZASTOSOWANIE	Utrzymywanie danych banku krwi łącznie z rejestracją dawcy, biorcy, analizami labolator.	Rozliczanie sprzedaży i zakupów, sterowanie zapasami, analiza sprzedaży
5. MAX. PAC	256 Kb	256 Kb
	taśmy magnet., dyski 100 MB	Dysk
7. SYSTEM OPERACYJNY	Interaktywny, wieloprogramowy	Interaktywny, wieloprogramowy
8. CENA ZAKUPU	ok. 100 tys. dol. USA	30-50 tys. dol. USA
9. PIERWSZA INSTAL.	1978	1975

1. NAZWA SYSTEMU	S-1500	FUNDAC
2. DOSTAWCA	Computer Concepts	Monchik - Weber
3. TYP EMC	DEC PDP-11	IBM-34
4. ZASTOSOWANIE	Sterowanie procesami produkcyjnymi fabryk produkcji żywnościowej. Rozliczanie produkcji.	Ubezpieczenia profesionalne
5. MAX. PAO	256 Kb	128 Kb
6. PAMIĘCI ZEWNĘTRZ.	Dysk	Dysk - 40 MB
7. SYSTEM OPERACYJNY	-	Interaktywny
8. CENA ZAKUPU	100 tys. dol. USA	100 tys. dol. USA
9. PIERWSZA INSTAL.	1974	1978

1. NAZWA SYSTEMU	Construction Accounting System	WRS
2. DOSTAWCA	-	TMI System
3. TYP EMC	DATAPOINT	DOC PDP-11
4. ZASTOSOWANIE	Przedsiębiorstwa budowlane w zakresie planowania i rozliczania prac budowlanych.	Obsługa operacji bankowych
5. MAX. PAO	128 Kb	-
6. PAMIĘCI ZEWNĘTRZ.	Dysk + kasety magnetyczne	taśma magnet. + dysk
7. SYSTEM OPERACYJNY	Interaktywny, wieloprogramowy	Interaktywny
8. CENA ZAKUPU	ok. 30 tys. dol. USA	-
9. PIERWSZA INSTAL.	1976	1979

Jan Sztajer 1/

BARIERY INFORMATYZACJI ZARZĄDZANIA PRZEDSIĘBIORSTWEM PRZEMYSŁOWYM

Długo zastanawiałem się nad do mnie skierowanym przez Organizatorów INFOGRYFU-84 zadaniem: przedstawić swój pogląd na temat przyczyn tak niezadawalającego stanu zastosowań informatyki dla potrzeb zarządzania przedsiębiorstwami przemysłowymi w naszym kraju.

Były dwa powody które wpływały na moje wahania. Pierwszy to fakt, iż jednoznaczna odpowiedź na tak postawione pytanie jest bardzo trudna oraz złożona i będzie zawsze subiektywna a tym samym dyskusyjna. Drugi powód jest związany ze stosunkowo dużą ilością wypowiedzi na ten temat opublikowanych przez informatyków profesjonalnych i nie tylko.

Jednak argumentacja użyta przez mojego kolegę dr inż.Edwarda Krama przekonała mnie do tego, aby przedstawić swój pogląd na tę sprawę. Tym bardziej, że profesjonalnie zajmowałem się "maszynowym" przetwarzaniem danych przez dwadzieścia sześć lat, współuczestnicząc w jakimś malusieńkim stopniu w powstawaniu POLSKIEJ INFORMATYKI. Był to argument mocny i przekonywujący.

Mój pogląd będzie dotyczył zastosowań informatyki wspomagającej zarządzanie przedsiębiorstwem przemysłowym a głównie przemysłu maszynowego, gdyż w większości tam pracowałem. Być może , że w innych przemysłach występowały i występują inne powody trudności w rozwoju informatyki, specyficzne dla danej gałęzi lub branży.

1/ Radiotechnika Wrocław

Również moje uwagi nie dotyczą zastosowań informatyki do sterowania procesami, prac inżynierskich itp. Ponieważ, jeżeli jedynym powodem stosunkowo niskiego poziomu zastosowań informatyki w tych obszarach był i jest brak odpowiednich specjalistycznych środków technicznych, to jest to w jakiś sposób usprawiedliwiające. Natomiast jeśli, oprócz strony technicznej, doszły inne problemy np. natury organizacyjnej, społecznej itp. to uważam, iż aby wypowiadać się w tych sprawach należy przeżyć to na "własnej skórze". Inaczej będą to tylko teoretyczne przypuszczenia.

Dotychczasowe rozważania wielu autorów na temat przyczyn i powodów niskiego poziomu zastosowań komputerów czy też stanu polskiej informatyki, w zasadzie były prezentowane z punktu widzenia trzech głównych problemów: technicznego, organizacyjnego i kadrowego. Niektórzy autorzy poruszali jeszcze problem efektywności zastosowań komputerów, szczególnie dla potrzeb zarządzania przedsiębiorstwami przemysłowymi. I ja w zasadzie będę się trzymał tego algorytmu, gdyż pozwala on spojrzeć na problemy polskiej informatyki prawie ze wszystkich stron.

Jednak chciałbym spojrzeń na te problemy trochę z innego punktu widzenia. Otóż uważam, że nieporównywalnie niski w stosunku do innych krajów nawet sąsiednich, stan zastosowań komputerów dla potrzeb zarządzania przedsiębiorstwami w naszym kraju, wynika przede wszystkim z kilkudziesięcioletniego braku ZAPOTRZEBOWANIA NA INFORMACJĘ PRAWDZIWĄ /odpowiadającą rzeczywistości/ DOKŁADNĄ I TERMINOWĄ. Inaczej mówiąc, oczekiwano i przyjmowano ze spokojem tylko taką informacją, na jaką było zapotrzebowanie i to najlepiej "przedterminową" a nie taką jaka była w wielu przypadkach w rzeczywistości. Upraszczając, nie zawsze chciano przyjąć do wiadomości, że dwa plus dwa jest cztery. Wszystkie pozostałe trudności /problemy/ polskiej informatyki uważam, że są pochodną wyżej przedstawionej tezy.

Zdaję sobie sprawę z tego, iż przedstawiony pogląd jest dyskusyjny, spotka się z krytyką nie tylko informatyków, wręcz może być źle odczytany. Jednak jeżeli spojrzymy na wyżej wymienione trzy problemy informatyki z tego punktu widzenia to doprawdy trudno szukać innych powodów. Zakrawa to trochę na paradoks ale właśnie sytuacja gospodarcza w jakiej znalazł się nasz kraj oraz podjęte historyczne decyzje w sprawie wdrażania reformy gospodarczej utwierdziły mnie w słuszności powyższej tezy. Przecież próby "likwidacji" korzystania z komputera przez przedsiębiorstwa jakie pojawiły się gdzieniegdzie w ostatnich latach, chociaż publicznie argumentowane nieefektywnością zastosowań, awaryjnością sprzętu itp. mają inne podłoże. Uważam, że niektórzy kierownicy przedsiębiorstw przemysłowych wychowani w poprzednich wymaganiach, które doprowadziły do "perfekcji" /chociażby w zakresie sprawozdawczości GUS-owskiej/ szybko zrozumieli, iż żądania reformy gospodarczej są jednoznaczne w tym zakresie i nie pozwolą na dowolne manipulowanie liczbami. Wcześniej czy później prawda o działalności przedsiębiorstwa wyjdzie na wierzch i np. Narodowy Bank Polski powie stop. W związku z tym, aby jednak w dalszym ciągu próbować wyjść na swoje, trzeba wydać walkę m.in. komputerom ale w sposób bardzo inteligentny, bo to przecież koniec dwudziestego wieku!

Dla poparcia powyższej tezy pozwolę sobie przedstawić kilka wybranych przykładów "z życia".

Za czasów maszyn licząco-analitycznych, który w naszym kraju powstawał pierwszy "mózg elektroniczny" aby kupić cały zestaw czy też np. mnożarkę lub tabulator, jedynym uzasadnieniem przyjmowanym przez decydenta, który przyznawał środki finansowe na zakup, był argument ilości zwolnionych pracowników administracyjnych. Nie ważne były merytoryczne potrzeby uzupełnienia zestawu maszyn licząco-analitycznych czy też konieczność unowocześnienia prac obrachunkowych itp. Jeżeli przedsiębiorstwo nie potrafiło wyliczyć ile ludzi zwolni, nie

otrzymywało środków inwestycyjnych na zakup sprzętu. Dochodziło zatem do komicznych uzasadnień jak np. mnożarka wykonuje działania średnio 3 tys. operacji na godzinę a człowiek 300 w związku z tym można zwolnić ... itd. itp.

Na początku lat sześćdziesiątych w okresie porządkowania i usprawniania gospodarki oraz zarządzania przedsiębiorstwami przemysłowymi, przystąpiono do wprowadzania m.in. norm technicznie uzasadnionych. W jednym z przedsiębiorstw przemysłu maszynowego posiadającego maszyny liczące-analityczne, opracowano roczno-kwartalny plan produkcji z praxie wszystkimi wymaganiami tego algorytmu. Jednym z obliczeń był bilans zatrudnienia pracowników bezpośrednio-produkcyjnych. Wydrukowana ilość tych pracowników niezbędna do wykonania planu produkcji wskazywała, iż stan rzeczywisty należałoby zmniejszyć o około trzysta osób. Dokładne sprawdzenie przez specjalistów prawdziwości wszystkich danych przekazanych do obliczeń nie stwierdziły błędu. Wynik był poprawnie obliczony. No i co z tego! Tabulogram nie mógł ujrzeć światła dziennego. Poleceniem kierownictwa przedsiębiorstwa należało obliczenia powtórzyć, oczywiście po uprzednim "odpowiednim skorygowaniu" bazy norm czasowych.

Dla jednego z dużych przedsiębiorstw przemysłowych wyliczano optymalizację rocznego planu produkcji na komputerze Elliot 803B, programem bibliotecznym. Dane zostały przygotowane zgodnie z wymaganiami programu. W ciągu kilku kolejnych podejść do komputera, nigdy nie wykonano obliczeń do końca. Powód był prosty, ilekroć komputer napotykał kolejny minusowy wynik w zakresie posiadanej mocy produkcyjnej, tylekroć kończył obliczenia. Program nie przewidywał możliwości wykonania produkcji w sytuacji gdy brak było mocy produkcyjnej. Ponieważ kolejne "dopasowywania" danych przez użytkownika nie dały rezultatu obliczenia przerwano. Tym niemniej plan został przyjęty do realizacji ponieważ okazało się, że program wykonany przez anglików nie przewidywał możliwości przez przedsiębiorstwo podjęcia zobowiązań z różnych okazji.

Dalsze przykłady można by było mnożyć i zapewne każdy informatyk zajmujący się wdrażaniem posiada ich dużo. Ot, chociażby efektywność wykorzystania tabulogramu materiałów niewykazujących ruchu przez ponad np. sześć miesięcy i więcej. Czy też tabulogram zawierający informacje o posiadanych rezerwach mocy produkcyjnej maszyn i urządzeń w przedsiębiorstwie.

Ilekroć zadawałem pytania w powyższych sprawach tylekroć słyszałem prawie tę samą odpowiedź: który to dyrektor chcący spokojnie kierować przedsiębiorstwem pokazuje jakie ma faktyczne rezerwy mocy produkcyjnej czy też kadrowe, nie mówiąc już o tym jakie ma ilości materiałów zbędnych lub nadmiernych, szczególnie w okresach występujących trudności zaopatrzeniowych?

Patrząc na te problemy z obszaru zastosowań informatyki w przedsiębiorstwie przemysłowym dziedzinami, gdzie dodawanie i odejmowanie tych samych liczb zawsze musi dać ten sam wynik są systemy ewidencyjne. Tym też chyba należy tłumaczyć największą ilość wdrożeń w zakresie ewidencji: stanu i ruchu materiałów, przedmiotów nietrwałych, środków trwałych, pracowników, wyrobów gotowych, wykonanej produkcji i obliczania zarobków.

Natomiast w tych przedsiębiorstwach, w których dyrekcja była świadoma celu stosowania nowoczesnych technicznych środków obliczeniowych, zrozumiała konieczność odpowiedniego przygotowania się do jej zastosowania, umiała wykorzystywać do zarządzania wyliczone informacje, tam przetwarzaniem objęto obszar technicznego przygotowania produkcji, planowania produkcji itp. Niestety takich przykładów jest stosunkowo nie wiele.

Najczęściej w statystykach zastosowań informatyki przyjmuje się nazwy zagadnień sugerujące o wiele większe obszary przetwarzania niż w rzeczywistości występują. Chociażby gospodarka materiałowa a w rzeczywistości jej mały fragment ewidencja stanu i ruchu materiałów, czy też planowanie produkcji a w rzeczywistości wyliczanie pracochłonności na poszczególne np. stanowiska lub wydziały według przyjętego planu itd. itp.

Takie zapotrzebowanie na informacje statystyczną o zakresie zastosowań, nie mówiąc już o "osiągniętych" i "wyliczonych" efektach wymiernych, doprowadziły do tego, że w pewnym okresie można było się dowiedzieć, iż w wielu dużych przedsiębiorstwach naszego kraju funkcjonują już zautomatyzowane systemy zarządzania oparte na bazach lub bankach danych realizowane na zestawie Odra 1304, wyposażonym w sześć jednostek taśmowych, pamięć operacyjną 32 KS, jedną drukarkę i jeden czytnik!

Muszę w tym miejscu zaznaczyć, iż wg mojego rozeznania w większości przypadków nie było to winą tylko informatyków. Takie stawiano nam zadania i nie przyjmowano do wiadomości "nagiej prawdy". Trzeba również i powiedzieć, że sami informatycy dokładali się do tej dezinformacji, poprzez np. publikowanie swoich nieprawdopodobnych pomysłów jako praktycznych wdrożeń z tym, iż w tej grupie dominowali "znawcy problemu", którzy nie pobrudzili sobie rąk ciężką robotą wdrażania. Zaś swoją wiedzę zdobyli bądź z literatury i prospektów zachodnich, bądź też w umiejętny sposób załatwianego udziału w różnego rodzaju kursach, sympozjach, konferencjach szczególnie w krajach drugiego obszaru. Oni to najczęściej pouczali i doradzali jak to należy robić w przedsiębiorstwie aby zastosować komputer, zbudować bazę lub bank danych jak też, krytykowali niski poziom wykorzystania komputerów, brak efektów szczególnie wymiernych, zły stan techniczny sprzętu itp. Niestety nie przyjmowali do wiadomości faktycznego, istniejącego stanu POLSKIEJ INFORMATYKI.

Jak stwierdziłem wyżej, wszystkie problemy polskiej informatyki są pochodną postawionej tezy. Spójrzmy na naszą informatykę od strony kadrowej bowiem uważam, że jest to problem najważniejszy zresztą w każdej dziedzinie nie tylko w informatyce. Nie będę wypowiadał się na tamat polityki kadrowej w ogóle, gdyż z jednej strony odbiega to od problemów niniejszego opracowania z drugiej zaś jest to temat drażliwy i trudny a w ostatnich kilku burzliwych latach zabiera dużo czasu najwyższym władzom naszego kraju jak i miejsca w publikatorach.

Warto w tym miejscu przytoczyć jedynie pogląd prof. Mieczysława Kabaja który stwierdza, iż "Wadą polityki kadrowej jest jej zamknięty charakter /.../ Nie funkcjonują sprawnie systemy rotacji i wymiany kadr /.../ A w wielu dziedzinach rotacja kadr stanowi warunek postępu". Profesor wyciąga wniosek, iż w polityce kadrowej - jeżeli ma ona spełniać funkcje motywacyjne - nie może być mechanizmów ukrytych i poufnych.

Jasność w tych sprawach osiąga wiele krajów poprzez rozgraniczenie dwóch podsystemów polityki kadrowej. Pierwszy podsystem obejmuje naczelne kadry kierownicze objęte nomenklaturą państwową lub partyjną. Tutaj dobór i awansowanie kadr opiera się na kryteriach merytorycznych i politycznych. Drugi podsystem obejmuje kadry specjalistów i średnie kadry kierownicze . W tym podsystemie stosowane są wyłącznie kryteria merytoryczne /wiedza, fachowość, uzdolnienia i umiejętności/ a formy doboru mają charakter całkowicie jawny i otwarty. Zdaniem profesora "główna słabość polityki kadrowej w naszym kraju polega na pomieszaniu tych dwóch podsystemów /2/.

Chciałbym natomiast przedstawić kilka swoich refleksji w tym delikatnym problemie odnośnie tylko informatyki. Przy czym dla uproszczenia całą społeczność informatyków dzielę na trzy grupy: "czarnoroboczą" zajmującą się projektowaniem, programowaniem, wdrażaniem, obsługą i naprawą sprzętu, druga grupa to "decydenci informatyki" mający m.in. uprawnienia podejmowania decyzji, zaś trzecia grupa to naukowcy. Z punktu widzenia omawianych problemów, kolejność grup jest przypadkowa. Z perspektywy ponad dwudziestu lat rozwoju polskiej informatyki można stwierdzić, iż pierwsza i trzecia grupa mogą pochwalić się autentycznymi, sprawdzalnymi osiągnięciami. Naukowcy i pedagodzy polscy w warunkach m.in. technicznych jakie im stworzono, potrafili wykształcić w ciągu tych lat podstawowy trzon kadry praktycznych informatyków. Przecież w zdecydowanej większości przypadków to właśnie ci informatycy, w tych konkretnych warunkach organizacyjnych i technicznych w naszym kraju potrafili

jednak wykazać się praktycznymi i konkretnymi wdrożeniami komputera dla potrzeb zarządzania przedsiębiorstwami. I tam gdzie trafili oni na mądrego dyrektora i pozostałe kierownictwo przedsiębiorstwa, które rozumiało potrzebę i możliwości komputera, osiągnęli pewien poziom zastosowań. Dobitnym tego potwierdzeniem jest fakt, iż nie znam przykładu z ostatnich lat aby te właśnie przedsiębiorstwa rezygnowały z komputeryzacji. Natomiast tam gdzie kupiono komputer na pokaz, bowiem "dyrektor bez komputera nie mógł być dobrym dyrektorem", a potem dopiero pytano informatyków co z tym fantem robić, już na początku reformy gospodarczej najczęściej dawano ogłoszenia o chęci jego sprzedaży. Oczywiście decyzję taką uzasadniano nieefektywnością, awaryjnością sprzętu komputerowego itp. Należy stwierdzić ponadto, że stosunkowo liczna grupa wykształconych informatyków w wielu przedsiębiorstwach nie była i nie jest w pełni wykorzystywana. Uważam, że właśnie w informatyce znajduje potwierdzenie pogląd prof. Mieczysława Kabaja który mówi: "... Ciągle istnieje ogromna przepaść między wielkim potencjałem wykształconych kadr, a mierną, często niezwykle niską jakością i sprawnością funkcjonowania różnych dziedzin życia społecznego i gospodarczego. Gospodarka uspołeczniona zatrudnia ponad milion pracowników z wyższym wykształceniem, w tym ponad 400 tys. inżynierów i ponad 5 milionów z wykształceniem średnim technicznym i zasadniczym zawodowym. Potencjał ten stwarza warunki szybszego postępu technicznego, organizacyjnego, wzrostu wydajności pracy. Nie jest on jednak dostatecznie spożytkowany. /.../ Rodzi to frustracje i rozczarowania, szczególnie wśród młodszego pokolenia Polaków /2/.

Pozostała druga grupa "decydentów informatyki" którzy, z punktu widzenia strategii rozwoju zarówno m.in. bazy technicznej, jak i organizacyjnej polskiej informatyki mieli największy wpływ, bowiem przygotowywali bądź podejmowali decyzje, których skutki rozciągają się na okresy wieloletnie.

Do grupy tej zaliczam wszystkich decydentów poczynając od dyrektorów przedsiębiorstw - producentów m.in. sprzętu komputerowego aż po najwyższą władzę gospodarczą kraju. Bowiem, jak pa-

miętamy swego czasu głównym informatykiem w kraju był również premier.

Wypowiadając się na temat skutków działalności tej grupy informatyków muszę poczynić następujące założenie. Informatyka, szeroko rozumiana, jest chyba jedną z nielicznych jeśli nie jedyną dziedziną działalności gospodarczej człowieka w ostatnim stuleciu, której funkcjonowanie i rozwój jest tak bardzo uzależniony od ścisłego i harmonijnego powiązania między sobą wielu składowych tj. techniki /głównie sprzętu, łączności itp./ projektowania i oprogramowania, kadr, organizacji własnej i użytkownika, nauki i badań. Żadna z tych składowych nie może być preferowana ani nie doceniana. Jakiekolwiek próby nadawania priorytetów, doprowadzają do sytuacji jaką mamy w polskiej informatyce.

Aby decydować o rozwoju tak złożonego systemu działalności gospodarczej, niezbędna jest odpowiednia wiedza merytoryczna ale i nie tylko. Absolutnie, co chciałem z całą mocą podkreślić nie twierdzę, że wiedzy tej nie można opanować. Jednak, jeżeli zarówno wykształcenie akademickie jak i wieloletnia działalność praktyczna nie miała nic wspólnego z informatyką, to opanowanie tej wiedzy uprawniające do podejmowania decyzji strategicznych wymaga albo długiego okresu nauki albo też, umiejętności wykorzystywania wysoko kwalifikowanych doradców specjalistów. Jeżeli dobrze pamiętą, to w całym dwudziestoleciu polskiej informatyki o jej rozwoju decydowali elektrycy, mechanicy, ekonomiści tylko nie informatycy, mający za sobą również praktyczne doświadczenia. Jeśli zatem brak jest wiedzy teoretycznej jak i umiejętności wykorzystywania specjalistów, to trudno się np. nie zgodzić z Krzysztofem Fronczakiem, który stwierdza "Dziś, z perspektywy czasu, zasługi "Mery" w rozwoju polskiej informatyki ocenia się na ogół dość chłodno. Na wspomnianej dyskusji sejmowej przyczyn złego dziś stanu informatyki upatruje się wręcz w "destrukcyjnej działalności byłego zjednoczenia "Mera", a produkcję komputerów nazwano chałupniczą" /3/. Ale w tym miejscu jednak należy postawić pytanie, czy

tylko dawne Zjednoczenie "Mera" należy obwiniać takim a nie innym stanem polskiej informatyki? Przecież nie tylko ono było odpowiedzialne za koordynację i rozwój tej dziedziny działalności gospodarczej kraju, jak i za produkcję różnego rodzaju sprzętu komputerowego a przede wszystkim bazy elementowej. Nie tylko ono opracowywało przeróżne prognozy ale i producenci jak i inni decydenci. Muszę przyznać rację dyrektorowi Meramatu Stanisławowi Gerczyńskiemu, który, w powołanym wyżej artykule Krzysztofa Frenczaka stwierdza, że "Programy na wyrost były prawdziwym nieszczęściem informatyki. Brak rozeznania sił i możliwości kosztował ciężkie pieniądze ładowane dosłownie w błoto" /3/.

Stwierdzenie powyższe jest tylko potwierdzeniem mojego poglądu na temat działalności tej grupy informatyków.

Bardzo bym nie chciał aby mój pogląd w tej sprawie był odczytywany jako np. "osobista wycieczka" lub coś w tym rodzaju. Absolutnie nie jest to moim zamiarem. Jestem głęboko przekonany, że rozwój informatyki w naszym kraju musi następować i należy wyciągnąć wnioski z dotychczasowych doświadczeń, stąd też jedynie pozwalam przedstawić swoje spostrzeżenia.

Kolejnym problemem, jako pochodnym tezy postawionej na początku niniejszego opracowania, jest organizacja nie tylko samej informatyki w kraju ale przede wszystkim obiektów działalności gospodarczej chcących wykorzystywać to nowoczesne narzędzie dla potrzeb zarządzania.

Problemy organizacyjne informatyki w naszym kraju są wielopłaszczyznowe i trudne do jednoznacznego rozwiązania. Przeto pozwolę sobie jedynie zacytować obszerny fragment referatu sprawozdawczego Zarządu Głównego Polskiego Towarzystwa Informatycznego /1/. Twierdzimy, że w dotychczasowym stosunku władz państwowych i gospodarczych do informatyki i jej zastosowań panował i panuje chaos, brak zrozumienia tego, czego można się po informatyce spodziewać i czego można od niej żądać. Uważamy, że prawidłowe zastosowania informatyki są na tyle ważne dla prawidłowego rozwoju gospodarczego w tej dziedzinie na tyle zagrażają rozwojowi gospodarczemu i społecznemu, a zaniedbania w tej dziedzinie na tyle zagrażają utrzymaniu kraju w grupie państw cywilizowanych, iż zastosowania informatyki powinny się znaleźć pod szczególną opieką ze strony państwa. Uważamy, że jedynym rozsądnym podejściem do kompleksu spraw związanych z informatyką jest uznanie prymatu zastosowań informatyki. Polityka państwa w odniesieniu do informatyki powinna przeto wywodzić się z jasno ustalonego planu rozwoju zastosowań, a inne poczynania, takie jak planowanie produkcji urządzeń informatyki, finansowanie badań, polityka podatkowa, celna i płacowa w zakresie informatyki itp., powinny być pochodną planu rozwoju zastosowań. Uważamy, że formułować politykę państwa w tej mierze oraz skutecznie czuwać nad jej realizacją może tylko niezależny organ - rządowa agencja informatyki, niezależna od wpływów przemysłu i innych potężnych sił nacisku, wspomagana w swym działaniu przez radę złożoną z fachowych informatyków.

Jeśli zaś chodzi o bariery informatyzacji przedsiębiorstw przemysłowych z punktu widzenia problemów organizacyjnych, to chciałbym przedstawić swój pogląd poprzez próbę odpowiedzi na pytanie: czy tylko brak odpowiedniej ilości i jakości sprzętu komputerowego jest powodem niezadawalającego poziomu informatyzacji? Odpowiedź na to pytanie musi być udzielona przez pryzmat elementów składowych warunkujących efektywność informatyzacji przedsiębiorstw przemysłowych.

Główne problemy jakie muszą być rozwiązane aby można było powiedzieć, że przedsiębiorstwo zostało skomputeryzowane są następujące:

1/ precyzyjnie określony /zdefiniowany/ cel i zakres informatyzacji,

2/ wykonanie lub adaptacja projektów systemu informatycznego,

3/ odpowiednio wykonane lub dobrane oprogramowanie,

4/ organizacyjne i merytoryczne przygotowanie przedsiębiorstwa zgodnie z wymaganiami systemu informatycznego,

5/ właściwie dobrany sprzęt komputerowy i pozostały, w tym również lokalnego lub zdalnego kontaktu, uwzględniający warunki i s t n i e j ą c e j i r e a l n e j perspektywy jego jakości i dostępności jak również stanu np. sieci telefonicznej lub telegraficznej,

6/ przygotowanie użytkowników zarówno podstawowych tzn. wypełniających dokumenty zródłowe, korzystających ze szczegółowych informacji komputerowych jak i kadry kierowniczej dla

której w konsekwencji budowany jest system informatyczny,

7/ organizacja własnego ośrodka obliczeniowego /budowa nowego specjalistycznego obiektu lub adaptacja istniejących pomieszczeń/ lub też dostęp do innych zasobów mocy obliczeniowej komputerów,

8/ pozyskanie kadry specjalistów o odpowiednich kwalifikacjach /operatorów urządzeń przygotowanie danych, operatorów komputera, konserwatorów wdrażania itp/, których ilość i rodzaj zależy od przyjętego wariantu rozwiązania np. własny ośrodek lub też korzystanie z usług, wykorzystanie istniejących rozwiązań systemowych lub też opracowywanie wszystkiego siłami własnymi itd.,

9/ oddziaływanie psychiczne, społeczne, finansowe, prawne i etyczne systemu informatycznego na pracowników przedsiębiorstwa /użytkowników systemu/,

10/ efektywne pozyskanie szeroko rozumianego użytkownika systemu do prac związanych z informatyzacją i jej wykorzystaniem.

Przedstawiona kolejność problemów nie świadczy absolutnie o ich priorytetach. Uważam, że wszystkie one w odpowiednim czasie powinny mieć najwyższy priorytet. Dlatego też należy stwierdzić, że efektywny sukces informatyzacji przedsiębiorstwa jest wtedy możliwy do osiągnięcia, gdy prawie równolegle w odpowiednim czasie rozwiązywane są wszystkie wymienione problemy. Jakiekolwiek zahamowania lub próby braku jednoznacznego rozwiązania chociażby jednego z tych problemów powoduje, że system

informatyczny kuleje, nie osiąga spodziewanych efektów, następuje bardzo szybkie zniechęcenie szczególnie użytkownika któremu, co tu dużo mówić, w początkowym etapie dochodzi wiele dodatkowej pracy. Jeżeli będziemy pamiętali o tym, że przecież nie wszyscy są entuzjastami czy wręcz hobbystami nowych rozwiązań, to każde potknięcie systemu jest natychmiast w odpowiedni sposób interpretowane i wykorzystywane. I tak potem powstaje informacja a następnie prawie przekonanie, że z informatyzacją w naszym kraju jest bardzo źle, komputery /tzn. "grube"miliony złotych/ są niewykorzystywane równie z winy np..złej jakości sprzętu itp.

Analiza przyczyn trudności występujących przy wdrażaniu systemów informatycznych w przedsiębiorstwach pozwala na sformułowanie następujących wniosków:

1/ wdrażanie systemu informatycznego należy traktować jako skomplikowany proces działania w obszarze zarządzania przedsiębiorstwem,

2/ system informatyczny powoduje konieczność przeprowadzenia zmian w dotychczas funkcjonującej strukturze organizacyjnej i metodach zarządzania przedsiębiorstwem,

3/ w trakcie wdrażania systemu informatycznego następuje konfrontacja przyjętych rozwiązań z rzeczywistością istniejącą w przedsiębiorstwie,

4/ wdrażanie systemu informatycznego pozwala na sprawdzenie stopnia przygotowania przedsiębiorstwa czy kadry kierowniczej i szeregowych pracowników do korzystania w zarządzaniu z metod i narzędzi bardziej nowoczesnych niż w systemie tradycyjnym,

5/ w zdecydowanej większości przypadków system informatyczny jest wdrażany w warunkach utrzymywania istniejącego rytmu działalności przedsiębiorstwa, co powoduje nasilenie prac w komórkach organizacyjnych, których problemy mają być przedmiotem przetwarzania na komputerze,

6/ wdrażanie systemu informatycznego powoduje napięcia i nieporozumienia w sferze psychologicznej i socjologicznej wśród kadry kierowniczej i pracowników szeregowych, związane z nieporozumieniem informatyki, osobistym zagrożeniem, napływem personelu z nowymi specjalnościami itd. itp.

Powyższe wnioski dotyczą głównie problemów występujących w obszarze użytkownika. Oprócz zagadnień czysto ludzkich, które są w konsekwencji najważniejsze i chyba najtrudniejsze do rozwiązania, występują i merytoryczne. Dlatego też warto zastanowić się kolejny raz, nad koniecznością i możliwością eliminowania trudności związanych z realizacją prac przygotowawczych w przedsiębiorstwie. Pomimo, iż nastały czasy m.in. samodzielności przedsiębiorstw uważam, że jednym z podstawowych działań pozwalających na szybszą realizację prac przygotowawczych jest ujednolicanie rozwiązań systemowych w zakresie m.in. budowy symboliki danych źródłowych, formularzy dokumentów, systemów ewidencyjnych, rozliczeniowych itp. Natomiast w obszarze informatycznym dotyczy to ujednolicania rozwiązań technologii komputerowego przetwarzania jak również maksymalnego wykorzystywania gotowych i wdrożonych rozwiązań programowych.

Wydaje mi się, że jeśli chodzi o trudności i opory jakie występują przy wykorzystywaniu "nie moich" opracowań informatycznych wynikają one z następujących głównych powodów:

1/ braku odpowiednio dokładnej i prawdziwej informacji o istniejących i wdrożonych rozwiązaniach,

2/ braku odpowiedniego "narzędzia przymusu" do wykorzystywania już istniejących i wdrożonych wyselekcjonowanych rozwiązań,

3/ braku wypracowanej i zrozumiałej metody rozpowszechniania "obcych" systemów, która najczęściej jest kojarzona tylko z faktem sprzedaży systemu,

4/ braku odpowiednio wykwalifikowanej kadry informatyków, którzy by potrafili wdrażać "obce" systemy /organizatorzy wdrażania/,

5/ przeświadczeniu wielu informatyków /i chyba nie bez racji/, że wdrażanie systemów a szczególnie obcych jest o wiele trudniejsze i bardziej odpowiedzialne niż projektowanie i programowanie własne od początku do końca,

6/ trudności w porozumieniu się informatyków między sobą tzn. operowaniu różnorodnymi określeniami, co w konsekwencji doprowadza do tego, że dokumentacja systemu opracowana jest językiem niezrozumiałym nie tylko dla profesjonalistów ale tym bardziej dla użytkowników.

Oczywistą jest sprawą, że za rozwiązania wszystkich problemów występujących w obu obszarach m.in. wymienionych wyżej nie mogą odpowiadać tylko i wyłącznie informatycy /tak jak np.w przedsiębiorstwach przemysłowych narzędziownia nie odpowiada za orga-

nizację produkcji podstawowej/.

Kolejną barierą w rozwoju polskiej informatyki, a wielu specjalistów uważa, że podstawową, jest szeroko rozumiana baza techniczna a w tym przede wszystkim sprzęt komputerowy. Takie stwierdzenie spotyka się w zasadzie we wszystkich wypowiedziach publikowanych i głoszonych w naszym kraju od wielu już lat. Jeżeli większość tak mówi to, zgodnie z zasadą, chyba mają rację. I ja zgadzam się z takim poglądem jednak nie w "układzie zero-jedynkowym". Chciałbym spojrzeć na ten problem trochę szerzej i nie tylko przez pryzmat porównań ze sprzętem np. firmy IBM. Nie chcę przez to powiedzieć, że będę próbował udawadniać, iż sprzęt polskiej produkcji jest dobrej jakości. Również nie będę cytował liczb i producentów charakteryzujących stan sprzętu informatycznego i wielkość nakładów inwestycyjnych w ostatnich latach na ten sprzęt w naszym kraju. Można to znaleźć w wielu opracowaniach.

Uważam za celowe przytoczenie w tym miejscu wypowiedzi prof. Michała Skarbińskiego cytowanej we wspomnianym już poprzednio artykule Ireny Dryll: "Czytamy o rewelacyjnych osiągnięciach w dziedzinie elektroniki takich krajów jak Korea Południowa, Tajwan i Tajlandia, podczas gdy u nas króluje wydobycie węgla, a bezbłędne wykonanie licencyjnego telewizora "Jowisz" lub ciągnika "Ferguson" natrafia na nieprzezwyciężone trudności /4/. Jeżeli cytowane rozumowanie rozciągniemy na przemysł komputerowy i bazy elementowej, to bariera techniczna polskiej informatyki wydaje się bardziej trudna i złożona niż wielu uważa i kładzie na karb tylko producenta sprzętu komputerowego.

Specjaliści od techniki komputerowej twierdzą, iż gdyby była baza elementowa na wysokim poziomie to i wyrób finalny byłby wysokiej jakości. Zapewne mają rację. Stąd też bezwzględnie koniecznym jest aby zastanowili się nad skutecznym rozwiązaniem tego problemu. Ale przecież wiadomo, że na zestaw komputerowy "made in Poland", składają się wyroby dostarczane z innych krajów, również nie najwyższej jakości i jak ten problem rozwiązać?

Wracając na krajowe podwórko, alarmującą jest informacja Krzysztofa Frenczaka, iż bydgoska "Eltra" jedyny w kraju producent złączy ma zamiar zmienić grubość "pokrycia styków złotem i nazywa je "ekonomicznymi". To co dla Eltry jest ekonomiczne wcale nie oznacza, że jest dla Meramatu do przyjęcia, bo w informatyce nie ma i być nie może pojęcia drugiego gatunku czy komputera zubożonego, "ekonomicznego" /4/. I dalej Autor ten pisze "Niepokój muszą natomiast budzić opinie konstruktorów, że nie wyobrażają sobie przyszłości polskiej techniki komputerowej wyłącznie na krajowych częściach. Dostawcy Zespołów czynią różne zabiegi, aby utrzymać produkcję przy permanentnym braku odpowiednich surowców. Produkują niezgodnie z polską normą, obniżają skwapliwie cenę i pytają odbiorcę o zgodę na takie warunki. A w urządzeniach informatycznych, jak w mało których, nie można oszczędzać na funkcji użytkowej /4/.

Jeżeli do powyższego stanu bazy elementowej dołoży swoje "trzy grosze" producent wyrobu finalnego to trudno się dziwić, że użytkownik krytykuje złą jakość sprzętu komputerowego.

Jednak obserwując ten problem od lat, ciągle nie znajduję odpowiedzi na kilka nasuwających się pytań. Czym tłumaczyć, że wiedząc o takiej a nie innej jakości sprzętu komputerowego, informatycy budują wielkie systemy zarządzania przedsiębiorstwami przy założeniu prawie bezawaryjnej pracy? Statystyka awaryjności wskazuje, że sprzęt komputerowy produkcji polskiej osiąga poziom kilkunastu procent, gdy tymczasem jak stwierdził poseł Waldemar Michna na listopadowym posiedzeniu Komisji Nauki i Postępu Technicznego, "stopień wykorzystania urządzeń komputerowych wynosi średnio nie więcej niż 50 procent faktycznych możliwości. Komputery pracują niespełna 10 godzin na dobę, a minikomputery 4,8 godziny"? Dlaczego informatycy w swoich rozwiązaniach nie uwzględniają rzeczywistego stanu rzeczy tylko udają, że tego nie widzą i przyjmują założenia praktycznie nierealne? Czym wytłumaczyć, iż polski komputer w naszym kraju pracuje źle a za granicą dobrze? A może trzeba będzie przyznać rację prof. Januszowi Tymowskiemu, który mówi: "Przeciętny Polak o swój sprzęt dba i tu wykazuje zrozumienie techniki, natomiast nie dba zupełnie o budynki, urządzenia i narzędzia nie będące jego własnością i tu ujawnia niekiedy całkowity analfabetyzm techniczny. Stąd ciągnik w gospodarstwie indywidualnym pracuje kilkakrotnie dłużej niż w PGR, obrabiarka przeznaczona w zakładzie uspołecznionym na złom, u prywatnego rzemieślnika pracuje jeszcze długie lata /3/.

Nie chcę absolutnie bronić producentów sprzętu komputerowego i udawadniać, iż produkujemy komputery na światowym poziomie, a wręcz odwrotnie. Ale nie mogę pogodzić się tylko z jednostronnym

szukaniem usprawiedliwień złego stanu polskiej informatyki.

Ramy niniejszego opracowania nie pozwalają na omawianie dalszych barier, jakie pojawiają się lub są stawiane na drodze rozwoju zastosowań informatyki dla potrzeb zarządzania przedsiębiorstwami przemysłowymi. Problematyka ta jest chyba tak samo złożona jak sama informatyka chociaż uważa się, że jej główny bohater komputer, to genialny idiota bo szybko liczy a zna tylko dwie cyfry. Jeżeli tak jest, to dlaczego mamy z nim tyle kłopotów?

Zgadzam się z Krzysztofem Fronczakiem iż "polska informatyka, niewolna od paradoksów i następstw niekonsekwentnego, chaotycznego rozwoju, przeżywa bez wątpienia trudne czasy ... Reforma gospodarcza w pewnym stopniu wyklarowała sytuację obnażając niedowład tego, co tak radośnie budowano przez ostatnie kilkanaście lat /3/.

Wyciągając wnioski z dotychczasowych doświadczeń widać, że problemów do rozwiązania - barier do usunięcia na drodze rozwoju polskiej informatyki jest wiele. Wydaje mi się, że dość szeroki katalog tych problemów zawarty jest we wspomnianym już referacie sprawozdawczym Zarządu Głównego Polskiego Towarzystwa Informatycznego. Należy obecnie doprowadzić do tego aby prawdziwi specjaliści informatycy i z innych dziedzin wypracowali praktyczne i realne metody rozwiązywania tych problemów. Bo aczkolwiek jak stwierdził na peselskim forum dyrektor Instytutu Problemów Informatyki PAN Zenon Szoda mamy słaby sprzęt i brak jest koordynacji różnych przedsięwzięć, to jednak "główny problem polega na tym, że potrzeby zastosowań informatyki nie wywoła się dotąd, dopóki w

polskiej gospodarce nie zaczną funkcjonować mechanizmy racjonalnego i efektywnego gospodarowania. Każda gospodarka dostatecznie rozwinięta, która nie przyswaja informatyki - jest gospodarką chorą" /3/. Nic dodać nic ująć.

Literatura

/1/ Biuletyn PTI nr 4 kwietnia 1984 r.,

/2/ Irena Dyrll, Praca na cenzurowanym, Życie Gospodarcze nr 22 z dnia 27.05.1984 r.,

/3/ Życie Gospodarcze nr 19 z dnia 16.05.1984 r.

/4/ Życie Gospodarcze nr 22 z dnia 27.05.1984 r.

Zdzisław Szyjewski [1]

METODA PROJEKTOWANIA I REALIZACJI OPROGRAMOWANIA ZŁOŻONEGO SYSTEMU INFORMATYCZNEGO NA PRZYKŁADZIE SYSTEMU TRANSTER

1. Wprowadzenie

Lata siedemdziesiąte dostarczyły wiele nowych metod projektowania i programowania systemów informatycznych. Metody te najczęściej powstawały w wyniku konkretnej potrzeby rozwiązania złożonego zadania projektowego. Publikacje wskazywały na ich zalety oraz uniwersalność, pozwalającą na ich stosowanie przy rozwiązywaniu innych, niekoniecznie podobnych problemów. Publikowane wyniki badań nad stosowalnością tych metod, pokazują mniejsze niż się można było spodziewać zainteresowanie praktyczne tymi metodami. Obserwacja doświadczeń krajowych mimo braku reprezentatywnych badań, wykazuje również tylko incydentalne zainteresowanie tymi metodami, szczególnie jeśli idzie o systemy przetwarzania danych.

Przyczyny takiego stanu są bardzo złożone i nie wynikają tylko z nieznajomości tych metod. Stosowanie tych metod, powstałych pod inne potrzeby, wymaga najczęściej modyfikacji i uzupełnień wynikających z występujących ograniczeń organizacyjnych, sprzętowych i narzędziowych.

Projektowanie systemu koordynacji przewozów dalekich ciężarowego transportu samochodowego było dobrym obiektem stosowania nowoczesnych metod projektowania i programowania w związku

---

1/ Instytut Informatyki i Zarządzania Politechniki Szczecińskiej.

z rozmiarem problemu i dużymi wymaganiami niezawodnościowymi. Doświadczenia praktyczne ze stosowania nowoczesnych metod projektowania i programowania przy projektowaniu i programowaniu tego systemu, są tematem dalszych szczegółowych rozważań.

2. Definicja problemu

Eliminowanie pustych przebiegów pojazdów ciężarowych, to główna idea systemu TRANSTER, opracowanego dla ciężarowego transportu samochodowego zorganizowanego w PKS i eksploatowanego od kilku lat z zadowalającymi efektami. Potrzeba rozszerzenia funkcjonalnego i przedmiotowego, wdrożenie zmian wynikających z kilkuletniej praktyki eksploatacji i wreszcie konieczność wymiany sprzętu wykorzystywanego w systemie informatycznym, to przesłanki które skłoniły PKS do podjęcia prac nad nową rozszerzoną wersją systemu.

Problem postawiony przed zespołem projektującym został najogólniej zdefiniowany, jako koordynacja przewozów dalekich w ciężarowym transporcie samochodowym. Jednostką zlecającą i po opracowaniu eksploatującą, jest PKS. Jednostką wykonującą jest zespół Politechniki Szczecińskiej. Autorzy systemu nie mając obciążeń praktyki transportowej, znają problemy transportu samochodowego, w którym PKS jest znaczącą, ale tylko częścią, Zleceniodawca jest zainteresowany praktycznym rozwiązaniem problemu ze swojego punktu widzenia, który nie we wszystkich szczegółach zgadza się z interesami ogólnospołecznymi transportu samochodowego.

Wyjściem z tak powstałego konfliktu merytorycznego okazało się "nadlerowskie" podejście do procesu projektowania. Autorzy systemu definiują idealny model funkcjonowania systemu, realizujący wszystkie teoretycznie możliwe do wystąpienia sytuacje przewozowe i określają algorytmy ich obsługi. W wyniku długotrwałych dyskusji z użytkownikiem systemu znajduje się model praktyczny, który jest wypadkową stanowisk obu stron.Model

ten nie we wszystkim zgadza się z teorią ale zawiera pewne sytuacje nadmiarowe, które z założenia użytkownik nie będzie wykorzystywał.

Określonych zostało sześć następujących sytuacji przewozowych, gdzie przez A oznacza się miejscowość macierzystą dla danej jednostki organizacyjnej, transportowej lub spedycyjnej a punkt B miejscowość skąd lub dokąd ma być wykonany przewóz transportowy.

1° - pojazd ma przewieźć określony ładunek z miejscowości A do miejscowości B,

2° - pojazd wyjeżdża bez ładunku z miejscowości A do miejscowości B, skąd ma przywieźć do miejscowości A określony ładunek.

3° - pojazd w miejscowości A jest wolny do dyspozycji na dowolny kierunek, ale przewiezienia masy ładunkowej.

4° - w miejscowości A występuje nadmiar masy ładunkowej, dla której nie ma pojazdów do jej wywiezienia.

5° - pojazd wyjechał z ładunkiem do miejscowości B i nie ma zapewnionego ładunku na drogę powrotną.

6° - w miejscowości B występuje masa ładunkowa, którą trzeba przywieźć do miejscowości A.

Te sześć sytuacji przewozowych wyczerpuje w pełni możliwe teoretycznie i praktycznie sytuacje transportowe [1/]. W zależności od specyfiki przewoźnika występują tylko niektóre sytuacje a inne z założenia nie występują lub są incydentalne. Algorytm postępowania dla każdej sytuacji musi być pełny i kompletny niezależnie od częstości występowania danej sytuacji przewozowej oraz co jest również bardzo istotne musi być niezależny od praktyki i przyzwyczajeń konkretnego użytkownika systemu transportowego.

1/ Brak tutaj oczywiście tzw. przebiegów zamkniętych, czyli sytuacji gdy pojazd ma ładunek na jazdę do jednostki docelowej oraz na jazdę powrotną, ale sytuacja taka z punktu widzenia koordynacji jest nieistotna, dlatego też została pominięta.

Przy tak określonym problemie, efekty systemu będą wzrastały wraz ze wzrostem ilości komplementarnych użytkowników transportu. Bogatszy zbiór zgłoszonych sytuacji przewozowych daje większe możliwości koordynacji. System informatyczny analizując zbiór zgłoszonych sytuacji przewozowych eliminuje jazdy przeciwbieżne, realizując je jednym pojazdem, wyszukuje ładunki dla pojazdów, które nie posiadają ładunku na części trasy, czy też dla wolnej masy ładunkowej wyszukuje pojazdy nie mające okresowo zatrudnienia. Kojarzenie sytuacji przewozowych komplementarnych nie ogranicza się tylko do miejscowości A i B ale również do ich geograficznego otoczenia i trasy przejazdu wyznaczonej maszynowo.

W związku z tym, że dla danej sytuacji przewozowej możliwe jest więcej niż jedno skojarzenie, należy wybrać skojarzenie najbardziej efektywne. Według specjalnej procedury obliczeniowej wybierane jest skojarzenie, które ma największy współczynnik efektywności z punktu widzenia wykorzystania ładowności pojazdu oraz jak najmniej dojazdów bez ładunku.

W szczegółowych dyskusjach z użytkownikiem systemu, szczególnie trudne było określenie zakresu systemu. Z uwagi na wyniki systemu w postaci planów pracy dla poszczególnych jednostek organizacyjnych, użytkownik chciałby widzieć w systemie funkcje decyzyjne i wykazywał niedoskonałości tych rozwiązań systemowych w porównaniu z sytuacją tradycyjną. Autorzy wykazywali, że decyzje przewozowe zapadają w momencie określenia konkretnego typu zlecenia dla danej sytuacji przewozowej i te decyzje są dopiero obiektem pracy systemu. System nie decyduje a jedynie koordynuje podjęte przez dyspozytorów decyzje przewozowe.

Takie określanie zakresu systemu stawia go na najwyższym poziomie w hierarchii decyzji przewozowych. Usytuowanie to pozwala na zmianę przez system, decyzji, niższego poziomu, czyli dyspozytora jednostki organizacyjnej. Zmiana decyzji następuje w wyniku analizy większej ilości przewozowych, w powiązaniu z rozmieszczeniem geograficznym i obliczeniami efektywności ekonomicz-

nej, nie jednej jednostki organizacyjnej a globalnie w skali ogólnospołecznej.

Definicji problemu poświęcono wyjątkowo dużo czasu i uwagi, co dało efekt w dalszych etapach prac nad systemem. W trakcie definiowania zadania projektowego powstał problem metod specyfikacji. Metody opisu słownego były mało przydatne z uwagi na nieczytelność powiązań i uwarunkowań różnorodnych sytuacji przewozowych.

3. Specyfikacja zadania projektowego

Precyzyjny opis założeń systemu /specyfikacja/ ma bardzo istotne znaczenie nie tylko w fazie realizacji projektu ale również, a może przedewszystkim, na etapie konserwacji projektu. Prawidłowa specyfikacja zadania projektowego pozwoli na wczesne wykrycie i wyjaśnienie elementów wątpliwych lub niejednoznacznych.

Przy realizacji omawianego projektu oddzialnie specyfikowano dane z wykorzystaniem pakietu PSL/PSA, oraz oddzielnie algorytmy wykorzystując metodę projektowania programów Jacksona. Walory obu zastosowanych narzędzi są znane, skoncentrujemy się więc na uwarunkowaniach konkretnego zastosowania oraz potrzebach modyfikacji.

Stosunkowo niewielka różnorodność danych oraz ich niewielka ilość nie pozwoliła na pełne skorzystanie z zalet PSL /PSA. Zastosowanie tego pakietu wprowadziło jednak pewną dyscyplinę w operowaniu danymi, w stosowanym nazewnictwie, oraz umożliwiło stosowanie w pełni biblioteki wspomagającej w odniesieniu do danych. Wszystkie dane były wykorzystywane w programach, jako przyłączane przy pomocy operatora %INCLUDE w języku PL/1, moduły źródłowe, opisane według jednolitych zasad, z nazwami stosowanymi według jednolitego standardu, z obszernym komentarzem opisującym zawartość zbiorów, rekordów i pól elementarnych.

Opis danych wykonany zgodnie z zasadami pakietu PSL, PSA, nie pozwolił jednak na pełne wyspecyfikowanie zależności i powiązań między danymi. Potrzeba ukrycia w danych w sposób niejawny położenia geograficznego, dla późniejszego określania trasy przejazdu, liczenia długości marszrut i określania sąsiedztwa, zmusiło do stosowania wyszukanych organizacji zbiorów i odpowiednich do nich metod dostępu do danych. Te elementy nie mogły być precyzyjnie wyspecyfikowane przy użyciu aparatu dostępnego w pakiecie PSL/PSA. Dla umożliwienia łatwego określenia powiązań między danymi, zdefiniowany został oddzielny zbiór zawierający tablicę inwersyjną opisującą te związki. Zastosowanie tego zbioru wraz z procedurami jego obsługi, kolosalnie uprościło problem komunikowania się między danymi , zlokalizowanymi w różnych zbiorach.

Dla specyfikacji algorytmu postępowania przy kojarzeniu zleceń przewozowych wykorzystano metodę Jacksona. Metoda Jacksona projektowania programów zakłada, że w pierwszym kroku specyfikuje się dane w postaci diagramów strukturalnych i dopiero na ich podstawie buduje się diagram strukturalny definiujący problem do rozwiązania. Podejście takie jest prawidłowe w większości przypadków przetwarzania danych. W omawianym projekcie, struktury danych możliwe do opisania w diagramie strukturalnym przy użyciu struktur sekwencji, wyboru i powtarzania, były trywialne i nie oddawały specyfiki problemu. W związku z tym zrezygnowano z tego kroku metody, przystępując od razu do tworzenia diagramu strukturalnego opisującego główny algorytm projektowanego systemu. Przy czym biorąc pod uwagę jego dużą złożoność zastosowano kilka poziomów szczegółowości. Podejście takie choć trudniejsze w realizacji, było wydaje się jedynym możliwym, jeśli chciało się skorzystać z zalet specyfikacji Jacksona.

Specyfikacja problemu z wykorzystaniem metody Jacksona pozwoliła na precyzyjny i sugestywny opis głównych idei algorytmów. Przedstawienie zasad funkcjonowania algorytmu na prostym rysunku /por. rys. 1/, pozwoliło po krótkim objaśnieniu,

wniknąć w dalsze szczegóły funkcjonowania algorytmu nie tylko informatykom nie obeznanym z metodą Jacksona, ale również użytkownikom systemu nie posiadającym przygotowania informatycznego. Graficzny obraz algorytmu pozwolił na prezentację zasad postępowania z uwzględnieniem skali czasu operowania algorytmem, na wystarczająco dużym poziomie ogólności. Ten sposób prezentacji zmuszał również do selektywnego myślenia pozwalającego zgłębić wszystkie niuanse możliwych do wystąpienia sytuacji.

Specyfikacja algorytmu postępowania, nie może być jednak określona w oderwaniu od danych na których algorytm operuje. Specyfikacja danych mimo że nie została wykonana zgodnie z założeniami metody Jacksona, musiała być podana w formie komentarza przed analizą diagramu opisującego algorytm. Szczególnie zasada połączenia klucza dostępu do danych z położeniem geograficznym i identyfikatorami zleceń przewozowych, musiała być znana osobom posiadającym przygotowanie informatyczne. Nie posiadający takiego przygotowania przechodzili nad tym do porządku dziennego, ale nie mogli się wgłębiać we wszystkie szczegóły postępowania. W związku z tym w zależności od stopnia wiedzy informatycznej, specyfikacja tak stworzona była dostępna na różnych poziomach szczegółowości.

4. Realizacja oprogramowania z wykorzystaniem ulepszonych metod programowania

Ulepszone metody programowania stanowią zbiór metod i technik programowania, których podstawowym celem jest wzrost jakości oprogramowania, zwiększenie efektywności prac programowych oraz ułatwienie konserwacji i modyfikacji oprogramowania. Oprogramowanie systemu TRENSTER, z uwagi na ścisły reżim czasowy przetwarzania w cyklu dziennym oraz potrzebę szybkiego reagowania na różnorodne zmiany i modyfikacje, musi się charakteryzować wysoką jakością i niezawodnością działania oraz łatwością modyfikowania.

Z uwagi na złożoność algorytmu głównego systemu, nie było możliwości podzielenia go na samodzielne programy bez całości realizacji musiała być wykonana przez jeden program. Program ten z założenia musiał być duży i niezawodny a właśnie dla tworzenia takich programów służą ulepszone metody programowania. Główne cechy ulepszonych metod programowania z których skorzystano przy opracowywaniu tego programu to:

- podział problemu na funkcjonalne moduły programowe,
- projektowanie struktury logicznej programów i modułów w sposób zdyscyplinowany z wykorzystaniem struktur sekwencji, wyboru i powtarzania,
- kodowanie zgodnie ze szczegółowymi zasadami metodycznymi, zwiększającymi czytelność kodu źródłowego /por. /3//,
- operowanie danymi w sposób zdyscyplinowany,
- testowanie w sposób zstępujący,
- organizacja prac programowych na wzór zespołu programisty wiodącego.

W pracach tych tworzeniu oprogramowania szczególną uwagę zwrócono na prawidłowe projektowanie programu wykorzystując metodę Jacksona, oraz podzielenie programu na moduły funkcjonalne z uwzględnieniem operowania danymi. Posiadając wcześniej opracowaną szczegółową specyfikację danych oraz moduły opisujące je, wykorzystywano bibliotekę wspomagającą prace programowe.

Podstawową metodą zmniejszenia złożoności programu jest rozłożenie go na zbiór niewielkich, wysoce niezależnych modułów. Rozkład taki nie może być przypadkowy ale zgodny ze strukturą problemu, który ma być rozwiązany. W konkretnym przypadku systemu TRANSTER rozkład taki nie był w pełni możliwy, ale udało się wydzielić liczne fragmenty, które można było traktować jako moduły programowe. Dokonując tego podziału szczególną uwagę zwracano na niezależność modułu, czyli maksymalizowano po-

wiązania wewnątrz modułowe a minimalizowano, w miarę możliwości, powiązania między modułowe.

Definiując moduły programowe udało się w większości przypadków osiągnąć moc funkcjonalną według skali podanej przez G. J. Myersa /por. /2//. W nielicznych przypadkach musiano zdefiniować moduły o słabszej mocy to znaczy o mocy informacyjnej i algorytmicznej. Zdając sobie sprawę z dużej wagi powiązań międzymodułowych, na niezawodność programu, szczególną uwagę poświęcono problemowi więzi modułowej wyrażającej się w dostępności do danych przez poszczególne moduły. Generalnie starano się zrezygnować z danych globalnych dla większej grupy modułów, czyli z tak zwanej więzi wspólnej /por. 2 str. 93/ na rzecz więzi danych, przekazywanych w postaci parametrów w instrukcji wywołania.

Konsekwentne stosowanie przekazywania danych przez parametry wywołania, miało istotny wpływ na organizację deklaracji i dostępność danych. W nielicznych przypadkach, kiedy ilość przekazywanych danych byłaby wyjątkowo duża, skorzystano z ograniczonej dostępności nazw danych w języku PL/1. Moduły, programowe były kodowane jako procedury języka PL/1 i włączane w kod źródłowy nie w postaci ładowalnej na etapie łączenia ale jako procedura wewnętrzna dołączona na etapie kompilacji przy użyciu operatora %INCLUDE. Rozwiązanie to stwarzało pewne trudności na etapie testowania tak kodowanego modułu, ale mimo wzrostu pracochłonności było to opłacalne.

Część modułów o mocy funkcjonalnej i więzi danych zakodowano jako procedury funkcyjne, co w istotny sposób uprościło ich wykorzystanie i zwiększyło czytelność kodu źródłowego realizującego algorytm.

Konsekwentne stosowanie metody Jacksona w pracach projektowania struktury programu musiało być zmodyfikowane do możliwości dostępnych narzędzi programowych. Zastosowanie języka programowania PL/1, który ma określone możliwości wspomagania za-

sad programowania strukturalnego, decydowało o takich a nie innych rozwiązaniach szczegółowych. Szczególnie dotkliwy okazał się brak instrukcji wyboru wielokrotnego. Realizacja wyboru wielokrotnego przy pomocy dostępnego aparatu instrukcji z wykorzystaniem instrukcji skoku bezwarunkowego nastręczała spore kłopotów.

Opracowanie listy operacji, które następnie należało kodować na diagram strukturalny programu nasunęło spore wątpliwości głównie z określeniem stopnia szczegółowości operacji. W liście występowały operacje zerowania pojedyńczych pól obok operacji złożonych, realizowanych przez moduł programowy. Rozmiar programu i związana z tym duża ilość operacji, nastręczały problem ze sporządzeniem pełnej listy. Doświadczenia posiadane z przykładów dydaktycznych opisujących metodę Jacksona, w tym zakresie przydały się niewiele. Dopiero wydzielenie pewnych grup operacji i dla nich sporządzenie list cząstkowych dało spodziewany efekt.

Z uwagi na zaangażowanie w pracach nad jednym programem grupy kilku programistów z różną wiedzą i praktyką programowania, trudne było osiągnąć pełne zdyscyplinowanie kodowania programu. Czytelność kodu źródłowego osiągnięto częściowo dzięki wymogom organizacyjnym, a głównie dzięki opracowaniu bogatej biblioteki wspomagającej, której wykorzystanie wspomagało ponadto proces kodowania.

5. Organizacja prac

W przypadku gdy na cel ostateczny składa się działanie grupy osób, istotne znaczenie ma dobra organizacja prac zespołu realizatorów. W konkretnym przypadku realizowanego systemu skorzystano ze wzorca znanego jako "zespół głównego programisty" /1/.

Zespół realizujący oprogramowanie składał się z czteroosobowego trzonu oraz programistów wspomagających. W czteroosobowym trzonie zespołu obsadzone zostały funkcje "głównego progra-

misty" i "sekretarza", pozostałe dwie osoby pełniły funkcje programistów samodzielnie realizujących dość duże wydzielone części oprogramowania. Ponadto zespół był wspomagany przez programistów realizujących pojedyncze moduły według ściśle określonej specyfikacji i z określonymi wymogami organizacji modułu, oraz przez konsultantów problemu, wykorzystywanych głównie na etapie testowania, i konsultantów języka PL/1 dla opiniowania sytuacji szczególnie złożonych lub specyficznych. Wszystkie prace były realizowane i weryfikowane przez programistę głównego, który ponadto opracowywał najważniejsze części oprogramowania. Sekretarz zespołu głównie zajmował się sprawami administracyjnymi i dokumentacyjnymi stąd jego udział w tworzeniu oprogramowania był minimalny. W zespole nie było zastępcy głównego programisty, którego brak dał się odczuć w niektórych fazach realizacji oprogramowania, szczególnie gdy należało przedyskutować jakąś złożoną sytuację i podjąć istotną decyzję projektową.

W związku z takim składem zespołu duża część istotnych prac projektowych i większość decyzji mających wpływ na realizację procesu programowania, pozostawało w gestii głównego programisty. Specyfikacja danych z wykorzystaniem pakietu PSL/PSA została przekazana do wykonania sekretarzowi, który ponadto we wstępnej fazie tworzenia oprogramowania założył i utrzymywał w stanie aktualnym bibliotekę wspomagającą.

W bibliotece tej w pierwszej kolejności utworzono moduły opisujące zbiory wykorzystywane w systemie [1/]. Następnie po wykonaniu podziału na moduły funkcjonalne, kodowano je, testowano i umieszczano w bibliotece. W pierwszej kolejności programowano moduły mające najszersze zastosowanie oraz będące na najniższym poziomie struktury hierarchicznej. W momencie kiedy przystępowano do głównych prac programowych w bibliotece były umieszczone opisy wszystkich zbiorów oraz procedury funkcyjne i moduły programowe realizujące wiele funkcji pomocniczych. Przystępując do

1/ Pod pojęciem "zbiór" rozumie się tutaj również opis dokumentów wejściowych oraz tabulogramy.

prac programowych całą uwagę można było poświęcić prawidłowemu rozwiązaniu problemu, bez potrzeby zajmowania się nieistotnymi szczegółami. Deklaracje danych ograniczały się tylko i wyłącznie do opisu danych wykorzystywanych jako robocze w programie a wszystkie opisy danych zewnętrznych były włączane z biblioteki wspomagającej na etapie kompilacji programu.

Dla wytestowania niezależnych modułów nie posiadano żadnego oprogramowania wspomagającego i należało to wykonywać metodami tradycyjnymi. Do testowania przystąpiono współbieżnie od modułów najniższego poziomu do coraz to wyższych poziomów oraz od testowania struktury całego oprogramowania do poziomów niższych. Testowanie z góry w dół wykonywane z wykorzystaniem modułów ślepych, w miejscach występowania modułów jeszcze nie opracowanych.

Etap testowania wykazał niekonsekwencje oraz niedoróbki wcześniej niezauważone. Brak dobrego narzędzia wyryfikacji modułów spowodował, że błędne funkcjonowanie było wykrywane często na poziomie wyższym. Tego typu błędy były najtrudniejsze do wykrycia, gdyż z założenia przyjmowało się poprawność funkcjonowania modułów z niższego poziomu, włączanych już po ich sprawdzeniu. Nie prowadzono szczegółowej statystyki błędów, ale większość błędów to błędy bardzo proste, typu brak wyzerowania pola roboczego, czy zwiększenia wskaźnika. Zastanawiała poprawność najtrudniejszych partii oprogramowania bezpośrednio po usunięciu błędów formalnych. Generalnie można wysunąć wniosek, że ulepszone metody oprogramowania pozwoliły na poprawne rozwiązanie problemu, a powstałe błędy wynikały z niekonsekwencji ich stosowania.

6. Wnioski

Z doświadczeń projektowania i programowania systemu koordynacji ciężarowego transportu samochodowego TRANSTER, można sformułować kilka wniosków mających zastosowanie przy projektowaniu dużych systemów oprogramowania:

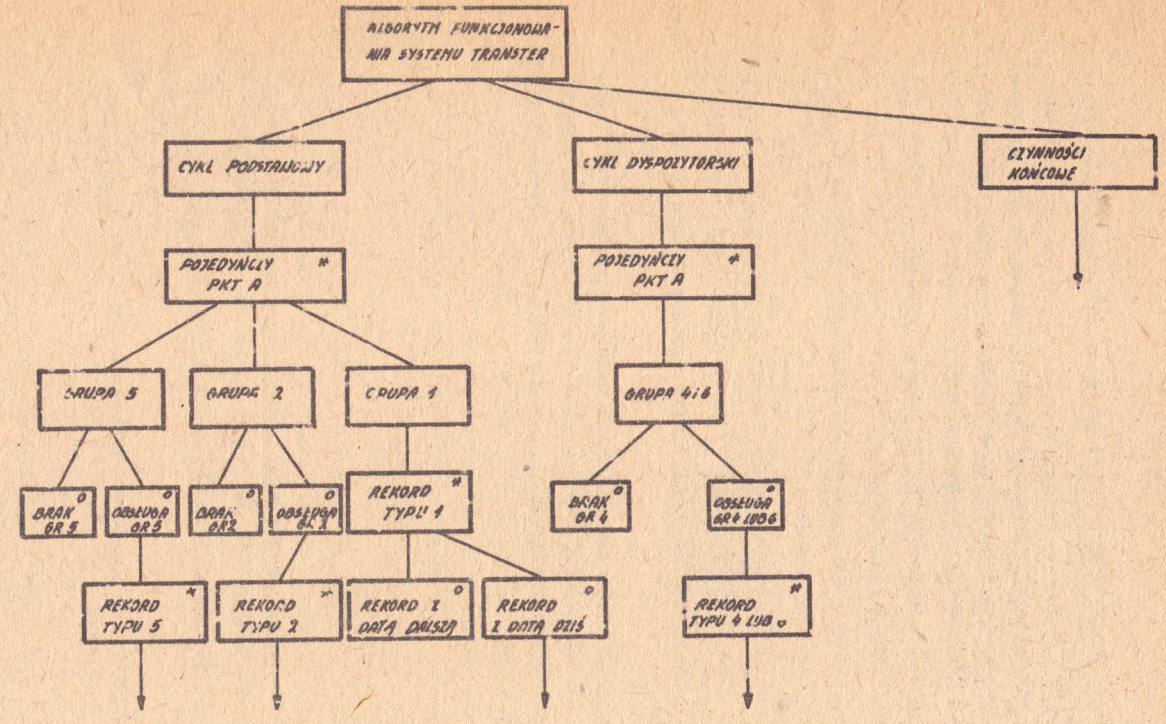
- opłacalne jest projektowanie i realizacja systemów "nadmiarowych" i otwartych,
- poświęcenie szczególnej uwagi definicji problemu oraz pełna i możliwie jednoznaczna specyfikacja zadania projektowego pozwala na sprawną realizację dalszych prac,
- stosowanie narzędzi wspomagających wpływa na szybkość realizacji prac projektowych i programowych,
- stosowanie biblioteki wspomagającej wprowadza korzystną dyscyplinę konstrukcji oprogramowania,
- dostępność danych i zorganizowany sposób komunikowania się między modułami mają istotny wpływ na niezawodność oprogramowania,
- istotny wpływ na sprawność realizacji ma organizacja i kierowanie zespołem,
- umiejętne wykorzystanie kwalifikacji członków zespołu i bezwzględne egzekwowanie przyjętych zasad postępowania, ma istotny wpływ na szybkość realizacji ale również na niezawodność produktu końcowego

Bibliografia

/1/ F.T. Baker, Chief programmer team management of production programming, IBM System Jouvnal, 1972, 11, 1, s. 56-73.

/2/ G.J. Myers, Projektowanie niezawodnego oprogramowania, WNT W-wa 1980.

/3/ [illegible]Z. Szyjewski, Jak kodować strukturalnie?, materiały konferencji INFOGRYF-80.



RYS. 1. SCHEMAT FUNKCJONOWANIA ALGORYTMU SYSTEMU TRANSTER

W.M. Turski [1/]

MIEJSCE SPECYFIKACJI W STRUKTURZE OPROGRAMOWANIA

Uwaga: niniejsze streszczenie powołuje się na numerowane foliogramy.

Foliogram 1 przedstawia ewolucję poglądów na wzajemne odpowiedniości różnych podstawowych składowych procesu budowy oprogramowania i twórczości matematycznej.

Foliogram 2 przedstawia elementy rozumowania pozwalającego stwierdzić, że Hehnerem i Abrialem, że "program jest relacją".

Foliogram 3 przedstawia podstawowe pojęcia składowe systemu lingwistycznego i teorii. Wprowadza się pojmowanie specyfikacji jako teorii.

Foliogram 4 wprowadza pojmowanie programu jako dziedziny, która może być modelem teorii /specyfikacji/. Zostaje sformułowane podstawowe zadanie programistyczne jako równanie w systemach lingwistycznych.

Foliogramy 5 i 6 przedstawiają kilka metod rozwiązywania podstawowego zadania programistycznego: interpolację, metodę kolejnych uściśleń i metodę rzymską.

Foliogramy 7-10 przedstawiają przykłady dwu teorii "recenzowania prac", oraz programu "recenzent" poprawnego w sensie przytoczonej "szczegółowej" teorii recenzowania.

Foliogram 11 ilustruje schemat powiązań między "światem realnym", "specyfikacją" i "programem".

1/ Instytut Informatyki UW

Przedstawiony pogląd na rolę i miejsce specyfikacji został opracowany na podstawie prac autora:

- Desing of large programs. w: Software Engineering - Entwurf und Spezifikation /C. Floyd und H. Kopetz, Hrsg./ B.G. Teubner, Stuttgart, 1981, 129-160,
- Specification as a theory wirt models in the computer world and in the real world. w: INFOTECH State of Art Raport "System Design", 1981, 363-377
- On what exactly goes on whoen software is developed stop-by-stop. /wspólna praca z T.S.E. Maibaumem/. Proc. 7 Int. Conf. on Software Engineering, Orlando, 1984, 528-533.
- Another look at software design methodology /wspólna praca z M. M. Lehmanem i V. Stenningiem/, Raport DOC Imperial College 83/13, 1983.

Matematyka	Programowanie
Twierdzenie	Program
Dowód	Weryfikacja

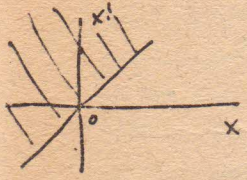
Twierdzenie	specyfikacja	De Millo
Dowód	program	Lipton
		Perlis
urojony		CACM 1979
dowód formalny	weryfikacja	Maj 271-280

Problem	specyfikacja	
Twierdzenie	Program	Scherlis
Dowód		Scott
	wytworzenie programu	IFIP 83

$>$

$x' > x$

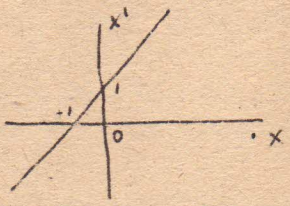
$>(x', x) \subset X' \times X$



$x := x + 1$

$x\,(x := x+1)\,x'$

$(x, x') \in R \times R$



samość	x costam x'	$x = x'$
ienie	$x\ P_1 ; P_2\ x'$	$\exists z ((x P_1 x')[z/x'] \,\&\, (x P_2 x')[z/x])$
ór	$x\ C \to P\ x'$	$C \,\&\, x P x'$
br	$x\ P \to C\ x'$	$x P x' \,\&\, C[x'/x]$
ernacja	$x\ P_1 \square P_2\ x'$	$x P_1 x' \vee x P_2 x'$
niunkcja	$x\ P_1 \,\&\, P_2\ x'$	$x P_1 x' \,\&\, x P_2 x'$
eracja	$x\ P^n\ x'$	$x R^n x'$
		$R = \{(x, x') : x P x'\}$
mknięcie	$x\ P^*\ x'$	$\exists n\ (x P^n x')$

PROGRAM JEST RELACJA

HEHNER
ABRIAL

SYSTEM LINGWISTYCZNY

1. Gramatyka z typami (kategoriami)
   Reguły poprawnej budowy

2. Logika
   Reguły wnioskowania + aksjomaty logiczne

TEORIA

Zbiór pozalogicznych aksjomatów A
w pewnym systemie lingwistycznym

$Cn(A)$ zbiór konsekwencji wyprowadzalnych z A w danym systemie lingwistycznym

TEORIA SPRZECZNA

$$\exists s: s \in Cn(A) \& \neg s \in Cn(A)$$

TEORIA PRZYZWALAJĄCA (PERMISSIVE)

$$\exists s: s \notin Cn(A) \& \neg s \notin Cn(A)$$

SPECYFIKACJA JEST TEORIĄ

- NIESPRZECZNĄ — OBY!
- PRZYZWALAJĄCĄ — ZAZWYCZAJ

KONKRETNY PROGRAM JEST DZIEDZINĄ : ZBIÓR OBIEKTÓW
RELACJI
FUNKCJI

$E(D)$ zbiór prawd (faktów) o dziedzinie

DZIEDZINA JEST MODELEM TEORII

istnieje taka interpretacja

Int
zmienne → obiekty
predykaty → relacje
symb. funkcji → funkcje

że każde zdanie z $Cn(A)$ w tej interpretacji staje się jednym z faktów dziedziny

MOŻE SIĘ ZDARZYĆ, ŻE DZIEDZINA JEST FORMALNA

$$E(D) \equiv Cn(Ax(D))$$

$$\vdash_D \mathfrak{s} \triangleq \mathfrak{s} \in Cn(Ax(D))$$

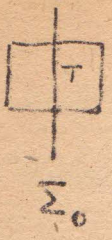
TEORIA T_2 JEST MODELEM TEORII T_1

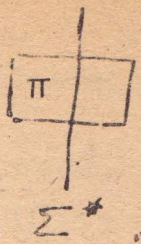
$$(\forall \mathfrak{s}) \vdash_{T_1} \mathfrak{s} \Rightarrow \vdash_{T_2} Int(\mathfrak{s})$$

ZADANIE PROGRAMISTYCZNE

ZADANE : 1. SYSTEM LINGWISTYCZNY Σ_0
2. SPECYFIKACJA (TEORIA) Γ
3. SYSTEM LINGWISTYCZNY Σ^*

ZNALEŹĆ 1. MODEL T w Σ^*
(2. ODPOWIEDNIA INTERPRETACJE)

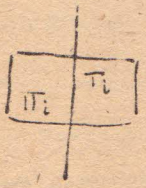


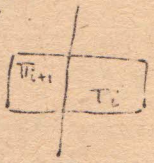


$$(\forall s \in Cn(T))\,(Int(s) \in E(\pi))$$

METODA OGÓLNA

ITERACYJNE POSTĘPOWANIE





Σ_i Σ_{i+1}

ZADANE : Σ_i, T_i

WYMYŚLONE : Σ_{i+1}, $Int : \Sigma_i \rightarrow \Sigma_{i+1}$

ZNALEŹĆ : π_{i+1}

METODA KOLEJNYCH UŚCIŚLEŃ
(OGRANICZANIA PRZYZWALALNOŚCI)

$T_i = Cn(Ax(T_i))$

$Ax(T_i) = A_{i1}, A_{i2}, \ldots, A_{in_i}$

$T_{i+1} = Cn(Ax(T_{i+1}))$

$Ax(T_{i+1}) = A_{i+1,1}, \ldots, A_{i+1,n_{i+1}}$

$(\forall j \in [1..n_i]\ \exists p,q,\ldots,r)\ Int(A_{ij}) \in Cn(A_{i+1,p}, A_{i+1,q}, \ldots, A_{i+1,r})$

$\therefore\ 1 \in Cn(Ax(T_i)) \Rightarrow Int(1) \in Cn(Ax(T_{i+1}))$

METODA (PSEUDO) RZYMSKA (DIVIDE & CONQUER)

$T_i \subset T_{i1} \cup T_{i2} \cup \ldots \cup T_{in_i}$

Dla każdej T_{ij} z pokrycia budujemy oddzielny model w 'swoim' poziomie lingwistycznym

- budujemy zachowawcze połączenie poziomów
- budujemy zachowawcze połączenie teorii - modeli (Twierdzenie Robinsona)

PRZYKŁAD

OGÓLNA TEORIA RECENZOWANIA PRAC

typ plik = ciąg (max_prac) obiektów typu praca;

const max_prac = do ustalenia;

typ praca = ciąg (max_słów) obiektów typu słowo;

const max_słów = do ustalenia;

typ słowo = do ustalenia;

typ kryteria = do ustalenia;

funkcja zaakceptowana (p: praca; c: kryteria): logiczna =
do ustalenia;

funkcja prace_recenzowane (r, s: plik, c: kryteria): logiczna =
(∀ p: praca)(p ∈ r ≡ (p ∈ s & zaakceptowana (p, c))

① aksjomat o pustych plikach

(∀ c: kryteria)(prace_recenzowane (null (plik), null (plik), c) ≡ true)

② aksjomat o przyjęciu pracy

(∀ r, s: plik; ∀ p: praca; ∀ c: kryteria)
((prace_recenzowane (r, s, c) & zaakceptowana (p, c)
⇒ prace_recenzowane (r # p, s # p, c))

③ aksjomat o odmowie pracy

(∀ r, s: plik; ∀ p: praca; ∀ c: kryteria)
((prace_recenzowane (r, s, c) & nie zaakceptowana (p, c)
⇒ prace_recenzowane (r, s # p, c)).

Bardziej szczegółowa teoria recenzowania prac

typ kryteria = ciąg(max_kryt) obiektów typu słowo;

const max_kryt = do ustalenia;

funkcja należy(w: słowo; c: kryteria): logiczna =
(∃i: [1.. max_kryt])(a = kryteria[i])

funkcja brak_błędów(p: praca; c: kryteria). logiczna =
(∀w: słowo)(w ∈ p ⇒ nie należy(w, c))

⊙ aksjomat o pustych pracach
brak_błędów(mk(praca), c) ≡ true

⊙ aksjomat o poprawnych słowach
(∀p: praca; ∀w: słowo; c: kryteria)
((brak_błędów(p, c) & nie należy(w, c))
⇒ brak_błędów(p#w, c))

⊙ aksjomat o błędnych słowach
(∀p: praca; ∀c: kryteria)
((∃w: słowo; w∈p & należy(w, c) ⇒ nie brak_błędów(p, c))

Program recenzent [illegible]

procedura recenzowanie (var r: plik, s: plik; c: kryteria) =

poczatek {prace_recenzowane (r, s, c)}

var i: całkowite;

i := 0;

r := null (plik);

petla {prace_recenzowane (r, s[1..i], c)};

jeśli i = max_prac to zakoncz;

i := i + 1;

jeśli wyk_zaakceptowana (s[i], c) to r := r # s[i];

altep

Koniec

funkcja wyk_zaakceptowana (p: praca; c: kryteria): logiczna =

poczatek {zaakceptowana (p, c)}

do ustalenia;

koniec

Dowód

I na wejściu do petli $s[1..0] \equiv null(plik)$

{prace_recenzowane (null (plik), null (plik), c} = {true}

aksj. o pustych plikach

II w petli, wyk_zaakceptowana = true

prace_recenzowane (r, s[1..i], c) & i ≠ max_prac & wyk_zaakceptowana (s[i+1], c)

= prace_recenzowane (r, s[1..i], c) & i ≠ max_prac & zaakceptowana (s[i+1], c)

⇒ prace_recenzowane (r # s[i+1], s[1..i+1], c)

aksj. o przyjęciu pracy

III w petli, wyk_zaakceptowana = false

podobnie, aksj. o odmowie pracy

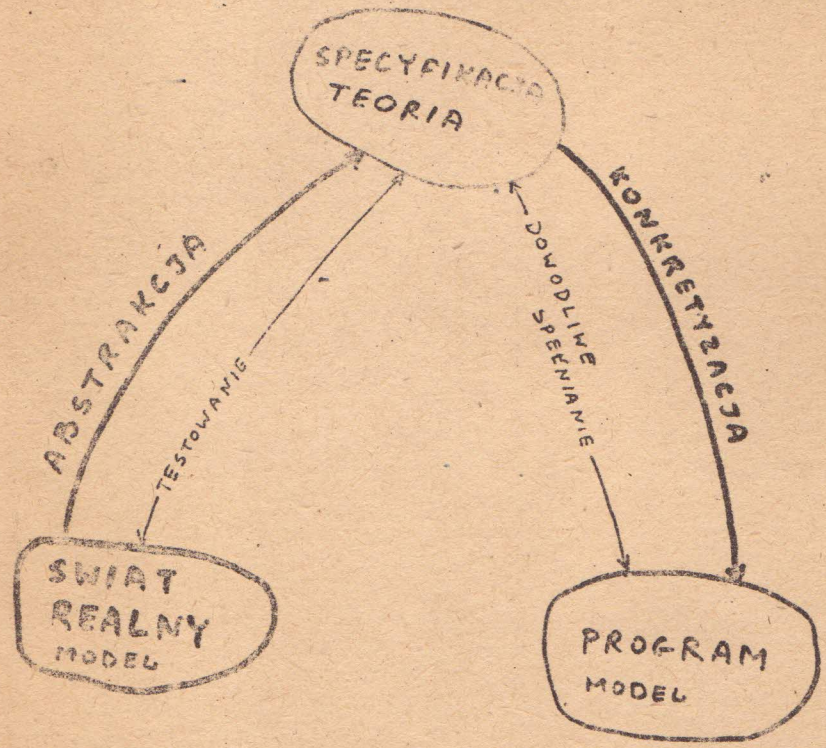
IV petla skończona

prace_recenzowane (r, s[1..i], c) & i = max_prac

= prace_recenzowane (...)

```
funkcja wyk_zaakceptowana (p: praca, c: kryteria) : logiczna =
    pocz { zaakceptowana (p, c) }
        var i : calkowite;
        i := 0;
        petla { brak_blędów (p[1..i], c) & i ∈ [1.. max_prac]}
            jeśli i = max_prac to (wynik := true; skończ);
            i := i+1;
            jeśli wyk_należy (p[i], c) to (wynik := false; skończ);
        altkp
    Koniec

funkcja wyk_należy (w: słowo; c: kryteria) : logiczna =
    pocz { należy (w, c)}
        do wstawienia
    koniec
```



WERYFIKACJA
EMPIRYCZNA

WERYFIKACJA
RACHUNKOWA
(FORMALNA)
DOWÓD

Stanisław Waligórski [1/]

MIĘDZY PRAKTYKĄ A TEORIĄ: O NOWYCH METODACH INFORMATYKI

Przetwarzanie tekstów jest dziedziną, z którą ma teraz do czynienia każdy użytkownik komputera. Dane lub programy, wprowadzone z klawiatury do pamięci komputera trzeba poprawiać i redagować. Każdy, kto używał jakiegokolwiek programu redagującego, wie, że aby dotrzeć do dowolnego miejsca w tekście, trzeba podać charakterystyczny dla tego miejsca zestaw znaków. Ten ciąg znaków /nazwijmy go wzorcem/, jest dopasowywany przez edytor do tekstu tak długo, aż odnajdzie się w tekście fragment identyczny z wzorcem. Takie dopasowywanie wzorców tekstowych potrafi zaprogramować każdy, kto ma nawet elementarną wprawę w programowaniu. Najbardziej się narzuca następujące rozwiązanie: dopasuj wzorzec do początku tekstu; jeśli nie pasuje, przesuń wzorzec o jeden znak i znowu sprawdź, czy pasuje; powtarzaj to dotąd, aż wzorzec będzie pasował, albo aż dojdziesz do końca tekstu.

Ta metoda jest prosta, ale zbyt kosztowna: przeciętna ilość elementarnych operacji, jakie należy wykonać, jest proporcjonalna do kwadratu długości tekstu. Również jest proporcjonalna do tego kwadratu ilość elementarnych operacji, które trzeba wykonać w najgorszym przypadku. Nie jest to zbyt istotne, gdy mamy do czynienia z tekstami krótkimi, ale przy długich tekstach spowolnienie jest wyraźnie odczuwalne. Jeżeli teraz redaguje się komputerowo całe książki, a więc teksty mające setki tysięcy znaków, taki wzrost czasu przeszukiwania tekstów ma już bardzo duże znaczenie, zwłaszcza że często używa się do tego niezbyt szybkich z natury rzeczy komputerów osobistych.

1/ Instytut Informatyki UW

Okazuje się, że takie dopasowywanie wzorców tekstowych może być wykonywane znacznie szybciej, w czasie rzeczywistym, to znaczy tak, że liczba operacji jest proporcjonalna do liczby znaków we wzorcu i nie zależy od długości tekstu. Znalezienie tej szybkiej metody wymagało jednak użycia środków znacznie mniej elementarnych, niż intuicyjnie oczywista metoda, opisana wyżej.

Ten prosty, ale bardzo ważny dla praktyki problem dopasowania wzorców jest dość charakterystycznym przykładem zagadnień, jakie muszą być rozwiązywane przez współczesną informatykę. Samo znalezienie poprawnego algorytmu dla jakiejś klasy zadań i umiejętność zaprogramowania go w jakimś języku już nie wystarczą. Trzeba umieć dobrać algorytm zajmujący możliwie mało czasu i pamięci, umieć porównać jego wydajność z wydajnością innych i móc ocenić, czy jest ona rzeczywiście wystarczająca, aby przy zadanych warunkach wykonywania obliczeń można było ten algorytm wykorzystywać w praktyce. Dlatego właśnie w badaniach nad nowymi metodami informatyki zwraca się tyle uwagi na złożoność algorytmów i dobieranie algorytmów najoprawniejszych, na stosowanie właściwych metod doboru algorytmów i programowania, jak również na sprawność obsługi przetwarzania danych w systemach komputerowych.

Zagadnienia takie nie miały większego znaczenia dla praktyki wtedy, gdy miało się do czynienia z prostym przetwarzaniem danych dla nieskomplikowanych zastosowań, a wszystkie obliczenia były wykonywane w trybie wsadowym. Wtedy to, czy jakiś program pracował nieco dłużej, czy nieco krócej, nie miało właściwie większego znaczenia. Przy takim podejściu do pracy na komputerze do zaprogramowania obliczeń potrzeba przede wszystkim dobrego zrozumienia zadania, znajomości wymagań tej dziedziny zastosowań, dla której zadania ma być wykonane, oraz znajomości odpowiedniego języka programowania. Metody informatyczne optymalnego rozwiązywania i programowania przetwarzania danych miały znaczenie raczej drugorzędne. Sytuacja zmienia się radykalnie przy przejściu na konwersacyjny tryb pracy i przy podejmowaniu zadań o znacznie większym stopniu komplikacji. Praca konwersacyjna nie ma sensu, jeżeli przeciętny czas reakcji komputera jest zbyt długi.

Klucz powodzenia leży jednak nie w zwiększaniu szybkości komputera, ale w sprawności stosowanych metod przetwarzania. Skomplikowanych zadań przetwarzania danych nie da się w ogóle zaprogramować bez użycia właściwych metod, właściwego doboru algorytmów, struktur danych i programów.

Znajomość sposobów doboru i oceny złożoności algorytmów daje również inną korzyść, mającą charakter bardziej ogólny. Pozwala mianowicie na badanie złożoności problemów, czyli minimalnej złożoności algorytmów, które pozwalają takie problemy rozwiązać. Dobre rozeznanie w złożoności problemów pozwala unikać stawiania zadań, których rozwiązanie byłoby zbyt kosztowne. Jeśli jednak to jest potrzebne, pomaga również znajdować problemy, których rozwiązanie jest bardzo trudne - taka umiejętność ma wielkie znaczenie praktyczne przy doborze właściwych sposobów ochrony danych, programów i procesu przetwarzania przed niepowołaną interwencją.

Jeżeli rozwiązanie problemu wymaga przetworzenia dużej liczby danych wejściowych, to opłaci się rozwiązywać go przy pomocy komputera tylko wtedy, gdy ma złożoność wielomianową możliwie niskiego stopnia. Mówimy, że problem ma złożoność wielomianową, gdy liczby operacji elementarnych, jakie trzeba wykonywać dla rozwiązania każdego konkretnego zadania, wyrażają się funkcją wielomianową liczby danych wejściowych tego zadania. Bierze się przy tym oszacowanie graniczne, gdy liczba danych rośnie nieograniczenie, gdyż przy takim szacowaniu interesują nas sytuacje, gdy liczba danych wejściowych jest rzeczywiście duża.

Na przykład wzięcie produktu kartezjańskiego dwu zbiorów, mających po n danych każdy, czyli skojarzenie elementów tych zbiorów w pary sposobem "każdy z każdym", wymaga wykonania n^2 operacji. Jest to więc problem o złożoności wielomianowej stopnia 2.

W przetwarzaniu danych trzeba często wykonywać operację łączenia zbiorów danych, gdy z zestawień pewnych dokumentów cząstkowych trzeba tworzyć zestawienia dokumentów zbiorczych. Weźmy dla przykładu dwa zbiory rekordów, które nazwiemy R i S, i załóżmy, że rekordy zapisane w R mają pola o nazwach A, B, C i D, zaś rekordy

z S mają pola A, B, E i F. Chcemy złączyć ze sobą dane z rekordów R i S, mających takie same zawartości wspólnych pól A i B, tworząc rekordy o polach A, B, C, D, E i F. Utworzony w ten sposób zbiór wszystkich nowych rekordów jest właśnie złączeniem zbiorów R i S.

Chcąc się dowiedzieć, jaka jest złożoność problemu łączenia zbiorów, możemy je łatwo oszacować z góry. Jest jasne, że jednym ze sposobów wykonywania złączenia dwu zbiorów jest kojarzenie ich elementów parami, każdy z każdym, przy czym pary nie mające jednakowych danych w polach A i B się odrzuca, a mające jednakowe dane używa do tworzenia nowego rekordu. Stąd wynika, że złożoność problemu łączenia jest nie większa, niż wielomianowa stopnia 2. Gdyby zbiór R miał m elementów, a S miał n elementów, trzeba by wykonać tyle operacji elementarnych, że ich liczba będzie proporcjonalna do iloczynu m·n, który jest wielomianem 2 stopnia.

Mając już to oszacowanie zauważmy, że jeśli zawartość pól A, B we wszystkich rekordach z R i S jest taka sama, to złączenie R i S nie jest niczym innym, jak produktem kartezjańskim tych dwu zbiorów. Tak więc złożoność złączenia jest taka sama, jak produktu kartezjańskiego.

Praktyk mógłby w tym miejscu zauważyć, że o takim wyniku zadecydowała jedna sytuacja, która w rzeczywistości raczej się nie zdarza. Zwykle w wielkich zbiorach rekordów w każdym polu rekordu z dużym prawdopodobieństwem mogą występować różne wartości. W przeciwnym przypadku, to znaczy gdyby w jakimś polu rekordu występowała zawsze ta sama stała wartość, dane z takiego pola nie miałyby właściwie żadnej wartości informacyjnej. Oznaczałoby to, że zapisy danych są źle zaprojektowane i że przechowuje się w pamięci wartości, które o niczym nie informują, a tylko zajmują wiele miejsca.

Można więc dalej zapytać: jaka jest złożoność łączenia zbiorów, występujących w praktyce? to pytanie jest jednak za mało precyzyjne. Aby mówić na nie uzyskać jakąkolwiek odpowiedź, trzeba dokładnie określić, jakie własności charakterystyczne mają mieć te "zbiory, występujące w praktyce".

Gdyby na przykład założyć, że w łączonych zbiorach wspólne mogą być tylko pola określające klucze główne tych zbiorów, a każdy ze zbiorów jest zawsze, w każdym momencie, posortowany względem klucza głównego, to wtedy złączenie można wyznaczyć przez jednokrotne jednoczesne przeglądnięcie obu zbiorów. Liczba wykonywanych przy tym operacji elementarnych jest proporcjonalna do sumy liczb rekordów w obu tych zbiorach. Jest to funkcja liniowa, czyli wielomian stopnia 1. Ale wynik ten uzyskaliśmy tylko dzięki przyjęciu silnego założenia, ograniczającego zarówno możliwości wykonywania złączeń, które nie mogą być dowolne, jak możliwości utrzymywania różnych kolejności rekordów w zbiorach. Utrzymywanie zbiorów cały czas w postaci posortowanej, wygodne dla łączenia, jest bardzo uciążliwe przy innych czynnościach, takich jak wprowadzanie danych, poprawianie lub aktualizacja, które wtedy stają się bardzo skomplikowane i czas ich wykonywania bardzo się przedłuża.

Trudności nie zostały więc usunięte, ale przerzucone w inne miejsce. Wobec tego zbiory, spełniające takie warunki, także trudno byłoby uznać za "występujące w praktyce".

Chodzi więc o to, aby określić dla rozważanych zbiorów warunki dobrane w taki sposób, aby nie spowodować nierozsądnego i nadmiernego komplikowania wykonywania pewnych czynności za cenę ułatwienia innych. Niestety nie jest jeszcze znana ogólna metoda, która by nam mówiła, jak to należy robić. Z braku takiej metody każdy, kto projektuje system zarządzania bazą danych i oprogramowanie dla niego, musi w ramach tego projektu dobierać własne rozwiązanie, które oczywiście może być dalekie od optymalnego. Wtedy dla takiej konkretnej bazy danych można próbować sformułować warunki, jakie w niej będą spełniać zbiory danych, a następnie próbować znaleźć algorytmy o minimalnej złożoności dla podstawowych operacji na danych, w tym dla operacji łączenia zbiorów. Takie szacowanie złożoności jest jednak w takich przypadkach zadaniem bardzo trudnym. Badania, mające na celu znalezienie dobrych algorytmów łączenia zbiorów dla różnych sytuacji wciąż trwają. Ich wyniki mają znaczenie nie tylko poznawcze: wprowadzenie algorytmów o najmniejszej możliwej złożoności poz-

wolilłoby przyspieszyć i potanić wiele obliczeń dla różnych zastosowań, wykonywanych obecnie najprawdopodobniej w dłuższym czasie i większym kosztem, niż to jest rzeczywiście konieczne. W każdym razie już teraz widać, że poszukiwanie algorytmów o najmniejszej złożoności oraz szacowanie złożoności problemów jest trudne teoretycznie i wymaga użycia skomplikowanego aparatu matematycznego.

We współczesnym, powszechnie stosowanym przetwarzaniu danych operacja łączenia należy do najkosztowniejszych. Inne typowe operacje, jak wyszukiwanie rekordu w zbiorze lub sortowanie są znacznie tańsze. Inaczej mówiąc, większość czynności, typowych dla przetwarzania danych /administracyjnych/ ma złożoność wielomianową stopnia nie przekraczającego 2. Złożoność wielomianową stopnia trzeciego ma problem pełnej analizy składniowej tekstów, których budowę określają dowolnie dobrane gramatyki bezkontekstowe, albo nawet pewne gramatyki kontekstowe. Mogą to być na przykład teksty w jakimś języku programowania, języku dostępu do baz danych, albo w nieco uproszczonym języku naturalnym. Ale takie zagadnienia występują tylko w fazie zapoznawania żądań, kierowanych do bazy danych, albo w fazie tłumaczenia programów. Kiedy dochodzi do właściwej fazy manipulacji danymi, takich obliczeń o dużej złożoności się unika dzięki założeniu, że wszystko zapisuje się przy pomocy rekordów, złożonych z pól, które zawierają dane elementarne, nie podlegające już dalszemu rozbiorowi.

Algorytmy o złożoności wielomianowej stopnia wyższego od 3 spotyka się przede wszystkim w badaniach naukowych oraz tam, gdzie koniecznie trzeba znaleźć szybko rozwiązanie problemu, nie wiedząc nic lub prawie nic o algorytmach, które do tego prowadzą - w szczególności nie znając algorytmów o odpowiednio małej złożoności, które pozwoliłyby to zadanie wykonać. W takich sytuacjach często stosuje się metodę prób i błędów, zwaną w programowaniu metodą nawrotów. W niekorzystnych przypadkach nieudane próby i wycofywanie się z nich mogą bardzo znacznie przedłużyć czas potrzebny na rozwiązanie problemu; czasem nawet poza granice realnych możliwości wykonania obliczeń.

Tych, którzy zajmują się ochroną systemów przetwarzania danych, interesują problemy o możliwie wysokim stopniu złożoności , gdyż włamanie się do chronionego systemu przetwarzania danych powinno być właśnie takim problemem. W hierarchii złożoności bezpośrednio po problemach o złożoności wielomianowej następują problemy o złożoności wykładniczej: liczba operacji, jakie muszą być wykonane przy ich rozwiązywaniu, jest proporcjonalna do funkcji wykładniczej liczby danych wejściowych, o ile tylko ta liczba danych wejściowych jest dostatecznie duża. Problemy o złożoności wykładniczej mogą być rozwiązywane tylko wtedy, gdy niezbędna liczba danych wejściowych jest niewielka. Gdy jest ona duża, problemu nie można rozwiązać żadnymi środkami i sposobami, jakie mogłyby być dostępne i to niezależnie od tego, jak szybkich komputerów chcielibyśmy przy tym użyć.

Istnieje duża klasa problemów, o których nie wiadomo, czy ich złożoność jest wielomianowa, czy wykładnicza. Są wśród nich takie ważne zagadnienia, jak problem spełnialności formuł rachunku zdań. problem pokrycia grafu, problem komiwojażera, programowanie w liczbach całkowitych albo problem plecakowy: mając dane n liczb całkowitych $a_1, \ldots, a_n$ sprawdzić, czy da się spośród nich wybrać takie, aby ich suma była równa danej liczbie całkowitej b.

O wszystkich tych problemach wiadomo, że dadzą się rozwiązać w czasie wielomianowym przy pomocy metod niedeterministycznych. Nie wiadomo jednak, czy ich złożoność jest wielomianowa, to znaczy czy istnieje algorytm /deterministyczny/, któryby wymagał wykonania tylko takiej liczby operacji elementarnych, która byłaby wielomianem liczby danych wejściowych. Ogółem katalog takich problemów liczy już ponad 2 tysiące pozycji i stale rośnie. Udowodnienie dla któregokolwiek problemu, wybranego z tego katalogu, że ma złożoność wielomianową, oznaczałoby znalezienie takiej samej własności dla wszystkich innych problemów w tym katalogu. Podobny skutek miałoby stwierdzenie przeciwne, że jakiś z tych problemów ma złożoność wykładniczą - wtedy oznaczałoby, że złożoność wykładniczą mają również wszystkie inne problemy z tego samego katalogu.

Udzielenie którejś z tych odpowiedzi na pytanie o złożoność tych problemów byłoby rozwiązaniem słynnego problemu, sformułowanego ponad dziesięć lat temu. Na razie, mimo wysiłków wielu badaczy, pozostaje on nie rozwiązany.

Na razie niektóre problemy tej klasy, jako bardzo trudne, wykorzystuje się w zabezpieczeniu danych. Uważa się, że przy obecnym stanie wiedzy najpewniejszym sposobem zabezpieczania wrażliwych danych przy ich transmisji jest szyfrowanie. Aby spełnić wymagania sprawnej organizacji rozproszonego przetwarzania danych i równocześnie zapewnić możliwie niezawodną ochronę, wprowadzono szyfry z publicznym kluczem. Polega to na tym, że dane można zaszyfrować przy pomocy klucza, który jest ogólnie znany i wobec tego każdy może się nim posłużyć. Raz zaszyfrowane dane mogą być jednak odszyfrowane tylko przez właściwego odbiorcę, który dysponuje drugim kluczem, nieznanym nikomu poza nim samym.

Jednym z szyfrów z publicznym kluczem jest szyfr plecakowy, którego idea jest oparta na wykorzystaniu wspomnianego wyżej problemu plecakowego. Złamanie szyfru jest tak złożone, jak rozwiązanie problemu plecakowego. Jeżeli jego złożoność jest wykładnicza, to szyfr jest bezpieczny. Jeśli nie, to znaczy jeśli kiedyś okaże się, że złożoność problemu plecakowego jest wielomianowa, to wszystko będzie zależało od stopnia tego wielomianu, którego teraz w żaden sposób nie potrafilibyśmy ocenić. Jak rozwiązanie wielkiego problemu teorii odbije się na praktyce, przyszłość pokaże. Miejmy nadzieję, że nie będziemy musieli długo na to czekać.

Wróćmy teraz do sformułowanego na początku problemu dopasowania wzorców tekstowych. Pozwoli on nam zwrócić uwagę na jeszcze jedną grupę zagadnień, z jakimi ma do czynienia współczesna informatyka. Najpierw przyjrzyjmy się, jak w ogólnym zarysie wygląda rozwiązanie swego problemu.

Aby móc wykonać dopasowanie wzorca w czasie rzeczywistym, w trakcie jego wprowadzania, trzeba przedtem nadać badanemu tekstowi odpowiednią strukturę. Prosty przykład wyjaśnia zasadniczą ideę metody. Mając tekst: TAKI-DZIKI-PRZYKŁADZIK tworzymy strukturę, której pełny opis zawiera tablica na rys. 1, a uproszczony obraz, narysowany z pominięciem pewnych szczegółów, które by zaciemniły rysunek, jest pokazany na rys. 2.

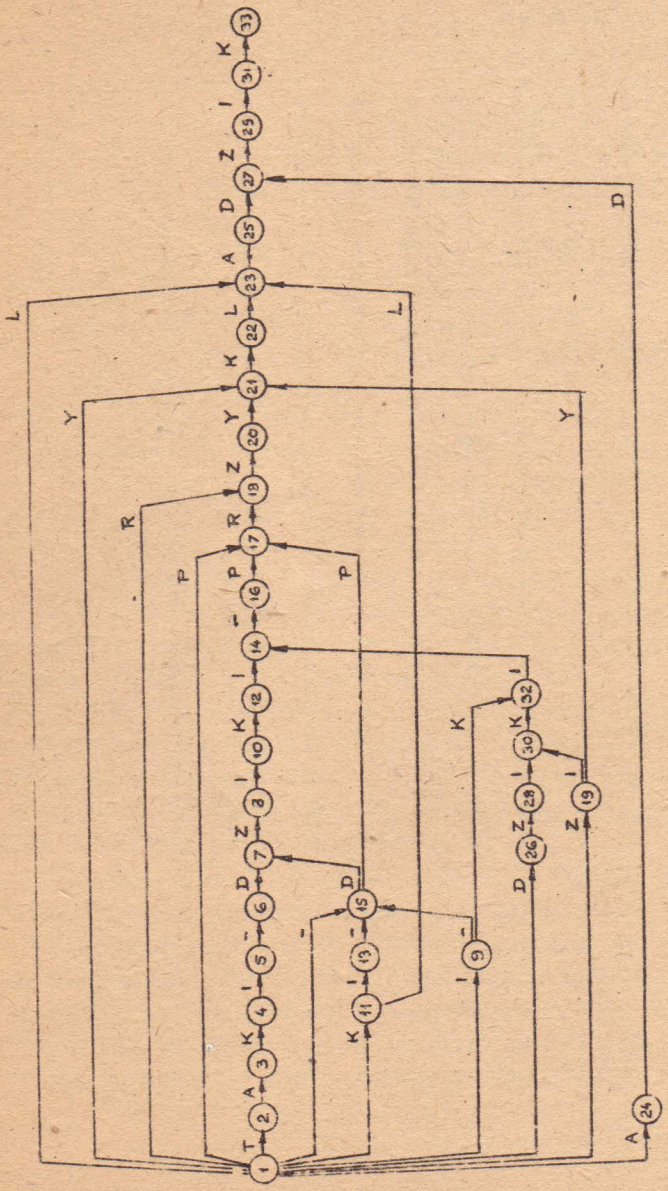
TEKST:

TAKI-DZIKI-PRZYKŁADZIK Rys. 1.

TABLICA:

	T	.A	K	I	-	D	Z	P	R	Y	L					
1	2	24	11	9	15	26	19	17	18	21	23	K				
2	0	3	0	0	0	0	0	0	0	0	0		1			
3	0	0	4	0	0	0	0	0	0	0	0		2			
4	0	0	0	5	0	0	0	0	0	0	0		3			
5	0	0	0	0	0	0	0	0	0	0	0		4			
6	0	0	0	0	0	7	0	0	0	0	0		5			
7	0	0	0	0	0	0	0	0	0	0	0		6			
8	0	0	0	10	0	0	0	0	0	0	0		7			
9	0	0	32	0	15	0	0	0	0	0	0		21	10	8	4
10	0	0	12	0	0	0	0	0	0	0	0		8			
11	0	0	0	13	0	0	0	0	0	0	23	K	22	16	9	3
12	0	0	0	14	0	0	0	0	0	0	0		9			
13	0	0	0	0	15	0	0	0	0	0	0		10	4		
14	0	0	0	0	16	0	0	0	0	0	0		10			
15	0	0	0	0	0	7	0	17	0	0	0		11	5		
16	0	0	0	0	0	0	0	17	0	0	0		11			
17	0	0	0	0	0	0	0	0	18	0	0		12			
18	0	0	0	0	0	0	20	0	0	0	0		13			
19	0	0	0	30	0	0	0	0	0	21	0		20	14	7	
20	0	0	0	0	0	0	0	0	0	21	0		14			
21	0	0	22	0	0	0	0	0	0	0	0		15			
22	0	0	0	0	0	0	0	0	0	0	23		16			
23	0	25	0	0	0	0	0	0	0	0	0		17			
24	0	0	4	0	0	27	0	0	0	0	0		18	2		
25	0	0	0	0	0	27	0	0	0	0	0		18			
26	0	0	0	0	0	0	28	0	0	0	0		19	6		
27	0	0	0	0	0	0	29	0	0	0	0		19			
28	0	0	0	30	0	0	0	0	0	0	0		20	7		
29	0	0	0	31	0	0	0	0	0	0	0		20			
30	0	0	32	0	0	0	0	0	0	0	0		21	8		
31	0	0	33	0	0	0	0	0	0	0	0		21			
32	0	0	0	14	0	0	0	0	0	0	0	K	22	9		
33	0	0	0	0	0	0	0	0	0	0	0	K	22			

RYS. 2.


T
A
K
I
D
Z
Y
L
R
P
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33

Dopasowanie wzorca do tekstu z taką strukturą polega na tym, że w trakcie wczytywania wzorca znak po znaku przechodzi się równocześnie po odpowiednich strzałkach, zaczynając od punktu początkowego, oznaczonego kółkiem z numerem 1. Przejście po strzałce jest możliwe tylko wtedy, gdy na wejściu jest taki sam znak, jaki jest napisany na strzałce. W tablicy na rys.1 pierwsza kolumna odpowiada punktom na rysunku, oznaczonym numerowanymi kółkami. Następne kolumny, opatrzone w nagłówku pojedynczymi literami, opisują strzałki wychodzące z tych punktów, opatrzonych literami, umieszczonymi na górze kolumny : punkt 0 ani strzałki prowadzące do niego nie zostały na rysunku pokazane, gdyż uczyniłoby go to mniej czytelnym. Przyjmujemy, że jeśli na rysunku z jakiegoś punktu nie wychodzi strzałka oznaczona pewną literą, to znaczy, że w rzeczywistości ona prowadzi z tego punktu do punktu 0. Wszystkie strzałki wychodzące z 0 prowadzą z powrotem do 0. Z każdym punktem, z wyjątkiem punktów 0 i 1, jest związany ciąg adresów, umieszczony w prawej skrajnej kolumnie tablicy. Ten szczegół jest także pominięty na rys. 2. Jak zobaczymy, ten ciąg adresów jest potrzebny do lokalizowania miejsc w tekście, identycznych z dopasowywanym wzorcem.

Weźmy na przykład wzorzec DZIK i prześledźmy na rys. 1 i 2 przebieg dopasowania. W miarę wczytywania kolejnych znaków wzorca przechodzimy od punktu 1 przez 26, 28, 30 do punktu 32. Temu numerowi odpowiada ciąg adresów 22 i 9. Rzeczywiście w dziewiątym i dwudziestym drugim miejscu tekstu kończą się fragmenty DZIK, identyczne z wzorcem.

Dla wzorca DZIKI poprzednia trasa przedłuża się od punktu 14, z którym jest związany tylko jeden adres. Rzeczywiście, taki fragment występuje w tekście tylko raz i kończy się w miejscu 10. Wzorzec ADZIK powoduje przejście od 1 przez 24, 27, 29, 31 do 33 i adresem końca pasującego fragmentu jest 22, czyli koniec tekstu. Natomiast wzorzec ADNIK powoduje przejście od 1 do 24, potem do 27, potem do 0, bo nie ma w strukturze tekstu strzałki, która by była oznaczona literą N i wychodząc z pun-

ktu 27, dochodziła do punktu różnego od 0. Jeśli się raz dojdzie do punktu 0, to się w nim pozostaje aż do końca dopasowania, niezależnie od tego, jakie będą dalsze litery wzorca. Dojście do 0 oznacza więc, że do tekstu nie pasuje ani ta część wzorca, która do tego doprowadziła, ani żadne jej przedłużenie. W tym przykładzie do 0 doprowadził ciąg ADN i właściwie na nim można by dopasowanie tego wzorca ADNIK zakończyć - dalsze badanie niczego nie zmieni.

Podobnie jest dla wzorca TADZIK - już część TAD doprowadza do 0, bo w tekście nie ma fragmentu tej postaci. Można więc na niej badanie skończyć, bo żadne jej przedłużenie tez do tekstu pasować nie będzie.

Tworzenie takiej struktury tekstu, umożliwiającej szybkie dopasowywanie wzorców, jest bardzo proste i program, który to wykonuje, jest niewielki. Algorytm ma złożoność liniową, a struktura może być tworzona w czasie rzeczywistym w trakcie wczytywania tekstu znak po znaku. Udowodnienie poprawności algorytmu wymaga jednak znajomości teorii automatów. Korzystając z tej teorii można udowodnić nie tylko to, że algorytm jest poprawny, ale także inne ważne własności otrzymywanych struktur tekstów, jak na przykład to, że dla każdego tekstu tak otrzymywana struktura ma najmniejszą możliwą liczbę punktów, oraz że liczba tych punktów jest mniejsza od dwukrotnej liczby znaków w tekście.

Opis struktury przykładowego tekstu został na rys. 1 przedstawiony w postaci prostokątnej tablicy tylko dlatego, że wtedy łatwiej śledzić szczegóły procesu dopasowywania wzorców. Jednak zapis w formie tablicy prostokątnej nie jest oszczędny, bo jest to zawsze tzw. tablica rzadka: większość miejsc jest zajęta przez zera i liczba miejsc, w których są zapisane informacje rzeczywiście istotne, jest w stosunku do rozmiarów tablicy niewielka. Można jednak zastosować któryś ze znanych algorytmów oszczędnego zapisywania tablic rzadkich w pamięci, tak że opis struktury tekstu zajmie tylko tyle miejsca, ile potrzeba na zapisanie minimum informacji niezbędnych. Będzie on miał postać struktury lis-

towej, czyli odpowiednio zbudowanej sieci, złożonej z elementów pamięci, przechowujących dane dotyczące struktury oraz powiązań między nimi.

Poruszony tu mimochodem problem kompresji zapisów, czyli doprowadzania ich do takiej postaci, aby zajmowały jak najmniej miejsca w pamięci, z możliwością doprowadzenia ich z powrotem do postaci wyjściowej, jeśli to będzie konieczne, ma duże znaczenie praktyczne we wszystkich zastosowaniach informatyki. Próbowano sobie z tym radzić już w epoce maszyn analityczne - liczących i nawet niektóre metody z tych czasów przetrwały do dziś. Ciągle jeszcze są wprowadzane do użytku formularze, na których obok informacji zapisanych w formie czytelnej dla człowieka trzeba wpisać w odpowiednich kratkach te same informacje w postaci zakodowanej jako ciągi cyfr. O tym, jak należy kodować, można się dowiedzieć ze specjalnych katalogów, rozsyłanych wszystkim, którzy będą takie formularze wypełniać. Cały ten czasochłonny i kłopotliwy proceder jest wprowadzany głównie po to, aby dane przetwarzane w komputerach miały możliwie niewielką objętość. W tym przypadku rozwiązanie problemu kompresji polega po prostu na zaangażowaniu odpowiedniej ilości ludzi, którzy przy okazji tworzenia dokumentów dla komputerowego przetwarzania danych dostają dodatkowe zajęcie - przedstawienie tych danych w formie "czytelnej dla komputera" i kosztem dużej straty czasu to wykonują, ręcznie oczywiście.

Nacisk, jaki się kładzie w takich językach, jak Fortran, na dokładne określanie formatu wszystkiego, co się wczytuje lub wpisuje, w szczególności konieczność dokładnego określania formatów danych tekstowych, także ma podobną genezę i też jest reliktem pozostałym po tamtej odległej epoce.

Odejście od stosowania kart perforowanych i od pracy czysto wsadowej do pracy interakcyjnej, wprowadzania danych z terminali do pamięci komputerów bezpośrednio przez samych użytkowników z pominięciem skomplikowanych czynności wstępnych, a w związku z tym ograniczenie stosowania nośników-papierowych we wszelkiej postaci oraz usprawnienie i skracanie czynności związanych z

użytkowaniem komputerów stają się przyczyną tego, że podobne do opisanych prymitywne metody stają się całkowicie niepraktyczne i nieużyteczne. Nie można dalej czynności związanych z optymalizacją obliczeń czy wstępną obróbką danych zwalać na ludzi. Dzięki temu można ich zwolnić z konieczności dbania o różne zbędne szczegóły, jak klasyfikacje, kodowanie i inne zabiegi optymalizacyjne. Cała wstępna obróbka danych i wstępna kontrola po ich wczytaniu nadanie im właściwej postaci i struktury, przystosowanych do dalszego przechowywania i przetwarzania, są wykonywane automatycznie. Komputer wykonuje wszystkie czynności optymalizacyjne, a więc także kompresję danych. Ponieważ jest to proces wykonywany automatycznie, nie musi się już jej ograniczać do mniej lub bardziej oszczędnego kodowania danych jednostkowych. Przeciwnie, najlepsze wyniki uzyskuje się przy kompresji dużych zbiorów danych, traktowanych jako jedna całość.

Z punktu widzenia procesu kompresji dowolny zbiór danych można traktować jako ciąg znaków, a więc teks. Ponieważ proces kompresji jest odwracalny, interesuje nas tylko forma tego tekstu, a nie jego treść i sens danych, które w tej formie są zapisane. W każdej metodzie kompresji bardzo istotną rolę odgrywa wykrycie w tekście wszystkich fragmentów, które się powtarzają. Tutaj właśnie przychodzi z pomocą opisana poprzednio metoda dopasowania wzorców, polegająca na wykorzystaniu specjalnej struktury, nadanej tekstowi. Mianowicie zaletą tej struktury jest nie tylko to, że ułatwia dopasowanie wzorców, ale również wskazuje wszystkie fragmenty tekstu, które się powstarzają.

Aby lepiej pokazać zasady i możliwości kompresji zbiorów danych, weźmy jedną z istniejących metod, mianowicie metodę podstawień tekstowych, zwaną też makrokodowaniem. W rozważanej tu wersji ta metoda polega na tym, że każdy fragment, występujący w tekście po raz drugi lub następny, jest /lub może być/ zastąpiony odsyłaczem do identycznego fragmentu, który wystąpił poprzednio. Odsyłacz musi identyfikować ten fragment poprzedzający, co można na przykład uzyskać, podając adres początku fragmentu w tekście /czyli numer początkowego znaku/ oraz długość fragmentu.

Oczywiście odsyłacz jest też ciągiem znaków o pewnej długości i wobec tego zastępować warto tylko te fragmenty, które są dłuższe, niż odsyłacz, który je zastąpi. W szczególności zastępowane fragmenty muszą zawierać więcej niż jedną literę.

Przykłady, które tu możemy rozważyć, z konieczności muszą być krótkie i wobec tego nie można przy ich pomocy realistycznie pokazać właściwych możliwości kompresji, która działa tym skuteczniej, im tekst jest dłuższy. Można więc tylko pokazać ogólne działanie opisanego schematu zastępowania, pozostawiając resztę wyobraźni czytelnika. Każdy odsyłacz będzie miał postać ujętej w nawiasy pary liczb, przedzielonych przecinkiem. Pierwsza liczba jest adresem w tekście początku wskazywanego fragmentu, druga długością tego fragmentu. Ponieważ tekst przykładowy jest krótki, będziemy zastępować odsyłaczem każdy powtarzający się fragment, mający przynajmniej dwie litery, wbrew podanej wyżej zasadzie zastępowania tylko tych fragmentów, które są dłuższe niż odpowiadający im odsyłacz. Ale tu chodzi przede wszystkim o poglądowe pokazanie sposobu zastępowania fragmentów tekstu, po podstawieniach jest dłuższy, niż był przedtem. W rzeczywistości jednak taka sytuacja nie wystąpi, a ten defekt jest tylko rezultatem przyjętych uproszczeń.

Weźmy ten sam tekst, który posłużył do skonstruowania struktury, pokazanej na rys. 2 i opisanej tablicą z rys. 1. Powtarzające się fragmenty widać od razu, bo tekst jest bardzo krótki, ale rzut oka na tablicę z rys. 1 pozwala upewnić się o poprawności wyboru. Odpowiadają one tym punktom, lub wierszom tablicy, które mają w ostatniej kolumnie co najmniej dwa adresy. Niektóre z nich odpowiadają fragmentom jednoliterowym: 9, 11, 19, 24, 26, którymi są odpowiednio I, K, Z, A, D. Pozostałym odpowiadają fragmenty dłuższe, chociaż te, które mają kilka odpowiedników, mogą mieć wśród nich także fragmenty jednoliterowe. Oto numery punktów razem z ciągami liter, które pozwalają przejść do nich z punktu 1:

13 KI

15 KI- lub I- lub -

28 DZ
30 DZI
32 DZIK

Innych powtarzających się fragmentów dwuliterowych lub dłuższych i nie będących częścią żadnego z tych ciągów w tekście nie ma. Dwa fragmenty są maksymalne i te przede wszystkim nas interesują: KI- oraz DZIK. Odpowiadają im odsyłacz /3,3/ i /6,4/, opisujące te miejsca w tekście, w których one po raz pierwszy występują. Tekst

TAKI-DZIKI-PRZYKŁADZIK

po przedstawieniu odsyłaczy będzie wyglądał tak:

TAKI-DZI/3,3/PRZYKŁA/6,4/.

Powrót do poprzedniej postaci nie nastręcza trudności. Trzeba tylko pamiętać, że adres, podany na pierwszym miejscu każdego odsyłacza, jest numerem znaku w tekście pierwotnym, a nie tym po kompresji.

Przedłużenie takiego tekstu nie zmienia wyniku raz wykonanej kompresji. Trzeba tylko prawidłowo rozmieścić odsyłacze w części dopisywanej:

TAKI-DZI/3,3/PRZYKŁA/6,4/-T/18,5/A

Tekst, który przed kompresją miał 30 znaków, po niej ma 18 znaków i 3 dowiązania. Gdyby te odsyłacze miały w sumie mniej niż 12 znaków, zamiana się opłaci.

Nietrudno zauważyć, że w bardziej skomplikowanych tekstach zmianę fragmentów na odsyłacze można wykonywać na wiele sposobów, i każde przekształcenie może dać w wyniku tekst nieco innej długości. Operację kompresji można wykonywać wielokrotnie: często otrzymany w wyniku kompresji ciąg znaków i odsyłaczy /które też są fragmentami, złożonymi ze znaków/ może mieć nowo wprowadzone powtórzenia, które ponowna kompresja może skrócić. Można stosować nieco inne i bardziej skomplikowane zasady zastępowania, niż pokazane tutaj.

Interesujący jest fakt, że chociaż wykonanie kompresji z takimi skomplikowanymi regułami zastępowania jest w dalszym ciągu operacją stosunkowo łatwą, to bardzo trudno przewidzieć, jaką długość będzie miał ostatecznie wynik końcowy takiej operacji.

Problem, czy dla zadanego tekstu i zadanej liczby K, mniejszej niż długość tekstu i większej od logarytmu dwójkowego tej długości, ostateczny wynik kompresji zmieści się w pamięci o pojemności K znaków, należy do wspomnianej wyżej klasy problemów, o których nie wiadomo, czy mają złożoność wykładniczą, czy wielomianową. Do tej samej klasy, jak pamiętamy, należy między innymi problem plecakowy.

Chociaż tak trudno przewidywać dokładny skutek kompresji, że wykonywanie dokładnych oszacowań jest nieopłacalne, to jednak na pewno zmniejszanie objętości zbiorów danych w taki sposób opłaca się bardzo. Dobrze dobrane szybkie algorytmy kompresji są właśnie tym, co tu jest potrzebne i teoria przychodzi tu ze skuteczną pomocą.

Na zakończenie wróćmy znów do spraw ogólnych. Przytoczony tu prosty przykład metody, wspomagającej dopasowywanie wzorców tekstowych i kompresję danych pokazuje również, jak ważne dla efektywnego wykonania algorytmów jest dobranie właściwych struktur danych. Fakt ten jest dość dobrze znany i operowanie właściwymi strukturami danych oraz umiejętność ich dobierania leżą u podstaw rozwoju wielu dziedzin informatyki, jak na przykład lingwistyka informatyczna, robotyka, projektowanie wspomagane komputerem, grafika komputerowa, czy wreszcie przetwarzanie danych administracyjnych. Tradycyjnie w większości tych dziedzin podstawowym rodzajem struktur są struktury listowe. W skład ich wchodzą elementy pamięci lub rekordy, w których są zapisane dane i które są przy pomocy odpowiednio dobranych dowiązań połączone w sieć. Kształt sieci zależy od jej przeznaczenia, rodzaju operacji, jakie mają być wykonywane na niej oraz na zapisanych w niej danych, jak również od przyjętych dla danej dziedziny ogólnych zasad operowania strukturami danych tego typu. I właśnie ostatnia z wymienionych dzie-

dzin, przetwarzanie danych, daje nam interesujący przykład kontrowersji pomiędzy zwolennikami różnych rodzajów struktur danych: sieciowych i relacyjnych.

Dyskusje na temat zalet i wad obu typów baz danych trwają już około dziesięciolecia, ale nie ma wątpliwości, że są one bardzo użyteczne, gdyż pobudziły rozwój zarówno teorii, jak rozwiązań praktycznych w zakresie systemów przetwarzania danych. Ale to już inna historia.

Literatura:

[1] Banachowski L., Kreczmar A.: Elementy analizy algorytmów. WNT Warszawa 1982.

[2] Ullman J.D.: Principles of database systems. Computer Science Press 1980. /albo tłumaczenie rosyjskie/.

[3] Sysło M.M: O złożoności obliczeniowej kombinatoryki i teorii grafów. Matematyka stosowana t. 14, 1980, s. 53-98.

[4] Willett M.: A tutorial on public key cryptography. Computers and security t. 1, 1982, s. 72-79.

[5] Storer J.A., Szymanski T.G.: Data compression via textual substitution. Journal of the ACM t. 29, 1982, s. 928-951.

[6] Delobel C.: An overview of the relational data theory. Information Processing 1980, North Holland, s. 413-426.

WUR ZSMP 294-84 400 C-5-173