9 1991

informatyka

Język specyfikacji SDL
Kompresja danych
Standaryzacja systemów powielarnych

WYDAWNICTWO SIGMA NOT

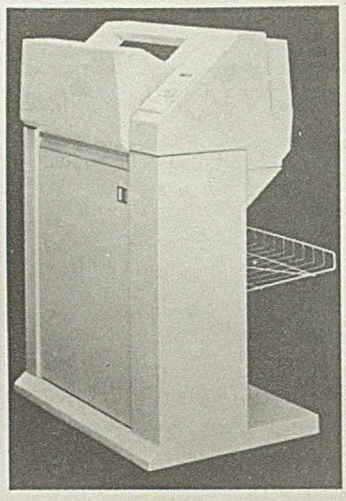
Cena 12 000 zł

SUPRA SP. Z O.O.
ul. Trybunalska 54, 60-325 POZNAŃ
telefon: 674392, telefaks: 236498
teleks: 412966

AUTORYZOWANY DYSTRYBUTOR FIRMY MANNESMANN TALLY OFERUJE DRUKARKI HEAVY DUTY

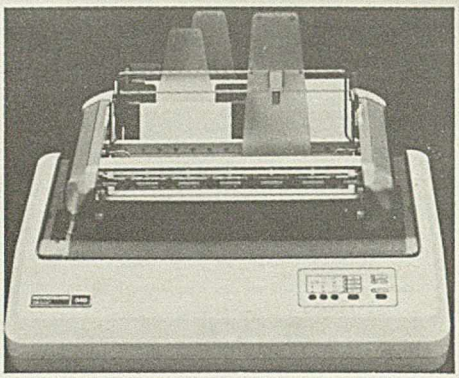
MT 645
drukarka mozaikowo-wierszowa

- doskonała jakość druku
- niezawodność 7000 h MTBF
- duża szybkość 450 wierszy/min.
- cicha praca
- bufor 27 KB
- liczba kopii 1 + 5
- pełna grafika 240 (dpi)
- polskie litery (opcja)
- wewnętrzny program testujący
- dwa interfejsy automatycznie przełączane
- podwójny system traktorów



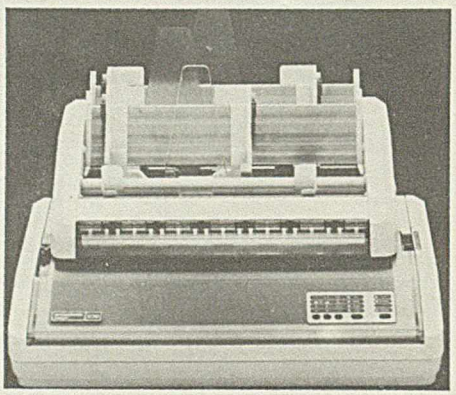
MT 340
drukarka mozaikowa (18 igłowa)

- doskonała jakość druku
- niezawodność
- szybkość 400 zn/sek.
- cicha praca
- polskie litery (opcja)
- druk kolorowy (opcja)



MT 230
drukarka mozaikowa (9, 18, 24 igłowa)

- doskonała jakość druku
- niezawodność
- szybkość 300 zn/sek.
- cicha praca
- polskie litery (opcja)
- druk kolorowy (opcja)



MANNESMANN TALLY

PRAWDOPODOBNIE NAJLEPSZE DRUKARKI NA ŚWIECIE!

0/2/91

informatyka

nr 9 1991
Wrzesień
Rok wydania XXVI

Miesięcznik
ISSN 0542-9951
INDEKS 36124

WYDAWCA:
Wydawnictwo Czasopism i Książek
Technicznych SIGMA NOT
Spółka z o.o.
ul. Biała 4
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja:
01-552 Warszawa,
Pl. Inwalidów 10, p. 104, 105
tel. 39-14-34

Materiałów nie zamówionych
redakcja nie zwraca

W numerze:

W najbliższych numerach:

- **Tomasz Kokowski, Marek Reformat i Maciej Stroiński charakteryzują rozszerzenia lokalnych sieci komputerowych i ich łączenie z sieciami rozległymi.**
- **Tomasz Kalinowski opisuje analizatory wykonania programów służące do uruchamiania, testowania i optymalizacji programów.**
- **Andrzej Maciej Wierzba odpowiada na pytanie: Jak pisać programy satysfakcjonującego użytkownika?**
- **Maria Bronowska podaje charakterystykę pakietu RAMTEK, ułatwiającego programowanie grafiki, zainstalowanego w Centrum Obliczeniowymi Instytutu Podstaw Informatyki PAN.**
- **Marian Kuraś omawia problemy komunikacji z użytkownikami i wewnątrz zespołów projektowych, opisuje znaczenie komunikacji w projektowaniu i podaje przyczyny nieskuteczności komunikacji w zespołach projektowych.**

Warunki prenumeraty

Przyjęcie prenumeraty – wyłącznie na podstawie dokonanej wpłaty na drukach dostarczanych dotychczasowym prenumeratorom przez Wydawnictwo, lub nowym – po uprzednim zgłoszeniu zapotrzebowania (pisemnie lub telefonicznie) w Zakładzie Kolportażu Wydawnictwa.

Blankiet wpłaty – powinien zawierać następujące informacje: dokładna nazwa i adres (z kodem pocztowym) zamawiającego, tytuły zamawianych czasopism, ich liczbę i okres prenumeraty.

Wpłata – zgodnie z podanymi cenami należy dokonać w banku lub w UPT na konto podane na naszym blankiecie, tj:
Państwowy Bank Kredytowy III O/Warszawa nr: 370015-1573-139-11

Prenumeratorzy zbiorowi – osoby prawne obowiązują blankiety „Wpłata-Zamówienie". Cena normalna.

Prenumeratorzy indywidualni – osoby fizyczne obowiązują blankiety typu przekazy dla wpłat na rachunki bankowe. Cena normalna.

Prenumerata ulgowa – zgodnie z podaną ceną ulgową przysługuje wyłącznie osobom fizycznym, będącym członkami SNT, studentom i uczniom szkół zawodowych. Uczniowie szkół ogólnokształcących mogą zamówić w prenumeracie ulgowej tylko miesięcznik „Aura"

Uwaga! w podanym okresie prenumeraty można zamówić tylko po jednym egzemplarzu z każdego tytułu.

Prenumerata ze zleceniem wysyłki za granicę – cena prenumeraty ze zleceniem wysyłki za granicę jest dwukrotnie wyższa od ceny normalnej.

Należy podać dokładny adres odbiorcy za granicą.

Terminy przyjmowania prenumeraty:
- do 10 listopada na I, II, III, IV kwartał następnego roku
- do 28 lutego na II, III, IV kwartał br.
- do 31 maja na III i IV kwartał br.
- do 31 sierpnia na IV kwartał br.

Zmiany w prenumeracie, np. zmiana liczby tytułów, liczby egzemplarzy, rezygnacja z prenumeraty, można zgłaszać tylko w podanych terminach z mocą obowiązującą od następnego kwartału.

Egzemplarze archiwalne (z lat ubiegłych)
Można nabyć za gotówkę w Klubie Prasy Technicznej, Warszawa, ul. Mazowiecka 12 (tel. 26-80-16) lub zamówić pisemnie w Zakładzie Kolportażu, Dział Handlowy, 00-950 Warszawa, skr. poczt. 1004 (tel. 40-37-31), na rachunek lub za zaliczeniem pocztowym.

Informacji o prenumeracie udziela: Zakład Kolportażu Wydawnictwa SIGMA-NOT Spółka z o.o., 00-716 Warszawa, ul. Bartycka 20, skr. 1004. Telefony: 40-00-21 wewn. 293, 295, 299 lub 40-30-86, 40-35-89.

Wstępna cena jednego egzemplarza na 1991 rok: normalna – 12 000 zł, ulgowa – 8400 zł

Wartość prenumeraty (w zł):
Normalna: kwartalna – 36 000, półroczna 72 000, roczna 144 000
Ulgowa: kwartalna – 25 200, półroczna 50 400, roczna 100 800

Uwaga: W przypadku zmiany cen w okresie objętym prenumeratą, prenumeratorzy zobowiązani są do dopłaty różnicy cen.

Skład i druk: Drukarnia SIGMA-NOT Sp. z o.o., z. 398/91 n. 3450

W Szczyrku między 17 a 21 czerwca odbyła się trzecia już wiosenna szkoła organizowana przez Górnośląski Oddział PTI. Oficjalna nazwa szkoły brzmiała: **III Górska Szkoła PTI.** Będzie to już nazwa kolejnych szkół odbywających się w Szczyrku, odróżniająca je od konkurencyjnej imprezy organizowanej w Świnoujściu. Można stwierdzić, że Szkoła szczyrkowska osiągnęła w swojej trzeciej edycji pełną dojrzałość. Pośród jej ponad 150 uczestników byli programiści, analitycy systemów oraz specjaliści do spraw informatyki z firm programistycznych, zakładów przemysłowych, instytutów i uczelni. Podobnie jak poprzednie, Szkoła dzieliła się na trzy nurty, obejmujące po kilka wykładów i komunikatów. Tematy w tym roku obejmowały: bazy danych, organizację prac w informatyce oraz problemy grafiki i wizji komputerowej. Teksty najbardziej interesujących wykładów Szkoły są publikowane w kolejnych, tegorocznych numerach INFORMATYKI, począwszy od nr 8 (artykuł Mariana Niedźwiedzińskiego).

Blok dotyczących baz danych otworzył Zbigniew Benedykt, z BEFAMY w Bielsku-Białej, wprowadzeniem do problematyki baz danych. Z kolei Waldemar Kapuściński, z Huty_Metali Nieżelaznych „Szopienice", omówił własne i kolegów doświadczenia z eksploatacji sieciowych baz danych: TOTAL, DMS 1100, USD. Kolejne cztery wykłady dotyczyły dostępnych na polskim rynku relacyjnych baz danych. Danuta Kosmowska-Mieszalska i Stanisław Wrona, z Wojskowego Instytutu Informatyki w Warszawie, przedstawili charakterystykę systemu INFORMIX z uwzględnieniem architektury programowej jego pakietów. Jacek Stoclak, z ICL POLAND, omówił na przykładzie relacyjnej bazy danych INGRES własności nowoczesnych, profesjonalnych baz danych. Elżbieta Kijowska, z firmy CSBI, zaprezentowała bazę danych PROGRESS, a Wacław Iszkowski, z Instytutu Informatyki Politechniki Warszawskiej, dokonał porównania różnych relacyjnych baz danych. Jan Baranowski, z Polcoloru Piaseczno, zaprezentował środowisko programowe baz danych firmy Novell jako alternatywę dla przedstawionych wcześniej rozwiązań. Zachowanie wśród wykładowców właściwych proporcji między teoretykami, przedstawicielami producentów i użytkownikami konkretnych narzędzi ożywiło bardzo dwa związane z tą tematyką wieczorne konwersatoria. Organizatorzy jak co roku przesadzili z liczbą interesujących wykładów, w wyniku czego zajęcia trwały praktycznie od dziewiątej rano do dziesiątej wieczorem. Na szczęście bar zamykał swoje podwoje około pierwszej nad ranem. Z materiałów bloku dotyczącego baz danych Oddział Górnośląski przygotował książkę, która będzie wkrótce wydana, wspólnie z wydawnictwem Lupus.

Wykłady drugiego bloku były efektem stałego zainteresowania Oddziału Górnośląskiego programowaniem profesjonalnych zasad działania projektantów i wykonawców systemów informatycznych. Problematyka organizacji prac w zespole informatycznym, pozyskiwania faktów, standaryzacji nie tylko języków programowania, ale i sposobów tworzenia aplikacji, bywa lekceważona na rzecz problematyki ściśle technicznej. Jest to istotne niedopatrzenie, któremu Oddział stara sie przeciwdziałać. Główny zrąb tego bloku ustalony został w czasie zeszłorocznej konferencji w Julinie. Za pomoc kolegom ze Szczecina, którzy organizują konferencje w Julinie, należą się serdeczne podziękowania.

Marian Niedźwiedziński, z Uniwersytetu Łódzkiego, spróbował odpowiedzieć na podstawowe pytanie: skąd brać pieniądze na inwestycje informatyczne? Marian Kuraś, z Akademii Ekonomicznej w Krakowie, omówił problemy komunikacji z użytkownikami i wewnątrz zespołu, tworzącego systemy informatyczne. Mariusz Klapper, z Sekcji Metodyki i Dokumentowania Projektów i Programów PTI, przedstawił praktyczne aspekty kompleksowej analizy funkcjonalnej przedsiębiorstw i metodykę ustalania faktów. Barbara Łukasik-Makowska, z Akademii Ekonomicznej we Wrocławiu, podkreśliła wagę standaryzacji systemów powielarnych. Andrzej Maciej Wierzba, z Instytutu Informatyki Uniwersytetu Warszawskiego, zastanawiał się nad tym, jak pisać programy satysfakcjonujące użytkownika. Katarzyna Stąpor i Lech Płowecki zaprezentowali praktyczne doświadczenia i korzyści z eksploatacji pakietu CASE firmy McDonnel Douglas. W uzupełnieniu Tomasz Cierpisz omówił problem i metody kompresji danych.

Trzeci blok wykładów otworzył Teodor Winkler, z CMG KOMAG w Katowicach, omawiając modelowanie graficzne i geometryczne w systemach CAD. Kolejne trzy wykłady dotyczyły trzech różnych, w pełni profesjonalnych narzędzi, opracowanych przez polskich programistów, a rozprowadzanych z powodzeniem na wymagających rynkach zachodnich. Michał Woźnica, z PEI Kopex w Katowicach, przedstawił System Protool. Marek Maniecki, z Inter-Design z Warszawy, zaprezentował system grafiki rastrowej TSL, a Andrzej Maciej Wierzba – w drugim wykładzie na tej samej Szkole – udowodnił, że stosuje się do wniosków pierwszego wykładu omawiając doświadczenia z implementacji edytora graficznego Euro-Chart. Na koniec Bogusław Jackowski, z Polskiego TeX-a, przedstawił problemy związane z modelowaniem odcieni szarości na drukarkach laserowych. Materiały konferencyjne dotyczące problematyki graficznej zostały wydane w pełni profesjonalnej formie, umożliwiającej śledzenie wszelkich finezji prezentowanych produktów i problemów.

Do obniżenia kosztów ponoszonych przez uczestników przyczynili się w znacznym stopniu sponsorzy, w kolejności alfabetycznej firmy: COMPAREX rezprezentowany przez firmę Andra z Warszawy, CSBI z Warszawy, EFEKT z Katowic oraz warszawski UNILOT. Niezależnie od wsparcia finansowego, sponsorzy wzbogacili program szkoły bardzo interesującymi prezentacjami swoich produktów.

Oprócz pracy uczestnicy bawili się między innymi wymyślaniem haseł szkoły. W ogłoszonym konkursie pierwszą nagrodę zdobyło hasło: *Nie ma chłopa bez laptopa*, z którym pozostajemy do czerwca przyszłego roku, oczekując zainteresowania czwartą Szkołą, które można już zgłaszać w biurze Górnośląskiego Oddziału PTI pod katowickim numerem telefonu 538-102.

PIOTR FUGLEWICZ

PIOTR OSTROWSKI
MAREK ŚREDNIAWA
Instytut Telekomunikacji
Politechnika Warszawska

Język specyfikacji SDL (1)

Światowa sieć telekomunikacyjna jest jednym z największych, a z pewnością jednym z najbardziej złożonych systemów czasu rzeczywistego dotychczas zbudowanych przez człowieka. Obecnie wszystkie projektowane i niemal wszystkie produkowane na świecie systemy telekomunikacyjne są sterowanymi programowo systemami cyfrowymi. Zagadnienia specyfikacji, projektowania, implementacji i eksploatacji oprogramowania tych systemów zalicza się do podstawowych elementów techniki telekomunikacyjnej.

Charakterystyka oprogramowania telekomunikacyjnego

Wymagania stawiane systemom telekomunikacyjnym, takie jak: zgodność ze standardami współpracy, niezawodność, praca w czasie rzeczywistym, długowieczność, podatność na zmiany konfiguracji i wprowadzania nowych usług oraz duże wymiary i złożoność strukturalna oprogramowania np. systemów komputacyjnych (przekraczająca dla pojedynczej konfiguracji centrali komputacyjnej milion wierszy programu w języku wysokiego poziomu) sprawiają, że projektowanie jest procesem trudnym, kosztownym i wymagającym stosowania właściwych technik oraz sformalizowanych metod.

Długi czas eksploatacji systemów w sieci, podyktowany względami ekonomicznymi, przy jednoczesnym szybkim tempie przemian technologicznych, prowadzi do dużej różnorodności technicznej elementów sieci, z punktu widzenia techniki realizacji, możliwości funkcjonalnych, postaci przetwarzanych sygnałów i systemów sygnalizacji (systemów współpracy urządzeń tworzących sieć). Sieć podlega ewolucji przy jednoczesnym wymaganiu zachowania spójności funkcjonalnej, rozumianej jako zapewnienie ciągłości świadczonych usług. W tej sytuacji koniecznością stało się uniezależnienie projektu funkcjonalnego systemu od jego faktycznej realizacji, a w konsekwencji wytworzyło potrzebę przyjęcia abstrakcyjnego modelu formalnego języka specyfikacji i opisu oraz wypracowania narzędzi wspomagających projektowanie, realizację i pielęgnację oprogramowania na dużą skalę.

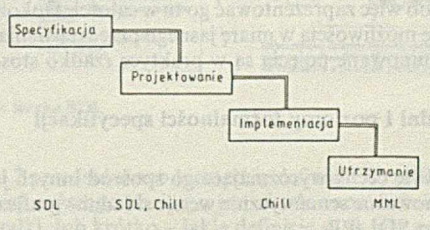
Świadomość tego stanu rzeczy skłoniła w 1968 r. CCITT (*Comité Consultatif International Telegrafique et Telephonique* – Międzynarodowy Doradczy Komitet Telegrafii i Telefonii) do systematycznego zajęcia się zagadnieniami oprogramowania w telekomunikacji. Uznano za konieczne opracowanie i standaryzację języków: specyfiki i opisu – SDL (*Specification and Description Language*) [4, 5], oprogramowania – CHILL (CCITT *High Level Language*) [2, 8] oraz komunikacji operatora z systemem – MML (*Man Machine Language*) [5]. Zamiarem CCITT było objęcie przez proponowane języki pełnego cyklu życia oprogramowania telekomunikacyjnego. Przedmiotem artykułu jest tylko język specyfikacji SDL, niemniej poruszono w nim także niektóre problemy projektowania i implementacji.

Zalecenia CCITT wyznaczają w praktyce międzynarodowe standardy w dziedzinie telekomunikacji. Kolejne wersje zaleceń są publikowane na zakończenie czteroletniej kadencji tej organizacji. Pierwsza wersja zalecenia dotyczącego SDL została wydana w 1976 r., natomiast następne – odpowiednio w latach 1980, 1984 i 1988. Zalecenie z 1988 r. jest podstawą przedstawionej tu wersji języka [4]. Stosunkowo długi okres stosowania i doskonalenia doprowadził do stabilizacji i upowszechnienia stosowania SDL w dużych firmach telekomunikacyjnych oraz administracjach łączności. SDL jest także wykorzystywany przez CCITT do formułowania konkretnych zaleceń systemowych (np. [3]). Równolegle z oficjalną działalnością CCITT wytworzyła się współpraca środowisk zainteresowanych rozwojem i stosowaniem języka. Działania te wspiera wydawnictwo „SDL Newsletter" oraz cykliczne, organizowane co dwa lata konferencje „SDL Forum". Wygłaszane na „SDL Forum" referaty są wydawane w postaci książkowej, np. [6, 25]. Dostępne są także podręczniki, np. [23, 24], krajowe publikacje oraz wydawnictwa wewnętrzne Instytutu Telekomunikacji Politechniki Warszawskiej (IT PW) poświęcone SDL i jego stosowaniu (np. [5, 15, 16, 20]). Problematyka SDL jest omawiana także przy innych okazjach, np. na corocznych konferencjach FORTE poświęconych formalnym technikom opisu. Ogólne wprowadzenie do zagadnień specyfikacji oprogramowania zawiera [7].

Równolegle z formowaniem się SDL tworzyły się w innych środowiskach i pod innym wpływem dwa, w pewnym sensie konkurencyjne, języki: ESTELLE oraz LOTOS. Chociaż języki te mają wiele zalet, to, zdaniem autorów, SDL mimo pewnych wad jest językiem o większych możliwościach i – choćby ze względu na jego obowiązywanie w bardzo licznym środowisku telekomunikacyjnym – mającym większe szanse rozwoju i rozpowszechnienia.

Środowisko i zakres zastosowań SDL

Przewidywany przez CCITT zakres zastosowań języka SDL obejmuje specyfikę wymaganego i opis rzeczywistego zachowania się systemów telekomunikacyjnych. Według pierwotnego zamysłu SDL był przeznaczony do stosowania przez organizacje normalizacyjne, administracje łączności oraz producentów i projektantów jako wspólny język przy specyfikowaniu, projektowaniu, opisie i dokumentowaniu systemów telekomunikacyjnych według klasycznego scenariusza „wodospadowego", przedstawionego na rys. 1. Wyraźnie oddzielono specyfikę oraz opis zachowania się systemu od jego implementacji. Te pierwsze służą do wyrażania właściwości funkcjonalnych systemu, istotnych przy zamawianiu, odbiorze i zapewnianiu współpracy sprzętu i oprogramowania pochodzącego od różnych producentów i reprezentujących różne technologie. Implementacja, czyli sposób realizowania wymagań funkcjonalnych, jest uwarunkowana względami technologicznymi i ekonomicznymi oraz pozostawiana do swobodnego wyboru producentom i dostawcom sprzętu i oprogramowania.



Rys. 1. Model wodospadowy cyklu życiowego oprogramowania

Dla użytkownika jest istotne, by urządzenia niezależne od zastosowania technologii spełniały wymagania dotyczące ich funkcji – by były określone funkcjonalnie. Innymi słowy, pojawia się tu problem nowego

poziomu abstrakcji, na którym należy określić, co urządzenie **ma robić**. Ten poziom nazywa się właśnie mianem specyfikacji funkcjonalnej. To, co rzeczywiście urządzenie **robi** (jakie funkcje rzeczywiście wykonuje), określa się mianem **opisu funkcjonalnego**.

Specyfikacja w SDL nie obejmuje wszystkich technicznych właściwości systemu. W związku z tym, zależnie od potrzeb, jest uzupełniania:
- opisem słownym, wprowadzającym w przedmiot specyfikacji i ułatwiającym jej zrozumienie; opis ten może zawierać inne nieformalne lub formalne elementy, np. wykresy czasowe, schematy interkacji, tabele, pomocnicze diagramy itp.;
- specyfikacją wymagań (parametrów) ilościowych, określających np. jakość obsługi, efektywność, niezawodność, gabaryty, warunki zasilania, wymagania klimatyczne, ergonomiczne, estetyczne itp.

Uwzględniając wymienione potrzeby, na początku prac nad SDL, CCITT określił pożądane cechy języka specyfikacji i opisu. Według tego określenia SDL powinien:
- być łatwy do nauczenia się, stosowania i interpretacji,
- zapewniać jednoznaczność specyfikacji załączonej do zamówień i ofert,
- umożliwiać stosowanie przy różnych metodologiach – paradygmatach specyfikacji i projektowania, bez zakładnia z góry żadnej z nich.

Współczesny stan rozwoju SDL znacznie przekroczył pierwotne zamierzenia CCITT. Język uległ znacznej rozbudowie i jako całość nie może być uważany za łatwy w żadnym sensie. Niemniej stale jest możliwe posługiwanie się skromnym jego podzbiorem, w sposób nieformalny, spełniającym pierwotne postulaty.

Podstawą koncepcji SDL są postulaty formułowane przez wielu autorów prac poświęconych projektowaniu oprogramowania dużych systemów:
- orientacja funkcjonalna.
- przystosowanie do dekompozycji,
- operowanie abstrakcjami i możliwość tworzenia wielu poziomów szczegółowości (równolegle z dekompozycją),
- sugestywność rozumiana jako zdolność do pobudzania inwencji projektanta,
- jednolitość,
- konstruktywność i weryfikalność.

Nie wszystkie z tych postulatów zostały zrealizowane zadowalająco (por. [19]), jednak trudno wskazać inny język, który stanowiłby lepszą podstawę do ich spełnienia w dalszym rozwoju.

Żadna z cech SDL nie ogranicza go do zastosowań telekomunikacyjnych. Może być oczywiście także stosowany w innych dziedzinach elektroniki, zwłaszcza wszędzie tam, gdzie system jest zagnieżdżony w swoim otoczeniu i wykonuje funkcje na rzecz otoczenia, komunikując się z nim za pomocą sygnałów dyskretnych, przenoszących informacje (wiadomości). Uogólniając, SDL może być stosowany do specyfikacji opisu zachowania systemów, dla których właściwym modelem jest zbiór procesów (rozszerzonych automatów skończonych) komunikujących się asynchronicznie.

Prezentacja języka

SDL jest językiem bardzo rozbudowanym i złożonym (definicja oficjalna SDL liczy 204 strony [4] zał. Z.100; liczba słów kluczowych 111) nie sposób więc zaprezentować go tu w całości. Dokonując wyboru kierowano się możliwością w miarę jasnego przedstawienia idei przewodniej. Wyeliminowane pojęcia są w praktyce rzadko stosowane.

Wersje składni i poziomy formalności specyfikacji

SDL ma dwie cechy wyróżniające go spośród innych języków:
- dwie równoważne semantycznie wersje składni – graficzną SDL/GR oraz tekstową SDL/PR,
- możliwość umieszczania w specyfikacji elementów nieformalnych, co pozwala na stopniowe uszczegółowianie i uzupełnianie specyfikacji stosownie do zaawansowania projektu lub przeznaczone dokumentacji.

Z uwagi na swoją czytelność notacja graficzna SDL/GR powinna być preferowana zarówno przy „ręcznym", jak i wspomaganym komputerowo opracowywaniu specyfikacji. Jej główną zaletą jest sugestywne wyeksponowanie schematu struktury specyfikacji i komunikacji za pomocą sygnałów między elementami tej struktury oraz schematu struktury sterowania w procesach i procedurach. W ten sposób notacja graficzna ma ułatwiać twórczy proces projektowania i rozumienia specyfikacji przez człowieka.

Notacja SDL/PR ma obecnie znaczenie pomocnicze i jest utrzymywana ze względu na te jeszcze eksploatowane systemy komputerowe, które nie dysponują możliwościami graficznymi. Ponadto środowisko programistów jest dość silnie przywiązane i przyzwyczajone do notacji tekstowej, typowej dla języków programowania. Korzystając z prezentacji tekstowej tworzy się również znormalizowaną reprezentację wspólną, pozwalającą na przenoszenie specyfikacji z jednego systemu wspomagającego (np. edytora graficznego) do innego systemu (np. translatora formalnej specyfikacji w SDL w program w języku Chill, C lub innym).

Specyfikacją nieformalną w SDL nazywa się specyfikację, w której nie definiuje się obiektów, a definicje akcji opisuje się tekstem nieformalnym w języku naturalnym. Opisy te stanowią odbicie intencji, w jakiej akcja ma być wykonana, np. zadanie z tekstem „zajmij łącze" jest rozumiane jako rozkaz zajęcia łącza. Z kontekstu musi być jasne, „jakiego" i „po co", nie jest natomiast powiedziane „jak". Także nazwy stanów mogą nieść intencję pragmatyczną, związaną z osiągnięciem stanu, np. nazwa „połączenie-ustanowione" oznacza, że osiągnięcie stanu następuje po nawiązaniu połączenia.

Specyfikacje nieformalne stosuje się na etapie formułowania wymagań, norm, zaleceń, dokumentacji itp. Zaleca się taką formę zawsze wtedy, gdy jest ona „praktycznie wystarczająca".

Specyfikacją formalną jest specyfikacja, w której obiekty mają zdefiniowane typy danych i są zadeklarowane, a akcje są zdefiniowane za pomocą wyrażen. Nie stosuje się tekstow nieformalnych. Można i należy stosować komentarze oraz uwagi. Specyfikację formalną można uważać za program, a więc moze być ona wykonalna.

W praktyce najczęściej stosuje się specyfikacje „mieszane", w których występują zarówno teksty nieformalne, jak i elementy formalne. Dopuszcza się formułowanie specyfikacji częściowych, obejmujących tylko jednostki strukturalne poniżej pewnego poziomu (można np. opuszczać poziom systemu). Dopuszcza się także formułowanie specyfikacji, obejmujących tylko pewien przedział poziomów, np. określających tylko strukturę systemu (jego podział na bloki połączeniowe kanałami). W takich przypadkach część definiująca procesy jest pomijana.

Pojęcia podstawowe

Wyobrażenie o działaniu np. systemu, opisanego językiem specyfikacji (opisu), „materializuje się" w modelu abstrakcyjnym tego systemu zbudowanym na podstawie jego definicji określonej specyfikacją (opisem). Odczytanie definicji oznacza więc wyobrażenie sobie działania takiego modelu. Na tej właśnie zasadzie język specyfikacji funkcjonalnej określa zachowanie się systemu. SDL określa zarówno strukturę statyczną systemu, jak i jego zachowanie się. Podstawowym elementem odpowiedzialnym za zachowanie się systemu o danej strukturze są **wystąpienia procesów** (ang. *process instances*), powołane do życia w akcjach tworzenia (kreacji) wystąpień procesów według definicji procesów określających typ tych wystąpień. Zachowanie się całego systemu określa więc dynamiczna sieć wystąpień procesów. Wystąpienia procesów komunikują się między sobą wymieniając sygnały. Gdy mówi się o zachowaniu systemu przez „proces", należy rozumieć pojęcie „wystąpienie procesu".

Model semantyczny

W modelu semantycznym SDL **system jest zbiorem bloków połączonych kanałami między sobą i z otoczeniem systemu.** Zachowanie się bloku jest modelowane przez należące do niego **procesy.** Procesy współdziałają z innymi procesami i z otoczeniem za pomocą sygnałów. Komunikacja za pomocą sygnałów jest asynchroniczna: proces-nadawca po wysłaniu sygnału do procesu-adresata nie jest przez niego wstrzymywany. Przez odpowiednią definicję procesu można określić liczbę jego wystąpień w chwili rozpoczęcia interpretacji systemu oraz maksymalną liczbę jednocześnie istniejących wystąpień danego typu.

W celu identyfikowania procesów wprowadzono predefiniowany typ danych **PId.** Z każdym wystąpieniem procesu związane są cztery zmienne typu PId, określające odpowiednio:

- identyfikator wystąpienia procesu-ojca (**PARENT**),
- tożsamość własną – unikalny identyfikator własny wystąpienia procesu (**SELF**),
- identyfikator wystąpienia ostatnio utworzonego potomka (**OFFSPRING**),
- identyfikator wystąpienia procesu, od którego został ostatnio odebrany sygnał (**SENDER**).

Przypisanie początkowych wartości zmiennym typu PId następuje w chwili tworzenia procesu.

Sygnał może przenosić dane z procesu do procesu. Sygnał nadany przez proces w akcji wyjścia jest przekazywany do procesu-adresata wskazanego we wcześniej opisany sposób. Mechanizm przenoszenia sygnałów między procesami w tym samym bloku jest prawie taki sam, jak między procesami różnych bloków: sygnały wytworzone w określonym procesie i kierowane do innego procesu są w nim odbierane (konsumowane) w kolejności wytworzenia (nadania).

Kanały stanowią drogi przenoszenia sygnałów między blokami. Modelem kanału jest kolejka FIFO o nieograniczonej pojemności. **Drogi sygnałowe** łączą procesy bloku między sobą oraz z kanałami dochodzącymi i wychodzącymi z bloku; przenoszą one sygnały między procesami oraz między procesami i kanałami. Drogi sygnałowe dochodzące do procesu są połączone z **portem procesu**. Modelem drogi sygnałowej jest kolejka FIFO o nieograniczonej pojemności. Całość drogi łączącej parę procesów i przenoszącej sygnały z procesu do procesu stanowi marszrutę sygnałów. Jeśli między każdą parą procesów sieci istnieje tylko jedna marszruta, to określamy ją mianem **kanału wirtualnego**. Gdy sygnał przybywa marszrutą (kanałem wirtualnym) do procesu, do którego jest on kierowany, tj. znajdzie się na końcu kolejki sygnałów zawartej w marszrucie, to zostanie on odebrany i przechowany w porcie procesu aż do odebrania tego sygnału. Sygnały są konsumowane w kolejności odbierania. Modelem portu jest także kolejka FIFO o nieograniczonej pojemności.

Proces w trakcie swego cyklu życiowego znajduje się bądź **w stanie** (STATE), oczekując na sygnał, bądź **w przejściu**, wykonując sekwencję **akcji**. Procesy powoływane do życia w chwili tworzenia sytemu mają wyróżnione miejsce – symbol startu, od którego rozpoczyna się przejście inicjujące proces. W przejściu po symbolu startu (START) może wystąpić dowolny właściwy dla przejścia symbol, w szczególności symbol tworzenia innego procesu.

Proces znajdujący się w stanie, czeka na sygnał. Pobiera on sygnał znajdujący się na początku kolejki portu procesu. Jeśli jest sygnał odpowiadający wyspecyfikowanemu **wejściu** (INPUT), to zostanie on w nim skonsumowany. Dane zawarte w sygnale stają się dostępne dla procesu i rozpoczyna się wykonanie przejścia związanego z danym wejściem. Jeśli jest to sygnał, który ma być **zachowany**, to pozostaje on w porcie i jest pobierany następny sygnał. Zachowanie sygnału jest wyrażane przez wyspecyfikowane czynności zachowania (ang. *save*), zamiast wejścia dla tego sygnału (w danym stanie). Jeśli jest to sygnał nie odpowiadający żadnemu wyspecyfikowanemu w tym stanie wejściu czy symbolowi zachowania, to zostanie on skonsumowany przez **„wejście niejawne"** związane z każdym stanem.

Przejście jest ciągiem akcji. Repertuar akcji obejmuje:

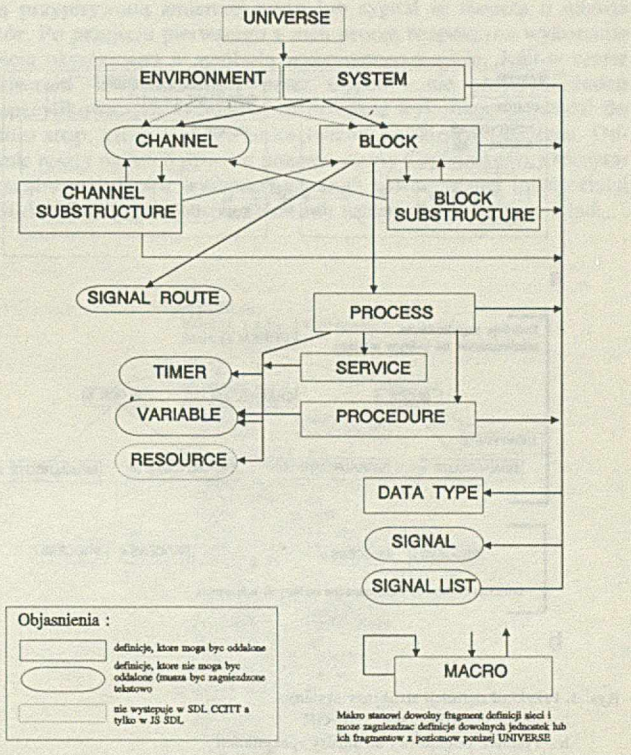
- zadania (TASK), odwołujące się do danych i czasu oraz modyfikujące dane,
- ustawianie (SET) i kasowanie (RESET) stoperów (timerów),
- wyjście (OUTPUT) – wysłanie sygnału,
- tworzenie wystąpienia procesu (CREATE),
- decyzję (DECISION),
- wybór opcji (ALTERNATIVE),
- wywołanie procedury (CALL),
- powrót z procedury,
- makrowywoływanie (MARCO),
- wskazanie następnego stanu lub miejsca kontynuacji oznaczonego etykietą (JOIN),
- unicestwienie wystąpienia procesu (STOP).

W trakcie wykonywania przejścia dane procesu mogą być modyfikowane (w wejściach i zadaniach), proces może wysyłać sygnał, przenoszący dane do innego procesu, a także może tworzyć inne procesy i wywoływać procedury. Przejście może się rozgałęziać w decyzjach uwarunkowanych wartościami obiektów. Istnieje również możliwość odwoływania się do wartości zmiennych należących do innych procesów, pod warunkiem użycia odpowiedniego trybu deklaracji danych. Przejście kończy się osiągnięciem przez proces następnego stanu lub „stopu", powodującego unicestwienie procesu.

W specyfikacji istnieje możliwość odwoływania się do czasu absolutnego reprezentowanego przez **NOW** oraz do czasu względnego, czyli wyrażania różnicy czasu i czasów trwania lub oczekiwania. Wprowadzony jest specjalny typ obiektu – **TIMER** służący do deklarowania stoperów odmierzających w stanie czas oczekiwania na sygnał (ang. *time-out*) lub w przejściu – czas opóźnienia.

Struktura specyfikacji

Specyfikacja (opis) w każdym etapie projektowania jest wykonywana na wielu poziomach szczegółowości (abstrakcji). Poziomy te tworzą strukturę logiczną specyfikacji. Pierwszy poziom (najmniejszej szczegółowości, największej abstrakcji) definiuje strukturę sieci, określając bloki i łączące je kanały z uwzględnieniem podziału bloków między system i otoczenie. Drugi poziom określa budowę bloków jako elementów złożonych z definicji procesów i dróg sygnałowych łączących procesy wewnątrz bloku oraz z kanałami wychodzącymi i przychodzącymi do tego bloku. Trzeci poziom określa substruktury bloków i kanałów, tj. ich uszczegółowienia w postaci substruktur „podsystemów" złożonych z bloków i kanałów. Czwarty poziom zawiera definicje procesów złożone z deklaracji obiektów i definicji akcji (diagramów procesów). Piąty poziom pozwala przedstawić proces jako sieć usług. Na każdym z wymienionych poziomów można umieszczać definicje typów danych i sygnałów oraz makrodefinicje.



Rys. 2. Struktura języka SDL

Na rysunku 2 przedstawiono hierarchię definicji. Strzałki informują, z jakich definicji niższego poziomu jest zbudowana definicja, z której prowadzą strzałki, lub krótko – jakie definicje są w niej zagnieżdżone. Rekurencyjne zagnieżdżanie bloków i substruktur bloków oraz kanałów i ich substruktur pozwala tworzyć dowolną liczbę poziomów szczegółowości konkretnej specyfikacji (opisu).

Definicje poziomów od pierwszego do trzeciego określają strukturę statyczną systemu, a więc jego architekturę, natomiast na pozostałych

poziomach są określane właściwości dynamiczne, a więc zachowanie się składników systemu. Zachowanie się całości systemu jest wynikiem współdziałania składników.

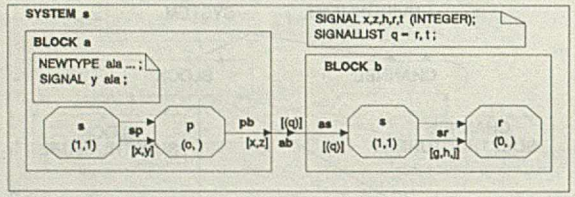
Reguły widoczności

Definicja jednostki może występować w specyfikacji tylko jeden raz, natomiast w definicjach innych jednostek, w deklaracjach obiektów itp. można ją używać wielokrotnie. W celu jednoznacznego wskazania właściwej definicji jednostki stosuje się identyfikator przywołujący żądaną definicję. Obowiązuje przy tym zasada, że można przywoływać tylko definicje widoczne z miejsca przywoływania. Identyfikator składa się z nazwy jednostki oraz kwalifikatora, określającego ścieżkę.

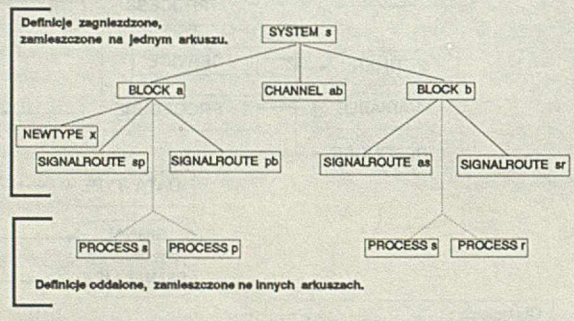
Zdefiniowana nazwa jest widoczna w jednostce, w której jest umieszczona oraz w jednostkach w niej zagnieżdżonych. Przy tworzeniu specyfikacji metodą zstępującą (ang. *top – down*) oznacza to, że widoczne są tylko definicje wcześniej sformułowane lub formułowane w ramach danej jednostki strukturalnej.

Struktura fizyczna dokumentacji

Dokumentacja specyfikacji (opisu) w SDL może być zbudowana na zasadzie zagnieżdżania definicji lub stosowania definicji oddalonych (ang. *referenced*). Metoda zagnieżdżania definicji polega na tekstowym (PR) lub graficznym (GP) zagnieżdżeniu definicji niższego poziomu definicji wyższego poziomu. Na przykład definicja struktury blokowej systemu jest zagnieżdżona w symbolu systemu, w symbolu bloku są zagnieżdżone symbole procesów i dróg sygnałowych (rys. 3), w symbolach procesów – grafy procesów. W notacji PR można tą metodą zagnieżdżać także definicje substruktur. W notacji GR możliwości zagnieżdżania się są znacznie ograniczone wymiarami arkusza rysunku.



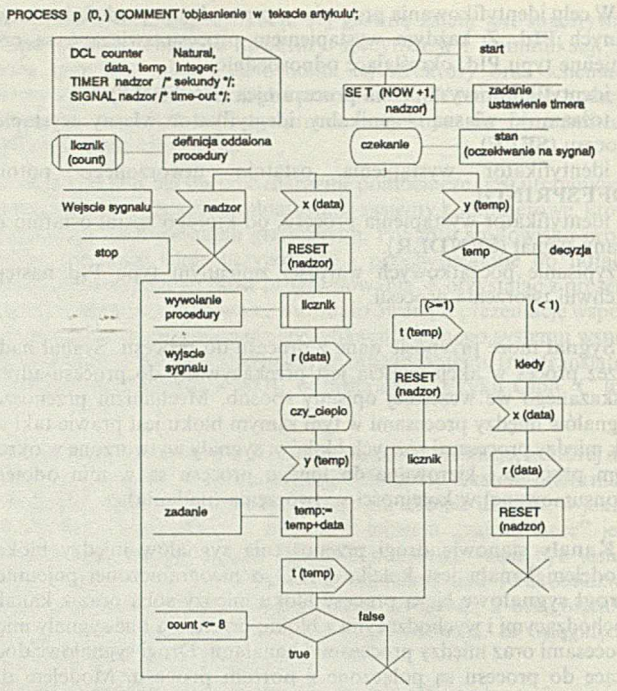
a



b

Rys. 3. Przykład definicji struktury systemu
a – definicja struktury w SDL/GR
b – rysunek poglądowy struktury specyfikacji

Metoda definicji oddalonych polega na definiowaniu na osobnych arkuszach każdego poziomu i każdej jednostki poziomów niższych. Dla przykładu, definicje procesów z rysunku 3 są oddalone. Jedną z tych definicji przedstawiono na rysunku 4. Dopuszcza się mieszanie w miarę potrzeb oraz wygody obu zasad budowy specyfikacji, np. zagnieżdżanie krótkich definicji oraz „oddalanie" długich, „oddalanie" typowych definicji procedur powoływanych z wielu miejsc oraz zagnieżdżanie definicji nietypowych itp. Przedstawiona na rys. 3 oraz 4 specyfikacja jest właśnie przykładem mieszania obu metod.



Rys. 4. Przykład definicji procesu SDL/GR

Dane

Podobnie jak w innych językach specyfikacji programowania, w SDL istnieją mechanizmy definiowania, deklarowania i odwoływania się do danych. Różnica w stosunku do języków programowania polega na przyjęciu matematycznego modelu danych, w których nie występują problemy związane z zakresem, skończonością i dokładnością reprezentacji.

W specyfikacji można odwoływać się do predefiniowanych typów danych lub samemu konstruować aksjomatyczne definicje nowych typów danych. Lista predefiniowanych typów danych obejmuje następujące typy:

- całkowity – **Integer**,
- rzeczywisty – **Real**,
- naturalny – **Natural**,
- logiczny – **Boolean**,
- znakowy – **Character**,
- identyfikacyjny – **PId**,
- czasowy: czas trwania – **Duration**,
- czasowy: moment czasu – **Time**,
- napisowy – **Charstring**.

Deklaracje zmiennych mogą pojawiać się tylko w definicjach procesów i procedur. Oznacza to, że zmienne są lokalne w procesach i procedurach, czyli nie ma zmiennych „globalnych", które mogły by być modyfikowane przez wiele procesów. Natomiast definicje typów danych mogą występować na poziomie systemu, bloku, substruktury bloku oraz w procesach i procedurach. Są one widoczne na zasadzie dziedziczenia nazw w jednostkach zagnieżdżonych. Ewentualny konflikt nazw jest rozstrzygany przez kwalifikację nazwy nazwą jednostki, w której została ona wprowadzona.

Jak wyjaśniono wcześniej, podstawowym środkiem przekazywania wartości danych między procesami jest komunikacja za pomocą sygnałów. Oprócz tego wprowadzono dodatkowy mechanizm udostępniania wartości lokalnej zmiennej procesu innemu procesowi. Występuje on w dwóch wariantach, które można ze sobą łączyć.

Pierwszy wariant polega na deklaracji zmiennej z atrybutem eksportowania w procesie-właścicielu zmiennej i umieszczeniu deklaracji importu w procesie(ach), który ma korzystać z wartości tej zmiennej. Proces-właściciel udostępnia wartość swojej zmiennej za pomocą akcji eksportu. Proces importujący może uzyskać wartość eksportowanej

zmiennej używając operatora importowania, którego dodatkowym argumentem jest tożsamość procesu-właściciela (określona przez identyfikatorów typu PId). Drugi wariant nie wymaga jawnej akcji udostępnienia wartości eksportowej zmiennej przez proces właściciela. Wymagane są natomiast analogiczne jak poprzednio deklaracje w procesie odsłaniającym oraz w procesach zaglądających. Wartość odsłanianej przez proces-właściciela zmiennej jest dostępna w innych procesach za pośrednictwem operatora zaglądania. Różnica między opisanymi wariantami polega na tym, że w pierwszym przypadku proces udostępniający wartość zmiennej kontroluje momenty udostępnienia jej wartości w sposób jawny. Oznacza to, że faktyczna wartość zmiennej może być w danej chwili różna od wartości wyeksportowanej.

SDL pozwala definiować nowe, abstrakcyjne typy danych. Własności tych typów są określane metodą aksjomatyczną, nie wymagającą odwoływania się do wspólnego modelu. Ponieważ zakres stosowania obu metod jest dowolny, wybór poziomu abstrakcji zależy od uznania specyfikującego, a zwłaszcza od potrzeb praktycznej przydatności specyfikacji. Sporządzanie definicji aksjomatycznych oraz korzystanie z nich jest trudne, a automatyczna translacja w program specyfikacji, zawierającej takie definicje, jest nierealizowalna.

Nowe typy danych są definiowane aksjomatycznie. Definicja nowego typu wymaga:
- podania nazwy typu,
- określenia zbioru wartości z uwzględnieniem literałów,
- zdefiniowania operatorów,
- podania zbioru aksjomatów.

Nowe typy danych można definiować w zwięzły sposób, na podstawie już zdefiniowanych typów danych, korzystając z mechanizmu dziedziczenia. Polega ona na wskazaniu, które elementy (operatory, literały, aksjomaty) są dziedziczone oraz na rozszerzeniu tego zbioru o wymagane dodatkowe cechy.

Dla ilustracji sposobu definiowania typów danych przytoczymy fragment definicji predefiniowanego typu logicznego – Boolean:

```
NEWTYPE Boolean
  LITERALS
    True, False;
  OPERATORS
    „NOT": Boolean ♦ Boolean;
    „AND": Boolean, Boolean ♦ Boolean;
    „OR": Boolean, Boolean ♦ Boolean;
  AXIOMS
    NOT True = = False;
    NOT False = = True;
  True AND True = = True;
    (★ dalsze aksjomaty – aż do pełnego zdefiniowania własności typu ★)

ENDNEWTYPE Boolean;
```

Wyrażanie uwarunkowań czasowych

Warunkiem przydatności języka do specyfikcji systemów czasu rzeczywistego jest dostępność środków wyrażania ograniczeń i uwarunkowań czasowych. Podstawowe warunki wyrażane podczas opisywania zachowania systemu są następujące:
- zdarzenie powinno zajść w określonym czasie; w przeciwnym razie należy podjąć inne działania,
- po upływie określonego czasu muszą być wykonane określone czynności.

Warunki te są wyrażane przez ustawienie stopera obejmującego stan i nadzorującego spełnienie ograniczeń czasowych. Można je realizować przez:
- odwołanie drogą wymiany sygnałów do zewnętrznego urządzenia odmierzającego czas,
- użycie przewidzianych w SDL mechanizmów czasu rzeczywistego.

W języku SDL przewidziano następujące mechanizmy:
- Predefiniowane typy danych **TIME** i **DURATION**. Pierwszy wyraża czas bezwzględny – wartość momentu czasu. Drugi reprezentuje czas trwania. Operator bezargumentowy **NOW** zwraca bieżący czas. Dla typu **TIME** zdefiniowano operatory sumy, różnicy i przypisania. Dla typu **DURATION** dodatkowo zdefiniowano operatory mnożenia i dzielenia. Ponadto dla obu typów zdefiniowano operatory reakcyjne,
- Deklaracja stoperów (TIMER), które w połączeniu z akcjami **SET** i **RESET** wprowadzają do specyfikacji mechanizm odmierzania czasów oczekiwania,
- Sygnał ciągły z uwarunkowaniem czasowym,
- Wejście uwarunkowane dozorem odwołującym się do czasu.

Przykład specyfikacji

Bardzo prosty przykład specyfikacji w SDL/GR przedstawiono na rysunkach 3a i 4. Struktura systemu jest zdefiniowana na rysunku 3a. System o nazwie *s* składa się z dwóch bloków o nazwach *a* i *b*, połączonych kanałem o nazwie *ab*. Kanał ten transmituje sygnały z listy *q* z bloku *a* do bloku *b*. Lista ta i zawarte w niej sygnały są zdefiniowane w symbolu tekstowym umieszczonym w definicji systemu. Symbole procesów (ściśle – definicji oddalonych procesów) są połączone defnicjami dróg sygnałowych (linie ciągłe) i tworzenia (linie przerywane). W symbolach procesów podano ich nazwy (kontekst zagnieżdżenia określa pełne ich identyfikatory) i w nawiasach liczby wystąpień. Pierwsza cyfra określa liczbę wystąpień procesów tworzonych w momencie tworzenia systemu, a druga – maksymalną liczbę wystąpień (brak tej cyfry oznacza, że liczba ta jest nieograniczona). Tak więc w przykładzie, prekreowany proces tworzy wystąpienia procesu *p*.

Na rysunku 3b przedstawiono poglądowo strukturę specyfikacji; nie jest to definicja SDL.

Na rysunku 4 przedstawiono przykładową definicję procesu. Zawiera ona deklarację zmiennych (DCL), definicję stopera i sygnału ze stopera do procesu, umieszczone w symbolu tekstowym, oraz definicję oddaloną procedury o nazwie licznik i definicję ciała procesu. Komentarze (symbole dołączone liniami przerywanymi) objaśniają symbole definicji procesu. Wystąpienie procesu rozpoczyna swój żywot w symbolu startu. Po ustawieniu stopera **nadzór** na czas uniwersalny „teraz + 1 sekunda" (NOW + 1), proces osiąga stan „czekanie" i czeka na sygnały *x* z wartością przypisywaną zmiennej **data**, sygnały *y* z wartością przypisywaną zmiennej **temp** lub sygnał ze stopera o nazwie **nadzór**. Po przejściu pierwszego z nich proces rozpoczyna wykonanie przejścia określonego w symbolu wykonanego wejścia. Jeśli w czasie oczekiwania – wyznaczonym przez stoper – nie przyszedł żaden z wyspecyfikowanych sygnałów, to przejście wykonane prowadzi do symbolu **stop**, który oznacza unicestwienie wystąpienia procesu. Odczytanie reszty definicji procesu pozostawiamy Czytelnikowi. Odczytanie zachowania bloku (systemu) wymaga równoczesnej interpretacji wszystkich definicji procesów (bloków) wchodzących w jego skład.

```
SYSTEM S;
 SIGNAL x,z,h,r,t (Integer);
 SIGNALLIST q = r,t;

BLOCK A;
 NEWTYPE ala ... ;
 SIGNAL y ala;
  PROCESS s  REFERENCED ;
  PROCESS p  REFERENCED ;
  SIGNALROUTE pb
   FROM p TO ENV WITH [x,z]
  ENDSIGNALROUTE pb;
  SIGNALROUTE sp
   FROM s TO p WITH [x,y]
  ENDSIGNALROUTE sp;
  CONNECT ab AND pb ;
 ENDBLOCK A;

BLOCK B;
  PROCESS s  REFERENCED ;
  PROCESS r  REFERENCED ;
  SIGNALROUTE as
   FROM ENV TO s WITH [(q)]
  ENDSIGNALROUTE as;
  SIGNALROUTE sr
   FROM s TO r WITH [g,h,j]
  ENDSIGNALROUTE sr;
  CONNECT ab AND as ;
 ENDBLOCK B;

  CHANNEL ab
   FROM a TO b WITH [(q)]
 ENDCHANNEL ab;

ENDSYSTEM S;
```

Rys. 5. Definicja struktury systemu z rysunku 3a w SDL/PR

Specyfikację tego samego systemu sporządzoną w SDL/PR przedstawiono na rysunkach 5 i 6, zachowując ten sam sposób zagnieżdżania, dla ułatwienia porównania obu form syntaktycznych SDL.

```
PROCESS P (0,) COMMENT 'objasnienia w tekscie artykulu';

 DCL counter Natural,
 data, temp Integer;
 TIMER nadzor /* sekundy */;
 SIGNAL nadzor /* time-out */;
 PROCEDURE licznik (count) REFERENCED ;

  START;
   SET (NOW,nadzor);
   NEXTSTATE czekanie;

  STATE  czy_cieplo;
   INPUT  y(temp);
    TASK  temp:=temp+data;
    OUTPUT  t(temp);
#2:  DECISION  count<=8;
    (TRUE) : JOIN #1
    (FALSE) : STOP;
   ENDDECISION;
  ENDSTATE czy_cieplo;

  STATE  czekanie;
   INPUT  nadzor;
    STOP;
   INPUT  x(data);
    RESET (nadzor);
#1:  CALL  licznik;
    OUTPUT  r(data);
    NEXTSTATE czy_cieplo;
   INPUT  y(temp);
    DECISION  temp;
     (>= 1) : OUTPUT  t(temp);
      TASK  RESET (nadzor);
      CALL  licznik;
      JOIN  #2;
     (< 1) : NEXTSTATE kiedy;
    ENDDECISION ;
  ENDSTATE czekanie;

  STATE  kiedy;
   INPUT  x(data);
    OUTPUT  r(data);
    RESET (nadzor);
    STOP;
  ENDSTATE kiedy;

 ENDPROCESS P;
```

Rys. 6 Definicja procesu z rysunku 4 w SDL/PR

LITERATURA

[1] Bursztynowski D., Ostrowski P.: System wspomagający konstruowanie sterowania komunikacją w sieci procesów. Krajowe Sympozjum Telekomunikacji'88. Bydgoszcz, 1988

[2] CCITT: CCITT High Level Language CHILL. Recommendation Z.200. Red Book, Fasc. VI. 11, UIT, Geneva, 1985

[3] CCITT: Recommendations Q.701 – Q.795 (Signalling System No. 7). Red Book. UIT, Geneva, 1985

[4] CCITT: Specification and Description Language (SDL). Recommendation Z.100. Blue Book, Vol. X, Fasc. X.1. UIT, 1989.
User Gudelines (Annex D to Z.100). Blue Book, Vol. X, Fasc. X.2., UIT, 1989
Formal Definition (Annexes to Z.100: F1 – Introduction, F2 – Static Semantics, F3 – Dynamic Semantics). Blue Book, Vol.X, Fasc. X.3,4,5. UIT, 1989

[5] Dąbrowski M.(Ed.): Sterowanie i oprogramowanie w telekomunikacyjnych sieciach zintegrowanych. WKiŁ, Warszawa, 1990

[6] Faergemand O., Margues M.M.(Eds.): SDL'89 The Language at Work. Proc. of the Forth SDL Forum, Lizbon, October 1989.
North-Holland, 1989

[7] Gehani N., McGettrick A.D.(Eds.): Software Specification Techniques. Addison-Wesley, Reading (MA) 1986

[8] Jarociński M., Średniawa M.: Wprowadzenie do języka CHILL. WKiŁ, Warszawa, 1988

[9] Kąkol B., Ostrowski P., Średniawa M.: The SDL Environment SPRITE: Experiences and plans. (Przedrukowano w [6])

[10] Kimbler K., Średniawa M., Wojtal M., Kąkol B., Skorupiński A., Gumuliński P., Dąbrowski M.: Development of an SDL Graphic Environment. (Przedrukowano w [25])

[11] Lubacz J., Ostrowski P., Średniawa M.: On SDL-Based System Design Methodology: Experience with the SPRITE Environment. Proc. of SEETSS'89, Bournemouth, IEE 306 Conference Publication, London, 1989

[12] Niedźwiecka A., Burakowski W.: Język specyfikacji CCITT SDL: Implementacja translacji z zapisu graficznego na zapis tekstowy. Krajowe Sympozjum Telekomunikacji'88. Bydgoszcz, 1988

[13] Ostrowski P.: Determinizm i funkcjonalność systemów specyfikowanych w SDL. Krajowe Sympozjum Telekomunikacji'88. Bydgoszcz, 1988

[14] Ostrowski P.: Integracja procesów w projektowaniu abstrakcyjnym. Przegląd Telekomunikacyjny, Vol. LXIII, nr 5–6, 1990

[15] Ostrowski P.: Język specyfikacji i opisu funkcjonalnego JS SDL. Część 1 – Jądro języka. Część 2 – Definicja JS SDL. Politechnika Warszawska, Instytut Telekomunikacji, Warszawa, 1987

[16] Ostrowski P.(Ed.): Kurs języka specyfikacji i opisu funkcjonalnego JS SDL i jego stosowania (7 zeszytów). Politechnika Warszawska, Instytut Telekomunikacji, Warszawa, 1988

[17] Ostrowski P.: Reprezentatywność i poprawność specyfikacji. Krajowe Sympozjum Telekomunikacji'87. Bydgoszcz, 1987

[18] Ostrowski P., Dziong Z.: New features in SDL – Correctness by contruction. (Przedrukowano w [25])

[19] Ostrowski P., Lubacz J., Średniawa M.: Upgrading SDL. (Przedrukowano w [6])

[20] Ostrowski P., Średniawa M.: Język specyfikacii i opisu funkcjonalnego JS SDL. Część 3 – Stosowanie JS SDL. Politechnika Warszawska, Instytut Telekomunikacji, Warszawa, 1987

[21] Ostrowski P., Średniawa M.: Środowisko wspomagające dla języka JS SDL. Krajowe Sympozjum Telekomunikacji'89. Bydgoszcz, 1989

[22] Piotrowski J., Smaczny M.: Środowisko komputerowej symulacji systemów telekomunikacyjnych opisanych w języku specyfikacji SDL/PR – ESPRIT. Praca dyplomowa, IT PW, 1991

[23] Rockstroem A.: An Introduction to the CCITT SLD. The Swedish Telecommunication Administration, Stockholm, 1985

[24] Saracco R., Smith J.R.W, Reed R.: Telecommunications Systems Engineering Using SDL. North-Holland, Amsterdam, 1989

[25] Saracco R., Tilanus P.A.J (Eds.): SDL'87 State o the Art and Future Trends. Proc. of the Third SDL Forum, The Hague, April 1987. North-Holland, 1987

[26] Średniawa M., Kąkol B., Gumuliński P.: SDL in Performance Evaluation. (Przedrukowano w [25]).

TOMASZ CIERPISZ
PTH „Bister" sp. z o.o.
Katowice

Problem i metody kompresji danych

Używane obecnie oprogramowanie często dotyczy przetwarzania obszernych zbiorów danych (zwłaszcza znakowanych). Bazy danych, źródłowe teksty programów lub pliki przetwarzane przez edytory tekstów wymagają stosowania urządzeń pamięci masowej o dużej pojemności. Jednocześnie zwiększa się ilość informacji transmitowanej na duże odległości przez sieci komputerowe i telekomunikacyjne.

W celu zminimalizowania wymagań dotyczących pojemności stosowanych urządzeń pamięci masowej oraz zmniejszenia kosztów transmisji danych opracowywane i stosowane są metody minimalizujące refundację postaci, w jakiej informacja jest reprezentowana – zwane powszechnie metodami kompresji. Dzięki temu, że po dokonaniu kompresji informacja w postaci upakowanej zajmuje mniej miejsca, na nośniku o określonej pojemności można zapamiętać więcej informacji, a także można je szybciej przesłać przez kanał o określonej przepustowości.

W ślad za zmniejszeniem refundacji informacji, kompresja zwiększa prawdopodobieństwo powstania błędu zapisu lub transmisji danych. Zastosowanie metody kompresji jest często ukryte dla użytkownika, przy czym może ona być realizowana sprzętowo lub oprogramowo. Często zdarza się, że sam użytkownik, i to nieświadomie, stosuje jakąś metodę kompresji danych. Użycie metody kompresji wiąże się z koniecznością określenia czasu potrzebnego na jej realizację. Czas pochłonięty przez algorytm kompresji bywa szczególnie istotny (kosztowny) w przypadku jego programowej realizacji. Reasumując, kompresję stosuje się wówczas, gdy zysk z transmisji lub przechowywania informacji w postaci upakowanej jest większy od strat wynikających z czasu realizacji kompresji.

Historia

Pierwsze metody kompresji stosowano w celu minimalizowania kosztów transmisji danych przy użyciu dalekopisów. Śladem tego faktu jest znak tabulacji, włączony do zestawu znaków ASCI. W następnej kolejności, świadomie tworzono algorytmy kompresji danych niosące informacje graficzne. Wynikało to z faktu, że reprezentacje obrazów charakteryzują się bardzo dużą pojemnością, a równocześnie bardzo dużą redundancją. Obecnie stosuje się wyszukane metody kompresji danych, powstałe w wyniku potrzeby minimalizacji kosztów przesyłania informacji przez komputerowe amatorskie i profesjonalne sieci komputerowe (często wykorzystujące sieć telekomunikacyjną). Popularne programy *pkpak, pkarc, pkzip, lharc* powstały właśnie w celu zminimalizowania kosztów transmisji w sieci FidoNet. Zastosowanie tych programów umożliwa zapisanie większej ilości informacji w pamięciach masowych zainstalowanych w komputerach osobistych. Dzięki temu zyskały sobie dużą popularność. Metody zastosowane w programach kompresji doczekały się również realizacji sprzętowej: niektóre sterowniki dysków stałych (twardych) są wyposażone w układy dokonujące kompresji zapisywanych danych.

Kompresji są poddawane często zbiory zawierające informacje pomocnicze programów narzędziowych (helpy) oraz zbiory słowników, wykorzystywanych przez korektory typograficzne i ortograficzne tekstów (ang. *spellchecker*). Tekst porównawczy stosowanych obecnie programów kompresji danych przedstawiono w artykule [5].

Kompresja, dekompresja, kompakcja i współczynnik kompresji

Kompresją nazywamy funkcję przekształcającą informację wejściową w postać upakowaną, o mniejszej objętości. Stosowanie kompresji ma sens wtedy, gdy istnieje przekształcenie odwrotne, zwane dekompresją, dzięki któremu na podstawie postaci upakowanej można odtworzyć początkową postać informacji.

Pokrewnym zagadnieniem, rozważanym łącznie z kompresją, jest kompakcja. W skrócie można określić, że kompakcja jest kompresją, dla której nie istnieje funkcja odwrotna. Oznacza to, że po dokonaniu kompakcji informacji, nie można jej odtworzyć w pierwotnej formie. Kompakcję stosuje się wtedy, gdy część infomacji przedstawionej w danej postaci uznamy ta zbyteczną, nadmiarową. Przykładem może być źródłowy program komputerowy, w którym ciągi separatorów zastąpiono pojedynczymi separatorami; program taki będzie nadal bezbłędnie kompilował się oraz zawierał wszystkie istotne dla kompilatora informacje, chociaż nie będzie możliwe przywrócenie jego pierwotnej, czytelnej postaci.

Jako miary skuteczności metody kompresji używa się tzw. współczynnika kompresji. Współczynnik kompresji określa ilość nośnika zaoszczędzonego w wyniku zastosowania określonej metody kompresji. Najczęściej wyraża się go w procentach, wyliczając wg następującego wzoru:

$$k = ((we - wy) / we) \star 100\%$$

gdzie:
k – współczynnik kompresji,
we – ilość nośnika zajętego przez pierwotną postać informacji,
wy – ilość nośnika zajmowanego przez postać upakowaną.

Inną stosowaną miarą jest stopień kompresji, wyrażany również w procentach:

$$s = (wy / we) \star 100\%$$

Dodatkowe uwagi i spostrzeżenia

Praktyczne znaczenie ma fakt, że prawdopodobieństwo błędnego zapisu lub błędnej transmisji upakowanej postaci informacji jest większe niż w przypadku zapisu lub transmisji informacji w postaci pierwotnej. Zjawisko to związane jest najczęściej z dwoma przyczynami. Po

Mgr inż. Tomasz CIERPISZ ukończył w 1989 r. studia w Politechnice Śląskiej w Gliwicach, na wydziale Automatyki i Informatyki ze specjalnością Aparatura Elektroniczna. Podczas studiów interesował się problemem komputerowej analizy i syntezy obwodów elektronicznych oraz oprogramowaniem sterowników przemysłowych. Brał udział w pracach nad systemem testera linii telefonicznych oraz systemem rejestracji czasu pracy. Interesuje się systemami pomiarowymi oraz zagadnieniami kompresji danych. Od roku współtworzy program korektora błędów dla tekstów w języku polskim.

pierwsze, dla każdego sposobu kodowania informacji na dysku magnetycznym (lub innym nośniku), można określić kombinację bitów, przy której powstanie błędnego zapisu jest najbardziej prawdopodobne. W praktyce takie ciągi bitów są mało prawdopodobne, jednak w upakowanej postaci informacji prawdopodobieństwo ich wystąpienia zazwyczaj zwiększa się. Po drugie, informacja przesyłana kanałami informacyjnymi bywa kodowana z użyciem kodów korekcyjnych, wprawdzie zwiększających redundancję postaci informacji, ale umożliwiających wykrycie i natychmiastową korekcję pewnej klasy błędów. Istotą kompresji jest natomiast minimalizowanie redundancji, a więc utrata wszelkich nadmiarowych wiadomości zawartych w danej postaci informacji. Kodowanie korekcyjne stanowi więc przeciwieństwo stosowania kompresji.

Nie ma uniwersalnej metody kompresji. Dla takiej metody można znaleźć taką postać informacji, która po dokonaniu kompresji zajmuje więcej nośnika. Każda metoda zakłada określone właściwości przetwarzanej informacji. Jeśli wejściowa postać informacji nie wykazuje takich właściwości, to wybrana metoda kompresji jest nieskuteczna. Dlatego też znaleziono wiele metod kompresji opartych na różnych założeniach dotyczących cech przetwarzanej informacji. Najczęściej w opisie metody podaje się zakres jej zastosowań. Istnieje sposób stwierdzenia a priori przydatności metody kompresji dla konkretnego przypadku. U podstaw każdej metody leży założenie, że informacja składa się z ciągu wiadomości. Podmiotem kompresji staje się nierównomierny rozkład prawdopodobieństw wystąpień wiadomości lub silne powiązania statytyczne między wiadomościami w ramach ciągu. Chciałbym podkreślić, że dla jednej postaci informacji istnieją różne definicje wiadomości.

Upakowana postać informacji, uzyskana przez zastosowanie określonej metody kompresji, zazwyczaj nie poddaje się powtórnej kompresji tą samą metodą. Natomiast zdarza się, że postać upakowana jedną metodą może podlegać kompresji inną metodą. Większość obecnych algorytmów kompresji stosuje kompresję wielostopniową, używając kolejno kilku różnych metod.

Przegląd stosowanych metod kompresji

Przedstawione poniżej metody kompresji odnoszą się do informacji dyskretnej, przedstawionej w postaci ciągu bajtów. Przyjmuje się, że ilość nośnika zajmowanego przez daną informację jest proporcjonalna do liczby bajtów, w której jest zakodowana. Upakowana postać informacji bywa rozpatrywana jako ciąg bajtów lub bitów.

Kodowanie ciągów powtarzających się znaków

Kodowanie ciągów powtarzających się znaków jest chyba najstarszą metodą kompresji. W literaturze anglojęzycznej jest ona określona nazwą *packing* (pakowanie). Z zestawu bajtów wyróżnia się bajt specjalny, którym powinien być bajt najrzadziej występujący w pakowanych ciągach bajtów. Jeśli w wejściowym ciągu bajtów znajdzie się podciąg co najmniej czterech identycznych, występujących bezpośrednio po sobie bajtów, to taki podciąg jest zastępowany trójką bajtów: pierwszy z nich jest bajtem specjalnym, drugi zawiera bajt występujący w podciągu, a trzeci określa długość podciągu pomniejszoną o trzy. Jeśli w wejściowym ciągu bajtów pojawi się bajt specjalny, to w ciągu upakowanym zostanie zastąpiony dwoma bajtami specjalnymi. Jeśli wyróżnionym bajtem będzie 0, to przykładowe ciągi – wejściowy i upakowany – mogą przedstawiać się następująco:

we = (3, 4, 24, 32, 32, 32, 32, 32, 32, 173, 0, 17)
we = (3, 4, 24, 0, 32, 3, 173, 0, 0, 17)

Istnieje wiele różnych odmian tej metody. Jeśli informacja jest kodowana z użyciem zestawu 128 znaków, to jeden bit bajtu jest niewykorzystany i może służyć jako specjalny wyróżnik. Jeśli bit jest ustawiony na wartość 1, to pozostałe bity zawierają długość podciągu pomniejszoną o trzy, natomiast następny bajt informuje, który znak jest powtarzany. Inną modyfikacją metody jest wyróżnienie jednego bajtu, który będzie równoważny określonemu podciągowi. Takim bajtem jest znak tabulacji, równoważny określonej liczbie znaków spacji. Jeśli zostanie znaleziony odpowiedni ciąg spacji, to zostanie on zastąpiony znakiem tabulacji.

Powyższa metoda jest powszechnie stosowana przy kompresji danych tekstowych. Najczęściej jest ona również pierwszym stopniem kompresji w metodach wielostopniowych. Upakowana informacja jest dalej przedstawiana w postaci ciągu bajtów. Metodę kodowania powtarzających się ciągów znaków można z powodzeniem stosować również dla ciągów bajtów reprezentujących informacje graficzne. Pierwotna postać informacji graficznej zawiera zazwyczaj dużą liczbę ciągów powtarzających się znaków.

Kodowanie o zmiennej długości

Kodowanie z zastosowaniem słowa kodowego o zmiennej długości jest klasyczną, najlepiej opisaną w literaturze, metodą kompresji. W klasie tych metod znajdują się: metoda Huffmana, algorytm Shannona-Fano, kodowanie z organiczoną zmiennością długości kodów. Kilka ważniejszych wariantów metody Huffmana przedstawiono w artykule [4]. Podstawą tych metod jest założenie, że informacja przedstawiona jest w postaci niezależnych wiadomości. W metodach tych nie rozpatruje się żadnych powiązań statystycznych między wiadomościami, natomiast są one oparte na rozkładzie częstości (prawdopodobieństw) wystąpień poszczególnych wiadomości w ciągu.

Dla wyjaśnienia przyjmijmy, że wiadomością jest wystąpienie bajtu. Kompresja z zastosowaniem metody kodowania ze zmienną długością kodu polega na zastąpieniu najczęściej powtarzających się bajtów kodami o mniejszej długości, a powtarzających się najrzadziej – kodami o większej długości. Problem sprowadza się zatem do znalezienia zestawu kodów różnej długości, przyporządkowanych bajtom w taki sposób, aby zastąpieniu bajtów odpowiednimi kodami liczba bitów upakowanej informacji była jak najmniejsza. Kody takie są jednoznacznie rozróżnialne, wtedy gdy żaden z nich nie jest przedrostkiem innego. Algorytm znajdujący optymalny zestaw kodów jest nazywany algorytmem Huffmana. Inne algorytmy, np. algorytm Shannona-Fano, są szybsze, lecz nie zapewniają, że znaleziony kod jest optymalny (może być on jedynie zbliżony do optymalnego). Jeszcze szybszą metodą (i na ogół mniej optymalną) jest kodowanie z ograniczoną zmiennością kodu. Kompresja dokonana metodą kodowania zmienną długością kodu daje tym lepsze rezultaty, im bardziej nierównomierny jest rozkład prawdopodobieństw występujących w ciągu wiadomości.

W praktyce, klasyczna metoda Huffmana jest realizowana w dwóch etapach. W pierwszym oblicza się częstości wystąpień znaków w danym ciągu i w ten sposób wyznacza się prawdopodobieństwo ich wystąpień (na podstawie klasycznej definicji prawdopodobieństwa). W drugim etapie jest konstruowany kod Huffmana, w którym przedstawia się pakowany ciąg znaków. Aby umożliwić dekompresję, do zakodowanego ciągu bitów dodaje się informację o sposobie przyporządkowania kodów poszczególnym znakom. Ponieważ informacja dodatkowa ma dla danego zbioru znaków w zasadzie stałą długość, niezależną od długości przetwarzanego ciągu, metoda Huffmana jest bardziej skuteczna dla ciągów długich, natomiast bywa nieskuteczna dla ciągów krótkich. Modyfikacją metody klasycznej jest tzw. dynamiczny algorytm Huffmana, w którym kod jest konstruowany w trakcie przetwarzania, a informacja o sposobie kodowania jest zawarta w wyjściowej postaci informacji. Dzięki temu przetwarzanie wstępne jest zbędne.

Wszelkie modyfikacje metody kodowania ze zmienną długością sprowadzają się do innego algorytmu poszukiwania kodu oraz innej reprezentacji informacji o sposobie kodowania. Jedną z ciekawszych modyfikacji jest metoda kodowania z ograniczoną zmiennością kodów. W pierwszym kroku porządkuje się znaki według malejących prawdopodobieństw wystąpień, a następnie tak uporządkowanym znakom przyporządkowuje się kody według ściśle określonego porządku, np.:

„00", „01", „10", „1100", „1101", „1110", „111100", „111101",...

Informacją dodatkową, dołączoną do zakodowanego ciągu, będzie jedynie kolejność znaków wg malejącego prawdopodobieństwa. Czasem stosuje się rozszerzenie, polegające na rozpatrywaniu kilku ustalonych sposobów kodowania. Do kolejności przyporządkowania dodaje się wówczas typ przyporządkowania. Opcjonalnym uporządkowaniem kodów może być np.:

„1", „10", „001", „0001",. „00001", „000001", ...

Wybór typu kodowania oczywiście jest uzależniony od wielkości uzyskiwanego współczynnika kompresji.

Ważną cechą wszystkich metod kodowania o zmiennej długości jest natychmiastowa dekodowalność kodu wyjściowego. Oznacza to, iż znaczenie danego kodu nie jest związane z wystąpieniem innych kodów w ciągu (wynika to wprost z założenia niezależności kodowanych wiadomości). Fakt ten odróżnia powyższe metody od tzw. kodów historycznych, w których znaczenie kolejnego kodu jest uwarunkowane wystąpieniami poprzednich kodów.

Metody słownikowe

Metody słownikowe, zwane również metodami podstawień tekstowych, tworzą obecnie najszerszą, charakteryzującą się największą różnorodnością, klasę metod kompresji. W zastosowaniu metody te wykazują największą uniwersalność i z tego powodu są podstawą większości uniwersalnych realizacji.

Podstawą każdej metody słownikowej jest założenie, że w rozpatrywanym ciągu znaków wiadomościami są podciągi znaków. Dobrą ilustracją jest metoda podstawień słownikowych, stosowana w zasadzie jedynie do kompresji ciągów znaków reprezentujących teksty. W pierwszym kroku tworzy się słownik wszystkich różnych wyrazów występujących w tekście wejściowym. W zbiorze wyjściowym zapisuje się najpierw słownik, a w następnej kolejności – zakodowane wyrazy. Każdy wyraz jest kodowany jako jego indeks w słowniku.

Przykład

Tekst wejściowy:
kiedy się idzie po miód z balonikiem,
to trzeba się starać,
żeby pszczoły nie wiedziały,
po co się idzie
Utworzony słownik:

słowo	indeks
kiedy	(0)
się	(1)
idzie	(2)
po	(3)
miód	(4)
z	(5)
balonikiem	(6)
to	(7)
trzeba	(8)
starać	(9)
żeby	(10)
pszczoły	(11)
nie	(12)
wiedziały	(13)
co	(14)
,	(15)

Zakodowany numerami indeksów tekst:
(0, 1, 2, 3, 4, 5, 6, 15, 7, 8, 1, 9, 15, 10, 11, 12, 13, 15, 3, 14, 1, 2,)

Efektywność metody podstawień słownikowych można zwiększyć tworząc słownik frekwencyjny. W słowniku frekwencyjnym obok słowa podana jest liczba wystąpień słowa w tekście. Numery indeksów są wówczas kodowane metodą Huffmana. Konstrukcja kodu Huffmana jest możliwa do odtworzenia z danych w słowniku frekwencyjnym. Wynika z tego podstawowe ograniczenie tej metody: w rozpatrywanym ciągu znaków muszą występować separatory oddzielające kolejne podciągi – słowa. Na ich podstawie tworzy się słownik. Jednak w odniesieniu do tekstów metoda ta daje bardzo dobre rezultaty. Jak wynika z badań statystycznych, ok. 20% słownika pokrywa ponad 70% tekstu. Dlatego opłaca się kodować indeksy słownika używając kodów zmiennej długości.

Jedną z najpowszechniej ostatnio stosowanych metod kompresji jest algorytm LZW (skrót od nazwisk twórców Lempel-Ziv-Welch). Metoda LZW wyróżnia się wśród innych bardzo korzystnymi właściwościami. Po pierwsze, jest „jednoprzebiegowa". Wejściowy ciąg nie jest poddawany wstępnemu przetwarzaniu. Dzięki temu algorytm LZW można realizować sprzętowo w trybie on-line. Po drugie, nie wymaga on od wejściowego ciągu znaków żadej cechy szczególnej (znaki wyróżnione, separatory itp.). Wreszcie po trzecie, algorytm ten okazuje się efektywny w odniesieniu do większości występujących w praktyce ciągów znaków (pliki różnego typu: tekstowe, pośrednie, wynikowe).

Algorytm LZW działa korzystając z rozszerzonego zestawu znaków. Dokonując kompresji 8-bitowych znaków ASCII rozszerza się alfabet stosując znaki 9-bitowe (lub dłuższe). Pierwszych 256 znaków będzie zawsze reprezentowało zbiór ASCII, natomiast pozostałe są przeznaczone na tablicę łańcuchów. Cały alfabet będzie reprezentowany przez tablicę, której pierwsze 256 pozycji zostanie zainicjowanych znakami ASCII. Algorytm przetwarzania znaków przedstawia się w zarysie następująco:

- Rozpocznij od pustego łańcucha.
- Przeczytaj kolejny znak i dodaj go do łańcucha.
- Jeśli łańcuch znajduje się w tablicy, to kontynuuj czytanie i dodawanie do łańcucha kolejnych znaków do chwili, gdy łańcuch ten nie zostanie znaleziony w tablicy.
- Dodaj łańcuch do tablicy łańcuchów.
- Zapisz do wyjściowego ciągu kod reprezentujący najdłuższy ciąg, który uprzednio został znaleziony w tablicy.
- Użyj ostatniego przeczytanego z ciągu wejściowego znaku jako początku następnego rozpatrywanego łańcucha
- Czynność powtarzaj aż do przetworzenia całego ciągu wejściowego.

W poniższym przykładzie pokazano działanie tego algorytmu (łącznie z przetwarzaniem odwrotnym) dla ciągu *jam jajko jaj*:

Wejście dekompresji	Tablica kompresji	Wyjście	Tablica
j	–	–	–
a	256 ↦ *j* + *a*	*j*	–
m	257 ↦ *a* + *m*	*a*	256 ↦ *j* + *a*
SP	258 ↦ *m* + *SP*	*m*	257 ↦ *a* + *m*
j	259 ↦ *SP* + *j*	*SP*	258 ↦ *m* + *SP*
a	–	–	–
j	260 ↦ 256 + *j*	256	259 ↦ *SP* + *j*
k	261 ↦ *j* + *k*	*j*	260 ↦ 256 + *j*
o	262 ↦ *k* + *o*	*k*	261 ↦ *j* + *k*
SP	263 ↦ *o* + *SP*	*o*	262 ↦ *k* + *o*
j	–	–	–
a	264 ↦ 259 + *a*	259	263 ↦ *o* + *SP*
j	265 ↦ *a* + *j*	*a*	264 ↦ 259 + *a*
SP	266 ↦ *j* + *SP*	*j*	265 ↦ *a* + *j*

Pierwszy łańcuch *j* znajduje się w zainicjowanej tablicy, czytany jest więc znak *a*, łańcuch *ja* nie zostaje znaleziony w tablicy, więc zostaje w niej zapisany, następnie w ciągu wyjściowym zapisuje się znak *j* (najdłuższy ciąg znaleziony w tablicy), a znak *a* jest początkiem następnego rozpatrywanego łańcucha.

Efekt kompresji pojawia się na początku kodowania słowa *jajko*. Łańcuch *j* znajduje się w tablicy, dołącza się więc znak *a*. Łańcuch *ja* znajduje się już w tablicy (pod numerem 256) więc jest dołączany znak *j*. Łańcucha *jaj* nie ma w tablicy, zatem zostaje w niej zapisany, a do ciągu wyjściowego zapisuje się numer najdłuższego znalezionego w tym cyklu łańcucha, czyli 256.

Łańcuchy są reprezentowane w tablicy jako powiązania łańcuchów ze znakami. Pierwszym elementem jest znany wcześniej łańcuch, drugim – znaleziony za nim w przetwarzanym ciągu znak. Dzięki temu tablica w trakcie przetwarzania zapełnia się wolniej.

Należy pamiętać, że pojedynczy znak w ciągu wyjściowym zajmuje 9 bitów. W przypadku stosowania tablic 12-bitowych znak zajmuje 12 bitów. Fakt ten powoduje zmniejszenie efektywności kompresji. Aby ten efekt zmniejszyć, rozpoczyna się przetwarzanie z użyciem tablicy 9-bitowej, a w chwili, gdy zostanie zapełniona, tablicę tę rozszerza się na 10-bitową i kontynuuje przetwarzania. Innym rozwiązaniem jest zerowanie tablicy łańcuchów w przypadku jej przepełnienia.

Ważną cechą opisanej metody jest to, że istnieje możliwość budowania tablicy dekompresji z użyciem jedynie informacji zawartych w upakowanym ciągu znaków. Dzięki temu zarówno kompresja, jak i dekompresja, są jednoprzebiegowe. Kolejną ważną cechą tej metody jest to, że nie zakłada, jakie łańcuchy znaków będą występowały i powtarzały się w przetwarzanym ciągu. Algorytm budując tablicę „uczy się" i adaptuje stosownie do rodzaju przetwarzanej informacji.

Wykorzystywanie porządku danych

Metoda wykorzystywania porządku danych najczęściej jest stosowana przy kodowaniu alfabetycznie uporządkowanych ciągów słów (słowników); czasem również do kodowania uporządkowanych ciągów kluczy lub indeksów. Podstawą metody jest stwierdzenie, że część kolejnego elementu ciągu (słowa) jest identyczna z częścią poprzedniego elementu ciągu. Każdy następny element jest zapisywany jako cyfra określająca liczbę pierwszych znaków elementu, pokrywających się z pierwszymi znakami poprzedniego elementu oraz ciąg znaków różniących się od znaków poprzednika.
Przykładowy ciąg słów zostanie przedstawiony następująco:

Słowo	Słowo upakowane
aut	0 *aut*
auto	3 *o*
autor	4 *r*
autora	5 *a*
awans	1 *wans*
awanse	5 *e*
awansu	5 *u*
awarła	3 *arła*
awarie	5 *e*
awarii	5 *i*
awiza	2 *iza*
awizo	4 *o*
azalia	1 *zalia*

Różnicowanie

Różnicowanie, podobnie jak metoda wykorzystywania porządku danych, opiera się na wykorzystywaniu określonych cech przetwarzanej informacji. W metodzie różnicowania przetwarzaną informację traktuje się jako ciąg liczb o wartościach z określonego przedziału. Jeśli zbiór różnic między sąsiednimi liczbami w ciągu mieści się w znacznie mniejszym przedziale lub jeśli rozkład prawdopodobieństw wystąpień jest nierównomierny, to metoda różnicowania okazuje się bardzo skuteczna. Zróżnicowany ciąg liczb składa się z pierwszej liczby ciągu wejściowego oraz z kolejnych różnic między następną i poprzednią liczbą w ciągu wejściowym.

Przykład

Dany jest ciąg liczb oznaczających numery wolnych od pracy dni grudnia 1991 r. Numer dnia jest liczbą z przedziału [1...31], a więc może zostać zakodowany z użyciem pięciu bitów.

(7, 8, 14, 15, 21, 22, 25, 26, 28, 29)

Po zróżnicowaniu ciąg liczb przedstawia się następująco:

(7, 1, 6, 1, 6, 1, 3, 1, 2, 1)

Zarówno pierwszy element, jak i wartości pozostałych różnic będą zawsze mieściły się w przedziale [1...8]. Każdy element zróżnicowanego ciągu można zakodować używając trzech bitów. Dzięki temu na każdym przetwarzanym elemencie ciągu oszczędzamy dwa bity. Współczynnik kompresji wynosi w tym przypadku 60%. Co więcej, rozkład częstości wystąpień wartości różnic jest bardzo nierównomierny. Dzięki temu ciąg zróżnicowany będzie można skutecznie zakodować, stosując jedną z metod kodowania ze zmienną długością słowa kodowego.

Powyższy przykład można rozszerzyć do problemu polegającego na zakodowaniu wszystkich dni wolnych od pracy w roku. Numery dni będą znajdowały się w zakresie [1...356], natomiast charakterystyka ciągu zróżnicowanego będzie podobna od uzyskanej w omówionym przykładzie.

Pierwszą odzyskiwaną w procesie dekompresji liczbą będzie pierwszy element przetworzonego ciągu liczb. Każda następna liczba będzie tworzona poprzez zsumowanie ostatniej odzyskanej liczby z następnym elementem zróżnicowanego ciągu.

Metodę różnicowania stosuje się z dużym powodzeniem przy kompresji danych reprezentujących informacje graficzne. Jeśli obraz jest pamiętany jako zbiór linii, z których każda jest ciągiem liczb reprezentujących kolory kolejnych, położonych obok siebie punktów, to istnieje duże prawdopodobieństwo, iż kolory sąsiednich punktów będą „zbliżone", a różnice między liczbami reprezentującymi sąsiednie punkty będą niewielkie (na pewno mniejsze od średniej wartości sąsiednich liczb).

Różnicowanie stosuje się również do kodowania uporządkowanych danych indeksujących (zbiorów indeksowych). Niektóre komputery z wbudowanymi maszynami baz danych stosują przy obsłudze zbiorów indeksowych pewne odmiany różnicowania.

Kompakcja

Metody kompakcji są stosowane znacznie rzadziej niż metody kompresji. Zazwyczaj stosuje się je wówczas, gdy trzeba wydzielić jedynie istotną część informacji. Jak już wspomniano, podstawową cechą kompakcji jest jej nieodwracalność. Na podstawie postaci informacji przetworzonej metodą kompakcji nie jest możliwe jej odtworzenie w postaci pierwotnej.

Kompakcję stosuje się w przypadku przejmowania informacji na inny użytek niż ten, dla którego została ona pierwotnie przygotowana. Jeśli w określonym systemie informatycznym znajdują się np. dane dotyczące wzrostu grupy osób (w centymetrach), natomiast inny system potrzebuje jedynie danych o tym, czy wzrost tych osób jest niski, średni czy wysoki, to informacja przekazywana z pierwszego do drugiego systemu jest poddawana właśnie kompakcji (zmiana określenia wzrostu z centymetrów na jeden z trzech atrybutów). Tego rodzaju przetworzenie jest możliwe tylko w jednym kierunku.

Innym występującym w praktyce przykładem jest kompakcja uporządkowanego ciągu kluczy. Stosuje się ją najczęściej wówczas, gdy przygotowuje się bazy danych służące wyłącznie do odczytywania, które nie podlegają dalszym modyfikacjom i są zapisywane na stałe przez producenta. Wówczas zapisuje się jedynie te fragmenty kluczy, które są niezbędne do ich jednoznaczego rozróżnienia.

* * *

Problem kompresji bardzo trudno poddaje się analizie teoretycznej. Dzieje się tak dlatego, że dotyczy trudnych do deterministycznego ujęcia pojęć: informacja, dane, postać informacji, nośnik. Niektóre metody kompresji są opisywane w literaturze za pomocą formalnego aparatu matematycznego. Taki opis utrudnia w praktyce stwierdzenie, jaką metodę w konkretnym przypadku opłaca się zastosować oraz wprowadza sztuczne ograniczenia możliwości stosowania użytej metody, ułatwiające jedynie jej implementację. Wynika to z faktu milczącego założenia w prawie każdym opisie formalnym, że informacja to treść tekstu, dane to tekst, zbiór słów lub liter, postać informacji to zbiór bajtów, a kryterium ilościowe to liczba bajtów. W poniższym artykule świadomie starano się używać terminów bardziej ogólnych.

Problem kompresji pojawia się wtedy, gdy istnieje konieczność pamiętania lub transmisji informacji (zawierającej określoną treść). Informacja jest przedstawiana w zdefiniowanej postaci, w jakiej jest zapamiętywana na nośniku (lub przesyłana). Zazwyczaj można określić kryterium ilościowe, określające zajętość nośnika. Określa ono ilość nośnika, na którym dana informacja jest pamiętana. Problem kompresji polega na znalezieniu optymalnej funkcji przekształcającej wejściową postać informacji w upakowaną, tzn. taką, aby ilość nośnika zajętego przez tę postać była jak najmniejsza. Formalnie jest to więc problem z pogranicza rachunku wariacyjnego (szukanie optymalnej funkcji przy określonym kryterium).

W literaturze dotyczącej kompresji stosuje się pojęcie entropii, definicję źródła Markowa pierwszego rzędu oraz bajt jako jednostkę informacji (wiadomość). Chciałbym podkreślić, że są to zawsze rozważania szczegółowe, w których stosownie do bieżących potrzeb formalizacji problemu definiuje się kryterium oraz pojęcia informacji, danej, postaci, wiadomości lub nośnika. W ogólnym podejściu kompresją staje

dokończenie na s. 31

W artykule wykorzystano materiały opracowane na I Szkołę Wiosenną PTI w Świnoujściu i na konferencję INFOGRYF '88

MARIUSZ KLAPPER
Polskie Towarzystwo Informatyczne
Sekcja Metodyki i Dokumentowania
Projektów i Programów
Kraków

Praktyczne aspekty kompleksowej analizy funkcjonalnej przedsiębiorstw i metodyka ustalania faktów

Jednym z pierwszych problemów pojawiających się w chwili rozpoczęcia prac nad kompleksową informatyzacją przedsiębiostwa jest konieczność dobrego poznania jego specyfiki, zasad funkcjonowania, przepływów informacji, powiązań strukturalnych wewnętrznych i zewnętrznych oraz innych cech, mających istotne znaczenie dla systemu informatycznego. Prace umożliwiające rozeznanie tych spraw będą w artykule określane mianem kompleksowej analizy funkcjonalnej przedsiębiorstwa. Sposób organizowania i wykonania tych prac będzie nazywany metodyką kompleksowej analizy funkcjonalnej. Sposób przygotowywania, wykonania i udokumentowania rozmów i wywiadów dostarczających wyjściowych informacji dla analizy funkcjonalnej będzie nazywany metodyką ustalania faktów.

Metoda kompleksowej analizy funkcjonalnej przedsiębiorstw, wzorowana na standardzie BISAD firmy Honeywell, umożliwia przeanalizowanie całości powiązań, uwarunkowań i zależności systemu informacyjnego przedsiębiorstwa. Pozwala określić cele, specyfikę, racjonalność, znaczenie i wzajemne powiązania poszczególnych obszarów funkcjonowania przedsiębiorstwa pod kątem ich informatyzacji. Przeprowadzenie analizy funkcjonalnej znacznie przyspiesza, ułatwia i zmniejsza koszt późniejszych prac aplikacyjnych przy tworzeniu kompleksowego systemu informatycznego przedsiębiorstw.

Celem kompleksowej analizy funkcjonalnej jest opracowanie założeń komputerowego wspomagania działalności przedsiębiorstwa. Analiza funkcjonalna polega na zewidencjonowaniu i przeanalizowaniu przepływu najważniejszych informacji związanych z funkcjonowaniem przedsiębiorstwa oraz na opracowaniu prototypu systemu informacyjnego przedsiębiorstwa.

Etapy analizy funkcjonalnej przedsiębiorstwa

Kompleksowa analiza funkcjonalna przedsiębiorstwa jest realizowana w pięciu etapach:

- wstępne uzgodnienia w przedsiębiorstwie,
- analiza podstawowa przedsiębiorstwa,
- analiza funkcjonalna przedsiębiorstwa,
- prototyp systemu informacyjnego przedsiębiorstwa,
- projekt informatyzacji przedsiębiorstwem.

Materiałem wyjściowym są informacje uzyskane podczas rozmów i wywiadów przeprowadzonych przez zespół wykonujący analizę funkcjonalną. Treść rozmów i wywiadów zostaje utrwalona w postaci notatek autoryzowanych przez rozmówców z przedsiębiorstwa. Ze względów proceduralnych i praktycznych notatki te są sporządzane w jednym egzemplarzu i przekazywane do przedsiębiorstwa razem z dokumentacją całości prac. Staranne przygotowanie wywiadów, umiejętne ich przeprowadzenie i dobre udokumentowanie mają kluczowe znaczenie dla poprawności i jakości analizy funkcjonalnej.

Wstępne uzgodnienia w przedsiębiorstwie

Celem uzgodnień jest ustalenie:
- zakresu prac,
- terminów realizacji prac,
- kosztu prac,
- zakresu uprawnień zespołu prowadzącego prace,
- form współpracy przedsiębiorstwa,
- formy i zawartości wyników końcowych,
- oczekiwań i potrzeb przedsiębiorstwa,
- oczekiwań i potrzeb wykonawców pracy.

Efektem końcowym wstępnych uzgodnień jest zazwyczaj kontrakt lub umowa o wykonanie analizy funkcjonalnej.

Analiza podstawowa przedsiębiorstwa

Celem analizy jest uzyskanie podstawowych informacji o przedsiębiorstwie obejmujących:
- krótką historię przedsiębiorstwa,
- charakterystykę profilu produkcji i działalności przedsiębiorstwa,
- informacje o związkach przedsiębiorstwa z otoczeniem (zakłady pokrewne, udział w zrzeszeniach, jednostki zwierzchnie, kooperanci, branża, odbiorcy, dostawcy itp.),
- schemat fizycznego przepływu dóbr w przedsiębiorstwie,
- schemat organizacyjny przedsiębiorstwa, z uwzględnieniem stanu zatrudnienia w poszczególnych komórkach organizacyjnych oraz nazwisk i sposobów komunikowania się z osobami pracującymi w tych komórkach na kierowniczych stanowiskach,
- krótki opis zadań poszczególnych pionów i komórek organizacyjnych przedsiębiorstwa,
- zestawienie formalnych i nieformalnych powiązań wewnątrz struktury organizacyjnej przedsiębiorstwa,
- charakterystykę zatrudnienia w przedsiębiorstwie,
- najważniejsze problemy i trudności występujące w pracy przedsiębiorstwa, z ich krótką charakterystyką,
- inne informacje uznane za ważne dla wiedzy o przedsiębiorstwie.

Mgr inż. Mariusz KLAPPER ukończył w 1971 r. Akademię Górniczo-Hutniczą w Krakowie o specjalności Automatyka Przemysłowa i Elektronika, sekcja Maszyn Cyfrowych i Programowania. Jedenaście lat pracował w przemyśle, pięć lat na wyższych uczelniach, od czterech lat zajmuje się działalnością usługową. Współtwórca kilkunastu systemów informatycznych funkcjonujących w różnych działach gospodarki. Autor i współautor publikacji i referatów specjalistycznych. W 1986 roku uzyskał I stopień specjalizacji zawodowej w dziedzinie inżynierii oprogramowania. Od 1983 roku czynnie działa w Polskim Towarzystwie Informatycznym. Współtwórca i Przewodniczący Sekcji Metodyki i Dokumentowania Projektów i Programów PTI. Aktualnie współwłaściciel prywatnej firmy usługowej.

Analiza funkcjonalna przedsiębiorstwa

Celem analizy funkcjonalnej jest wydzielenie głównych funkcji przedsiębiorstwa i związanych z nimi czynnościami oraz zewidencjonowanie przepływu podstawowych informacji między tymi funkcjami. Kryteria podziału są w zasadzie umowne, choć przy ich ustalaniu bierze się pod uwagę względy racjonalności i reprezentatywności dla opisywanego przedsiębiorstwa. Uzyskany schemat funkcjonalny przedsiębiorstwa jest dość ogólny, co objawia się np. słabą reprezentacją w schemacie szczegółowej specyfiki przedsiębiorstwa (dla analizy funkcjonalnej nie jest ona istotna). Podział funkcjonalny na ogół nie pokrywa się ze strukturą organizacyjną przedsiębiorstwa, gdyż wiele komórek organizacyjnych uczestniczy w realizacji kilku funkcji lub czynności równocześnie. Schemat funkcjonalny stanowi punkt wyjścia do opracowania prototypu systemu informacyjnego przedsiębiorstwa.

Prototyp systemu informacyjnego przedsiębiorstwa

Celem opracowania prototypu systemu informacyjnego przedsiębiorstwa jest ukazanie racjonalnego powiązania najważniejszych informacji i wykonywanych przetwarzań w jego ramach. Jako punkt wyjścia przyjmuje się ustalony podczas analizy funkcjonalnej podział przedsiębiorstwa na funkcje. Dla każdej funkcji są definiowane czynności prototypu, wyodrębnione pod kątem przetwarzania najważniejszych czynności informacji, oraz są określane informacje przetwarzane przez tę czynność. Podstawowymi kryteriami podziału na czynności i informacje są celowość, racjonalność i kompletność. Dlatego prototyp systemu informacyjnego przedsiębiorstwa nie musi ilustrować stanu istniejącego, natomiast powinien odzwierciedlać wszystkie racjonalne potrzeby, które nie są uwzględnione w jego bieżącej działalności. Schemat prototypu systemu informacyjnego jest odniesiony do struktury organizacyjnej przedsiębiorstwa, gdyż dla definiowanych czynności prototypu są określane odpowiednie komórki organizacyjne przedsiębiorstwa, w których czynności te są wykonywane. Schemat prototypu systemu informacyjnego stanowi punkt wyjścia do określania obszarów informatyzacji przedsiębiorstwa oraz do ustalenia powiązań i zależności między tymi obszarami. Jest on niezbędny do opracowania założeń kompleksowej informatyzacji przedsiębiorstwa, gdyż zawiera pełny obraz przetwarzań i powiązań informacji występujących w przedsiębiorstwie.

Projekt informatyzacji przedsiębiorstwa

Projekt ten stanowi podsumowanie wcześniej wykonanych prac. Zawiera szczegółową specyfikację i charakterystykę systemów informatycznych proponowanych do wdrożenia w przedsiębiorstwie, a określonych na podstawie analizy funkcjonalnej i prototypu systemu informacyjnego. Każdy system jest opisany w jednolitym układzie zawierającym symbol, opis systemu, obszar zastosowań, cel i zadania, korzyści, ograniczenia i powiązania oraz nakłady potrzebne do jego wdrożenia (praca, czas, sprzęt itp.). Na podstawie zestawienia proponowanych systemów informatycznych zostaje opracowany harmonogram informatyzacji przedsiębiorstwa, określający kolejność działań wdrożeniowych oraz uwzględniający tematy, nakłady, uwarunkowania, obszary priorytetowe, a także możliwości etapowego i równoległego prowadzenia prac. Materiał zawarty w tym opracowaniu stanowi podstawę do podejmowania szczegółowych decyzji dotyczących tematyki, zakresu, tempa i nakładów na informatyzację przedsiębiorstwa.

Dokumentacja końcowa prac kompleksowej analizy funkcjonalnej obejmuje zazwyczaj od kilkuset do kilku tysięcy stron i składa się z pięciu części:

- **Analiza podstawowa** – zawiera podstawowe informacje o przedsiębiorstwie.
- **Analiza funkcjonalna** – zawiera podział przedsiębiorstwa na funkcje, a także specyfikacje najważniejszych informacji przekazywanych między tymi funkcjami. Opisuje stan istniejący w przedsiębiorstwie i dotyczy jego funkcjonowania, a nie struktury organizacyjnej.
- **Prototyp systemu informacyjnego** – zawiera projekt prototypu systemu informacyjnego przedsiębiorstwa, tzn. wydzielenie w ramach przyjętego podziału funkcjonalnego przedsiębiorstwa najważniejszych czynności i informacji związanych z jego funkcjonowaniem. Prototyp jest wykonywany pod kątem możliwości wprowadzenia metod informatycznych w przetwarzaniu informacji, racjonalizacji i kompletności obiektu informacji, oraz jest odniesiony do aktualnej struktury organizacyjnej przedsiębiorstwa.
- **Projekt informatyzacji przedsiębiorstwa** – zawiera opisy najważniejszych systemów informatycznych, możliwych do wprowadzenia i potrzebnych w przedsiębiorstwie, a także założenia projektu kompleksowej, etapowej informatyzacji przedsiębiorstwa, ze wskazaniem kolejności i wzajemnych powiązań prac wdrożeniowych, oraz najważniejszych uwarunkowań informatyzacji (tematy, terminy, nakłady, sprzęt, ludzie itp.).
- **Materiały pomocnicze**, głównie notatki z wywiadów przeprowadzonych w trakcie realizacji prac.

Kompleksową analizę funkcjonalną w przedsiębiorstwie średniej wielkości (od 1000 do 2000 zatrudnionych) może wykonać zespół 4–6 specjalistów w czasie 8–12 tygodni. Przeciętny koszt takich prac (w warunkach cenowych i podatkowych z IV kwartału 1990 roku) wynosi od 20 do 60 mln. zł.

Zaprezentowaną metodykę wykorzystano w kilku przedsiębiorstwach o różnych profilach działalności (budownictwo, turystyka, przemysł metalowy, spółdzielczość produkcyjna). We wszystkich przypadkach analiza funkcjonalna była początkiem prac aplikacyjnych, opartych na wynikach analizy. Prace te zwykle były prowadzone przez zespoły ludzi dobranych po wykonaniu analizy. Z przeprowadzonych obserwacji można wyciągnąć następujące wnioski.

- Przydatność merytoryczna i praktyczna materiałów powstałych w wyniku analizy funkcjonalnej jest bezdyskusyjna. Dysponowanie takimi materiałami znacznie upraszcza, przyspiesza i zmniejsza koszt prac aplikacyjnych. Zapewnia też spójność wszystkich aplikacji.

- Koszt poniesiony na wykonanie analizy funkcjonalnej zwraca się w wyniku przyspieszenia i uproszczenia wstępnej części prac aplikacyjnych oraz zapewnienia ich wzajemnej spójności.

- Analiza funkcjonalna przynosiła zazwyczaj korzyści uboczne polegające na zewidencjonowaniu i zracjonalizowaniu funkcjonowania przedsiębiorstwa.

- Skuteczność i wzajemna spójność prac aplikacyjnych była zazwyczaj uzależniona od ich tempa oraz zgodności z ustaleniami analizy funkcjonalnej.

- Niepowodzenia prac aplikacyjnych miały zazwyczaj źródło w dezintegracji zespołów realizujących, odstępstwach od ustaleń analizy funkcjonalnej lub znacznym upływie czasu, dezaktualizującym niektóre ustalenia analizy.

- Dla powodzenia prac nad analizą funkcjonalną duże znaczenie miało zapewnienie zespołowi realizacyjnemu warunków umożliwiających pracę staranną i dokładną (niezbyt napięte terminy realizacji, wystarczające środki materialne, dobra i harmonijna współpraca wewnątrz zespołu itp.).

- Analiza funkcjonalna jest przedsięwzięciem pracochłonnym, gdyż wymaga starannego opracowania dużej ilości dokumentacji o zastosowaniu jednostkowym. Z reguły opłacalność prac przy analizie funkcjonalnej, mierzona relacjami między zyskami a wkładem bezpośredniej pracy, była mniejsza niż dla prac aplikacyjnych. Ze względu na istniejące bariery cenowe, prace przy analizie funkcjonalnej są dla firm usługowych mniej opłacalne od prac aplikacyjnych.

- Występujące obecnie częste i szybkie zmiany podstawowych zasad funkcjonowania gospodarki zazwyczaj nie miały istotnego znaczenia dla ogólnych wyników analiz funkcjonalnych (np. projektu prototypu systemu informacyjnego), chociaż czasem zmieniały obszary zastosowań oraz funkcje systemów dziedzinowych. Powodowało to konieczność weryfikacji wyników analizy funkcjonalnej przed podjęciem prac aplikacyjnych.

- Występujące ostatnio trudności finansowe przedsiębiorstw z jednej strony ograniczyły zainteresowanie wykonywaniem analiz funkcjonalnych, nie przynoszących natychmiastowych efektów, ale z drugiej – zwiększyły zainteresowanie racjonalizacją funkcjonowania przedsiębiorstw, do czego analiza funkcjonalna jest bardzo przydatna.

- Należy przypuszczać, że w miarę zwiększania się zainteresowania funkcjonalnością i wydajnością gospodarki będzie się zwiększać zainteresowanie przedsiębiorstw wykonywaniem analiz funkcjonalnych. Ujednolicenie struktur i zasad funkcjonowania gospodarki pozwoli zmniej-

szyć pracochłonność tych analiz, a więc zwiększy zainteresowanie ich wykonywaniem przez wyspecjalizowane firmy usługowe.

● Dotychczasowe doświadczenia praktyczne wykonywania analiz funkcjonalnych należy uznać za umiarkowanie zachęcające, a perspektywy rozwoju tego kierunku prac za obiecujące i warte dalszego zainteresowania.

Metodyka ustalania faktów

Ustalenie potrzebnych faktów i zgromadzenie niezbędnych informacji jest jednym z najważniejszych zadań stojących przed zespołem analityków, rozpoczynających analizę funkcjonalną lub opracowywanie i wdrażanie systemu informatycznego. W pewnym uproszczeniu można przyjąć, że system informatyczny jest wart co najwyżej tyle, ile były warte prace analityczne mające na celu ustalenie faktów i okoliczności związanych z genezą, funkcjonowaniem i oczekiwaniami użytkowników wobec systemu. Problem ten staje się szczególnie ważny, jeśli zespół opracowujący system przychodzi z zewnątrz, a zatem nie zna wystarczająco dobrze specyfiki i funkcjonowania procesów, które ma informatyzować. Metodyka przygotowania i prowadzenia wywiadów oraz rejestrowania, analizowania i weryfikowania uzyskanych informacji jest obecnie sporą dziedziną wiedzy, a w wielu zawodach umiejętność prawidłowego ustalenia potrzebnych faktów decyduje o sukcesie. Wykorzystuje się przy tym wiedzę i doświadczenia zaczerpnięte z wielu dziedzin nauki (socjologia, psychologia, logika, ekonomia, retoryka, kryminalistyka itp.). Dlatego warto zaprezentować kilka doświadczeń praktycznych, zebranych podczas wykonywania kilku kompleksowych analiz funkcjonalnych przedsiębiorstw. Przy opracowywaniu omawianych metod nie korzystano z bogatych doświadczeń instytucji zajmujących się zawodowo umiejętnym prowadzeniem wywiadów i ustalaniem potrzebnych faktów, gdyż tego rodzaju doświadczenia na ogół nie są publikowane. Wykorzystano natomiast fragmentaryczne dane i zalecenia dostępne w literaturze („Wstęp do analizy systemów" Danielsa i Yatesa), a przede wszystkim własne doświadczenia, przemyślenia i eksperymenty.

Staranne i dokładne przygotowanie cyklu wywiadów jest kluczową sprawą dla końcowego sukcesu prac nad ustaleniem potrzebnych faktów. Prace i nakłady poniesione na przygotowanie, przeprowadzenie i opracowanie wywiadów bardzo szybko procentują mniejszym wysiłkiem włożonym w opracowanie szczegółowych założeń funkcjonalnych systemów oraz lepszą jakością, dokładnością i kompletnością wiedzy o informatyzowanych problemach. Dobre przygotowanie i przeprowadzenie wywiadów ma szczególne znaczenie w kompleksowej analizie funkcjonalnej przedsiębiorstw, gdyż wywiady są dla niej podstawowym, a często jedynym źródłem informacji.

Procedury przygotowania i przeprowadzenia wywiadów można podzielić na następujące etapy:

● Zestawienie celów, problemów i tematów analizy.
● Zestawienie problemów wymagających przeprowadzenia wywiadów.
● Określenie wzajemnych uzależnień i powiązań między problemami wymagającymi wyjaśnienia.
● Zestawienie tematów i kolejności prowadzenia wywiadów.
● Ustalenie komórek organizacyjnych i osób, z którymi należy przeprowadzić wywiady.
● Ustalenie harmonogramu wywiadów.
● Opracowanie scenariuszy wywiadów.
● Przeprowadzenie wywiadów.
● Udokumentowanie wywiadów (sporządzenie notatek i protokołów) oraz ich autoryzacja.
● Weryfikcja ustalonych faktów i danych oraz zestawienie stwierdzonych braków, niejasności i niezgodności.
● Wyjaśnienie istniejących wątpliwości.
● Umieszczenie dokumentacji wywiadów w zwartym i uporządkowanym wolumenie.

Oto przykłady zestawień wymienionych w pierwszych trzech etapach, a dotyczących gospodarowania materiałami w przedsiębiorstwie.

Określanie zapotrzebowania materiałowego

● źródła, rodzaje i miejsca powstawania zapotrzebowania materiałowego w przedsiębiorstwie,
● metody określania zapotrzebowania materiałowego, informacje potrzebne dla określenia zapotrzebowania, sposoby określania i dokumentowania zapotrzebowania,
● uprawnienia do określania i zatwierdzania zapotrzebowania materiałowego,
● rejestrowanie i bilansowanie zapotrzebowania materiałowego,
● relacje czasowe zapotrzebowania, zamawiania i dostaw materiałowych.

Zaopatrzenie materiałowe

● rozeznanie dostawców i rynku;
● wystawianie i lokowanie zamówień, negocjacja dostaw: materiały podlegające rozdzielnictwu oraz dostępne na rynku;
● zapewnienie transportu (z podziałem na samochodowy, kolejowy i inny) dla dostaw materiałowych: transport własny, dostawcy, klienta, inny;
● planowanie i harmonogramowanie dostaw: dostawy planowane z wyprzedzeniem oraz na bieżąco.

Dyspozycja materiałów

● możliwości dysponowania dostawami: awizowanie, informacje o dostawach,
● kryteria dysponowania materiałami i wstrzymywania dostaw: dyspozycje planowane i nieplanowane,
● uprawnienia dysponowania dostawami,
● zmiany dyspozycji dostaw: skala, wpływ na gospodarkę przedsiębiorstwa, wpływ na koszty produkcji.

Kontrola gospodarki materiałami

● magazynowanie (z podziałem wg magazynów stałych i podręcznych): rozmieszczenie terytorialne, struktura branżowa, stan techniczny, obsługa magazynów (etapy, kwalifikacje, odpowiedzialność), sposób prowadzenia ewidencji magazynowej);
● dokumentacja obrotu materiałowego zawierająca: miejsce rejestrowania, rodzaj operacji, formy rejestrowania, uprawnienia, obieg dokumentacji (schematy, relacje czasowe), warunki rejestracji, w podziale na: przychody, rozchody, zapasy (rodzaje, kryteria klasyfikacji, uprawnienia do kwalifikowania, zapasy niejawne, dysponowanie i gospodarowanie zapasami, wpływ na koszty, rola banku, odpady, przedmioty nietrwałe, materiały obce), inwentaryzacje (rodzaje i terminy, rzetelność danych, wykorzystanie wyników);
● weryfikacja zużycia materiałów w stosunku do norm: źródła i rodzaje norm, planowanie zużycia materiałów, porównywanie zużycia faktycznego i normatywnego, postępowanie w wypadku rozbieżności.

Inne

● Zasady organizacyjne obrotu materiałowego w przedsiębiorstwie: przepisy ogólne, przepisy wewnętrzne.

Po zestawieniu problemów wymagających wyjaśnienia należy opracować harmonogram wywiadów. Będzie on zawierał listę tematów, miejsc i kolejności wywiadów, a także osoby, z którymi należy te wywiady przeprowadzić. Harmonogram wywiadów powinien uwzględniać wzajemne związki między ustalanymi faktami i powinien być przygotowany w sposób zapewniający logiczną kolejność wywiadów.

Po ustaleniu harmonogramu wywiadów należy przygotować scenariusze wywiadów.

Przykładowy scenariusz wywiadu

● Przedstawiamy się, ustalamy dane rozmówcy.
● Omawiany krótko:
– powody przeprowadzenia wywiadu (krótkie informacje o przyczynach, o metodyce prac oraz celu i zakresie tematycznym wywiadu),
– przeznaczenie uzyskanych informacji (podkreślić, że nie jest to kontrola!),
– sposób przeprowadzenia wywiadu: środki techniczne, forma rejestracji i autoryzacji wywiadu).
● Uzyskujemy akceptację i deklarację współpracy rozmówcy lub uwzględniamy jego ewentualne zastrzeżenia i uwagi.
● Przeprowadzamy wywiad. Pamiętamy, że:
– przebiegiem rozmowy powinien sterować prowadzący wywiad,
– nie należy ograniczać swobody wypowiedzi rozmówcy,
– nie należy dopuścić do znacznych odstępstw od planowanego tematu

rozmowy (o ile treści uboczne nie interesują prowadzącego wywiad),
– należy na bieżąco rozładowywać ew. konflikty, zdenerwowanie lub nieufność rozmówcy.
– należy na bieżąco wyjaśniać wszelkie kwestie wątpliwe i kontrowersyjne,
– należy dbać o sympatyczną i swobodną atmosferę rozmowy,
– nie wolno sugerować rozmówcy treści wypowiedzi,
– należy na bieżąco kontrolować stan techniczny i sprawność używanego sprzętu.

- Krótko podsumowujemy ustalenia wywiadu (o ile to możliwe).
- Pytamy o sprawy nie uwzględnione w dotychczasowej rozmowie, które zdaniem rozmówcy mogą mieć znaczenie dla tematyki wywiadu.
- Pytamy o ewentualne sugestie rozmówcy dotyczące dalszego rozeznawania tematyki wywiadu.
- Uwzględniamy formę i terminy dalszych kontaktów (autoryzacja wywiadu, uzupełnienia itp.).
- Dziękujemy za rozmowę, żegnamy się.

Na pierwszy rzut oka scenariusz wywiadu wygląda trywialnie i potrzeba jego przygotowania wydaje się wątpliwa. Jest jednak nieoceniony w sytuacji, kiedy wywiad jest prowadzony w trudnych warunkach (ruchliwe biuro, hałaśliwy wydział produkcyjny, pośpiech itp). Scenariusz w połączeniu z przygotowanym zestawieniem problemów merytorycznych daje prowadzącemu wywiad gwarancję, że żadna istotna sprawa nie zostanie podczas wywiadu zapomniana, a przebieg wywiadu i uzyskane informacje będą odpowiadać celom i oczekiwaniom. Daje on też prowadzącemu wywiad pewnego rodzaju komfort psychiczny, wynikający z faktu, że można skoncentrować się na problemach merytorycznych, gdy problemy formalne i proceduralne ma uporządkowane. Przygotowanie scenariusza wywiadu zmusza też do starannego przemyślenia treści, sposobu prowadzenia, celów i oczekiwanych rezultatów wywiadu. W znacznej mierze eliminuje to czynniki przypadkowości i improwizacji, które są podstawowymi powodami błędów i niekompletności ustaleń podczas wywiadów.

Kolejnym problemem jest sposób rejestrowania przebiegu wywiadu. Bardzo praktyczną i pożyteczną formą rejestrowania jest nagrywanie całego wywiadu na taśmie magnetofonowej, a następnie sporządzanie notatek z wywiadu na podstawie nagrania. Metoda ta ma kilka bardzo ważnych zalet:
– usprawnia prowadzenie wywiadu, eliminując kłopoty związane z koniecznością bieżącego notowania ustaleń,
– umożliwia uporządkowanie notatek według tematów, gdyż ustalenia dotyczące tego samego tematu można zebrać z różnych fragmentów i spisać razem,
– umożliwia dokładne zanotowanie wypowiedzi rozmówcy, gdyż fragmenty niezrozumiałe można przesłuchać wielokrotnie,
– umożliwia zapoznanie się z treścią i przebiegiem wywiadu osobom, które nie brały w nim bezpośredniego udziału,
– umożliwia zarejestrowanie faktów i okoliczności ulotnych (stan emocjonalny, rozmówcy, atmosfera rozmowy, warunki prowadzenia wywiadu, błędy metodyczne popełnione przy jego prowadzeniu itp.),
– umożliwia zarejestrowanie i wynotowanie maksymalnej ilości informacji uzyskanych podczas wywiadu.

Metoda nagrywania wywiadu ma również kilka wad:
– często wywołuje rezerwę i nieufność rozmówcy (bezwarunkowo należy je rozładować przed rozpoczęciem wywiadu!), wymaga zatem jego akceptacji; w razie braku takiej akceptacji należy zrezygnować z nagrywania treści wywiadu, poprzestając na notatkach sporządzanych ręcznie;
– stwarza komplikacje, jeśli wywiad dotyczy spraw poufnych;
– angażuje dodatkowe środki techniczne i wprowadza dodatkowe ryzyko związane z ich wykorzystaniem (sprawność, niezawodność, konieczność kontrolowania i nadzoru);
– zwiększa pracochłonność opracowywania wywiadu (jest to wada względna, gdyż nagranie wywiadu dostarcza znacznie więcej informacji niż notatki spisane podczas wywiadu).

Ostatnio pojawiły się na rynku krajowym dyktafony produkcji japońskiej, znacznie ułatwiające nagrywanie wywiadów. Są one niezbyt drogie (ok. 80 $) i bardzo praktyczne w użyciu: mają niewielkie rozmiary, są wyposażone w wiele ułatwień obsługi (kasety typu Compact, licznik przewijania taśmy, szybkie przewijanie nagrania z podsłuchem, klawisz „stop dynamiczny", zmienną szybkość odtwarzania nagrania, proste i wydajne źródła zasilania, wbudowany mikrofon o bardzo dobrych parametrach, wbudowany głośnik). Użycie dyktafonu eliminuje podstawowe wady związane z używaniem konwencjonalnych magnetofonów, gdyż dyktafon jest znacznie mniejszy, bardziej niezawodny technicznie i prostszy w obsłudze. Nie bez znaczenia jest przy tym fakt, że używanie dyktafonu na ogół znacznie mniej krępuje rozmówcę, gdyż dyktafon jest urządzeniem bardziej dyskretnym i mniej rzucającym się w oczy niż duży magnetofon z mikrofonem bezpośrednio podstawianym rozmówcy. Praktyczne doświadczenia stosowania dyktafonów dowiodły, że są to urządzenia bardzo przydatne i godne polecenia.

Na ogół najbardziej pracochłonną częścią całego procesu przeprowadzenia wywiadu jest sporządzanie notatek na podstawie nagrania. Wymaga to bowiem starannego (zazwyczaj kilkakrotnego) przesłuchania nagrania, uporządkowania zarejestrowanych informacji i spisania ich w zwartej, usystematyzowanej formie. Z dotychczasowych doświadczeń praktycznych wynika, że jedna godzina nagrania na taśmie magnetofonowej wymaga około 6–8 godzin opracowywania notatek. Nakłady te są jednak rekompensowane bogatą zawartością treściową notatek i ich usystematyzowaniem. Można zatem przyjąć, że technika nagrywania wywiadów zmniejsza kilkakrotnie łączną czaso- i pracochłonność ustalenia porównywalnego zasobu informacji przy użyciu metodyki tradycyjnej.

Zarejestrowane notatki z wywiadu są następnie przekazywane do autoryzacji osobom, z którymi wywiad był przeprowadzany. Jest to końcowy, ale bardzo istotny element weryfikacji informacji ustalonych podczas wywiadu. Przy autoryzacji należy osobie autoryzującej zapewnić warunki dokładnego zapoznania się z treścią notatek, umożliwić jej naniesienie ewentualnych poprawek lub uzupełnień oraz uzyskać jej opinię o kompletności zarejestrowanych informacji. Autoryzacja notatek z wywiadu powinna być uwierzytelniona podpisem osoby autoryzującej. Osobie autoryzującej notatki należy też zwrócić uwagę na znaczenie, jakie dla późniejszych prac ma prawidłowość i rzetelność zarejestrowanych informacji.

Zaproponowane powyżej metody przygotowania, prowadzenia i dokumentowania wywiadów były wielokrotnie stosowane w praktyce z bardzo dobrymi wynikami. Są one zatem godne polecenia wszędzie tam, gdzie sprawne i prawidłowe ustalenie faktów ma kluczowe znaczenie dla wyników przedsięwzięcia (typowym przypadkiem jest tu kompleksowa analiza funkcjonalna). Metody te można też uogólnić i adaptować w innych dziedzinach zastosowań, gdyż ich podstawowa geneza wywodzi się z zasady, że wszelkie działania będą sensowne i efektywne, jeśli zostaną wcześniej starannie przemyślane, dobrze przygotowane, i sprawnie zrealizowane z użyciem dostępnych środków technicznych.

Wskazówki dla Autorów

Nadsyłane artykuły nie mogą być publikowane lub przeznaczone do opublikowania w innych czasopismach.

Materiał oprócz tekstu zasadniczego powinien zawierać:
- ★ krótki życiorys zawodowy Autora i jego zdjęcie,
- ★ wykaz literatury,
- ★ tabele,
- ★ materiał ilustracyjny (rysunki, zdjęcia czarno-białe, wydruki) dołączony do artykułu (nie wklejać materiału w tekst),
- ★ podpisy pod ilustracje.

Na osobnej stronie prosimy podać tytuł naukowy, imię i nazwisko, nazwę zakładu pracy, adres domowy (koniecznie!), numer telefonu oraz informację, jaką drogą przekazać honorarium – kasa Wydawnictwa, poczta, bank.

Tekst artykułu prosimy dostarczyć w formie maszynopisu lub wydruku komputerowego, pisany jednostronnie, z podwójnym odstępem (30 wierszy na stronie i 60 znaków w wierszu).

Rysunki winny być czytelne i mogą być wykonane ołówkiem.

Autor opublikowanego w INFORMATYCE artykułu otrzymuje honorarium i bezpłatny egzemplarz okazowy.

Materiałów nie zamówionych redakcja nie zwraca.

P – CIM

PERSONAL COMPUTER INTEGRATED SYSTEM

kompletny system oprogramowania SCADA dla zastosowań przemysłowych

W procesie produkcji są wykorzystywane cztery podstawowe składniki: kapitał i park maszynowy, surowce, zasoby ludzkie oraz informacja. W ciągu ostatnich lat największy postęp w dziedzinie wytwarzania nastąpił tylko w dwóch z wymienionych składników: udoskonalono park maszynowy oraz uznano, że czynnik ludzki ma wiodącą rolę. W pozostałych składnikach (surowce, informacja) nie pojawiło się tak wiele zmian. Ponieważ możliwości poprawy surowców są oczywiście w sposób naturalny ograniczone, największego przełomu oczekiwano w dziedzinie informacji.

Dlaczego systemy SCADA?

Zasoby informacyjne są podstawą dla rozwijającej się ostatnio dziedziny nadzoru sterowania i pozyskiwania danych (*Supervisory Control And Data Acquaisition* – SCADA). Standardowy system SCADA jest zbiorem programów komputerowych, który ma zastąpić przestarzałe, dedykowane i z tego powodu ograniczone i mniej elastyczne systemy informatyczne, jakich używano w przeszłości. Systemy SCADA realizują funkcje:

- zbierania danych
- analizy danych
- wizualizacji parametrów
- zarządzania procesami.

Czym jest P-CIM?

System P-CIM obejmuje zintegrowane oprogramowanie obsługi alarmów, obróbki zdarzeń, sterowania oraz pozyskiwania i analizy danych. Działa on na komputerach IBM PC XT/AT, PS/2 lub kompatybilnych, wyposażonych w programowane sterowniki i odpowiednie urządzenia peryferyjne. W zależności od wybranego czasu przeglądania, system zbiera w czasie rzeczywistym dane pochodzące z instalacji, gromadzi je w bazie danych i automatycznie przetwarza w zależności od poleceń użytkownika. Dzięki temu użytkownik może zdefiniować warunki, po spełnieniu których nastąpi wygenerowanie błędu wytwarzania bądź informacji o zdarzeniu, albo też przeprowadzenie obliczenia na podstawie uzyskanych wartości. System P-CIM pozwala użytkownikowi zaprojektować dynamiczną prezentację wszystkich aspektów nadzorowanego procesu w postaci graficznej lub tabelarycznej. Użytkownik decyduje o tym, który element instalacji i w jaki sposób ma być prezentowany na ekranie oraz czy operator może zmienić wielkość. Oprócz tego upoważniony użytkownik ma dostęp do opcji pozwalających na bezpośrednią interwencję w przebieg procesu. Operator ma więc na ekranie wyświetlony w skoncentrowanej postaci dynamiczny obraz i tekst każdego procesu wytwarzania, realizowanego w instalacji nadzorowanej przez system P-CIM.

Charakterystyka systemu P-CIM

System P-CIM jest produktem amerykańskiej firmy Afcon. Celem uzyskania maksymalnie efektywnego oprogramowania rozwiązanie systemu oparto na trzech podstawowych założeniach:

Otwarta architektura. Systemy o otwartej architekturze charakteryzuje modułowość. Otwarta architektura to również możliwość rozwoju poziomego i pionowego. Rozwój poziomy oznacza, że ten sam system może być użyty w instalacji z niewielką oraz dużą liczbą sterowników danego typu. Rozwój pionowy oznacza, że ten sam system można stosować w instalacjach różnego typu, na przykład w przemyśle spożywczym, chemicznym, papierniczym, w laboratoriach i hutach.

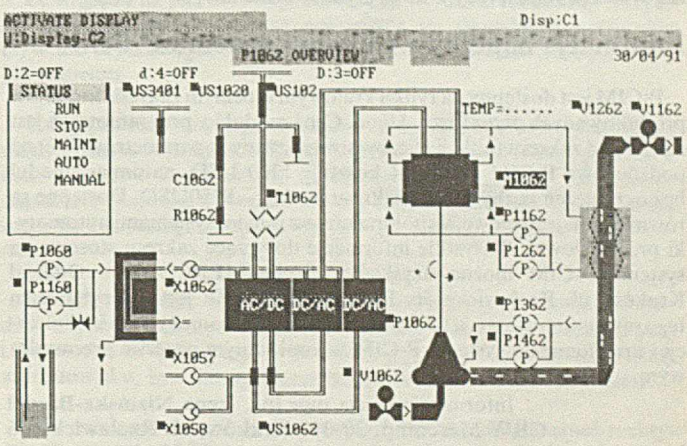
W systemie P-CIM każdy użytkownik może – w miarę potrzeb – dołączyć swoje dodatkowe oprogramowanie napisane w języku C lub innym popularnym języku programowania. Możliwe jest również pisanie przez użytkownika własnych protokołów komunikacyjnych.

Optymalny dostęp do informacji. Podczas eksploatacji systemu P-CIM dostęp do niego będzie miało wielu użytkowników o różnych potrzebach, wiedzy i doświadczeniu. System powinien więc nie tracić czasu na prezentowanie informacji zbędnych dla użytkownika. Użytkownik ten powinien koncentrować się wyłącznie na działaniu instalacji, a nie na obsłudze komputera. Podejście takie spowoduje skrócenie czasu opanowania obsługi systemu do niezbędnego minimum.

Prostota obsługi. Komputer jest wyłącznie narzędziem, które nie zastąpi oceny, inicjatywy i zdolności twórczych człowieka. Interfejs operator-proces powinien zapewniać człowiekowi (a nie systemowi) pełną kontrolę nad procesem. Można to osiągnąć dzięki tworzonej przez użytkownika bazie danych, ekranom informującym na bieżąco o przebiegu procesu oraz automatycznemu generowaniu raportów i ładowaniu receptur do sterownika PLC (*Programmable Logical Controller*). Należy więc zapewnić proste w obsłudze narzędzia do sterowania, monitorowania, obsługi awarii i zbierania informacji o instalacji z możliwością ich dostosowania do przyszłych potrzeb. Wszystko to zrealizowano w systemie P-CIM przez zastosowanie języka czwartej generacji, który pozwala całkowicie zautomatyzować instalację bez umiejętności programowania. Dzięki temu instalacja może być – w zależności od ustalonego wcześniej czasu, daty, warunków lub kombinacji tych czynników – włączana lub wyłączana według ustalonej procedury procesu przemysłowego.

Zalety systemu P-CIM

● **Kompatybilność.** System P-CIM został wykonany jako standard IBM, wykorzystuje system operacyjny MS-DOS i może być uruchomiony na dowolnym modelu komputera IBM XT, AT, 7531, 5531 lub kompatybilnym, a także na wszystkich modelach rodziny IBM PS/2. Przyjęto grafikę sprzętu IBM na kartach CGA, EGA lub VGA, jak również wykorzystanie graficznej biblioteki VDI. P-CIM jest systemem



Przykład schematu technologicznego w systemie P-CIM

w pełni wielozadaniowym, pozwalającym uruchamiać w tle własne zadania jako zadania pierwszoplanowe.

- **Uniwersalność.** P-CIM współpracuje z większością dostępnych obecnie na rynku sterowników przemysłowych. Te sterowniki PLC, których system P-CIM standardowo nie obsługuje, można w prosty sposób z nim zintegrować.
- **Zasobność.** W systemie są dostępne praktycznie wszystkie funkcje wspomagające prowadzenie procesu przemysłowego.
- **Elastyczność.** W zależności od wymagań użytkownika jest możliwe zainstalowanie takiej wersji systemu, która wymagania te spełnia najlepiej.
- **Sprawność działania.** Obsługa wejść i wyjść procesowych oraz alarmów jest realizowana w czasie rzeczywistym na tle pracy komputera, a wielozadaniowość zapewnia operatorowi przyspieszenie bezpośredniej komunikacji z procesem.
- **Pojemność systemu.** Możliwość obsługi dużej liczby danych procesu z możliwością ich przechowywania na stosunkowo małym obszarze pamięci wewnętrznej.
- **Łatwość obsługi.** P-CIM jest dostarczany z pełnym tekstem podpowiedzi i dokumentacją. Łatwość nauczenia się obsługi systemu, okienka, menu oraz kolorowe ekrany sprawiają, że P-CIM jest systemem przyjaznym dla użytkownika.

Wymagania sprzętowe i programowe

Wymagania sprzętowe:
IBM PC XT/AT, PS/2 lub sprzęt kompatybilny
pamięć minimum 512 KB (zaleca się 640 KB)
dysk twardy: minimum 5 MB
karta graficzna: EGA, VGA lub oryginalna IBM CGA z odpowiednim monitorem (karta EGA wymaga dodatkowych 256 KB pamięci)
port równoległy dla drukarki
port szeregowy RS 232C (1 lub 2) dla komunikacji z siecią sterowników obiektowych.

Opcje sprzętowe:
koprocesor 8087/80287
moduł komunikacyjny systemu P-CIM (*P-CIM Communication Module* – PCOM)
drukarka kompatybilna ze standardem IBM
port szeregowy dla myszy.

Wymagania programowe:
PC-DOS lub MS-DOS wersja 3.1. lub późniejsza
dla komputerów rodziny PS/2 – DOS wersja 3.3.

Sterowniki obiektowe oraz inne urządzenia peryferyjne:
Komputer przyłącza się do instalacji przemysłowej przez sterowniki obiektowe wykorzystując port szeregowy RS 232C i odpowiedni protokół komunikacyjny. Protokół komunikacyjny jest oprogramowaniem, przesyłającym rozkazy do sterowników obiektowych w celu uzyskania informacji. Istnieje również możliwość adaptacji własnego protokołu komunikacyjnego do współpracy z systemem P-CIM za pomocą pakietu narzędziowego do tworzenia oprogramowania komunikacyjnego (*Communication Development Toolkit* – CDT), dostępnego w firmie Afcon. Narzędzie to, stanowiące część systemu P-CIM, umożliwia programistom prostą modyfikację protokołu komunikacyjnego w celu współpracy z dowolnym systemem.

Informacja handlowa na temat systemu P-CIM

P-CIM jest dostępny na rynku krajowym w pełnym zakresie modułów produkowanych przez firmę Afcon. Cena modułów programowych jest zależna od zakresu funkcji i liczby przetwarzanych parametrów. Wersja podstawowa P-CIM JUNIOR kosztuje 1200 USD, natomiast moduł bez ograniczeń liczby wejść i zakresu funkcji – 8250USD. Dostępne są również wersje sieciowe, współpracujące z innymi systemami automatyki przemysłowej. Wszystkie informacje dotyczące zakresu stosowania systemu P-CIM można uzyskać w firmie MERCOMP – Oddział Kraków, ul. Racławicka 56. Firma MERCOMP jest dystrybutorem tego oprogramowania, a także udziela wszelkiej pomocy przy kompletacji i uruchomieniu systemu P-CIM w konkretnym procesie przemysłowym.

Informacji udziela **mgr inż. Irena Nizińska-Bardel**
CBW Mercomp, 30-017 Kraków, ul. Racławicka 56
tel. Kraków 34-50-00, 34-51-00, faks 33-00-67

BARBARA ŁUKASIK-MAKOWSKA
Instytut Informatyki Ekonomicznej
Akademia Ekonomiczna
Wrocław

Standaryzacja systemów powielarnych

Od kilku lat krajowy rynek produktów informatycznych rozwija się żywiołowo, głównie za sprawą sprzętu mikrokomputerowego. Wobec umiarkowanych cen i powszechnej dostępności tego sprzętu nastąpiło zwiększenie zapotrzebowania na produkty usprawniające różnorodne prace administracyjne i obliczeniowe, głównie w dziedzinie ewidencji gospodarczej i szeroko pojętego zarządzania przedsiębiorstwem. Zjawisko to powinno cieszyć wszystkich zainteresowanych, lecz niestety badanie ofert producentów, konfrontowanie z nimi oczekiwań klientów, a także śledzenie losów wielu zawartych transakcji, ukazują tak wiele nieprawidłowości, iż wyłania się z tego całkowicie odmienny i wręcz przygnębiający obraz sytuacji.

Sednem wielu nieporozumień i problemów są systemy użytkowe, które nie odpowiadają potrzebom nabywców, często są dostarczane w wersji roboczej (nie dokończonej), kryją w sobie wiele ukrytych błędów, a poza tym są zwykle bardzo mało odporne na wszystkie zjawiska zewnętrzne, zmieniające zakres potrzeb ich użytkowników (brak jest możliwości ich bieżącej adaptacji).

Poza patologicznymi przypadkami świadomych nadużyć ze strony twórców tych produktów można stwierdzić, iż wiele nieprawidłowości wynika z braku odpowiednich ustaleń „warsztatowych". Analiza dotychczasowych doświadczeń krajowych w tej dziedzinie wskazuje, iż to właśnie brak standardów dotyczących rozwiązań funkcjonalnych, konstrukcyjnych i technologicznych stanowi jedno z głównych źródeł błędów, nieefektywności i niepowodzeń w trakcie wdrażania i użytkowania systemów w praktyce gospodarczej. Brak standaryzacji i rozwiązań zwiększa koszty i wydłuża czas tworzenia systemów. Jest to zarazem bariera na drodze upowszechnienia informatyki w ponadewidencyjnych zastosowaniach gospodarczych. Jednocześnie obserwowane przemiany krajowej gospodarki wymagają możliwie szybkiego i skutecznego wsparcia zarządzania nowoczesnymi technikami przetwarzania informacji.

Z tego też względu standaryzację należy uznać za środek niekonfliktowego, aktywnego i racjonalnego oddziaływania na rynek produktów informatycznych, przyspieszający jego stabilizację i normalizację (w sensie doprowadzenia do stanu, jaki można obserwować na rynkach informatycznych innych krajów).

Znaczenie standaryzacji dla rozwoju systemów informatycznych

Standaryzacja jest to świadomie i celowo realizowany proces zmierzający do zdefiniowania wzorca. Bardzo często standaryzację utożsamia się z normalizacją. Ta ostatnia jest traktowana jako proces wprowadzania obowiązujących przepisów (norm), dotyczących przedmiotu badań. Istnieje jednak istotna różnica między standaryzacją a normalizacją – druga z nich jest realizowana zazwyczaj przez określony przedmiot, mający uprawnienia legislacyjne, a przestrzeganie rozwiązań wzorcowych może być stymulowane środkami przymusu prawnego. W tym sensie standaryzacja jest pojęciem szerszym, a zarazem mniej kategorycznym niż normalizacja. Standard może być rozwiązaniem zalecanym, akceptowanym zwyczajowo, o zasięgu zróżnicowanym (firmowym, branżowym, krajowym lub międzynarodowym).

Dla swych użytkowników standard musi stanowić jednolity przedmiotowo zbiór jednoznacznych i niesprzecznych wewnętrznie norm. Zatem normę można traktować jako jednostkowy element standardu. Normą może być wskazówka, charakterystyka, ograniczenie, wymaganie. Równie niejednorodna może być postać graficzna, poziom szczegółowości i poziom formalizacji normy. Oprócz wskazanych wyżej cech (jednoznaczności i niesprzeczności) normy powinny mieć następujące właściwości:

- czytelność,
- precyzję,
- jednolitość interpretacyjną,
- spójność zewnętrzną.

Brak wskazanych właściwości w istotnym stopniu ogranicza przydatność, zasadność i zakres oddziaływania norm.

Standaryzacja nie jest procesem jednolitym. Sposób jej realizacji wykazuje bardzo dużą zależność od przedmiotu standaryzacji i rodzaju standardu, który ma stanowić efekt prac. W odniesieniu do informatyki można mówić o standaryzacji procesów i standaryzacji wytworów. **Standaryzacja procesów** prowadzi do powstania standardów metodycznych (jak wykonać dane zadanie); **standaryzacja wytworów** umożliwia opracowanie standardów produktów. Procesy tworzenia produktów informatycznych są bardzo zróżnicowane, z jednej strony specyfiką tychże produktów, z drugiej dysponowanym zbiorem narzędzi technik i technologii wytwarzania, itp. Produkty informatyczne są bardzo niejednorodne. Przykładowo można więc mówić o standardach:

- sprzętu komputerowego,
- systemów operacyjnych,
- oprogramowania narzędziowego,
- systemów użytkowych,
- usług informatycznych.

Można stwierdzić, że interesują nas reguły standaryzacji tej grupy produktów, które określa się umownie użytkowymi systemami informatycznymi. Nie determinuje to jeszcze sposobu realizacji prac standaryzacyjnych. Efekty tej działalności mogą być bowiem różnorodne, a czynnikami decydującymi o tym będą:

- istnienie (lub nieistnienie) w rzeczywistości produktów standaryzowanych,
- zakres i szczegółowość standaryzacji,
- przewidywany zasięg opracowanego standardu.

Ze względu na pierwszy czynnik można wyróżnić **standardy aktywne**, gdy standard poprzedza pojawienie się określonego produktu (systemu użytkowego) i **reaktywne**, gdy standard ujednolica produkty już istniejące.

Uwzględnienie drugiego czynnika pozwala wyróżnić **standardy pełne i cząstkowe**. Standardy pełne dotyczą całych systemów, wszystkich aspektów i wszystkich elementów strukturalnych. Standardy cząstkowe obejmują wybrane elementy systemowe, nie przesądzając ani istnienia i postaci pozostałych elementów, ani ich wzajemnych powiązań.

Biorąc pod uwagę ostatni czynnik można wskazać istnienie standardów międzynarodowych, krajowych, środowiskowych (branżowych) i firmowych. W tych ostatnich można by ponadto mówić o randze firmy w środowisku, lecz dla uproszczenia aspekt ten zostanie pominięty.

Jako minimum zaspokojenia bieżących potrzeb należy uznać reaktywne standardy cząstkowe o zasięgu firmowym lub środowiskowym,

oczekiwane maksimum powinny natomiast stanowić aktywne lub reaktywne kompletne standardy o zasięgu minimum krajowym.

Należy jednak zdawać sobie sprawę, że dywersyfikacja standardów wynika nie tylko z czynników zewnętrznych. Do podstawowych czynników wewnętrznych wpływających na postać standardów należy zaliczyć charakter używanych norm, poziom ich uwarunkowania oraz wzajemną zgodność (niesprzeczność). Normy tworzone w ramach standardów mogą mieć charakter:

- **norm przedmiotowych**, wyznaczających zakres działalności i obiekty realizacyjne, odnoszących się zwłaszcza do elementów struktury konstrukcyjnej systemu,
- **norm czynnościowych**, określających technologię przetwarzania danych,
- **norm czasowych**, wyznaczających dopuszczalne czasy realizacji zadań użytkowych lub ich wyróżnionych części,
- **norm semantycznych**, dotyczących głównie stosowanej terminologii, odnoszących się jednak także do procesu komunikacji użytkownika z systemem,
- **norm informacyjnych**, związanych z zasadami kodowania danych, obowiązującymi cennikami, taryfikatorami itp.,
- **norm mieszanych**, gdy przedmiot standaryzacji jest złożony i wymaga wielostronnej i wyczerpującej analizy.

Poziom uwarunkowania norm różnicuje je na zdeterminowane i autonomicze. **Normy zdeterminowane** to takie, których postać i treść jest implikowana jednoznacznymi wynikami badań teoretycznych i praktycznych, w zasadzie nie do podważenia lub wariantowania przez użytkowników standardu. **Normy autonomiczne** to normy, których uzasadnienie nie jest tak precyzyjne, wyczerpujące i jednoznaczne (bierze się pod uwagę badania wycinkowe, indywidualne bądź grupowe doświadczenia praktyczne, typowe sytuacje i rutynowe działania). Normy autonomiczne są zazwyczaj mniej trwałe i stabilne, zwłaszcza gdy dotyczą problemów i obiektów znajdujących się w stanie intensywnego rozwoju.

Konkretna norma może znaleźć zastosowanie w dowolnej liczbie standardów. Ustalenie zbioru norm, które złożą się na określony standard, jest realizowane podczas prac standaryzacyjnych.

Standaryzacja użytkowych systemów informatycznych jest procesem umożliwiającym ograniczanie wielu możliwych rodzajów systemów (realizujących określony kompleks zadań użytkowych) oraz ich elementów. Standaryzacja prowadzi także do ustalenia wspólnego, jednakowego wzorca w procesie tworzenia użytkowych systemów informatycznych, pozwalając na racjonalną organizację tego procesu.

Standardowy system użytkowy powinien łatwo poddawać się modyfikacjom przez redukcję, dołączanie lub wymianę poszczególnych jego elementów, co zwiększa elastyczność oraz sprawność funkcjonalną tego systemu. Walory te powodują, że postępująca standaryzacja powinna zwiększać adaptowalność użytkowych systemów informatycznych w stosunku do zmieniających się wymagań użytkownika oraz zmian w otoczeniu systemu.

Można zatem stwierdzić, że celem standaryzacji użytkowych systemów informatycznych jest z jednej strony **typizacja**, to jest ograniczenie różnorodności systemów realizujących określony kompleks zadań użytkowych, z drugiej zaś – **unifikacja**, to znaczy wprowadzenie wspólnych elementów do różnych systemów, co prowadzi do nadania wielu różnym systemom wspólnych cech użytkowych. Celami pośrednimi, osiąganymi równolegle do wymienionych, jest zwiększenie elastyczności systemu oraz jego modularności.

Pojęciem, które wymaga jednoznacznego zdefiniowania, jest termin „system powielarny". Często w literaturze oraz przez producentów systemów użytkowych są używane synonimy tego terminu – „system powtarzalny" lub „system typowy". **System powielarny** jest to użytkowy system informatyczny mający charakter rozwiązania uniwersalnego dla danej klasy zastosowań.

Dotychczasowa praktyka implementacji systemów powielarnych wskazuje, iż uniwersalność tych systemów jest z reguły mocno ograniczona. Wynika to z faktu, że systemy powielarne są na ogół tworzone na podstawie wybranego systemu informatycznego uznanego arbitralnie za typowy dla danej klasy zastosowań. Występujące w praktyce różnice w specyfice działania systemów określonej klasy (nawet w obrębie tej samej branży) oraz niestabilność systemu zarządzania powodują znaczne ograniczenie deklarowanej przez twórców uniwersalności tych produktów, mierzonej stopniem ich dopasowania dla większej liczby potencjalnych użytkowników.

Za systemy powielarne przyjęto uznawać użytkowe systemy informatyczne, które pojawiły się na rynku oprogramowania użytkowego i zostały wdrożone w co najmniej kilku obiektach. System powielarny wcale nie musi być zatem systemem standardowym (w rozumieniu niniejszego opracowania). Z drugiej strony można stwierdzić, że użytkowy system standardowy zawsze będzie systemem powielarnym. Wynika to bowiem z istoty procesu standaryzacji, którego powodzenie zależy przede wszystkim od stopnia uwzględnienia zróżnicowanych potrzeb użytkowników systemów danej klasy.

Pytanie o zasadność prowadzenia standaryzacji użytkowych systemów informatycznych jest podstawową kwestią podnoszoną w wielu dyskusjach. W warunkach światowych główną przesłanką do podejmowania działań standaryzacyjnych jest potrzeba zapewnienia dużej niezawodności programów komputerowych w tzw. zastosowaniach krytycznych [6]. Pojęcie to dotyczy tych sfer zastosowań, w których niesprawność informatyki wiąże się z bezpośrednim zagrożeniem dla życia i zdrowia lub też z istotnymi (w skali społecznej) stratami materialnymi. Jako przykład można tu wymienić pozyskiwanie energii atomowej, badania kosmosu, systemy medyczne, systemy obrony militarnej, przemysł naftowy, chemiczny itp. Standardy wypracowane w tych przedsięwzięciach były następnie przenoszone na inne obszary zastosowań.

W warunkach krajowych nie podejmowano dotychczas tak rozległych i strategicznych tematów komputeryzacji. Specyfiką krajowych zastosowań informatyki, zwłaszcza w ostatnim okresie przemian gospodarczych, jest nieomal masowe zainteresowanie systemami wspomagania zarządzania (i to prawie wyłącznie na najniższym, ewidencyjnym poziomie). A niestety w tej sferze nie zwracano dotychczas uwagi na stosunki i koszty podejmowania określonych decyzji. Dlatego w warunkach krajowych jest potrzebne całowicie inne podejście do standaryzacji.

Należy podkreślić, że podstawowym przedmiotem (i efektem) standaryzacji realizowanej z takim zamysłem powinny być przede wszystkim systemy powielarne o znacznym poziomie elastyczności rozwiązań. Tylko takie bowiem mogą funkcjonować na tym rynku wystarczająco długo, aby spełnić przypisaną im rolę stymulatora rynku. Elastyczność zapewnia bowiem pewną uniwersalność i przydatność systemu:

- różnym użytkownikom, mającym podobne (lecz niekoniecznie identyczne) potrzeby obliczeniowe,
- konkretnemu użytkownikowi przez dłuższy czas eksploatacji tego samego systemu,
- posiadaczom różnych zestawów sprzętu komputerowego (o różnych parametrach eksploatacyjnych),
- użytkownikom dysponującym różnymi (a często również zmiennymi) warunkami eksploatacyjnymi dla systemu,
- do wymiany informacji pomiędzy kilku różnymi użytkownikami tego samego systemu,
- do budowania systemów nadrzędnych opartych na danych pochodzących z analogicznych systemów eksploatowanych przez różnych użytkowników.

Z elastycznością wiąże się ściśle kolejna cecha, którą powinien charakteryzować się standardowy użytkowy system informatyczny – jest nią **modularność** systemu. Modularność wyraża się możliwością:
– wyboru określonej wersji systemu, przez wskazanie potrzebnych modułów funkcjonalnych systemu,
– rozbudowy systemu przez tworzenie i włączanie do struktury systemu własnych modułów (realizujących specyficzne funkcje użytkowe nie objęte standardem),
– rezygnacji z użytkowania określonych modułów funkcjonalnych, których obszar wybiega poza bieżące potrzeby konkretnego użytkownika,
– swobodnego sprzęgania ze sobą różnych standardowych systemów użytkowych (usprawnienie realizacji systemów wielodziedzinowych).

Osiągnięcie założonych celów standaryzacji użytkowych systemów informatycznych, tj. ich typizacji, unifikacji, elastyczności i modularno-

ści, implikuje wiele korzyści, zarówno po stronie producenta systemów użytkowych, jak i ich użytkowników.

Po stronie **twórców użytkowych systemów informatycznych** wśród istotnych korzyści należy wymienić:
- zmniejszenie nakładów na tworzenie i rozwój systemów,
- skrócenie czasu tworzenia systemu,
- ułatwienie tworzenia systemu przez możliwość zwiększenia stopnia specjalizacji projektantów i programistów,
- ułatwienie i przyspieszenie procesu sporządzania dokumentacji projektowo-programowej i wdrożeniowo-eksploatacyjnej,
- możliwość ciągłej weryfikacji i doskonalenia przyjętych standardów,
- możliwość tworzenia i upowszechniania nowoczesnych, zweryfikowanych rozwiązań,
- ułatwienie rozwoju i doskonalenia własnych systemów.

Korzyści osiągane przez **użytkowników systemów informatycznych** to przede wszystkim:
– ułatwienie wyboru spośród systemów oferowanych na rynku (typizacja powoduje ograniczenie ich liczby od tych, które odpowiadają standardowi),
– możliwość porównania i wstępnej oceny oferowanych produktów na podstawie standardowej dokumentacji, spełniającej wymagania jednolitości i komunikatywności,
– możliwość atestacji nowo pojawiających się na rynku produktów programowych (zwiększa to poczucie bezpieczeństwa użytkownika – oferowany produkt, nie spełniający wymagań standardu, może być odrzucony już we wstępnej fazie wyboru),
– ułatwienie i przyspieszenie procesu wdrażania systemu,
– znaczne ułatwienia w modyfikowaniu i rozbudowywaniu eksploatowanego systemu użytkowego (dotyczy to zarówno zmian w oprogramowaniu użytkowym, jak i zmian związanych z techniczną rekonfiguracją posiadanego zestawu sprzętu),
– zwiększenie możliwości integracji użytkowych systemów informatycznych (dotyczących różnych kompleksów zadań użytkowych), przez wymagane standardem, sprzęgi międzynarodowe,
– mniejsze jednostkowe koszty produktu,
– zwiększenie możliwości wymiany doświadczeń między użytkownikami systemów o podobnych zakresach merytorycznych.

Wynika stąd, że standaryzacja użytkowych systemów informatycznych przynosi obu stronom znaczące i wymierne korzyści. Powoduje również korzystne **zmiany na rynku oprogramowania użytkowego** przez:
- naturalną selekcję produktów niedopracowanych (o niedostatecznej elastyczności i modularności, niezgodnych z przyjętymi standardami),
- naturalną eliminację producentów, których produkty nie spełniają wymagań standaryzacyjnych,
- stały dopływ na rynek kolejnych wersji (generacji) systemów użytkowych, które zyskały aprobatę odbiorców,
- szerszą ofertę producentów w dziedzinie systemów ponadewidencyjnych (których podstawą realizacji będą obowiązujące standardy systemów użytkowych na poziomie ewidencyjnym).

Upowszechnienie standaryzacji spowoduje, iż każdy producent, chcąc utrzymać się na rynku, będzie sam dążyć do uzyskania atestu na oferowane przez siebie oprogramowanie.

Wszystkie korzyści, których źródłem jest standaryzacja oraz stymulowane przez nią pozytywne zmiany na rynku oprogramowania użytkowego, należy uznać za ważne przesłanki skłaniające do intensywnych działań nad standaryzacją użytkowych systemów informatycznych.

Kierunki i zakres standaryzacji informatycznych systemów zarządzania

Standard należy traktować analogicznie, jak pozostałe produkty rynku informatycznego (z wyjątkiem niektórych standardów firmowych opracowanych na użytek wewnętrzny i celowo chronionych przed rozpowszechnieniem). Ogłoszony publicznie standard nie jest chroniony prawami autorskimi; przeciwnie – po to publikuje się standard, aby był on jak najszerzej stosowany [1].

Standard systemu informatycznego może być używany do:

- oceny i wyboru systemów gotowych (oferowanych na rynku), przez potencjalnych użytkowników,
- tworzenia systemów użytkowych,
- atestacji systemów gotowych.

Te trzy kierunki działań wyznaczają trzy całkowicie różne grupy odbiorców standardu (użytkownicy, autorzy systemów i zespoły atestacyjne), których interesy nie zawsze są zbieżne. Dlatego ich przekonanie o potrzebie istnienia standardów systemów informatycznych oraz szerokiej ich promocji bywa w praktyce bardzo różne.

Szczególna rola standardu powoduje, że istotne jest nie tylko poinformowanie odbiorców o istnieniu konkretnego standardu, ale także skłonienie ich do jego używania. Konieczne jest ponadto spowodowanie takiego odzewu ze strony odbiorców, który będzie pełnić funkcję sprzężenia zwrotnego umacniającego dany standard lub zmierzającego do jego likwidacji (jeśli proponowane w standardzie rozwiązania stałyby się nieracjonalne, nienowoczesne itp.).

Najtrudniejszym odbiorcą standardu jest potencjalny użytkownik systemu informatycznego. Wiele lat propagowania indywidualnych systemów informatycznych jako najwłaściwszego podejścia do komputeryzacji, wyrobiło w użytkownikach całkowicie mylne przekonanie o tym, iż systemy powielarne lub standardowe są gorszą kategorią produktów informatycznych. Użytkowanie takich systemów traktują jako ostateczność, a wszystkie wysiłki koncentrują na pozyskaniu autorów (zespołów), którzy podejmą się realizacji indywidualnych systemów, odpowiadających ich specyficznym potrzebom. Tymczasem specyfika ta, to zazwyczaj jedynie nieco inny zakres funkcjonalny systemu, pewien charakterystyczny zakres danych lub określone wymagania dotyczące szczegółowości kontroli danych. Zazwyczaj wszystkie te oczekiwania mogą się znaleźć bez większego trudu w systemie powielarnym, pod warunkiem jednak, iż zostaną one wyartykułowane albo w procesie tworzenia systemu, albo przez zakres odpowiedniego standardu. Tę grupę odbiorców trzeba zatem najpierw przekonać do samej zasadności tworzenia i użytkowania standardów, a dopiero później do wykorzystania konkretnych standardów przy ocenie i wyborze sytemu użytkowego.

Druga grupa odbiorców standardów to autorzy systemów użytkowych (projektanci i programiści). Są to odbiorcy bardzo zróżnicowani, reprezentujący różne środowiska, różny poziom wiedzy i praktyki informatycznej, działający w różnych układach zawodowych i organizacyjnych. Brak tu jakiejkolwiek jednolitej postawy autorów systemów wobec standardów. Jedni z nich są do takich przedsięwzięć całkowicie przekonani, skłonni zarówno włączyć się w prace standaryzacyjne, jak i w nie inwestować, doceniając korzyści, jakie im przyniesie osadzenie tworzonych systemów na akceptowanych standardach. Inni, wręcz przeciwnie, uznają standaryzację za zamach na własną swobodę działania, a standardy traktują jako usztywnienie i odebranie ich produktom czynnika twórczego. Oprócz takich zdecydowanych postaw jest także wielu autorów, którzy w samym fakcie standaryzacji nie widzą niebezpieczeństwa lub naruszenia własnej swobody, pod warunkiem, iż sami poznają i zaakceptują proponowane standardy.

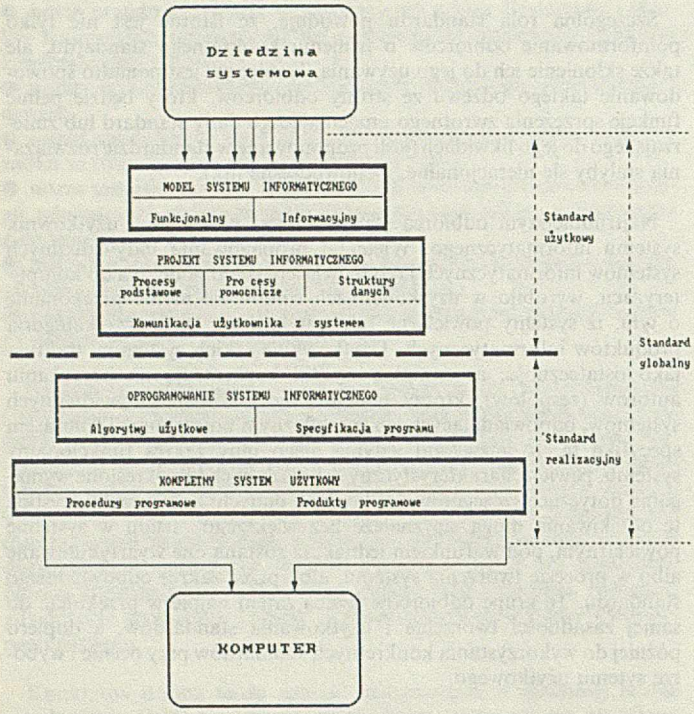
Trzecia grupa – zespoły atestujące – to odbiorca, którego trudno wskazać a priori, bowiem do zespołów takich mogą być powoływani różni specjaliści, zarówno użytkownicy, jak i autorzy, a ich działanie w takim zespole może mieć charakter jednorazowego zadania. Ta grupa odbiorców jest zainteresowana informacją o standardach i dostępem do nich jedynie w toku prac atestacyjnych. Dla zespołu atestacyjnego standard jest narzędziem wykonania powierzonego mu zadania.

Jak więc widać, dla pierwszej i trzeciej grupy odbiorców standard ma znaczenie wyłącznie konfrontacyjne i stanowi wzorzec do porównania i oceny określonego (już istniejącego) produktu. Rola standardu dla twórców systemu jest całkowicie odmienna, bo kreatywna.

Obserwacja **dynamiki** tworzenia systemu informatycznego wskazuje, iż w miarę realizacji kolejnych etapów budowy systemu, są przyjmowane odpowiednie normy, tworzące kolejne, nakładające się na siebie, warstwy standaryzacji (rys. 1).

Analiza dziedziny przedmiotowej oraz potrzeb i wymagań użytkownika umożliwia **modelowanie** przyszłego systemu informatycznego ze szczególnym uwzględnieniem funkcjonalnej i informacyjnej struktury systemu. Na etapie modelowania może być zatem przyjęty standard merytoryczny danej dziedziny przedmiotowej.

Kolejny etap – **projektowanie** systemu informatycznego – obejmuje swoim zakresem dwa obszary przedmiotowe [5]:
- **zasoby informacyjne**, tj. struktury danych na wejściu i wyjściu systemu oraz w bazie danych,
- **procesy przetwarzania**, obejmujące procesy podstawowe i pomocnicze oraz komunikację użytkownika z systemem.



Rys. 1. Warstwy standaryzacji systemu informatycznego

Każdy z wyróżnionych elementów staje się następnie przedmiotem projektowania; w obszarze zasobów informacyjnych należy zaprojektować:
- wejście systemu, tzn. określić zakres i postać informacji wprowadzonych do systemu oraz sposób ich wprowadzania,
- zbiory-bazę danych, tj. ich strukturę logiczną i fizyczną,
- wyjście systemu, tj. zakres i postać informacji (wyników przetwarzania) wyprowadzanych z systemu oraz sposób ich wyprowadzania.

Projektowanie wejścia, zbiorów-bazy danych i wyjścia systemu sprowadza się przede wszystkim do zaprojektowania **struktur danych** dla każdego z wymienionych przedmiotów projektowania.

W obszarze procesów przetwarzania przedmiotem projektowania są:
procesy podstawowe, odpowiadające procesom realizacji użytkowych funkcji merytorycznych systemu,
procesy pomocnicze (wynikające z uwarunkowań przyjętej w systemie technologii przetwarzania danych), których celem jest uzyskanie odpowiedniego poziomu sprawności i skuteczności realizacji procesów podstawowych oraz racjonalności eksploatacji systemu,
komunikacja użytkownika z systemem, dotycząca metod, technik i sposobów wprowadzania i wyprowadzania danych do lub z innego systemu informatycznego.

Już sama specyfikacja przedmiotów projektowania wyraźnie wskazuje, że podczas projektowania tych wszystkich elementów systemu jest możliwe przyjmowanie odpowiednich norm, dotyczących z jednej strony aspektów merytorycznych systemu, a z drugiej także aspektów konstrukcyjnych. Uzyskany w ten sposób standard cząstkowy można nazwać **standardem użytkowym** systemu informatycznego. Dotyczy on bowiem głównie tych elementów, które interesują użytkownika systemu.

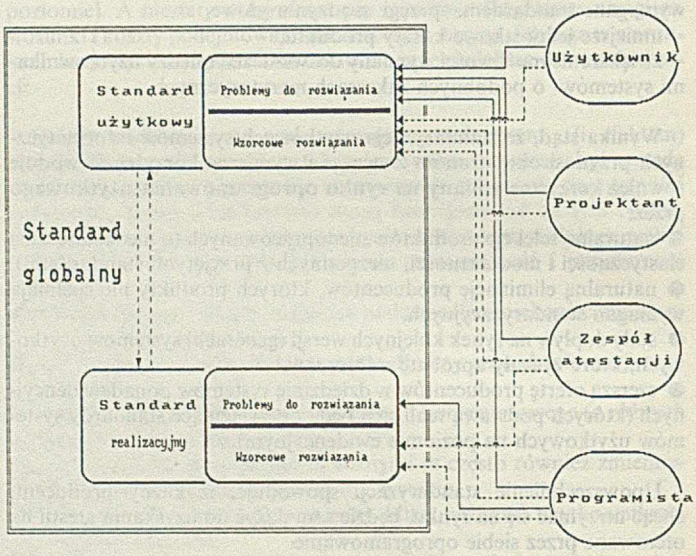
Po zakończeniu prac projektowych rozpoczyna się etap prac realizacyjnych, tj. **oprogramowanie** systemu. Etap ten, utrwalając normy i zalecenia standardu użytkowego, narzuca dodatkowo wymagania standaryzacyjne o charakterze technologicznym. Przedmiotem standaryzacji mogą tu być określone procedury programowe lub produkty programowe. Jest to zatem **standard realizacyjny** systemu informatycznego.

Standard użytkowy łącznie ze standardem realizującym tworzą **standard globalny** systemu.

Należy zwrócić uwagę, że standard realizacyjny – w odróżnieniu od użytkowego – jest w pewnym stopniu niezależny od dziedziny przedmiotowej i wymagań użytkowników, dotyczy bowiem sposobów komputerowej realizacji zadań wchodzących w zakres funkcjonalny systemu. Implikacją tego faktu jest pewna niezależność standardów użytkowych (związanych z określonymi dziedzinami przedmiotowymi) i standardu realizacyjnego, który może być wspólny dla wielu systemów użytkowych.

Każdy ze standardów cząstkowych obejmuje swoim zakresem zarówno listę problemów do rozwiązania, jak i wzorcowe rozwiązania. Oba te elementy stanowią zatem składowe standardu.

Zarówno standard użytkowy, jak i realizacyjny dla określonej klasy systemu informatycznego są tworzone w dwóch zasadniczych etapach. Pierwszy z nich ma na celu specyfikację wszelkich możliwych problemów, których standaryzacja jest pożądana i możliwa. Drugi etap to rozwiązanie tych problemów. W wyniku tego w ramach standardu użytkowego powstają wzorcowe rozwiązania, stanowiące normy standaryzacyjne, dotyczące poszczególnych elementów systemu (procesy podstawowe, struktury danych, procesy pomocnicze i komunikacja użytkownika z systemem) [4]. Wzorcowe rozwiązania w standardzie realizacyjnym stanowią procedury bądź integralne moduły programowe (rys. 2).



Rys. 2. Aplikacja standardu systemu informatycznego

Prześledźmy ten proces na przykładzie projektowania dialogu, który ma istotne znaczenie dla komunikacji użytkownika z systemem. Szczegółowe prace nad dialogiem musi poprzedzać określenie konwencji, w jakiej dialog zostanie zrealizowany (a następnie będzie wykonywany w procesie przetwarzania systemu).

Konwencja dialogu to ustalenie, jakie musi sformułować projektant (lub zespół projektantów), dotyczące zarówno metody strukturalizacji dialogu, jak i sposobu rozwiązywania pewnych, powtarzalnych w systemie sytuacji, a także postaci ekranu, brzmienia komunikatów, sposobu wybierania funkcji itp. Elementami dialogu mogą być:
- formatki (postacie ekranów),
- okna (ang. *window*), tzn. wyróżnione fragmenty ekranów,
- listy wyboru (menu),
- komunikaty,
- funkcje pomocy (ang. *help*).

Formatki to zaprojektowane postacie ekranu, które powinny pojawiać się w wybranych momentach działania systemu. Jako ekran należy rozumieć całą powierzchnię monitora. Parametry techniczne ekranu mogą być różne (z uwagi na pojemność znakową, tzn. liczbę wierszy i znaków w wierszu), a więc określenie co najmniej kategorii sprzętu, na którym system będzie eksploatowany jest koniecznością na tym poziomie prac. Wśród formatek można wyróżnić pewne szczególne grupy, a mianowicie:
– strony tytułowe, które powinny zawierać: nazwę systemu, jego wersję i (lub) generację, nazwę producenta (ewentualnie nazwisko autora), adres do komunikacji z autorami, rok opracowania systemu; czasem na tych stronach są także umieszczane znaki firmowe producenta lub graficzne plansze mające zainteresować użytkownika,
– formatki wejściowe, służące do wprowadzania danych źródłowych;
– formatki wyjściowe, służące do prezentacji na ekranie wyników przetwarzania;
– formatki wyboru funkcji, czyli te ekrany, za pomocą których użytkownik przechodzi z jednego działania w inne;
– formatki informacyjne, czyli ekrany, na których są wyświetlane zazwyczaj tekstowe informacje o systemie, jego funkcjach, ograniczeniach, algorytmach itp. (formatki te są realizacją jednego z poziomów funkcji pomocy);
– formatki „strony końcowej", czyli ekrany kończące pracę systemu.

Wszystkie formatki merytoryczne (poza stroną tytułową i stroną końcową) muszą być jednolicie zaprojetowane i rozplanowane na ekranie. Standaryzacja formatek obejmuje:
1. Rodzaj formatki.
2. Zakres informacji na formatce, w podziale na informacje:
– stałe, stanowiące makietę formatki,
– zmienne, wpisywane przez użytkownika lub komputer.
3. Sposób rozmieszczenia poszczególnych informacji na formatce.
4. Stosowane repertuary znaków i kroje pisma.
5. Sposób wypełnienia pustej powierzchni formatki.

Okno stanowi wyróżniony fragment ekranu. Często bywa ono obwiedzione ramką, która ma wyraźnie oddzielić podstawową powierzchnię formatki od okna. W tym samym celu wykorzystuje się dodatkowe tło („szarość" lub kolor) powierzchni okna. Właściwością okna jest fakt, iż w chwili jego przywołania nakłada się ono na aktualny obraz na ekranie (zakrywając te fragmenty, które się nakładają), a rezygnacja z wyświetlania okna przywraca poprzednią postać ekranu. Takie rozwiązanie pozwala umieszczać w oknach różne informacje pomocnicze, po wykorzystaniu których można powrócić do bieżącego toku pracy. Istotne jest, iż pewne dane lub parametry wybrane w chwili przywołania okna mogą być automatycznie przeniesione do głównej części programu. Kolejne okna można przywoływać wewnątrz okien już wyświetlonych na ekranie, jednak budową takich wielopoziomowych odwołań musi zapewniać przejrzystość i czytelność informacji na ekranie.

Standaryzacja okien może dotyczyć:
1. Zakresu informacji umieszczanych w oknie, w podziale na:
– informacje stałe,
– informacje zmienne.
2. Miejsca wyświetlenia okna na ekranie.
3. Sposobu przywołania i odwołania okna.
4. Sposobu przeglądania w oknie kolejnych „stron".
5. Stosowanych repertuarów znaków i krojów pisma.
6. Znaków użytych jako ramka okna.
7. Sposobu wypełnienia pustej powierzchni okna.
8. Liczby poziomów zagłębienia okien.

Na formatce ekranu lub na oknie można realizować następujące operacje:
- wprowadzać dane lub parametry,
- wybierać funkcje do realizacji,
- prezentować wynik realizacji wybranej funkcji.

Metody wyboru na ekranie funkcji do realizacji (stosowane także do wskazywania poleceń, wyboru parametrów, plików itp.), realizowane głównie przez wskazanie na liście funkcji (menu) może być dokonane przez:
- podświetlenie (inwersję); podświetlenie może być przemieszczane na liście za pomocą klawiszy ze strzałkami, zazwyczaj tylko w pionie lub tylko w poziomie,
- wskazanie numeru funkcji na liście,
- naciśnięcie klawisza funkcyjnego odpowiadającego zazwyczaj numerowi funkcji na liście,
- podanie wybranej (zazwyczaj pierwszej) litery nazwy funkcji,
- wpisanie pełnej lub skróconej nazwy funkcji,
- wskazanie miniaturowego, schematycznego rysunku (piktogramu) kojarzącego się z wybieraną funkcją,
- dowolny wskaźnik znakowy (np. znak „★", „<", „>").

Standaryzacja wyboru funkcji może dotyczyć:
1. Sposobu rozmieszczenia menu na ekranie (w kolumnie lub wierszu).
2. Liczby pozycji menu (mieszczące się na stronie lub dłuższe).
3. Sposobu nazywania (określania) poszczególnych pozycji menu.
4. Sposobu wskazywania i wyboru pozycji do wykonania.
5. Stosowanego repertuaru znaków i krojów pisma.

Wszystkie teksty pochodzące z systemu i pojawiające się na ekranie noszą miano **komunikatów**. Można wśród nich wyróżnić dwie grupy: komunikaty informacyjne i komunikaty błędów. Rola **komunikatów informacyjnych** polega na informowaniu użytkownika o bieżących działaniach systemu oraz wyjaśnianiu i prezentowaniu innych możliwości działania. Bieżące komunikaty informacyjne są istotne zwłaszcza w odniesieniu do procesów długotrwałych (trwających powyżej kilku minut), których realizacja może zaniepokoić użytkownika. Taka sytuacja wymaga wyświetlania informacji pozwalających śledzić postęp przetwarzania.

Komunikaty błędów stanowią element istotny dla prawidłowego działania systemu. Są one sygnałem wprowadzenia niepoprawnych danych lub parametrów, błędnego wybrania funkcji, nieprawidłowych skutków podjętych działań oraz sygnałem próby naruszenia integralności systemu. Komunikaty błędów można podzielić na dwa rodzaje:
– ostrzegawcze, których zadaniem jest tylko wskazanie użytkownikowi systuacji nieprawidłowej, a decyzję o dalszym działaniu podejmuje on sam (na własne ryzyko, może bowiem przewidzieć skutki swych działań); zazwyczaj po komunikacie ostrzegawczym istnieje alternatywa kontynuowania działań lub przejścia do poprawy elementu błędnego (danej, parametru itp.),
– blokujące, które nie tylko informują o powstałym błędzie, ale także wymuszają tryb dalszego działania (uniemożliwiają dokonanie działania niepoprawnego), aż do chwili, gdy cały proces powróci do sytuacji poprawnej.

Brzmienie komunikatów błędów może być bardzo różne, w zależności od tego, na ile dokładnie system diagnozuje domniemaną przyczynę powstania błędów i jakich wskazówek należy udzielić użytkownikowi, aby mógł on następnie podjąć poprawne działanie. Jednocześnie należy zabiegać o pewną unifikację brzmienia komunikatów błędów dotyczących podobnych sytuacji, które występują w różnych miejscach systemu lub też dotyczą różnych obiektów.

Formułowanie komunikatów w języku polskim jest często związane z wieloma trudnościami spowodowanymi specyfiką języka, a w szczególności z:
- problemem braku polskich liter w wielu narzędziach wykorzystywanych do budowy systemów (zniekształcenie wyrazów wskutek braku polskich liter na ekranie powoduje czasem utratę ich sensu, często prowadzi do niejasności, a zazwyczaj ośmiesza autorów systemu w oczach użytkowników),
- częstym posługiwaniem się długimi wyrazami, których „upakowanie" na ekranie bywa kłopotliwe,
- koniecznością upraszczania komunikatów i formułowania ich wprost do użytkownika, bez przyjętych kulturowo zwrotów grzecznościowych, co bywa traktowane przez użytkowników jako zbyt protekcjonalne,
- koniecznością odmian gramatycznych pojęć stosowanych do budowy komunikatów (np. w komunikacie „n lat", jeśli $n = 2$, to komunikat brzmi „2 lata", a gdy $n = 5$, to „5 lat" itp.),
- problemem subiektywizmu znaczeniowego określonych pojęć, które nie zawsze są dostatecznie dobrze rozumiane przez autorów dialogu; często zresztą zbyt dosłowne tłumaczenie analogicznych komunikatów angielskich prowadzi do ich śmiesznego brzmienia w języku polskim.

Standaryzacja komunikatów może obejmować:
1. Rodzaje komunikatów.
2. Budowę komunikatów, tj. wyróżnienie:
– części stałej,

– części zmiennej.
3. Unifikację brzmienia komunikatów (w tym odmiany gramatyczne).
4. Postać wyświetlania komunikatów (normalna, efekty dodatkowe).
5. Stosowany repertuar znaków i krojów pisma.
6. Sposób działania systemu po komunikacie (kontynuacja lub zawieszenie).

Specyficzną rolę w dialogu odgrywają **funkcje pomocy** (*help*), które nie służą bezpośrednio realizacji żadnego zadania użytkowego, lecz są przeznaczone do usprawniania i wspomagania użytkownika w realizacji innych działań. Najprostsza realizacja funkcji pomocy sprowadza się do włączenia do systemu pewnej liczby bloków informacyjnych, wyjaśniających zakres systemu i poszczególnych funkcji, sposób wypełniania formatek, operowanie klawiaturą itp. Jest to zatem skomputeryzowana dokumentacja użytkowa systemu, przeznaczona do czytania za pośrednictwem komputera. Rozwiązanie bardziej złożone, a jednocześnie korzystniejsze dla użytkowników, to pomoc kontekstowa (okolicznościowa), przywoływana w wybranych chwilach działania systemu. Można wyróżnić trzy poziomy pomocy konteksowej, stosowanej przy:
- wyborze elementu listy funkcji (menu główne i dowolne podmenu; jest przywołany komunikat informacyjny wyjaśniający zakres funkcji, jej parametry, sposób realizacji itp.),
- wypełnianiu konkretnego pola (realizowane np. przez przywołanie odpowiednich tablic kodowych),
- wypełnianiu formatki dokumentu (np. możliwość skopiowania wypełnionej wcześniej analogicznej formatki i dokonanie na niej stosownych korekt).

Standaryzacja funkcji pomocy może obejmować:
1. Wymagane w systemie poziomy pomocy.
2. Zakres i szczegółowość wyjaśnień,
3. Sposób prezentacji pomocy (pełnoekranowy, wyłącznie w oknie).
4. Sposób wyjścia z funkcji pomocy (odwołanie).
5. Stosowany repertuar znaków i krojów pisma.

Skutkiem braku dostatecznie precyzyjnych ustaleń projektowych są różnorodne błędy, wśród których jako najpopularniejsze można wskazać:
- dialog „na intuicję" (autora dialogu); jest to dialog mało konsekwentny i źle udokumentowany na ekranie; w założeniu autora użytkownik powinien przewidzieć kolejność działań, znać na pamięć kody, nazwy itp.;
- dialog „zdublowany"; jest to takie rozwiązanie, w którym wymaga się potwierdzenia każdego działania (np. jest polecenie „podaj numer wybranej funkcji", a następnie pytanie „czy na pewno dobrze wybrałeś?"), niezależnie od złożoności i ewentualnych skutków podejmowanych działań,
- zła sygnalizacja błędnych danych (np. brak jakiegokolwiek dodatkowego zwrócenia uwagi użytkownika na fakt pojawienia się błędu, poza rutynowym komunikatem „błędne dane" u dołu ekranu),
- pozostawienie w dialogu „ślepych zaułków" tzn. takich rozgałęzień programu, z których praktycznie nie ma wyjścia; najczęściej sytuacja taka zdarza się wówczas, gdy system wymusza wprowadzenie poprawnych danych (często dotyczy to parametrów), a użytkownik ich nie zna i nie jest w stanie odgadnąć (żadne z przypadkowo wprowadzanych danych nie są prawdziwe); taka sytuacja wymaga praktycznie interwencyjnego przerwania pracy całego programu.

Stwierdzenie poprawości dialogu jest pracochłonne i wymaga bardzo starannej weryfikacji. Zarówno oprogramowanie dialogu, jak i jego weryfikacja są dużo łatwiejsze, gdy dysponujemy szczegółowymi wytycznymi (standardami). Znaczenie standaryzacji dialogu staje się jeszcze istotniejsze, gdy realizowany system obejmuje więcej niż jedną dziedzinę problemową, a jednocześnie trafia do rąk tych samych użytkowniów [3].

* * *

Społeczny rachunek kosztów i efektów standaryzacji jest bardzo trudny, zwłaszcza że w obecnej chwili brak jest skutecznych metod i technik obliczania efektywności komputeryzacji nawet pojedynczych przedsięwzięć. Jednak nie ulega wątpliwości, iż standaryzacja jest jedną z metod poprawiania jakości poszczególnych produktów (systemów) oraz powiększania asortymentu produktów na rynku. Z tego więc względu należy intensyfikować prace standaryzacyjne oraz promować takie działania, a jednocześnie poszukiwać źródeł ich finansowania.

LITERATURA

[1] Analiza celowości, możliwości i ogólnych zasad standaryzacji użytkowych systemów informatycznych. B. Łukasik-Makowska (oprac.). Raport badawczy w ramach tematu: Standaryzacja Atestacja Użytkowych Systemów Informatycznych. AE, Wrocław, 1989 (maszynopis)

[2] Łukasik-Makowska B., Skwarnik M.: Standaryzacja procesów pomocniczych. INFOGRYF'90, TNOiK i Uniwersytet Szczeciński, Międzyzdroje, 1990

[3] Łukasik-Makowska B.: Powielarne systemy informatyczne. Standaryzacja. Weryfikacja. Wdrożenie PWE, Warszawa (w druku)

[4] Metodyka standaryzacji i atestacji użytkowych systemów informatycznych. B. Łukasik-Makowska (oprac.). Raport badawczy w ramach tematu: Standaryzacja i Atestancja Użytkowych Systemów Informatycznych. AE, Wrocław 1990

[5] Projektowanie systemów informatycznych. E. Niedzielska, M. Skwarnik (oprac). Wyd. 3 zmienione, PWE Warszawa (w druku)

[6] Zalewski J.: Zalecenia normalizacyjne stosowane w okresie istnienia oprogramowania. Przewodnik problemowy. Materiały szkoleniowe nr 1, 1989, Oddział Górnośląski PTI, Katowice 1989.

Dr inż. Barbara ŁUKASIK-MAKOWSKA ukończyła w 1970 r. Wydział Elektroniki Politechniki Wrocławskiej. W tym samym roku rozpoczęła Studia doktoranckie z informatyki na Akademii Ekonomicznej. Pracę doktorską obroniła w 1975 roku, a od 1976 roku pracuje jako adiunkt w Instytucie Informatyki Akademii Ekonomicznej. Głównym obszarem jej zainteresowań jest problematyka tworzenia systemów informatycznych wspomagających zarządzanie oraz systemów powielarnych a także standaryzacja systemów informatycznych. Ma na swoim koncie znaczną liczbę publikacji, w tym współautorstwo 14 książek i skryptów (z czego 4. jest redaktorem), autorstwo ponad 60 artykułów i referatów konferencyjnych

Zmiany na rynku komputerowym

W roku 1984 Stany Zjednoczone miały 78% udziału w światowym rynku komputerowym, podczas gdy Japonia tylko 11%. W ciągu pięciu lat udział USA zmniejszył się o 19 punktów – do 59%, natomiast udział Japonii zwiększył się o 14 punktów – do 25% w roku 1989. Największą zmianę zaobserwowano w latach 1986–1987. W roku 1993 Japonia ma nadzieję dogonić i wyprzedzić Stany Zjednoczone, jako lider rynku informatycznego. W dziedzinie elementów półprzewodnikowych dokonało się to już w 1986 r.

Poniższe zestawienie pokazuje pozycję rynkową 10 największych producentów sprzętu komputerowego w 1984 i 1989 roku.

1984	1989
1. IBM	1. IBM
2. DEC	2. DEC
3. Borroughs	3. NEC
4. Sperry	4. Fujitsu
5. Fujitsu	5. Unisys
6. Control Data	6. Hitachi
7. NCR	.7 HP
8. HP	8. Bull
9. NEC	9. Siemens
10. Siemens	10. Olivetti

Tylko dwie główne firmy amerykańskie IBM i DEC utrzymały swoje czołowe pozycje, oraz Hewlett Packard, która nawet awansowała z 8. na 7. miejsce. Pozostałe cztery firmy USA znikły z tabeli, a na ich miejsce pozostała tylko nowo utworzona Unisys (Sperry Univac + Burroughs). Natomiast japońska firma NEC awansowała o 6 pozycji (z 9. na 3.), a Fujitsu przesunęła się o jedno miejsce w górę (z 5. na 4.). Pojawia się też firma Hitachi, zajmująca w 1989 r. 6. miejsce. Wśród pierwszej dziesiątki znalazły się dwie nowe firmy europejskie – francuska Bull (na 8. pozycji) i włoska Olivetti (na ostatniej), zaś niemiecki Siemens przesunął się z 10. na 9.

Tak więc po pięciu latach z 7. firm amerykańskich pozostało tylko 4, natomiast liczba firm europejskich potroiła się, a japońskich wzrosła o 50%. **(J.R.)**

MEDCOM

CHCESZ ZABEZPIECZYĆ BAZY DANYCH PRZED UTRATĄ INFORMACJI?

CHCESZ ZAPEWNIĆ PRAWIDŁOWĄ PRACĘ

KOMPUTERA * CENTRALI TELEFONICZNEJ * TELEFAKSÓW

W WARUNKACH ZANIKU NAPIĘCIA ZASILAJĄCEGO

kup ZASILACZE BEZPRZERWOWE

naszej produkcji
UPS – 140 (abonenckie centrale telefoniczne i telefaksy)
UPS – 450
UPS – 850

produkcji BEST POWER TECHNOLOGY
ON – LINE: 500 VA – 18 kVA

produkcji
AMERICAN POWER CONVERSION:

AP 400I
AP 600I
AP 900I
AP 1250I
AP 2000I
AP 800I
AP 1200I

Sprzedajemy również

* AKUMULATORY BEZOBSŁUGOWE, HERMETYCZNE FIRMY PANASONIC do zasilania urządzeń alarmowych, jedyne dopuszczone do używania w pomieszczeniach zamkniętych,
* GENERATORY PRĄDOTWÓRCZE FIRMY ENDRESS O MOCY OD 1 kVA DO 350 kVA

Twoje problemy w dziedzinie zasilania awaryjnego rozwiążą specjaliści z firmy MEDCOM.

BIURO
Al. Ujazdowskie 26/39
00-478 Warszawa
tel., faks (02) 628-93-57, teleks 817060

Oddział w Gliwicach
ul. Chorzowska 50, 44-100 Gliwice
tel. 31-90-45 w. 139, 140
teleks 036215 iptif pl

0/20/90

Programowanie w języku awk (1)

W 1977 roku trójka znanych informatyków amerykańskich: Alfred V. Aho, Brian W. Kernighan i Peter J. Weinberger zaprojektowała i zaimplementowała język **awk** [1]. W zamyśle autorów miał on służyć jako narzędzie do testowania programów **grep** i **sed** wchodzących w skład programów użytkowych systemu UNIX i wykorzystujących wyrażenia regularne do wyszukiwania tekstów. W krótkim czasie **awk** okazał się narzędziem na tyle atrakcyjnym, że zaczęto go stosować w prostych translatorach czy prostych bazach danych. Skłoniło to autorów do opracowania w 1985 r. rozszerzonej wersji języka.

Awk zaczął bardzo szybko zdobywać popularność – głównym atutem była duża zwartość programów napisanych w tym języku. Dodatkową atrakcją było to, że **awk**, a ściślej mówiąc, jego interpretator, wszedł w skład standardu Unix System V. Natychmiast w literaturze dotyczącej systemu Unix zaczęły masowo pojawiać się skrypty shella zawierające odwołania do **awk**. Dzisiaj można zaryzykować twierdzenie, że w każdym szanującym się czasopiśmie poruszającym problematykę systemu UNIX znajdzie się przynajmniej jedno odwołanie do **awk**.

Celem niniejszej publikacji jest pokazanie, na przykładach kilku wybranych problemów, tych cech języka **awk**, które stanowią o jego sile i atrakcyjności. Autor zakłada, że Czytelnicy znają już podstawy języka **awk** chociażby z opracowania [3] publikowanego na łamach „Informatyki". Pełny opis języka można znaleźć w bardzo dobrym, autorskim podręczniku [2], chociaż nie wszystkie implementacje **awk** są w całości z nim zgodne.

Wyszukiwanie informacji i sporządzanie raportów

Wyobraźmy sobie, że dysponując systemem Unix chcemy jak najmniejszym wysiłkiem prowadzić dokumentację naszych finansów, nie mając dostępu do specjalistycznego oprogramowania, typu arkusza kalkulacyjnego czy bazy danych. Za pomocą edytora **vi** możemy utworzyć plik **dw** zawierający informacje o naszych dochodach i wydatkach:

```
dochod  honorarium za wyklad
data    91/05/15
kwota   60000

wydatek zywnosc
data    91/05/22
kwota   182000

kwota   500000
dochod  pensja
data    91/05/28
uwagi   wyjasnic dlaczego tak malo

wydatek teatr
kwota   60000
data    91/05/26
```

Format pliku jest jednym z wielu możliwych: wybraliśmy właśnie taki, bo jest on dosyć czytelny. Kolejność pól w każdej informacji o dochodzie lub wydatku jest dowolna. Nie chcemy być przywiązani do stałego formatu informacji; ustalamy jedynie, że powinno wystąpić pole **kwota** i albo pole **dochód**, albo pole **wydatek.** Pole **data** i **uwagi** są opcjonalne. Puste wiersze są separatorami rekordów.

Mając tak sporządzony plik stawiamy sobie zadanie opracowania programu, który sumowałby nam dochody i wydatki oraz podawał saldo. Dodatkowo chcemy, żeby podano nam informację o każdym wydatku czy dochodzie, co do którego mamy **uwagi**.

W tym celu możemy napisać program w języku C – działamy przecież pod UNIX-em. Wydaje się jednak, że najmniej kosztowne będzie utworzenie pliku **bilans** zawierającego następujący program w języku **awk**:

```
/^wydatek/   { wyd = 1 ; tresc(t) ; next }
/^dochod/    { doch = 1 ; tresc(t) ; next }
/^kwota/     { ile = $2 ; next }
/^data/              { dat = $2 ; next }
/^uwagi/     { tresc(uw) ; next }
/^$/                 { bilans() }

END          { bilans()
               printf("\ndochody=%d  wydatki=%d  saldo=%d\n",
                    dochody,wydatki,dochody-wydatki)
             }

function bilans() {
    if( wyd )
            wydatki += ile
    else if( doch )
            dochody += ile
    if( uw != "" )
            print dat, ",", ile, ",", t, ",", uw
    t = dat = uw = ""
    wyd = doch = ile = 0
}

function tresc(tekst) {
    i = 1
    while( ++i<=NF )
            tekst = tekst " " $i
}
```

Po wywołaniu

awk -f bilans dw

na standardowym wyjściu powinien pojawić się tekst

91/05/28, 500000, pensja, wyjaścić dlaczego tak mało
dochody = 560000 wydatki = 242000 saldo = 318000

Kończąc tę część artykułu zwróćmy uwagę na kilka charakterystycznych cech programu. W przypadku pliku **dw** mamy do czynienia z rekordami złożonymi z kilku wierszy. Wywołanie zdefiniowanej przez nas funkcji **bilans** następuje zawsze po wystąpieniu wiersza pustego lub końca pliku. Sposób bilansowania zależy od wartości flag **doch** i **wyd**. Funkcja **tresc**, również zdefiniowana przez programistę, eliminuje z wiersza pliku **dw** pierwsze słowo formalne. Przedstawione rozwiązanie problemu nie pretenduje do miana rozwiązania najlepszego. Autor będzie wdzięczny Czytelnikom za nadsyłanie ciekawszych rozwiązań problemu.

(cdn)

WŁADYSŁAW MAJERSKI

LITERATURA

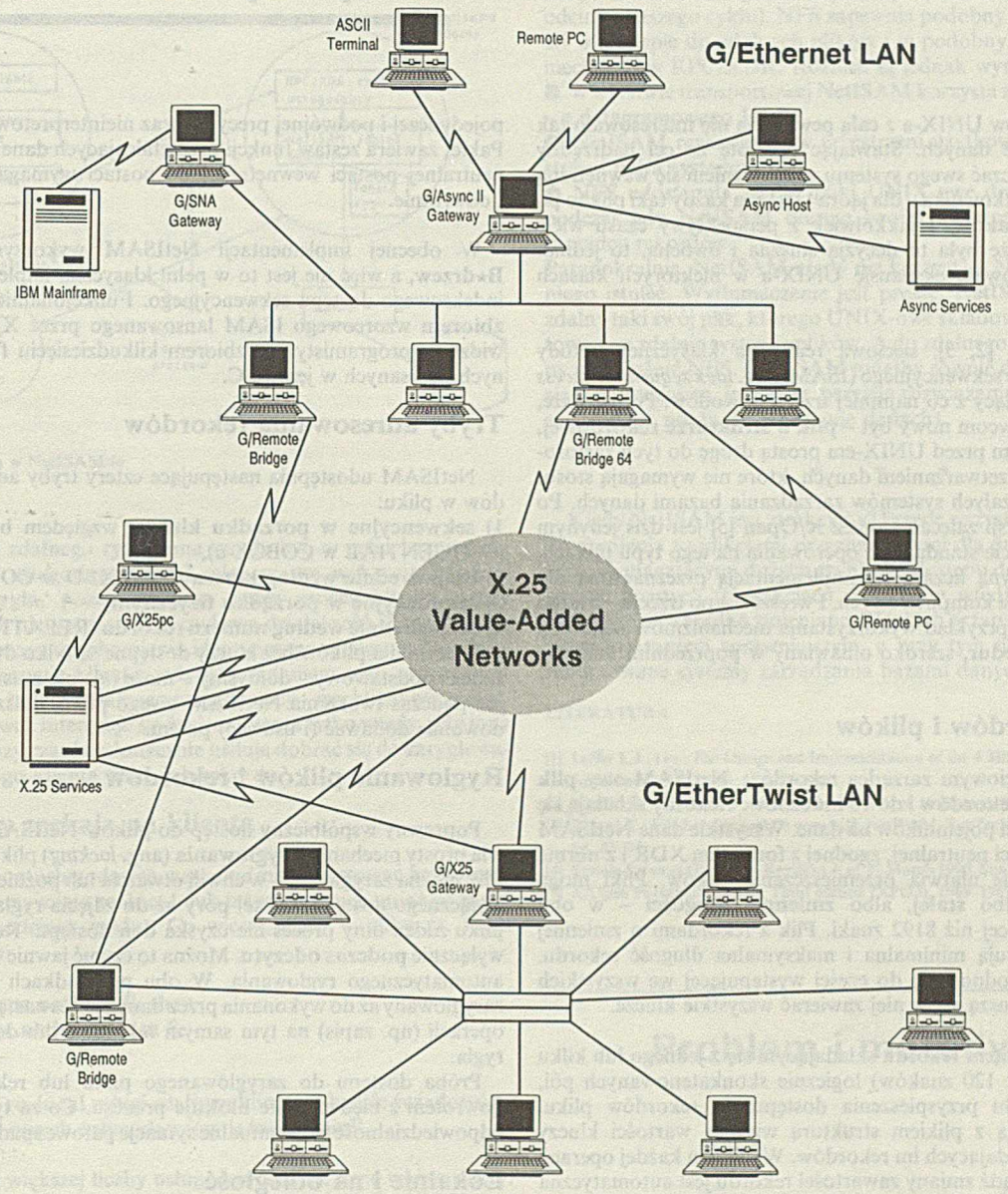
[1] Aho A.V., Kernighan B.W., Weinberger P.J., AWK – a pattern scanning and processing language, Software – Practice and Experience, April 1979

[2] Aho A.V., Kernighan, B.W., Weinberger P.J., The AWK Programming Language, Addison-Wesley, 1988

[3] Płoski Z.: Język awk – przetwarzanie wzorców, Informatyka nr 11–12, 1987

Produkty sieciowe LAN i WAN

Gateway communications, inc.



BEZKONKURENCYJNE PRODUKTY
SIECI KOMPUTEROWYCH NAGRADZANE PRZEZ:

PC MAGAZINE, LAN MAGAZINE, INFO WORLD

OFERUJE AUTORYZOWANY DYSTRYBUTOR:

MIKROB

P.W.P.T. MIKROB SP. Z O.O.

20-346 Lublin, ul. Długa 5

Tel. (0-81) 422-28, faks 415-43, teleks 643776 MIKRO PL

Na życzenie wysyłamy katalog produktów. Atrakcyjny program współpracy dla dealer'ów.

O/7/91

/jak unix z unix-em

Studium klienta i usługodawcy, czyli o Sun-owej koncepcji sieci otwartych (2)

Na przykład NetISAM

Pierwszych twórców UNIX-a z całą pewnością nie interesowało tak zwane przetwarzanie danych. Stawiając prostotę za cel nadrzędny postanowili nie obarczać swego systemu zajmowaniem się wewnętrzną strukturą plików użytkownika – dla jądra UNIX-a każdy taki plik to po prostu **strumień znaków**. I jakkolwiek z perspektywy czasu wiele przemawia za tym, że była to decyzja słuszna i owocna, to jednak niewątpliwie zahamowała ekspansję UNIX-a w niektórych klasach zastosowań.

Pakiet **NetISAM** [2, 3], sieciowa realizacja klasycznej metody indeksowego dostępu sekwencyjnego (ISAM, ang. *idex sequential access method*) jest interesujący z co najmniej trzech powodów. Po pierwsze, udostępnia UNIX-owcom nowy byt – **plik o strukturze rekordowej**, otwierając tym samym przed UNIX-em prostą drogę do tych zastosowań związanych z przetwarzaniem danych, które nie wymagają stosowania drogich i ociężałych systemów zarządzania bazami danych. Po drugie, ISAM w wersji zalecanej przez **X/Open** [5] jest dziś jedynym w UNIX-owym świecie standardem operowania na tego typu plikach, zaś NetISAM – jedyną liczącą się implementacją przeznaczoną dla heterogenicznych sieci komputerowych. I wreszcie – po trzecie – trudno o bardziej klarowny przykład wykorzystania mechanizmów **zdalnego wywoływania procedur**, szeroko omawiany w poprzednim numerze *INFORMATYKI*.

Struktura rekordów i plików

NetISAM jest sieciowym zarządcą rekordów. NetISAM-owy **plik** składa się ze zbioru **rekordów** i do 16 **indeksów**. Rekordy składają się z **pól** – elementarnych pojemników na dane. Wszystkie dane NetISAM przechowuje w postaci neutralnej, zgodnej z formatem XDR i z normą IEEE, co znakomicie ułatwia przemieszczenie plików. Pliki mogą zawierać rekordy albo **stałej**, albo **zmiennej długości** – w obu przypadkach nie więcej niż 8192 znaki. Plik z rekordami o zmiennej długości charakteryzują minimalna i maksymalna długość rekordu. Długość minimalna odnosi się do części występującej we wszystkich rekordach pliku – muszą się w niej zawierać wszystkie klucze.

Klucz jest fragmentem rekordu składającym się z jednego lub kilku (do 8, nie więcej niż 120 znaków) logicznie skonkatenowanych pól, wyróżnionych w celu przyspieszenia dostępu do rekordów pliku. **Indeks** jest związaną z plikiem strukturą wiążącą wartości kluczy z numerami odpowiadających im rekordów. Wynikiem każdej operacji dodania, usunięcia bądź zmiany zawartości rekordu jest automatyczna aktualizacja wszystkich indeksów.

Swoje pliki NetISAM realizuje po prostu jako **jeden, dwa lub trzy pliki UNIX-owe**, w zależności od tego, czy rekordy zawierają klucze i są stałej, czy zmiennej długości. Do przeniesienia NetISAM-owego pliku wystarcza więc *mv* lub *cp*, należy tylko pamiętać, by przenieść wszystkie odpowiadające mu pliki UNIX-owe.

Właścicielem pliku NetISAM-owego – właścicielem odpowiednich plików UNIX-owych – jest jego twórca, a **prawa dostępu** do tego pliku są równoważne prawom dostępu do plików UNIX-owych i wynikają z bieżącej wartości *umask*. Pakiet nie wprowadza żadnych dodatkowych metod autoryzacji dostępu.

Pola NetISAM-owych rekordów mogą przechowywać dane znakowe, krótkie i długie liczby całkowite, liczby zmiennoprzecinkowe pojedynczej i podwójnej precyzji, oraz nieinterpretowane ciągi bajtów. Pakiet zawiera zestaw funkcji przesztalcających dane z NetISAM-owej neutralnej postaci wewnętrznej do postaci wymaganej w programie i odwrotnie.

W obecnej implementacji NetISAM wykorzystuje mechanizm **B★drzew**, a więc nie jest to w pełni klasyczna implementacja metody indeksowego dostępu sekwencyjnego. Funkcjonalnie pakiet jest **nadzbiorem** wzorcowego ISAM lansowanego przez X/Open. Z punktu widzenia programisty jest zbiorem kilkudziesięciu funkcji bibliotecznych napisanych w języku C.

Tryby adresowania rekordów

NetISAM udostępnia następujące cztery tryby adresowania rekordów w pliku:
1) **sekwencyjne w porządku klucza**, względem bieżącego rekordu (SEQUENTIAL w COBOL-u),
2) **bezpośrednie według klucza** (INDEXED w COBOL-u),
3) **sekwencyjne w porządku fizycznym,**
4) **bezpośrednie według numeru rekordu** (RELATIVE w COBOL-u),
przy czym dla plików bez kluczy dostępne są tylko dwa ostatnie tryby. Indeks podstawowy – domyślny – może (ale nie musi) zostać założony już podczas tworzenia NetISAM-owego pliku, dalsze indeksy można dowolnie dodawać (i usuwać) potem.

Ryglowanie plików i rekordów

Poprawny współbieżny dostęp do plików NetISAM-owych umożliwia prosty mechanizm **ryglowania** (ang. *locking*) plików lub rekordów. Plik można zaryglować – w chwili otwarcia lub później – jawnie żądając wyłącznego dostępu. Od tej pory aż do zdjęcia rygla lub zamknięcia pliku żaden inny proces nie uzyska doń dostępu. Rekordy rygluje się wyłącznie **podczas odczytu**. Można to czynić jawnie albo z góry żądać automatycznego ryglowania. W obu przypadkach rekord pozostaje zaryglowany aż do wykonania przez dany proces następnej nieryglującej operacji (np. zapis) na tym samym rekordzie lub do jawnego zdjęcia rygla.

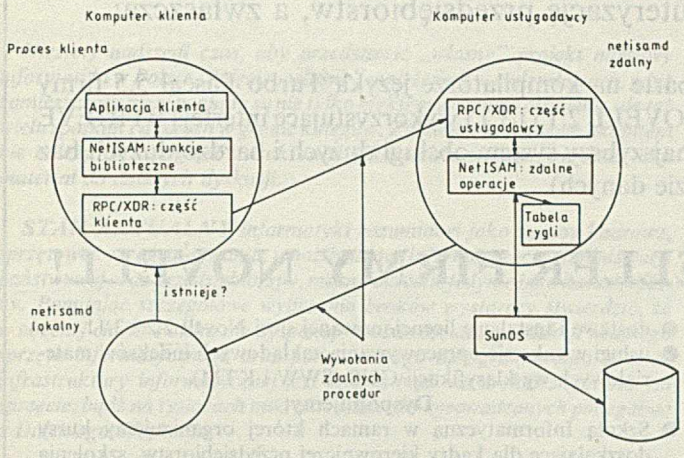
Próba dostępu do zaryglowanego pliku lub rekordu kończy się powrotem z błędem i **nie blokuje** procesu. Co za tym idzie, wszelka odpowiedzialność za ewentualne sytuacje patowe spada na programistę.

Lokalnie i na odległość

NetISAM ma swoje silne strony tak w warunkach pracy lokalnej, jak i w sieciach. Wykorzystywany lokalnie czyni użytek z możliwości **odwzorowywania plików w pamięć**, udostępnianej przez najnowsze wersje SunOS [4]. Unikając w dostępie do pliku pośrednictwa funkcji systemowych *read*() i *write* () uzyskuje się na szybkości dwojako: znikają narzuty na wywołanie funkcji systemowych i na zbędne kopiowanie danych – dane czyta się z lub pisze się do buforów jądra. Dzięki temu można osiągać szybkości przetwarzania kilkuset rzędów operacji NetISAM-owych na sekundę przy działających jednocześnie kilkudziesięciu procesach klientów.

Jakkolwiek z punktu widzenia programisty działanie na NetISAM-owych plikach zdalnych niczym nie różni się od działania na plikach lokalnych (korzysta się z tych samych funkcji bibliotecznych), to sama realizacja większości operacji jest w obu przypadkach zupełnie inna.

O ile wszelkie akcje na plikach lokalnych przeprowadza sam proces klienta, o tyle w przypadku plików zdalnych operacje faktycznie realizuje proces zdalnego usługodawcy – **netisamd**, z którym klient komunikuje się korzystając z **RPC/XDR** – Sun-owego mechanizmu wywoływania zdalnych procedur (rys.). Ponieważ niektóre NetISAM-owe operacje – na przykład ryglowanie – są z natury stanowe, w warstwie komunikacyjnej NetISAM z konieczności korzysta z protokołu **TCP** (por. poprzedni odcinek tego cyklu)[1].



Klient i usługodawca w NetISAM-ie

Aby realizacja zdalnego ryglowana mogła być w pełni bezpieczna, proces zdalnego usługodawcy musi mieć szansę wykrycia każdego **osieroconego rygla**, pozostawionego przez proces klienta, który przedwcześnie lub nietypowo zakończył swą działalność. Z tego powodu NetISAM wymaga uruchomienia identycznego procesu usługodawcy również w maszynie klienta; jedynym zadaniem tego procesu jest sprawdzanie – na żądanie zdalnego usługodawcy – czy klient wciąż żyje. Zdalny usługodawca interesuje się życiem klienta tylko wtedy, gdy inny klient przez dłuższy czas bezskutecznie usiłuje dobrać się do zaryglowanego przez naszego klienta pliku lub rekordu.

Usługodawcy czekają na klienta

Jak każdy demon-usługodawca, *netisamd* może startować bezpośrednio podczas wykonywania **rc. local** lub też może uruchamiać go **inetd** (por. poprzedni odcinek cyklu). Co więcej, jeżeli w pliku **rc. local** znajdzie się fraza:

```
if [– f/user/etc/rpc.netisamd]; then
    /user/etc/rpc.netisamd – m 8;
fi
```

to po wykonaniu **rc. local** usługi zdalnym klientom będzie świadczyć aż ośmiu NetISAM-owych usługodawców jednocześnie[2].

Uruchamianie większej liczby usługodawców ma dwie ważne zalety. Po pierwsze, połączenie z każdym klientem angażuje jeden UNIX-owy deskryptor pliku usługodawcy, a liczba deskryptorów przypadających na proces jest ograniczona. Więcej procesów usługodawców to więcej jednoczesnych połączeń. Po drugie, umożliwia to sprawną współbieżną realizację więcej niż jednego zdalnego żądania na raz.

Operowanie na zdalnych plikach wcale nie musi być wolniejsze niż w przypadkach plików lokalnych. Przeciwnie, często bywa szybsze – zwłaszcza gdy maszyna usługodawcy to potężny komputer, maszyna klienta ma mało pamięci i znacznie wolniejsze dyski, zaś aplikacja wymaga częstych wstawień do plików z wieloma indeksami.

NetISAM a NFS

NetISAM świadczy usługi analogiczne do tych, które oferuje Sun-owy Network File System (zajmiemy się nim dokładniej w następnym odcinku naszego cyklu). NFS zapewnia podobny poziom przeroczystości w dostępie do zdalnych plików i w podobny sposób wykorzystuje mechanizmy RPC/XDR. Różnice są jednak wyraźne:

- w warstwie transportowej NetISAM korzysta z protokołu TCP, NFS – z datagramowego UDP,
- NetISAM realizuje się w przestrzeni użytkownika, NFS – w przestrzeni jądra systemu,
- NFS udostępnia zdalne pliki UNIX-owe dowolnym programom, podczas gdy NetISAM oferuje swoim klientom funkcje sieciowego zarządcy rekordów.

Paradoksalnie, NetISAM nigdy nie korzysta z NFS, lecz nie może bez niego istnieć. Wytłumaczenie jest proste: NetISAM rozpoznaje jako zdalny taki swój plik, którego UNIX-owe składowe lokują się w zamontowanym zdalnie systemie plików. A do zdalnego montowania niezbędny jest właśnie NFS. (NetISAM mógłby równie dobrze nie interesować się montowaniem, jednakże przyjęte rozwiązanie niewątpliwie ułatwia zachowanie poufności i ochronę danych).

* * *

Kiedy NetISAM jest atrakcyjny? Przede wszystkim wtedy, gdy mamy do czynienia z aplikcjami przeznaczonymi dla dużej liczby użytkowników, wymagającymi dużych szybkości dostępu do danych, lecz o stosunkowo prostych transakcjach. Zwłaszcza wtedy, gdy dane są często odczytywane i rzadko zmieniane. W takich przypadkach zapewne okaże się rozwiązaniem sprawniejszym, a przy tym na pewno tańszym niż renomowane systemy zarządzania bazami danych.

LITERATURA

[1] Leffer L.J., i in.: The Design and Implementation of the 4.3BSD UNIX Operating System. Addison-Wesley, 1989, pp. 285–286
[2] NetISAM Programmer's Guide. Revision A of 11 August 1989. Sun Microsystems. Inc.
[3] Rogers T.: File handling made easy with NetISAM. SunTech Journal. Vol. 3, No. 2, 1990, pp. 85–94
[4] System Services Overview. Revision A of 27 March 1990. Sun Microsystems, Inc., pp. 3–14
[5] X/Open Portability Guide. Third edition (XPG3), 1988–1989. Vol. 5: Data Management.

JACEK SURMA

[1] UNIX jest zastrzeżonym znakiem towarowym AT & T. Co nie zmienia faktu, że NetISAM nie do końca polega na połączeniu wirtualnym realizowanym przez TCP/IP. Na przykład, klient nigdy nie dowie się o zerwaniu połączenia, o ile zostanie ono nawiązane ponownie odpowiednio szybko, nawet jeżeli maszyna usługodawcy została w tym czasie restartowana.

[2] Procesy usługodawców dzielą wspólne gniazdko (ang. *socket*) oczekując na zgłoszenia klientów. Kiedy nowy zdalny klient żąda otwarcia swego pierwszego NetISAM-owego pliku, próbuje połączyć się (funkcja *connect*(), [1]) poprzez to właśnie gniazdko. Prośbę połączenia spełnia (funkcja *accept*()) pierwszy wolny w tej chwili usługodawca, uzyskując przy tym przydział nowego gniazdka specjalnie do komunikacji z tym właśnie klintem. Połączenie zostaje nawiązane i od tej pory wszystkie żądania naszego klienta realizuje ten sam usługodawca.

Problem i metody kompresji danych

dokończenie ze s. 13

się ujmowanie połowy symetrycznego widma sygnału akustycznego, przesyłanego falami radiowymi przy modulacji amplitudy (zmniejszenie zakresu widma częstotliwościowego zajmowanego przez pojedynczą rozgłośnię), zastosowanie kodowania RLL w dyskach stałych w stosunku do kodowania MFM, zastosowanie logarytmicznej charakterystyki przetwarzania przetworników AC i CA informacji akustycznej, zapisywanej na dyskach optycznych (ang. *compact disc*).

LITERATURA

[1] Apiki S.: Lossless Data Compression. Byte, March 1991
[2] Chojcan J., Rutkowski J.: Zbiór zadań z teorii informacji i kodowania. Skrypt uczelniany Politechniki Śląskiej nr 1240, Gliwice 1986
[3] Reghbati H.K.: An Overview of Data Compression Techniques. Computer, April 1981
[4] Skaba W.: Współczesne metody kompresji danych: teoria, algorytmy i zastosowania. Informatyka nr 1, 1991
[5] Stożek J.: Pakery na PC – test. Enter, kwiecień 1991.

POZNAJ SIŁĘ NASZYCH SYSTEMÓW INFORMATYCZNYCH!

JUNISOFTEX Sp. z o.o.
44-100 Gliwice ul. Konstytucji 11
Tel. 31-75-10, 31-90-81 do 88 wewn. 250, 272, telefaks: 31-75-10, teleks 036233

Już od 3 lat oferuje kompleksową komputeryzację przedsiębiorstw, a zwłaszcza:

- profesjonalne nowoczesne systemy informatyczne oparte na kompilatorze języka Turbo Pascal 5.5 firmy BORLAND (typu multiusers), pracujące w sieciach NOVELL 2.15 i 3.1 i wykorzystujące interfejs BTRIEVE amerykańskiej firmy SoftCraft – najefektywniejszy i najszybszy system obsługi dużych i bardzo dużych baz danych (do 4 miliardów rekordów w pojedynczej bazie danych).

AUTORYZOWANY RESELLER FIRMY NOVELL!

Polecamy szczególnie systemy zintegrowane

- wielodostępny system finansowo-księgowy wg trójstopniowej analityki w podstawowym układzie: xxx-xxxx-xxxx-xxxxxx z możliwością graficznej prezentacji wyników,
- wielodostępny system gospodarki materiałowej wykorzystujący w strukturze indeksowej SWW i KTM z automatycznym wyliczaniem cyfry kontrolnej (długość indeksu materiałowego 13 + 3 cyfry),
- wielodostępny system gospodarki wyrobami gotowymi wykorzystujący w strukturze indeksowej SWW i KTM,
- wielodostępny system gospodarki przedmiotami nietrwałymi wykorzystujący w strukturze indeksowej SWW i KTM,
- wielodostępny system środków trwałych wykorzystujący klasyfikację środków trwałych GUS,
- wielodostępny system fakturowania sprzedaży,
- wielodostępny system kadrowo-płacowy.

Wszystkie oferowane systemy są zintegrowane ze sobą zgodnie z zasadami rachunkowości księgowej przedstawionymi w miesięczniku RACHUNKOWOŚĆ nr 9, 1990.

Oferujemy również:

- sprzęt komputerowy (sprawdzony w eksploatacji) firm:
  - ALR (Advanced Logic Research) i Storage Dimensions z USA,
  - WEARNES TECHNOLOGY, i inne z Singapuru,
- dostawę i instalację licencjonowanej sieci Novell 2.2 i 3.11,
- usługi w zakresie opracowywania zakładowych indeksów materiałowych wg klasyfikacji GUS (SWW i KTM).

Dysponujemy:

- Szkołą Informatyczną w ramach której organizujemy kursy doszkalające dla kadry kierowniczej przedsiębiorstw, szkolenia w zakresie obsługi naszych systemów aplikacyjnych,
- laboratorium systemu sieciowego NOVELL.

Organizujemy:

- przy współpracy ze Szkołą Komputerową „IMPULS" kursy: DOS, Word Perfect, Pascal, Podstawy obsługi systemu operacyjnego NetWare firmy NOVELL; szkolenia prowadzone są przez doświadczonych wykładowców Politechniki Śląskiej Wydziału Informatyki i Automatyki;
- prezentację naszych systemów w ww. szkołach a także u naszych kontrahentów.

Gwarantujemy:

- najwyższą jakość naszych systemów i usług,
- najszybsze w Polsce wdrożenia systemów (od 1 do 3 miesięcy).

W przypadku niedotrzymania przez nas terminów zwracamy 25% poniesionych nakładów za każdy miesiąc zwłoki. Dla klientów zamawiających kompleksowe usługi – znaczne zniżki cenowe.

Poszukujemy niezawodnych

- pośredników i dealer'ów naszych systemów informatycznych.

Z siły naszych systemów skorzystały dotąd następujące firmy:

Huta ŁAZISKA w Łaziskach Górnych	tel. 24-14-06
Walcownie Metali w Czechowicach-Dziedzicach	tel. 52-35-1
Centrala Zaopatrzenia Górnictwa w Bytomiu	tel. 81-92-91
Mikołowska Fabryka Transformatorów w Mikołowie	tel. 26-25-41
BPiKD PREFAMET-ZREMB w Gliwicach	tel. 38-20-51
BPiKD METALCHEM w Gliwicach	tel. 31-56-33
Instytut Informatyki Teoretycznej i Stosowanej PAN w Gliwicach	tel. 31-71-50
Instytut Spawalnictwa w Gliwicach	tel 31-00-11
PP ENERGOROZRUCH w Gliwicach	tel. 37-66-25
Fabryka Palenisk Mechanicznych w Mikołowie	tel. 26-20-02
Zakłady Mechaniczne Górnictwa i Energetyki WIROMET w Mikołowie	tel. 26-23-46
Przędzalnia Bawełny PRZYJAŹŃ w Zawierciu	tel. 22-39
PRT TERMOIZOLACJA w Zabrzu	tel. 71-40-21
Zakład Informatyki i Techniki Komputerowej MEGA w Gliwicach	tel. 31-52-42
Mikołowska Fabryka Maszyn Górniczych MIFAMA w Mikołowie	tel. 26-22-66
Zakłady Produkcji Urządzeń Mechanicznych ELWO w Pszczynie	tel. 30-61
Zakłady Sprzętu Sieciowego i Elektroinstalacyjnego POLAM w Katowicach	tel. 52-80-31
Zakłady Elektromaszynowe CELMA w Cieszynie	tel. 21-58-1
PSS SPOŁEM w Zawierciu	tel. 21-46-17

Chcesz mieć pełną gwarancję i niezawodne efekty komputeryzacji swojego przedsiębiorstwa – współpracuj z firmą JUNISOFTEX!

0/1/91

Projekt naprawy informatyki

Niestety nadszedł czas, aby przedstawić „własny" projekt naprawy informatyki w Polsce. Pojęcie „własny" umieściłem w cudzysłowach, gdyż zamieszczone niżej poglądy są nie tylko efektem własnego olśnienia, ale też wielu spotkań i dyskusji w gronie kolegów. Zgodnie z końcowym akapitem nie należy tych poglądów traktować jako ostateczne, ale bardziej jako materiał do dalszych dyskusji.

STAN AKTUALNY informatyki rozumianej jako zasoby kadrowe, sprzętowe i oprogramowania, umożliwiające informatyzację administracji państwowej oraz przedsiębiorstw, można określić jedynie jako katastrofalny. Pomijając szczegółowe wyliczenia braków wystarczy stwierdzić, że w obecnej chwili Polska nie dysponuje możliwościami produkcji własnego sprzętu informatycznego oraz podstawowego oprogramowania. Całość infrastruktury informatycznej jest oparta bądź na archaicznym starym sprzęcie, bądź na tysiącach mikrokomputerów sprowadzonych półlegalnie z Dalekiego Wschodu.

Jednakże do najbardziej negatywnych elementów tego zjawiska musimy zaliczyć gwałtownie zmniejszający się stan liczbowy wykwalifikowanej kadry informatycznej, spowodowany emigracją oraz zmniejszającymi się możliwościami intensywnego kształcenia. Zamiast dobrze wykształconych informatyków możemy zauważyć zjawisko coraz powszechniejszego kształcenia przypadkowego przez pseudoinformatyków, głównie rzemieślników-programistów. Prawie zerowe są możliwości kształcenia doświadczonych projektantów systemów informatycznych. Jednocześnie należy zauważyć, że ten zasób informatyki jest i będzie najtrudniejszy do odtworzenia.

ZAGROŻENIA. Katastrofalny stan informatyki niesie ogromne zagrożenia dla obecnego i przyszłego rozwoju ekonomicznego, politycznego i społecznego kraju. Całkowity brak własnej infrastruktury informatycznej uzależnia nas całkowicie od zachodnich firm, przy czym uzależnienie to będzie znacznie głębsze niż w jakiejkolwiek innej dziedzinie. Firmy te, przy braku polskiej kadry informatycznej, będą narzucały własne rozwiązania (bez niezależnej weryfikacji), a podczas realizacji i późniejszej eksploatacji systemów informatycznych będą miały dostęp do wszystkich informacji gospodarczych. Taka sytuacja, przy sprywatyzowanym przemyśle i decentralizacji zarządzania, szybko doprowadzi do pełnego uzależnienia naszej gospodarki od obcego kapitału i to na bardzo niezdrowych zasadach.

Jednocześnie warto podkreślić, że obecnie i w przyszłości o sprawności systemu finansowo-gospodarczego kraju decydują i coraz bardziej będą decydować sprawne systemy informatyczne. Powiązanie naszej gospodarki z systemem EWG i USA będzie wymagało podejmowania na każdym szczeblu szybkich decyzji ekonomicznych, gdyż tylko takie mogą przynosić zysk.

PROGRAM ROZWOJU. W obecnej chwili można jedynie mówić o programie minimalizowania skutków obecnie trwającej katastrofy informatycznej. Zdaję sobie bowiem sprawę z uwarunkowań ekonomicznych oraz ograniczeń, nawet natury sprawności decyzyjnej i legislacyjnej, obecnego zarządzania krajem.

Na początek uważam za konieczne ratowanie najtrudniejszego do odtworzenia zasobu – wykwalifikowanej kadry. W tym celu konieczne jest:

1. Finansowe wspomożenie kierunków informatycznych na uczelniach – zwiększając wynagrodzenie dla kadry nauczającej i przydzielając im specjalne granty na wyposażenie dydaktyczne. Należy wreszcie zróżnicować wynagrodzenia w dziedzinach podstawowych dla gospodarki, kosztem dziedzin obecnie mniej ważnych. Nie waham się tutaj zaliczyć do nich wielu nauk humanistycznych i matematyczno-fizycznych, takich jak astronomia, fizyka teoretyczna, technologia elektroniczna itp. Wiem, że sprowadzam na siebie tym samym gromy, ale – tutaj zwracam się do przedstawicieli tych nauk – czy istnieje dalsza możliwość ich prawidłowego rozwoju bez dobrze funkcjonującej informatyki? Kilka samodzielnie kupionych, systemów, najczęściej mikrokomputerowych, nie rozwiąże niczego. Potrzebne jest pełne, dobrze zorganizowane profesjonalne zaplecze informatyczne.

2. Częściowe wykorzystanie pieniędzy przeznaczonych na pomoc dla Polski na utworzenie szkół projektowania systemów informatycznych i zasad prawidłowej informatyzacji administracji oraz przedsiębiorstw (na wzór szkół zarządzania – na przykład pod patronatem PTI).

3. Utworzenie polskich specjalistycznych zespołów projektantów (początkowo dodatkowo szkolonych i wspomaganych przez ekspertów zachodnich) do opracowania strategii funkcjonowania globalnych systemów informatycznych w administracji państwowej, bankach oraz w wybranych przedsiębiorstwach (w tym ostatnim przypadku byłyby to projekty realizowane na zamówienie); jednocześnie należy przez odpowiedni poziom wynagrodzenia stworzyć z tych zespołów niezależne grupy konsultacyjne, uzupełnione emigracyjnymi specjalistami polskimi.

Następnym elementem programu jest przyspieszenie uregulowań prawnych w zakresie informatyki. Za najpilniejsze uważam:
4. Dokładne analizowanie skutków podpisania traktatów handlowych i umów rządowych dotyczących zobowiązań i „wzajemnych" korzyści w sferze informatyki.

5. Wprowadzenie ustawy o ochronie oprogramowania, a następnie jej rzeczywiste egzekwowanie. Być może należy utworzyć agencję zajmującą się z urzędu tą sprawą.

6. Wprowadzenie ustawy o ochronie dóbr osobistych i również jej rzeczywiste egzekwowanie.

7. Zakończenie procesów normalizacji kodów polskich liter dla klawiatur, ekranów, terminali oraz drukarek stosowanych w różnych typach komputerów i sieci informatycznych, a następnie pilnowanie ich stosowania na forum międzynarodowym i krajowym.

8. Dokończenia prac w sferze standaryzacji sprzętu (typów mikro- i minikomputerów, rodzajów kart sieciowych) oraz oprogramowania podstawowego (system UNIX, protokoły sieciowe, SQL-owe bazy danych), wykorzystywanego w administracji państwowej i lokalnej.

Trzecim elementem programu jest stymulowanie prawidłowości wdrażania systemów informatycznych opartych na rozwiązniach obcych. Dla realizacji tego zadania, w obecnych uwarunkowaniach organizacyjno-prawnych, możliwe jest zastosowanie następujących rozwiązań:

9. Wprowadzenie ceł na sprzęt i oprogramowania nie spełniające polskich standardów.

10. Zmniejszenie lub nawet zaniechanie pobierania ceł na oprogramowanie narzędziowe, służące do realizacji aplikacji, oraz oprogramowania użytkowego (w celu przyspieszenia wprowadzenia na polskim rynku oprogramowania licencjonowanego).

11. Wprowadzenie preferencyjnej polityki podatkowej dla zachodnich firm sprzedających swój sprzęt oraz oprogramowanie we współpracy z polskimi firmami (łącznie ze szkoleniem polskich informatyków i tworzeniem zespołów do realizacji polskich aplikacji).

12. Egzekwowanie podatków od firm zachodnich wyłącznie sprzedających swoje produkty bez inwestowania w rozwój swojej działalności na rynku polskim.

13. Opracowanie prawnej formuły kontraktów na zakup sprzętu i oprogramowania od zagranicznych firm, uwzględniającej nasze krajowe interesy (zgodność z polskimi przepisami, polski arbitraż itp.).

Podany szkic programu jest jedynie pierwszym krokiem w kierunku zahamowania obecnej, katastrofalnej sytuacji. W dalszej perspektywie konieczne będzie jeszcze realizowanie podstawowej dla rozwoju informatyki czynności – opracowanie strategii rozwoju gospodarczo-ekonomicznego kraju w sferze administracji państwowej, lokalnej, obronności, nauki oraz opieki zdrowotnej i socjalnej.

Zdaję sobie sprawę, że opracowanie takiej kompleksowej strategii przekracza w obecnej chwili możliwości Sejmu, Senatu i Rządu. Dlatego też postulat ten zostawiam na następne lata. Jednak przy podejmowaniu nawet cząstkowych ustaleń należy zwrócić uwagę, że ich integralną częścią powinny być ustalenia związane z formą zastosowania w nich odpowiedniego systemu informatycznego.

WACŁAW ISZKOWSKI

P.S. zBITki tu prezentowanych poglądów i spostrzeżeń nie są oficjalnym stanowiskiem Redakcji ani żadnej innej organizacji. Ich prezentacja ma służyć pobudzeniu dyskusji na dany temat, nawet przez wzburzenie moich ewentualnych adwersarzy.

Ze świata

Plotery dla systemów CAD

dokończenie z III strony okładki

już 80% jakości rysunku po 20 latach, pod warunkiem, że rysunek będzie przechowywany bez dostępu światła.

Technika termiczna daje rysunki tylko czarno-białe, ponieważ pod działaniem temperatury papier czernieje. Próby zastosowania dwóch poziomów temperatury w celu otrzymania na papierze dwóch kolorów (czarnego i czerwonego) nie dały wyników spełniających wymagania profesjonalne.

Ze względu na dużą wydajność, dla wielu użytowników najlepszym rozwiązaniem może być ploter elektrostatyczny. Na miejscach naładowanych elektrostatycznie pojawia się ciekły toner. Jest to technika już wypróbowana, budząca znaczne zainteresowanie. Plotery elektrostatyczne wytwarzają rysunki kolorowe lub monochromatyczne, dobrze wypełniają obrysowane pola, znaki opisów tekstowych są ostre i możliwe jest otrzymywanie efektów przestrzennych. Rysują szybko, nie wymagają częstego uzupełniania tonera i mogą pracować bez dozoru. Głównymi dostawcami tego typu ploterów są firmy Versatec, Hewlett-Packard i Calcomp. Kolorowe plotery elektrostatyczne są coraz bardziej popularne wśród inżynierów elektroników, projektujących płytki drukowane i układy scalone. Są one jednak stosunkowo drogie, ponieważ kosztują od 13 000 do 70 000 dolarów. Koszty eksploatacji są jednak mniejsze, niż się tego można by spodziewać, mimo że potrzebne są specjalne, wielowarstwowe nośniki. Problemem jest jednak szkodliwość tonera dla środowiska. Toner jest produktem naftopochodnym, a więc jest palny i jako odpad stwarza trudności składowania. Wielu dostawców rozwiązało ten problem przez gromadzenie zużytych pojemników i regenerowanie tonera. Stało się to normą w Zachodniej Europie.

Najszybsze są plotery laserowe, które mogą wykonywać do dziesięciu rysunków formatu A0 w ciągu minuty. Również one stosują ładowanie elektrostatyczne, ale nie papieru, lecz wirującego bębna. Następnie do naładowanych miejsc bębna przyklejają się drobiny suchego tonera, który zostaje przeniesiony na papier i utrwalony termicznie. Plotery laserowe pracują na zwykłym papierze i dają doskonałą jakość kreski, wypełnień i tekstów. Jednak dokładność jest gorsza niż innych ploterów rastrowych, ponadto są drogie – od 35 000 dolarów. Obecnie oferowane plotery wielkoformatowe są tylko monochromatyczne.

Wiele spośród zalet opisanych technik łączy w sobie stosunkowo rzadko stosowana technika natryskowa (ang. *inkjet*). Dysze, osadzone w nieruchomej głowicy natryskują na papier tusz z naboi umieszczonych w głowicy. Poza doskonałą jakością reprodukcji monochromatycznych, można uzyskiwać dowolne kolory i stosować powszechnie dostępne standardowe nośniki. Technika natryskowa jest wolniejsza od innych technik rastrowych, ale i tak szybsza od pisakowej. Poza tym jest środowiskowo czysta, ponieważ nie pozostawia żadnych szkodliwych materiałów. Jej ważną zaletą jest możliwość drukowania na zwykłych papierach.

Technikę natryskową opracowano początkowo dla drukarek i ploterów małoformatowych. Wysiłki zmierzające do zastosowania tej niezawodnej techniki w urządzeniach wielkoformatowych otwierają możliwości bardzo szybkiego wzrostu popularności ploterów natryskowych. Specjaliści są przekonani, że w przyszłości będzie to najlepsza technika, a dostawy takich ploterów wzrosną od 80 sztuk w 1990 r. do 5000 w 1993 r.

Wielkoformatowe plotery natryskowe mogą mieć znakomitą wydajność za stosunkowo niską cenę i stać się najbardziej dynamicznie rozwijającym się typem ploterów, ponieważ są lepsze przy wypełnianiu obszarów, dobre do drukowania tekstów i dają bajeczne kolorowe rysunki, twierdzi Porell. Uważa się, że ceny mogą obniżyć się do poziomu 20 000 dolarów.

Choć wytwórcy na ogół ociągają się z podawaniem szczegółowych informacji o nowo wprowadzanych wyrobach, Michael Reinhardt, wprowadzający wielkoformatowe plotery fimy Hewlett-Packard w Europie, był gotowy do ujawnienia szczegółów tej techniki. *Nasz sukces u użytkowników komputerów osobistych, jaki odnieśliśmy z wyrobami opartymi na technice inkjet, zachęcił nas do zbadania jej potencjalnych możliwości w sektorze CAD. Sprawa ta ma bardzo dobre rokowania i wierzymy, że technika natryskowa może mieć ogromne zalety w szerokim zakresie zastosowań.* Na razie wielkoformatowe plotery natryskowe są przeznaczone tylko dla określonych bardzo wyspecjalizowanych zastosowań i z uwagi na cenę są niedostępne dla typowych użytkowników systemów CAD.

Należy się spodziewać, że zapotrzebowanie rynku na plotery będzie rosło w tempie około 10% rocznie, podobnie jak cały rynek sprzętu komputerowego, a udziały poszczególnych krajów i stref geograficznych nie ulegną większym zmianom. Według Porella *...nie dziwi fakt, że w systemach CAD najwięcej ploterów pracuje w Wielkiej Brytanii i w Niemczech, najbardziej uprzemysłowionych krajach Europy.* Z 31 100 ploterów sprzedanych w 1990 r. 24% trafiło do Niemiec, 18% do Wielkiej Brytanii i niemal 15% do Francji.

Choć specjaliści nadal dyskutują o wadach i zaletach różnych technik, tą są zgodni co do tego, że nie ma jednej, jedynej techniki, która mogłaby stanowić najtrafniejsze rozwiązanie dla wszystkich zastosowań w systemach CAD. Należy spodziewać się, że do końca dekady lat dziewięćdziesiątych różne rozwiązania będą stosowane równolegle.

Ostrowski P., Średniawa M.: Język specyfikacji SDL (1)
INFORMATYKA 1991, Nr. 9, s. 3
Pierwsza część charakterystyki języka SDL, zawierająca historię rozwoju oraz szczegółowe omówienie struktury, możliwości funkcjonalnych i zasad jego użytkowania.

Ostrowski P., Średniawa M.: The SDL specification language (1)
INFORMATYKA 1991, Nr. 9, p. 3
First part of the SDL language characteristics, which includes history of development, as well as detailed discussion of its structure, functional possibilities and usage rules.

Ostrowski P., Średniawa M.: SDL-Spezifizierungssprache (1)
INFORMATYKA 1991, Nr. 9, S. 3
Erster Teil einer Charakteristik von der SDL-Sprache, der die Entwicklungsgeschichte dieser Sprache, sowie eine detaillierte Bresprechung ihrer Struktur, Funktionsmöglichkeiten und Anwendungsregel umfasst.

Cierpisz T.: Problem i metody kompresji danych
INFORMATYKA 1991, Nr. 9, s. 10
Charakterystyka problemu oraz omówienie najczęściej stosowanych i najbardziej efektywnych metod kompresji danych.

Cierpisz T.: The problem and methods of data compression
INFORMATYKA 1991, Nr. 9, p. 10
Charakteristics of the problem and discussion of mostly applied and most effective methods for data compression.

Cierpisz T.: Das Problem und Methoden der Datenverdichtung
INFORMATYKA 1991, Nr. 9, S. 10
Eine Charakteristik des Problems und eine Besprechung der am häufigsten verwendeten und effektivsten Methoden der Datenverdichtung.

Klapper M.: Praktyczne aspekty kompleksowej analizy funkcjonalnej przedsiębiorstwa i metodyka ustalania faktów
INFORMATYKA 1991, Nr. 9, s. 14
Charakterystyka organizacji kompleksowej analizy funkcjonalnej przedsiębiorstwa oraz efektywnej metody ustalania faktów dla potrzeb takiej analizy.

Klapper M: Practical aspects of enterprise integraged functional analysis and methodology for facts establishing
INFORMATYKA 1991, Nr. 9, p. 14
Charakteristics of the organization for enterprise integrated functional analysis and of effective method for facts establishing.

Klapper M.: Praktische Aspekte der integrierten Funktionalanalyse eines Unternehmens und Methodik der Faktenfestlegung
INFORMATYKA 1991, Nr. 9, S. 14
Eine Charakteristik von Organisation eier integrierten Funktionalanalyse eines Untermehmens und von effektiven Methoden der Faktenfestlegung für solche Analyse.

Łukasik-Makowska B.: Standaryzacja systemów powielarnych
INFORMATYKA 1991, Nr. 9, s. 21
Znaczenie standaryzacji systemów informatycznych dla rozwoju i efektywności zastosowań oraz omówienie sposobów realizacji tego zadania.

Łukasik-Makowska B.: Standarization of adaptable systems
INFORMATYKA 1991, Nr. 9, p. 21
Significance of data processing systems standarization for application development and effectiveness, as well as discussion of realization methods for this purpose.

Łukasik-Makowska B.: Standardsierung von adaptablen Systemen
INFORMATYKA 1991, Nr. 9, S. 21
Bedeutung von Standardisierung der EDV-Systeme für Entwicklung und Effektivität von Anwendungen, sowie eine Besprechung der Realisierungsweise von solchen Aufgaben.

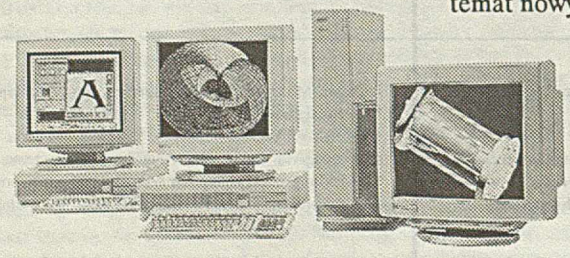
Liczby przemawiają głośniej, niż słowa.

57

57 MIPS/$ 15K

76

76 MIPS/$ 25K



Aby dowiedzieć się więcej na temat nowych stacji roboczych

HP Apollo seria 700 RISC prosimy dzwonić do nas pod numer 37 47 82 i prosić Włodzimierza Kaweckiego.

HEWLETT PACKARD

Hewlett-Packard Polska sp. z o.o. , 01-447 Warszawa, ul.Newelska 6, tlx 813919, fax 374783

0/12/91

Plotery dla systemów CAD

Czym byłby system CAD bez stosownego plotera? Wyświetlana na ekranie kolorowa grafika, subtelne cieniowanie obrazów i wspaniałe efekty przestrzenne byłyby mało użyteczne, gdyby nie można było utrwalić ich na papierze. W pracy inżyniera mechanika, projektanta układów półprzewodnikowych czy geologa sporządzającego mapy, o powodzeniu systemu CAD w rozstrzygającym stopniu decyduje trafny wybór plotera. Oferta tych urządzeń jest bardzo bogata. Najtańsze można nabyć już za około 10 000 dolarów, a cena najdroższych przekracza 70 000 dolarów. Także rozwiązania techniczne mogą być bardzo różne. Decyzje zakupu, jakie podejmują użytkownicy, są zazwyczaj kompromisem między kosztem a technicznymi możliwościami urządzenia.

Plotery dla systemów CAD można podzielić na dwa podstawowe rodzaje **wektorowe** i **rastrowe**. Rozwiązania wektorowe przypominają kreślenie ręczne (uzborojona w pisaki ruchoma głowica plotera przesuwa się nad powierzchią papieru). Aby więc wykreślić np. dużą literę X, ploter wektorowy zacznie od jej górnego lewego narożnika i będzie kreślić linię prostą w dół ku dolnemu prawemu, po czym przeniesie pisak do górnego narożnika i wykreśli drugi odcinek prostej w lewo w dół.

Plotery rastrowe tworzą natomiast obrazy z kolejnych poziomych linii, podobnie jak powstaje obraz telewizyjny. Głowica kreśląca takiego plotera obejmuje całą szerokość papieru, przy czym w jednej linii może rysować z gęstością sięgającą 406 kropek na cal (16 kropek na milimetr). Porusza się w nim nie głowica, lecz papier. Przy kreśleniu dużego X najpierw pojawią się więc dwa pionowe, zbieżne odcinki utworzone z kropek, które po przecięciu się w środku litery, ponownie się rozejdą.

Najbardziej zawiły rysunek, np. topologię płytki drukowanej czy układu scalonego, ploter rastrowy teoretycznie może wykreślić równie szybko, jak prosty rysunek kwadratu. Plotery rastrowe są droższe od wektorowych, a pomimo to nie kreślą równie ostro i czysto jak wektorowe.

Najpowszechniej dotąd stosowane są plotery wektorowe. Ich ceny zaczynają się od poziomu poniżej 10 000 dolarów. W 1990 r. na rynku europejskim ploterów wektorowych sprzedano 27 900 sztuk, co stanowiło ok. 90% wszystkich sprzedanych ploterów. Również ich koszty eksploatacji są niższe, gdyż mogą one kreślić na różnych gatunkach papieru, a tusz do pisaków jest stosunkowo tani.

Trzeba jednak pamiętać, że plotery wektorowe są mało sprawne przy tworzeniu dużych, kolorowych powierzchni, a cieniować w ogóle nie mogą. Przy kreśleniu skomplikowanych rysunków są stosunkowo powolne, ponieważ muszą kreślić każdą linię osobno. Przy systematycznym kreśleniu pisaki mogą się zatykać i wymagają ciągłego dozoru. Technika wektorowa jest jednak bardziej atrakcyjna dla nowych użytkowników, bo przypomina ręczne kreślenie. Plotery pisakowe są idealne do sporządzania stosunkowo prostych dwuwymiarowych rysunków, kreślonych pojedynczymi odcinkami, jakie występują u architektów czy konstruktorów mechaników. Koszt tych ploterów jest umiarkowany, mogą kreślić na najróżniejszych nośnikach, a jakość kolorowych rysunków – doskonała. Według Greg Porella, dyrektora europejskiej grupy ds. typografii elektronicznej w ramach BIS Strategic Decisions, kreślenie pisakiem to doskonale dopracowana i wysoce niezawodna technika, która niemal osiągnęła granice możliwości, jeśli chodzi o szybkość i jakość wyciągania linii. Jej życie przedłużają takie cechy, jak np. podawanie papieru z rolki, aby ploter mógł bez obsługi sporządzać ciągi kolejnych rysunków, czy też odciążenie jednostki centralnej komputera przez wykorzystywanie buforowych pamięci. Plotery pisakowe, produkowane przez takie firmy, jak Hewlett-Packard, Calcomp i OCE, to najrozsądniejsze rozwiązanie dla użytkowników sporządzających do piętnastu rysunków dziennie.

Niemniej Porell uważa, że techniczne ulepszenia i obniżka cen ploterów rastrowych doprowadzą w połowie 1994 r. do zmniejszenia się udziału ploterów pisakowych (wektorowych) na rzecz rastrowych. Już teraz plotery rastrowe okazują się być bardziej opłacalne dla użytkowników sporządzających ponad 25 rysunków dziennie, czy sporządzających rysunki bardziej skomplikowane. Obecnie w ploterach rastrowych są stosowane cztery rozwiązania sposobów rysowania: bezpośrednie termiczne, elektrostatyczne, laserowe i natryskowe (inkjet).

Technika bezpośredniego termicznego rysowania (ang. *direct thermal*) jest prosta i choć stosunkowo nowa, jest już dobrze znana, ponieważ jest stosowana w telefaksach. Kropki powstają na termoczułym papierze bezpośrednio pod działaniem gorących igieł. Pewną jej wadą jest to, że po wytworzeniu rysunku reakcje chemiczne nie ustają i obraz stopniowo ulega rozmyciu, co przy sporządzaniu ważnych dokumentów może być niedopuszczalne. Ponadto wielu użytkowników nie lubi posługiwać się papierem „telefaksowym", nasyconym substancją szkodliwą dla środowiska.

Pomimo tego, stosunkowo niedrogie (10 000–15 000 dolarów) plotery rastrowe rysujące na termoczułym papierze stały się dość popularne. Są szybsze od pisakowych i coraz częściej kupowane. Ponadto notuje się pewien postęp w trwałości wydruków: niektórzy producenci gwarantują

dokończenie na s. 34

5 + 1 DNI

COMMUNICHATION

Na targach SYSTEMS 91 prezentowani są wszyscy najważniejsi krajowi i zagraniczni wystawcy ze swoimi najnowszymi i najatrakcyjniejszymi produktami. Europa jest wiodącym i ciągle rozwijającym się rynkiem w zakresie **techniki komputerowej, łączności, software, technologii sieci, komponentów systemowych** i wszelkich związanych z tą branżą usług. Targi specjalistyczne SYSTEMS i Kongres prezentują aktualne osiągnięcia w dziedzinie łączności. Po raz pierwszy Monachijskie Towarzystwo Wystaw i Targów przedłużyło targi o 1 dzień. 5 + 1. Ten dzień to więcej czasu na uzyskanie dodatkowych informacji. To również dodatkowy plus dla nowych kontaktów na targach i w Monachium.

SYS® **SYSTEMS 91**

Komputery i Łączność
12 Międzynarodowe Targi Specjalistyczne
i Międzynarodowy Kongres
Monachium 21–26 październik 1991
Sobota na Targach i w Monachium

Informacji udzielają:
Monachijskie Towarzystwo Wystaw i Targów
Skr. poczt. 121009, D-8000 Monachium 12
Telefon 089/5107-0
Przedstawicielstwo na Polskę:
Przedsiębiorstwo Wystaw i Targów Zagranicznych
POLEXPO
02-232 Warszawa, ul. Łopuszańska 38
Tel. 46 45 92
Tlx. 813633 polex pl, Fax. 46 45 91

MESSE MÜNCHEN INTERNATIONAL