dr inż. Jerzy Zając
mgr inż. Witold Czmich
Katedra Systemów Wytwarzania
Politechnika Krakowska

DCOM jako narzędzie komunikacji w sterowaniu dyskretnymi systemami wytwarzania

Streszczenie: W referacie omówiono zasadnicze elementy technologii obiektów rozproszonych DCOM. Przedstawiono nową koncepcję polegającą na zastosowaniu technologii DCOM do sterowania dyskretnymi systemami wytwarzania w oparciu o jednolite, konfigurowalne, inteligentne i kooperatywne moduły programowe.

DCOM as a Communication Tool in Discrete Event Manufacturing Processes

Summary: The paper presents fundamental elements of Distributed Component Object Model (DCOM) technology. A new concept of DCOM technology application to control of discrete event manufacturing processes is described. It is based on cooperation of unified, easy to re-configure and intelligent software modules.

1. WSTĘP

Dynamiczny rozwój współczesnych systemów produkcyjnych związany jest z wprowadzaniem nowych technologii wytwarzania, wykorzystywaniem materiałów o nowych właściwościach, stosowaniem efektywnych rozwiązań organizacyjnych oraz implementowaniem najnowszych technologii informatycznych. Nowoczesne rozwiązania informatyczne to w zakresie sprzętowym m.in. wysokowydajne układy mikroprocesorowe oraz szybkie sieci komputerowe, a w zakresie programowym nowoczesne technologie obiektowe, w tym także technologie obiektów rozproszonych. Wymagania stawiane współczesnym systemom produkcyjnym, takie jak: otwartość, elastyczność czy inteligencja są możliwe do osiągnięcia jedynie poprzez zastosowanie nowoczesnych rozwiązań modelowych pozwalających na wykorzystanie zalet współczesnych technologii informatycznych. Środowisko informatyczne współczesnych systemów wytwarzania staje się w coraz większym stopniu środowiskiem rozproszonym. Wynika to zarówno z rozproszenia źródeł informacji, jak i szerokiego stosowania technologii klient/serwer. W pracy [4] zaproponowano koncepcję sterowania dyskretnymi systemami wytwarzania opartą na integracji inteligentnych obiektów w środowisku rozproszonym. Zaproponowana koncepcja wychodzi naprzeciw najnowszym tendencjom rozwojowym współczesnych otwartych systemów sterowania bazujących na komputerach PC. Wzrost mocy obliczeniowej najnowszej generacji mikroprocesorów umożliwia bowiem przeniesienie części zadań realizowanych dotychczas przez sprzęt do oprogramowania. Wyposażenie nowej generacji urządzeń technologicznych takich jak obrabiarki czy roboty w komputerowe systemy sterowania oparte na sprzęcie klasy PC, umożliwi szerokie wprowadzanie inteligentnych

samoorganizujących się systemów wytwarzania. Realizacja systemu sterowania bazującego na koncepcji rozproszonej inteligencji wymaga zastosowania efektywnych sieciowych narzędzi integracyjnych. W niniejszym referacie omówiono, z konieczności w sposób bardzo pobieżny, podstawowe elementy technologii DCOM [2] oraz przedstawiono rozwiązanie pozwalające na zastosowanie DCOM'a do sterowania dyskretnymi systemami wytwarzania.

2. DCOM - NARZĘDZIE INTEGRACJI OBIEKTÓW W ŚRODOWISKU ROZPROSZONYM

Idea budowy złożonych aplikacji w oparciu o integrację samodzielnych, wyspecjalizowanych komponentów (obiektów) była intensywnie rozwijana od końca lat 80-tych bieżącego stulecia. Efektem tych działań było pojawienie się dwóch konkurencyjnych technologii integracji obiektów w środowisku rozproszonym tj. CORBA i DCOM. Umożliwiają one współpracę między obiektami znajdującymi się w różnych aplikacjach, działającymi na komputerach o różnej architekturze oraz wykorzystującymi różne systemy operacyjne. W roku 1991 została opublikowana specyfikacja CORBA (Common Object Request Broker Architecture) opracowana przez organizację OMG (Object Management Group)[7] skupiającą obecnie ponad siedemset przedsiębiorstw, głównie z branży komputerowej. Aplikacje zgodne ze specyfikacją CORBA dostępne są obecnie dla większości platform sprzętowych i systemów operacyjnych. Technologia CORBA jest technologią dojrzałą, skierowaną na zastosowania przemysłowe i wspieraną przez szerokie grono potentatów przemysłu komputerowego W roku 1996, wraz z wersją 4.0 systemu operacyjnego Windows NT, pojawiła się technologia DCOM (Distributed Component Object Model). Technologia ta opracowana przez Microsoft stanowi rozszerzenie znanego od 1993 roku modelu COM (Component Object Model). DCOM dostępny jest obecnie na platformy Windows 95, Windows NT (Intel i Alpha) oraz Solaris. W ciągu najbliższych miesięcy mają pojawić się wersje DCOM na inne platformy sprzętowe m.in. HP, IBM, Digital oraz inne systemy operacyjne np. Linux lub OpenVMS. Omówienie i porównanie cech obydwu konkurencyjnych technologii znaleźć można w pracach [3][9].

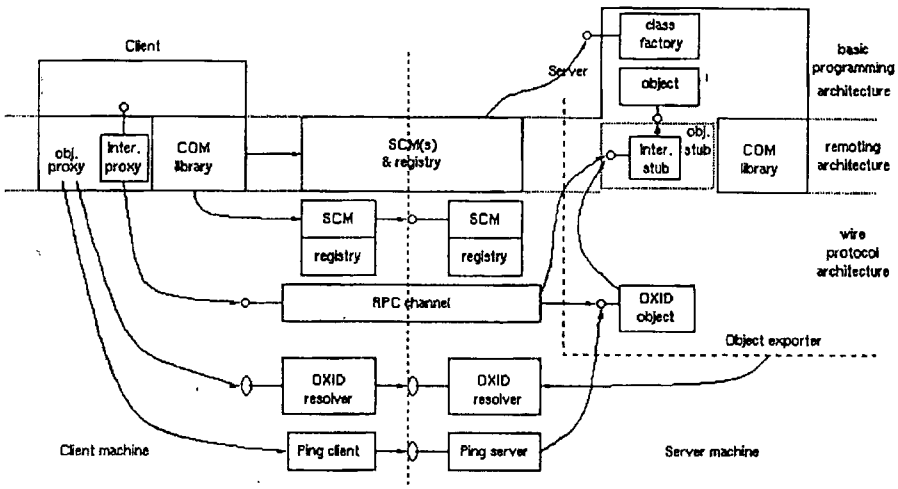
Dla praktycznej weryfikacji koncepcji rozproszonego systemu sterowania dyskretnymi systemami wytwarzania, jako narzędzie integracyjne wybrano technologię DCOM. Jest to spowodowane łatwą dostępnością oprogramowania (dostarczane wraz z systemem operacyjnym lub możliwe do skopiowania z Internetu), wsparciem technicznym oferowanym przez Microsoft (w tym szeroki dostęp do dokumentacji, list dyskusyjnych itp.), stabilnością standardu tworzonego przez jedną firmę, oraz prostym dostępem do nowych wersji oprogramowania. Nie bez znaczenia był również fakt rosnącego znaczenia systemów informacyjnych w przedsiębiorstwach czyli tzw. korporacyjnych Intranetów. Według ocen wyspecjalizowanych firm analitycznych system Windows NT stanie się na przełomie wieków wiodącym systemem operacyjnym także na rynku korporacyjnych Intranetów. Należy się więc spodziewać, że technologie rozwijane wraz z tym systemem wykorzystywane będą szeroko w przedsiębiorstwach, w tym również do zarządzania i sterowania produkcją.

Technologia DCOM jest technologią klient/serwer. Dla zrozumienia zasady działania tej technologii niezbędna jest znajomość kilku podstawowych pojęć:

interfejs - identyfikowana przez nazwę struktura danych zawierająca grupę prototypów procedur (metod) o wspólnej funkcjonalności,
klasa obiektu - pełna implementacja jednego lub więcej interfejsów,
obiekt - konkretna postać (*ang. instance*) określonej klasy obiektu,
serwer - proces odpowiedzialny za tworzenie i działanie obiektów,
klient - proces, który wywołuje metodę zawartą w obiekcie.

W ramach technologii DCOM serwery implementowane są wewnątrz plików wykonawczych

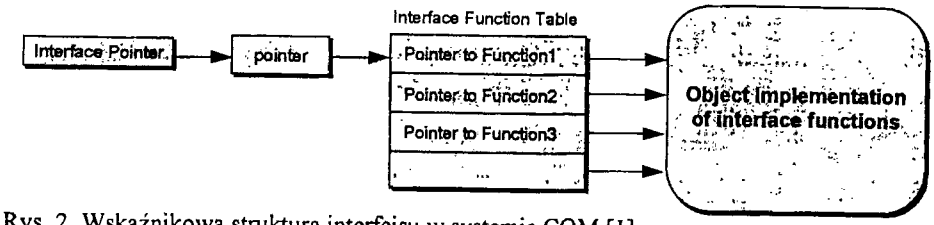
(EXE) lub bibliotek dynamicznych (DLL). Realizacja procesu klient/serwer bazuje na zdalnym wywoływaniu procedur (RPC). Aby wywołać określoną metodę niezbędne jest jednoznaczne zidentyfikowanie klasy obiektu oferującego tę metodę oraz interfejsu, przy pomocy którego będziemy mogli uzyskać dostęp do oferowanych serwisów. Jednoznaczność identyfikacji poszczególnych klas obiektów oraz ich interfejsów uzyskuje się poprzez przyporządkowanie im unikalnych 128-bitowych identyfikatorów GUID (*ang. globally unique identifier*). Dla klasy obiektu identyfikator ten nosi nazwę CLSID (*ang. class identifier*) a dla interfejsu IID *(ang. interface identifier*). Wszystkie klasy obiektów oraz wszystkie interfejsy przed wykorzystaniem muszą zostać zarejestrowane w systemie operacyjnym. Dopiero po rejestracji są „widoczne" dla klientów i mogą zostać użyte. Każdy interfejs w dowolnym momencie może zostać wzbogacony o nowe składowe bez konieczności modyfikowania współpracujących z nim klientów. Gdy programista chce poszerzyć właściwości już istniejącej klasy obiektu, może np. dodać do niego nowy zestaw metod. O ich dostępności klienci mogą dowiedzieć się za pośrednictwem funkcji QueryInterface. Dodanie nowego interfejsu (lub rozszerzenie istniejącego) nie wpływa na współpracę klientów z obiektem.



Rys.1. Architektura systemu DCOM [3].

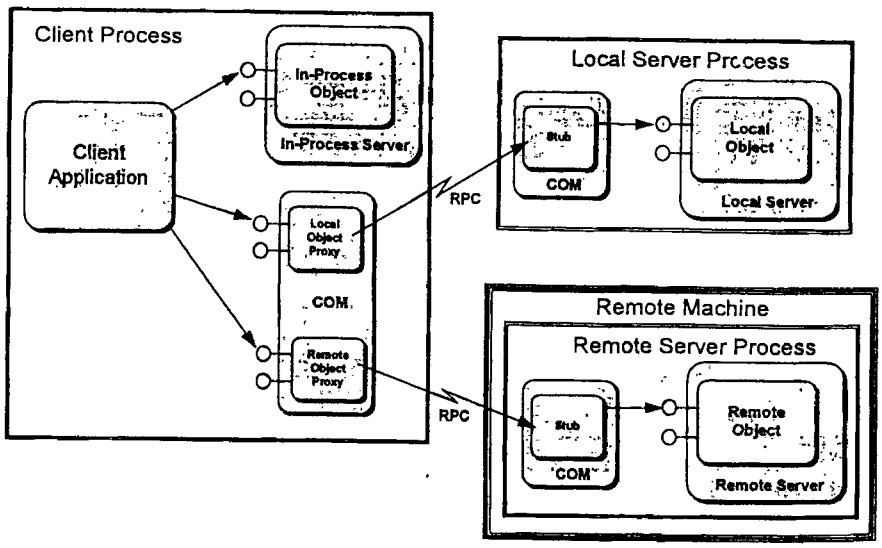
Rys. 1 przedstawia ogólny schemat prezentujący trójpoziomową architekturę systemu DCOM oraz elementy składowe biorące udział w realizacji procesu klient/serwer. Klient określa CLSID klasy obiektu, IID interfejsu oraz wprowadza do rejestru systemowego nazwę komputera, na którym dany obiekt będzie uruchamiany. Serwer natomiast tworzy obiekt i zwraca klientowi wskaźnik jego interfejsu IID. Otrzymany wskaźnik wykorzystywany jest przez klienta do wywoływania metod obiektu. Technologia DCOM umożliwia trzy warianty realizacji procesu klient/serwer: (1) klient i serwer znajdują się w tej samej przestrzeni adresowej, (2) klient i serwer znajdują się na tym samym komputerze ale w różnej przestrzeni adresowej, (3) klient i serwer znajdują się na różnych komputerach. Dla programisty istotnym ułatwieniem jest fakt, że każda z trzech ww. możliwości realizowana jest przy pomocy tego samego kodu. Różnice w położeniu obiektów są więc ukryte dla programisty, a identyfikacją oraz sposobem realizacji konkretnego wariantu zajmuje się SCM (*ang. service control manager*) wykorzystując do tego celu zawartość rejestru systemowego (*ang. registry*).
W przypadku pierwszym klient i serwer znajdują się w tej samej przestrzeni adresowej (patrz

rys.3 - In-Process Server), więc wywołanie metody następuje bezpośrednio przy pomocy otrzymanego wskaźnika interfejsu (*ang. interface pointer*). Uzyskany wskaźnik interfejsu, tak jak to przedstawiono na rys. 2, jest wskaźnikiem do wskaźnika do tablicy wskaźników wskazujących na funkcje (metody) oferowane przez obiekt serwera. W przypadku pierwszym w realizacji procesu klient/serwer nie biorą udziału inne elementy systemu COM.



Rys. 2. Wskaźnikowa struktura interfejsu w systemie COM [1].

W sytuacji gdy klient i serwer znajdują się na tym samym komputerze ale w różnej przestrzeni adresowej (przypadek drugi), to w operacji przekazywania klientowi wskaźnika do interfejsu o żądanym IID udział biorą dodatkowe obiekty COM połączone za pomocą kanału RPC (patrz rys.3 - Local Server). Są to: pośrednik obiektowy (*ang. proxy*) po stronie klienta oraz szkielet obiektu (*ang. stub*) po stronie serwera. Kiedy klient wywołuje metodę, parametry wywołania są pakowane (*ang. marshaling*) przez pośrednika i odpakowywane (*ang. unmarshaling*) przez szkielet obiektu. Uzyskane w wyniku realizacji metody rezultaty są pakowane tym razem przez szkielet obiektu i odpakowywane przez pośrednika, który także zwraca je klientowi.



Rys. 3. Warianty technologii klient/serwer w ramach DCOM [8].

W przypadku trzecim, tzn. gdy klient i serwer znajdują się na różnych komputerach, procedura uzyskiwania wskaźnika interfejsu i wywoływania metody jest z punktu widzenia użytkownika realizowana podobnie jak w przypadku drugim, przy czym należy uwzględnić fakt, że wymienione wyżej operacje realizowane są przy pomocy złożonych mechanizmów sieciowych.

Jedną z najistotniejszych cech technologii DCOM jest to, że specyfikacja interfejsów obiektów ma charakter binarny. Interfejsy definiowane przy pomocy języka IDL (*ang. Interface Definition Language*) są strukturą danych zawierających listę prototypów funkcji (metod) udostępnianych przez obiekt serwera. Przyjęcie jednoznacznej - binarnej definicji interfejsów pozwala na przygotowywanie obiektów przy pomocy różnych narzędzi programistycznych.
Zastosowanie technologii DCOM do integracji komponentów (obiektów) w środowisku rozproszonym nie rozwiązuje wszystkich problemów komunikacyjnych występujących w środowisku systemów sterowania. Do realizacji komunikacji komputerów PC ze sterownikami PLC, NC czy RC wykorzystuje się najczęściej protokoły DDE, NetDDE lub ich następcę OPC.

3. OPC - INTEGRACJA OBIEKTÓW Z UKŁADAMI RZECZYWISTYMI

OLE for Process Control (OPC) opiera się o te same mechanizmy niskiego poziomu co DCOM będąc jednocześnie protokołem zoptymalizowanym pod kątem zadań, jakie ma spełniać w systemach czasu rzeczywistego. O ile DCOM umożliwia wymianę złożonych i dowolnie konfigurowalnych struktur danych pomiędzy obiektami, o tyle OPC powstał aby zaspokoić potrzebę szybkiej transmisji danych o prostej strukturze. Należy tutaj wyraźnie podkreślić, iż wydajność OPC jest wyższa niż DCOM, co wynika z różnych wymagań jakie stawiane są obu protokołom oraz większej elastyczności i uniwersalności DCOM'a. Cechą, która wyraźnie wyróżnia technologię DCOM, są wbudowane mechanizmy zdalnego uruchamiania procedur, co znacznie poszerza jego funkcjonalność wykraczając poza funkcjonalność tradycyjnie rozumianego protokołu komunikacyjnego. Oznacza to m.in. możliwość pełnego wykorzystania mocy obliczeniowej komputerów połączonych w sieć, co może mieć znaczenie przy wykorzystaniu złożonych algorytmów predykcyjnych i optymalizacyjnych. Z tych względów technologii OPC nie należy traktować jako alternatywy dla DCOM'a, lecz jako następcę popularnego w wielu systemach protokołu DDE oraz jego odpowiednika NetDDE w środowisku rozproszonym. Ponieważ DDE powstał w czasie, kiedy sieci komputerowe nie miały tak dużego udziału w sterowaniu oraz zarządzaniu, siłą rzeczy nie wytrzymał próby czasu. Opracowanie wydajnego protokołu komunikacyjnego zorientowanego na maksymalne wykorzystanie zasobów sieciowych stało się nieodzowne. Oprócz znaczącego wzrostu wydajności, twórcy OPC opracowali strukturę transmisji danych określaną jako VTQ (*ang. Value, Time and Quality)*. Wraz z każdą przesyłaną wartością wędruje także informacja o czasie, w którym została zebrana (*ang. Time Stamping)* oraz jej jakości czy - inaczej nazywając - wiarygodności. Cecha ta pozwala na każdym etapie analizy danych stwierdzić czy nie zostały one zebrane np. podczas awarii urządzenia pomiarowego, uszkodzenia kanału transmisyjnego lub innej sytuacji zakłócającej przekazywaną wartość.
Wspólny korzeń, z którego wywodzą się technologie DCOM oraz OPC pozwala na łatwiejszą i stabilniejszą integrację systemu sterowania opartego na DCOM'ie z programami komunikacyjnymi opartymi na OPC, co nie wyklucza oczywiście zastosowania dużej liczby istniejących programów komunikacyjnych opartych np. na NetDDE.

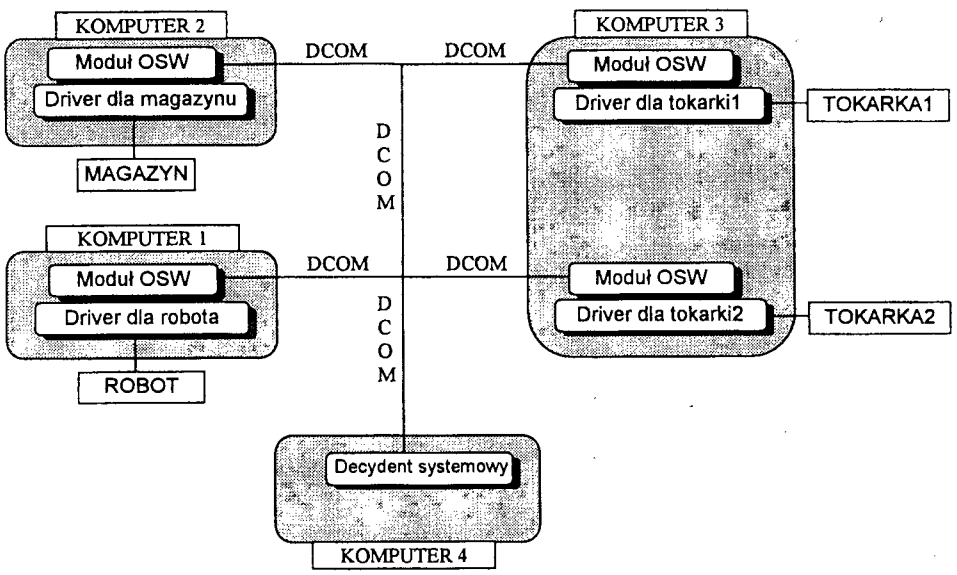
4. ZASTOSOWANIE DCOM DO INTEGRACJI OBIEKTÓW W DYSKRETNYCH SYSTEMACH WYTWARZANIA

Wymiana informacji w ramach współczesnych systemów wytwarzania odbywa się głównie na zasadzie wymiany komunikatów. Zaawansowane systemy przemysłowe wykorzystują najczęściej technikę przesyłania komunikatów bazującą na specyfikacji MMS (*ang. Manufacturing Message Specyfication*) zaprojektowaną do sterowania i monitorowania

urządzeń przemysłowych. Mimo technicznego zaawansowania MMS nie osiągnął pełnego sukcesu [6]. Jako główne słabości przyjmuje się dużą złożoność, duży koszt i niewystarczającą wydajność. Systemy wytwarzania nadchodzącego stulecia staną przed nowymi wyzwaniami rynku, które wymuszą stosowanie większej liczby otwartych i dostosowujących się do aktualnych wymagań systemów sterowania. Aby sprostać tym wymaganiom, tworzy się koncepcje budowy systemów sterowania w oparciu o integrację samodzielnych inteligentnych obiektów. W pracy [4] przedstawiono koncepcję rozproszonego systemu sterowania dyskretnymi systemami wytwarzania bazującą na zastosowaniu jednolitych, konfigurowalnych, inteligentnych i kooperatywnych modułów (obiektów). Zaproponowano trzy typy konfigurowalnych modułów programowych wyposażonych w interfejsy pozwalające na budowę dowolnej struktury układu sterowania w środowisku sieciowym. Tymi modułami są:

- moduł zarządzający zwany dalej decydentem systemowym,
- moduł reprezentujący działanie obiektu elementarnego zwany dalej obiektem systemu wytwarzania (OSW),
- moduł dostosowujący zwany dalej driverem a służący do integracji OSW z układem rzeczywistym.

Jako protokół komunikacyjny przyjęto DCOM. Każdy z ww. modułów programowych pełni rolę zarówno klienta jak i serwera DCOM. Przyjęto koncepcję integracji obiektów poprzez wymianę komunikatów, stąd poszczególne moduły programowe zostały wyposażone w interfejsy umożliwiające wysyłanie i odbieranie komunikatów. Dzięki zastosowaniu technologii DCOM możliwe jest m in. zdalne uruchamianie obiektów, statyczne i dynamiczne ich konfigurowanie, odłączanie i dołączanie obiektów bez konieczności wyłączania systemu.



Rys. 4. Przykład konfiguracji systemu sterowania rozproszonego.

Zastosowanie koncepcji jednolitych, konfigurowalnych obiektów spowoduje poprawienie niezawodności całości systemu sterowania (moduł programowy OSW będzie bowiem wielokrotnie testowany, często w całkowicie odmiennych sytuacjach) oraz zdecydowanie obniży koszty przygotowania systemu. Jedynym modułem, który wymaga indywidualnej

konfiguracji jest driver czyli moduł programowy odpowiedzialny za komunikację z układami rzeczywistymi. W wersji bazowej jest on wyposażony jedynie w narzędzia komunikacyjne z obiektem OSW. Musi więc być „wypełniony indywidualną treścią" dla każdego obiektu. Nie są stawiane żadne ograniczenia co do protokołu komunikacyjnego łączącego driver z układem rzeczywistym np. sterownikiem PLC, NC czy RC. Na rysunku 4 został przedstawiony przykład konfiguracji systemu sterowania zrobotyzowanego gniazda produkcyjnego, oparty na integracji sieciowej inteligentnych, konfigurowalnych modułów programowych zintegrowanych technologią DCOM.

5. WNIOSKI

Technologia DCOM dostępna jest dopiero od połowy 1996 roku, jednak należy stwierdzić, że jest już technologią stabilną. Fakt, że jej twórcą jest jedna firma, stanowi w tym wypadku raczej zaletę niż wadę. Szczególnie można to zaobserwować na konkurencyjnym standardzie CORBA, dla którego powstało wiele nie zawsze w pełni kompatybilnych rozwiązań. Pewną nadzieję na upowszechnienie standardu DCOM jest przekazanie go przez firmę Microsoft niezależnemu konsorcjum Open Group oraz tworzenie tzw. bramek integrujących obie konkurencyjne technologie. Należy się również spodziewać, że wzrost zainteresowania technologią OPC umocni pozycję technologii DCOM lub Real-Time DCOM jako znaczącego elementu także w zakresie sterowania systemów.

LITERATURA

[1] Brockschmidt K.: What OLE Is Really About. Microsoft Corporation 1996. http://www.microsoft.com/oledev/olecom/aboutole.htm

[2] Brown N., Kindel C.: Distributed Component Object Model Protocol - DCOM/1.0. http://www.microsoft.com/oledev/olecom/draft-brown-v1-spec-01.txt

[3] Chung P. E. i inni: DCOM and CORBA Side by Side Step by Step and Layer by Layer. September 3 1997. http://www.research.att.com/~ymwang/vita/vita.htm#Publications

[4] Cyklis J., Zając J.: Koncepcja rozproszonego systemu sterowania dyskretnymi systemami wytwarzania. Materiały z Konferencji AUTOMATION'98.

[5] DCOM Technical Overview. White Paper. Microsoft Corporation 1996. http://www.microsoft.com/cominfo

[6] Guyonnet G., Gressier-Soudan E., Weis F.: COOL-MMS: a CORBA approach for ISO-MMS. http://cedric.cnam.fr/personne/gressier/ECOOP.html

[7] The Common Object Request Broker Architecture. Object Management Group html://www.omg.org/corba/corbiiop.htm

[8] The Component Object Model Specification. Microsoft Corporation and Digital Equipment Corporation Version 0.9 1995. http://www.microsoft.com/oledev/olecom/title.htm

[9] Thompson D., Watkins D.: Comparisons between CORBA and DCOM: Architectures for Distributed Computing. http://www.sd.monash.edu.au/research/publications/1997/ABSTRACTS.html#P97-1

[10] Wang Y., Damani O., Lee W.: Reliability and Availability Issues in Distributed Component Object Model (DCOM). Proceedings International Workshop on Community Networking (CN4), s. 59-63, Sept. 1997.