STUDIA INFORMATICA 2002
Volume 23 Number 2A (48)

Krzysztof DOBOSZ
Politechnika Śląska, Instytut Informatyki

NANOTECHNOLOGIA W PROGRAMOWANIU KOMPUTERÓW[1)]

Streszczenie. W publikacji pokazano, że elementy nanotechnologii, charakterystyczne dla molekularnych nanosystemów, mogą być również wykorzystywane w procesie projektowania i tworzenia programów komputerowych.

NANOTECHNOLOGY IN COMPUTER PROGRAMMING

Summary. The paper presents nanotechnological elements, which are characteristic for molecular nanosystems. Those can be used in the process of development and computer program creating.

1. Wstęp

Nanotechnologia to pojęcie opisujące dziedzinę zjawisk fizycznych, chemicznych i biologicznych zachodzących w skali molekularnej. Obejmuje ono zarówno mechanizmy wytwarzania molekularnego, jak i techniki projektowania i analizy nanosystemów [1][2][3]. Budowa molekularnych struktur z atomowych „klocków Lego", czy projektowanie nanomaszyn to jeden z najnowszych trendów w światowej nauce. W niniejszym artykule skupimy uwagę na tych elementach nanotechnologii, które można wykorzystać nie tylko w skali nanoelementów, ale także w większej – makroskali, jaką z pewnością jest środowisko pamięci operacyjnej współczesnych komputerów.

[1)] Projekt KBN nr 7 T11C 017 21

2. Elementy nanotechnologii w programowaniu komputerów

Organizmy biologiczne zbudowane są z pojedynczych komórek. Pełnią one różne funkcje, ale są do siebie stosunkowo bardzo podobne pod względem budowy fizycznej. Mają one też pewną autonomię, ponieważ same mogą się replikować zgodnie z planem rozwoju organizmu zapisanym w swoim kodzie genetycznym. Poza tym mogą się one kontaktować między sobą i z otoczeniem.

Trudno pokazać takie zachowania w przypadku programów komputerowych przede wszystkim dlatego, że nie mamy tutaj do czynienia z replikacją obiektów materialnych. Jeśli jednak pominiemy ten fakt, to zauważymy, że elementy nanotechnologii są rozpoznawalne również w przypadku niektórych aplikacji programowych. Można wtedy wskazać takie elementy nanotechnologii, jak:

- komórkę inicjującą – zawierającą molekularny zapis programu i nanotechnologiczny sposób jego rozwoju,
- rozwój programu przebiegający poprzez samoreplikację według określonego schematu – od komórki inicjującej poprzez embrion, aż do osiągnięcia zaplanowanej postaci programu,
- wielostopniowość rozwoju pojedynczej komórki.

W dalszej części artykułu słowo „komórka" określać będzie część składową programowej aplikacji rozproszonej, którą jest proces lub wątek.

2.1. Komórka inicjująca

Komórką inicjującą można nazwać komórkę zawierającą w sobie skompresowany ekstrakt struktury, której jest zarodkiem, oraz opis nanotechnologicznego rozwoju struktury. Zatem komórką inicjującą programu będzie pojedynczy proces zawierający wszystkie składniki przyszłej aplikacji oraz posiadający informacje o sposobie jego rozkompresowania, a więc schemat rozwoju od pojedynczej komórki aż do końcowej postaci aplikacji.

Jak wiadomo, najlepiej kompresowane są fragmenty informacji bardzo podobne do siebie lub wręcz identyczne. Tak więc już teraz można ocenić, że budowa komórki inicjującej, a więc budowa programu metodą nanotechnologiczną, będzie miała największy sens w takich przypadkach, gdy aplikacja, mająca powstać z tej komórki, będzie wykonywała wiele procesów podobnych do siebie w budowie.

2.2. Rozwój programu

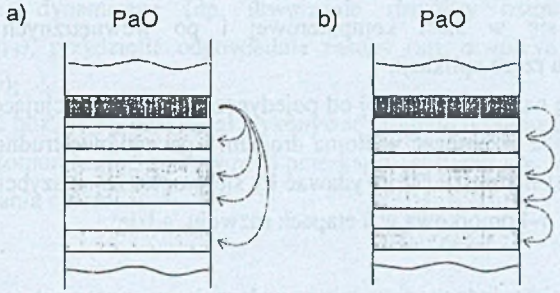
Aby program mógł się rozwinąć z pojedynczej komórki inicjującej do wielokomórkowej aplikacji, musi istnieć **schemat rozwoju** będący analogią kodu genetycznego. Ów schemat rozwoju musi opisywać, jak komórka się rozwija, replikuje i transformuje do postaci ostatecznej. Zjawisko samoreplikacji oprogramowania w środowisku pamięci operacyjnej komputerów jest możliwe i może być realizowane przez języki programowania na różnych poziomach abstrakcji [4]. Uogólniając, wprowadźmy pojęcie **operacji elementarnej** pochodzące z teorii systemów rozwijających się [5][6]:

L – (ang. *local replication*) – samoreplikacja komórki w obrębie tego samego komputera;

R – (ang. *remote replication*) – samoreplikacja, podczas której kopia komórki zostaje utworzona w innym komputerze;

S – (ang. *stagnation*) – stagnacja, jeśli chodzi o replikację; jednocześnie oznacza to przejście do trybu przetwarzania informacji na rzecz aplikacji.

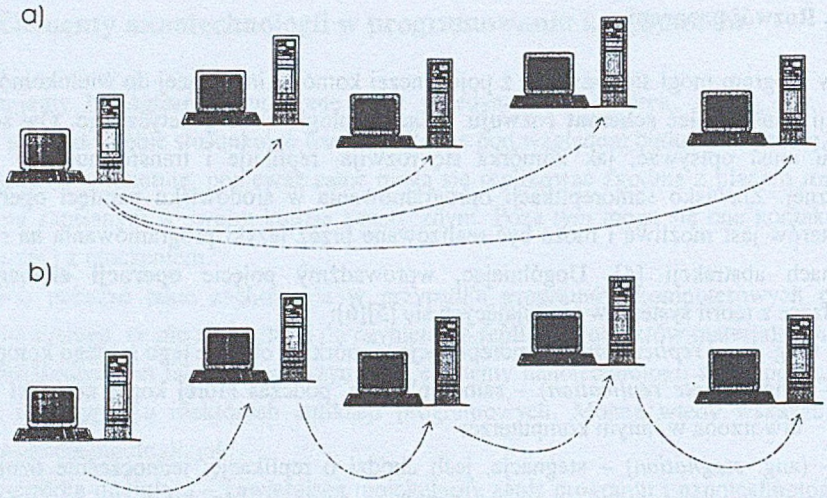
Samoreplikacja w obrębie jednego komputera może być realizowana dwojako, co pokazano obrazowo na rys. 1.



Rys. 1. Dwa sposoby replikacji komórek w pamięci operacyjnej
Fig. 1. Two methods of cell replication in the operating memory

Jak widać na rys.1a), źródłem wszystkich komórek może być pojedyncza komórka lub też każda nowo powstała komórka może być twórcą kolejnej (rys.1b). Równie dobrze można zastosować sposób mieszany wykorzystujący oba rozwiązania. Jest to zależne w dużej mierze od tego, w jakim języku programowania realizowana będzie taka aplikacja, ponieważ każdy język niesie z sobą trochę inne mechanizmy pozwalające na wykonanie replikacji.

Podobnie jest z elementarną operacją R, podczas której kopia komórki źródłowej zostaje utworzona i uaktywniona na innym komputerze. Również tutaj możemy wyodrębnić dwa sposoby dystrybucji komórek, co przedstawiono na rys.2.



Rys. 2. Dwa sposoby replikacji komórek pomiędzy komputerami
Fig. 2. Two methods of cell replication between computers

Rozpoczęcie przez komórkę wykonywania operacji S oznacza dla niej zakończenie fazy rozprzestrzeniania się w sieci komputerowej i po wewnętrznych przekształceniach rozpoczęcie pracy na rzecz aplikacji.

Zwróćmy uwagę na fakt, że rozwój od pojedynczej komórki inicjującej aż do ostatecznej postaci aplikacji może przebiegać wieloma drogami i początkowo trudno rozstrzygnąć, jaki sposób replikacji jest najlepszy [7]. Wydawać by się mogło, że najszybciej można otrzymać kompletną aplikację n–komórkową w k etapach rozwoju, gdzie:

$$k = \lceil \log_2 n \rceil$$

Twierdzenie to jest jednak prawdziwe wyłącznie w systemach rozwijających się o synchronicznym trybie rozwoju [5][6]. Gdy przyjrzymy się naturze, zauważymy, że w systemach biologicznych nie wszystkie operacje lementarne wykonują się w tym samym czasie. Najpierw z komórki inicjującej rozwija się **embrion** zbudowany z komórek, z których każda jest elementem początkowym określonej specjalizowanej części składowej [8]. W ten sposób uzyskujemy takie zrównoleglenie procesu replikacji, które prowadzi do najszybszego uzyskania końcowej postaci systemu wielokomórkowego.

W przypadku aplikacji rozproszonej każdy projekt należy rozpatrywać oddzielnie, gdyż należy oszacować, które części składowe aplikacji będą się tworzyły najwolniej. Ogólnie jednak można zauważyć, że w przypadku operacji replikacji o wiele bardziej złożone jest wykonanie replikacji komórki na inny komputer niż w obrębie komputera rodzimego komórki. Tak więc schemat rozwoju należy budować tak, aby najpierw wykonywały się

operacje R - generując pojedyncze komórki na każdym z komputerów, a następnie operacje L.

Ponadto należy zwrócić uwagę na to, iż każda replikująca komórka może posiadać (czy posiada - to zależy od programisty) kompletny zapis rozwoju programu analogicznie do kodu genetycznego. Oznacza to, że w razie zaistnienia potrzeby odbudowania zniszczonej aplikacji wystarczy do tego pojedyncza komórka.

2.3. Specjalizacja elementów składowych

Komórka, która nie podlega dalszej replikacji, najczęściej nie jest gotowa do pracy zaraz po powstaniu. Tak jak można zauważyć w nanosystemach biologicznych, w takiej komórce następują procesy wewnętrzne przeobrażające komórkę w pełni funkcjonalny element całego systemu. Podobną sytuację można zauważyć podczas powstawania aplikacji metodą nanotechnologiczną. Komórka takiej aplikacji, aby zacząć w pełni funkcjonować, może wykonywać następujące operacje:

- zainicjować potrzebne struktury danych: statyczne (np. zainicjowanie tablic stałą wartością) i dynamiczne (np. utworzenie struktury listowej z danymi do przetwarzania), przydzielić odpowiednie zasoby (np. otworzyć pliki, przechwycić porty we/wy);
- przeładować plik, aby proces mógł wykonywać program od samego początku;
- nawiązać komunikację z innymi komórkami systemu w celu prawidłowego funkcjonowania całości.

3. Projektowanie nanotechniczne

Projektowanie aplikacji z zastosowaniem elementów nanotechnologii sprowadza się praktycznie do utworzenia komórki inicjującej, a więc przygotowania ekstraktu przyszłego programu oraz zaprojektowania schematu rozwoju. Prześledźmy metodę projektowania aplikacji metodą nanotechnologiczną na konkretnym przykładzie.

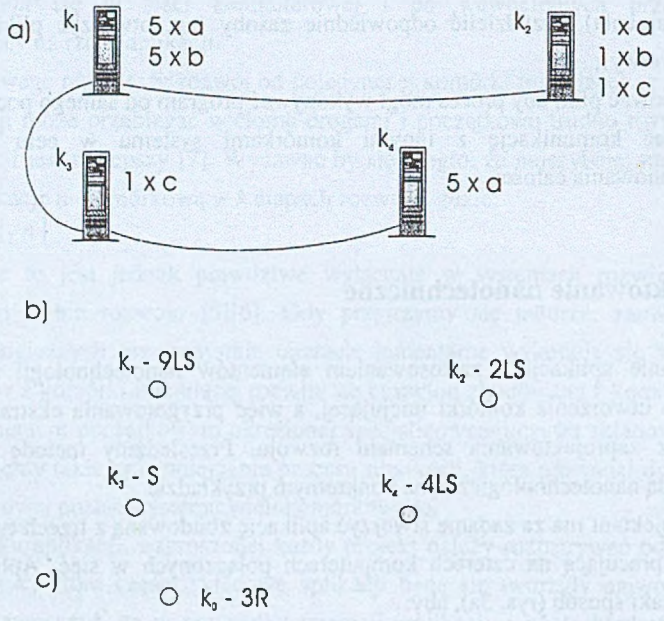
Pewien projektant ma za zadanie stworzyć aplikację zbudowaną z trzech typów procesów (lub wątków) pracującą na czterech komputerach połączonych w sieć. Aplikacja ma być zbudowana w taki sposób (rys. 3a), aby:

- na pierwszym komputerze pracowało 5 procesów typu *a*, 5 – typu *b* oraz jeden typu *c*,
- na drugim komputerze pracowało po jednym procesie każdego rodzaju,
- na trzecim – pojedynczy proces typu *c*,
- na czwartym – pięć procesów typu *a*.

Informacja o liczbie procesów, które mają pracować na każdym z komputerów, mówi nam, ile razy na każdym z nich musi dojść do replikacji procesu (rys. 3b), aby otrzymać określoną liczbę funkcjonalnych komórek systemu. Zwróćmy uwagę na fakt, iż replikację w ramach pamięci operacyjnej pojedynczego komputera wykonuje się sekwencyjnie – a więc niekoniecznie najszybciej. Jednakże, aby dokonać zrównoleglenia tej operacji, należałoby wprowadzić dodatkową operację elementarną.

Tak więc źródłem wszystkich komórek na danym komputerze będzie pojedyncza komórka wykonująca operację zgodną ze schematem *mLS*, gdzie *m* określa liczbę replikacji. Aplikacja ma pracować na czterech komputerach, a więc podczas rozwoju musi istnieć stan przejściowy aplikacji – embrion, zbudowany z czterech komórek, od którego to stanu operacje wykonywane są równolegle. Drogę rozwoju od komórki inicjującej do embriona opisuje więc sekwencja trzech realizacji operacji elementarnej R.

Ostatecznie otrzymujemy schemat rozwoju opisany następującą sekwencją operacji elementarnych: 3R 9LS 2LS S 4LS. Schemat ten będzie przenoszony do każdej nowej komórki wraz z operacją replikacji.



Rys. 3. Etapy projektowania aplikacji
Fig. 3. Phases of application project

Opisana w tym rozdziale droga znajdowania schematu rozwoju nie jest jedyna, gdyż jak się można łatwo domyśleć, może istnieć więcej niż jeden schemat rozwoju prowadzący do powstania określonej wieloprocesowej aplikacji [7]. Wymagałoby to jednak dokładniejszego sprecyzowania działań wykonywanych przez operacje elementarne.

Pozostaje jeszcze sporządzenie ekstraktu przyszłej aplikacji i umieszczenie go w komórce inicjującej. Omawiana aplikacja zbudowana jest z trzech typów procesów: *a*, *b* i *c*. Wystarczy więc, aby komórka inicjująca zawierała kod każdego z tych typów. Dodatkowo, o ile przyniesie to zysk, kod ten może być skompresowany. Wtedy jednak do każdej nowo tworzonej komórki należy dołączać procedurę rozkompresowującą, aby komórka w momencie wykonywania operacji S mogła wykorzystać sekwencję kodu odpowiednią dla danego typu komórki.

4. Uwagi końcowe

Podsumowując, możemy stwierdzić, że elementy charakterystyczne dla nanotechnologii mogą znaleźć zastosowanie w programowaniu komputerów, które nie są wykonane w skali nano. Aplikację zbudowaną zgodnie z regułami wywodzącymi się z nanotechnologii możemy nazwać systemem zbudowanym z *m* typów komórek realizujących swe elementarne zadania w zbiorze $K = \{K_1, K_2, ..., K_n\}$ komputerów, przy czym warunkiem początkowym powstania takiego systemu jest istnienie odpowiednio zaprojektowanej komórki inicjującej.

Należy zadać pytanie: czy program zaprojektowany metodą nanotechniczną można nazwać nanoprogramem? Pomimo cech charakterystycznych dla nanosystemów, stworzona za pomocą nanotechnologii aplikacja rozproszona nie pracuje w środowisku o wymiarach w skali nano. Gdy mowa o przetwarzaniu informacji, za podstawową jednostkę uważany jest bit. Przez analogię do rozmiarów nanoelementów, bardzo małą jednostkę informacji można nazwać nanoinformacją. Trudno jednak jest sporządzić aplikację, która korzysta z protokołów sieciowych i jednocześnie operuje na pojedynczych bitach.

Czym więc jest omawiana aplikacja? Jest ona tylko dowodem na to, iż metody stosowane w materialnej skali nano można również z powodzeniem stosować w wirtualnej skali makro.

LITERATURA

1. Czempel Cz.: Nanonauka – nanotechnologia, źródła i perspektywy. Nauka, nr 3, 1999.
2. Regis E.: Nanotechnologia – narodziny nowej nauki, czyli świat cząsteczka po cząsteczce. Prószyński i S-ka, Warszawa 2001.

3. Nowak S.: Nanotechnology and nanocomputers. Archiwum Informatyki Teoretycznej i Stosowanej, tom 13 – zeszyt 2, s.195-204, Katowice 2001.
4. Dobosz K.: A practical approach to self-replication of programs. Archiwum Informatyki Teoretycznej i Stosowanej, w druku.
5. Węgrzyn S., Gille J.-C., Vidal P.: Developmental Systems. Springer-Verlag 1990.
6. Węgrzyn S., Gille J.-C., Vidal P.: Genetyka procesów rozwoju. Zeszyty Naukowe Politechniki Śląskiej nr19, Gliwice 1992.
7. Dobosz K.: Rozpoznawanie słowa genetycznego systemu rozwijającego się. Studia Informatica, vol.22, no 4 (46), Gliwice 2001, str. 27 - 43.
8. Węgrzyn S.: Embryonic organisation of developmental processes and algorithms structures. Gliwice 1999.

Recenzent: Prof. dr inż. Stefan Węgrzyn

Wpłynęło do Redakcji 4 kwietnia 2002 r.

Abstract

The paper shows nanotechnological elements, which are characteristic for molecular nanosystems: initial cell, development of nanosystem, multilevel development of a single cell (chapter 2). Those can be used in the process of development and computer program creating. An initial cell consists of compressed extract of the program, and development description (section 2.1). The development from the initial cell to final multicellural system is provided through elementary operation: self-replication. In the section 2.2 self-replication in the single computer operating memory (fig. 1) and self-replication in the computer net (fig. 2) are shown. The last described nanotechnological element is a specialization of a single cell (section 2.3). Sample internal transformations are pointed out. Chapter 3 indicates how to use nanotechnological elements in distributed application design. Fig. 3 illustrates several design phases. And at the end, chapter 4 contains conclusions.