ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1994

Seria: INFORMATYKA z. 27 Nr kol. 1259

Jarosław FRANCIK
Przemysław SZMAL

WIZUALIZACJA SYNCHRONICZNA ALGORYTMÓW — REALIZACJA W SYSTEMIE SANAL

Streszczenie. Wizualizacja algorytmów (i programów) polega na graficznym zobrazowaniu kolejnych kroków ich realizacji, z nawiązaniem do ich tekstu. Wizualizacja synehroniczna — jednoczesna — kilku przebiegów programu umożliwia subiektywne porównywanie różnych aspektów ich pracy. W artykule przedstawiono metody synchronizacji wizualizowanych przebiegów wykorzystane w systemie SANAL oraz budowę i sposób działania całego systemu.

SYNCHRONOUS VISUALIZATION OF ALGORITHMS — A REALIZATION IN THE SANAL SYSTEM

Summary. The visualization of algorithms (and programs) relies on graphical illustration of successive steps of their execution, with reference to their code. Synchronous (simultaneous) visualization of several program runs lets us compare various aspects of their work. In the paper the methods for synchronization of program runs which are used in the SANAL system are presented, as well as the construction and the way of operation of the whole system.

VISUALISATION SYNCHRONIQUE DES ALGORITHMES — UNE REALISATION DANS LE SYSTEME SANAL

Résumé. La visualisation des algorithmes (et des programmes) consiste dans l'illustration graphique de pas successifs de leur execution, avec rapport à leur

texte. La visualisation synchronique (simultanée) de plusieurs programmes nous donne la possibilité à comparer des aspects divers de leur travail. Dans l'article on présente les méthodes de la synchronization des cours de programmes visualisés lesquelles on a utilisées au système SANAL, ainsi que la construction et le principe d'agir du système entier.

1. Wstęp

Znaczna część zastosowań komputerów wiąże się z wprowadzaniem pewnych danych wejściowych, przetwarzaniem ich przez program komputerowy zgodnie z pewnym algorytmem, a następnie z wyprowadzaniem wyników. Jeśli odbiorcą wyników ma być człowiek, powstaje problem ich wizualizacji, tj. przedstawienia w postaci dla niego zrozumiałej i możliwie łatwej do interpretacji.

Tam, gdzie istotne jest uchwycenie zależności jakościowych, szczególnie korzystne jest zastosowanie środków graficznych. Pojęcie wizualizacji zwykle zawęża się do wizualizacji graficznej. Wizualizacja może być statyczna — gdy w grę wchodzi zobrazowanie wybranego stanu obiektu, modelowanego za pomocą komputera, lub dynamiczna — gdy wizualizowana jest sekwencja następujących po sobie stanów.

Specjalny przypadek zachodzi, gdy przedmiotem zainteresowania jest jakiś algorytm przetwarzający dane, albo program lub jego część działająca na podstawie tego algorytmu. Z sytuacją taką mamy do czynienia w dydaktyce informatyki, zwłaszcza w nauce języków i technik programowania, oraz w analizie i projektowaniu algorytmów — w wymiarze dydaktycznym, badawczym i profesjonalnym. Wizualizacja stanu programu obejmuje problem przedstawienia przebiegu sterowania, oraz danych — tak wartości pojedyńczych, jak i wchodzących w skład abstrakcyjnych struktur danych. Struktury te obejmują w szczególności:

— zbiory liczbowe,
— ciągi wartości występujących pojedynczo, w parach, w n-tkach,
— tablice jedno- i wielowymiarowe,
— rekordy,
— struktury dynamiczne (listy, drzewa, sieci).

Problemom dynamicznej wizualizacji (często mówimy o animacji) algorytmów i programów poświęcono w ostatnich latach wiele uwagi. Opracowano wiele systemów programowych umożliwiających jej realizację. Najbardziej znany spośród nich jest system *BALSA* [1, 2] oraz systemy nowszej generacji *TANGO* [7] i *ANIMUS* [3, 4]. Podstawową koncepcję wizualizacji, przyjętą w systemie *BALSA*, a zakładającą wykorzystanie tzw.

widoków, z którymi komunikacja odbywa się za pomocą tzw. zdarzeń specjalnych, zastosowano w systemie SANAL [6, 8].

W chwili gdy dysponujemy narzędziem umożliwiającym obserwację działania pojedynczego algorytmu, w sposób naturalny pojawia się problem stworzenia warunków do porównania zachowania się (w szczególności tempa wykonania) kilku algorytmów. W tym celu trzeba stworzyć środowisko umożliwiające jednoczesną realizację tych algotymów połączoną z ich wizualizacją. Okazuje się, że najprostsze rozwiązanie, polegające na uruchomieniu wszystkich badanych procesów w środowisku wieloprogramowym, może nie dać pożądanych efektów. Wiąże się to ze znacznym narzutem, jakie wnosi konieczność realizacji operacji graficznych związanych z wizualizacją, i z uzależnieniem tego narzutu od kontekstu, w jakim realizowany jest pokaz/projekcja. Na ten kontekst składa się między innymi:

— kształt i wielkość obiektów graficznych reprezentujących dane: pośrednio może to być związane z wielkością okna, w którym prezentowane są dane,

— obecność lub brak elementów graficznych drugoplanowych stanowiących tło czy też poziom odniesienia ułatwiający obserwatorowi obserwację lub ocenę.

Oznacza to, że przebiegi poszczególnych programów muszą być ze sobą synchronizowane dla zapewnienia obserwatorowi subiektywnego wrażenia wykonania jednoczesnego, z tempem rozwoju wydarzeń w poszczególnych programach adekwatnym do tego, jakie występowałoby w przypadku wykonania „normalnego", nie obciążonego narzutem na wizualizację.

Okazuje się, że synchronizowana wizualizacja przebiegów realizowana w systemach z najuboższym mechanizmem pracy quasi-współbieżnej, opartym na współprocesach, może dać interesujące efekty. System SANAL korzysta obecnie właśnie z takiego środowiska. Zastosowane w systemie rozwiązania, które przedstawiamy w dalszej części tego artykułu, można będzie z powodzeniem wykorzystać w bardziej rozwiniętych środowiskach pracy wieloprogramowej.

2. Charakterystyka systemu SANAL

System animacji algorytmów SANAL powstał w Instytucie Informatyki Politechniki Śląskiej w roku 1989. Kolejne wersje tworzono w latach 1990-1993[1]. Obecnie dostępna

[1] Źródła finansowania: w latach 1989-1990 — Resortowy Program Badawczo-Rozwojowy *RI-14*, w latach 1990-1991 — grant MEN nr *T/15/036/90-2 (G-615/RAu2/91)*, w roku 1993 — fundusz tzw. Badań Własnych (BW-331/RAu/93).

jest wersja systemu SANAL 3.1. Podobnie jak wersje pierwotne, napisana jest w języku *Turbo Pascal*. System jest przystosowany do pracy na komputerach zgodnych z *IBM PC* pod systemem *DOS*.

System jest wyposażony w obszerny zestaw środków prezentacji graficznej danych i przebiegu sterowania programów. Posiada mechanizmy pozwalające na dostosowanie treści i tempa pokazu/projekcji do potrzeb użytkownika oraz narzędzia ułatwiające przygotowanie programów do pokazu — dostosowanie ich do współpracy z systemem. W skład systemu SANAL wchodzi też zestaw programów gotowych do animacji.

Przygotowanie algorytmów do animacji w systemie SANAL przebiega w kilku etapach. Algorytm musi być dostarczony w formie programu napisanego w *Turbo Pascalu*. Po zdecydowaniu, co i w jaki sposób powinno zostać zobrazowane podczas projekcji, wprowadzane są do tekstu programu wywołania funkcji usługowych systemu SANAL. Prezentację przygotowanego programu możemy zlecić systemowi, gdyż dalsza obróbka następuje automatycznie, i obejmuje: automatyczne wstawienie dodatkowych instrukcji służących do wizualizacji przepływu sterowania programu, kompilację i uruchomienie otrzymanego programu.

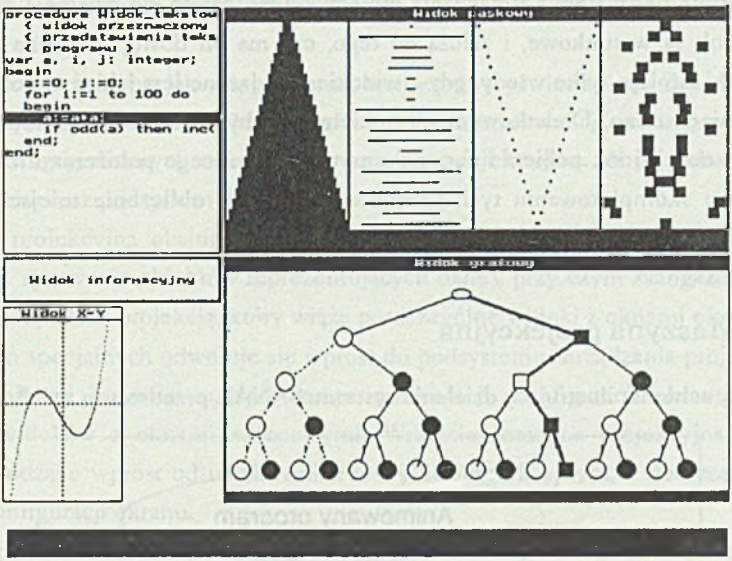
2.1. Mechanizm wizualizacji

Wstawianie wywołań funkcji usługowych do tekstu samodzielnie animowanego programu wymaga od użytkownika wiedzy na temat ogólnych zasad zarządzania projekcją i mechanizmu wizualizacji, a także zawartości biblioteki funkcji usługowych systemu.

U podstaw wizualizacji przebiegu programów w systemie SANAL leży pojęcie widoku. Dla programu animowanego widok jest obiektem oferującym określony zasób środków do graficznej prezentacji danych. Wyróżnia się kilka typów widoków. Charakterystyczne dla nich formy prezentacji zebrano na rysunku 1. Zestaw klas widoków jest otwarty (użytkownik może tworzyć nowe rodzaje we własnym zakresie). Standardowo wyróżnia się:

— widoki tekstowe, przeznaczone do wyświetlania tekstu źródłowego programu, z podświetleniem aktualnie wykonywanej instrukcji. Przygotowanie tekstów programów do obsługi widoków tej klasy jest zautomatyzowane,
— widoki informacyjne, pozwalające na wyświetlenie informacji w postaci łańcucha tekstowego,
— widoki X-Y, prezentujące pary wartości liczbowych w postaci punktów (lub innych elementów graficznych) na wykresie,

— widoki paskowe, wizualizujące wartości liczbowe w postaci pasków (belek), kresek lub punktów, o długości lub rozpiętości proporcjonalnej do prezentowanej wartości,
— widoki grafowe, prezentujące struktury grafowe i drzewiaste.



Rys. 1. Typy widoków występujące w systemie SANAL
Fig. 1. Types of views used in the SANAL system

W zależności od typu widoki oferują różne operacje graficzne, takie jak wykreślenie tekstu, punktu, paska, czy też węzła lub krawędzi grafu. Widok jest związany również z zestawem charakteryzujących go parametrów, określających organizację widoku, sposób odwzorowania wartości i rysowania elementów graficznych. Widok nie zawiera natomiast informacji dotyczących położenia i rozmiaru obszaru ekranu, na którym mają być zlokalizowane tworzone elementy wizualizacji graficznej.

Poza niewielkim obszarem przeznaczonym do wyświetlania komunikatów systemowych, obsługiwany przez system SANAL podczas projekcji ekran zawiera pewną liczbę prostokątnych okien, o położeniu i rozmiarze kontrolowanym przez animowany program lub użytkownika. Okna są numerowane, i mogą być włączone lub schowane. Włączenie okna nie przesądza o jego widoczności na ekranie: jeśli okno nie zawiera żadnych elementów graficznych, pozostaje niewidoczne.

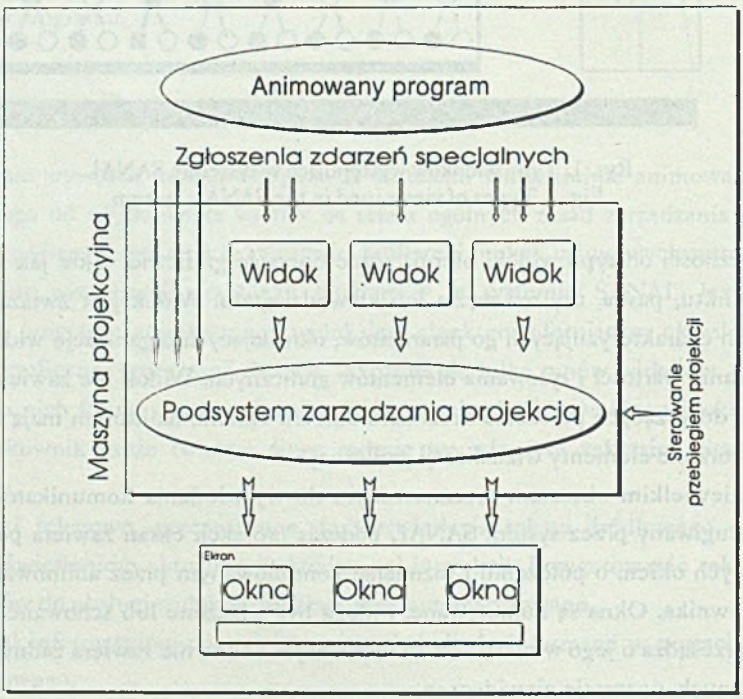
Pomiędzy widokami biorącymi udział w projekcji, a podzielonym na okna ekranem pośredniczy podsystem zarządzania projekcją: przydziela on okna poszczególnym widokom, co umożliwia wyświetlenie elementów graficznych widoku w określonym miej-

scu na ekranie. Związek widoku z oknem może być dowolnie nawiązywany, zrywany, przywracany i zmieniany.

Każda operacja graficzna wykonywana przez widok odbywa się za pośrednictwem i pod kontrolą podsystemu zarządzania projekcją. Oznacza to, że działania podejmowane przez widok są warunkowe, i zależą od tego, czy ma on dostęp do okna ekranowego. Dostęp taki istnieje tylko wtedy, gdy z widokiem związane jest jakieś okno; mówimy, że widok posiada okno. Dodatkowym warunkiem jest, by okno nie było akurat schowane (niewidoczne). Widok pobiera informację na temat bieżącego położenia i rozmiaru okna. Dopiero po skompletowaniu tych danych możliwe jest obliczenie miejsca na ekranie, w którym narysowany zostanie element graficzny.

2.2. Maszyna projekcyjna

Ogólny schemat ilustrujący działanie systemu SANAL przedstawia rys. 2.



Rys. 2. Uproszczony schemat mechanizmu obsługi projekcji w systemie SANAL
Fig. 2. Simplified scheme of the projection handling mechanism in the SANAL system

Wszystkie operacje na zawartości widoków oraz konfiguracji okien i innych parametrów systemu są wykonywane na żądanie programu animowanego (z wyjątkiem operacji wykonywanych w trybie interakcyjnym). Żądanie takie jest traktowane jako zgłoszenie tzw. **zdarzenia specjalnego**. Jest ono realizowane przez wywołanie odpowiedniej procedury.

Zgłaszanie zdarzeń specjalnych jest formą sterowania wyświetlaniem. Zespół elementów zapewniających prawidłową obsługę zgłoszeń i, w konsekwencji, realizujących określony zasób usług graficznych nazywamy **maszyną projekcyjną**. W jej skład wchodzi zestaw widoków oraz podsystem zarządzania projekcją.

Maszyna projekcyjna obsługuje zdarzenia specjalne odnoszące się do określonych widoków (np. rysowanie obiektów reprezentujących dane), przy czym zaangażowany jest podsystem zarządzania projekcją, który wiąże poszczególne widoki z oknami ekranowymi. Część zdarzeń specjalnych odwołuje się wprost do podsystemu zarządzania projekcją: dotyczą one, między innymi, ustalania tempa prezentacji, konfiguracji ekranu, a także powiązania widoków z oknami ekranowymi. Wreszcie maszyna projekcyjna realizuje żądania pochodzące wprost od użytkownika w trybie interakcyjnym, a dotyczące tempa projekcji i konfiguracji ekranu.

Ponieważ w prawie każdej prezentacji zaangażowanych jest więcej niż jeden widok, zachodzi potrzeba kierowania zdarzeń specjalnych do odpowiednich widoków. Z punktu widzenia użytkownika w zasadzie wystarczające jest podanie nazwy zdarzenia specjalnego (tj. nazwy procedury je obsługującej), gdyż nazwy te zawierają określenie typu widoku. Podczas prezentacji algorytmu tylko w wyjątkowych sytuacjach stosuje się kilka widoków tego samego typu. Z tego powodu dodatkowe podawanie identyfikatora konkretnego widoku jest niepotrzebne.

W obecnej wersji systemu zastosowano zorientowaną obiektowo strukturę widoków. Maszyna projekcyjna zawiera zmienne obiektowe reprezentujące tzw. widoki domyślne, po jednej na każdy typ widoku (mimo że dopuszczalna jest większa liczba widoków tego samego typu). Procedury obsługi zdarzeń specjalnych dotyczących widoku określonego typu przekazują sterowanie do odpowiedniej metody usługowej widoku domyślnego tego typu. Widoki poszczególnych typów zaimplementowano jako klasy pochodne względem klasy tzw. widoków abstrakcyjnych, które realizują pewne operacje wspólne dla wszystkich widoków, a dotyczące przede wszystkim komunikacji z podsystemem zarządzania projekcją.

W przypadku prezentacji wymagających zastosowania więcej niż jednego widoku tej samej klasy tylko jeden z nich jest uznawany za domyślny, i są do niego kierowane wszystkie zgłoszenia zdarzeń specjalnych związanych z widokami tej klasy. W takim przy-

padku użytkownik może nadać status widoku domyślnego jednemu z pozostałych widoków w kolejności dyktowanej potrzebami projekcji. Może również korzystać wprost z obiektowo zorientowanej implementacji widoków.

3. Środowisko projekcji synchronicznej

W obecnej wersji systemu SANAL wprowadzono środki umożliwiające synchroniczną prezentację kilku algorytmów jednocześnie. Każdy z tych algorytmów jest zaimplementowany w postaci sekwencji instrukcji, będących wydzieloną procedurą lub programem, które są wykonywane jednocześnie, przynajmniej w subiektywnym odczuciu użytkownika.

Program przygotowany do prezentacji indywidualnej nadaje się w systemie SANAL do projekcji synchronicznej bez żadnych modyfikacji[2], i może być ewentualnie wzbogacony o dodatkowe zgłoszenia zdarzeń specjalnych związanych z precyzyjną synchronizacją procesów. Przygotowanie projekcji synchronicznej wymaga od użytkownika jedynie wskazania (w trybie interakcyjnym), prezentację których algorytmów należy zrealizować, i, ewentualnie, w jakim trybie, tempie *etc.*

3.1. Organizacja projekcji quasi-równoległej

System SANAL korzysta ze specjalnie zaprojektowanego modułu implementującego quasi-równoległe (naprzemienne) wykonywanie programów (procesów) w *Turbo Pascalu [5]*. Przełączenie sterowania pomiędzy procesami odbywa się jawnie, na skutek wywołania odpowiedniej procedury.

Poszczególne biorące udział w prezentacji algorytmy są reprezentowane przez podprogramy umieszczone w oddzielnych modułach. Główny program realizujący projekcję inicjalizuje wykonywanie wszystkich podprogramów jako procesów quasi-równoległych. Następnie sterowanie przekazywane jest do pierwszego procesu. Po pewnym czasie proces ten przekazuje sterowanie z powrotem do programu głównego. Program główny realizuje przełączenie do kolejnego procesu, który także po pewnym czasie zwraca sterowanie do programu głównego, i tak dalej. W ten sposób, cyklicznie, sterowanie jest kolej-

[2] Wyjątek stanowią programy ustalające rozmiar okienek ekranowych w sposób arbitralny: może to prowadzić do konfliktów w przypadku jednoczesnego uruchomienia kilku z nich.

no przekazywane do wszystkich aktywnych procesów, aż do chwili zakończenia ostatniego z nich. Takie rozwiązanie zapewnia żywotność wszystkich procesów.

Z myślą o uwolnieniu użytkownika od troski o wprowadzenie wywołań procedury przełączającej procesy do wszystkich tekstów programów przygotowywanych do animacji synchronicznej wywołania te zaszyto w treści procedur obsługujących zdarzenia specjalne powodujące wyświetlenie elementów graficznych na ekranie: częstsze przełączenia i tak nie byłyby widoczne. Poza tym programy przygotowane do projekcji indywidualnej są jednocześnie prawidłowo animowane w trybie synchronicznym.

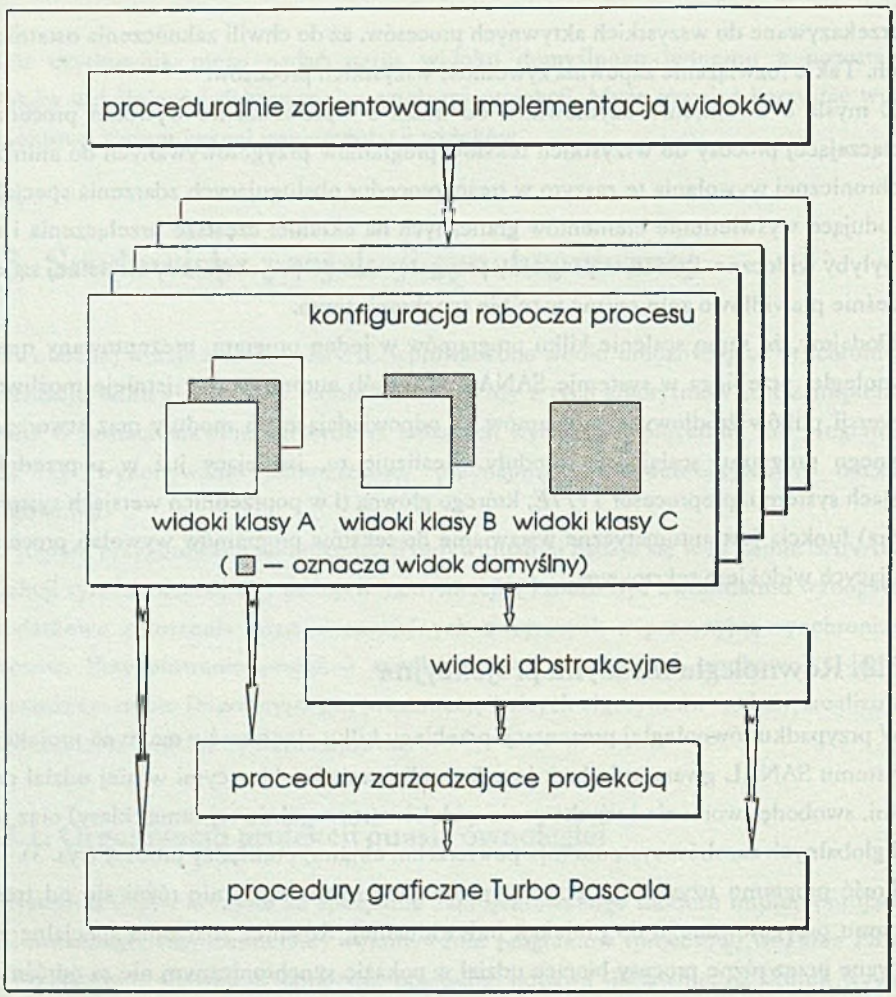
Dodajmy, że samo scalenie kilku programów w jeden program, prezentowany quasi-równolegle, przebiega w systemie SANAL w sposób automatyczny: istnieje możliwość konwersji plików źródłowych programów na odpowiadające im moduły oraz stworzenia głównego programu scalającego moduły. Realizuje to, istniejący już w poprzednich wersjach systemu, preprocesor *TVPIE*, którego główną (i w poprzednich wersjach systemu jedyną) funkcją jest automatyczne wstawianie do tekstów programów wywołań procedur sterujących widokiem tekstowym.

3.2. Równoległa maszyna projekcyjna

W przypadku równoległej prezentacji przebiegu kilku algorytmów maszyna projekcyjna systemu SANAL gwarantuje brak interferencji pomiędzy biorącymi w niej udział procesami, swobodę tworzenia i użytkowania widoków (szczególnie tej samej klasy) oraz podział globalnych zasobów systemu (np. powierzchni ekranu) pomiędzy procesy (rys. 3).

Treść programu używanego podczas prezentacji równoległej nie różni się od treści programu przygotowanego do projekcji indywidualnej. Również zdarzenia specjalne generowane przez różne procesy biorące udział w pokazie synchronicznym nie są odróżnialne, to znaczy, nie zawierają parametrów pozwalających na odróżnienie procesów. Mimo to maszyna projekcyjna w zróżnicowany sposób obsługuje formalnie identyczne zgłoszenia nadchodzące z różnych procesów. Jest to warunek konieczny do tego, by każdy biorący udział w pokazie proces miał zagwarantowane wirtualne środowisko, składające się między innymi ze swobodnie tworzonych i dysponowanych widoków i okien ekranowych, oraz by środowisko to odpowiadało określonemu przez system podzbiorowi dostępnych fizycznie zasobów, który to podzbiór jest odrębny od podzbiorów przydzielonych pozostałym współprocesom.

Zastosowane narzędzia implementujące podstawowy mechanizm quasi-równoległości pozwalają na sprawdzenie, który proces jest w danej chwili aktywny (są one odróżniane na podstawie przypisanych im identyfikatorów liczbowych).



Rys. 3. Struktura równoległej maszyny projekcyjnej systemu SANAL
Fig. 3. Structure of the parallel projection machine in the SANAL system

Stosując obiektowo zorientowany model struktury widoków, a więc model, którego istotną cechą jest hermetyzacja danych, dość łatwo jest zagwarantować brak interferencji między procesami na tym poziomie[3]. Model ten znosi również wszelkie ograniczenia swobody tworzenia i użytkowania widoków, również tej samej klasy.

Na poziomie procedur obsługi zdarzeń specjalnych powielono zmienne wskazujące na domyślne widoki, tak by określić można było widok domyślny dla każdego typu widoku i

[3] Była to jedna z najistotniejszych przesłanek, jakie zadecydowały o przebudowie architektury systemu SANAL na zorientowaną obiektowo.

identyfikatora procesu. Podczas prezentacji quasi-równoległej maszyna projekcyjna przekazuje sterowanie do odpowiedniej metody usługowej widoku domyślnego danego typu w obrębie aktywnego procesu.

Zmodyfikowano mechanizm przydzielania okien ekranowych. Maszyna projekcyjna przydziela poszczególnym procesom okna o określonych (konfigurowalnych) zakresach numerów. Dostępna dla programów numeracja okien jest względna, a oknom o tych samych numerach logicznych w obrębie różnych procesów odpowiadają różne okna fizyczne.

3.3. Synchronizacja projekcji

Procesy w systemie SANAL nie odwołują się do klasycznych środków synchronizacji (jak np. semafory), gdyż nie występują między nimi zależności przyczynowo-skutkowe. Można je zatem zaliczyć do procesów niesynchronizujących się, a z formalnego punktu widzenia to, czy zostaną one wykonane sekwencyjnie, czy równolegle, nie ma wpływu na poprawność ich wykonania. W praktyce projekcja algorytmów w sposób sekwencyjny, jeden po drugim, nie byłaby, rzecz jasna, zbyt pouczająca. Skoro wprowadzenie możliwości quasi-równoległego wykonywania procesów w systemie SANAL miało na celu uzyskanie subiektywnego wrażenia równoczesności wykonywania procesów, wprowadzenie mechanizmów synchronizujących motywuje się potrzebą uzyskania wrażenia równomierności tempa wykonywania procesów.

Możliwość porównania cech charakterystycznych dla kilku prezentowanych jednocześnie programów jest najlepsza wtedy, gdy w każdej chwili wszystkie programy wykonują analogiczne operacje, np. w sposób zsynchronizowany realizują kolejne, porównywalne kroki algorytmów. Skuteczniej oszacować można również względną złożoność obliczeniową algorytmów, gdyż na wyniki porównań algorytmów nie ma wpływu czasochłonność operacji uznawanych za elementarne. Z drugiej strony należy zauważyć, że czasami wprowadzenie synchronizacji uwarunkowanej względami wizualnymi może zakłócić oszacowanie złożoności, a nawet prowadzić do błędnych wniosków w tym zakresie.

W kolejnych podpunktach omówimy wszystkie dostępne w systemie SANAL środki służące synchronizacji projekcji quasi-równoległych.

Synchronizacja naturalna

Często się zdarza, że tempo wykonywania różnych procesów w projekcji quasi-równoległej sprawia wrażenie wystarczająco jednolitego. Wynika to z faktu, że przełączenie sterowania pomiędzy procesami następuje w trakcie wykonywania procedur obsłu-

gujących zdarzenia specjalne. Algorytmy, których specyfika narzuca wykorzystanie podobnych środków wizualizacji, często zgłaszają zdarzenia specjalne z taką samą lub podobną częstością.

Maskowanie zdarzeń specjalnych

Maskowanie zdarzeń specjalnych jest narzędziem pozwalającym na wzmocnienie opisanej wyżej synchronizacji naturalnej poprzez wskazanie, obsługa których zdarzeń ma powodować przełączenie sterowania. Standardowo przełączeń nie powodują zdarzenia reżyserskie, to jest nie powodujące wykreślenia nowych elementów graficznych na ekranie.

Jawne przełączenie sterowania

Użytkownik ma zapewniony dostęp do procedury przełączającej sterowanie do następnego, biorącego udział w projekcji, procesu.

Określanie poziomu procesu

Poziom procesu jest wartością z przedziału *0..1*, pozwalającą na dostrojenie szybkości wykonywania danego procesu względem innych procesów. Ustalenie poziomu na *0* jest *de facto* równoznaczne całkowitemu jego wstrzymaniu. Wartości większe od zera określają względną częstość przekazywania sterowania do danego procesu.

Punkty synchronizacji

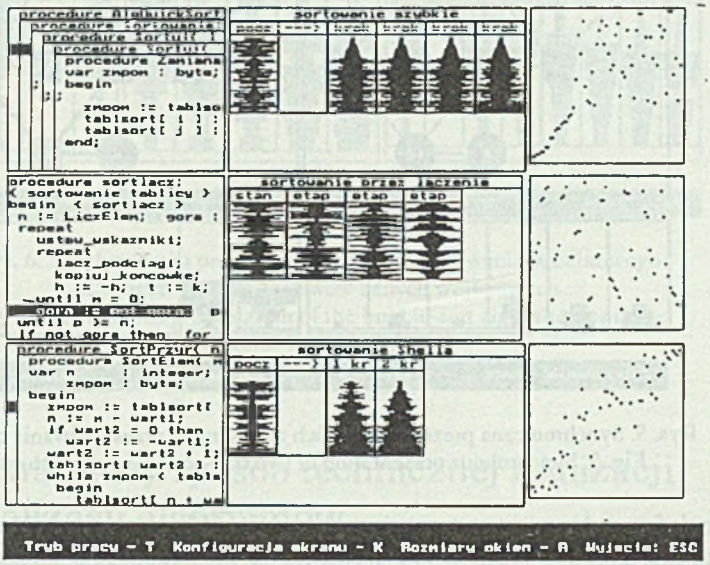
Punkty synchronizacji to najbardziej precyzyjne spośród narzędzi synchronizujących dostępnych w systemie SANAL. Zaimplementowano je w postaci specjalnej procedury, która wstrzymuje wykonanie procesu aż do momentu, gdy wszystkie uruchomione procesy również osiągną punkty synchronizacji. Umieszczając w różnych algorytmach wywołanie tej procedury w zbliżonych znaczeniowo miejscach, szczególnie na granicach głównych etapów pracy, otrzymamy precyzyjne zsynchronizowanie procesów.

Punkty synchronizacji można różnicować poprzez nadawanie im priorytetów. Jeśli chociaż jeden proces osiągnie punkt synchronizacji o pewnym priorytecie, wszystkie punkty o priorytecie niższym są ignorowane. Jeśli jakieś procesy zostały zatrzymane na punkcie synchronizacji o niższym priorytecie, zostaną niezwłocznie wznowione; osiągnięcie nowych punktów o niższym priorytecie również zostanie uznane za niebyłe.

Punkt o najwyższym priorytecie (równym 255) jest wykorzystany przez system SANAL do synchronizacji zakończenia pracy różnych procesów. Jest on pierwszą instrukcją standardowej procedury *IEFinish*, która powinna być ostatnią instrukcją wykonywaną przez programy przygotowane do animacji. Dzięki temu zakończenie animacji następuje dopiero po zakończeniu wszystkich biorących w niej udział procesów.

4. Przykłady działania systemu SANAL

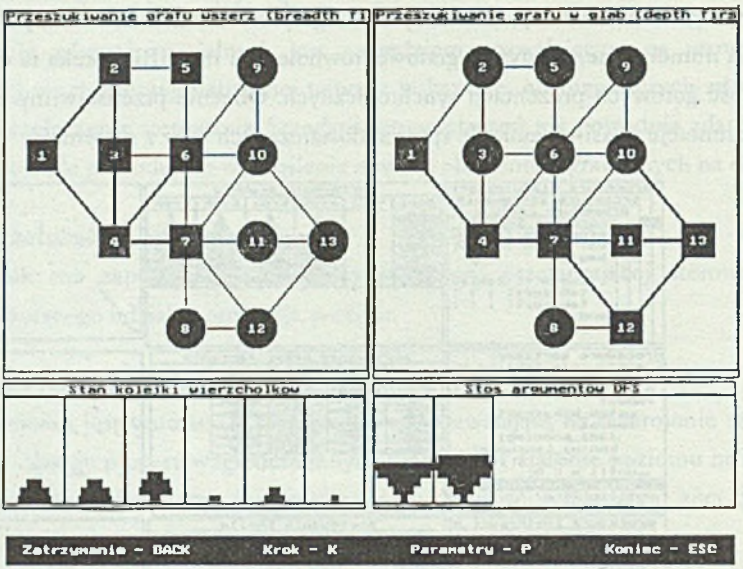
Wraz z systemem SANAL 3.1 rozpowszechniana jest biblioteka przygotowanych do animacji programów, realizujących klasyczne algorytmy z takich dziedzin, jak sortowanie, obliczenia numeryczne, algorytmy grafowe, równoległe i inne. Biblioteka ta obejmuje też pewną ilość gotowych prezentacji synchronicznych. Obecnie przedstawimy trzy przykładowe prezentacje quasi-równoległe spośród dostarczanych wraz z systemem.



Rys. 4. Synchroniczna prezentacja trzech zaawansowanych algorytmów sortowania
Fig. 4. Synchronous presentation of three advanced sort algorithms

Rysunek 4 przedstawia porównanie trzech zaawansowanych algorytmów sortowania: sortowania szybkiego *(quick sort)*, sortowania przez łączenie *(merge sort)* i sortowania Shella *(Shell sort)*. Oprócz widoków tekstowych, każdy z algorytmów jest wizualizowany przy użyciu widoków paskowego i X-Y, co pozwala dojrzeć charakterystyczny, makroskopowy układ sortowanych elementów ciągu i poznać zasady strategii działania każdego z nich, często nieuchwytne — nawet po dogłębnej analizie. I tak, w górnym rzędzie okienek widzimy efekt działania algorytmu sortowania szybkiego (mniej więcej w połowie pracy), który metodycznie dzielił sortowany ciąg na podciągi i, jak dotąd, posortował już całą pierwszą połowę. Zauważmy, że obrazek ten jest charakterystyczny tylko dla rekurencyjnej wersji tego algorytmu. Poniżej, ułożony w charakterystyczne pasma, ciąg sortowany metodą przez łączenie. W następnym kroku pary sąsiednich pasem połączą się w nowe, dwu-

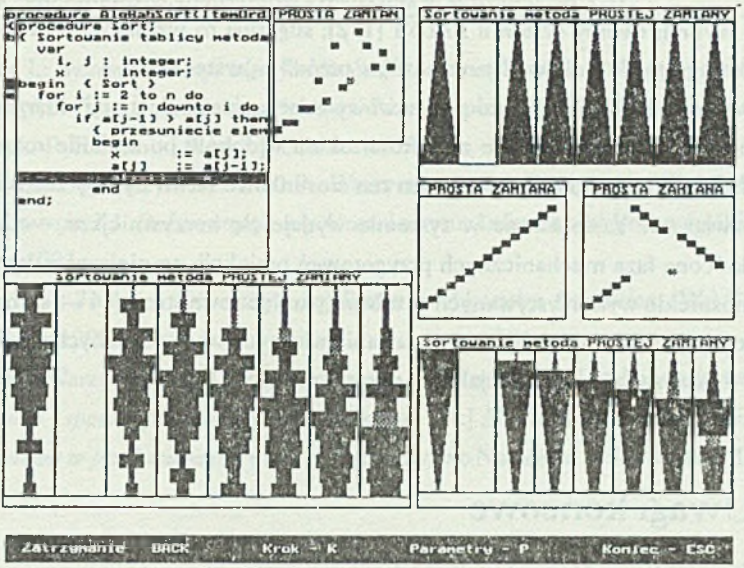
krotnie szersze pasma. Na samym dole obserwujemy algorytm Shella, który, działając przez cały czas z równą skutecznością na każdy element ciągu, utworzył rodzaj mgławicy: będzie się ona stawać coraz smuklejsza aż do całkowitego uporządkowania elementów.



Rys. 5. Synchroniczna prezentacja dwóch algorytmów przeszukiwania drzew
Fig. 5. Synchronous presentation of two tree-searching algorithms

Rysunek 5 przedstawia zastosowanie systemu SANAL do porównania dwóch strategii przeszukiwania grafu: z lewej strony wizualizację przeszukiwania wszerz *(breadth first search)*, a z prawej — w głąb *(depth first search)*. W tym przypadku jednoczesna prezentacja pozwala zademonstrować różnicę kolejności przeszukiwania wierzchołków w obu metodach. Kwadraty na rysunku reprezentują wierzchołki grafu już sprawdzone. Zaznaczone grubszą kreską krawędzie grafu oznaczają drzewo rozpinające, którego kształt również zależy od wybranego algorytmu. Dodatkowo, użytkownik obserwować może stan kolejki wierzchołków (przy przeszukiwaniu wszerz) i stos argumentów (przy przeszukiwaniu w głąb): szerokość pasków odpowiada numerowi zapamiętanego wierzchołka.

SANAL pozwala na równoczesną obserwację działania tego samego algorytmu z różnymi zestawami danych wejściowych. Rysunek 6 przedstawia wizualizację algorytmu sortowania przez prostą zamianę (sortowanie bąbelkowe, *bubble sort)*, zastosowanego do ciągu o losowym układzie elementów, uporządkowanego rosnąco oraz malejąco.



Rys. 6. Synchroniczna prezentacja algorytmu sortowania bąbelkowego dla trzech różnych zestawów danych wejściowych
Fig. 6. Synchronous presentation of the bubble-sort algorithm for three various input data sets

5. Alternatywny sposób technicznej realizacji wizualizacji algorytmów

Przedstawiona w poprzednich rozdziałach koncepcja synchronicznej prezentacji kilku przebiegów programów w ramach systemu SANAL zakłada dostarczenie dla maszyny projekcyjnej danych do wyświetlenia „na żywo" — są one generowane na bieżąco przez zbiór procedur, zawierających implementacje wybranych algorytmów, aktywizowanych cyklicznie według określonych zasad.

Można sobie wyobrazić inny tryb realizacji projekcji. Przyjmijmy, że wykonujemy sekwencyjnie wszystkie przebiegi algorytmów „cząstkowych" i rejestrujemy (np. w oddzielnych plikach dyskowych) wszystkie zdarzenia specjalne, jakie mają miejsce w trakcie tych przebiegów. W trakcie właściwej projekcji wystarczałoby odczytywać w sposób skorelowany zarejestrowane parametry zdarzeń przebiegów cząstkowych i przekazywać je do maszyny projekcyjnej. Analogia takiego rozwiązania z projekcją filmową jest wyjątkowo silna. Jak się wydaje, podobną drogę w stosunku do pojedynczych

projekcji wybrali twórcy systemu *BALSA* [1, 2]; sugerują to wzmianki o próbach wizualizacji poszczególnych kroków algorytmów „w przód" i „wstecz".

W systemie SANAL istnieje podstawowy mechanizm rejestracji zdarzeń — dzięki niemu możliwe jest odtwarzanie zawartości okien widoków po zmianie rozmiarów okna w dowolnej chwili projekcji. Mechanizm ten stosunkowo łatwo byłoby zaadaptować. Jednakże rozwiązanie zastosowane w systemie wydaje się korzystniejsze — dzięki niemu została skrócona faza mechanicznych przygotowań projekcji, zmniejszono wymagania systemu odnośnie do wykorzystywanych zasobów (pamięć zewnętrzna!) i — co najważniejsze — użytkownik zachował możliwość podawania własnych wartości danych wejściowych, co znacznie rozszerza zakres potencjalnych eksperymentów.

6. Uwagi końcowe

System SANAL stanowi od 1989 swojego rodzaju poligon doświadczalny dla prac dotyczących wizualizacji algorytmów prowadzonych w Instytucie Informatyki Politechniki Śląskiej. Postęp w zakresie narzędzi i środowisk programowych stymuluje powstawanie nowych pomysłów i daje techniczną możliwość ich realizacji. I tak, w ostatnim okresie system zyskał nowe możliwości dzięki zmodyfikowaniu struktury z modułowej na zorientowaną obiektowo. Z punktu widzenia tematu tego artykułu szczególnie ciekawe będzie dostosowanie systemu do współpracy z *MS Windows* (wersja obecna współpracuje z systemem *DOS).* Stworzy to okazję do dalszego doskonalenia mechanizmów synchronicznej prezentacji przebiegu algorytmów.

LITERATURA

[1] Brown M. H., Sedgewick R.: *A System for Algorithm Animation.* Computer Graphics: SIGGRAPH'84 Conference Proceedings. Minneapolis, Minn. 18(3) July 23-27 1984.

[2] Brown M. H.: *Exploring Algorithms Using Balsa.* W: *Computer,* Vol. 21, No 5, May 1988.

[3] Duisberg R.A.: *Animated Graphical Interfaces Using Temporal Constraints.* W: *Proc. ACM SIGCHI* 86. April 1986.

[4] Duisberg R. A.: *Animation Using Temporal Constraints, an Overview of the Animus System.* Computer Research Laboratory, Tektronix, Inc.

[5] Francik J.: *System animacji algorytmów SANAL z możliwością realizacji projekcji quasi-równoległych (wersja jednostanowiskowa).* Praca dyplomowa magisterska. Instytut Informatyki Politechniki Śląskiej, Gliwice 1993.

[6] Francik J., Lepiarczyk S., Szmal P., Warzyszyńska B., Wielgus A., Wolek S.: *System SANAL wersja 3.1. Dokumentacja użytkowa.* Instytut Informatyki Politechniki Śląskiej, Gliwice 1993.

[7] Stasko J. T.: *Tango: A Framework and System for Algorithm Animation.* [W:] Computer, September 1990.

[8] Szmal P., Warzyszyńska B.: *System animacji algorytmów SANAL. Koncepcja wizualizacji oraz środki i sposoby prezentacji danych w systemie.* [W:] *Materiały Sympozjum „Wizualizacja komputerowa w przekazie edukacyjnym".* Wydawnictwo Naukowe WSP, Kraków 1991.

Recenzent: Dr inż. Maciej Bargielski

Wpłynęło do Redakcji 12 stycznia 1994 r.

Abstract

The system for algorithm animation SANAL makes possible graphical visualization of successive steps of execution of selected algorithms with reference to their code written in the programming language *Pascal*. In the system the conception of event-driven program visualization was applied, using abstract areas, called views, for data presentation, analogously as in the BALSA system [1, 2]. During its demonstrative run (called projection) the program being visualized collaborates with so-called projection machine (fig. 3). The projection machine is a complex software product composed conforming to object oriented programming technique. In particular, various types of views and the configurations of individual processes are represented by objects of respective classes.

A novelty in current system version (SANAL 3.1) is a possibility of simultaneous, synchronous visualization of different programs (as shown in fig. fig. 4 and 5), or of different runs of the same program (as shown at fig. 6). These new possibilities arose thanks to the generalization of the projection management mechanism.

In order to get a subjective impression of simultaneous realization of several program runs, one can use mechanisms of natural synchronization, with or without additional syn-

chronization points. It is also possible to mask "interesting events", to switch control flow explicitly, and to define process levels.

The basic switching mechanism is hidden in the procedures that handle events. This is why each program prepared to an individual projection can be presented — without extra manipulations — in the group-projection mode.

In the paper an alternative possibility of realization of synchronous projection is outlined. It should consist in replaying previously registered sequences of events. However, this method is more restrictive, from the user's point of view.