ISSN 0542 9951

5
1996

NAJSTARSZE PISMO INFORMATYCZNE W POLSCE – UKAZUJE SIĘ OD 1965 ROKU

informatyka

miesięcznik profesjonalistów

infosystem

Tegoroczne Targi INFOSYSTEM obchodziły jubileusz dziesięciolecia, a razem z nimi świętowało 130 - te urodziny najstarsze pismo techniczne w Europie - „Przegląd Techniczny". To właśnie w tej redakcji narodził się pomysł, by uczcić swoje 120-lecie zorganizowaniem międzynarodowej wystawy komputerowej. Na miejsce tej nowatorskiej imprezy wybrano Wrocław, ówczesną polską „Dolinę Krzemową", gdzie królował gigant naszej elektroniki - dziś już nie istniejące zakłady ELWRO. Od tamtego czasu upłynęło tylko 10 lat, ale lepiej powiedzieć „aż" 10 lat, bowiem dla polskiego rynku informatycznego to cała epoka. Najprościej można by streścić ten okres jednym zdaniem: od fascynacji „pudełkami" przeszliśmy do zrozumienia, że są one tylko poręcznymi narzędziami, których właściwe wykorzystanie zależy od tego, jakie zadania i w jaki sposób potrafimy im powierzyć. Tę zmianę dostrzegły działające na naszym rynku firmy komputerowe. Było to wyraźnie widoczne na warszawskich targach Komputer Expo '96, gdzie wiele liczących się światowych firm już nie próbowało epatować potencjalnych klientów i zwiedzających nowymi konfiguracjami swoich komputerów, ale prezentowało je w otoczeniu partnerów, którzy udowadniali, co ten sprzęt potrafi zrobić dzięki ich oprogramowaniu czy wykorzystaniu w oferowanym przez nie systemie.

Na jubileuszowym INFOSYSTEM-ie ta zmiana była jeszcze bardziej widoczna i w dodatku zaowocowała zmianami na liście wystawców. Zabrakło na niej kilku znaczących dystrybutorów, którzy nie byli obecni jako wystawcy za to oferowali swoim handlowym partnerom wieczorne spotkania towarzyskie.

dokończenie na s. 23

W medycynie istnieje cały system ciągłego podnoszenia kwalifikacji lekarzy, związany z uzyskiwaniem kolejnych stopni specjalizacyjnych. Po przejściu przeszkoleń w różnych trybach, praktyk i zdaniu egzaminów lekarz może uzyskać kolejny stopień specjalizacji w konkretnej dziedzinie. Z systemem tym związana jest cała gama zachęt: od warunków koniecznych do zatrudnienia na poszczególnych stanowiskach, przez większe szanse w konkursach aż po prozaiczne dodatki do uposażeń. Wszystkie te uwarunkowania powodują, że system nie jest martwy - - lekarze starają się stale podnosić swoje kwalifikacje.

Analogiczny system próbowano wprowadzić w Polsce w dziedzinie nauk technicznych. Z różnych powodów jego efekty są raczej mierne. Idea zdobywania stopni specjalizacji zawodowej inżynierów nie znalazła, jak do tej pory, szerokiego oddźwięku ani wśród pracodawców, ani wśród zawodowej grupy inżynierów. W niektórych dziedzinach (np. budownictwie) zastąpił ją niejako system uprawnień zawodowych.

Informatyka jest dziedziną dość specyficzną. W czystej naukowej postaci występuje bardzo rzadko, w praktyce natomiast posługują się nią (w czynny lub bierny sposób) prawie wszyscy aktywni zawodowo. Jest też jedną z najszybciej rozwijających się dziedzin dostępnych ogółowi społeczeństwa. Roczne-dwuletnie opóźnienie w stosowaniu technologii informatycznych jest porównywalne z dziesiątkami lat w innych dziedzinach. Ta dynamiczność zmian, a także stale rozszerzające się obszary zastosowań, stawiają coraz to nowe zadania przed ludźmi zajmującymi się szeroko pojętą technologię informatyczną. Wszyscy oni powinni stale podnosić swoje kwalifikacje.

O programowaniu obiektowym, czyli układaniu większej całości z gotowych elementów

czytaj s. 3-35

Artykuł Piotra Fuglewicza i Marka Miłosza o informatycznych stopniach zawodowych - strona 36

WYDAWNICTWO SIGMA NOT

Cena 5,00 zł (50 000 zł)

Aż tak mała?!

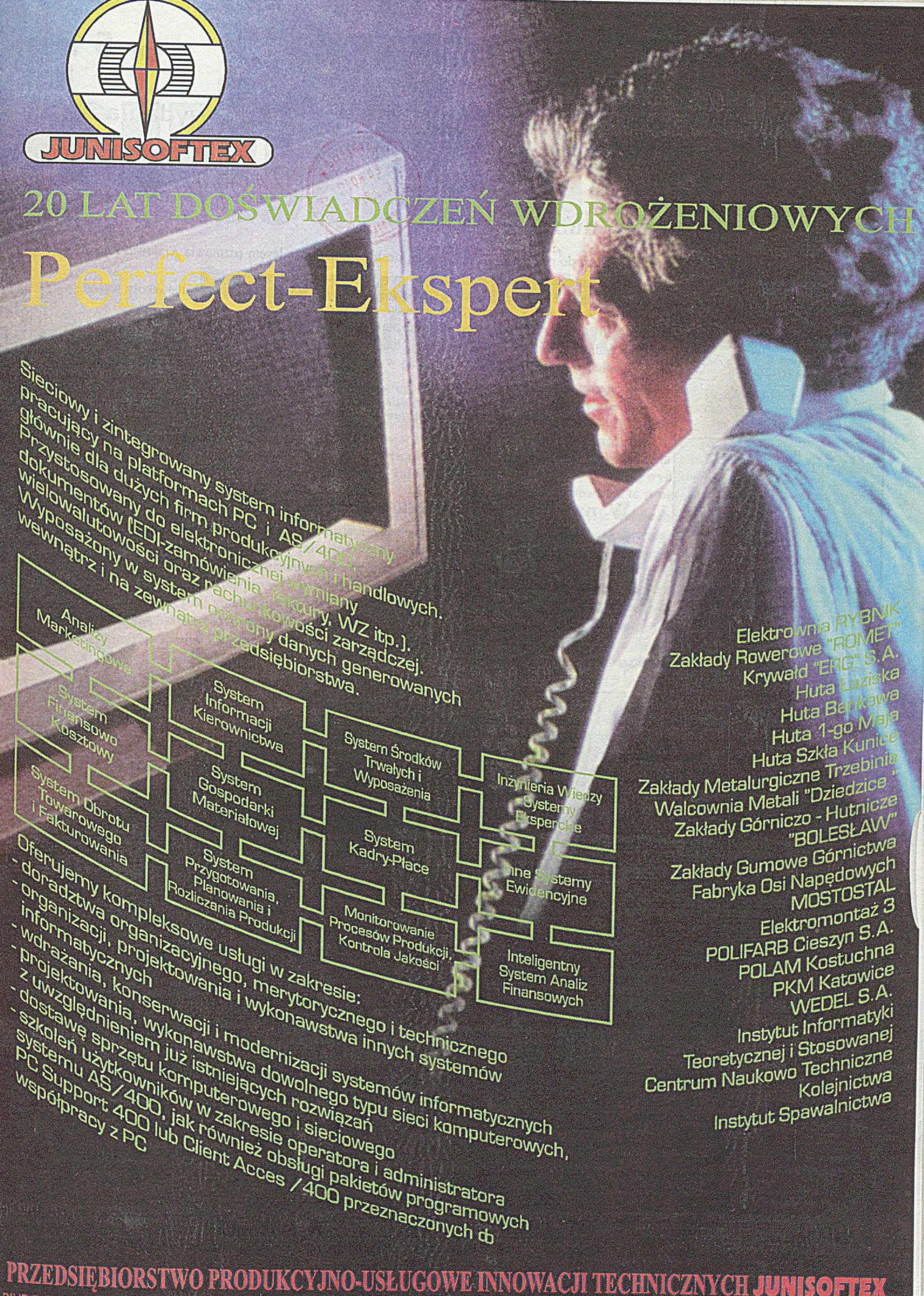
OKIPAGE 4w

drukarka stronicowa

Tak - i możesz ją mieć!

Tak mała, bo mniejsza od zwykłej kartki papieru formatu A4, a drukuje z dokładnością lasera i rozdzielczością 600 dpi. Dzięki specjalnym systemom oszczędzania energii i tonera oraz trwałości elementów (5 lat gwarancji na głowicę) jest tania w eksploatacji. **Tak mało** kosztuje - bo tylko 999 zł. Przeznaczona do pracy w środowisku Windows 3.1, Windows 95. Tak tania, tak mała i tak dobra **za 999 zł - i możesz ją mieć.**

OKI
People to People Technology


JUNISOFTEX
20 LAT DOŚWIADCZEŃ WDROŻENIOWYCH
Perfect-Ekspert
Sieciowy i zintegrowany system informatyczny pracujący na platformach PC i AS/400, głównie dla dużych firm produkcyjnych i handlowych. Przystosowany do elektronicznej wymiany dokumentów (EDI-zamówienia, faktury, WZ itp.). Wyposażony w system ochrony danych generowanych wewnątrz i na zewnątrz przedsiębiorstwa.
Analizy Marketingowe
System Informacji Kierownictwa
System Środków Trwałych i Wyposażenia
System Finansowo Kosztowy
System Gospodarki Materiałowej
Inżynieria Wiedzy Systemy Ekspertyzy
System Obrotu Towarowego i Fakturowania
System Kadry-Płace
Inne Systemy Ewidencyjne
System Przygotowania, Planowania i Rozliczania Produkcji
Monitorowanie Procesów Produkcji, Kontrola Jakości
Inteligentny System Analiz Finansowych
Oferujemy kompleksowe usługi w zakresie:
- doradztwa organizacyjnego, merytorycznego i technicznego
- organizacji, projektowania i wykonawstwa innych systemów informatycznych
- wdrażania, konserwacji i modernizacji systemów informatycznych
- projektowania, wykonawstwa dowolnego typu sieci komputerowych, z uwzględnieniem już istniejących rozwiązań
- dostawę sprzętu komputerowego i sieciowego
- szkoleń użytkowników w zakresie operatora i administratora systemu AS/400, jak również obsługi pakietów programowych PC Support 400 lub Client Acces /400 przeznaczonych do współpracy z PC
Elektrownia RYBNIK
Zakłady Rowerowe "ROMET"
Krywałd "ERG" S.A.
Huta Łaziska
Huta Bankowa
Huta 1-go Maja
Huta Szkła Kunice
Zakłady Metalurgiczne Trzebinia
Walcownia Metali "Dziedzice"
Zakłady Górniczo - Hutnicze "BOLESŁAW"
Zakłady Gumowe Górnictwa
Fabryka Osi Napędowych MOSTOSTAL
Elektromontaż 3
POLIFARB Cieszyn S.A.
POLAM Kostuchna
PKM Katowice
WEDEL S.A.
Instytut Informatyki Teoretycznej i Stosowanej
Centrum Naukowo Techniczne Kolejnictwa
Instytut Spawalnictwa
PRZEDSIĘBIORSTWO PRODUKCYJNO-USŁUGOWE INNOWACJI TECHNICZNYCH JUNISOFTEX

informatyka

Miesięcznik
ISSN 0542-9951
INDEKS 36124

Uwaga, dzięki gościnności Polska Online ukazuje się już elektroniczna wersja naszego miesięcznika. Oto nasz adres w Internecie:
http://www.pol.pl/informatyka/

nr 5 1996
Maj
Rok wydania XXXII

WYDAWCA:
Wydawnictwo Czasopism i Książek Technicznych SIGMA NOT Spółka z o.o.
ul. Ratuszowa 11
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja:
00-950 Warszawa,
ul. Ratuszowa 11, p. 628 i 644
skrytka pocztowa 1004
tel., fax: 619-11-61
tel.: 619-22-41 w. 159
e-mail:
informat@pol.pl

Materiałów nie zamówionych redakcja nie zwraca

Autorzy artykułów proszeni są o przysyłanie tekstów na dyskietkach $3^1/_2$" – w czystych kodach ASCII (Latin II, Mazovia) lub edytorach: Word 2, Word 6 – oraz odbitki na papierze. Teksty nie mogą przekraczać 20 tys. znaków.

Redakcja nie ingeruje w treść i formę ogłoszeń i innych materiałów reklamowych, w związku z tym nie ponosi za nie odpowiedzialności.

W numerze:

W najbliższych numerach:

- **Marek Prokop omawia możliwości wykorzystania cech obiektowych języka Progress 4GL w oprogramowaniu relacyjnych baz danych.**
- **Marcin Gorawski i Andrzej Konopacki podają kompendium wiedzy na temat Magic-a, generatora aplikacji baz danych.**
- **Andrzej Wolski charakteryzuje nowy język obiektowy Java.**
- **Jacek Sedel i Michał Kolano prezentują język SQLWindows – nowoczesne narzędzie do tworzenia aplikacji korzystających z baz danych.**
- **Maciej Piasecki opisuje koncepcję modelu dla obiektowej bazy danych.**

Warunki prenumeraty na 1996 r.

Zamówienia na prenumeratę czasopism wydawanych przez Wydawnictwo SIGMA-NOT można składać w dowolnym terminie. Mogą one obejmować dowolny okres czasu, tzn. dotyczyć dowolnej liczby kolejnych zeszytów każdego czasopisma.

Zamawiający może otrzymywać zaprenumerowany przez siebie tytuł począwszy od następnego miesiąca po dokonaniu wpłaty. Zamówienia na zeszyty sprzed daty otrzymania wpłaty będą realizowane w miarę możliwości – z posiadanych zapasów magazynowych.

Warunkiem przyjęcia i realizacji zamówienia jest otrzymanie z banku potwierdzenia dokonania wpłaty przez prenumeratora. Dokument wpłaty jest równoznaczny ze złożeniem zamówienia.

Wpłat na prenumeratę można dokonywać na ogólnie dostępnych blankietach w Urzędach Pocztowych (przekazy pieniężne) lub Bankach (polecenie przelewu), przekazując środki pod adres:
Wydawnictwo SIGMA-NOT Spółka z o.o.
Zakład Kolportażu
00-716 Warszawa, skr. poczt. 1004
konto:
PBK S.A. III 0/Warszawa nr 370015-1573-139-11

Wpłaty na prenumeratę przyjmują także wszystkie urzędy pocztowe nadawczo-odbiorcze oraz doręczyciele na terenie całego kraju

Na blankiecie wpłaty należy czytelnie podać nazwę zamawianego czasopisma, liczbę zamawianych egzemplarzy, okres prenumeraty oraz własny adres.
Na życzenie prenumeratora, zgłoszone np. telefonicznie, Zakład Kolportażu ul. Bartycka 20, 00-950 Warszawa, (telefony: 40-30-86, 40-35-89 oraz 40-00-21 wew. 249, 293, 299) wysyła specjalne blankiety zamówień wraz z aktualną listą tytułów i cennikiem czasopism.

Odbiorcy zagraniczni mogą otrzymywać czasopisma poprzez prenumeratę dewizową (wpłata dokonywana poza granicami Polski w dewizach, wg cennika dewizowego z cenami podanymi w dolarach amerykańskich) lub poprzez zamówioną w kraju prenumeratę ze zleceniem wysyłki za granicę (zamawiający podaje dokładny adres odbiorcy za granicą, dokonując równocześnie wpłaty w wysokości dwukrotnie wyższej niż cena normalnej prenumeraty krajowej).

Egzemplarze archiwalne (sprzedaż przelewowa lub za zaliczeniem pocztowym) można zamawiać pisemnie, kierując zamówienia pod adresem: Wydawnictwo SIGMA NOT, Spółka z o.o. Zakład Kolportażu, 00-716 Warszawa, ul. Bartycka 20, paw. B, tel. 40-37-31, natomiast za gotówkę można je nabyć w Klubie Prasy Technicznej w Warszawie ul. Mazowieckiej 12, tel. 26-80-17.

Istnieje możliwość zaprenumerowania 1 egz. czasopisma po cenie ulgowej przez indywidualnych członków stowarzyszeń naukowo-technicznych zrzeszonych w FSNT oraz przez uczniów zawodowych i studentów szkół wyższych. Blankiet wpłaty na prenumeratę ulgową musi być opatrzony na wszystkich odcinkach pieczęcią koła SNT lub szkoły.

W przypadku zmiany cen w okresie objętym prenumeratą Wydawnictwo zastrzega sobie prawo do wystąpienia o dopłatę różnicy cen oraz prawo do realizowania prenumeraty tylko w pełni opłaconej.
Cena jednego egzemplarza: normalna 5,00 zł (50 000 zł), ulgowa 3,75 zł (37 500 zł)
Wartość prenumeraty w zł:
Normalna: kwartalna 15,00 zł (150 000 zł), półroczna 30,00 zł (300 000 zł), roczna 60,00 zł (600 000 zł)
Ulgowa: kwartalna 11,25 zł (112 500 zł), półroczna 22,50 zł (225 000 zł), roczna 45,00 zł (450 000 zł)

Skład i druk: Drukarnia SIGMA NOT Sp. z o.o. z. 188/96

publikacje

Programowanie obiektowe - nowa jakość programowania

Gry komputerowe oprócz pełnienia funkcji rozrywkowych stanowią doskonały sposób promocji informatyki, szczególnie wśród młodzieży, oraz zachęcają programistów, aby w nowy sposób podchodzili do wytwarzania oprogramowania. Jedną z najwcześniej stosowanych jest prosta zabawa polegająca na układaniu klocków. Marzeniem układających są klocki typu lego, które pozwalają na maksymalne rozwinięcie inwencji konstruktorów i pozwalają budować niewyobrażalne wprost formy. Atrybutem tego typu klocków jest standardowa modułowa budowa i z tej cechy wynika możliwość ich łączenia w konfiguracje wynikające z potrzeb konstruktora. Informatyczna zabawa w klocki, której wynikiem jest wytwarzanie oprogramowania to programowanie obiektowe.

Każdy aktywny programista obok standardowych narzędzi programowania i bibliotek ma i stosuje własne moduły programowe, które wspomagają go w procesie tworzenia oprogramowania. Własne standardy są tworzone w celu ułatwienia i przyspieszenia procesu programowania. Własne uniwersalne moduły programowe są na różnym poziomie szczegółowości i najczęściej uwzględniają oryginalny styl programowania, przyzwyczajenia i potrzeb programisty.

Szybki rozwój technologii informatycznych powoduje, że wytworzone biblioteki prywatne bardzo często muszą być modyfikowane ze względu na nowe potrzeby zmieniającego się środowiska sprzętowego i programowego. Stosowany styl pracy programistów zmusza ich do częstych modyfikacji własnego środowiska lub wypracowywania nowego zestawu stosowanych własnych narzędzi wytwarzania oprogramowania. Zastosowanie standardów w programowaniu i wspomaganie ich uniwersalnymi modułami programowymi, niezależnymi od środowiska, wychodzi naprzeciw oczekiwaniu programistów.

Programowanie obiektowe to naturalny rozwój technologii procedur i modułów programowych oraz innych metod wytwarzania oprogramowania, których źródło jest w zasadzie „dziel i rządź". Podział złożonego problemu na mniejsze części pozwala na łatwiejsze zrozumienie i podział pracy wśród członków zespołu realizatorów. Części, na które zostanie rozłożony problem, powinny być tak skonstruowane, aby łatwo można było je łączyć w większe obiekty, zgodnie z potrzebami rozwiązywanego problemu.

Mając odpowiednio dużą kolekcję klocków, obiektów programowych, które łatwo dają się zestawiać, programista może skoncentrować się na rozwiązywaniu problemu. Szczegółowe rozwiązania programowe są ukryte w obiektach programowych i podobnie, jak w przypadku konstruowania z wykorzystaniem klocków lego, konstruktor nie rozprasza się na szczegóły, tylko buduje większą konstrukcję wykorzystując cechy mniejszych obiektów i możliwości ich łączenia dla celów tworzonej budowli, systemu informatycznego.

Taki styl wytwarzania oprogramowania wymaga nowego podejścia i indywidualnych cech poszczególnych członków zespołu realizatorów. Przeniesienie akcentów z problemów szczegółowych na problemy całej konstrukcji pozwala nie tylko szybciej wytwarzać, ale również umożliwia rozwiązywanie większych i bardziej skomplikowanych problemów. Dotychczasowe metody wytwarzania powodowały zagubienie się informatyków w licznych szczegółach rozwiązania, co uniemożliwiało zakończenie prac w wymaganym terminie.

Przyczyny nieudanych zastosowań systemów informatycznych są bardzo złożone, a eliminacja każdej z nich zwiększa gwarancje sukcesu. Programowanie obiektowe przyspiesza prace programowe równocześnie zwiększając niezawodność oprogramowania. Przeniesienie akcentów w procesie produkcji oprogramowania na problemy konstrukcyjne pozwala na włączenie przyszłego użytkownika w proces jego wytwarzania. Wspólne konstruowanie kolejnych prototypów systemów pozwala na bieżące modyfikowanie wymagań użytkownika i lepsze dopasowanie ostatecznej wersji systemu informatycznego do jego wymagań.

Podejście obiektowe do wytwarzania oprogramowania przenosi akcenty na fazę projektową, co nie tylko upraszcza proces tworzenia systemu, ale wymaga nieco innych predyspozycji od jego twórców. Prace programistyczne wykonywane w środowisku proceduralnych języków programowania wymagały innego podejścia niż konstrukcja oprogramowania z obiektów (podobnie jak składa się konstrukcję z klocków). Umiejętność oderwania się od szczegółowych rozwiązań algorytmicznych w połączeniu z inwencją konstruktorską pozwala na osiągnięcie nowej jakości wytwarzanych systemów informatycznych.

Programowanie obiektowe, aby było efektywne, musi zapewniać programistom odpowiednio dużą kolekcję gotowych obiektów programowych, proste mechanizmy ich łączenia oraz narzędzia tworzenia własnych obiektów. Programowanie obiektowe to ogólna metoda podejścia do procesu tworzenia oprogramowania, a o jej sukcesie i stosowaniu decyduje otoczenie wspomagające. Narzędzia wspomagające programowanie obiektowe są zwykle omawiane w kolejnych Wiosennych Szkołach PTI.

Życzę Państwu, aby każdy programista mógł wzbogacić swój własny warsztat pracy o nowe narzędzia, które przybliżą go do obiektowego wytwarzania programów.

Zdzisław Szyjewski

Java - programowanie w sieci

Piotr Wolski
SUN Microsystems Poland
Warszawa

Burzliwy rozwój sieci Internet, jaki obserwuje się ostatnio na całym świecie, spowodowany jest w dużym stopniu popularnością serwisu WWW. Możliwości, które oferuje on swoim użytkownikom, prostota obsługi, atrakcyjne opakowanie - wszystko to sprawia, że coraz więcej firm komercyjnych zainteresowanych jest tym tematem. Nic dziwnego zatem, iż każda nowa technologia dotycząca tego zagadnienia wzbudza powszechne zainteresowanie. W ostatnim czasie olbrzymią popularność zdobywa hasło "Java!" - nowy produkt firmy Sun Microsystems. Niektórzy wręcz sugerują, że jest to przełomowy moment dla rozwoju szeroko pojmowanych usług sieciowych. Na czym polega ten fenomen i czym jest Java?

Przez długi czas, na rynku komputerowym obowiązywał tradycyjny model zależności sprzęt - oprogramowanie: typ komputera (procesor) określał system operacyjny, system operacyjny - zbiór dostępnych aplikacji. Wady takiego modelu stały się szczególnie dokuczliwe w momencie, gdy komputery zaczęto łączyć w sieci - a szczególnie w sieć sieci, czyli Internet. Choć podstawowe różnice w sposobie komunikacji dość szybko zlikwidowano, to jednak przez dłuższy czas nie było wygodnego narzędzia wymiany informacji dla niedoświadczonych użytkowników. Przełomowym momentem stało się opracowanie serwisu World Wide Web - w skrócie nazywanego WWW lub W3. Informacja w sieci stała się dostępna praktycznie dla każdego. Wkrótce jednak okazało się, że i ta technologia ma swoje wady. Największą z nich było ograniczenie typu informacji. Tradycyjne przeglądarki internetowe umożliwiały oglądanie jedynie statycznych dokumentów, w niewielkim stopniu wykorzystując dodatkowe możliwości jakie daje komputer. Pojawiały się wprawdzie mniej lub bardziej udane rozszerzenia, posiadały jednak ciągle tę samą wadę: nie przynosiły radykalnej zmiany w zakresie usług serwisu W3.

Mniej więcej pięć lat temu, firma Sun postanowiła opracować produkt, który radykalnie zmieni sposoby dostępu do zasobów sieciowych. Podstawowym celem stało się opracowanie nowej technologii umożliwiającej uniezależnienie warstwy programowej od bazy sprzętowej. Drugim krokiem było zintegrowanie tej technologii z serwisem W3. Osiągnięto w ten sposób uniwersalny mechanizm dystrybucji programów po sieci, z możliwością ich uruchamiania na lokalnym sprzęcie użytkownika, niezależnie od jego typu. Dla zrealizowania tego planu kluczowym zagadnieniem było opracowanie uniwersalnego języka programowania wysokiego poziomu. Najlepiej takiego, który ma rozbudowane możliwości komunikacji sieciowej. Tak powstała JAVA - nowy język obiektowy. Cecha wyróżniająca go spośród innych języków programowania, to heterogeniczność. Programy napisane w Javie działają na każdym systemie wyposażonym w maszynę wirtualną Javy, bez konieczności rekompilacji kodu źródłowego. Dzięki temu możliwe stało się opracowanie mechanizmu łączenia takich programów z zawartością stron HTML - "nośnika" informacji w serwisie WWW. Takie programy, działające za pośrednictwem przeglądarek internetowych, nazwano *Appletami*.

Sposób, w jaki osiągnięto pełną przenośność aplikacji Javy, jest prosty i skuteczny: tekst źródłowy programu jest kompilowany do tzw. *byte-code*'u, interpretowanego przez maszynę wirtualną Javy. Szybkość działania takiego kodu jest oczywiście mniejsza niż języka wewnętrznego maszyny, na ogół jednak zupełnie wystarczająca. Przykładowe czasy wykonywania podstawowych operacji na stacjach roboczych Sun są następujące:

```
new Object      8.4 msek.   (119.000/sek.)
obj.method ()   1.7 msek.   (590.000/sek.)
```

Taka szybkość działania jest w przeważającej mierze przypadków zupełnie wystarczająca. Mimo to istnieje możliwość polepszenia tych wyników. Użytkownicy wymagający największej sprawności mogą skorzystać z dodatkowej możliwości oferowanej przez środowisko Javy - kompilacji do języka wewnętrznego maszyny. Odbywa się to na bieżąco podczas działania programu. Wirtualna maszyna Javy pełni wtedy rolę jednoprzebiegowego inerpretera.

Kompilacja programów Javy do *byte-code*'u, to nie jedyna zaleta tego języka. Równie istotną cechą jest dynamiczny proces konsolidacji realizowany przez środowisko *run-time*. Program napisany w Javie składa się z klas. Po skompilowaniu tekstu źródłowego kod każdej klasy jest umieszczany w oddzielnym pliku. Użytkownik uruchamia aplikację wskazując jedną z klas. Kod dodatkowych fragmentów programu jest dołączany automatycznie w tym momencie, w którym następuje odwołanie do brakującej klasy. Programiści uzyskują dzięki temu jeszcze jedną zaletę - skraca się znacznie cykl tworzenia oprogramowania. Nie tylko ze względu na brak fazy konsolidacji programu. Przede wszystkim dlatego, że zmiana kodu jednej z klas nie wymaga rekompilacji (!) wszystkich pozostałych fragmentów programu, w których występują odwołania do tej klasy. W tradycyjnym języku C++ taka sytuacja spowodowałaby wygenerowanie błędnego kodu aplikacji.

Dynamiczna konsolidacja to cecha cenna również dla potencjalnych twórców nowych bibliotek. Standardowe pakiety klas, zawarte w środowisku Javy, są dołączane do programu na tej samej zasadzie, jak klasy tworzone przez użytkownika. Dzięki temu dodatkowe fragmenty kodu, umieszczone w odpowiednich miejscach istniejącej struktury środowiska, stają się jego integralną częścią i od tej pory są dostępne dla wszystkich aplikacji stworzonych w Javie. Przykładem może tu być system dynamicznych typów danych zaimplementowany w przeglądarce HotJava.

Java a język C++

Twórcy języka Java starali się, aby był on prosty i mógł być szybko przyswojony przez programistów. Zdecydowano zatem, że składnia Javy będzie masymalnie zbliżona do jednego z najpopularniejszych obecnie języków obiektowych - C++. Najprostszy program w Javie może wyglądać, na przykład, tak:

```
class HelloWorld
   {
   static public void main (String args[])
      {
      System.out.println ("Hello World!");
      }
   }
```

Jest to prosta deklaracja klasy HelloWorld zawierającej jedną tylko metodę `main()`. Jej treść, to wywołanie metody `printLn()` dla obiektu `out`. Obiekt ten jest zarazem jedną ze zmiennych klasy `System`. Po uruchomieniu interpretera Javy z argumentem HelloWorld metoda `main()` zostanie wywołana automatycznie i wypisze na standardowym urządzeniu wyjściowym tekst "Hello World!".

Pierwsza wyraźna różnica, to brak zewnętrznych funkcji - zadeklarowanych poza klasami. Język Java to język wysokiego poziomu i język w pełni obiektowy. Z tego powodu usunięto w nim przede wszystkim cechy nieobiektowe, które w C++ ciągną się jako pozostałość po zwykłym C. Ponadto wyłączono wszystkie elementy związane z bazą sprzętową. Między innymi wprowadzono spójny system podstawowych typów danych, rozróżniając liczby całkowite, znaki tekstowe oraz wartości logiczne true/false.

```
typy całkowitoliczbowe:
   byte     8 bitów
   short    16 bitów
   int      32 bity
   long     64 bity

typy zmiennoprzecinkowe:
   float 32 bity
   double   64 bity

typ znakowy:
   char     16 bitów (!)

typ logiczny:
   boolean  true / flase
```

Aby zabezpieczyć się przed nieśmiertelnymi problemami z lokalizacją, typ znakowy `char` określony został jako typ 16-to bitowy reprezentujący znak w standardzie *Unicode*. Usunięto natomiast typ `unsigned`, wprowadzając w zamian operator `>>>` oznaczający logiczne przesunięcie bitów w prawo.

Kolejna różnica w stosunku do C i C++ to realizacja tablic. W Javie są to konkretne obiekty, które można deklarować, inicjalizować i usuwać. Rozmiar tablicy jest znany - określa się go przy inicjalizacji. Każde odwołanie do elementów tablicy jest testowane pod względem zgodności indeksu. Zabezpiecza to programistę przed błędami dostępu do pamięci. Zaimplementowano również metodę `length()` zwracającą liczbę elementów tablicy.

```
Object array[];    // deklaracja tablicy
                   // obiektów Object

array = new Object [5];     // inicjalizacja
                            // tablicy:
                            // pięć elementów
                            // klasy Object

array[3] = new Object ();   // utworzenie
                            // czwartego
                            // elementu
                            // tablicy

len = array.length ();      // długość
                            // tablicy
                            // len == 5
```

Łańcuchy znaków zostały wyodrębnione jako oddzielna klasa. Podobnie jak w przypadku tablic, o długości `String`-u informuje liczba zwracana przez metodę `length()`. Dla wygody programisty stworzono grupę metod i operatorów ułatwiających tworzenie aplikacji operujących na tekstach - między innymi operator konkatenacji + (plus).

Całkowicie usunięto instrukcję `goto` - prowokującą często programistów do używania tylko dlatego, "że jest". Jedynym przypadkiem, w którym jest ona użyteczna, są przerwania wielopoziomowych pętli. Problem ten usunięto rozszerzając możliwości instrukcji `break` i `continue`. Java umożliwia zastosowanie tych instrukcji wraz z etykietą. Określa ona poziom zagłębienia pętli, który należy przerwać lub od którego kontynuować dalsze wykonanie programu.

```
loop: for (i=0; i<10; i++)
         for (j=0; j<5; j++)
            {
            if (i == 3 && j == 4)
               continue loop;
            ...
```

Znaczna część błędów popełnianych przez programistów to błędy związane z pamięcią. Niezainicjowane wskaźniki, zwracanie nigdy nie pobranych obszarów pamięci - to ich typowe przykłady. Twórcy Javy pomyśleli o tym problemie i wprowadzili system automatycznego zarządzania pamięcią, jako integralną część środowiska *run-time*. Przydzielanie nowych obszarów pamięci, zwalnianie nie używanych - te działania są wykonywane automatycznie, bez ingerencji programisty. Zasada działania tego

mechanizmu jest prosta: w momencie przydzielania nowego bloku pamięci są tworzone do niego referencje. Środowisko Javy na bieżąco śledzi stan zasobów pamięciowych - obszary, dla których nie ma już istniejących referencji, są zaznaczane jako wolne. Przy najbliższej okazji są one zwracane do zasobów systemu. Operacja ta odbywa się w tle głównego wątka procesu, przy czym niski priorytet gwarantuje efektywne wykorzystanie procesora. Wprowadzenie takiego automatycznego systemu zarządzania pamięcią pozwoliło na całkowite usunięcie pojęcia wskaźników. Jędyną ich pozostałością w Javie jest zmienna `this` określana automatycznie w każdym obiekcie, pełniąca identyczną rolę jak w C++.

Największy nacisk został położony oczywiście na dopracowanie podstawowego elementu języka obiektowego - klas. Zrezygnowano z wielokrotnej dziedziczności wprowadzając w zamian dwie nowe konstrukcje: klasy abstrakcyjne oraz interfejsy. Klasy abstrakcyjne to rozszerzenie pojęcia funkcji wirtualnych. Mają one zastosowanie jedynie jako klasy bazowe służące do deklaracji zbioru metod i/lub zmiennych dziedziczonych przez klasy potomne. Podobne zastosowanie mają interfejsy. Są to deklaracje grupy metod implementowanych przez inne klasy. Główna różnica pomiędzy tymi tworami polega na tym, że interfejsy mogą jedynie deklarować metody i/lub stałe. Klasy abstrakcyjne nie mają żadnych ograniczeń, poza tym że nie można tworzyć dla nich obiektów.

Na wzór języka Ada, utworzono również jednostkę organizacji klas - pakiety. Ich zastosowanie zwiększa przejrzystość kodu źródłowego i ułatwia posługiwanie się bibliotekami. Z pakietami wiąże się jeszcze jeden nowy element języka - dostępność klas i ich metod/zmiennych. W języku C++ istnieją trzy atrybuty: `private`, `protected` i `public`. Java rozszerza ten zbiór o dodatkowy element oznaczający dostępność metody/zmiennej/klasy wewnątrz pakietu. Jest on przyjmowany domyślnie, gdy nie podano żadnego z trzech wspomnianych powyżej.

Ważną funkcję pełnią również nazwy pakietów. Ich struktura jest hierarchiczna - podobnie jak w sieciowym serwisie DNS - i określa sposób rozmieszczenia zbiorów skompilowanych klas na dysku. Nazwa pakietu stanowi jednocześnie rozszerzenie nazwy klasy i może zostać użyta do wskazania tej klasy z innego pakietu. Przykładowo:

```
Oiltanker.java:
   ==============
   package transport.ships;

   public class Oiltanker
      {
      ...

   Boeing.java:
   ===========
   package transport.planes;

   class Boeing
      {
      ...
```

W powyższym przykładzie pakiet `transport` składa się z dwóch części: `ships` i `planes`. W każdym z pakietów jest zadeklarowana jedna klasa (`Oiltanker`, `Boeing`). Po skompilowaniu zbiorów `Oiltanker.java`, `Boeing.java` wynikowe pliki `Oiltanker.class`, `Boeing.class` są umieszczone w katalogach - odpowiednio - classes/transport/ships i classes/transport/planes. Klasa `Oiltanker` została zadeklarowana z atrybutem `public` - może być więc wykorzystana na zewnątrz poza pakietem. Przy deklaracji klasy `Boeing` nie podano wprost żadnego z atrybutów, co ogranicza jej użycie jedynie do kodu pakietu `transport.planes`.

Wszystkie standardowe biblioteki środowiska Javy są pogrupowane w pakiety, przy czym nie różnią się w żaden sposób od tych tworzonych przez programistę. Klas pochodzących z innych pakietów używać można na trzy sposoby: używając pełnej nazwy klasy, importując wybraną klasę lub importując pakiet zawierający tę klasę.

```
class Cargo
   {
   transport.ships.Oiltanker cargoShip;
   ...

import transport.ships.Oiltanker;
class Cargo
   {
   Oiltanker cargoShip;
   ...

import transport.ships.*   // lub nawet:
                           // import transport.*
class Cargo
   {
   Oiltanker cargoShip;
   ...
```

Język Java ma również wbudowane mechanizmy umożliwiające (i ułatwiające) tworzenie programów wielowątkowych. Podstawowym elementem stworzonym do tego celu jest klasa `Thread`. Powołanie nowego obiektu tej klasy rozwidla działanie programu - podobnie jak funkcja `fork()` w systemie UNIX. Podobieństwo kończy się jednak w tym miejscu, gdyż `fork()` tworzy nowy proces, natomiast `Thread` - nowy wątek tego samego procesu. Klasa `Thread` to jednak tylko jeden z elementów, które składają się na mechanizm wielowątkowości języka Java. Równie istotne jest zapewnienie poprawnego działania programów wykorzystujących tę technikę. Składają się na to dwa elementy: biblioteki *thread-safe* (gwarantujące poprawne działanie w programach wielowątkowych) oraz mechanizmy umożliwiające tworzenie takich bibliotek przez programistę. Java spełnia oba warunki - wszystkie pakiety klas Javy są *thread-safe*, a do składni języka włączono nowe słowo kluczowe `synchronized`. Jest to nowy atrybut, określany dla kilku metod tej samej klasy, gwarantujący ich wykonanie w oddzielnych fragmentach czasu. Na poziomie środowiska *run-time* jest to zapewnione za pomocą tzw. *Monitorów*. Są one tworzone automatycznie dla każdego z obiektów klas zawierających tak oznaczone metody.

Bezpieczeństwo

Środowisko języka Java, jako środowisko do tworzenia oprogramowania działającego w sieci, powinno gwarantować bezpieczeństwo. Zapewnienie stuprocentowego bezpieczeństwa, w przypadku systemów rozproszonych jest bardzo trudne, o ile w ogóle możliwe. Tym niemniej kompilator Javy i jej środowisko zawierają kilka wbudowanych zabezpieczeń znacząco przyczyniających się do zwiększenia tego bezpieczeństwa.

Pierwszy element systemu zabezpieczeń Javy, to sam kompilator. W przeciwieństwie do tradycyjnych kompilatorów C i C++, nie decyduje on tym jaka będzie struktura pamięci podczas działania programu. Decyzje te są przesunięte do fazy wykonywania programu i zależą w dużym stopniu od bazy sprzętowej i konkretnego systemu operacyjnego. Dodatkowo, usunięcie wskaźników ze składni języka uniemożliwia praktycznie programiście niekontrolowane grzebanie po pamięci. Zamiast tego, Java wprowadza rozbudowany system referencji, których powiązanie z fizycznymi obszarami pamięci jest określane dopiero przez interpreter Javy podczas działania programu.

Niezależnie od idei bezpiecznego kompilatora, zabezpieczenia zostały także wprowadzone do środowiska wykonującego gotowe programy. Funkcje takie (między innymi) pełnią dwa moduły: moduł ładujący kod programu oraz moduł testujący jego poprawność.

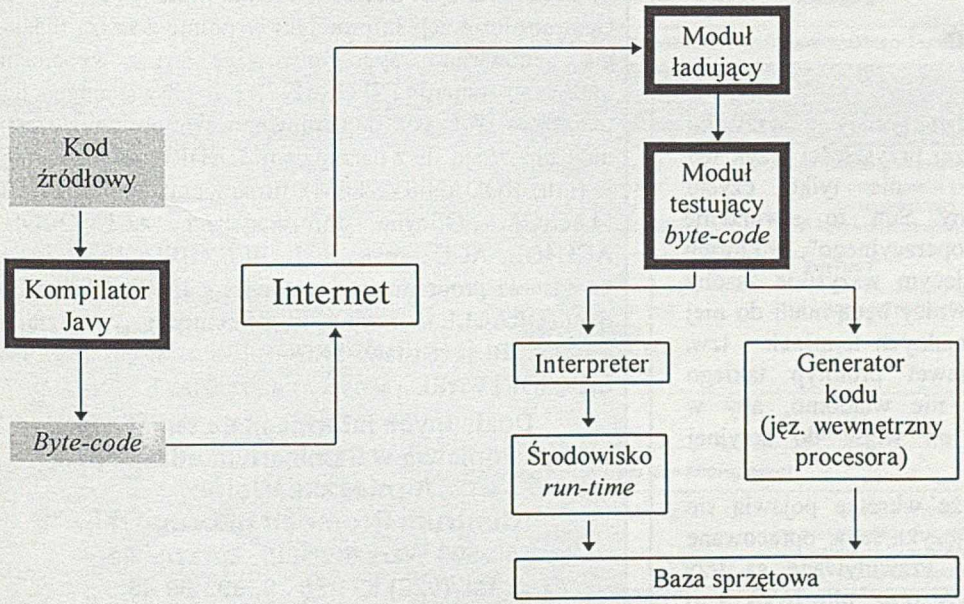
Weryfikacja *byte-code*'u to najważniejszy element systemu zabezpieczeń Javy. Poza sprawdzeniem ogólnej jego poprawności, testy mają na celu zbadanie, czy kod ładowanego programu zachowuje ograniczenia dostępu do pamięci i czy używa zadeklarowanych obiektów zgodnie z ich przeznaczeniem.

Ostatni - uzupełniający element struktury bezpieczeństwa - to moduł ładujący kod poszczególnych klas. Jego główna funkcja, to zabezpieczenie przed tak zwanym *spoofing*'iem. W miarę wykonywania programu są dołączane dynamicznie dodatkowe fragmenty kodu - niekoniecznie (lub raczej: rzadko) z lokalnej maszyny. Moduł ładujący dzieli go na zbiory według źródła pochodzenia, dla których mogą być - i z reguły są - określone różne prawa dostępu do zasobów lokalnej maszyny. Nie jest możliwe, aby klasa zaimportowana z zewnątrz - przez sieć - udawała klasę lokalną.

HotJava

Jako przykład możliwości środowiska programowego Javy firma Sun opracowała nową przeglądarkę internetową: HotJava. Podstawową cechą wyróżniającą ten program spośród grupy innych przeglądarek jest możliwość dołączania i wykonywania programów napisanych w Javie. Dla zwykłego użytkownika objawia się to przez dynamiczną zawartość stron internetowych - dla programistów przez nową strukturę programu i możliwości jego rozbudowy.

Pojęcie dynamicznej zawartości stron W3 stało się już dziś bardzo popularne - niewątpliwie w dużym również stopniu za sprawą firmy Netscape, która szybko podchwyciła pomysł Sun-a i wprowadziła go również do swoich produktów. A pomysł był prosty - ściągane po sieci programy napisane w Javie (*applety*) są uruchamiane z poziomu przeglądarki, a efekty ich działania na bieżąco prezentowane na powierzchni stron W3, obok zwykłych statycznych tekstów i rysunków. Takich programów może być oczywiście na jednej stronie nawet kilka, przy czym działają one jednocześnie. Namiastką tych możliwości są skrypty CGI, jednakże informacje otrzymane za ich pośrednictwem zawsze są statyczne i wymagają czasu procesora serwera WWW, a nie klienta - jak to jest w przypadku *appletów*.



System zabezpieczeń języka Java

Jak już wspomniano HotJava, to również nowa struktura programu. Jest to struktura otwarta: przeglądarka zna pewien określony zbiór formatów danych i protokołów sieciowych, umożliwia jednak w prosty sposób dołączanie nowych. Co istotne, odbywa się to bez ingerencji użytkownika.

Przykładowo, jeżeli podczas ładowania zawartości strony W3, HotJava napotka na element nieznanego typu (np. rysunek w nowym formacie graficznym), automatycznie wyśle zapytanie do serwera o program do jego obsługi. Po otrzymaniu żądanego kodu dołączy go do programu i za jego pomocą odczyta otrzymane wcześniej dane. Co więcej - gdy ponownie napotka taki obiekt, będzie już umiała się nim posługiwać, tak jak każdym innym ze standardowego jej zestawu.

Identyczny mechanizm został zaimplementowany dla protokołów sieciowych. Wybór połączenia URL zawierającego nazwę nowego - nieznanego przeglądarce - protokołu uruchamia podobną sekwencję: załadowanie programu obsługi i nawiązanie połączenia za pomocą otrzymanego kodu.

Możliwość uruchamiania kodu otrzymanego za pośrednictwem sieci bez ingerencji (i wiedzy) użytkownika stwarza niestety nowe zagrożenia dla lokalnego systemu. W programie HotJava wprowadzono zatem trójstopniowy system zabezpieczeń:

1) możliwość całkowitego wyłączenia opcji ładowania *appletów*,
2) możliwość określenia grupy domen i/lub pojedynczych "zaufanych" systemów, z których *applety* mogą być ładowane,
3) możliwość określenia dla *appletów* dostępu do zasobów dyskowych systemu użytkownika (oddzielnie odczyt i zapis).

Konfiguracja tych parametrów może być również przeprowadzona przez administratora dla całej lokalnej sieci. Użytkownicy mają wtedy możliwość jedynie zaostrzenia narzuconych odgórnie zabezpieczeń.

Najbliższa przyszłość

Java to obecnie temat bardzo gorący i wszystko wskazuje na to, że w najbliższej przyszłości język ten znajdzie wiele zastosowań - nie tylko czysto informatycznych. Wizja firmy Sun to stworzenie nowego "sieciowego systemu operacyjnego", w którym Java będzie elementem spajającym wszystkie zasoby rozproszone po sieci a użytkownicy będą mieli do niej dostęp za pośrednictwem specjalnych terminali - tzw. Java-device. Powstał już nawet prototyp takiego urządzenia, na razie jednak nie wiadomo, aby w najbliższej przyszłości miał on wejść do seryjnej produkcji.

Wiadomo już natomiast, że wkrótce pojawią się specjalizowane procesory dla języka Java, opracowane oczywiście przez firmę Sun. Przwidywane są trzy modele: picoJAVA, microJAVA oraz ultraJAVA - o różnej cenie i różnych obszarach zastosowań. PicoJAVA - model najprostszy, w cenie nie przekraczającej $25 - oczekiwany jest już w połowie tego roku i jego przeznaczeniem są zastosowania najprostsze: telefony komórkowe, drukarki i inne urządzenia zewnętrzne. MicroJAVA oraz ultraJAVA - to modele droższe, spodziewane odpowiednio na początku i pod koniec przyszłego roku - przeznaczone do bardziej wymagających zastosowań (m.in. do grafiki trójwymiarowej i multimediów).

Java wchodzi stopniowo na nowe obszary zastosowań *software*'owych. Opublikowana została już specyfikacja interfejsu dla Javy, umożliwiającego ujednolicony dostęp do różnych baz danych. Firmy uczestniczące w tym projekcie (m.in. Informix, Oracle, Gupta) pracują już nad stworzeniem kodu współpracującego z ich własnymi bazami.

Ważą się obecnie dalsze losy języka VRML. W listopadzie zeszłego roku jego twórcy (VRML Architecture Group) wybrali Javę jako język odniesienia przy tworzeniu aplikacji oraz nowych standardów VRML. Już w lutym tego roku firmy Silicon Graphics i Netscape opracowały propozycję nowego standardu tego języka - "*Moving Worlds*". Kluczową rolę w ożywieniu - martwych do tej pory - obiektów odegra oczywiście Java. Ponad pięćdziesiąt innych firm zapowiedziało wsparcie dla tego projektu.

Gdzie można znaleźć więcej informacji na temat Javy? Główne źródło, to Sun-owski serwer WWW: http://java.sun.com. Jest tam dostępna między innymi dokumentacja do języka wraz z opisem bibliotek. Są również informacje o tym, jak i skąd zdobyć samą Javę.

COMNET

WARSZAWA '96

EuroInfo'96 - Poland

II Międzynarodowa Konferencja i Wystawa Technik Telekomunikacyjnych i Sieciowych ComNet Warszawa 19-21 czerwca 1996 Pałac Kultury i Nauki

Skrócony Program Konferencji ComNet Warszawa' 96

Środa 19 czerwca 1996	Wykład 1:Technologia ATMu i jej zastosowanie (Część I i II)	Wykład 2: Wprowadzenie do technologii Bezprzewodowych (Część I i II)	Wykład 3: Zarządzanie Sieciami Publicznymi i Prywatnymi (Część I i II)	Wykład 4: Technologie Przyszłości(Część I i II)
9.00 - 17.30	Jill Kaufman, Roger Kosak, Stephan Horvath, Doug O'Leary, Chuck Rush, (ATM Forum)	prof. Marian Dąbrowski (PW, MŁ), Johan Wickman (Telia Research AB)	Prof. Andrzej Jajszczyk (EFP)	Dr Jens Bodenkamp (Intel GmbH) Lilian Goleniowski (The LIDO Organization)
Czwartek 20 czerwca 1996	Kierunek 1: Rozwiązania Sieci Korporacyjnych	Kierunek 2: Technologie i Usługi Telekomunikacyjne	Kierunek 3: Globalna Infrastruktura Informacyjna	Kierunek 4: EuroInfo'96 Poland
9.00 - 10.30	Platformy zarządzania sieciami (przewodniczący prof. Janusz Filipiak, AGH)	Usługi bezprzewodowe: Systemy Satelitarne (przewodniczący prof Daniel Bem, PWr)	Dostęp do Internetu: Technologia, Koszty i Dostawcy (przewodniczący Jerzy Goraziński, Inter-net Polska)	Media: Elektroniczne zasoby informacyjne dla prasy, radia i telewizji
10.30 - 10.45	*Przerwa I*			
10.45 - 12.15	Sieci Korporacyjne: Koszty i Korzyści (dr Guenter Honisch, Cisco)	PCN (przewod. prof. Marian Dąbrowski)	Koszty i Taryfy Dostępu do Internetu (przewodniczący prof. Tomasz Hofmokl,) NASK	Media: Elektroniczne zasoby informacyjne dla prasy, radia i telewizji
12.15 - 13.00	*Lunch*			
13.00 - 13.45	*Keynote 1: Stephan Horvath Prezydent ATM Forum EMAC: ATM w Europie*			
14.00 - 15.30	Sieci Korporacyjne: Koszty i Korzyści (przewodniczący dr Jerzy Kalinowski, Price Waterhouse)	Usługi Multimedialne: Dostępność, Korzyści i Koszty (przewodniczący dr Krzysztof Amborski, TP SA)	Prawne Aspekty Internetu (przewodniczący dr Andrzej Adamski, UMK)	Administracja Państwowa: Strategia zastosowań infostrad w administracji państwowej
15.30 - 15.45	*Przerwa II*			
16.00 - 17.30	*Sesja Plenarna: Ewolucja Połączeń Sieciowych, (przewodniczący Leszek Stachów,Booz, Allen & Hamilton Int.)*			
Piątek 21 czerwca 1996	Kierunek 1 Rozwiązania Sieci Korporacyjnych	Kierunek 2 Technologie i Usługi Telekomunikacyjne	Kierunek 3 Globalna Infrastruktura Informacyjna	Kierunek 4: EuroInfo'96 Poland
9.00 - 10.30	Multimedia: - - Zastosowania i Korzyści (przewodniczący dr Marek Średniawa, PW)	Nowe usługi oferowane przez systemy ruchome (przewodniczący prof. Witold Hołubowicz, EFP)	Bezpieczeństwo Internetu: Jak chronić swoją sieć (przewodniczący dr Ryszard Kossowski, PW)	Nauka: Internet w nauce i dla nauki
10.30 - 10.45	*Przerwa I*			
10.45 - 12.15	Systemy okablowania strukturalnego (przewodniczący red. Bronisław Piwowar, IDG NetWorld)	Ewolucja dostępu do sieci (przewodniczący prof. Henryk Lasota, PG)	Stare i Nowe narzędzia do WWW	Biznes: Zasoby elektronicznej informacji biznesowej
12.15 - 13.00	*Lunch*			
13.00 - 13.45	*Keynote 2: John Gage (Sun Microsystems)*			
14.00 - 15.30	Architektura Klient/Serwer (przewodniczący Leszek Wroński, Price Waterhouse)	Sieci dostępu bezprzewodowego (przewodniczący dyr. Wojciech Hałka, MŁ)	Serwery WWW: Jak przedstawiać swoją informację w Internecie (przewodniczący Juliusz Donajski)	Internet - WWW: Surfing po WWW, zasoby i ich oceny
15.30 - 17.00	*Szczęśliwe Godziny*			

Szczegółowe informacje i zgłoszenia: IDG Expo ul. Tytoniowa 20; 04-228 Warszawa tel. (0-22) 613-25-44 fax: 15-44-95

Otwarta droga do programowania obiektowego, czyli OpenROAD/Windows4GL 3.0

OpenROAD (*Open Rapid Object Application Development*) jest środowiskiem służącym do programowania obiektowego. Jest pierwszym językiem czwartej generacji (4GL) umożliwiającym opracowywanie graficznych aplikacji klient-serwer. Nie jest więc tylko środowiskiem graficznym użytkownika (GUI) ani też tylko językiem programowania 4GL, lecz formą, łączącą funkcje i zalety obu tych narzędzi.

Narzędzia programowania OpenROAD

OpenROAD/Windows4GL jako język programowania należy do rodziny narzędzi OpenROAD, zawierającej wszystkie niezbędne elementy potrzebne do budowy prototypu aplikacji, jej wdrażania, testowania i eksploatacji. Aplikacje te mogą pracować na różnych platformach sprzętowych, bazach danych i korzystać z różnego oprogramowania do zarządzania okienkami. Szczególnie zaś istotne jest to, że mogą to być równocześnie różne środowiska programowo-sprzętowe. W zakresie sprzętowym mogą to być komputery od PC do RISC-owych stacji roboczych.

Język programowania OpenROAD/Windows4GL umożliwia tworzenie nawet bardzo złożonych aplikacji dla klienta w architekturze klient-serwer. Można w nich wykorzystywać takie typowe elementy architektury klient-serwer jak przetwarzanie transakcji, obrazowanie graficzne, przetwarzanie wsadowe, wspomaganie decyzji i praca w czasie rzeczywistym. W praktyce oznacza to, że można wbudować do aplikacji obsługę telefonu, obsługę klientów oraz sterowanie procesami.

OpenROAD/Windows4GL został opracowany z myślą o tworzeniu aplikacji graficznych. Szczególnie ostatnio klienci zaczęli się domagać środowiska graficznego, które ich zdaniem powinno znacznie ułatwić pracę użytkownikom oraz skrócić czas i uciążliwość szkoleń. Liczba okienek jakie można zaprojektować w OpenROAD/Windows4GL przekracza 500, co powinno zaspokoić nawet najbardziej wymagających pod tym względem klientów.

Edytory wizualne

OpenROAD/Windows4GL stwarza programistom komfortowe środowisko pracy, dzięki zestawowi edytorów wizualnych. Na przykład, korzystając z takiego edytora można za pomocą myszki wprowadzać obiekty do programu. Elementy ekranowe mogą być grupowane według dowolnego kryterium, określonego przez programistę. Na ekranie można zarządzać mapami bitowymi, przyciskami, tekstem i każdym innym elementem tam występującym. Jako elementy ekranowe można wykorzystać także pola z baz danych, co jest szczególnie przydatne w aplikacjach obsługujących bazy danych.

Programowanie obiektowe umożliwia bezpośrednią współpracę z użytkownikami, którzy mogą zgłaszać swoje propozycje dotyczące najwygodniejszego dla nich interfejsu graficznego. Po stworzeniu graficznego interfejsu ekranowego, programista może opracować w języku OpenROAD/Windows4GL procedury, obsługujące operacje wykonywane na ekranie w ramach dostępnych mechanizmów graficznego interfejsu użytkownika.

Dla każdego z obiektów prezentowanych na ekranie można dobrać kolory, odcienie, czcionki, linie, wielkości punktów. Można też tworzyć dla nich szablony i nadawać wspólne cechy wybranym grupom obiektów. Opcja ta znacznie skraca czas przygotowywania aplikacji, zmniejszając przy tym ilość żmudnej pracy.

Interaktywny program testujący

W OpenROAD/Windows4GL znajduje się dialogowy program testujący (ang. *debugger*) na poziomie kodu źródłowego. Program ten ściśle współdziała z całym środowiskiem programistycznym. Na przykład, nie stanowi to różnicy dla programisty czy aplikacja jest uruchamiana równocześnie z programem testującym, czy też bez niego. Ponieważ program testujący korzysta z tego samego środowiska graficznego co OpenROAD/Windows4GL, programista nie musi uczyć się i przyzwyczajać do pracy z różnymi interfejsami przy opracowywaniu, testowaniu i uruchamianiu aplikacji.

Pracując z programem testującym programista śledzi działanie aplikacji, obserwując na ekranie jej aktualny kod źródłowy. Aplikacja może być oglądana i modyfikowana na bieżąco na poziomie kodu źródłowego, stosów zdarzeń i zmiennych. Program testujący umożliwia nawet manipulację zdarzeniami, śledzenie wsteczne stosu, określanie cech obiektów, ich modernizowanie i śledzenie wprowadzanych danych podczas całej sesji testowania. Program testujący umożliwia wyspecyfikowanie punktów kontrolnych i zdarzeń. W każdym momencie pracy aplikacji można sprawdzić aktualny stan wszystkich elementów, które są aktualnie z nią związane. Dzięki temu można łatwo zidentyfikować i poprawić lub usunąć nieprawidłowe elementy aplikacji.

Biblioteki klas obiektów

OpenROAD/Windows4GL jest wyposażony w ponad 120 systemowych klas obiektów i ponad 1200 funkcji, zaprojektowanych specjalnie w ten sposób, aby ułatwić

projektowanie i oprogramowanie graficznych aplikacji klient-serwer. Definicje klas są zintegrowane ze środowiskiem programowania i językiem 4GL. Takie rozwiązanie daje programistom możliwość korzystania z udogodnień, jakie niesie ze sobą technika obiektowa, nie zmuszając ich przy okazji do zapoznania się z możliwościami nowoczesnych technik programowania obiektowego. Programiści mogą zdefiniować własne klasy jako proste struktury lub jako w pełni obiektowe klasy. Klasy zdefiniowane przez użytkownika są wyposażone, tak jak klasy dołączane standardowo, w własności dziedziczenia, hermetyzacji oraz polimorfizmu.

Szablony

Wiele składników aplikacji jest podobnych do siebie i dlatego wykorzystanie szablonów znacznie skraca czas przygotowania aplikacji. OpenROAD/Windows4GL umożliwia tworzenie szablonów, których wykorzystanie ułatwia tworzenie elementów aplikacji opartych na standardowych strukturach. Są to tak zwane szablony ramek i pól. Szablony ramek służą do definiowania formularzy początkowych, menu i elementów sterujących oraz do tworzenia okienek. Szablony pól określają cechy pól początkowych, elementów sterujących, a 4GL określa kolumny lub scalone elementy sterujące.

Proste zintegrowane środowisko

W odróżnieniu od narzędzi, które były początkowo zaprojektowane dla terminali znakowych, OpenROAD/Windows4GL był od początku zaprojektowany z myślą o graficznym interfejsie użytkownika. 4GL wykorzystuje również możliwości charakterystyczne dla aplikacji graficznych. OpenROAD/Windows4GL jest szczególnie przydatny w środowisku sprzętowo-programowym składającym się z produktów wielu dostawców, gdyż w środowisku klient-serwer znajduje się po stronie klienta. Aplikacja napisana w OpenROAD/Windows4GL może być rozprowadzona do różnego typu okienkowych interfejsów zarządzających i na wielu różnych platformach, bez potrzeby wprowadzania jakichkolwiek modyfikacji kodu na różnych stanowiskach. Cecha przenośności jaką ma OpenROAD/Windows4GL pozwala na zintegrowane wykorzystanie wszystkich zasobów informatycznych firmy.

Automatyczne dopasowanie do środowiska okienkowego

OpenROAD/Windows4GL jest dostępny na różnych platformach sprzętowo-programowych i okienkowych systemach zarządzających. Aplikacja została napisana w jednym środowisku, a uruchomiona na innym automatycznie przyjmuje wygląd i cechy typowe dla tego środowiska. Programista nie musi się martwić, czy któryś z elementów ekranowych jest dostępny na innej platformie niż ta, na jakiej tworzone jest oprogramowanie. Spośród wielu okienkowych systemów zarządzających, na jakich mogą pracować aplikacje tworzone przy pomocy OpenROAD/Windows4GL, wymieńmy choćby Microsoft Windows i OSF/Motif.

Aplikacje składają się z bloków wydarzeń. Przed wystąpieniem danego wydarzenia jest wykonywana odpowiednia sekwencja kodów. OpenROAD/Windows4GL jest wyposażony w wiele kodów dla bloków wydarzeń, często wykorzystywanych w aplikacjach graficznych. Do takich typowych zdarzeń należy, na przykład, kliknięcie myszki, zmiana rozmiarów okienka, czy potwierdzenie odebrania informacji otrzymanego z innego okienka lub źródła zewnętrznego. Programowanie oparte na operowaniu zdarzeniami jest podstawą aplikacji graficznych, w których każdy użytkownik wybiera własną ścieżkę komunikacji z programem.

Wiele aktywnych okienek

Aplikacje pisane w OpenROAD/Windows4GL mogą przesyłać informacje pomiędzy wieloma okienkami, którymi mogą być dane, jak i wyniki operacji na bazach danych. Na przykład, aplikacja składania zamówień może mieć w jednym okienku informacje główne a w drugim wykaz zamówień. Użytkownik może swobodnie wymieniać dane między okienkami, a przy zmianie danych w jednym, odpowiednie dane zmienią się automatycznie w drugim.

Dodatkowo, możliwość pracy wielozadaniowej pozwala relacyjnej bazie danych OpenINGRES/Intelligent Database inicjować komunikację z aplikacją. Na przykład, program uaktualniający dane w bazie danych może wysłać informację do aplikacji użytkownika o tym, że zakończył swoją operację. W takim przypadku ekran klienta zmieni swój obraz, przedstawiając najbardziej aktualny stan danych w bazie.

Aplikacje dynamiczne

Aplikacje pisane w OpenROAD/Windows4GL mogą dynamicznie zmieniać wygląd i sposób funkcjonowania w czasie pracy. Dzięki obiektowym własnościom 4GL programista może definiować nowe pola, klasy użytkownika, wyrażenia 4GL. Formularze mogą być tworzone za każdym razem po uruchomieniu programu. Można zaprogramować takie bloki wydarzeń ogólnych, które są inicjowane dynamiczne w wyniku działania aplikacji, wystąpienia określonej wartości zmiennej lub wystartowane przez użytkownika.

Inne cechy integracji

OpenROAD/Windows4GL integruje język SQL ze składnią 4GL, tak więc programiści mogą budować kody dla

dokończenie na s. 23 →

Przegląd cech obiektowych języka PROGRESS 4GL w oprogramowaniu relacyjnych baz danych

Marek Prokop
CSBI S.A.

PROGRESS 4GL - charakterystyka języka

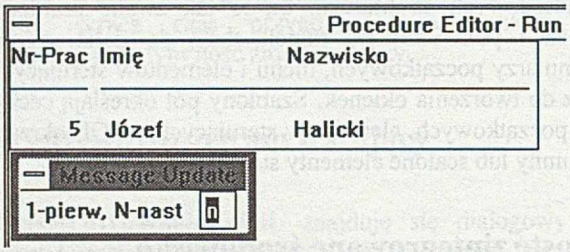
Język PROGRESS 4GL, w jaki został wyposażony system zarządzania relacyjnymi bazami danych Progress - produkt amerykańskiej firmy Progress Software Corporation, przeznaczony do obsługi baz danych działających w środowiskach DOS/Novell/MS-Windows/UNIX/Motif - w wersji 7.3 oraz 8 posiada szereg konstrukcji umożliwiających praktyczne stosowanie technik obiektowych. W niniejszym artykule omówione zostaną właśnie możliwości obiektowe zawarte w tym języku. Na pierwszy rzut oka PROGRESS 4GL jest klasycznym, proceduralnym językiem programowania wysokiego poziomu, specjalizowanym do obsługi baz danych i przenośnym pomiędzy różnymi systemami operacyjnymi (DOS, UNIX) oraz środowiskami interfejsu użytkownika (znakowe, MS-Windows, Motif). Autorzy wzbogacili go jednak w dodatkowe środki, które pozwalają tworzyć programy w trzech podstawowych stylach programowania: klasycznym, zdarzeniowym i obiektowym oraz ich mieszankach. Styl zdarzeniowy i obiektowy wykorzystują szereg cech języka uważanych za obiektowe. Ponadto wersja 8 języka została wyposażona w opracowany przez firmę Progress Software standard ADM (*Application Developement Methodology*). Jest to zestaw biblioteczny uogólnionych obiektów - komponentów aplikacji - tzw. *SmartObjects* wraz z technologią ich tworzenia i wykorzystywania. Na technologię tę składają się kody źródłowe obiektów, bogata dokumentacja opisująca rozwiązania przyjęte w standardzie oraz rozbudowany w stosunku do wersji 7 program narzędziowy UIB (*User Interface Builder*) umożliwiający składanie aplikacji z wcieleń wcześniej zbudowanych uogólnionych obiektów *SmartObjects*. Z powodu tych rozszerzeń wersja 8 języka została opatrzona przydomkiem OO4GL (*Object Oriented 4th Generation Language*).

Styl klasyczny

W stylu klasycznym (*procedural approach*) - sterowanie przebiega tradycyjnie, tj. z góry na dół - instrukcja po ninstrukcji. Poniżej pokazano próbkę programu zgodnego z tym stylem, realizującego przeglądanie rekordów:

```
/* nav1.p */
DEF VAR odp AS CHAR FORMAT "x".
FIND FIRST pracownik.
REPEAT:
  DISP prac-nr imie nazwisko.
  MESSAGE "1-pierw, N-nast" UPDATE odp.
  IF odp = "1" THEN FIND FIRST pracownik.
  IF odp = "N" THEN FIND NEXT  pracownik.
END.
```

Rys.1. Program nav1.p podczas wykonania



Styl taki stosuje się we wszelkiego zadaniach wymagających typowego sekwencyjnego przetwarzania zawartości bazy danych, najczęściej bez większej interakcji z użytkownikiem. W zasadzie jest to jedyne możliwe podejście stosowane w starszej wersji 6 języka PROGRESS.

Styl zdarzeniowy

W stylu zdarzeniowym (*event-driven programming*) - konkretny przebieg sterowania determinują zdarzenia (*events*) zachodzące w otoczeniu systemu. Podejście takie jest możliwe dzięki temu, że system Progress został wyposażony w mechanizmy automatycznego rozpoznawania zdarzeń i reagowania na nie. Zdarzenia mogą dotyczyć elementów interfejsu użytkownika lub być związane z konkretnymi operacjami wykonywanymi na bazie danych. Programista definiuje automatyczne reakcje na takie zdarzenia przy pomocy bloków trygerów. Procedura realizująca styl zdarzeniowy składa się zwykle z kilku funkcjonalnie rozdzielonych sekcji kodu (w nawiasach podano nazwy instrukcji odpowiadającym poszczególnym akcjom):

- sekcja definiująca obiekty ekranowe (DEF ..., fraza VIEW-AS),
- sekcja definiująca trygery (ON *zdarzenie* OF *obiekt* DO: ...END.) - czyli odpowiednie reakcje na

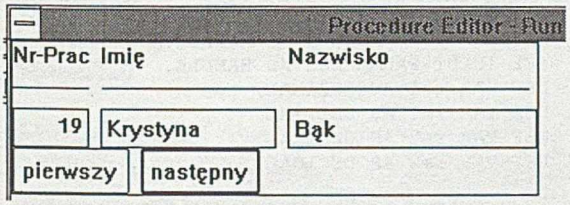
wyróżnione pary: (*zdarzenie,obiekt*), gdzie *obiektem* może być element interfejsu lub rekord (ewentualnie pole rekordu) z konkretnej tabeli bazy, natomiast *zdarzenie* jest nazwą, reprezentującą w języku konkretne zdarzenie zachodzące w otoczeniu systemu

- sekcja główna - gdzie wykonuje się kreację, wyświetlenie (DISPLAY...) i uaktywnienie (ENABLE...) obiektów wykorzystywanych w programie, a następnie przełącza się system w stan oczekiwania na zdarzenia, ich rozpoznawania i automatycznej obsługi (WAIT-FOR ...)

Opcjonalnie może wystąpić dodatkowa sekcja zawierająca definicje tzw. podprocedur wewnętrznych (*internal procedures*). Styl ten jest najwygodniejszą metodą stosowaną przy oprogramowaniu interfejsu użytkownika. Poniżej przedstawiono próbkę programu realizującego przeglądanie rekordów wg stylu zdarzeniowego:

```
/* nav2.p */
/*** przyciski sterujące ***/
DEF BUTTON btn-pierw LABEL "pierwszy".
DEF BUTTON btn-nast  LABEL "następny".
/*** ramki ***/
DEF FRAME f1
  pracownik.prac-nr imie nazwisko
  SKIP btn-pierw btn-nast.
/*** trygery ***/
ON CHOOSE OF btn-pierw DO:
  FIND FIRST pracownik.
  DISP prac-nr imie nazwisko WITH FRAME f1.
END.
ON CHOOSE OF btn-nast DO:
  FIND NEXT pracownik.
  DISP prac-nr imie nazwisko WITH FRAME f1.
END.
/*** main ***/
ENABLE ALL WITH FRAME f1.
WAIT-FOR WINDOW-CLOSE OF DEFAULT-WINDOW.
```

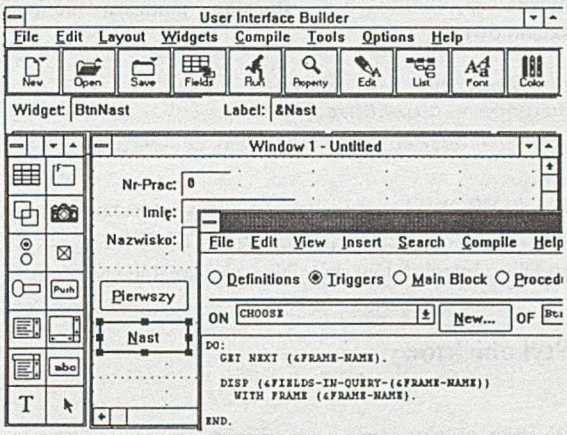
Rys.2. Program nav2.p podczas wykonania



Bardziej rozbudowane programy obsługujące interfejs użytkownika najwygodniej w Progressie tworzyć przy pomocy generatora UIB (*User Interface Builder*). UIB jest wygodnym programem narzędziowym przy pomocy którego projektant zapełnia makietę projektowanego okna gotowymi elementami interfejsu (obiektami ekranowymi) wybieranymi myszą bezpośrednio z palety dostępnych obiektów (tzw. programowanie wizualne).

Dla każdego wprowadzonego obiektu ekranowego przy pomocy edytora charakterystyki obiektu można określić jego atrybuty (np. rozmiar, kolor, etykietę itp.) oraz zdefiniować reakcję na typowe zdarzenia związane z obiektem - w postaci bloku trygera. Możliwe jest także jednoczesne projektowanie kilku dynamicznie określanych wariantów (*layouts*) tego samego okna, w szczególności wariantu dla obsługi interfejsu znakowego. Z poziomu UIB testuje się wykonanie zaprojektowanego programu. Kod procedury w języku PROGRESS 4GL generowany na wyjściu przez UIB odpowiada stylowi zdarzeniowemu i składa się z szeregu sekcji, z których projektant może mieć bezpośredni wpływ na cztery: Definitions, Control Triggers, Main Block i Internal Procedures. UIB pozwala również zapamiętywać w odrębnych plikach fragmenty projektowanego w sposób ogólny interfejsu (tj. z wykorzystaniem definicji wbudowanego preprocesora) lub ogólne makiety całych okien (*templates*). Zaprojektowane w ten sposób ogólne fragmenty mogą być następnie wykorzystywane w wielu innych oknach jako włączane elementy standardowe (*reusability*).

Rys.3. Definiowanie kodu trygera dla przycisku BtnNast w trakcie sesji z programem UIB



Wspomniane wcześniej obiekty ekranowe (*widgets*) stanowią wbudowaną w język liczną grupę gotowych elementów, z których programista jak klocków składa interfejs użytkownika. Można je podzielić na trzy kategorie:

- obiekty organizacyjne - WINDOW, DIALOG-BOX, FRAME
- obiekty do prezentacji akcji - BUTTON, MENU itp.
- obiekty do prezentacji wartości - FILL-IN, COMBO-BOX, BROWSE, SELECTION-LIST itp.

Dodatkowo w języku występują wbudowane obiekty systemowe (*pseudo widgets*) oferujące określone zasoby systemu oraz określające jego stan. Do grupy tej można zaliczyć takie obiekty jak: SESSION, ERROR-STATUS czy CLIPBOARD. Z każdym obiektem wbudowanym w język wiąże się ustalony zestaw atrybutów określających charakterystykę obiektu oraz zbiór metod

umożliwiających różnego rodzaju specjalistyczne manipulacje wykonywane na rzecz danego obiektu. Poniżej dla przykładu podano wybrane atrybuty i metody związane z obiektem BROWSE oraz odpowiadającą im notację składniową.

Wybrane atrybuty obiektu BROWSE

```
BGCOLOR      COLUMN
DOWN       FGCOLOR
FONT       POPUP-MENU
FRAM       PREV-SIBLING
FRAME-X      ROW
FRAME-Y      SENSITIVE
HANDLE       HEIGHT-CHARS
HELP       HIDDEN
```

Wybrane metody dla obiektu BROWSE:

```
DELETE-SELECTED-ROW( )
DESELECT-SELECTED-ROW( )
FETCH-SELECTED-ROW( )
IS-ROW-SELECTED( )
MOVE-TO-BOTTOM( )
MOVE-TO-TOP( )
REFRESH( )
SCROLL-TO-CURRENT-ROW( )
SCROLL-TO-SELECTED-ROW( )
SELECT-ROW( )
```

Do atrybutów i metod obiektów wbudowanych w języku można się odwoływać stosując poniższą notację składniową:

```
obiekt:atrybut = wartość
zmienna = obiekt:atrybut
wynik = obiekt:metoda(parametry-metody)
```

Obiekt BROWSE jest wykorzystywany jako standardowy obiekt do prezentacji zapytań (QUERY), czyli podzbiorów rekordów spełniających określone warunki.

Styl obiektowy

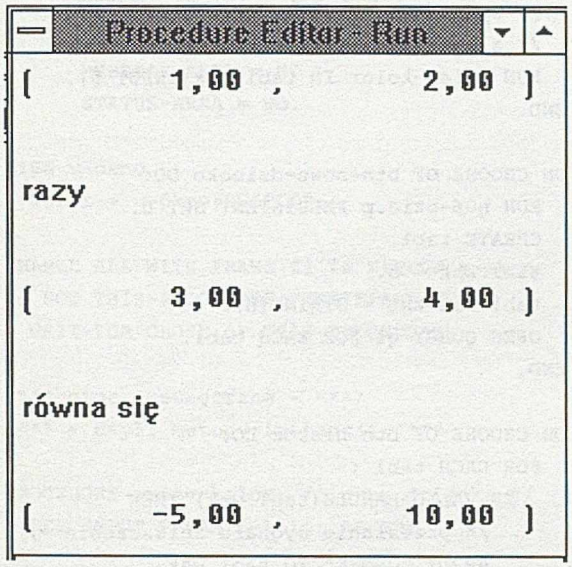
W stylu obiektowym - sterowanie jest determinowane przez komunikaty (sygnały lub wywołania metod) wymieniane pomiędzy obiektami, ciała procedur traktowane są podobnie jak definicje klas obiektów w językach obiektowych. Język PROGRESS 4GL nie posiada jednostki składniowej służącej do definiowania klasy ani nawet pojęcia typu obiektowego. Jednak dzięki specjalnej postaci instrukcji wywołania procedury - RUN ... PERSISTENT... - możliwe jest traktowanie pliku źródłowego procedury jako definicji klasy. W takim podejściu - zmienne lokalne odpowiadają atrybutom, natomiast procedury wewnętrzne metodom klasy. Wywołanie RUN...PERSISTENT... ładuje do pamięci instancję (wcielenie) procedury, która nie ginie (a zatem istnieje jako niezależny obiekt) po zakończniu wywołania. Instancja ta jest obiektem zawierającym dane lokalne oraz identyfikowanym przez konkretny wskaźnik typu HANDLE. Poniżej przedstawiono realizację"podręcznikowego" przykładu definicji klasy liczb zespolonych w języku Progress:

```
/* complex.p */
/* parametry */
DEF INPUT PARAM pRe AS DECIMAL.
DEF INPUT PARAM pIm AS DECIMAL.

/* definicja klasy liczb zespolonych */
/* atrybuty - zmienne lokalne */
DEF VAR re AS DECIM. /* cz. rzeczywista */
DEF VAR im AS DECIM. /* cz. urojona     */

/* inicjalizacja */
re = pRe.
im = pIm.

/* metody  - podprocedury wewnętrzne*/
PROCEDURE DispCx:
  DISP "(" re "," im ")" WITH NO-LABELS NO-
BOX.
END.

PROCEDURE GetCx:
  DEF OUTPUT PARAM pRe AS DECIMAL.
  DEF OUTPUT PARAM pIm AS DECIMAL.

  pRe = re.
  pIm = im.
END.

PROCEDURE SetCx:
  DEF INPUT PARAM pRe AS DECIMAL.
  DEF INPUT PARAM pIm AS DECIMAL.

  re = pRe.
  im = pIm.
END.

PROCEDURE AddCx:       /* c0 = c1 + c2 */
  DEF INPUT PARAM hc0 AS HANDLE.
  DEF INPUT PARAM hc1 AS HANDLE.
  DEF INPUT PARAM hc2 AS HANDLE.
  DEF VAR re1 AS DECIMAL.
  DEF VAR im1 AS DECIMAL.
  DEF VAR re2 AS DECIMAL.
  DEF VAR im2 AS DECIMAL.

IF VALID-HANDLE(hc0) AND
     VALID-HANDLE(hc1) AND
     VALID-HANDLE(hc2) THEN DO:
      RUN GetCx IN hc1(OUTPUT re1, OUTPUT
im1).
      RUN GetCx IN hc2(OUTPUT re2, OUTPUT
im2).
    RUN SetCx IN hc0(re1 + re2, im1 + im2).
  END.
END.

PROCEDURE MulCx:       /* c0 = c1 * c2 */
  DEF INPUT PARAM hc0 AS HANDLE.
  DEF INPUT PARAM hc1 AS HANDLE.
  DEF INPUT PARAM hc2 AS HANDLE.
  DEF VAR re1 AS DECIMAL.
  DEF VAR im1 AS DECIMAL.
  DEF VAR re2 AS DECIMAL.
  DEF VAR im2 AS DECIMAL.

IF VALID-HANDLE(hc0) AND
     VALID-HANDLE(hc1) AND
     VALID-HANDLE(hc2) THEN DO:
      RUN GetCx IN hc1(OUTPUT re1, OUTPUT
im1).
      RUN GetCx IN hc2(OUTPUT re2, OUTPUT
im2).
    RUN SetCx IN hc0(re1 * re2 - im1 * im2,
                     im1 * re2 + im2 *
re1).
  END.
END.
```

Poniżej przedstawiono przykład generujący kilka instancji klasy complex.p oraz efekt jego wykonania.

```
/* usecx.p */
DEF VAR hc0 AS HANDLE.
DEF VAR hc1 AS HANDLE.
RUN complex.p PERSISTENT SET hc0 (INPUT 1,
INPUT 2).
RUN DispCx IN hc0.
DISPLAY "razy" WITH FRAME f1 NO-BOX.
RUN complex.p PERSISTENT SET hc1 (INPUT 3,
INPUT 4).
RUN DispCx IN hc1.

DISPLAY "równa się" WITH FRAME f2 NO-BOX.
RUN MulCx IN hc0 (INPUT hc0, INPUT hc0,
INPUT hc1).
RUN DispCx IN hc0.
```

Rys.4. Efekt wykonania programu usecx.p



Wykonanie instrukcji RUN complex.p PERSISTENT SET hc0 (INPUT 1, INPUT 2). składa się z trzech kroków:

(1) załadowania ciała procedury complex.p do pamięci (tj. kodu + danych lokalnych)

(2) wykonaniu wszystkich instrukcji tej procedury (tu: nadanie wartości początkowych atrybutom re i im)

(3) po zakończeniu instrukcji RUN ciało procedury pozostaje nadal w pamięci - jest to właśnie instancja obiektu, której wskaźnik zostaje zapamiętany na zmiennej hc0

Wskaźniki instancji klas mają ten sam ogólny typ HANDLE - nie jest zatem możliwe rozróżnienie "typów" obiektowych dla wskaźników obiektów różnych klas. Wewnątrz danej instancji można odczytać wskaźnik "do siebie samej" - zwraca go funkcja THIS-PROCEDURE. Element językowy THIS-PROCEDURE jest także obiektem systemowym posiadającym następujące atrybuty: CURRENT-WINDOW - wskaźnik okna związanego z instancją, FILE-NAME - nazwa wywołanego pliku procedury instancji, INTERNAL-ENTRIES - tekstowa lista nazw podprocedur wewnętrznych danej instancji, NEXT-SIBLING - kolejna instancja PERSISTENT w sesji, PERSISTENT - znacznik logiczny czy wywołanie PERSISTENT, PREV-SIBLING - poprzednia instancja PERSISTENT w sesji, PRIVATE-DATA - wartość tekstowa do wykorzystania przez programistę, TYPE - wewnętrzny typ obiektu (tu: "procedure"). Do weryfikacji poprawności wskaźnika instancji służy funkcja VALID-HANDLE:

```
IF VALID-HANDLE(wsk-instancji) THEN
  MESSAGE "obiekt istnieje".
ELSE
  MESSAGE "obiektu już nie ma".
```

Usunięcie instancji można wykonać przy pomocy instrukcji DELETE PROCEDURE *wsk-instancji*. W efekcie cała pamięć zajmowana przez instancję jest zwalniana, a wskaźnik staje się nieprawidłowy (tj. wynikiem testu funkcji VALID-HANDLE dla tego wskaźnika jest wartość fałsz). Komunikację pomiędzy obiektami (a zatem przebieg sterowania) można realizować przez wywołania z poziomu jednego obiektu metod zdefiniowanych w ciele instancji innego obiektu (czyli podprocedur wewnętrznych - bez parametrów lub z parametrami). Wywołanie takie ma ogólną postać:

```
RUN metoda IN wsk-instancji(parametry).
```

Alternatywny sposób komunikacji to programowe generowanie bezparametrowych sygnałów - zdarzeń, wysyłanych przy pomocy instrukcji APPLY do instancji obiektu. Zdarzenia takie są następnie obsługiwane przez instancję w odpowiednich blokach trygerów - w trakcie realizacji instrukcji WAIT-FOR w głównej procedurze startowej. Na przykład przesłanie zdarzenia CLOSE:

```
APPLY "CLOSE" TO wsk-instancji.
```

stosuje się zwyczajowo jako sygnał zniszczenia instancji. Wewnątrz instancji, do której przesyłane jest takie zdarzenie definiuje się tryger realizujący w odpowiedzi na zdarzenie CLOSE akcję samozniszczenia:

```
ON CLOSE OF THIS-PROCEDURE
DO: /*autodestrukcja */
  IF THIS-PROCEDURE:PERSISTENT THEN
    DELETE PROCEDURE THIS-PROCEDURE.
END.
```

W procedurach kodowanych wg stylu "zdarzeniowego", które mogą być wywoływane w zwykły sposób lub jako PERSISTENT umieszcza się zwykle test wykluczający użycie instrukcji WAIT-FOR wewnątrz procedury wywołanej jako PERSISTENT:

```
IF NOT THIS-PROCEDURE:PERSISTENT THEN
  WAIT-FOR ...
```

Chodzi tu o to, żeby zapobiec ewentualnemu uwięzieniu sterowania wewnątrz instancji obiektu.

W stylu „obiektowym" istotne jest przyjęcie jakiegoś standardu kodowania i obsługi obiektów. W szczególności należy przestrzegać następujących ogólnych zasad:

- „ciężar" sterowania spoczywa w jednej instrukcji WAIT-FOR umieszczonej w głównej procedurze startowej
- istniejący obiekt, który generuje jakąś inną instancję musi się o nią troszczyć (tj. przechowywać jej wskaźnik, wykorzystywać stosowne metody do komunikacji, a także w ustalony sposób niszczyć instancję jak spełni już swoje zadanie)
- wszystkie obiekty powinny realizować jednolity sposób komunikacji i zapamiętywania powiązań (tj. wskaźników do innych obiektów) - np. w lokalnych tablicach TEMP-TABLE

Ilustrację opisanych tu technik można prześledzić w niżej zamieszczonych przykładach programów. Program h05-rodz.p jest nadrzędną instancją obiektu „rodzica", z którego mogą zrodzić się przy pomocy instrukcji RUN ... PERSISTENT „dzieci" - zdefiniowane jako klasa w programie h06-dzie.p. Na rys. 4 widzimy efekt wykonania tego programu - w tytułach okien wyświetlane są w postaci tekstowej wartości wskaźników poszczególnych instancji. Możemy także prześledzić technikę komunikacji na styku obiektów „rodzic"<->"dziecko". Komunikaty są przesyłane z poziomu rodzica do dziecka przez wywołanie metody (np. w przypadku zmiany koloru dziecka) lub przez sygnały (w przypadku niszczenia obiektu dziecka - przesłanie instrukcją APPLY zdarzenia CLOSE do instancji dziecka). Nic nie stoi na przeszkodzie, żeby komunikacja przebiegała w inną stronę (np. od „dziecka" do „rodzica" lub pomiędzy „dziećmi") - należy wtedy w poszczególnych instancjach przechowywać informacje o wskaźnikach obiektów, z którymi dana instancja może nawiązać kontakt. W wypadku obiektu „rodzica" wykorzystano do tego celu tablicę tymczasową TEMP-TABLE. Należy pamiętać przy tym, że tak jak w stylu „zdarzeniowym" praktycznie całe sterowanie jest realizowane podczas wykonania instrukcji WAIT-FOR umieszczonej w procedurze startowej - w naszym przypadku w instancji „rodzica". Poniżej przedstawiono teksty programów h05-rodz.p i h06-dzie.p:

```
/* h05-rodz.p */
/*** definicje ***/
DEF TEMP-TABLE tab1 /* tabela na wskaźniki
*/
  FIELD wsk AS HANDLE
  FIELD wsk-txt AS CHAR FORMAT "x(20)".
DEF VAR h AS HANDLE.
DEF QUERY q1 FOR tab1. /* zapytanie */
DEF BROWSE b1 QUERY q1 /* browser */
DISPLAY wsk-txt WITH 5 DOWN.

DEF BUTTON btn-nowe-dziecko
LABEL "nowe dziecko".
DEF BUTTON btn-zamknij LABEL "Zamknij".
DEF BUTTON btn-niszcz LABEL "Niszcz
dzieci".
DEF BUTTON btn-kolor LABEL "Zmiana koloru".

/*** ramki ***/
DEF FRAME f1
  btn-nowe-dziecko AT ROW 1 COL 1
  btn-niszcz       AT ROW 2 COL 1
  btn-kolor        AT ROW 3 COL 1
  b1               AT ROW 1 COL 20
  btn-zamknij      AT ROW 4 COL 1.

/*** trygery ***/
ON CHOOSE OF btn-kolor DO:
  /* wywołanie procedury wewnętrznej */
  /* z konkretnej instancji */
  RUN zmien-kolor IN tab1.wsk(INPUT 5).
END.

ON CHOOSE OF btn-nowe-dziecko DO:
  RUN h06-dzie.p PERSISTENT SET h.
  CREATE tab1.
  tab1.wsk = h.
  tab1.wsk-txt = STRING(h).
  OPEN QUERY q1 FOR EACH tab1.
END.

ON CHOOSE OF btn-niszcz DO:
  FOR EACH tab1 :
    IF VALID-HANDLE(tab1.wsk) THEN
      /* przesłanie sygnału zniszczenia */
      APPLY "CLOSE" TO tab1.wsk.
  END.
END.

/*** main ***/
ASSIGN
  DEFAULT-WINDOW:HEIGHT = 6
  DEFAULT-WINDOW:WIDTH = 50
  DEFAULT-WINDOW:TITLE = "rodzic" +
    " " + STRING(THIS-PROCEDURE).

ENABLE ALL WITH FRAME f1.
WAIT-FOR CHOOSE OF btn-zamknij.
```

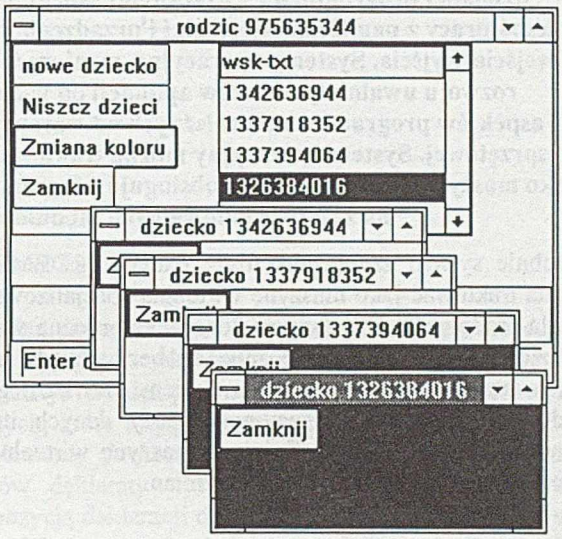
```
/* h06-dzie.p */
/*** definicje ***/
DEF VAR v-kolor AS INT.
DEF VAR v-okno AS WIDGET-HANDLE.
DEF BUTTON btn-zamknij LABEL "Zamknij".

/*** ramki ***/
DEF FRAME f1
  btn-zamknij.
```

```
/*** trygery ***/
ON CLOSE OF THIS-PROCEDURE DO:
  DELETE WIDGET v-okno.
  IF THIS-PROCEDURE:PERSISTENT THEN
     DELETE PROCEDURE THIS-PROCEDURE.
END.

ON CHOOSE OF btn-zamknij DO:
  APPLY "CLOSE" TO THIS-PROCEDURE.
END.

/*** main ***/
CREATE WINDOW v-okno
   ASSIGN
      HEIGHT = 3
      WIDTH = 30
      TITLE = "dziecko" + " " +
        STRING(THIS-PROCEDURE)
      MESSAGE-AREA = NO
      STATUS-AREA = NO.

VIEW v-okno.
v-kolor = v-okno:BGCOLOR.

ENABLE ALL WITH FRAME f1 IN WINDOW v-okno.
IF NOT THIS-PROCEDURE:PERSISTENT THEN
  WAIT-FOR CLOSE OF THIS-PROCEDURE.

/*** proc. wewnętrzne - ***/
/*** - czyli metody instancji ***/

PROCEDURE zmien-kolor: /* okna dziecka */
  DEF INPUT PARAM p-kolor AS INT.

  IF v-kolor = v-okno:BGCOLOR THEN
     v-okno:BGCOLOR = p-kolor.
  ELSE
     v-okno:BGCOLOR = v-kolor.
END.
```

Możliwości rozszerzeń stylu obiektowego

Kolejnym krokiem jest opracowanie konsekwentnej techniki dziedziczenia (w Progressie jest to możliwe w ograniczonym stopniu przy pomocy dyrektyw wbudowanego preprocesora i plików włączanych) oraz jakiejś realizacji polimorfizmu (np. przy pomocy ogólnej metody *dispatch*, która w efekcie ma wywołać inną metodę wirtualną - jedną z kilku dziedziczonych odmian o podobnych nazwach, której główny fragment nazwy jest przekazywany jako parametr metody *dispatch*). Porządna realizacja tych zadań jest jednak stosunkowo pracochłonna i wymaga w zasadzie w wersji 7 Progressa implementacji od zera. Opracowanie sensownego i wydajnego własnego standardu, który byłby w miarę uniwersalny wcale nie jest sprawą łatwą. Jest to w gruncie rzeczy przejście podobne do kroku pomiędzy surową definicją obiektowego języka programowania, a językiem obiektowym wyposażonym w biblioteki klas abstrakcyjnych. W wersji 8 Progressa nie trzeba (na szczęście) startować od zera - większość typowych zadań można z powodzeniem realizować wykorzystując dostarczony standard ADM zawierający wzorce gotowych obiektów uogólnionych - wspomnianych na wstępie tzw. *SmartObjects* (np. *SmartBrowser, SmartViewer, SmartPanel, SmartQuery*). Wzorce te odpowiadają w zasadzie definicjom klas abstrakcyjnych spotykanych w obiektowych językach programowania. Standard ADM w dużej mierze opiera się na opisanych w tym artykule technikach - jednak chociażby pobieżne go opisanie stanowi osobny temat.

Rys.5. Efekt wywołania programu h05-rodz.p



Polska wersja Office Professional dla Windows 95

Microsoft wprowadził na rynek polską wersję pakietu oprogramowania Microsoft Office Professional dla Windows 95. Pakiet składa się z edytora tekstów Microsoft Word 7.0, arkusza obliczeniowego Microsoft Excel 7.0, pakietu grafiki prezentacyjnej Microsoft PowerPoint 7.0, programu do planowania zajęć Microsoft Schedule + 7.0 oraz systemu zarządzania relacyjnymi bazami danych Microsoft Access 7.0. W pakiecie zastosowano ulepszoną technologię IntelliSense, ułatwiającą wykonywanie skomplikowanych zadań i automatyzującą czynności rutynowe. Na przykład, program automatycznie poprawia błędy w trakcie pisania tekstu, daje szybki dostęp do najważniejszych danych zgromadzonych w arkuszu kalkulacyjnym, system Kreatorów pomaga w stworzeniu estetycznej prezentacji. Technika OfficeLinks pozwala na wymianę informacji między wszystkimi aplikacjami wchodzącymi w skład pakietu. Office Binder (Spinacz) umożliwia łączenie kilku plików w jeden pakiet. Microsoft Office Professional został zoptymalizowany pod kątem Windows 95. Wszystkie aplikacje są w pełni 32-bitowe i wielowątkowe, co umożliwia jednoczesne wykonywanie wielu zadań. (k)

MAGIC - kompendium wiedzy

Marcin Gorawski
Andrzej Konopacki
Pracownia Podstaw i Zastosowań Informatyki
Zakład Karbochemii PAN
Gliwice

Zanim pojawiły się systemy operacyjne, pełniące rolę pośrednika między programami a sprzętem, twórcy aplikacji musieli zajmować się nie tylko logiką działania programu, ale i fizycznymi aspektami współpracy z pamięcią, rejestrami i urządzeniami wejścia-wyjścia. Systemy operacyjne, w miarę ich rozwoju uwalniały twórców aplikacji od wielu aspektów programowania zależnych od warstwy sprzętowej. System operacyjny można traktować jako maszynę wirtualną, która obsługuje programy na zasadzie ładowalnych modułów.

Podobnie system zarządzania bazą danych (RDBMS) można traktować jako maszynę wirtualną specjalizowaną dla środowiska baz danych. Idąc dalej można wyobrazić sobie projektowanie aplikacji bez względu na ich związek z docelowym środowiskiem. Dla każdego środowiska, systemu operacyjnego, bazy danych itd. musiałby istnieć moduł definiujący maszynę wirtualną, natomiast program nie wymagałby zmian.

Koncepcja i metodyka działania generatora aplikacji baz danych Magic

Koncepcja taka jest krokiem naprzód w stosunku do konwencjonalnych języków 3GL lub 4GL i jest określana jako „ponad-4GL". Zadaniem systemów „ponad 4GL" jest umożliwienie użytkownikowi skoncentrowania się w pełni na projektowaniu aplikacji przy automatycznej obsłudze szczegółów implementacyjnych. Języki „ponad 4GL" są zorientowane obiektowo i problemowo, pozwalają na szybkie tworzenie aplikacji bez pracochłonnego konstruowania algorytmów i kodowania. Dlatego nazywane są generatorami aplikacji.

Na proces tworzenia aplikacji przy pomocy generatora składają się następujące nierozdzielne elementy :

- projektowanie,
- tworzenie prototypu aplikacji,
- testowanie i modyfikacja prototypu.

Różnice między tradycyjną i bezkodową metodą tworzenia aplikacji przedstawiono schematycznie na rys.1.

Koncepcja Magic'a

Koncepcję maszyny wirtualnej wykorzystano również w generatorze aplikacji Magic. Maszyna wirtualna uniezależnia aplikację od platformy sprzętowej (PC, VAX, IBM RISC/System, AS/400, SunSPARC, itd.), systemowej (DOS, NetWare, UNIX, Solaris, VMS) oraz systemu zarządzania bazą danych (Btrieve, dBASE, Oracle, Informix, itp.). Generalne możliwości integracyjne Magic'a dla różnych środowisk przedstawiono na rys.2.

Na maszynę wirtualną Magic'a składają się trzy moduły :

- moduł wykonawczy (ang. *runtime*) pozwalający użytkownikowi uruchomić aplikację na danej platformie sprzętowej i systemowej;
- moduł narzędziowy (ang. *development*) pozwalający programiście utworzyć i modyfikować aplikację dla danej platformy sprzętowej i systemowej oraz uruchamiać aplikację na tej platformie; moduł narzędziowy pracuje w dwóch trybach: *toolkit* - tworzenie aplikacji oraz *runtime* - testowanie aplikacji;
- bramę do bazy danych (ang. *gateway*) pozwalającą na dostęp do różnych systemów baz danych zarówno użytkownikom, jak i programistom.

Istotny jest fakt, że aplikacje utworzone przy pomocy modułu narzędziowego dla pewnej platformy systemowej mogą być przenoszone do modułów narzędziowych innych platform, a po niewielkich modyfikacjach uruchamiane za pomocą modułu wykonawczego dowolnej platformy.

Autor aplikacji (programista) może się teraz skoncentrować na istotnych szczegółach realizacji systemu, pozostawiając algorytm wykonania zaprojektowanych działań maszynie wirtualnej. Nie musi również programować wielodostępu. Maszyna wirtualna Magic'a zawiera integralne mechanizmy wielodostępu.

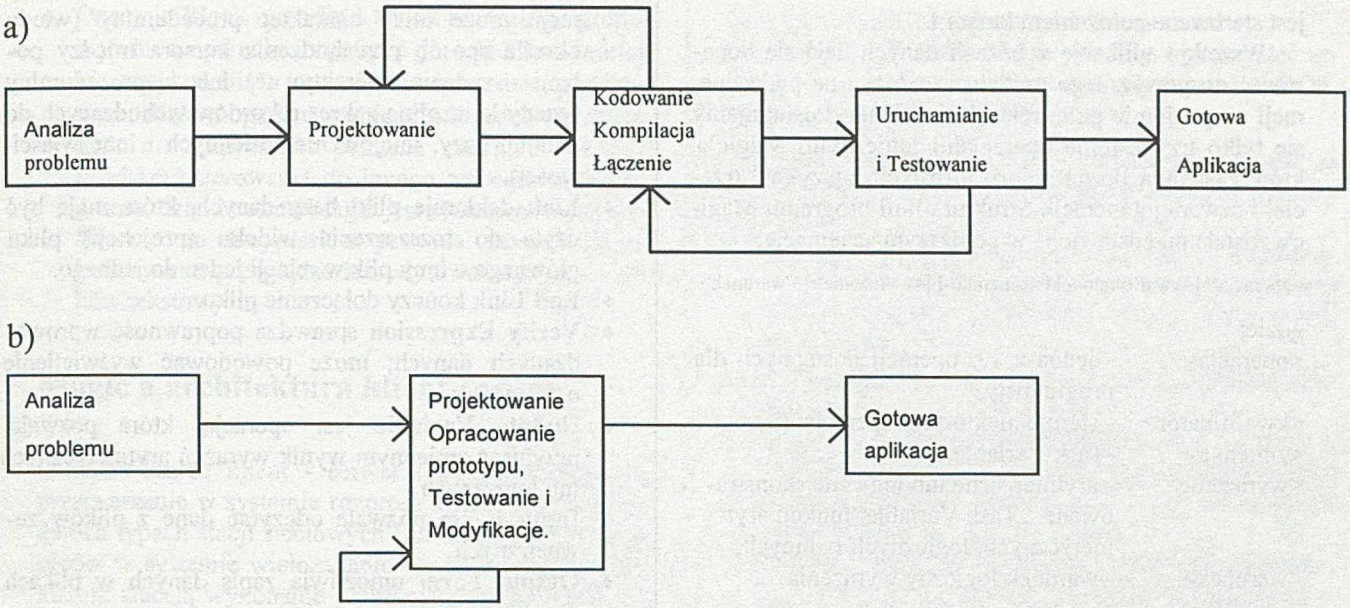
Metodyka Magic'a

Programowanie w Magic'u polega na:

- poznaniu i opisaniu wymagań końcowego użytkownika,
- określeniu funkcji aplikacji i logiki programów (zadań),
- utworzeniu specyfikacji wysokiego poziomu za pomocą wypełnienia odpowiednich tablic i słowników.

Każdy program Magic'a składa się z kilku zadań opisanych za pomocą struktury zwanej Tablicą Sterowania (ang. *task execution table*), która odzwierciedla cykliczne działanie maszyny wirtualnej, co schematycznie przedstawiono na rys.3.

Operacje z poziomu zadania są wykonywane w czasie inicjalizacji zadania (ang. *Task Prefix*) i jego za-



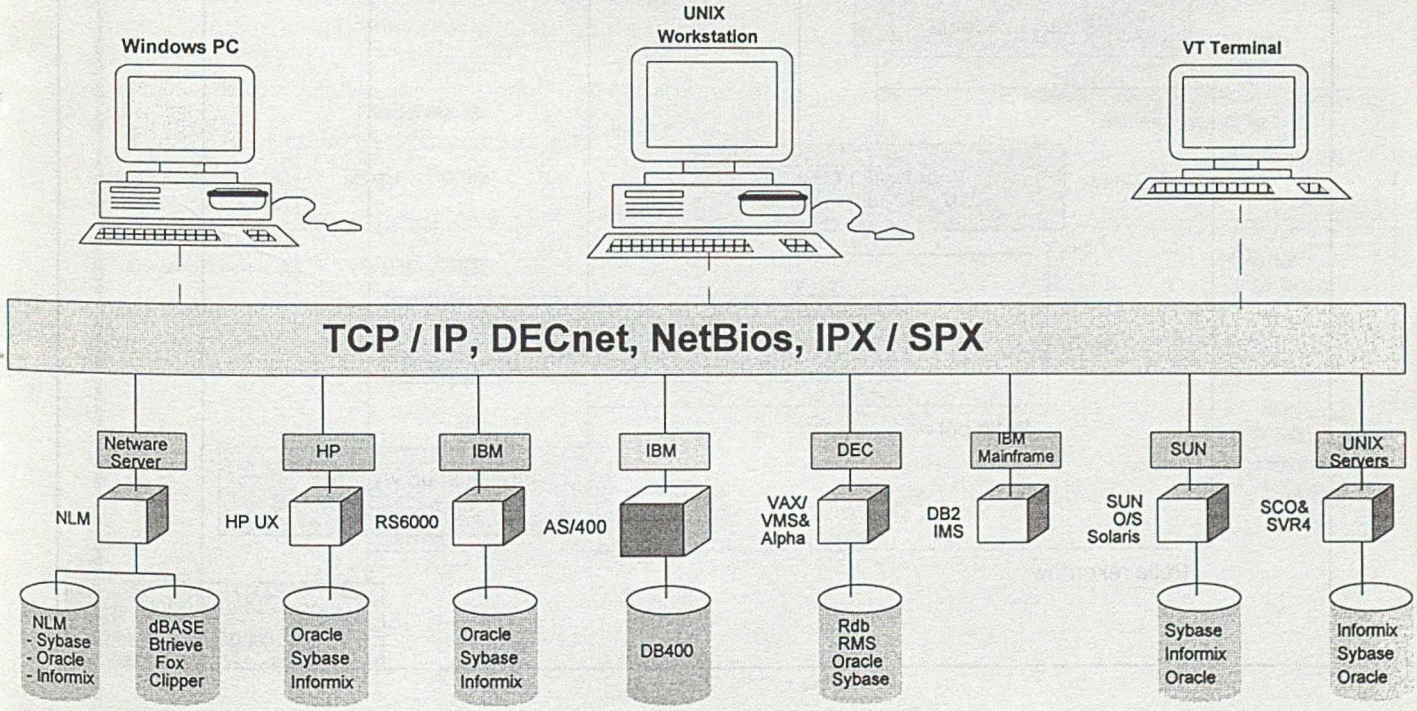
Rys.1. Tradycyjny i bezkodowy sposób tworzenia aplikacji
a) tradycyjna metoda tworzenia aplikacji, b) bezkodowa metoda tworzenia aplikacji

kończenia (ang. *Task Suffix*). Te operacje musi wyspecyfikować programista.

Operacje z poziomu rekordu są wykonywane dla inicjowania przetwarzania rekordu (ang. *Record Prefix*), w czasie interaktywnego przetwarzania pól rekordu (ang. *Record Main*) oraz w fazie zakończenia przetwarzania rekordu (ang. *Record Suffix*). Te operacje również w sposób jawny musi określić programista.

Zadania Magic'a są zbudowane z elementów *proceduralnych i nieproceduralnych*. Elementy nieproceduralne to te, których kolejność wykonywania jest sterowana przez maszynę wirtualną. Elementy proceduralne to te, których sekwencja jest pod kontrolą programisty. Istnieje również pewna grupa elementów nieproceduralnych, które mają wpływ na przebieg wykonania. Są to elementy *deklaratywnie nieproceduralne*. Dla elementów deklaratywnych istnieje korelacja między fizyczną pozycją deklaracji danych w zadaniu, a domyślną kolejnością poruszania się kursora między polami danych na ekranie. Elementem deklaratywnym jest, na przykład, Typ Zadania. Zależnie od tego czy zadanie jest wsadowe, czy interaktywne, przetwarzanie danych jest lub nie

Rys. 2. Integracja środowiskowa generatora Magic



jest sterowane położeniem kursora.

Wszelkie aplikacje w bazach danych dają się dopasować do powyższego modelu i rozłożyć na pętle operacji na poziomie pola, rekordu i zadania. Posługujemy się tylko trzynastoma operacjami logicznymi Magic'a, które zastępują liczne zbiory instrukcji z języków trzeciej i czwartej generacji. Struktura linii programu Magic'a została przedstawiona w poniższym schemacie:

<operacja> [<kwalifikator>] [<zmienna>] [<wyrażenie>] <warunek>

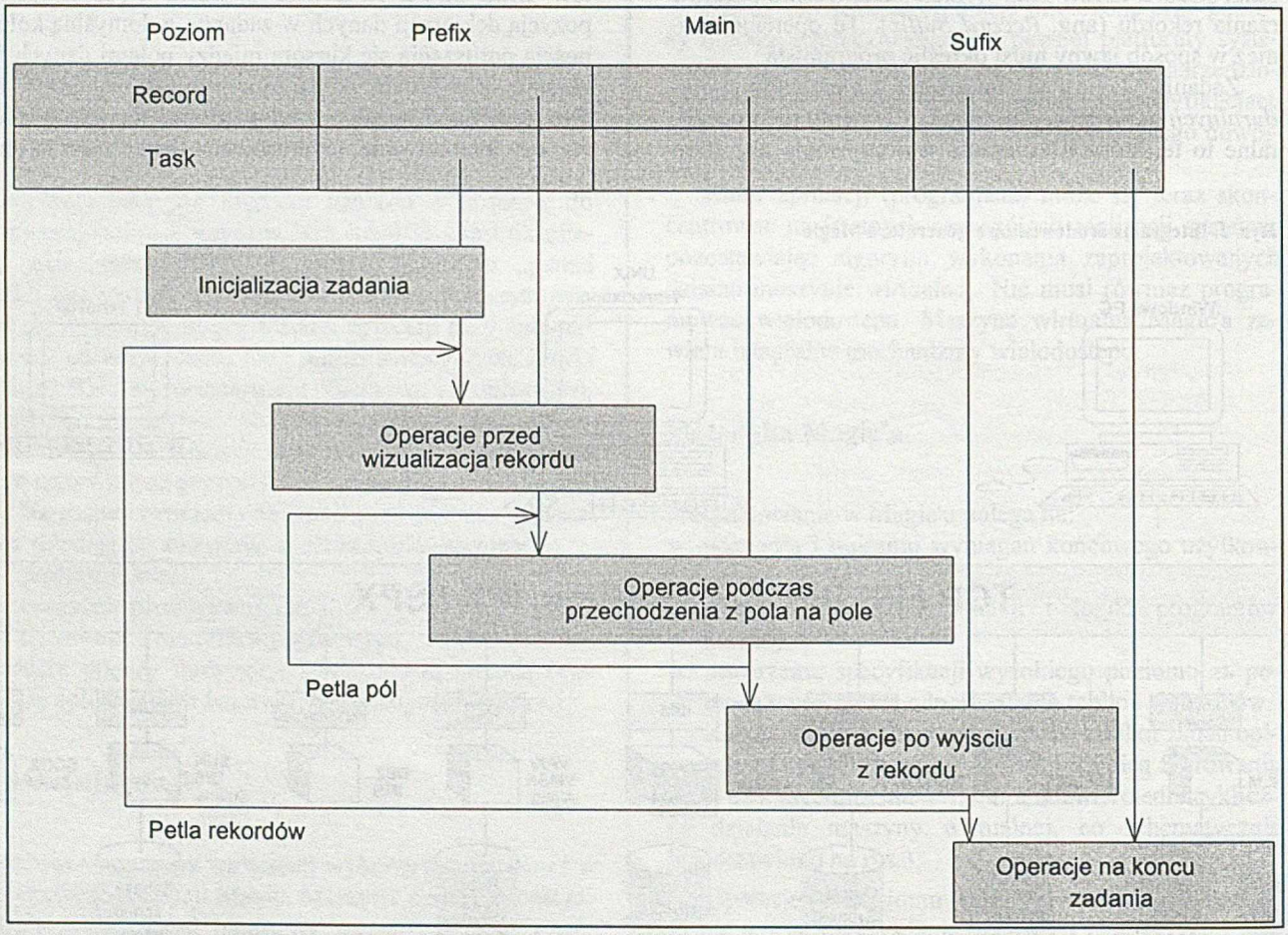
gdzie:

<operacja>	- jedna z 13. operacji dostępnych dla programisty,
<kwalifikator>	- steruje niektórymi operacjami,
<zmienna>	-Task Variable,
<wyrażenie>	- arytmetyczne lub logiczne skonstruowane z Task Variable, funkcji arytmetycznych, logicznych i innych,
<warunek>	- warunek logiczny wyrażenia

Moduł wykonawczy maszyny wirtualnej sprawdza warunek, jeśli jest on prawdziwy - wykonuje operację. **Oto opis 13. operacji Magic'a dostępnych dla programisty:**

- Operacja **Select** może być typu Real lub Virtual i określa, która ze zmiennych rzeczywistych lub wirtualnych (roboczych) jest dostępna dla zadania. Operacja może mieć charakter proceduralny (wtedy określa sposób przechodzenia kursora między polami - zadania interaktywne) lub nieproceduralny (wtedy kontroluje zakres rekordów wchodzących do widoku bazy, inicjowanie zmiennych i inne właściwości).
- **Link** deklaruje pliki bazy danych, które mają być użyte do rozszerzenia widoku (projekcji) pliku głównego o inny plik w relacji jeden do jednego.
- **End Link** kończy dołączanie pliku.
- **Verify Expression** sprawdza poprawność wprowadzanych danych; może powodować wyświetlenie ostrzeżeń.
- **Update Variable** jest operacją, która pozwala przypisać zmiennym wynik wyrażeń arytmetycznych lub logicznych.
- **Input Form** pozwala odczytać dane z plików zewnętrznych.
- **Output Form** umożliwia zapis danych w plikach zewnętrznych.
- Operacja **Call** pozwala aktualnie wykonywanemu zadaniu wywołać podzadanie, inny niezależny program Magic'a należący do tej aplikacji, lub program napisany w innym języku programowania.
- **Evaluate Expression** pozwala wywoływać procedury zdefiniowane w środowisku Magic'a.

Rys. 3. Schemat działania maszyny wirtualnej Magic'a



- Operacje **Block oraz End Block** pozwalają zgrupować inne operacje i opatrzyć je wspólnym warunkiem wykonania, co czyni zadanie bardziej przejrzystym. Bloki mogą być zagnieżdżone jeden w drugim.
- **Exit on Expression** pozwala zadaniom Magic'a przekazać sterowanie do innego zewnętrznego programu, dzięki czemu Magic jest systemem otwartym.
- **Browse on Expression** pozwala użytkownikowi końcowemu przeglądać lub edytować zawartość plików tekstowych.

Magic a architektura klient-serwer

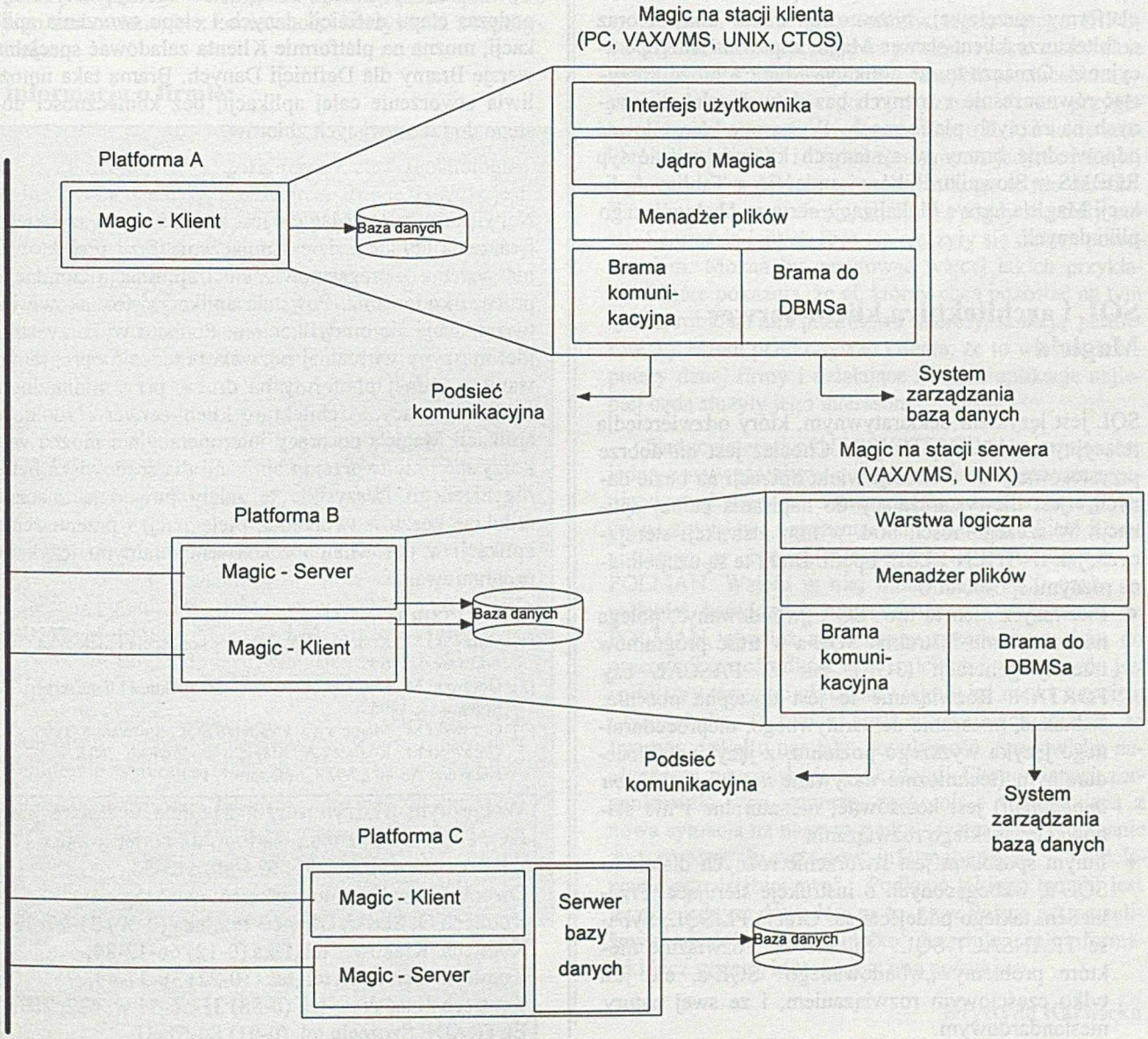
O architekturze klient - serwer mówimy wtedy, gdy przetwarzanie w systemie rozproszonym odbywa się na dwóch typach stacji sieciowych (lub dwóch typach procesów w systemie wielozadaniowym). Stacje (procesy) klienta zlecają wykonanie wybranych usług innym stacjom (procesom) określanym mianem serwerów.

Maszyna wirtualna Magic pozwala wykorzystać architekturę klient-serwer dla takich platform, jak DOS, NetWare, VAX/VMS, UNIX i CTOS.

System klienta w Magic'u można przedstawić za pomocą modelu czterowarstwowego. Kolejne warstwy (rys.4) to :

1. **Interfejs użytkownika**, który zajmuje się współpracą z użytkownikiem i prezentacją ekranową aplikacji oraz interfejsem graficznym. Istnieje kilka wersji tego modułu.
2. **Jądro Magic'a**, które steruje wykonywaniem aplikacji.
3. **Menadżer plików,** który zajmuje się łącznością z lokalnymi bazami danych (przez odpowiednie bramy), innymi plikami i systemem operacyjnym stacji roboczej.
4. **Warstwa bram**, która składa się z dwóch modułów:
   - ***bramy do baz danych*** - zajmuje się łącznością z

Rys. 4. Organizacja modułu klient - serwer Magic'a



konkretną bazą danych zainstalowaną na komputerze kliencie, co zapewnia niezależność od bazy danych;

- *bramy komunikacyjnej* - zajmuje się komunikacją ze zdalnymi serwerami. Ten moduł zawiera sterowniki dla wielu protokołów komunikacyjnych min. TCP/IP i DECnet, co zapewnia niezależność od konfiguracji sieciowej.

Systemy serwera w Magic'u istnieją dla systemów operacyjnych: VAX/VMS i UNIX. System serwera można opisać za pomocą modelu trójwarstwowego. Oto kolejne warstwy (rys.4):

1. **Warstwa logiczna**, która odbiera sygnały żądania usług od klientów i odpowiednio wysyła do nich wyniki.
2. **Menadżer plików,** który pełni tę samą rolę, co odpowiednia warstwa systemu klienta.
3. **Warstwy bram**, która jest identyczna, jak odpowiednia warstwa bram systemu klienta.

Możliwość wykorzystania architektury klient-serwer przez aplikacje Magic'a zapewnia znacznie efektywniejszy dostęp do danych i zmniejsza liczbę pakietów danych transmitowanych siecią. Dzięki niezależności od platformy sprzętowej, systemowej i bazy danych oraz architekturze klient-serwer Magic zapewnia interoperacyjność. Oznacza to, że aplikacja Magic'a może korzystać równocześnie z różnych baz danych, zlokalizowanych na różnych platformach. Wystarczy zainstalować odpowiednie bramy w systemach klienta, wpisać typ RDBMS w Słowniku Plików i umieścić w Tablicy Aplikacji Magic'a nazwę i lokalizację serwera obsługującego pliki danych.

SQL i architektura klient-serwer Magic'a

SQL jest językiem deklaratywnym, który odzwierciedla relacyjny model bazy danych. Chociaż jest on dobrze przystosowany do realizacji wielu operacji na bazie danych, jest niewystarczający do napisania pełnej aplikacji. W szczególności, brak w nim instrukcji sterujących, jak IF-THEN-ELSE, i pętli. Braki te są uzupełniane różnymi sposobami:

- Pierwszy z nich to tzw. SQL „wbudowany", polega na wstawianiu instrukcji SQL-a w treść programów trzeciej generacji (3GL), jak C, PASCAL czy FORTAN. Rozwiązanie to jest dostępne obecnie. Jednakże, mieszanie deklaratywnego, nieproceduralnego języka wyższego poziomu, z językiem proceduralnym (technicznie nazywane *niedopasowaniem impedancji*) jest kosztowne, nienaturalne i nie stanowi eleganckiego rozwiązania.
- Innym sposobem jest stworzenie różnych dialektów SQL-a, wzbogaconych o instrukcje sterujące. Przykładem takiego podejścia są: Oracle PL/SQL, Sybase TRANSACT-SQL. Sposób ten rozwiązuje niektóre problemy „wbudowanego" SQL-a, ale jest tylko częściowym rozwiązaniem, i ze swej natury, niestandardowym.
- Długofalowym rozwiązaniem jest stworzenie nowego standardu (SQL-3), ale na razie spotyka on wiele problemów. Jak dotąd, szkic specyfikacji liczy sobie już ponad 700 stron i chyba nikt nie oczekuje szybkiego wejścia nowego standardu SQL-3 w życie, ani wejścia na rynek przed rokiem 2000 produktu spełniającego jego wymogi.

Magic obsługuje SQL-a dwoma sposobami:

- **niejawnym** - zlecenia SQL-a są generowane automatycznie przez Magic'a, jako wynik odwołania aplikacji do SQL-owej bazy danych. Ponieważ generacja tej kategorii zleceń jest całkowicie pod kontrolą Magic'a, zatem mogą one dzięki Bramie Bazy Danych uwzględniać specyfikę danego typu bazy danych. Znajomość SQL-a przez programistę nie jest wymagana.
- **jawnym** - wbudowane w aplikację zlecenia SQL-a są napisane przez programistę. Wbudowane komendy SQL-a, muszą oczywiście uwzględniać składnię wersji SQL-a, który obsługuje bazę. Polecenie jest następnie przesyłane przez odpowiednią Bramę do Serwera Bazy Danych.

By uniknąć potrzeby dostępu do rzeczywistej bazy, podczas etapu definicji danych i etapu tworzenia aplikacji, można na platformie Klienta załadować specjalną wersję Bramy dla Definicji Danych. Brama taka umożliwia utworzenie całej aplikacji, bez konieczności dostępu do rzeczywistych zbiorów.

* * *

Największą zaletą Magic'a jest umożliwienie szybkiego tworzenia aplikacji dzięki połączeniu fazy projektowania systemu z programowaniem i eliminacją żmudnego procesu kodowania. Powstała aplikacja stosunkowo łatwo poddaje się modyfikacjom. Ponadto wykorzystanie idei maszyny wirtualnej pozwala przenosić oprogramowanie z jednej platformy na drugą, przy minimalnym nakładzie pracy. Architekturę klient-serwer i zdolność aplikacji Magic'a do pracy interoperacyjnej można wykorzystać przy tworzeniu aplikacji dla środowiska heterogenicznego. Wszystkie te zalety powodują znaczną redukcję kosztów tworzenia, pielęgnacji i przenoszenia aplikacji w porównaniu z konwencjonalnymi językami programowania.

LITERATURA:

[1] Gorawski M., Konopacki A.: Magic - koncepcja i metodyka. Software 4, 1995

[2] Gorawski M., Koziatek A.: Magic - architektura klient/server. Software 5, 1995

[3] Gorawski M.: Magic a język zapytań SQL. Software 6, 1995
Gorawski M., Koziatek A.: Magic w środowisku UNIX, NetWare, AS/400, VMS. Software 7, 1995.

Otwarta droga do programoqwania obiektowego, czyli OpenROAD/Windows4GL 3.0

dokończenie ze s. 11

przetwarzania danych, tworzyć definicje, sterować procesami i obliczeniami używając tego samego języka. Używając OpenROAD/Windows4GL i SQL programiści mogą pisać aplikacje, które będą miały dostęp do relacyjnych baz danych, takich jak: OpenINGRES/Intelligent DataBase, Oracle, SQL Server lub Sybase System 10.

OpenROAD/Windows4GL umożliwia także wprowadzenie procedur pisanych w 3GL do tworzonych w 4GL.

Zarządzanie aplikacją złożoną

OpenROAD/Windows4GL świetnie nadaje się do tworze-nia także bardzo dużych programów. Dzięki użyciu rela-cyjnych baz danych, jako repozytorium, może automa-tycznie śledzić położenie wszystkich składników w całym cyklu życia aplikacji. Programista wprowadzając zmiany do jednego z elementów aplikacji może sprawdzić jego zachowanie, bez konieczności korzystania z repozytorium.

Informacja o firmie:

Założona w 1983 roku firma Cadre Technologies Inc. (z siedzibą w Providence, Rhode Island) jest czołowym na świecie dostawcą narzędzi do programowania i usług w zakresie pełnego cyklu budowy aplikacji. Współpracujące ze sobą pakiety umożliwiają stosowanie dwóch najpopularniejszych metod tworzenia oprogramowania: obiektowej i strukturalnej. Platformą systemową mogą być stacje robocze pracujące pod SunOS, Solaris, AIX, HP-UX, Digital VMS i MS Windows.

Biura techniczne firmy znajdują się w USA, Holandii, Niemczech, Wielkiej Brytanii i Australii, zaś biura handlowe - w większości rozwiniętych krajów.

Istnieje już ponad 50 tys. instalacji produktów Cadre. Spośród ich użytkowników warto wymienić takich potentatów, jak: General Motors, Kodak, IBM, Motorola, AT&T, KLM Airlines, Lufthansa, Bank of England, Deutsche Bundespost, Siemens, Mercedes Benz.

W opinii znanej firmy specjalizującej się w ba-da-niach rynku Gartner Group „Cadre jest liderem w budowie systemów OO", zaś firma IDC o podobnym profilu uważa, iż „Cadre jest wiodącym na rynku dostawcą OO CASE".

Lesław Wawrzonek

INFOSYSTEM' 96

dokończenie z I okł.

Wielu producentów sprzętu i oprogramowania reprezentowanych było jedynie przez dealerów, inni mieli wyłącznie stoiska informacyjne, niektóre z dobrze zaopatrzoną kawiarnią, gdzie lekko i przyjemnie rozmawiało się o przyszłych, nie zawsze łatwych wspólnych interesach. Nawet młodzież i wszelkiej maści "wymiatacze" folderów, toreb i drobnych firmowych gadżetów bardziej tłoczyli się w znakomicie zorganizowanej Kawiarni Internetowej i przy komputerach z multimedialnym oprogramowaniem, niż przy najnowszym sprzęcie. Nic dziwnego, dla coraz większego kręgu młodzieży komputer jest jedynie wygodnym narzędziem ułatwiającym wykonywanie różnych czynności i umożliwiającym, dzięki sieci Internet, wirtualny dostęp do świata. Są do komputera tak przyzwyczajeni, jak do telewizora czy telefonu. Ich już nie uda się zafascynować jedynie "pudełkiem", co oczywiście nie znaczy, że nie marzą o własnym komputerze z Pentium.

Zdaniem wielu obserwatorów tegorocznego INFOSYSTEMU nową sytuację na polskim rynku znakomicie wyczuł SAP - lider wśród światowych producentów oprogramowania służącego zarządzaniu. Firma przygotowała polską wersję swojego systemu R/3, a takie znakomitości jak IBM i Digital Equipments jedynie jako jej partnerzy wystąpiły na targach. I nikogo to nie dziwiło. Krzesła w salce seminaryjnej SAP-a nigdy nie stały puste. Seminaria innych firm też cieszyły się dużym powodzeniem. Można by zacytować więcej takich przykładów, które pokazują, że ci, którzy chcą pozostać na tym rynku, robić na nim prawdziwe interesy, szukają partnerów, by razem przekonywać klienta, że to właśnie komputery danej firmy i działające na nich aplikacje najlepiej będą służyły jego interesom.

Tradycyjnie już z INFOSYSTEMEM skojarzona jest jedna z najważniejszych imprez informatycznych organizowanych pod patronatem KBN - konferencja połączona z wystawą i prezentacjami - Miejskie Sieci Komputerowe w Nauce, Gospodarce i Administracji - POLMAN. Wzięło w niej udział około 500 osób, co najlepiej świadczy o randze imprezy. Wydaje się, że POLMAN, chociaż adresowany przede wszystkim do pracowników ośrodków akademickich i naukowych, jest najbliższy temu, co określamy jako więź nauki z życiem i gospodarką. To rozwój sieci w uczelniach sprawił, że Internet zyskał u nas taką popularność. Środowisko naukowców poznańskich z Akademii Ekonomicznej zorganizowało inną imprezę, znakomicie współgrającą z nową sytuacją na naszym rynku - warsztaty "Wdrażanie zintegrowanych systemów informatycznych - modelowanie przedsiębiorstwa". Podjęcie takiego tematu jest jeszcze jednym dowodem na to, że w Polsce już zaczęliśmy rozumieć, że nie "pudełko" decyduje o roli informatyki lecz system, którego jest ono nośnikiem.

Krystyna Karwicka

Implementacja aplikacji narzędziami CASE firmy Oracle

Anna Ławrynowicz
Katedra Technologii Maszyn
Akademia Techniczno-Rolnicza
Bydgoszcz

Rosnące zapotrzebowanie na systemy informatyczne w dużych organizacjach, a w szczególności oczekiwania klientów na szybkie ich wdrażanie przyspieszają rozwój w zakresie komputerowego wspomagania inżynierii oprogramowania - CASE (ang. *Computer Aided Software Engineering*). Stosowanie narzędzi CASE ułatwia analitykom definiowanie wymagań użytkowników i pełne zaangażowanie ich w tworzenie systemu. Ponadto narzędzia komputerowego wspomagania inżynierii oprogramowania automatyzują prace związane z kodowaniem programów i przygotowywaniem dokumentacji. W przypadku budowy dużych systemów informatycznych można skorzystać z narzędzi CASE firmy Oracle. Produkty tej firmy wspomagają cały proces realizacji systemu informatycznego, począwszy od narzędzi CASE służących do planowania i analizy, przez relacyjny system zarządzania bazą danych (RDBMS) i narzędzia programowania czwartej generacji, a na metodologii zarządzania całym procesem kończąc [5].

Projektowanie dużych systemów informatycznych jest czynnością bardzo złożoną i trudną. Sposób podejścia do projektowania powinien być zatem możliwie sprawdzony i usystematyzowany. Firma Oracle w swoich narzędziach CASE zastosowała etapową metodę projektowania o nazwie CASE*Method[1]. Metodyka CASE*Method reprezentuje podejście projektowania zstępującego (ang. *topdown*). Kolejne etapy projektowania rozpoczynają się na poziomie najbardziej ogólnym, a następnie są uzupełniane o informacje bardziej szczegółowe. Rezultatem każdego z etapów są m.in. takie produkty, jak: projekty, plany, diagramy, programy lub inne specyficzne produkty. Produkty te wyszczególniono w tabeli 1.

Zgodnie z metodyką CASE*Method modelowanie organizacji rozpoczyna się od rozpoznania jej strategii, we współpracy z kluczowym kierownictwem. Uzgadniane są cele organizacji i ich priorytety, określa się również zasięg przedsięwzięcia informatycznego. W trakcie cyklicznie prowadzonych wywiadów są zbierane podstawowe informacje o organizacji i wymaganiach użytkowników dotyczących tworzonego systemu informatycznego. Modelowane są specyficzne funkcje i dane, które są wielokrotnie opiniowane i weryfikowane. Efektem tej fazy jest ramowy plan informatyzacji oraz modele struktury organizacji - funkcjonalnej i informacyjnej. Etap analizy weryfikuje wytworzone specyfikacje i rozszerza je o elementy bardziej szczegółowe. Etap projektowania przetwarza analityczny model systemu w

Etapy CASE*METHOD	Produkty
Strategia	Zdefiniowane jednostki organizacyjne Diagram hierarchii funkcji Diagram związków encji Ogólne funkcje macierzy encji Diagram przepływu danych Zależności funkcyjne Sprzężenia zwrotne Zasięg systemu Sterowanie produktem
Analiza	Szczegółowy diagram związków encji Pełna hierarchia funkcji Szczegółowe funkcje macierzy encji Szczegółowy rozkład wymagań Przepływ danych / Zdefiniowane magazyny danych Cykl - życia encji Diagram wdrażania stanów Szczegółowa logika funkcji
Projektowanie	Projekt wartości domyślnych bazy danych Prognoza wielkości bazy danych Program / Zdefiniowane moduły Zdefiniowane moduły sieci Encje / Implementacja tablic Implementacja modułów funkcji Tablice / Zastosowanie modułów Architektura sieci Zdefiniowane interfejsy użytkownika
Budowa - dokumentowanie	Wygenerowany program Baza danych / Utworzone pliki Program / Dokumentacja systemu Przewodnik użytkownika Materiały testowe
Wdrożenie	Konwersja danych Wygenerowane dane testowe Zarządzanie konfiguracją
Utrzymanie i rozwój	Macierze powiązań między informacjami Zmiany w definiowaniu i dokumentacji Sterowanie zmianami Sterowanie aktualną wersją

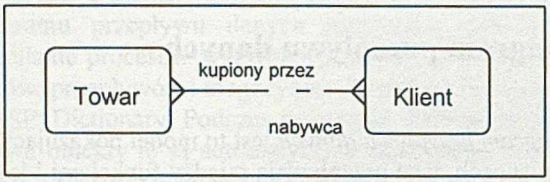
[1] Metodyka CASE firmy Oracle była prezentowana na łamach *Informatyki* w pracach [3], [4].

model jego implementacji informatycznej. Zaś etap wdrożenia implementuje system wykorzystując definicję informatyczną.

Zarówno na etapie strategii, jak i analizy, wykorzystuje się diagramy hierarchii funkcji, diagramy związków encji oraz diagramy przepływu danych.

Diagram związków encji

Diagram związków encji modeluje dane i sposób widzenia ich struktury. *Encja* to cokolwiek, o czym chcemy przechowywać informacje. Na przykład, dla danej instytucji możemy przechowywać informacje o klientach, towarach itp. Są to przykłady encji [5]. Reprezentowane na diagramach związki opisują zależności między encjami. Diagram związków encji - ERD (ang. *Entity Relationship Diagram*) można rysować z użyciem pakietu CASE*Designer. Rys. 1 przedstawia prosty diagram związków encji.



Rys. 1. Przykładowy diagram związków encji [5]

Specyficzne informacje o encjach, które chcemy przechowywać, są nazywane *atrybutami*. Możemy przechowywać dla każdego klienta jego nazwisko, adres, limit kredytowy itd. Są to atrybuty encji Klient. Podczas etapu strategicznego nie jest konieczne zdefiniowanie wszystkich atrybutów każdej encji. Należy jednak określić wszystkie atrybuty jednoznacznie identyfikujące każdą encję. Wartość takiego atrybutu, będącego identyfikatorem, pozwala wyróżnić konkretny egzemplarz encji. W konwencjach CASE*Method każdy związek jest rysowany jako linia między dwoma encjami. W istocie każda linia reprezentuje dwa związki o przeciwnych kierunkach działania. Użycie linii ciągłej w pobliżu encji oznacza, że dla każdego jej wystąpienia jest wymagany związek z drugą encją. Możemy to odczytać *Każda <encja 1> musi być* .. Linia przerywana wskazuje na związek opcjonalny, tzn. *Każda <encja 1> może być*.. Typ związku jest zapisywany nazwą umieszczoną w pobliżu encji. Konstruując dalej, zapiszemy:

Każda <encja 1> musi być <nazwa związku 1>.

Drugi koniec linii reprezentującej związek może stanowić pojedynczą linię, co wskazuje na związek z dokładnie jednym egzemplarzem drugiej encji. Kurza łapka - rozgałęzienie na trzy linie (ang. *crows feet*) na danym końcu oznacza natomiast związek z jednym lub większą liczbą egzemplarzy. Oba warianty wystąpią w naszym zdaniu jako:

Każda <encja 1> musi być <nazwa związku 1> jednej i tylko jednej <encja 2>.

lub

Każda <encja 1> musi być <nazwa związku 1> jednej lub więcej <encja 2>.

Drugi związek odczytujemy w podobny sposób tylko w przeciwną stronę tzn. od encji 2 do 1.

Związek miedzy encjami klienta i towaru na rys. 1 można odczytać jako:

Każdy klient może być nabywcą jednego lub więcej towarów.

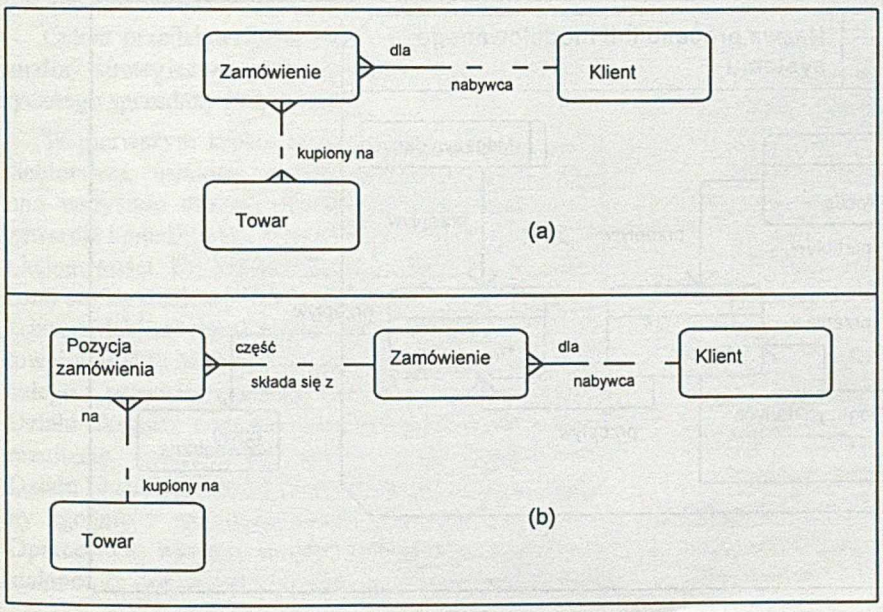
lub

Każdy towar musi być zakupiony przez jednego lub więcej klientów.

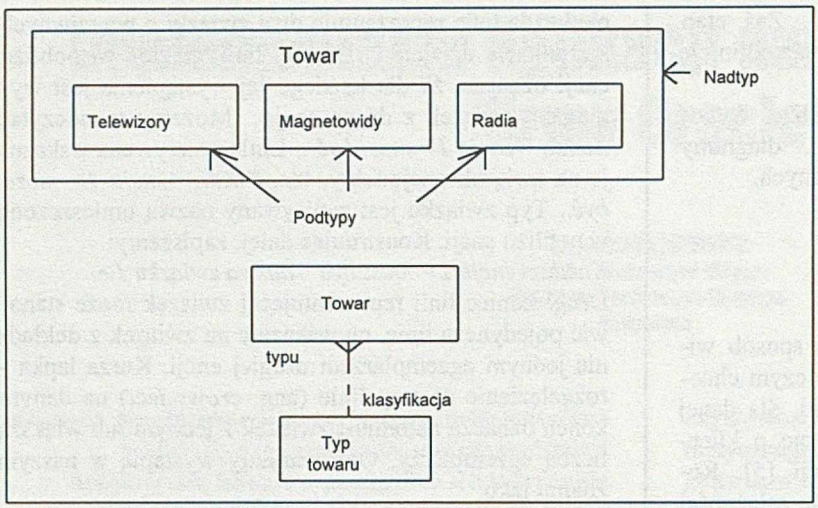
Istnieje kilka rodzajów związków encji:

- Jeden-do-jednego: Rzadki rodzaj. Zwykle wskazuje na niezrozumienie jednej lub obu encji.
- Jeden-do-wielu: Najpopularniejszy. Zwykle po stronie kurzej łapki jest wymagany, a po stronie przeciwnej opcjonalny.
- Wiele-do-wielu: Ten typ jest także popularny. W takich przypadkach należy dążyć do wprowadzenia *pośredniczącej* encji między dwoma związanymi encjami. Encja pośrednicząca często ma swoją własną nazwę. Na przykład, związek wiele-do-wielu między klientami i towarami może być rozbity w sposób pokazany na rysunku 2. Encją pośredniczącą jest tutaj zamówienie.

Rys. 2. Encje pośredniczące [5]



Niektóre encje mogą zawierać podklasy. W pewnym przedsiębiorstwie można wyróżnić, na przykład, podklasy: telewizory, magnetowidy, radia. Są one podtypami encji Towary (rys.3). Encję Towary nazywa się nadtypem. Podtypy wykazuje się na diagramie jako encje wewnątrz encji nadtypu. Podtypy dziedziczą wszystkie atrybuty i związki nadtypu. Jeżeli np. towar ma jako atrybut nr katalogowy, to podtypy telewizory, radia i magnetowidy odziedziczą ten



Rys. 3. Podtypy encji towary

atrybut. Atrybuty można obejrzeć korzystając z atrybutów domyślnych w CASE*Dictionary. W przypadku większej liczby podklas diagram może być nieczytelny. Zaleca się wtedy utworzyć nową encję.

Hierarchia funkcji

Hierarchia funkcji służy do reprezentacji zadań organizacji, tzn. tego co dana firma robi lub powinna robić. Funkcje wyższego poziomu opisują szersze dziedziny niż funkcje niskich poziomów. Funkcja najwyższego poziomu określa cel systemu rzeczywistego. Najniższy poziom zawiera takie funkcje, które nie mogą być dalej rozkładane, tzw. elementarne funkcje. Pakiet CASE*Designer firmy Oracle zawiera moduł kreślenia diagramów do konstruowania hierarchii funkcji.

Opis funkcji należy rozpoczynać czasownikiem, po którym powinien nastąpić rzeczownik stanowiący dopełnienie. Rzeczowniki występujące w opisie funkcji to encje, stany encji lub atrybuty encji.

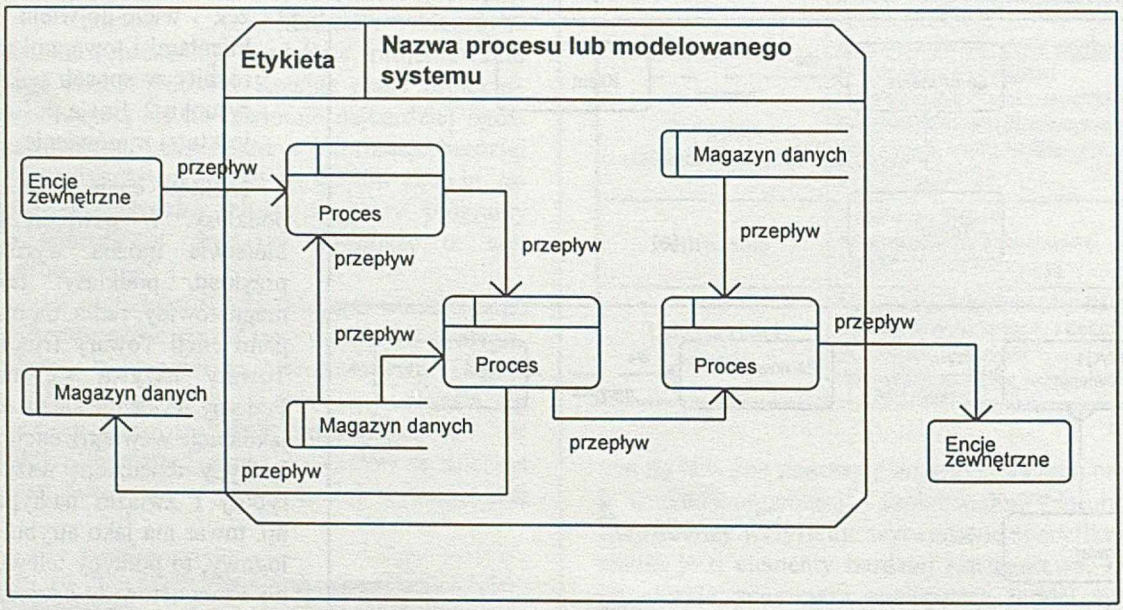
Podczas modelowania obiektów rzeczywistych można otrzymać bardzo dużą liczbę funkcji. Jedną z metod zmniejszania listy funkcji modelu to korzystanie ze *zdarzeń* (ang. *events*). Zdarzenia uaktywniają jedną lub więcej funkcji. Istnienie funkcji, która nie jest uaktywniana ani zdarzeniem, ani przez inną funkcję, wskazuje na nieuwzględnienie pewnej informacji. Funkcja ta może też być redundantna. Efektem wykonania funkcji może z kolei być zdarzenie. Zdarzenia uaktywniające i uaktywniane funkcje definiuje się dla każdej funkcji za pomocą CASE*Dictionary. Zdefiniowanie zdarzenia i określenie funkcji, które wywołuje, jest dobrym testem kompletności.

Diagram przepływu danych

Diagram przepływu danych jest to model pokazujący, w jaki sposób dane przepływają między funkcjami i jak są przez nie używane. Funkcje mogą być ręczne, zautomatyzowane lub stanowić ich kombinację [5]. Na rys. 4 wyróżniono elementy składające się na diagram przepływu danych:

- procesy, odpowiadające funkcjom z hierarchii; za ich pomocą opisuje się w jaki sposób przetwarza się dane wejściowe na wyjściowe,
- encje zewnętrzne, dostarczające systemowi danych wejściowych i odbierające dane wyjściowe; czasami nazywa się je źródłami oraz odpływami,
- przepływy, pokazujące ruch danych między procesami i magazynami danych; są one rysowane jako strzałki łączące dwa procesy, z grotem wskazującym

Rys. 4. Diagram przepływu danych [1]



kierunek przepływu,

- magazyny danych, służące do czasowego przechowywania informacji, również do późniejszego wykorzystania; zgromadzona w nich informacja stanowi podstawę związków encji.

Diagram przepływu danych buduje się poziomami. Wyższy poziom ustala kontekst systemowy, każdy następny zaś dodaje więcej szczegółów. W ten sposób najwyższy poziom, zwany poziomem kontekstu, zawiera pojedynczy proces oraz wszystkie wchodzące i wychodzące przepływy zewnętrzne. Bezpośrednio niższy następny poziom rozdziela ten proces na kilka głównych procesów składowych, a na kolejnym niższym poziomie każdy jest rozbijany na bardziej szczegółowe itd. Rozbicie procesu w diagramie przepływu danych odpowiada dekompozycji funkcyjnej w hierarchiach funkcji, używanych podczas strategii. Sprawdzenie, czy rozbicie niższego poziomu uwzględnia wszystkie przepływy danych z poziomu wyższego, nazywamy balansowaniem poziomów [5].

CASE*Designer zawiera narzędzia do rysowania diagramu przepływu danych zstępująco, tzn. przez rozbijanie procesów wyższego poziomu. Definicje procesów, przepływów i magazynów danych są tworzone w CASE*Dictionary. Podczas rysowania diagramów przepływu obiekty te są automatycznie zapisywane w bazie danych CASE*Dictionary. Przepływy danych i magazyny są opisywane w CASE*Dictionary w terminach encji i atrybutów. Dzięki temu podczas rysowania diagramów wystarczy nadać im nazwy. Natomiast zdefiniowanie ich treści musi być poprzedzone pełnym opisaniem diagramu związków encji. Zadanie to ułatwia diagram związków encji z fazy strategii.

Studium problemowe: System sprzedaży eksportowej

W tej części artykułu zostanie przedstawione studium strategiczne dla przedsiębiorstwa przemysłowego produkującego kable.

Celem przedsięwzięcia, który uzgodniono na etapie analizy strategicznej, jest wdrożenie systemu informatycznego sprzedaży eksportowej.

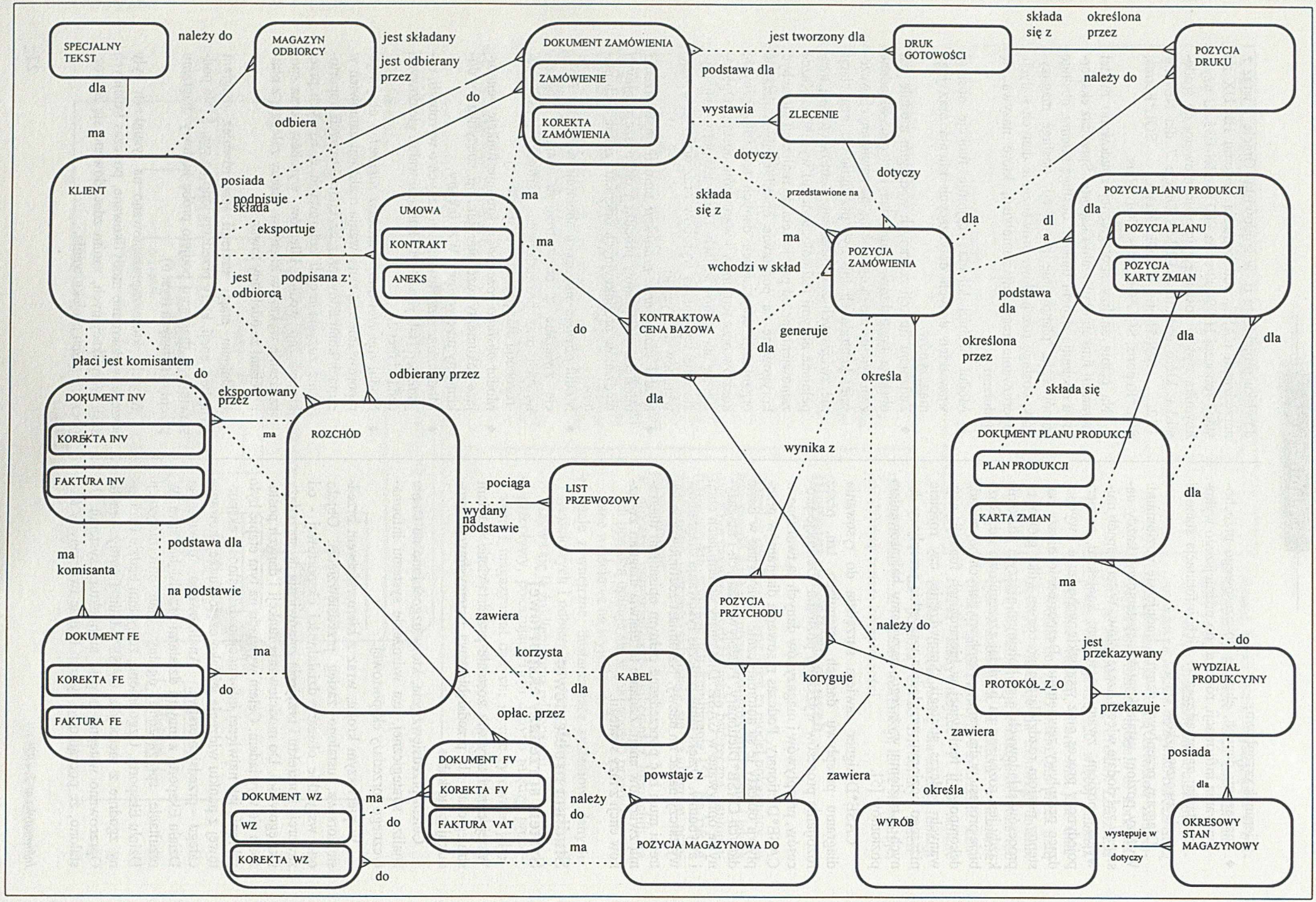
W pierwszym kroku, wraz z kierownictwem przedsiębiorstwa, ustalono zasięg przedsięwzięcia. Objęło ono wszystkie obszary działalności eksportowej - od zawarcia kontraktu i przyjęcia zamówienia do sprzedaży i księgowości. Do modelowania funkcji i danych posłużono się wywiadem. Celem wywiadu na tym etapie było zebranie jak najwięcej informacji o działalności eksportowej z punktu widzenia jej jako części funkcjonowania całego przedsiębiorstwa. Analizowano powiązania Działu Eksportu z innymi działami przedsiębiorstwa. W rezultacie sporządzono ogólny plan informatyzacji Działu Eksportu. Uzgodniono, że system będzie tworzony zgodnie z metodyką CASE*Method firmy Oracle. Opracowano wstępny projekt architektury systemu. Ustalono, że początkowo system będzie tworzony w środowisku projektanta tj. w środowisku Oracle 7 wraz z narzędziami CASE, pod kontrolą systemu HP UX na stacji roboczej HP 9000 seria 800 Model E35. U użytkownika system zostanie zaimplementowany do środowiska, które obejmie RDBMS Oracle 7 dla Novella 3.12, SQL*Net, SQL*Menu, SQL*Forms, SQL*Reports, SQL*Graphics, SQL*Plus.

Na etapie analizy strategicznej zbudowano model hierarchii funkcji dla Działu Eksportu. Sprzedaż eksportowa obejmuje siedem funkcji: obsługę klienta, obsługę zamówienia, planowanie produkcji, obsługę magazynów, obsługę rozchodów, prowadzenie danych stałych, raportowanie realizacji zamówień, które nazywa się modułami.

- Moduł obsługi klienta obejmuje funkcje obsługi wszystkich danych dotyczących klienta oraz jego magazynów.
- Zadaniem funkcji zebranych w drugim module jest obsługa kontraktów, zamówień i zleceń. Składane zamówienia mogą opierać się na zawieranych wcześniej kontraktach lub być przedmiotem negocjacji nie związanych z kontraktem. Zamówienia są uzupełniane danymi technologicznymi wyrobów. Każde zamówienie przed przyjęciem do realizacji może być korygowane na podstawie opinii różnych działów organizacji dotyczących danych technologicznych, kosztów oraz terminów i warunków dostaw. Opierając się na zamówieniach Dział Eksportu generuje zlecenia wykonania wyrobów oraz druki gotowości wyrobów do wysyłki.
- Moduł *planowanie produkcji* umożliwia tworzenie miesięcznych planów produkcji dla poszczególnych wydziałów produkcyjnych oraz kart zmian tych planów.
- Moduł *obsługa magazynu* obejmuje wszystkie funkcje związane z prowadzeniem stanów magazynowych oraz prowadzeniem kartoteki przychodu-rozchodu dla sprzedaży eksportowej.
- Moduł *obsługa rozchodów* grupuje funkcje generujące dokumenty rozchodu, jak. np. specyfikacje, dokumenty przewozowe, WZ, faktury.
- Moduł *dane stałe* to obsługa danych o wyrobach, organizacji (fabryka kabli), kursami walut, giełdowymi cenami metali.
- Moduł *raportowanie realizacji zamówień* obejmuje raportowanie zaawansowania realizacji zamówień w ramach kontraktów oraz bezkontraktowych, raportowanie zużycia metali kolorowych w różnych przekrojach (np. według klientów, wyrobów) oraz sporządzanie wykazów klientów, którzy zalegają z płatnościami za odebrany towar.

Produktem etapu strategii jest również diagram związków encji, który przedstawiono na rys. 5. Na podstawie tego diagramu podjęto próbę budowy diagramu przepływu danych (rys. 6).

Modele, które zaprezentowano na rysunkach były jeszcze wielokrotnie modyfikowane, podczas kolejnych sesji weryfikacyjnych, zanim zdecydowano się na rozpoczęcie etapu projektowania.



Rys.5. Diagram związków encji

Stosowanie metodologii i narzędzi CASE w znacznym stopniu przyczyniło się do eliminowania błędów w oprogramowaniu i skrócenia czasu implementacji systemu sprzedaży eksportowej. Wśród wielu czynników, które warunkują osiąganie sukcesu we wdrażaniu systemów informatycznych, bardzo ważną rolę odegrały wnikliwie prowadzone sesje weryfikacyjne, które pozwoliły szczegółowo poznać wymagania użytkowników. Dlatego nie należy rezygnować ze stosunkowo długotrwałego modelowania, proponowanego przez CASE*Method na etapach strategii i analizy, mimo nacisków wywieranych przez użytkowników na szybką realizację oprogramowania. Etapy te mają bowiem spowodować nie tylko akceptację systemu informatycznego przez użytkowników, ale także ich żywe zainteresowanie rozwojem systemu, przez współuczestnictwo w podejmowaniu decyzji sterujących rozwojem. Sesje weryfikacyjne, w których aktywnie uczestniczą użytkownicy tworzonych systemów, często prowadzą do twórczych rozwiązań w zakresie funkcjonowania różnych obszarów działalności organizacji. Odkrywane są nowe potrzeby w zakresie informatyzacji. W przypadku tworzenia systemu informatycznego sprzedaży eksportowej pojawiło się m.in. zapotrzebowanie na wdrożenie systemu informatycznego w Dziale Opakowań i magazynach.

LITERATURA

[1] Barker R., Longman C.: CASE*Method. Function and Process Modelling. Addison-Wesley Publishing Company, 1992

[2] Barker R.: CASE*Method. Tasks and Deliverables. Addison-Wesley Publishing Company, 1991

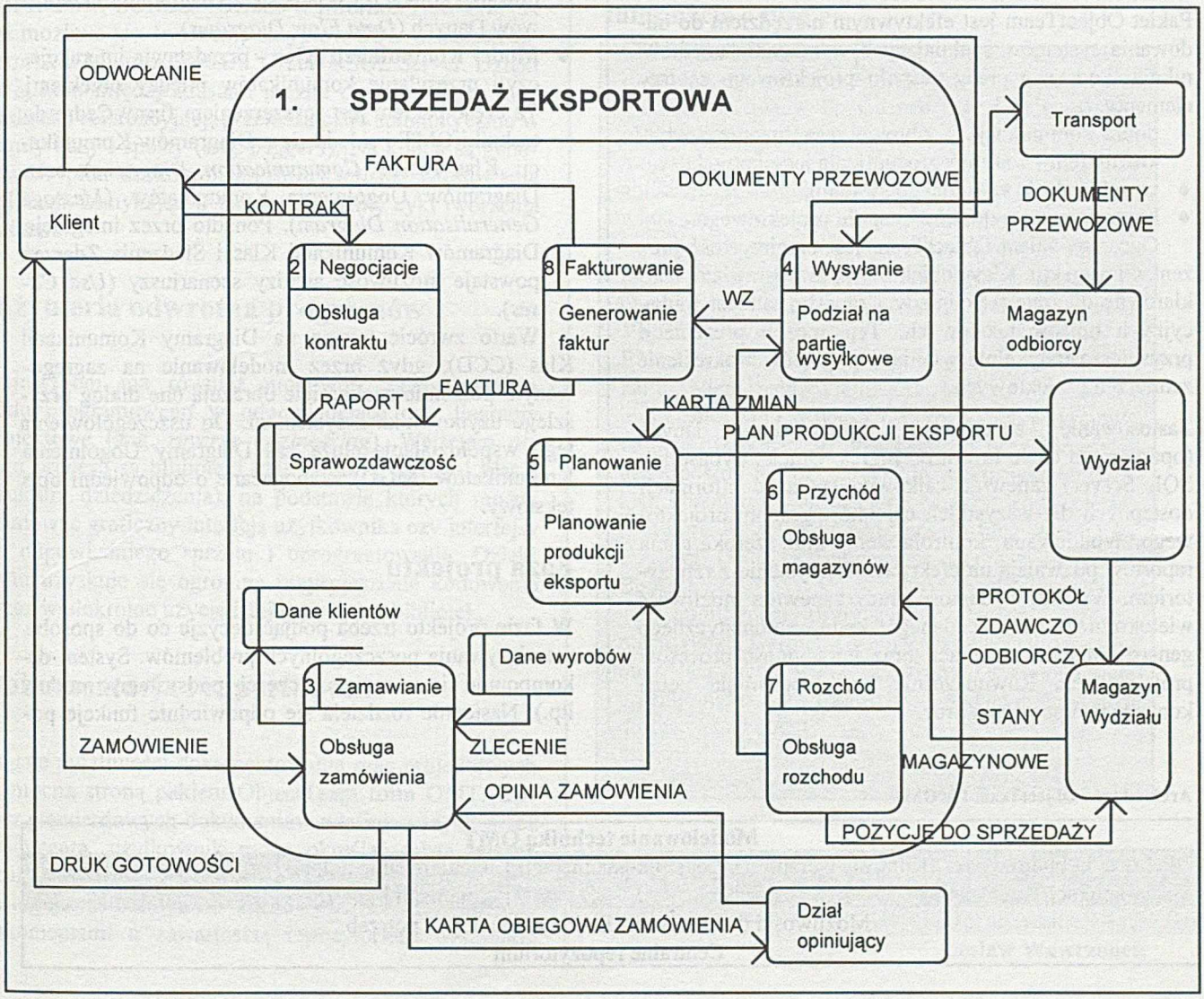
[3] Hołodnik-Janczura G., Płonka-Szydlak E., Zabawska E.: Realizacja przedsięwzięć informatycznych z wykorzystaniem metodyki CASE firmy Oracle. Informatyka nr 2,1994

[4] Hołodnik-Janczura G., Płonka-Szydlak E., Zabawska E.: CASE firmy Oracle do analizy organizacji oraz definiowania strategii informatyzacji. Informatyka nr 2, 1995

[5] Rodgers U.: Oracle. Przewodnik projektanta baz danych. Wydawnictwa Naukowo-Techniczne, 1995.

W jednym z następnych numerów zostanie zamieszczony artykuł o nowej wersji Oracle CASE-Designer/2000.

Rys. 6. Diagram przepływu danych



ObjectTeam for OMT

ObjectTeam for OMT firmy Cadre jest środowiskiem narzędziowym do budowania obiektowych systemów aplikacyjnych. Rozwijane przez Cadre od 1988 roku technologie obiektowe OO (*Object Oriented*) są oparte na międzynarodowych standardach narzędzi CASE.

Metodyka OMT (*Object Modelling Technique*) zaproponowana przez J.Rumbaugha, wraz z uzupełnieniami firmy Cadre, jest podstawą opracowania modeli analitycznych i projektowych dla skomplikowanych systemów technicznych i ekonomicznych.

Ważną cechą ObjectTeam for OMT jest jego elastyczna adaptacja do wymagań użytkownika. Z łatwością można nie tylko zmienić odpowiednio domyślną konfigurację produktów, ale także uzupełnić je wieloma innymi narzędziami i bibliotekami klas ogólnie dostępnymi na rynku.

Możliwości ObjectTeam

Pakiet ObjectTeam jest efektywnym narzędziem do budowania systemów spełniających wymagania użytkowników. Podstawą pracy zespołu projektowego są trzy elementy:

- dobra komunikacja w obrębie zespołu oraz z użytkownikiem - warunek zrozumienia jego potrzeb,
- wysoka jakość tworzonego systemu,
- kontrola pracy członków zespołu projektowego.

Ogromną zaletą ObjectTeam jest przejrzystość prezentacji projektu. Klasy obiektów są przedstawiane jako klarowne diagramy, podglądy i raporty, zamiast tradycyjnych opisów tekstowych. Ten sposób prezentacji przyspiesza radykalnie wymianę poglądów i określenie zamierzeń projektowych.

Zastosowanie centralnego repozytorium danych (opartego na bazie Informix, Ingres, Oracle, Sybase lub SQL Server) zapewnia całkowitą spójność informacji dostępnych dla wszystkich członków zespołu projektowego. Wbudowana kontrola wersji oraz szeroka gama raportów pozwalają na efektywne korzystanie z repozytorium. Wysoką wydajność pracy zapewnia możliwość wielokrotnego użycia danego kodu, automatycznego generowania dokumentacji oraz iteracyjność procesów projektowych. Równocześnie jest zapewniona pełna kontrola nad postępem prac.

Faza analizy

Praca w ramach cyklu rozwojowego aplikacji jest podzielona na fazy. We wszystkich fazach jest stosowany wspólny zestaw technik. Użycie tych samych modeli we wszystkich fazach umożliwia budowanie oprogramowania w sposób ewolucyjny.

W ramach projektu powstaje kilka współzależnych modeli opisujących różne aspekty systemu:

- **Model obiektów** - przedstawia klasy i ich relacje, tworząc podstawę do analizy i projektu. W skład modelu wchodzą Diagramy Związków Klas (*Class Association Diagrams*).
- **Model Dynamiczny** - reprezentuje zachowania elementów systemu oraz kolejność operacji. Składa się z Diagramów Śledzenia Zdarzeń (*Event Trace Diagrams*) oraz Diagramów Stanów-Przejść (*State Transition Diagrams*).
- **Model Funkcjonalny** - obrazuje funkcjonalność budowanej aplikacji i składa się z Diagramów Przepływów Danych (*Data Flow Diagrams*).
- **Model Komunikacji Klas** - przedstawia interakcje, czyli przesyłanie komunikatów między obiektami. Ta część pakietu jest rozszerzeniem firmy Cadre do techniki OMT i składa się z Diagramów Komunikacji Klas (*Class Communication Diagrams*) oraz Diagramów Uogólnienia Komunikatów (*Message Generalisation Diagram*). Ponadto przez integrację Diagramów Komunikacji Klas i Śledzenia Zdarzeń powstaje możliwość analizy scenariuszy (*Use Cases*).

Warto zwrócić uwagę na Diagramy Komunikacji Klas (CCD), gdyż przez modelowanie na zagregowanym poziomie znakomicie obrazują one dialog przyszłego użytkownika z systemem. Do uszczegółowienia tego współdziałania służą zaś Diagramy Uogólnienia Komunikatów (MGD), wzbogacane o odpowiedni opis tekstowy.

Faza projektu

W fazie projektu trzeba podjąć decyzje co do sposobu rozwiązywania poszczególnych problemów. System dekomponuje się na mniejsze części (podsystemy, moduły itp.). Następnie rozdziela się odpowiednie funkcje po-

Architektura ObjectTeam for OMT

Modelowanie techniką OMT				
Pełny cykl budowy aplikacji	Kontrola wersji	Konstrukcja aplikacji	Generowanie dokumentacji i raporty	Otwarty interfejs
Możliwości dostosowania do indywidualnych potrzeb				
Centralne repozytorium				

między oprogramowanie klienta i serwera, optymalizując je przez minimalizację wzajemnych interakcji.Szczególnie przydatny jest do tego Model Komunikacji Klas, pokazujący zachowania obiektów. Jego uzupełnieniem są Diagramy Przepływów Danych.

Faza projektowania obiektów i implementacji

W fazie projektowania obiektów projekt systemu jest uszczegółowiany, tworząc podstawę do implementacji. Opracowane poprzednio modele są optymalizowane, ulepszane i rozszerzane. Narzędzia ObjectTeam służą do zapewnienia spójności między początkowymi modelami a ich wersjami szczegółowymi.
Następnie w kolejnych iteracjach diagramy są transformowane do postaci kodu, przy użyciu bogatego generatora kodu. Jedną z jego znakomitych możliwości jest to, że wszelkie zmiany w projekcie systemu są automatycznie przenoszone do odpowiednich fragmentów kodu.

Programy mogą być generowane w językach C++, ESQL, Smalltalk, Ada (83+95) Java, Visual Basic, Power Builder, CORBA_IDL lub w Informix New Era, jak też możliwe jest tworzenie skryptów SQL w bazach Informix, CA-Ingres, Sybase, Oracle i SQL Server. Dzięki związkom między klasami można określić więzy integralności referencyjnej, przekształcane automatycznie w reguły bazy i spusty (ang. *triggers*). Na końcu są generowane pliki do kompilacji, uruchamiania i testowania aplikacji, zamykając w ten sposób pełny cykl tworzenia systemu.

Inżynieria odwrotna programów

ObjectTeam ma również możliwość przekształcania kodu programowego w odpowiadające mu diagramy projektowe (ang. *reverse engineering*). Wejściem do tego procesu są biblioteki klas (wraz z cechami klas i strukturą dziedziczenia), na podstawie których można zbudować graficzny interfejs użytkownika czy interfejsy do odpowiedniego sprzętu i oprogramowania. Dzięki temu uzyskuje się ogromne przyspieszenie kodowania przez wielokrotne użycie istniejących już bibliotek.

Dokumentacja i raportowanie

Bogate możliwości dokumentowania prac projektowych są mocną stroną pakietu ObjectTeam form OMT. Oprócz standardowych dokumentów zdefiniowanych przez producenta, użytkownik może określać także własne struktury dokumentów. Zaś na szczególne podkreślenie zasługuje automatyczne zachowywanie więzi między dokumentami a zawartością repozytorium: wszystkie zmiany projektowe są natychmiast uwzględ-niane w generowanym dokumencie.

Narzędzie Docwriter działa w połączeniu z pakietem do przygotowywania publikacji (DTP). Dockwriter jest zintegrowany z pakietami FrameMaker, Interleaf, WordPerfect i Word; można też korzystać z innych pakietów. Raporty są tworzone przy użyciu narzędzia Report Writer, którego język jest zbliżony do SQL.

Platformy narzędziowe i skalowalność pakietu

ObjectTeam for OMT jest zbudowany w architekturze klient-serwer, gdzie różne zadania mogą być realizowane na różnych komputerach. Oczywiście cała konfiguracja wykorzystywana przez zespół projektowy może pracować w sieci TCP/IP lub NFS, ze stacjami roboczymi od strony klienta, pracującymi z wykorzystaniem Windows 95, Windows NT i UNIX oraz serwerami firm: DEC, HP, IBM, ICL, Siemens, Sun zarządzane systemami Windows NT i UNIX.

Informacja o firmie:

Założona w 1983 roku firma Cadre Technologies Inc. (z siedzibą w Providence, Rhode Island) jest czołowym na świecie dostawcą narzędzi do programowania i usług w zakresie pełnego cyklu budowy aplikacji. Współpracujące ze sobą pakiety umożliwiają stosowanie dwóch najpopularniejszych metod tworzenia oprogramowania: obiektowej i strukturalnej. Platformą systemową mogą być stacje robocze pracujące pod SunOS, Solaris, AIX, HP-UX, Digital VMS i MS Windows.
Biura techniczne firmy znajdują się w USA, Holandii, Niemczech, Wielkiej Brytanii i Australii, zaś biura handlowe - w większości rozwiniętych krajów. Istnieje już ponad 50 tys. instalacji produktów Cadre. Spośród ich użytkowników warto wymienić takich potentatów, jak: General Motors, Kodak, IBM, Motorola, AT&T, KLM Airlines, Lufthansa, Bank of England, Deutsche Bundespost, Siemens, Mercedes Benz.

W opinii znanej firmy specjalizującej się w badaniach rynku Gartner Group „Cadre jest liderem w budowie systemów OO", zaś firma IDC o podobnym profilu uważa, iż „Cadre jest wiodącym na rynku dostawcą OO CASE".

Opracował:
Lesław Wawrzonek

Współpraca pomiędzy aplikacjami w Microsoft Office Professional dla Windows 95

Michał Gołębiewski
Microsoft Sp. z o.o.

Microsoft Office jest najpopularniejszym na świecie pakietem biurowym. Mimo swojej nazwy, może być on również stosowany przez prawników, inżynierów, naukowców, ludzi interesu itp. Jednym słowem jest on przydatny wszędzie tam, gdzie potrzeba tworzyć profesjonalnie wyglądające dokumenty, zawierające dane różnego rodzaju.

Na rynku pojawiła się niedawno polska wersja Microsoft Office Professional dla Windows 95. Jedną z jej najważniejszych cech jest bardzo daleko idąca integracja poszczególnych programów oraz możliwość wymiany informacji pomiędzy nimi. Tym zagadnieniom pragnę poświęcić ten artykuł.

Microsoft Office – rozwiązanie kompleksowe

Microsoft Office oferuje praktycznie wszystkie narzędzia, które mogą być przydatne w codziennej pracy. W jego skład wchodzi bowiem edytor tekstów Microsoft Word, arkusz kalkulacyjny Microsoft Excel, baza danych Microsoft Access, program do tworzenia prezentacji Microsoft PowerPoint oraz kalendarz / agenda Microsoft Schedule+. Ta piątka uzupełniona jest tzw. Spinaczem (ang. Binder), który umożliwia przechowywanie w jednym pliku kilku dokumentów oraz licznymi serwerami OLE – m.in. WordArt, Graph, Data Map, Equation, Organizaton Chart.

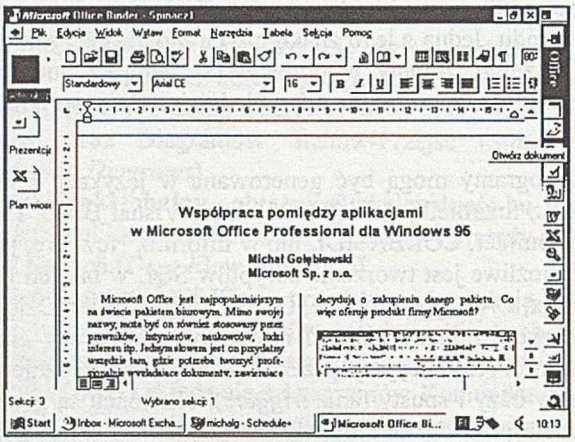
Dostępne narzędzia pozwalają na stworzenie dowolnie skomplikowanych dokumentów bez korzystania z dodatkowych aplikacji. Oferowane mechanizmy z jednej strony są bardzo profesjonalne (tabele przestawne, analiza statystyczna, tworzenie aplikacji za pomocą Visual Basic for Application itp.), natomiast z drugiej – łatwe w użyciu. Wiele operacji może być wspomaganych poprzez moduły typu Kreator, które krok po kroku prowadzą użytkownika z rękę oraz wykonywanych automatycznie dzięki mechanizmowi IntelliSense.

Jest to więc rozwiązanie kompleksowe, a na dodatek oferujące doskonałą integrację pomiędzy programami.

Integracja pomiędzy aplikacjami

Nawet najlepsze narzędzia, jeśli nie potrafią odpowiednio porozumiewać się ze sobą, są bezużyteczne. Integracje i współpraca są jednymi z tych czynników, które wielokrotnie decydują o zakupieniu danego pakietu. Co więc oferuje produkt firmy Microsoft?

Przede wszystkim aplikacje z pakietu Microsoft Office wyglądają i zachowują się bardzo podobnie. Zastosowany został ten sam wygląd ekranu, podobne listwy narzędzi i skróty klawiszowe, taka sama terminologia. Jeśli więc użytkownik nauczy się obsługiwać jeden z programów, wówczas nie będzie miał problemu z pozostałymi.



Istnieje również możliwość uruchamiania jednej aplikacji z poziomu drugiej. Np. wstawiając do dokumentu Worda tabelę Excela, automatycznie uruchamia się arkusz kalkulacyjny. Co więcej, wiele mechanizmów wykorzystywanych przez programy są wspólne – słownik ortograficzny, autokorekta, filtry do plików, obsługa zbiorów itp.

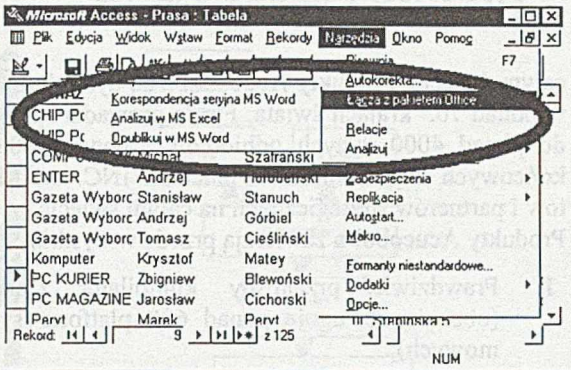
W codziennej pracy bardzo przydatny okazuje się Pasek skrótów Microsoft Office – niewielka aplikacja, która spełnia rolę czegoś w rodzaju centrum dowodzenia. Jest to po prostu pasek z ikonami, który zawsze znajduje się na wierzchu pozostałych okien. Ikony, które się na nim znajdują mogą być dowolnie modyfikowane, jednak najbardziej przydatne są dwie z nich – Zacznij nowy dokument i Otwórz dokument.

Tak więc integracja programów poszła bardzo daleko i wprawdzie są to oddzielne aplikacje, jednak zachowują się one tak, jakby stanowiły nierozerwalną całość.

Metody wymiany informacji

Z uwagi na fakt, że Microsoft Office pracuje w Windows 95, oferuje on standardowe dla tego systemu mechanizmy wymiany danych – OLE i DDE.

Ale pakiet ten może jeszcze więcej, zaimplementowano w nim bowiem mechanizm OfficeLinks. Jest on odpowiedzialny za tworzenie połączeń między dokumentami pochodzącymi z różnych aplikacji, jak również za możliwość przenoszenia danych pomiędzy programami.



Najlepiej widocznym przykładem zastosowania tego mechanizmu jest Microsoft Binder (Spinacz). Dzięki niemu można z kilku plików tworzyć jeden dokument. Może on zawierać list z Worda, arkusz z Excela czy prezentację z PowerPointa. Oferowana przez Spinacz metoda grupowania tematycznego dokumentów może w znaczny sposób przyczynić się do zaprowadzenia porządku na dysku.

Ale mechanizm OfficeLinks pomaga również przy codziennej pracy – przykładami jego zastosowań jest tworzenie korespondencji seryjnej w Wordzie (adresy mogą pochodzić z Excela, Accessa lub Schedule+), analizowanie danych z Accessa za pomocą tabel przestawnych w Excelu, tworzenie raportów na podstawie informacji z arkusza przy wykorzystaniu mechanizmów bazy danych itp. Jednym słowem użytkownik może w pełni skupić się na wykonywanym zadaniu i nie musi zastanawiać się, czy lepiej wykorzystać Accessa czy Excela – on po prostu zaczyna pracę, a Microsoft Office podpowie, który program należy uruchomić.

Omawiając wymianę danych, nie można zapomnieć o pracy grupowej – nad niektórymi dokumentami musi jednocześnie pracować kilka osób. Również wówczas pakiet Microsoft Office będzie bardzo przydatny. Pozwala on bowiem na wprowadzanie przez różnych użytkowników poprawek do jednego dokumentu Worda i późniejsze ich porównywanie, tworzenie różnych wariantów arkuszy kalkulacyjnych a także pracę nad jedną bazą danych przez kilka osób. Dodatkowo kalendarz / agenda Schedule+ pozwala na przeglądanie i edytowanie terminarzy innych pracowników a także umożliwia organizowanie wspólnych spotkań i sam wyszukuje termin, który będzie odpowiadał wszystkim uczestnikom. Przy korzystaniu z Microsoft Exchange można automatycznie wysłać do współpracowników informację o takim spotkani i prosić ich o potwierdzenie uczestnictwa.

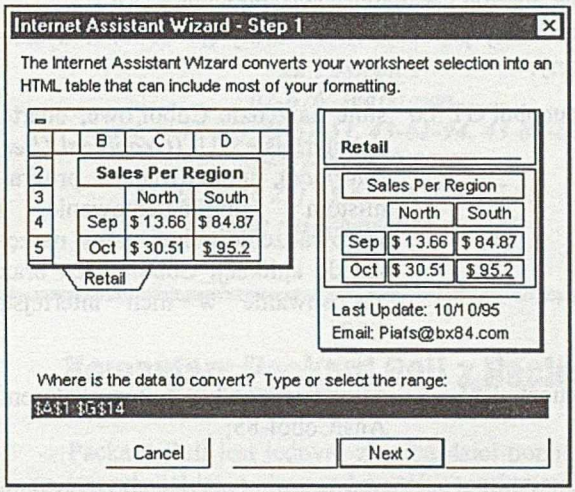
Wszystko to odbywa się w bardzo prosty sposób i z mechanizmów wymiany danych (zarówno pomiędzy aplikacjami, jak i osobami) może korzystać nawet początkujący użytkownik.

Internet i Intranet

Microsoft Office posiada jeszcze jedną, niezwykle istotną cechę – jest przygotowany do współpracy z Intranetem i Internetem.

Co to jest Intranet? To nic innego jak wewnętrzny (firmowy) Internet. Użytkownik ma dostęp do stron WWW, może ściągać pliki itp. Różnica w stosunku do Internetu polega jedynie na tym, że Intranet nie jest ogólnie dostępny i korzystać z niego mogą jedynie określone osoby (np. pracownicy i współpracownicy danej firmy).

Tak więc przeglądarki, narzędzia do tworzenia stron WWW oraz serwery są takie same dla obydwu sieci – inny jest jedynie odbiorca.



Wprawdzie Microsoft Office nie posiada wbudowanej możliwości tworzenia stron w języku HTML, jednak gotowe są już dodatkowe moduły (Wizards), które pozwalają na konwersję dokumentów na strony WWW. Dodatki te rozpowszechniane są za darmo i każdy może z nich korzystać.

Obecnie dostępny jest Internet Assistant dla Microsoft Word, Internet Wizard dla Microsoft Excel oraz Internet Wizard dla Microsoft PowerPoint. Tego typu moduł dla Microsoft Access jest w przygotowaniu i w najbliższym czasie również będzie dostępny w Internecie.

Tak krótki przegląd pakietu nie może ukazać wszystkich zalet płynących z jego używania. Zasygnalizowałem tu jedynie niektóre jego możliwości i mam nadzieję, że w ten sposób zachęciłem Państwa do bliższego zapoznania się z Microsoft Office.

Na koniec pragnę zaprosić wszystkich Czytelników Informatyki do odwiedzenia polskiej strony WWW firmy Microsoft. Oto adres:

http://www.microsoft.com/poland/

ACUCOBOL po 35 latach

FLEXGEN - wytwarzanie aplikacji „od ręki"

Acucobol INC jest firmą wiodącą w rozwiązaniach Cobolowych systemów otwartych i serwisowych. Produkty Acucobol'a są projektowane tak, aby zapewnić im i aplikacjom przez nie wytworzonym pełną zgodność pomiędzy wieloma platformami sprzętowymi i systemowymi.

Rdzenne produkty z rodziny Acucobol zostały zaprojektowane zgodnie z linią systemów otwartych przeznaczonych na różne platformy sprzętowe i systemowe. Każdy produkt występuje w wersji przystosowanej do setek maszyn i środowisk systemowych.
Do rodziny produktów Acu firmy Acucobol INC zaliczamy:

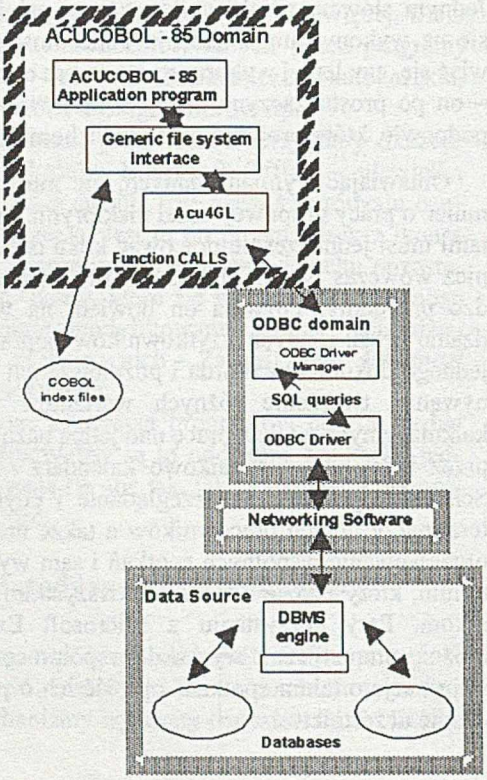
Acucobol GT 3.0	silne narzędzie Cobol'owe, oparte na interfejsie GUI (*Graphical User Interface*), umożliwiające programistom zaimplementowanie i wprowadzenie pełnej gamy narzędzi do aplikacji Cobolowych oraz zastosowanie w nich interfejsu GUI;
Acucobol-85	narzędzie zgodne z kompilatorem AnsiCobol-85;
Acu4GL	plastyczny interfejs do systemu zarządzania relacyjnymi bazami danych (RDBMS);
AcuScreens	graficzny interfejs z narzędziami do projektowania interfejsu użytkownika;
AcuEdit	w pełni zintegrowany, przenośny edytor do pisania programów Cobol'owych w standardzie ANSI lub trybie terminalowym;
AcuServer	produkt zapewniający pracę w technologii „klient-serwer".

Firma Acucobol świadczy również usługi konsultingowe w zakresie migracji aplikacji Cobol'owych na platformy systemów otwartych.

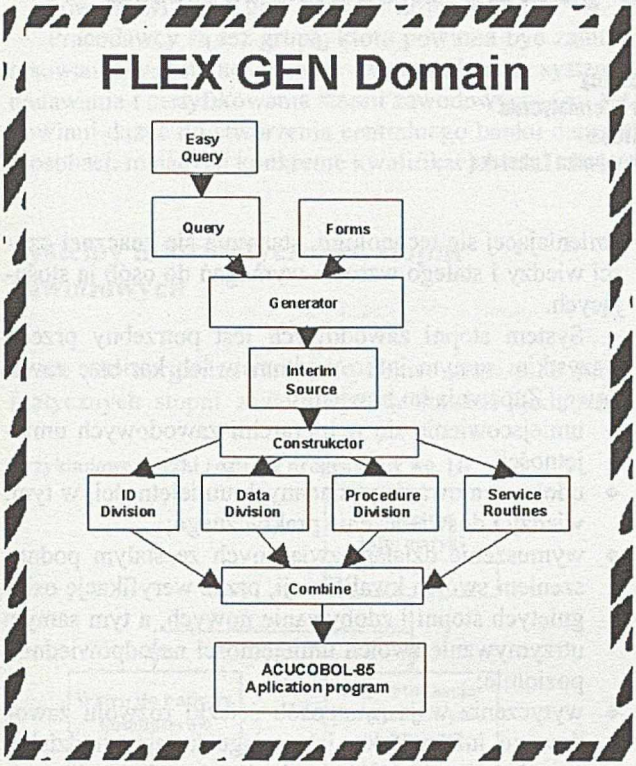
Główna siedziba firmy Acucobol znajduje się w San Diego w Kalifornii. Dodatkowo firma ma pięć oddziałów w następujących krajach: Niemczech, Irlandii, Włoszech, Skandynawii i Wielkiej Brytanii. W oddziałach tych jest zrzeszonych wielu dystrybutorów na całym świecie. Produkty Acucobol'a są dystrybuowane w ponad 70. krajach świata. Firma prowadzi sprzedaż do ponad 4000 dużych odbiorców i ponad 500.000 końcowych użytkowników. Acucobol INC. ma klientów i partnerów strategicznych na całym świecie.
Produkty Acucobol'a zawierają przede wszystkim:

1. Prawdziwie przenośny kompilator Cobol'a (obecnie obsługuje ponad 600 platform systemowych);
2. Narzędzia dynamicznie generujące interfejs użytkownika i narzędzia projektowe;
3. Interfejs relacyjnych baz danych (w trakcie załatwiania patentu w USA);
4. Wspomaganie obsługi sieci dla aplikacji wygenerowanych w Acucobol-85 (praca w technologii klient-serwer);
5. Graficzny interfejs użytkownika (GUI) dla języka Cobol;
6. W pełni zintegrowany pakiet graficzny dla języka Cobol.

ACUCOBOL z otoczeniem



FLEX GEN z otoczeniem



FLEX GEN jest narzędziem do wytwarzania aplikacji „od ręki". Zdefiniowana aplikacja może być wykonana bezpośrednio we FLEX GEN-ie lub może być stworzony kod źródłowy w Acucobol-u.

Przyjdź i spróbuj aplikacji od ręki !

Misją Acucobola jest gładkie przejście z aplikacjami w XXI wiek.

SAFECOMP' 96

15. Międzynarodowa Konferencja o Bezpieczeństwie, Niezawodności i Ochronie odbędzie się w dniach 23-25 października 1996 r. w Wiedniu (Austria), w hotelu Arcotel Wimberger.

Konferencja jest organizowana przez Austriackie Centrum Badań Naukowych Seibersdorf wspólnie z Federalnym Centrum Badań Naukowych i Testowania Arsenal.

SAFECOMP jest corocznym przeglądem aktualnego stanu wiedzy oraz doświadczeń w dziedzinie komputerowego bezpieczeństwa, niezawodności i ochrony. Została zainicjowana przez EWICSTC7 (*European Works hop on Industrial Computer Systems, Technical Committee* 7) w 1979 r. i od tego czasu odbyła się w Niemczech, W. Brytanii, USA, Włoszech, Francji, Austrii, Norwegii, Szwajcarii i Polsce. Konferencja koncentru- je się na krytycznych zastosowaniach komputera. Jej intencją jest stworzenie płaszczyzny wymiany technologii między wyższymi uczelniami, przemysłem i placówkami badawczymi. Zakres tematyczny konferencji obejmuje te wszystkie aspekty systemów komputerowych, których istotą jest bezpieczeństwo, niezawodność i ochrona, głównie w przemysłach lotniczym, kosmicznym, w procesach chemicznych, w transporcie kolejowym i drogowym oraz w placówkach badawczych. Językiem konferencji jest język angielski (bez symultanicznego tłumaczenia).

Adres do korespondencji:
E. Schoitsch, Austrian Research, Centre Seibersdorf,
A-2444 Seibersdorf, Austria
tel. +43 2254 780 3117, fax +43 2254 72133
email: schoitsch@zdfzs.arcs.ac.at

Komputery Packard Bell z Hectora

Packard-Bell jest jednym z najbardziej popularnych na amerykańskim rynku producentów markowych komputerów, adresowanych przede wszystkim do rynku SOHO. Obecnie komputery tej marki, cieszące się bardzo dobrą opinią, zaczęły pojawiać się i w Polsce.

Warszawska firma Hector S.A. oferuje komputery Packard Bell z bardzo atrakcyjnym wyposażeniem i oprogramowaniem. Można wybierać między procesorem Pentium 75 lub 100 Mhz, wszystkie komputery standardowo wyposażone są w pamięć 8 MB EDO RAM, dysk twardy 1 GB , kontroler PCI, pamięć cache 286 kB, I/0, napęd dysków elastycznych, podwójny kontroler dysków twardych ISA na płycie głownej, karta grafiki PCI SVGA - 5440 z pamięcią 1MB (możliwość rozszerzenia do 3 MB). Monitor kolorowy cyfrowy (2020) 15". Dodatkowe wyposażenie to CD-ROM, faksmodem 28.800 kbps, karta Sound Blaster 2.0, głośniki aktywne montowane do monitora, mikrofon. Do wyboru jest oprogramowanie Windows 95 lub DOS + Windows for Workgroups. Do tego dołączane są trzy pakiety użytkowe: edytor, obsługa łączności z Internetem, program do nauki języka angielskiego i bardzo atrakcyjny pakiet - Corel Draw. Firma Hector udziela na te komputery trzyletniej gwarancji, a ich cena ma być porównywalna z ceną komputerów niemarkowych. (k)

Informatyczne stopnie zawodowe i komputerowe prawo jazdy użytkownika

Piotr Fuglewicz
Polskie Towarzystwo Informatyczne
Marek Miłosz
Katedra Informatyki, Politechnika Lubelska

Dokończenie z pierwszej strony okładki

Pracowników, z punktu widzenia ewentualnego systemu stopni zawodowych, można podzielić na dwie kategrie:

- *zawodowców*, którzy profesjonalnie zajmują się tą lub inną dziedziną informatyki;
- *użytkowników*, dla których informatyka i jej środki są jedynie pomocą w ich podstawowej pracy.

Dla pracowników pierwszej grupy powinien istnieć ***system stopni zawodowych*** podobny do systemu specjalizacji lekarzy, choć może bardziej szczegółowy i precyzyjny. Użytkownicy powinni uzyskiwać rodzaj ***komputerowego prawa jazdy*** uprawniającego lub zaświadczającego, że jego posiadacz ma umiejętności wykorzystania technik informatycznych na wystarczającym poziomie.

Komu są potrzebne informatyczne stopnie zawodowe i „komputerowe prawo jazdy"?

Stopnie zawodowe „umiejscawiają" konkretnego informatyka wśród innych. Określą jego przynależność do grupy:

- programistów aplikacji,
- programistów systemowych,
- analityków i projektantów systemów informatycznych,
- administratorów systemów i sieci,
- kierowników projektów informatycznych,
- menedżerów przetwarzania danych,
- konserwatorów sprzętu komputerowego,
- wdrożeniowców

lub też jakiejś innej.

Umożliwiają także bardziej szczegółową klasyfikację zakresu i poziomu wiedzy teoretycznej a także doświadczenia danego informatyka. Inne umiejętności są bowiem potrzebne programistom aplikacji, pracujących np. w czasie rzeczywistym, a inne dla systemów baz danych.

Przy odpowiednio dobrze zorganizowanym systemie nadawania stopni zawodowych stanowią one jednocześnie pewną gwarancję i potwierdzenie poziomu umiejętności danej osoby. Raz osiągnięty stopień może wymagać okresowej (np. co 5 lub 10 lat) weryfikacji, na podobieństwo utraty uprawnień, przy braku praktyki, przez lekarzy. Weryfikacja ta jest konieczna w sytuacji szybko zmieniającej się technologii, starzenia się znacznej części wiedzy i stałego wzrostu wymagań do osób ją stosujących.

System stopni zawodowych jest potrzebny przede wszystkim samym informatykom w ich karierze zawodowej. Zapewnia im bowiem :

- umiejscowienie się w hierarchii zawodowych umiejętności;
- udokumentowanie posiadanych umiejętności, w tym: wiedzy i doświadczenia praktycznego;
- wymuszenie działań, związanych ze stałym podnoszeniem swoich kwalifikacji, przez weryfikację osiągniętych stopni i zdobywanie nowych, a tym samym utrzymywanie swoich umiejętności na odpowiednim poziomie;
- wytyczenie w jasny sposób ścieżki rozwoju zawodowego informatyka, jako ciągu kolejnych działań (np. kursów, studiów podyplomowych, realizowanych projektów itp.);
- możliwość planowania rozwoju własnego informatyka i dostosowania go do wymagań rynku.

System stopni zawodowych będzie miał zatem stymulującą rolę — będzie wymuszał na informatykach inne spojrzenie na własne kwalifikacje i uświadamiał konieczność ciągłego rozwoju.

Podobną rolę dla użytkowników systemów informatycznych może pełnić ***komputerowe prawo jazdy***. Jest ono rodzajem certyfikatu, świadczącego o posiadanych umiejętnościach w obsłudze systemów informatycznych. Certyfikat ten może być przeznaczony dla „średnich użytkowników" i bazować na ogólnie akceptowalnych wymaganiach czy też normach umiejętności. „Komputerowe prawo jazdy" użytkownika powinno być różne dla różnych kategorii użytkowników systemów informatycznych: inne dla zastosowań biurowych, inne dla inżynierskich itp. Odnawialność certyfikatu zapewnia jego stymulującą rolę, analogiczną dla stopni zawodowych.

Pracodawcy są grupą, która wydaje się być najbardziej zainteresowana we wdrożeniu sprawnego systemu informatycznych stopni zawodowych i „komputerowego prawa jazdy" dla użytkowników. Systemy te umożliwią przede wszystkim:

- jasną specyfikację wymagań w stosunku do pracowników;
- poszukiwanie, dobór i zatrudnianie pracowników, z kwalifikacjami adekwatnymi do potrzeb na danym stanowisku (nie za niskimi, ale także i nie za wysokimi);

- wzrost zaufania pracodawcy w stosunku do deklarowanych przez pracownika umiejętności;
- powiązanie płacy z poziomem umiejętności;
- planowanie procesu rozwoju kwalifikacji pracowników i powiązanie go z planami firmy.

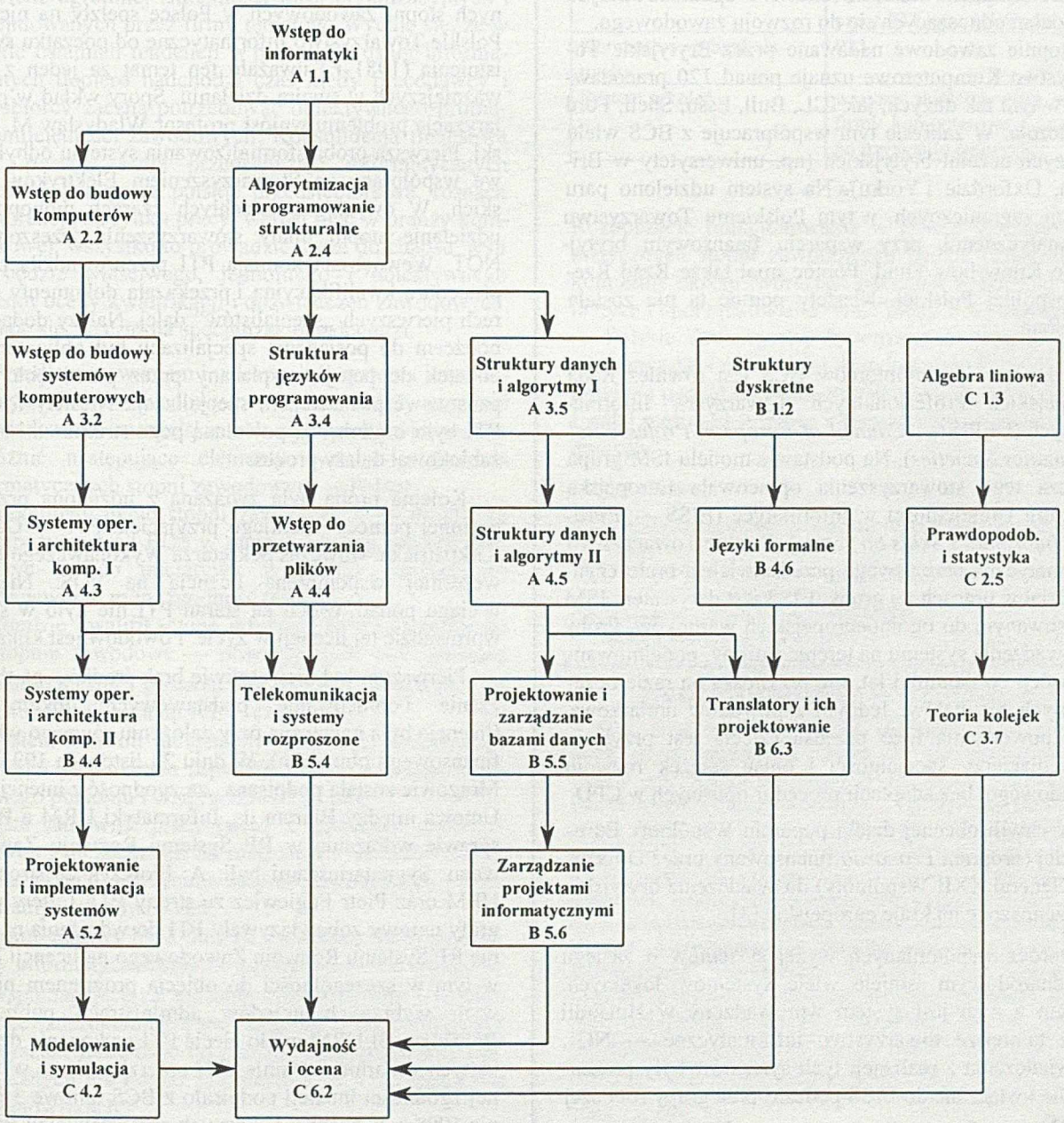
Pracodawcy są też grupą, która powinna być zainteresowana w rzetelności oraz wiarygodności systemu nadawania i certyfikowania stopni zawodowych. Oni też powinni dążyć do stworzenia centralnego banku danych o osobach mających konkretne kwalifikacje.

Systemy informatycznych stopni zawodowych

Prace nad zorganizowaniem systemu nadawania informatycznych stopni zawodowych w Stanach Zjednoczonych i Europie były podjęte dość dawno. Koordynowała je m.in. UNESCO [1], pod auspicjami której powstał modułowy system rozwoju zawodowo-naukowego w informatyce (*Modular Curriculum in Computer Science*). System ten składa się z modułów-specjalizacji informatycznych, powiązanych w skomplikowaną sieć. Ścieżki w tej sieci wyznaczają konkretne ścieżki rozwoju kariery zawodowej (i naukowej) pracownika. Istnieje więc ścieżka (a właściwie grupa ścieżek) programisty (aplikacji, systemowego i inżynierii oprogramowania), jak i np. ścieżka kierownika projektów informatycznych. Poszczególne ścieżki wytyczają następstwo i głębokość poznawanych dziedzin wiedzy, ukrytych za nazwami modułów (rys.). Nie da się ukryć, że opracowany w latach 80. system modułów nieco się już zestarzał.

System UNESCO był uzupełniony wymaganiami w stosunku do organizacji systemu szkoleń (wykładowcy, sprzęt i literatura).

Przykładowe ścieżki rozwoju programisty wg. [1]



W chwili obecnej do krajów europejskich najbardziej zaawansowanych w dziedzinie stopni zawodowych informatyków należy Wielka Brytania. W niej to został opracowany [2] i wdrożony przez Brytyjskie Towarzystwo Komputerowe (BCS — *British Computer Society*) System Rozwoju Zawodowego Informatyków (PDS — *Professional Development Scheme*). System ten jest powiązany z:

- Modelem Strukturalnym Zawodu Informatyka (ISM — *Industry Structure Model*), w którym wyróżniono ponad 200 różnych funkcji pracowników, związanych z systemami informatycznymi, na 10 poziomach umiejętności i doświadczenia;
- Komisją Egzaminacyjną Systemów Informatycznych (ISEB — *Information System Examination Board*), która przeprowadza weryfikacje i certyfikacje pracowników;
- Systemem Ustawicznego Rozwoju Zawodowego (CPD — *Continuous Professional Development*), jako układem kursów, szkoleń, spotkań i innych działań odnoszących się do rozwoju zawodowego.

Stopnie zawodowe nadawane przez Brytyjskie Towarzystwo Komputerowe uznaje ponad 120 pracodawców, w tym tak dużych, jak ICL, Bull, Esso, Shell, Ford i Motorola. W zakresie tym współpracuje z BCS wiele wyższych uczelni brytyjskich (np. uniwersytety w Brighton, Oxfordzie i Yorku). Na system udzielono paru licencji zagranicznych, w tym Polskiemu Towarzystwu Informatycznemu, przy wsparciu finansowym brytyjskiego Know-how Fund. Pomóc miał także Rząd Rzeczypospolitej Polskiej. Niestety pomoc ta nie została udzielona.

Jednym z licencjobiorców BCS jest również Rada Europejskich Profesjonalnych Towarzystw Informatycznych (CEPIS — *Council of European Professional Informatics Societies*). Na podstawie modelu ISM grupa robocza tego stowarzyszenia opracowała Europejską Strukturę Umiejętności w Informatyce (EISS — *European Informatics Skills Structure*). Polskie Towarzystwo Informatyczne przez swego przedstawiciela brało czynny udział w pracach tej grupy. EISS jest derywatem ISM dopasowanym do ogólnoeuropejskich warunków. Próby wprowadzenia systemu na terenie Europy, podejmowane od dwóch co najmniej lat, nie przynoszą na razie oczekiwanych rezultatów. Jednym z powodów umiarkowanego powodzenia tego przedsięwzięcia jest przyjęcie tylko macierzy specjalności i opisu ścieżek rozwoju zawodowego, bez adaptacji procedur opisanych w CPD.

W chwili obecnej dzięki poparciu Wspólnoty Europejskiej (program *Leonardo* finansowany przez Directorate General XXII Wspólnoty) doświadczenia brytyjskie są przenoszone na kraje europejskie [3].

Oprócz wspomnianych wyżej systemów o zasięgu ponadnarodowym istnieje wiele systemów lokalnych. Jednym z nich jest system wprowadzony w Holandii przez tamtejsze towarzystwo informatyczne — NGI. Doświadczenia z realizacji tych systemów były prezentowane i włączane do EISS podczas prac grupy roboczej CEPIS.

Inicjatywa wprowadzenia do praktyki certyfikatów dla użytkowników komputerów została podjęta w Finlandii w 1994 roku. W chwili obecnej trwa rozprzestrzenienie doświadczeń fińskich w całej Europie [3] — przy czym w pierwszej kolejności w krajach Skandynawskich, Irlandii i Holandii. Do prac nad opracowaniem „komputerowego prawa jazdy" przystąpiły także Austria, Francja i Wielka Brytania. Europeizacja fińskiego systemu dokonuje się również pod auspicjami CEPIS. Polska ma zamiar formalnie zgłosić swego przedstawiciela do prac odpowiedniej grupy roboczej w pierwszej połowie 1996 roku.

Organizacja systemu nadawania informatycznych stopni zawodowych w Polsce

Dotychczasowe próby wdrożenia systemu informatycznych stopni zawodowych w Polsce spełzły na niczym. Polskie Towarzystwo Informatyczne od początku swego istnienia (1981 r.) uważało ten temat za jeden z najważniejszych w swoim działaniu. Spory wkład w popularyzację problemu wniósł profesor Władysław M. Turski. Pierwsza próba sformalizowania systemu odbyła się we współpracy ze Stowarzyszeniem Elektryków Polskich. W owych zamierzchłych czasach monopol na udzielanie stopni miały stowarzyszenia zrzeszone w NOT. Wewnętrzna komisja PTI przeprowadziła pełną procedurę kwalifikacyjną i przekazała dokumenty czterech pierwszych „specjalistów" dalej. Należy dodać, że bodźcem do posiadania specjalizacji był obligatoryjny dodatek do pensji, wypłacany przez przedsiębiorstwa państwowe posiadaczom specjalizacji. Niestety fakt, iż PTI było organizacją powołaną poza strukturami NOT, zablokował dalszy proces.

Kolejna próba była związana z udzieloną przy ogromnej pomocy wielkiego przyjaciela PTI — Gavina Kirkpatricka, wówczas Sekretarza Wykonawczego BCS, wcześniej wspomnianą licencją na PDS. Niestety w ciągu ponad dwóch lat starań PTI nie było w stanie wprowadzić tej licencji w życie. Powodów jest kilka.

Pierwszym był niewątpliwie brak środków na tłumaczenie i opracowanie podstawowych dokumentów (licencja była nadawana przy założeniu równego wkładu finansowego obu stron). W dniu 23 listopada 1993 r. w Mrągowie została podpisana „za zgodność z intencjami" Umowa między Biurem ds. Informatyki URM a PTI w sprawie wdrażania w RP Systemu Rozwoju Zawodowego. Sygnatariuszami byli: A. Florczyk ze strony BI URM oraz Piotr Fuglewicz ze strony PTI. Cztery paragrafy umowy zobowiązywały PTI do wdrażania na terenie RP Systemu Rozwoju Zawodowego na licencji BCS, w tym w szczególności do objęcia programem pilotowym wybranych urzędów administracji publicznej. W zamian BI URM miało zlecić PTI wykonanie działań ujętych w harmonogramie. PTI opierając się na wyrażonej zgodności intencji podpisało z BCS umowę 3 grudnia 1993 r. w obecności licznych przedstawicieli Rządu,

Ambasady Brytyjskiej, prasy i innych. Wkrótce okazało się, że żadne zlecenie dotyczące realizacji ww. działań do PTI nie wpłynęło.

Gdyby jednak problem ograniczał się jedynie do pieniędzy, to można liczyć, że w warunkach gospodarki rynkowej możliwe byłoby znalezienie innych źródeł finansowania. Wizyty studialne polskich specjalistów w Wielkiej Brytanii uświadomiły, iż system brytyjski jest mocno związany ze specyfiką tamtejszego rynku pracy. Główną cechą różniącą oba rynki jest względna stabilizacja brytyjskiego i dynamiczna zmienność polskiego. Propozycja wdrożenia systemu w Polsce spotkała się z umiarkowanym werbalnym zainteresowaniem pracodawców i niekiedy z ostrym sprzeciwem pracobiorców. Ostatnie lata były okresem budowania struktur firm informatycznych. Największym powodzeniem na rynku pracy cieszył się pracownik o bardzo szerokim spektrum umiejętności i dużej dyspozycyjności. Planowanie rozwoju zawodowego było zaniedbywane w świetle ogromnej zmienności form organizacyjnych i podejmowanych przez firmy tematów. Wydaje się, że w świetle ostatnich tendencji w przemyśle usług informatycznych dopiero nadchodzi czas kiedy pracodawcy i pracobiorcy zaczną potrzebować obiektywnego probierza umiejętności zawodowych. Konsolidacja finansowa firm, rosnąca skala przedsięwzięć informatycznych realizowanych przez polskie przedsiębiorstwa, rosnąca konkurencja na rynku pracy, wzrost płac w branży komputerowej, wszystko to prowadzić może do chaosu, jeśli nie będzie właściwego, jednolitego i obiektywnego systemu oceny kwalifikacji i doświadczeń zawodowych. Wydaje się, że czas na specjalizacje nadchodzi.

Przed opracowaniem i wdrożeniem systemu informatycznych stopni zawodowych w Polsce należy uogólnić doświadczenia europejskie. Na ich podstawie można wyróżnić następujące elementy systemu nadawania informatycznych stopni zawodowych w Polsce:

- modułowy układ stopni zawodowych — można go zapożyczyć od BCS w ramach licencji lub też włączyć się do inicjatywy Wspólnoty Europejskiej (zapewni to m.in. zgodność stopni);
- komisje kwalifikacyjne informatyków i nadające im stopnie zawodowe — powinny mieć one strukturę hierarchiczną i posiadać kompetencje adekwatne do kwalifikacji tworzących je członków;
- system kontroli (adiustacji) zasadności nadawanych stopni — powinien zapewnić utrzymanie odpowiedniego poziomu i wiarygodności stopni;
- lista stanowisk pracy (wraz z zakresami obowiązków) i niezbędnych na nich stopni zawodowych — ma na celu powiązanie stopnia zawodowego z konkretnym stanowiskiem;
- komputerowy bank danych o osobach, które uzyskały informatyczne stopnie zawodowe— to narzędzie wspomagające zarządzanie systemem (np. wyszukujące zdezaktualizowane stopnie) i wspomagające wyszukiwanie osób do realizacji projektów informatycznych lub też do zatrudnienia na poszczególnych stanowiskach;
- system autoryzowanych szkoleń informatycznych— uzupełnia wiedzę pracowników do poziomu wymaganego przy uzyskaniu poszczególnych stopni zawodowych;
- organizacja koordynująca działania systemu;
- porozumienie pracodawców, sankcjonujące stopnie.

W tabeli poniżej został przedstawiony proponowany rozdział prac nad organizacją systemu informatycznych stopni zawodowych.

Proponowany rozdział prac nad organizacją systemu informatycznych stopni zawodowych

Element systemu	Realizator
Koordynacja powstania i funkcjonowania systemu	PTI (licencja BCS lub EISS) — wydzielony organ
Komisje kwalifikacyjne	PTI we współpracy z firmami komputerowymi, uczelniami, innymi towarzystwami informatycznymi
System kontroli	Niezależna komisja powołana przez PTI i pracodawców
Komputerowy bank danych	PTI
System szkoleń	wszyscy zainteresowani, w tym firmy komputerowe i uczelnie po uzyskaniu autoryzacji

* * *

Rozpoczęcie funkcjonowania w Polsce systemu informatycznych stopni zawodowych będzie oznaczało zakończenie okresu twórczego bałaganu w polskiej informatyce i uporządkowanie rynku pracy w tej dziedzinie.

Polskie Towarzystwo Informatyczne jako organizacja zrzeszająca najbardziej doświadczone kadry krajowe powinno podjąć się trudu zorganizowania, uruchomienia i stałej realizacji systemu informatycznych stopni zawodowych.

LITERATURA

[1] A modular curriculum in computer science. Unesco-IFIP, France, 1984

[2] Internetowy serwis informacyjny British Computer Society: http://www.bcs.org.uk

[3] Internetowy serwis informacyjny CEPIS: http://www.bcs.org.uk/cepis.htm

Acer Computer w Polsce

Acer, firma zajmująca siódme miejsce na świecie wśród producentów komputerów osobistych, piąte w produkcji serwerów oraz trzecie w produkcji monitorów, otworzyła w Polsce swoje przedstawicielstwo. Dyrektorem generalnym został mianowany Tomasz Niesłuchowski. Firma Acer, założona w 1976 r., ma 80 oddziałów w 38 krajach i zatrudnia ok. 13 tys. osób. Jej główna siedziba mieści się na Tajwanie. Firma jest producentem komputerów osobistych, płyt głównych, urządzeń peryferyjnych, faksów, pamięci dynamicznych RDAM, specjalizowanych układów scalonych (ASIC), mikroelementów hybrydowych. W 1991 r. jako pierwsza na świecie opracowała system uaktualniania jednostek centralnych komputera (ChipUp), stanowiący obecnie standard przemysłowy. Acer zajmuje drugie miejsce na amerykańskim rynku produktów elektroniki użytkowej. Przychody firmy wyniosły w ub.r. 5,7 mld USD i wzrosły w stosunku do 1994 r. o 77%. (k)

SYSTEMY CZASU RZECZYWISTEGO

ROK III • MAJ • 1996 • 18

Metody projektowania oprogramowania

Projektowanie strukturalne systemów czasu rzeczywistego (2)

Jan Kwiatkowski
Wydziałowy Zakład Informatyki
Politechnika Wrocławska
Stanisław Szejko
Zakład Zastosowań Informatyki
Politechnika Gdańska

Praca przedstawia strukturalną metodę projektowania systemów czasu rzeczywistego. Krótkie omówienie przebiegu i rezultatów analizy poprzedza właściwą część artykułu, poświęconą fazie projektowania. Kolejne kroki, ilustrowane przykładem, obejmują określenie granicy komputeryzacji, projekt środowiska i oprogramowania, projektowanie plików oraz utworzenie specyfikacji konstrukcyjnej.

Projektowanie - projekt szczegółowy

Kolejnym etapem prac jest projekt organizacji oprogramowania każdego z procesorów[1]. Należy pogrupować przyjęte transformacje w *zadania* (*procesy, jednostki przetwarzania*), dobrać dla nich środowisko programowe — system operacyjny, bazę danych, pakiety programów usługowych itp. — i powiązać w jedną strukturę, której działanie zapewni realizację zakładanych funkcji. Jednocześnie, należy określić, jak wyidealizowane rozwiązania modelu logicznego będą realizowane w implementacyjnej rzeczywistości (np. przez zastąpienie próbkowaniem nieprzerwanej transformacji *Nadzór stanu czujnika*).

Wyodrębnienie zadań

Pojęcie *zadania* jest związane z aktywnością procesora - stanowi ono jednostkę, której przydzielany jest czas procesora. Oznacza to, że zadania zdefiniowane dla jednego procesora wykonują się współbieżnie pod nadzorem systemu operacyjnego. Natomiast transformacje wchodzące w skład zadania wykonują się sekwencyjnie, zgodnie z przyjętym dla zadania przepływem sterowania (nie dotyczy to systemów równoległych).

Podczas przypisywania transformacji do zadań trzeba uwzględnić powiązania komunikacyjne — przepływ danych i sterowania pomiędzy zadaniami jest zwykle znacznie wolniejszy i bardziej kosztowny niż w obrębie zadania.

Zgrupowanie wybranych transformacji w jedno zadanie wymaga scalenia związanych z nimi opisów sterowania, niejako odwrócenia porządku czynności opisanych w pierwszej części artykułu (Informatyka nr 4, 1996 r.).

Dobór środowiska programowego

Pierwszoplanowym elementem architektury oprogramowania procesora jest system operacyjny. Może on zadanie uaktywnić, zawiesić, wznowić lub usunąć, może też dostarczać określonych usług, eliminując konieczność samodzielnego wykonania odpowiadającego im oprogramowania. Taką właśnie realizację zegara programowanego, wysyłającego sygnał *upływu czasu* (patrz rys. 7b w pierwszej części artykułu), założono w omawianym przykładzie (rys. 1).

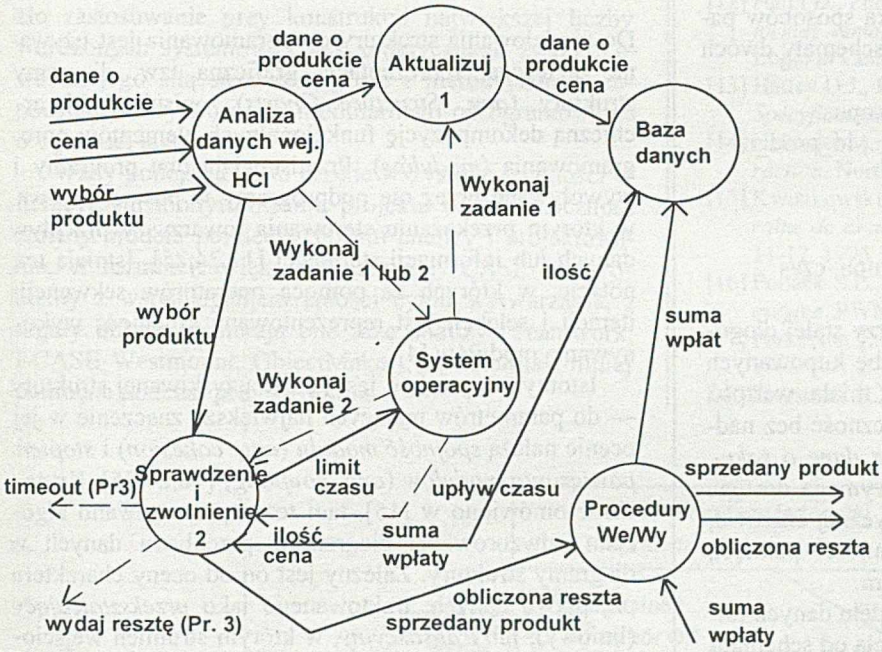
Także inne elementy oprogramowania są dobierane w celu utworzenia środowiska programowego, co podobnie jak w przypadku systemu operacyjnego eliminuje nakład pracy związany z ich przygotowaniem. Do

[1] Zwany *modelem środowiska programowego* [11] lub *modelem zadaniowym* [27, 29].

najczęściej stosowanych należą oprogramowanie standardowego interfejsu, procedury komunikacyjne, specjalizowane pakiety obliczeniowe i systemy baz danych.

Podstawowe kryteria, brane pod uwagę przy doborze środowiska programowego (o ile nie jesteśmy ograniczeni zdefiniowanymi a priori wymaganiami implementacyjnymi) obejmują: funkcjonalność i zgodność z wymaganiami oraz koszt i czas uzyskania. Pod uwagę należy wziąć również korzyści i straty wynikające z samodzielnego rozwijania oprogramowania — przede wszystkim jego niezawodność, elastyczność, dostosowanie do zamierzonej platformy sprzętowej i wymogu przenośności.

Obok doboru składników środowiska programowego i zdefiniowania ich powiązań z wyodrębnionymi zadaniami jest nieodzowne określenie, które z elementów modelu logicznego będą implementowane przez gotowe składniki oprogramowania. Wpływa to istotnie na odwzorowanie modelu procesorowego w model środowiska programowego — rys. 1 ilustruje model przyjęty dla *Procesora*1.



Rys. 1.
Automat sprzedający - dekompozycja oprogramowania i powiązanie jednostek *Procesora* 1

Wprowadzenie cech implementacji

S. Goldsmith [11] wskazuje pięć głównych obszarów, wymagających odwzorowania rozwiązań wyidealizowanych w rozwiązania implementacyjne:

- Zastąpienie zdarzeń uaktywniających transformacje rzeczywistymi mechanizmami dostępnymi w przyjętej architekturze.
- Określenie, które z transformacji reprezentujących w modelu przetwarzanie ciągłe będą w rzeczywistości wykonywane cyklicznie, pozwalając efektywnie wykorzystać czas procesora. Typowym przykładem może tu być zamiana *Monitorowania stanu czujnika* na *Odczyt stanu czujnika*, wykonywany w odpowiedzi na sygnał czasu lub zdarzenie kończące pewien proces.
- Zweryfikowanie, czy system nadąża z przetwarzaniem danych dyskretnych, i zdefiniowanie kolejek tam, gdzie jest to nieodzowne — lub zakup mocniejszego procesora. Jednocześnie, jest konieczne określenie reakcji systemu na asynchroniczne pojawianie się danych. Zwykle z przepływem danych dyskretnych związuje się zdarzenie (np. przerwanie), sygnalizujące ich przybycie.
- Zapewnienie, by zdefiniowane współbieżne zadania wykonywały się w pożądanej kolejności, i uwzględnienie czasu niezbędnego do ich przełączania.
- Zapewnienie bezpieczeństwa danych współdzielonych pomiędzy zadania przez dodanie zachowania zapewniającego właściwy dostęp do magazynów.

Odwzorowanie modelu procesorowego

Odwzorowanie modelu procesorowego w model zadaniowy kończy etap projektowania oprogramowania i zależy od decyzji podjętych przy rozwiązywaniu problemów opisanych w poprzednich rozdziałach. Wynikowy model powinien obrazować ustaloną architekturę oprogramowania, ilustrować powiązanie zadań z przyjętymi elementami środowiska programowego, przedstawiać, jak są realizowane poszczególne funkcje (w obrębie zadań czy przez jednostki środowiska) oraz odwzorowywać wszystkie elementy i zmiany wprowadzone przy nadawaniu konstrukcji cech implementacyjnych. Rys. 1 przedstawia dekompozycję oprogramowania *Procesora*1 sterującego wydawaniem produktu i dokonującego aktualizacji danych o produktach — pojawił się w nim m.in. przepływ powodujący ustawienie limitu czasu oczekiwania, nie ma natomiast magazynów *Suma wpłaty* i *Produkt*, zawartych w jednostce *Baza danych*. Inne decyzje to projekt wykorzystania standardowych procedur interfejsu do odczytu danych wprowadzanych przez operatora i klienta oraz użycie procedur komunikacji z *Procesorem* 3, nie wchodzących w skład systemu operacyjnego.

Projektowanie danych

Zadaniem tego etapu jest odwzorowanie logicznej reprezentacji danych w struktury implementacyjne. Wcześniejsze prace pozwoliły na określenie modelu logicznego danych (w postaci modelu powiązań danych lub od-

powiadającego mu modelu relacyjnego), ich pożądanej struktury (specyfikacja) oraz wiedzy o sposobie wykorzystania i wymaganiach dostępu do nich. Na tej podstawie jest możliwe bezpośrednie zdefiniowanie reprezentacji fizycznej — plików i ich rekordów lub tablic relacyjnej bazy danych, zależnie od przyjętych ustaleń. Proces ten wspiera wiele narzędzi CASE, automatycznie generując pliki definicyjne lub ciągi instrukcji SQL-owych, określających strukturę bazy danych na podstawie zawartości repozytorium projektowego.

Problem w tym, że niejednokrotnie odwzorowanie to jest nieefektywne z punktu widzenia zajętości pamięci masowej lub (co ważne w systemach czasu rzeczywistego) czasu przetwarzania i wielkości wykorzystywanej pamięci operacyjnej. W celu poprawienia tych parametrów, pomiędzy etapy projektowania logicznego i fizycznego danych wprowadza się etap strojenia (*ang. flexing*) modelu logicznego danych. Potrzebę jego wprowadzenia ilustruje następujący przykład: załóżmy, że dla potrzeb rozwojowych lub statystycznych w bazie danych *Procesora*1 chcemy przechowywać informację o kliencie, zakupionych przez niego produktach, dacie i czasie trwania operacji zakupu i, co więcej, że mamy dostęp do potrzebnej informacji. Możliwych jest kilka sposobów pamiętania zachodzącej relacji (rys. 2) — schematy dwóch z nich przedstawiono poniżej:

1. KLIENT (id_klienta, dane_k, data_zakupu, czas_trwania_transakcji, id_prod_1,..., id_prod_n)
   PRODUKT (id_prod, dane_p)
2. KLIENT (id_klienta, dane_k)
   PRODUKT (id_prod, dane_p)
   ZAKUP (id_klienta, id_prod, data_zakupu, czas trwania transakcji)

W pierwszym przypadku musimy (przy stałej długości rekordów) założyć maksymalną liczbę kupowanych produktów, a spora część pól będzie miała wartość NULL, w drugim — mamy dużą elastyczność bez nadmiarowości. A jednak, transakcja *Zapisz dane o zakupach klienta* w pierwszym przypadku wymaga dostępu do jednej tablicy mniej i w sytuacji krytycznej czasowo, gdy nie występują ograniczenia pojemności pamięci, mogłaby być rozwiązaniem preferowanym.

Trzy popularne techniki strojenia modelu danych to:

- Łączenie tablic, jak w sytuacji przejścia od schematu 2 do schematu 1 przykładu.
- Dzielenie tablic, w przypadku gdy część atrybutów tablicy jest rzadko wykorzystywana. Podział tablicy na dwie spowoduje, że krytyczne czasowo transakcje przetwarzać będą rekordy o mniejszej liczbie pól, stawiając mniejsze wymagania pamięci operacyjnej i unikając częstego jej wymiatania.
- Dodawanie atrybutów, które w zamian za wprowadzaną nadmiarowość powodują zwiększenie efektywności, jak np. liczniki wystąpień zamiast zliczania rekordów.

W pracy [24] można znaleźć dwie techniki, pozwalające dokonać oceny przyjętego schematu logicznego danych: diagramy, wiążące transakcje z atrybutami przechowywanych obiektów danych i relacji (*ang. Transaction - Attribute Grids*), oraz reprezentacje logicznego dostępu (*ang. Logical Access Maps*), ilustrujące powiązanie kolejnych dostępów do danych z przechowywanymi danymi.

Projektowanie implementacji - końcowy etap projektowania danych, stanowi odwzorowanie przyjętego schematu w założone środowisko programowe, bazę danych, czy też system plików oferowany przez wybrane narzędzie. Nie różni się on istotnie od przebiegu prac wykonywanych przy konstrukcji plików [24] lub systemów baz danych [6, 26].

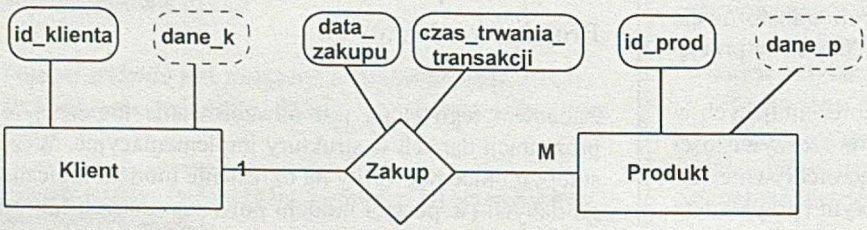
Projektowanie struktury oprogramowania

Do modelowania struktury oprogramowania jest używana zazwyczaj reprezentacja graficzna tzw. diagramy struktury (*ang. Structure Charts*), opisujące hierarchiczną dekompozycję funkcjonalnych elementów oprogramowania (*modułów*). Przedstawiają one programy i wywoływane przez nie podprogramy w postaci drzewa, w którym przekazaniu sterowania towarzyszy przepływ danych lub informacji sterującej [11,24,25]. Istnieją też notacje, w których za pomocą operatorów sekwencji, iteracji i selekcji jest reprezentowana kolejność wykonywania modułów [4].

Istotnym kryterium jest jakość uzyskiwanej struktury — do parametrów mających największe znaczenie w jej ocenie należą *spójność modułu* (*ang. cohesion*) i *stopień powiązania modułów* (*ang. coupling*) [18,24,25]. Kryteria te omówiono w [15], tam też zaprezentowano algorytm odwzorowania diagramów przepływu danych w diagramy struktury. Zależny jest on od oceny charakteru przepływu danych, traktowanego jako *przekształcający* (liniowy), lub *transakcyjny*, w którym strumień wejściowy powoduje przepływy wzdłuż wielu ścieżek aktywności. Pierwsze, zgrubne odwzorowanie diagramu pozwala na określenie modułów leżących najwyżej w hierarchii, kolejne odwzorowania pozwalają uściślić strukturę programu. Należy podkreślić rolę heurystyki w uzyskaniu dobrej jakościowo struktury oprogramowania. Szczegółowa konstrukcja diagramu obejmuje też specyfikację interfejsów modułów, deklaracje struktur danych i powiązań międzymodułowych.

Z uwagi na szczegółowe omówienie procesu przekształcenia w [15] w niniejszym artykule prezentujemy jedynie wynikowy diagram, otrzymany dla zadania *Sprawdzenie i zwolnienie* (rys. 3) *Procesora*1.

Rys. 2. Przykład relacji 1:M, wiążącej Klienta i dokonywane przezeń zakupy produktów



Dokumentacja

Rezultaty przedstawionego zbioru prac stanowią specyfikację konstrukcyjną, stanowiącą podstawę kolejnego etapu prac - implementacji. Specyfikacja ta przyjmuje postać dokumentacji projektowej, której najważniejszy składnik stanowi Dokument Projektu Konstrukcyjnego (*ang. Architectural Design Document*) [9, 18]. Zawiera on przegląd struktur i przepływów danych, określenie struktury oprogramowania i opis powiązań. Dla każdego modułu określa jego usytuowanie w strukturze, definiuje interfejs i używane dane, a także podaje język implementacji. Opisywane są także pliki: struktura danych i rekordów, zasady dostępu, integralność. Całość uzupełniają zasady przeprowadzenia testów i informacje pomocnicze.

Inne dokumenty powstające w fazie projektowania to m.in. plany organizacji etapu implementacji i plany weryfikacji oraz zapewnienia jakości oprogramowania.

* * *

Można szacować, że projektowanie strukturalne znalazło zastosowanie przy konstrukcji największej liczby wdrożonych systemów. Obok praktycznego zweryfikowania, jego siłą są zrozumiałość i metodyczność, zapewniające wytworzenie modularnego oprogramowania o wysokiej jakości.

Wady podejścia — to przede wszystkim trudności w iteracyjnym modyfikowaniu projektu oraz konieczność zmiany modelu pomiędzy fazami analizy i projektowania. W rezultacie większość narzędzi CASE ma problemy z wspomaganiem całości cyklu wytwarzania, z reguły dobrze wspierają one fazę analizy (Teamwork, I-CASE Westmount, ObjectMaker), są natomiast mniej pomocne podczas projektowania.

LITERATURA

[1] Avison D.E. and Fitzgerald G.: *Information Systems Development: Methodologies, Techniques and Tools.* Blackwell Scientific Publications, 1988

[2] Coad P. and Yourdon E.: Object-Oriented Analysis/Design Yourdon Press, 1991

[3] Cohen B., Harwood W.T., Jackson M.I.: *The Specification of Complex Systems.* Addison-Wesley, 1986

[4] Curtis G.: *Business Information Systems. Analysis, Design and Practice.* Addison-Wesley Publ. Comp., 1989

[5] Cysewski G., Łyskawa H., Misiak D., Nowak P., Piechówka M., Szejko S.: *Design prototyping: can it be a remedy?* Materiały konf. EUROMICRO'95, Como, Italy. IEEE Computer Society Press, 1995

[6] Date C. J.: *An Introduction to Database Systems*, Addison- Wesley Publ Co, 1986

[7] DeMarco T.: *Structured Analysis: Systems Specifications. Prentice-Hall*, 1979

[8] Dunn R.: *Software Quality Assurance*, Prentice-Hall, 1990

[9] *ESA Software Engineering Standards*, ESA PSS-0-0 Issue 2, 1991

[10] Farmer R.: *Vending Machine: A Case Study in Teamwork.* De Montfort Unversity, Leicester 1995

[11] Goldsmith S.: *A Practical Guide to Real-Systems Development.* Prentice-Hall Int., 1993

[12] Harel D., Pnueli A., Schmidt J.P. and Sherman R.: *On the formal semantics of statecharts.* Proc. Symposium on Logic in Computer Science, New York, June 1987

[13] Hatley D.J., Pirbhai I.A.: *Strategies for Real-Time System Specyfication.* Dorset House, 1987

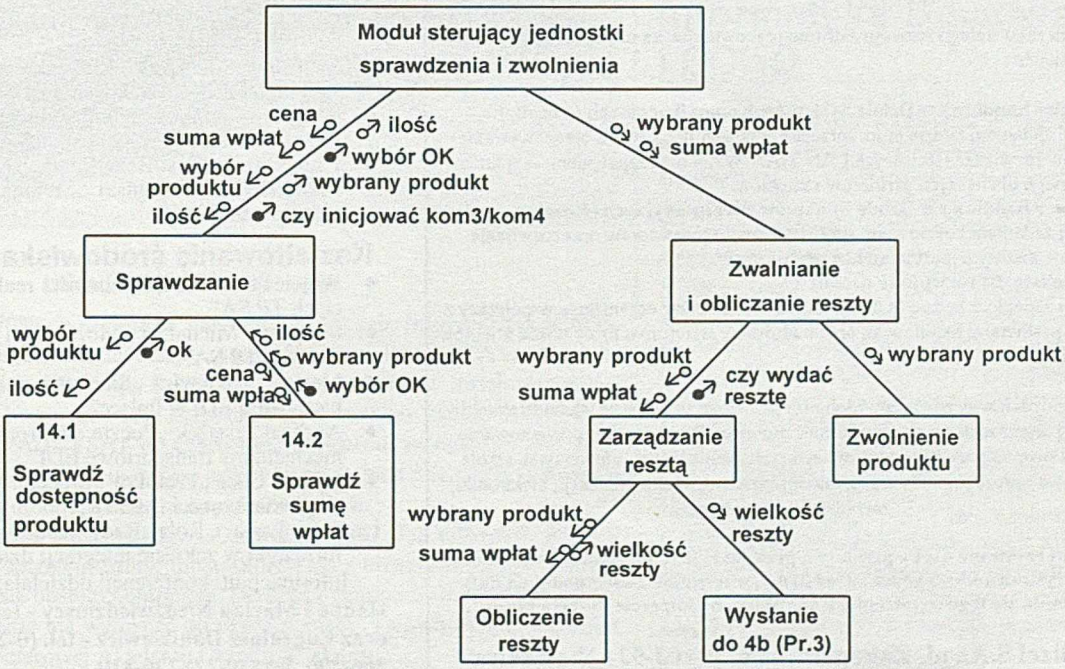
[14] Helander M. (ed.): *Handbook of Human - Computer Interaction.* North-Holland, Elsvier Sci. Publishers, 1991

[15] Kwiatkowski J., Misiak D., Szejko S.: *Podejście strukturalne do analizy i projektowania systemów.* Informatyka, 11-12, 1995

[16] Pollack S.L., Hicks H.T.(Jr.), Harrison W.J.: *Tablice decyzyjne.* PWN, 1975

[17] Parkinson J.: *Making CASE Work.* NCC Blackwell Ltd., 1991

[18] Pressman R.S.: *Software Engineering. A Practitioner's*



Rys. 3. *Automat sprzedający* - diagram struktury zadania *Sprawdzenia i zwolnienia*

Approach. McGraw-Hill, Inc., 1992
[19] Reisig W.: *Sieci Petriego*. WNT, Warszawa, 1988
[20] Rumbaugh J., Blaha M., Premerlani W., Eddy S., Lorensen W.: *Object-oriented Modeling and Design*. Prentice--Hall 1991
[21] Sacha K.: *Problemy jakości w projektowaniu oprogramowania systemów wbudowanych*. Informatyka nr 9, 1995
[22] Sannella D.: *A survey of formal software development methods*. Intagrated Systems Factory Study rep., 1989
[23] Skidmore S., Farmer R. and Mills G.: *SSADM Version 4. Models and Methods*. NCC Blackwell Ltd., 1992
[24] Skidmore S. and Wroe B.: *Introducing Systems Analysis/Design*. NCC Blackwell 1988-1990
[25] Sommerville I.: *Software Engineering*. Addison-Wesley, 1992
[26] Ullman J.D.: *Systemy baz danych*. WNT, 1988
[27] Ward P. T. and Mellor S. J.: *Structured Development for Real-Time Systems*. Yourdon Press, 1985
[28] Wing J.M.: *A specifier's introduction to formal methods*. COMPUTER 1990, pp.8 -24
[29] Yourdon E.: *Modern Structured Analysis*. Prentice-Hall, 1989.

Kontakt z Autorami:
Jan Kwiatkowski
E-mail: **kwiatkowski@ci-1.ci.pwr.wroc.pl**
Stanisław Szejko
E-mail: **stasz@sunrise.pg.gda.pl**

IV Krajowa Konferencja EDI
- planowany program

Dobiegają końca przygotowania związane z IV Krajową Konferencją EDI, która odbędzie się w Arturówku k. Łodzi w dniach 3-5 czerwca 1996 r. Oto planowany program tematyczny konferencji i w podziale na bloki problemowe.

Zastosowanie EDI w bankowości

- Andrzej Poręba „Standaryzacja electronic banking i EDI".
- Peter Green „Electronic Trade payments and Global Cash Management".
- Krzysztof Michalski „Elektroniczny system rozliczeniowy w Niemczech".
- Tomas Martoch „System rozliczeń elektronicznych firmy EDITEL".
- Jacek Piotrowski „Elektroniczna bankowość - przegląd i ocena sytuacji w Polsce".
- Małgorzata Ziemecka „Zastosowanie EDI w bankowości".

Zastosowanie EDI w obrocie handlowym

- Przemysław Polak, Marek Tokarski „Rola EDI w integracji łańcucha logistycznego".
- Stefan Abt „EDI - uwarunkowaniem budowy Logistycznych Centrów Dystrybucji".
- Elżbieta Hałas, Romuald Swarcewicz „EANCOM w efektywnej strategii obsługi klienta - ECR.
- Anna Kosmacz-Chodorowska „Znakowanie jednostek wysyłkowych dla potrzeb EDI.
- Marek Miłosz „Wpływ EDI na parametry dynamiczne sieci logistycznych".
- Michał Goliński „Rynki elektroniczne - podstawowe aspekty teoretyczne. Implikacje gospodarcze".

Zastosowanie EDI w transporcie

- Dariusz Dziuba „Technologia EDI dla potrzeb (globalnych) systemów rezerwacji lotniczej".
- Izabella Mitraszewska „Problemy budowy zinformatyzowanych SIK dla przedsiębiorstw transportowo-spedycyjnych".

Problemy budowy systemów EDI

- Małgorzata Pańkowska, Elżbieta Kocioł „Zarządzanie strategiczne i przebudowa procesów organizacyjnych jako przesłanki wprowadzenia EDI w organizacji".
- Andrzej Małachowski, Elżbieta Niedzielska „Wirtualna organizacja gospodarcza i jej infrastruktura komunikacyjna".
- Stanisław Strzelczak „EDI jako platforma międzyorganizacyjnych SIZ. Zastosowanie EDI do integracji".
- Stanisław Stańko „Symulacyjne określanie efektywności zastosowania EDI w jednostkach gospodarczych".
- Paul Nowak „Cost/Benefit analysis in EDI".
- Marian Niedźwiedziński „Metoda JAD w projektowaniu zastosowań EDI".

Kształtowanie środowiska dla potrzeb EDI

- Wojciech Molisz „Możliwości realizacji usług EDI w sieciach TP SA".
- Sławomir Michalski „Możliwości realizacji usług EDI w sieciach TP SA".
- Marian Suskiewicz „Sieci transmisji danych TP SA jako baza usług EDI w Polsce".
- Andrzej Pilaszek „Poczta elektroniczna oraz Internet jako mechanizmy transportowe EDI".
- Jolanta Chęć „Protokół i procedury EDI w systemach obsługi wiadomości (MHS)".

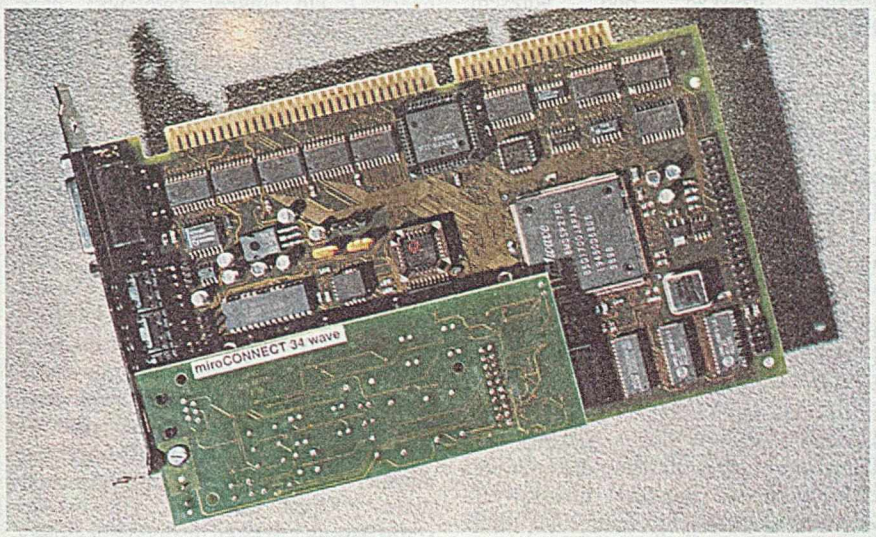
Lech Szukszta „Rola Rady Koordynacyjnej ds. Teleinformatyki w zakresie integracji działań".

Informacji nt. konferencji udzielają:
Hanna i Marian Niedźwiedzińscy - tel., fax (0-42) 70-72-86 oraz Eugeniusz Danikiewicz - tel. (0-22) 266-065 lub 266-296, fax: (0-22) 276-810

informacje

OPTIMUS w sieci

Optimus TeleConnect to nowa seria komputerów multimedialno-telekomunikacyjnych nowąsądeckiej spółki, których premiera odbyła się na INFOSYSTEMIE. Wyposażone są one fabrycznie w wielofunkcyjną kartę rozszerzającą MiroConnect 34 Wave, która ma oprogramowanie na emulator terminala, pakiet dostępowy do Internetu Chameleon, program obsługujący faks, telefon i automatyczną sekretarkę. W przyszłości użytkownik karty będzie mógł ściągnąć z Internetu nową wersję jej oprogramowania systemowego. Wkrótce ma być dostępny moduł rozszerzający do karty MiroConnect 34 Wave umożliwiający odbieranie na komputerze programów radiowych. Do komputera Optimus SA TeleConnect oprócz standardowego oprogramowani operacyjnego Windows 95 dodawany jest WORLDGROUP MENAGER - specjalny pakiet do pracy w komercyjnej sieci transmisji danych OptimusNET, której inauguracja odbyła się na INFOSYSTEMIE. Na OptimusNET składa się osiem serwisów OnLine opartych na działających firmowych BBS -ach. Wkrótce za jej pośrednictwem będzie można podłączyć się do sieci Internet. Jak poinformowano dziennikarzy na konferencji prasowej Optimus podpisał z TP S.A. umowę na dzierżawę łączy wykorzystywanych przez OptimusNet.



micro CONNECT 34 wave

Należąca w ponad 70% do Optimusa Powszechna Agencja Informacyjna (PAGI) zaprezentowała na Infosystemie satelitarny system łączności PAGI-VSAT, który pozwala na komunikację i transmisję danych między terminalami instalowanymi u klienta. Umożliwia to tworzenie sieci korporacyjnej o zasięgu ogólnoeuropejskim. Optimus wykorzystuje PAGI-VSAT do uruchamiania systemu zarządzania SAP R/3. Zaletą PAGI-VSAT jest m. in. to, iż można instalować ją w tych miejscach, w których położenie kabla jest utrudnione, a czasem wręcz niemożliwe.

Na uwagę zasługuje też inna inicjatywa Optimusa: każdy, kto odwiedzi jeden z ponad 100 sklepów ze sprzętem firmy, może skorzystać z dwugodzin-nego, bezpłatnego szkolenia w zakresie obsługi komputera. Do końca tego roku sieć sklepów oferujących bezpłatne szkolenie ma wzrosnąć do około ty-siąca. Ta niebanalna inicjatywa otrzymała nazwę Komputerowej Akademii Optimusa. **(kar)**

Nagrody KBN na POLMAN '96

Tradycyjnie już na odbywającej się podczas INFOSYSTEMU wystawy, konferencji i prezentacji POLMAN '96 - jednej z najważniejszych imprez organizowanych pod patronatem KBN - wręczono nagrody przewodniczącego Komitetu Badań Naukowych prof. Aleksandra Łuczaka. W kategorii na najlepszy produkt sieciowy prezentowany przez jednostki naukowe I nagrodę otrzymała grupa pracowników Instytutu Informatyki Politechniki Poznańskiej za "System Informatyczny Dziekanatu Wyższej Uczelni - DZIEKANAT 96". II nagrodę przyznano za "Komputerowy System Pomiarowy SP-2000", który powstał w Instytucie Elektroniki i Telekomunikacji Politechniki Poznańskiej. III nagroda przypadła w udziale Politechnice Gdańkiej za "Komputerowy Rozkład Jazdy Pociągów i Samolotów". W konkursie na najciekawszą ekspozycję firmową w ramach wystawy POLMAN '96 nagrody otrzymały następujące firmy: ATM Sp. z o.o., 3COM GmbH oddział w Warszawie, Przedsiębiorstwo Produkcyjno-Wdrożeniowe TELEFON 2000 Sp. z o.o. **(k)**

Informatyka Domowa i SOHO

Od 30 maja do 2 czerwca w salach Muzeum Techniki w Warszawie odbędzie się pierwsza tego typu impreza w Polsce: wystawa połączona z festynem promocyjnym i kiermaszem: KOMPUTER DLA CIEBIE - INFORMATYKA DOMOWA i SOHO (Small Office, Home Office). Organizatorzy imprezy - Centrum Promocji Informatyki i Muzeum Techniki - chcą m.in. pokazać warszawiakom i przyjezdnym gościom, że dziś ceny komputerów markowych i legalnych programów konkurują z cenami na giełdach i u pokątnych sprzedawców, którzy nie zapewniają ani jakości, ani serwisu. Podczas wystawy będzie funkcjonować Cafe Internet z kilkunastoma stanowiskami zapewniającymi dostęp do Internetu, sala kinowa z projekcjami filmów, prezentacjami, seminariami, konkursami itp. **(k)**

Nowym produktem Informixa jest OnLine Workgroup Server, służący do niezbyt skomplikowanych aplikacji wymagających jednak efektywnego przetwarzania. Produkt ten został oparty głównie na technologii Informix OnLine DSA. Workgroup Server pozwala tworzyć prostsze aplikacje mniejszym kosztem przy zachowaniu większości zalet technologii Informixa, takich jak skalowalność, szybkość czy wielowątkowość przetwarzania.

*

Microsoft stworzył API Direct3D, interfejs programistyczny nowej generacji, przeznaczony do tworzenia interaktywnej grafiki trójwymiarowej, działającej w czasie rzeczywistym na typowych komputerach PC i w Internecie. Ponad 80 wiodących producentów oprogramowania i sprzętu zgłosiło już zamiar dostarczania wyrobów korzystających z Direct3D. Nowy produkt Microsoft jest uzupełnieniem serii popularnych multimedialnych technologii interaktywnych z serii DirectX.

*

Brytyjska firma Plasmon zaprojektowała CD-ROM Jukebox CD150J-4, który umożliwia szybszy i łatwiejszy dostęp OnLine do 150 dysków kompaktowych (trzy magazynki). Jukebox jest wyposażony w cztery poczwórnej prędkości napędy oraz interfejs SCSI, który zapewnia łatwiejszą integrację oraz wysokiej jakości pracę urządzenia. Krótki czas wymiany dysków pozwala na dostęp do danych w ciągu kilkunastu sekund. CD150J-4 zapewnia ochronę danych przenosząc dyski zawsze w "tackach". Inną nowością tej firmy jest napęd optyczny PD 2000 opary na technologii wielokrotnego zapisu zmiany fazy na dyskach optycznych. Oferuje on dodatkową możliwość zapisu i odczytu CD-ROM z poczwórną prędkością. Technologia zmiany fazy pozwala na bezpośredni powtórny zapis w ciągu jednego nagrania, co umożliwia dokonywanie backupów twardego dysku i przechowywania dużych plików. Pojemność jednego dysku wynosi 650 MB.

*

Waldemar Sielski - dyrektor generalny Microsoft Sp. z o.o. i Marek Racieski - dyrektor generalny Digital Equipment Polska Sp. z o.o. podpisali wspólny biznes plan dotyczący współpracy między obydwoma firmami. Uzgodniona została lista projektów, w których będzie oferowana technologia Microsoft BackOffice i serwery z procesorem Alpha. Najważniejszym celem podpisanego planu jest wprowadzenie na polski rynek produktów Microsoft Exchange Server oraz Microsoft Information Server. Warto przypomnieć, że w 1995 r. centrale firm Microsoft i Digital zawarły strategiczne porozumienie w zakresie rozwijania sieci rozległych i oprogramowania działającego w środowisku rozproszonym.

*

Szwedzka firma Business Security wprowadza na nasz rynek, za pośrednictwem Polish Security Systems Sp. z o.o., urządzenia szyfrujące utajniające transmisję informacji przekazywanej bądź przez publiczną sieć telefoniczną (telefon, telefax), bądź przez sieć komputerową. Wszystkie urządzenia produkowane przez Business Security są oparte na tym samym - będącym własnością producenta - algorytmie SBLH oraz systemie dystrybucji kluczy na programowalnych kartach elektronicznych. Linia urządzeń szyfrujących transmisję danych zawiera urządzenia dostosowane do wszystkich standardów sieci zgodnych z zaleceniami ITU-T.

*

Autodesk oferuje pakiet narzędzi przeznaczony do wspomagania wydajności i integracji danych - AutoCAD DataExtension (ADE) Release 2. Program pomaga projektantom, kreślarzom i kierownikom projektów w organizacji projektu i integruje różne obszary rysowania oraz powiązane z nimi bazy danych w jednorodne środowisko AutoCD, które może być wykorzystywane przez wszystkich członków zespołu projektowego. Przeznaczony do pracy grupowej program pozwala na zwiększenie wydajności i elastyczności zespołu. Więcej informacji można uzyskać pod adresem: **http://www.autodesk.com**

*

Firma GAMBIT z Krakowa wprowadziła do swojej oferty biblioteki numeryczne do C/C++ i do Fortranu. Biblioteki zawierają ponad 1000 procedur numerycznych. Dostępne są do kompilatorów na platformę PC (DOS, WIN) oraz na platformy UNIX-owe.

*

Firmy Hewlett-Packard oraz Aplied Communications Inc. i Vento Software zawarły porozumienie dotyczące wspólnego oferowania klientom z sektora bankowego systemów do zarządzania bankomatami i innymi formami dokonywania elektronicznych operacji finansowych. Zdaniem ekspertów do roku 2000 liczba operacji elektronicznych tylko w Stanach Zjednoczonych osiągnie poziom 60 mld USD rocznie, ponad dwukrotnie więcej niż w 1994 r. Oprogramowanie firmy Vento zostało opracowane dla serwerów i sieci roboczych HP 9000, wyposażonych w procesory PA-RISC.

*

Computer Associates International wypuścił kolejną wersję narzędzia do tworzenia aplikacji baz danych dla Windows 3.1 - -CA-Visual Objects 1.0c. Nowy produkt bazujący na standardzie Xbase pozwala użytkownikom Clippera oraz wersji dBase i FoxPro dla DOS w prosty sposób przenieść aplikacje do środowiska graficznego. Update z poprzednich wersji do 1.0c jest dostępny przez Internet pod adresem: **ftp://mf.cai.com/caproducts/vo**

*

Firma RAD Data Communications wprowadza na rynek dwa nowe moduły kompresji głosu do multipleksera T1/E1 Megaplex-2000. Moduły VC-PBX i VC- -3 obsługują odpowiednio sześć i trzy kanały, działają przy niskiej przepustowości, pozwalają na przesyłanie głosu i transmisje faksowe. Moduły te obniżają koszty połączeń WAN i dają dodatkowe pasmo do przesyłania danych i ruchu LAN.

*

LEXMARK (firma powstała w 1991 r. w wyniku przejęcia od IBM działu produktów informacyjnych, głównie drukarek) zaprezentował nowe modele drukarek igłowych z serii Forms Printers 2300 Plus, przeznaczone dla dużych instytucji. Drukarka 2300 potrafi drukować przez sześć warstw, firma gwarantuje też aż 10 000 MTBF (*Mean Time Between Failure* - średni czas międzyawaryjny). Drukarki i materiały eksploatacyjne firmy Lexmark będą wykorzystywane do drukowania wyników konkurencji olimpijskich w Atlancie. Dystrybutorem drukarek Lexmarka w Polsce jest firma Printmark Polska z Wrocławia.

*

Compaq ogłosił, iż w 1995 r. uzyskał rekordowe przychody w wysokości 14,8 mld USD, co oznacza 36-procentowy zysk w porównaniu z 1994 r. Zysk netto wyniósł 1 mld USD. Wyniki te stawiają Compaq Computer Corporation na piątym miejscu wśród największych firm komputerowych świata.

*

Firma Hewlett-Packard poinformowała, iż najnowsza wersja jej systemu operacyjnego HP-UX uzyskała prawo do posługiwania się znakiem UNIX 95 (zwanego dawniej SPEC1170), licencjonowanym przez X/Open Company. HP roześle certyfikowaną wersję Unixa do klientów.

*

Novell wprowadza na rynek nowe zaawansowane narzędzia, które umożliwiają przedsiębiorstwom łatwe i szybkie przechodzenie z systemów Microsoft NT Server, Microsoft LAN Manager i IBM LAN Server na system NetWare 4.1 oraz usługi NDS (*NetWare Directory Services*). Narzędzia te są dostępne w specjalnej wersji Migration Edition zestawu Novell Consulting ToolKit, która jest rozpowszechniana wśród partnerów biorących udział w forum Enterprise Consulting Partners.

Opracowała Krystyna Karwicka

Książki nadesłane

Wydawnictwa Naukowo - Techniczne

Ronald R. Yager, Dimitar P. Filev: Podstawy modelowania i sterowania rozmytego. Z angielskiego przełożyli: Stanisław Jankowski, Cezary Mazur i Ryszard Wańczuk. WNT Warszawa 1996 r. wyd. 1, s. 388, cena 22,0 zł ISBN 83-204-1909-3

Sterowanie rozmyte jest swego rodzaju fenomenem we współczesnej technice i nauce. Przez wiele lat dziedzina ta była uprawiana przez grupę zapaleńców, nie mogła się przebić przez mur niechęci, a nawet wrogości. Koronnym argumentem przeciw niej był zawsze brak zastosowań praktycznych. Europa Zachodnia i Stany Zjednoczone, w odróżnieniu od Japonii, nie dostrzegały możliwości sterowania rozmytego. Sytuacja uległa zmianie dopiero po uruchomieniu w Sendai pociągu metra sterowanego regulatorem rozmytym. Wielkie firmy światowe przystąpiły do prac nad stosowaniem sterowania rozmytego i do wyścigu z Japonią.

Książka przedstawia istotę sterowania rozmytego oraz środki i narzędzia formalne - bez zbędnych szczegółów technicznych, czytelnych jedynie dla wąskiego grona specjalistów - automatyków. Omówiono w niej teorię zbiorów rozmytych, operacje agregacji na zbiorach rozmytych, wnioskowanie przybliżone i podstawy formalne sterowania rozmytego. Przedstawiono modele systemów rozmytych i ich budowę, a więc zagadnienia pozyskiwania; reprezentacji i przetwarzania wiedzy od ekspertów. Omówiono też analizę teoretyczną regulatorów rozmytych i zagadnienia tzw. defuzyfikacji.

Książka jest przeznaczona dla studentów i pracowników naukowych takich kierunków, jak informatyka, matematyka i elektronika. Napisana bardzo przystępnym językiem może stanowić interesującą lekturę także dla laików.

Jan J. Mulawka: Systemy ekspertowe. WNT, Warszawa 1996 r., wyd.1, s. 236, cena 11,0 zł, ISBN 83-204-1890-9

Sztuczna inteligencja to dynamicznie rozwijająca się dziedzina, mająca istotny wpływ na rozwój techniki, zarządzania, gospodarki, medycyny, wojskowości itp. Systemy ekspertowe są programami komputerowymi przeznaczonymi do rozwiązywania specjalistycznych problemów wymagających profesjonalnej ekspertyzy. Stosowanie systemów ekspertowych umożliwia polepszenie jakości produkowanych wyrobów, osiągnięcie znacznych oszczędności, zwiększenie wydajności pracy a dzięki temu zmniejszenie zatrudnienia.

W książce, jednej z nielicznych w Polsce monografii poświęconych sztucznej inteligencji, przedstawiono podstawowe właściwości i metody konstruowania systemów ekspertowych. Autor wprowadza Czytelnika w tajniki systemów ekspertowych i związanych z nimi problemów. Wiele uwagi poświęca metodom reprezentacji i przetwarzania wiedzy. Przedstawia podstawowe metody wnioskowania i strategie przeszukiwania przestrzeni rozwiązań. Podaje podstawowe architektury systemów ekspertowych, a także metody pozyskiwania wiedzy. Omawia różne narzędzia do tworzenia tych systemów i szczegółowo opisuje język Clips. Dużo miejsca poświęca też omówieniu nowoczesnych systemów do przetwarzania wiedzy, tzw. systemów hybrydowych. Szczególną uwagę zwraca na zagadnienia związane z uczeniem się maszyn i algorytmami genetycznymi.

Książka jest przeznaczona dla studentów kierunków informatycznych i wszystkich zainteresowanych systemami ekspertowymi.

W. Richard Stevens: Programowanie zastosowań sieciowych w systemie UNIX. Z ang. przełożyli: Krzysztof Czaja i Jowita Koncewicz-Krzemień. WNT, Warszawa 1995 r., wyd.1, s. 822, cena 35,0 zł ISBN 83-204-1872-0

Jest to podstawowa książka na temat tworzenia programów wykonywanych w sieciach komputerowych, będąca zarówno bogatym źródłem wiedzy o współczesnym oprogramowaniu sieciowym, jak i praktycznym podręcznikiem programowania.

Książka zawiera opisy używanych obecnie popularnych protokołów sieciowych - w tym: TCP/IP Internetu, protokołów sieci Xerox NS, SNA, NetBIOS, OSI oraz UUCP. Omówiono w niej także najważniejsze narzędzia programisty: interfejsy warstwy transportu, dostępne w Unixie 4.3BSD (gniazda Berkeley) oraz w Systemie V (TLI). Zamieszczono liczne przykłady programów, opatrzone obszernymi komentarzami. Pomagają one zrozumieć, jak działa podstawowe, usługowe oprogramowanie sieci komputerowych, jak można z niego korzystać oraz jak powinno się pisać własne programy użytkowe. Przykłady dotyczą typowych zastosowań sieciowych, takich jak: przesyłanie plików w sieci, buforowanie drukarek, zdalne wykonywanie poleceń, zdalne zgłaszanie się drugiego systemu, zdalne wywoływanie procedur i in.

Książka jest przeznaczona dla studentów kierunków informatycznych, a także dla tych wszystkich osób, które zawodowo zajmują się programowaniem i zastosowaniem sieci komputerowych. **(t)**

Zaprosili nas...

Polskie Towarzystwo Użytkowników NetWare na walne zebranie Towarzystwa.

IDC Poland S.A., organizator II Międzynarodowej Konferencji i Wystawy Technik Telekomunikacyjnych i Sieci Komputerowych ComNet Warszawa'96 na konferencję prasową omawiającą tematy tegorocznej imprezy.

Microsoft Sp. z o.o. na spotkanie na temat „Strategia Microsoft w Internecie" połączone z oficjalnym ogłoszeniem produktów dla Internetu: Microsoft Internet Explorer 2.0 PL oraz Microsoft® Internet Information Server.

TELMONT Sp. z o.o. i OPTIMUS S.A. na uroczystość podpisania umowy o współpracy.

SOLIDEX na seminarium „współczesne systemy sieciowe" inaugurujące otwarcie filii firmy w Trójmieście.

IBM Polska na konferencję prasową, w trakcie której przedstawiono nową strategię firmy w zakresie oprogramowania, obejmującą m.in. program wspomagania sprzedaży BESTeam.

Bentley Systems Polska na spotkanie omawiające strategię firmy w bieżącym roku.

Microsoft na uroczystą premierę Office Professional po polsku dla Windows'95 w tym Access po polsku dla Windows'95.

Digital Equipment Polska Sp. z o.o. na prezentację najnowszego notebooka Digital HiNote Ultra II oraz nową rodzinę komputerów AlphaStation z grafiką PowerStorm.

Centrum Prasowe PAI S.A. w imieniu Centrum Promocji Informatyki oraz Polskiej Izby Informatyki i Telekomunikacji na konferencję omawiającą obecność polskich firm na targach CeBIT'96.

Daniel Le Cognic - dyrektor generalny Bull Polska Sp. z o.o. na konferencję na temat wyników finansowych oraz wzajemnego zakupu udziałów przez firmy Bull, NEC i Packard Bell.

Informix Software na przedświąteczne spotkanie informujące o rezultatach działalności Informixa i prezentację nowych produktów, które będą prezentowane na targach Infosystem'96.

Hewlett-Packard Polska na konferencję poświęconą nowym produktom HP.

Alcatel Business Systems Poland na VII Międzynarodowe Targi Łączności INTERTELECOM'96.

do Poznania na INFOSYSTEM'96

Hewlett-Packard Polska - na swoje stoiska,

Microsoft - na prezentacje i konferencje,

Sun Microsystems na seminarium oraz światową i polską premierę systemu ULTRA ENTERPRISE. **(t)**

IBM wspomaga partnerów

IBM zaprezentował polskim partnerom handlowym nowy program wspomagania sprzedaży BESTeam, bardzo już popularny w Ameryce i wysoko oceniany w Europie. Warto przypomnieć, z czego może nie na co dzień zdajemy sobie sprawę, że IBM jest jednym z największych w świecie producentów oprogramowania. W 1995 r. z jego sprzedaży, bez produktów Lotusa, osiągnął 12 mld USD. By zwiększyć sprzedaż IBM stworzył program wspomagania partnerów handlowych pod nazwą BESTeam (*Business Enterprise Solution Team*). W ramach tego programu IBM oferuje wszechstronną pomoc techniczną, udostępnia materiały techniczne, szkolenia, certyfikację oraz wspólne działania marketingowe. Oprogramowanie IBM zostało podzielone na siedem obszarów, uczestnicy programu BESTeam mogą wybrać specjalizację w jednym obszarze oprogramowania lub produktu. Partnerzy IBM - uczestnicy programu - muszą mieć w firmie co najmniej jednego inżyniera posiadającego certyfikat IBM, który wybrał specjalizację w jednym z siedmiu obszarów oprogramowania. BESTeam pokazuje, że IBM podjął poważną batalię mającą na celu zwiększenie konkurencyjności swojego oprogramowania.

Nowe serwery Suna

Równocześnie w USA i w Polsce (16 kwietnia na INFOSYSTEMIE) Sun Microsystems zaprezentował nową rodzinę serwerów Ultra Enterprise z procesorami UltraSPARC. Nowe maszyny to: Entreprise 1 - mały "biurkowy" serwer wykorzystujący w pełni nową architekturę procesora UltraSPARC; Enterprise 2 - dwuprocesorowy serwer średniej mocy; Enterprise 150 - jednoprocesorowy serwer podwyższonej niezawodności; Enterprise 3000 - wieloprocesorowy serwer (do 6 procesorów) dużej mocy, który może obsłużyć kilkuset użytkowników; Enterprise 4000 - (do 14 procesorów) w oryginalnej przemysłowej obudowie, silnie skalowany, może być stosowany również do zaawansowanych obliczeń numerycznych; Enterprise 5000 - (do 14 procesorów) przeznaczony do współpracy z dużymi pamięciami masowymi; Enterprise 6000 (do 30 procesorów) o dużej skalowalności i wielkiej mocy obliczeniowej, specjalnie przygotowany do zadań wymagających dużej niezawodności, np. do centrów informatycznych, oraz ostatni z tej serii Enterprise Clusters - dwuklastrowy serwer przeznaczony do stosowania w systemach wymagających najwyższego poziomu niezawodności. Wszystkie modele tej serii są komputerami o architekturze symetrycznej typu SMP, w których wszystkie procesory mają równoprawny dostęp do wspólnych zasobów pamięciowych i dyskowych. Prezentacja tych nowości jednocześnie w USA i w Polsce wymownie chyba świadczy o tym, jaką wagę przywiązuje Sun do naszego rynku.

Serwery dla Internetu

Intergraph Computer Systems wdrożył do produkcji pierwszą przemysłową linię serwerów sieciowych przystosowanych do obsługi Internetu. Komputery InterServe Web pracują pod kontrolą systemu operacyjnego Windows NT Server i wykorzystują jeden, dwa lub cztery procesory Pentium lub Pentium Pro. Preinstalowane oprogramowanie zawiera m.in. Microsoft Internet Information Server, narzędzia do obsługi HTML (*HyperText Markup Language*), intergraphowski program maszynowego tłumaczenia tekstów Transced (angielski, francuski, hiszpański i niemiecki) oraz wiele innych narzędzi. InterServe Web ma wszystko co jest niezbędne do obsługi własnych stron World Wide Web o wysokiej jakości, w tym również 24-bitowe akceleratory graficzne. Dostępna jest też cała kolekcja dodatkowych urządzeń i programów do przechowywania danych, backupu, administrowania systemem i siecią (InterSTOR), InterConX i serie produktów InterSite).

Optimus współdziała z PUT TELMONT

Optimus S.A. i PUT TELMONT Sp. z o.o. podpisały umowę o współpracy, która umożliwi obu firmom oferowanie zintegrowanych usług teleinformatycznych. TELMONT istnieje od lipca 1963 r. (do 1990 r. pod nazwą Przedsiębiorstwo Montażowe Urządzeń Teleelektronicznych TELKOM-TELMONT) i oferuje m.in. usługi w zakresie budowy sieci telefonicznych, komputerowych i energetycznych, wykonuje roboty elektryczne, dostarcza siłownie, agregaty i UPS-y, prowadzi hurtownię ze sprzętem teletechnicznym, kablami, przewodami itp. Główni klienci firmy to TP S.A., kopalnie, urzędy państwowe, wojsko, policja, szpitale, banki, zakłady przemysłowe. Zrealizowała też wiele inwestycji zagranicznych. Firma współpracuje m.in. z AT&T, Alcatelem, Siemensem. Obie firmy liczą na wzmocnienie ich pozycji na rynku teleinformatycznym dzięki temu, iż mogą oferować kompleksową dostawę towarów i usług. Ma to szczególne znaczenie przy dużych przetargach

ComNet po raz drugi

Już po raz drugi odbędzie się w Warszawie (19-21 czerwca, Pałac Kultury i Nauki) wystawa i konferencja ComNet. W Ameryce w styczniu tego roku impreza miała już osiemnastą edycję. Jej tematykę zdominowały zagadnienia związane z budowanymi, nierzadko na skalę globalną, sieciami korporacyjnymi obejmującymi sieci rozległe oraz lokalne. Wśród najczęściej poruszanych na konferencji tematów znalazły się: ATM, ISDN, SMDS, Frame Relay, Internet oraz usługi OnLine, komunikacja ruchoma i bezprzewodowa. Konferencja objęła ogółem 140 sesji, w wystawie wzięło udział ponad 400 wystawców, odwiedziło ją ponad 50 tys. osób. Warszawska impreza będzie niewątpliwie skromniejsza, ale zapowiedziały w niej swój udział wszystkie najważniejsze firmy teleinformatyczne działające na naszym rynku. Będzie czynna kawiarnia internetowa "Cyberkarczma", a na towarzyszących wystawie konferencjach będzie można omówić i przedyskutować problemy i przyszłość m.in. rozwoju w Polsce Internetu, usług typu OnLine, zarządzania sieciami komputerowymi i systemami operacyjnymi.

Digital z SAP-em

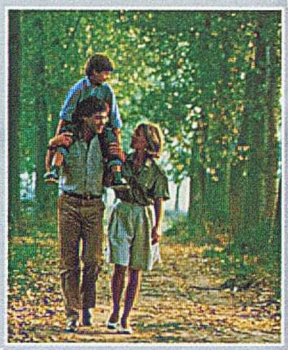
Digital Equipment Corporation i SAP AG, czołowy dostawca standardowego oprogramowania wspomagającego zarządzanie przedsiębiorstwem (pierwsze miejsce w Europie i piąte na świecie, ponad 5 tys. klientów w 41 państwach), ogłosiły, iż system SAP R/3 będzie dostępny dla rodziny komputerów Alpha Server pracujących nie tylko z systemem operacyjnym UNIX, ale także z Windows NT. Stwarza to użytkownikom możliwość wyboru między bazami danych Oracle 7 Enterprise Server i Microsoft SQL Server. Digital, który ma kilkaset instalacji systemu R/3, otworzył w marcu br. w Niemczech nowy Ośrodek Technologiczny Digital/SAP, w którym zespoły inżynierskie z obu firm zajmować się będą opracowywaniem najlepszych implementacji systemu R/3 dla zastosowań w różnych gałęziach gospodarki

Współpraca Suna z Informixem

W wyniku porozumienia zawartego między firmami Sun Microsystems i Informix Software, w Eschborn pod Frankfurtem, otworzono paneuropejskie centrum konsultacyjno-doradcze: Sun-Informix-Expertise Center. Zadaniem nowego ośrodka jest dostarczanie klientom obu firm fachowej pomocy, obejmującej wszystkie aspekty tworzenia i eksploatacji baz danych INFORMIX dla systemów Sun. Zakłada się, że Centrum będzie służyło pomocą firmom tworzącym rozbudowane i skomplikowane systemy informacyjne, które dziś powstają głównie dla obsługi sektora finansowego i telekomunikacyjnego. W Polsce istnieje wiele firm, które przygotowują systemy wykorzystujące technologię Informixa na platformie Suna. W związku z tym polskie oddziały obu firm podjęły rozmowy o współpracy na krajowym rynku. Pierwszym efektem tej współpracy są rozwiązania, które zaprezentowano na tegorocznym Infosystemie, złożone ze sprzętu firmy Sun, systemu Solaris i bazy danych Informixa.

Opracowała Krystyna Karwicka

Przekonaj się, jak wiele możliwości daje Microsoft Office dla Windows 95.



Start

Szukasz programu, który znacznie ułatwi Ci pracę? Programu, który sprawi, że codzienne zadania można będzie wykonywać o wiele szybciej i łatwiej, a czas i energię zaoszczędzone w ten sposób przeznaczyć na ulubione zajęcia?

Znalazłeś – Microsoft Office dla Windows 95 - po polsku!

To zestaw biurowy o nowych **możliwościach**, dających Ci **możliwość** spędzania czasu tak, jak lubisz najbardziej. Jak to **możliwe**? To proste:

Microsoft Office dla Windows 95 pozwala pełniej wykorzystywać dostępne funkcje pakietu biurowego, czyniąc zestaw bardziej intuicyjnym w obsłudze.

Office dla Windows 95 pozwala traktować komputer jako narzędzie pracy i nie wymaga, by użytkownicy byli ekspertami od oprogramowania.

Cały pakiet zaprojektowano tak, aby ułatwić dostęp do danych oraz do łączności z komputerami na całym świecie. Udział w globalnej wymianie informacji stanie się teraz możliwy.

Aplikacje w Office dla Windows 95 są w pełni **32-bitowe** i działają wydajniej. Korzystają ze zwiększonych zasobów systemowych, stabilności i wielozadaniowości Windows 95.

Szybsze jest przewijanie i otwieranie dokumentu. Excel wykorzystuje możliwości środowiska 32-bitowego, dzięki czemu obliczenia trwają teraz do 50 procent krócej.

Dzięki technologii **IntelliSense** wykonywanie zadań staje się łatwiejsze i szybsze niż kiedykolwiek.

Przełomowym rozwiązaniem jest również **Office Links**. Aplikacje Office dla Windows 95 współpracują ze sobą tak, jakby stanowiły jeden program.

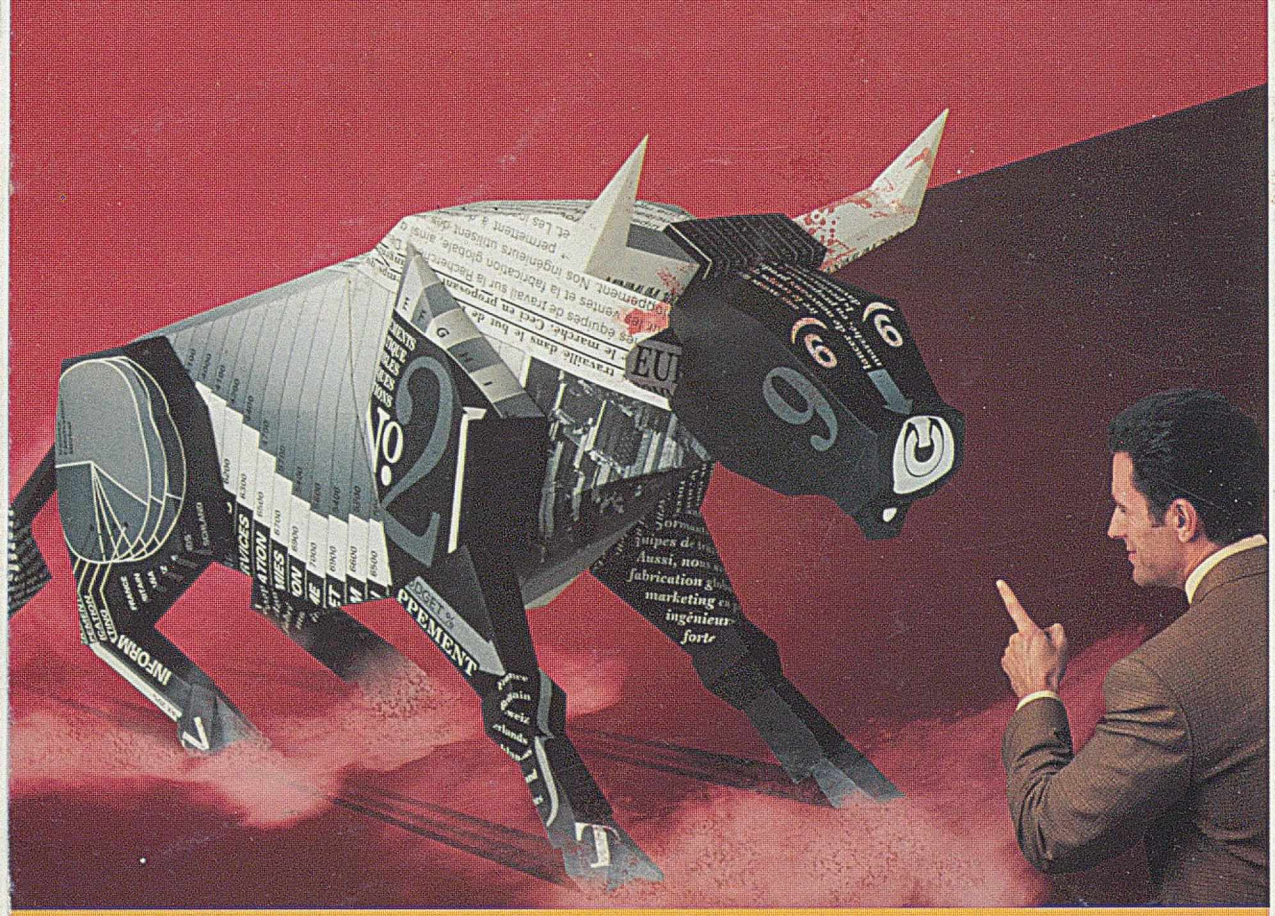
Microsoft Office dla Windows 95 – program komputerowy, który daje wiele możliwości. Kup go i przekonaj się o tym już dziś!

Microsoft Office 95 po polsku - To się dobrze składa !

Zapraszamy do odwiedzenia polskiej strony WWW firmy Microsoft:
http:/www.microsoft.com/poland

Microsoft®

WHERE DO YOU WANT TO GO TODAY?™



Niezwykłe.
Drukarki HP, które możesz zatrzymać w pełnym biegu.

Znane z ogromnej siły przetrwania, potrafią znieść trzęsienia ziemi, pożary i powodzie. Żywioły są wobec nich bezsilne. Nawet mrówki faraona nie są ich w stanie zatrzymać. Drukarki HP wykonują swoje zadania do końca, w każdych warunkach.
A teraz uwaga!
Nowa HP 5 potrafi więcej. Silna jak byk ale łagodna jak baranek. Zaopatrzona w specjalny przycisk "anulowanie pracy", gotowa zatrzymać się w każdej chwili podczas pracy za dotknięciem Twojego palca. To niezwykła zaleta dla użytkownika - może on własnoręcznie zatrzymać proces drukowania.
I ogromna ulga dla administratorów sieci - koniec z telefonami od zrozpaczonych użytkowników.
HP LaserJet 5 oprócz przycisku "anulowanie pracy" posiada ulepszoną tablicę kontrolną. Wyróżnia się też nowymi rozwiązaniami typu HP FontSmart, optional Flash, Windows 95 capability i Optional InfraRed adaptor.

Wszystko to sprawia, że HP LasertJet 5 jest godną - a co więcej, tańszą - następczynią HP LaserJet 4 Plus. Przy tym jest wyjątkowo prosta w obsłudze dla całego zespołu użytkowników. Zaawansowana technologia oprogramowania i oprzyrządowanie sprawia, że HP LaserJet 5 błyskawicznie reaguje na twoje polecenia. Posiada procesor 33 MHz i biegle włada naszym nowym językiem PCL 6. Dostępna jest w trzech różnych wariantach.
Network capable HP LaserJet 5.
LaserJet 5N dla sieci Ethernet 10 Base T.
LaserJet 5M dla zróżnicowanych środowisk.
Wszystkie one gwarantują doskonałą operatywność, redukują koszty bieżące i ..... i ...

O tak, tu można by się rozpisywać. Ale czasami warto się na chwilę zatrzymać wśród biegu wydarzeń i zastanowić. Warto ją mieć. Ten zakup procentuje.

DRUKARKI HEWLETT PACKARD