STUDIA INFORMATICA 2001
Volume 22 Number 3 (45)

Andrzej BAŃKOWSKI
PWPT WASKO Sp. z o.o., Zakład Systemów Telekomunikacyjnych

CORBA – UNIWERSALNE ŚRODOWISKO KOMUNIKACYJNE NOWOCZESNYCH SYSTEMÓW ZARZĄDZANIA

Streszczenie. Artykuł przedstawia ogólną charakterystykę dużych systemów zarządzania sieciowego ze względu na ich cechy komunikacyjne, specyficzne potrzeby i tendencje rozwojowe. Na tym tle zaprezentowano zalety rozproszonej architektury obiektowej CORBA, jej zastosowanie we współczesnym zarządzaniu oraz możliwości rozwiązania niektórych problemów związanych z zarządzaniem w środowiskach heterogenicznych.

CORBA – UNIVERSAL COMMUNICATION ENVIRONMENT FOR MODERN MANAGEMENT SYSTEMS

Summary. This article presents general characteristics of large scale network management systems, regarding their communication features, specific needs and trends. On this basis, it presents the advantage of distributed object architecture CORBA, its application in modern management and ability to solve some management problems in heterogenous environment.

1. Wstęp

Informacja – obiekt abstrakcyjny, który w postaci zakodowanej (tzw. danych) może być przechowywany, przesyłany, przetwarzany i użyty do sterowania. Informację można uważać za ślad jakiegoś faktu, zdarzenia lub wytworu wyobraźni, które zaistniały lub powinny zaistnieć. Może ona dotyczyć cech, stanu lub zachowania się obiektu zarówno fizycznego, jak i abstrakcyjnego. Informacją jest każdy czynnik, dzięki któremu odbierający go obiekt (człowiek, organizm żywy, organizacja, urządzenie automatyczne) może polepszyć swoją znajomość otoczenia i bardziej sprawnie przeprowadzić celowe działanie.

Informacja jest niezbędna do poprawnego funkcjonowania każdego podmiotu zdolnego do podejmowania decyzji, a jej znaczenie jest tym większe, im wyższy jest poziom inteligencji tego podmiotu. Ze względu więc na swoją wagę w społeczeństwie inteligentnym, informacja coraz częściej staje się również towarem o wymiernej wartości i określonej postaci fizycznej. Powstają i rozwijają się całe gałęzie nauki i przemysłu zajmujące się jej uzyskiwaniem, przetwarzaniem, przechowywaniem, przesyłaniem, udostępnianiem.

Ponieważ informacja bywa towarem „delikatnym" i kosztownym, a jej uszkodzenie i utrata są zazwyczaj nieodwracalne, coraz częściej zwraca się uwagę na problemy związane z zapewnieniem dostępności, poufności, a zwłaszcza niezawodności w obsłudze danych. Coraz częściej korzystne okazuje się zainwestowanie dodatkowych, często niemałych środków w ochronę i zabezpieczenia danych. Stąd obserwowany coraz większy nacisk na rozwój systemów zarządzania: zwiększanie ich niezawodności, elastyczności, rozszerzanie zasięgu przestrzennego i logicznego.

2. Model obiektowy

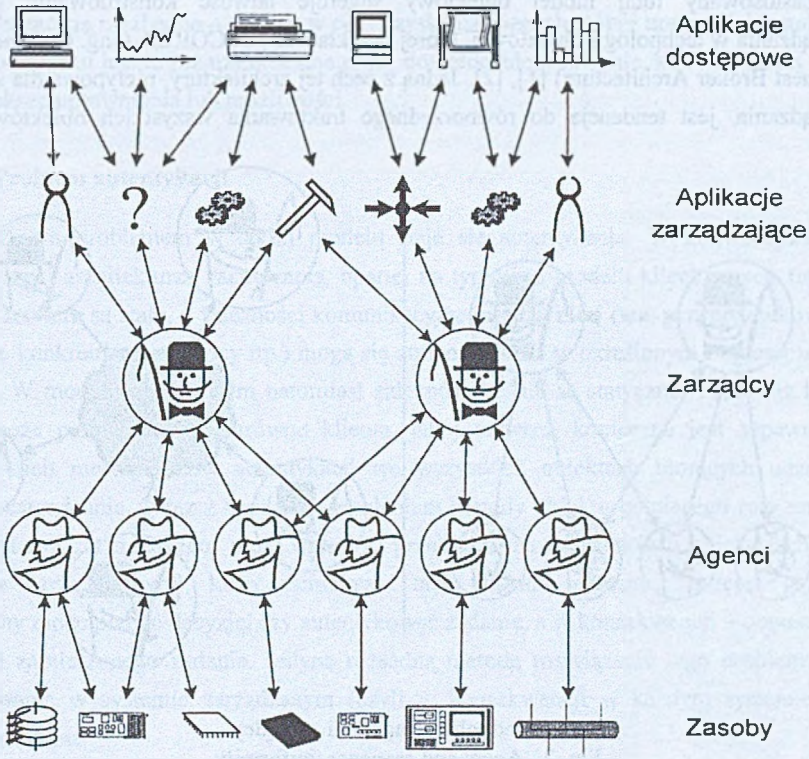
W ostatnich latach model obiektowy upowszechnił się jako technika o podejściu uniwersalnym i upraszczającym. Podejmowane są próby zaadaptowania go we wszystkich możliwych dziedzinach, ale najlepiej sprawdza się on w przypadku dużych i złożonych struktur.

2.1. Obiektowe cechy systemów zarządzania

W rozbudowanym systemie zarządzania można wyodrębnić niezależne jednostki logiczne, spełniające samodzielne funkcje. Charakter obiektowy można dostrzec na wszystkich poziomach złożoności systemu (rys. 1):

- rzeczywiste, elementarne obiekty zarządzane na najniższym poziomie złożoności, stanowiące cel istnienia systemów zarządzania; przenikają one do warstwy logicznej systemu, gdzie mają odzwierciedlenie w postaci obiektów abstrakcyjnych;
- moduły agentów zarządzania, stanowiące punkty styku sieci zarządzającej z siecią zarządzaną; każdemu agentowi może podlegać pewien zbiór obiektów zarządzanych, a pewne podzbiory obiektów mogą być obsługiwane przez więcej niż jednego agenta;
- serwery zarządzania utrzymujące oprogramowanie zarządcy i współpracujące z podległymi agentami; przyporządkowanie agentów do serwerów również może być nadmiarowe i niejednoznaczne;

– aplikacje odpowiedzialne za wykonywanie poszczególnych elementarnych funkcji zarządzania;
– aplikacje dostępowe użytkowników umożliwiające korzystanie z funkcjonalności systemu zarządzania.



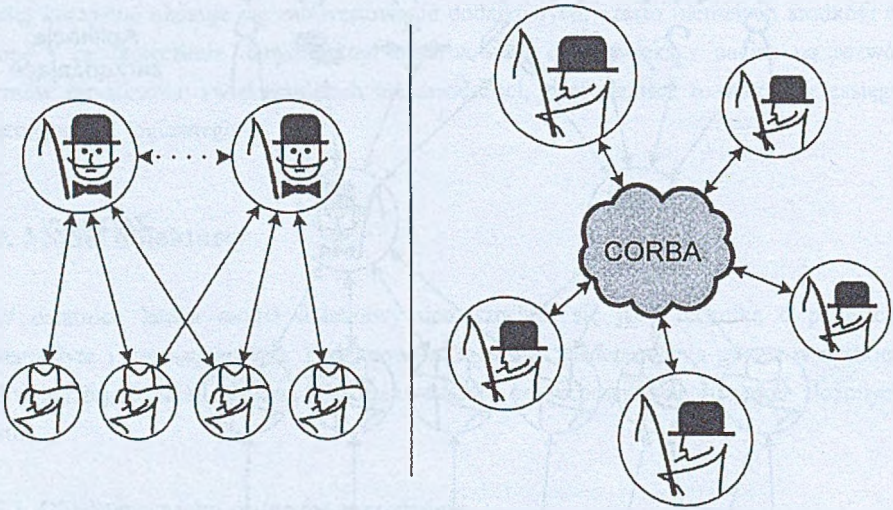
Rys. 1. Obiektowe cechy systemów zarządzania
Fig. 1. Object features of management systems

Dalsze upraszczanie modelu obiektowego prowadzi do ujednolicenia mechanizmów komunikacji pomiędzy poszczególnymi komponentami systemu. W skrajnym przypadku można wszystkie rozpatrywane elementy traktować jako całkowicie niezależne obiekty zanurzone we wspólnym medium komunikacyjnym. Główną zaletą tego podejścia jest uniwersalność mechanizmów dostępowych, zdejmująca z programisty problemy związane z organizowaniem połączeń między komponentami. Niestety, nie jest możliwe zastosowanie takiego uniwersalnego modelu na wszystkich poziomach złożoności, m.in. ze względu na

konieczność dopasowania się do istniejących i powszechnych standardów, a także trudności implementacyjne w warstwach o niższym poziomie inteligencji technicznej.

2.2. Ujednolicenie agenta i zarządcy

Zastosowany tutaj model obiektowy sugeruje łatwość konstruowania systemów zarządzania w technologii obiektowej, której przykładem jest CORBA (ang. Common Object Request Broker Architecture) [1], [2]. Jedną z cech tej architektury, nietypową dla systemów zarządzania, jest tendencja do równorzędnego traktowania wszystkich obiektów. Ma to



Rys. 2. Jednolitość agenta i zarządcy
Fig. 2. Agent and manager uniformity

poważne konsekwencje, gdy postać obiektowa zostanie zastosowana zarówno do elementów zarządzanych, jak i zarządzających. Pozytywnym tego skutkiem jest poszerzenie modelu funkcjonalnego: oprócz dotychczas realizowanej komunikacji na linii zarządca-agent lub zarządca-zarządca, możliwa jest także komunikacja pomiędzy agentami. Pełniejsze zastosowanie modelu architektury CORBA pozwala ujednolicić nie tylko funkcje komunikacyjne (przekazywania informacji), ale także funkcje zarządzania (przetwarzania informacji i podejmowania decyzji). Prowadzi to do utraty charakteru: agent, zarządca – i sprawia, że role systemów biorących udział w procesie zarządzania mogą zmieniać się dynamicznie. W skrajnym przypadku możliwe jest całkowite rozmycie funkcjonalności zarządzania, a określenia: „agent" i „zarządca" przestają trwale wiązać się z konkretnym

systemem, ale z aktualnie wykonywanymi czynnościami (rys. 2). Tak rozumiana obiektowość umożliwia uzyskanie pożądanej decentralizacji funkcjonalności i rozproszenia mocy obliczeniowej.

Ujednolicenie agenta i zarządcy prowadzi do utraty hierarchicznego charakteru systemu: wszystkie jego elementy stają się równorzędne. Pewne rozróżnienie między nimi może jednak okazać się niezbędne – choćby w celu uzyskania różnych stopni uprzywilejowania. W takim środowisku jest to już uzyskiwalne przez dowolne zdecydowanie, które elementy mają mieć większe uprawnienia lub możliwości.

2.3. Problem autentykacji

Poważnym problemem w takim modelu staje się autentykacja. W konwencjonalnej, hierarchicznej architekturze zarządzania, opartej na typowym modelu klient/serwer, funkcje klienta i serwera są stałe, a zależności komunikacyjne między nimi (tzn. przyporządkowanie agenta do konkretnego zarządcy itp.) mogą się zmieniać tylko w określonym i ograniczonym zakresie. W modelu obiektowym natomiast zależności te nie są statyczne. Ponieważ każdy obiekt może pełnić funkcje zarówno klienta, jak i serwera, konieczne jest zapewnienie odpowiednich mechanizmów autentykacji we wszystkich obiektach biorących udział w procesie zarządzania. Wraz z każdym wywołaniem metody obiektu pełniącego rolę serwera (czyli systemu zarządzanego) musi więc być przekazywana informacja o obiekcie-kliencie (systemie zarządzającym), który zainicjował takie żądanie dostępu. Następnie system zarządzany może podjąć decyzję, czy autentykować żądanie, a w konsekwencji – dopuścić do realizacji zamierzonego zadania. Jedyną rozsądną metodą rozwiązania tego problemu jest utrzymywanie w systemie zarządzanym (czyli w konsekwencji w każdym systemie) list kontroli dostępu.

Ponieważ bezpieczeństwo informacji wydaje się najsłabszą stroną architektury CORBA, oparte na niej zaawansowane techniki zarządzania nie są preferowane w sieciach o charakterze publicznodostępnym.

2.4. Model informacyjny zasobów zarządzanych

W systemie zarządzania opartym na architekturze CORBA, informacja zarządzania nie jest – jak w systemach tradycyjnych – definiowana w języku ASN.1 [3], gdyż nie sprzyja on konwencjom obiektowym (zwłaszcza dziedziczeniu). Opis struktur zarządzanych realizowany jest tu przy użyciu mechanizmów architektury CORBA. Definiowanie obiektów odbywa się poprzez specyfikację ich interfejsów w języku IDL (ang. Interface Definition Language). Każdej klasie obiektu zarządzanego przypisany zostaje jeden interfejs IDL

opisujący atrybuty związane z tym obiektem i operacje, jakie można na nim wykonywać. Tu też ujawniają się duże możliwości technologii obiektowej, która dzięki mechanizmom dziedziczenia pozwala uprościć i zorganizować modele informacyjne.

Zastosowanie języka IDL pozwala na ujednolicenie dostępu do obiektów osłaniając ich złożoność przed klientem. Klient wykonując żądanie dostępu do obiektu nie musi (a nawet nie może) znać szczegółów jego rzeczywistej struktury, takich jak np. architektura, system operacyjny czy język, w którym zaimplementowany został serwer (obiekt).

Dostęp do tak tworzonych opisów obiektów jest ułatwiony dzięki istnieniu tzw. składnicy interfejsów (ang. Interface Repository), stanowiącej rodzaj rozproszonej bazy danych przechowującej informacje o wszystkich dostępnych, zdefiniowanych interfejsach. Wprowadzenie do systemu nowej implementacji obiektu związane jest z zarejestrowaniem interfejsu tego obiektu w składnicy interfejsów, dzięki czemu nowy obiekt staje się automatycznie dostępny dla istniejącej infrastruktury systemu. Klient, chcąc wykonać operację na obiekcie, najpierw kieruje zapytanie do składnicy interfejsów w celu poznania struktury i składni usług obiektu, po czym, dzięki istnieniu dynamicznego interfejsu wywołań (ang. Dynamic Invocation Interface), ma już swobodę konstruowania żądań dostępu do usług obiektu zgodnych z tą składnią.

Połączenie mechanizmów składnicy interfejsów i dynamicznego interfejsu wywołań pozwala odseparować system zarządzania od zmian mających miejsce w środowisku zarządzanym. Dostrojenie się zarządcy do ewoluującego środowiska może następować w sposób niemal automatyczny. Ma to ścisły związek z dodatkową, istotną korzyścią: nie jest konieczne przechowywanie dokładnego obrazu każdego obiektu zarządzanego w pamięci systemu zarządzającego, jak dzieje się to w przypadku konwencjonalnych systemów zarządzania wymagających znajomości odpowiednich baz MIB.

2.5. Lokalizacja obiektów

Obiektowe podejście architektury CORBA w połączeniu z maskującym charakterem interfejsów IDL sprawia, że wiele właściwości obiektów zostaje ukrytych przed klientem (obiekty identyfikowane są przez wskaźnik do obiektu – ang. Object Reference). Zwłaszcza dotyczy to właściwości różniących między sobą obiekty (ujednolicenie przez IDL), a więc pozwalających je identyfikować, takich jak lokalizacja obiektu. Z punktu widzenia funkcjonalności zarządzania jest to niekorzystne, ponieważ informacja lokalizacyjna jest niezbędna w sytuacjach zakłócenia poprawnej pracy środowiska, przed którymi system zarządzania ma zabezpieczać. Problem ten można rozwiązać korzystając z metod składnicy interfejsów, w ten sposób uzyskując funkcjonalność analogiczną do automatycznego odkrywania sieci.

3. Zarządca rozproszony

Współczesne systemy zarządzania cechują się coraz bardziej globalnym charakterem. Obecny poziom rozwoju technik telekomunikacyjnych i teleinformatycznych łatwo pozwala pojedynczemu dostawcy objąć zasięgiem swoich usług ogromną powierzchnię. Poszczególne węzły systemu, odległe przestrzennie, komunikują się ze sobą przekazując nie tylko dane, ale coraz częściej dzieląc się ze sobą również usługami. Tym samym kod programowy może być traktowany jako jedna z wielu postaci zasobów współdzielonych w sieci. Obserwowane w systemach zarządzania tendencje zdążają do coraz silniejszego rozproszenia inteligencji i mocy obliczeniowej. Bardzo atrakcyjna wydaje się koncepcja takiego systemu w postaci sieci zarządzającej pokrywającej sieć zarządzaną.

3.1. Anachronizm platform scentralizowanych

Dotychczasowa koncepcja systemów zarządzania opiera je na rozbudowanych, wyspecjalizowanych, dysponujących dużymi zasobami centralnych punktach kontroli, zwanych platformami zarządzania. Podczas gdy kolekcja danych zarządzania i wykonywanie akcji są realizowane w sposób rozproszony, obsługa danych i większość czynności obliczeniowych pozostaje scentralizowana na platformie zarządzającej [4]. Na tym modelu opiera się wiele znanych i stosowanych rozwiązań, takich jak: SolsticeEM [5] czy HP OpenView [6]. Dalsze rozbudowywanie centralnych platform zarządzania wydaje się brnięciem w ślepą uliczkę. Ich coraz większe wymagania sprzętowe, komunikacyjne oraz złożoność sprawiają, że zamiast zapewniać rosnącą niezawodność – same stają się słabymi punktami [7].

Główną wadą centralizacji zarządzania są problemy skalowalności. Poprawność funkcjonowania takiego systemu jest związana z obiektywnymi czynnikami, takimi jak:

- liczba elementów zarządzanych,
- poziom złożoności zarządzania,
- dostępność pasma komunikacyjnego.

Obecne tendencje rozwojowe zmieniają te parametry na niekorzyść, a zmiana na niekorzyść któregokolwiek z nich może spowodować, że system zarządzania stanie się „wąskim gardłem", w konsekwencji czego cała sieć przestanie być zarządzalna.

3.2. Rozproszenie lokalizacyjne

Rozwiązaniem tego problemu jest odejście od modelu zarządcy scentralizowanego na korzyść zarządcy rozproszonego, czemu sprzyja zastosowanie obiektowej architektury CORBA. Pierwszym krokiem w tym kierunku jest podział platformy zarządzającej i

pracujących na jej szczycie aplikacji na mniejsze części. Można to uzyskać wprowadzając systemy zarządzania poziomu pośredniego, rozmieszczone bliżej zasobów zarządzanych. Systemy te mogą wykonywać część funkcji zarządzania (jak np. filtrowanie i korelacja zdarzeń, logowanie, akcje automatyczne itp.) na rzecz innych systemów zarządzających. Mogą też zawierać podręczną pamięć często używanych informacji (jak np. dane topologiczne). Dzięki temu strumienie danych związanych z zarządzaniem ulegają zagęszczeniu redukując ruch niezbędny do utrzymania funkcjonalności.

3.3. Rozproszenie aplikacyjne

Kolejnym etapem na drodze do rozproszenia zarządcy jest ujednolicenie aplikacji zarządzających w postaci usług i ogólne udostępnienie ich w środowisku zarządzania. Dzięki temu funkcjonalność niezbędna do realizacji większości czynności zarządzania jest zdefiniowana i zaimplementowana tylko raz, po czym może być używana wspólnie przez wszystkie rodzaje aplikacji biorących udział w czynnościach zarządzania. Zaimplementowanie usług w postaci obiektów i wykorzystanie mechanizmów dziedziczenia, składnicy interfejsów oraz dynamicznego interfejsu wywołań pozwala, by każdy system, zarówno zarządzający, jak i zarządzany (o ile takie rozróżnienie jest jeszcze stosowane), mógł swobodnie skorzystać z potrzebnej funkcjonalności tych współdzielonych usług. Możliwe jest wtedy zwiększenie skalowalności dzięki przerzuceniu funkcji typowych dla zarządcy do systemów leżących niżej w hierarchii zarządzania.

Usługi zarządzania w postaci obiektowej można wprowadzić przez utworzenie całkowicie nowych, pożądanych funkcji i udostępnienie ich jako nowych udogodnień architektury CORBA. Jednak bardziej atrakcyjna jest koncepcja wykorzystania funkcji zdefiniowanych w już istniejących architekturach i zaadaptowanie ich przez enkapsulację mechanizmami architektury CORBA, zwłaszcza, że umożliwia to integrację tych środowisk.

3.4. Przesunięcie funkcjonalności do agentów

Usługowe ukierunkowanie zarządzania zakłada, że jego funkcjonalność nie jest związana z systemem zarządzającym, ale może być rozproszona i używana przez wszystkie systemy biorące udział w czynnościach zarządzania. Narzędzia usług zobiektowanych pozwalają rozszerzyć koncepcję systemów zarządzania poziomu pośredniego przerzucając ich funkcje do inteligentnych agentów [8], [9].

Głównym atutem jest fakt, iż przesunięcie funkcjonalności może być realizowane dynamicznie, w czasie działania agenta. Dzięki temu systemy zarządzane nie muszą

statycznie utrzymywać nadmiernej funkcjonalności zarządzania, ale mogą zażądać dostarczenia jej w zależności od potrzeb. Skutkuje to uniwersalną i małą strukturą agenta.

Przy zastosowaniu uogólnionego obiektowego modelu rozproszonego, zarządzanie przez przesunięcie funkcjonalności może działać pomiędzy dowolnymi rodzajami obiektów. Możliwe jest więc, by agenci współpracowali ze sobą bez udziału systemu zarządzającego, co może występować zwłaszcza w przypadku funkcji niższego poziomu. Agent wykrywając niepoprawność funkcjonowania podległych mu zasobów może od innego agenta zażądać wykonania odpowiednich procedur testujących. Szczegółowe dane nie muszą być tu przekazywane, ponieważ informacja o tym, jakie usługi są dostępne w danych okolicznościach, może być dynamicznie uzyskana ze składnicy interfejsów. Na tej podstawie agent za pośrednictwem dynamicznego interfejsu wywołań generuje żądanie, a odpowiednie usługi zostają do niego przekazane i mogą być wykonane. W akcji tej biorą udział identyczne moduły agentów, którym role systemu zarządzającego i zarządzanego zostały przyporządkowane dynamicznie.

W takiej architekturze rola nadrzędnego systemu zarządzającego, którym teoretycznie może być jeden z wyróżnionych agentów rozbudowany o funkcje interfejsu użytkownika, zostaje ograniczona do funkcji wymagających interwencji operatora lub umożliwiających mu wgląd w aktualny stan środowiska zarządzanego.

Ważną zaletą przedstawionej tu konwencji jest łatwość uzyskania redundancji, i to zarówno w dostępie do informacji zarządzania, jak i w dostępie do funkcjonalności systemu. Możliwe jest też manipulowanie rozkładem obciążenia (równoważenie obciążenia) pozwalając w sposób zoptymalizowany wykorzystywać zasoby zarządzające. Naturalną cechą łatwą do uzyskania jest tu samoczynne rozkładanie się obciążenia w sposób równomierny.

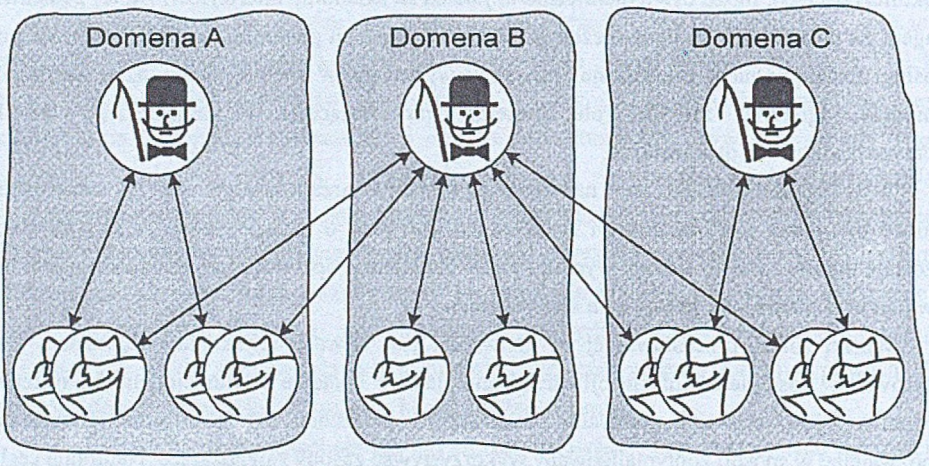
4. Współpraca różnych środowisk zarządzania

Ze względu na dużą skalę i upowszechnienie tradycyjnych i sprawdzonych standardów zarządzania, przy tworzeniu nowej technologii niezbędne jest zapewnienie jej współpracy z systemami już istniejącymi. Pewne zasoby zarządzane będą nadal funkcjonowały zgodnie z nie zmienionymi zasadami, zatem zmiana konwencji spowodowałaby wykluczenie ich z domeny zarządzania. Aby nowy system był dobry, musi zapewniać nie tylko współistnienie, ale także przezroczystą współpracę z innymi architekturami.

W podejściu tradycyjnym najczęściej stosowane są dwa modele integracyjne: na poziomie zasobów zarządzanych lub na poziomie zarządcy [10].

4.1. Integracja na poziomie zasobów

Integracja na poziomie zasobów zarządzanych (rys. 3) opiera się na tworzeniu agentów zdolnych do współpracy z różnymi architekturami zarządzania. Jednak agent realizujący wiele protokołów zarządzania staje się modułem zbyt rozbudowanym, a zatem trudnym do skonstruowania i obsługi. Wiele zasobów ma zbyt prostą konstrukcję, by utrzymywać złożone lub nietypowe oprogramowanie, lub też ma wbudowane przez producenta i niemodyfikowalne oprogramowanie agenta. Trzeba też uwzględnić wymaganie zachowania prostoty w niektórych urządzeniach (domenach).



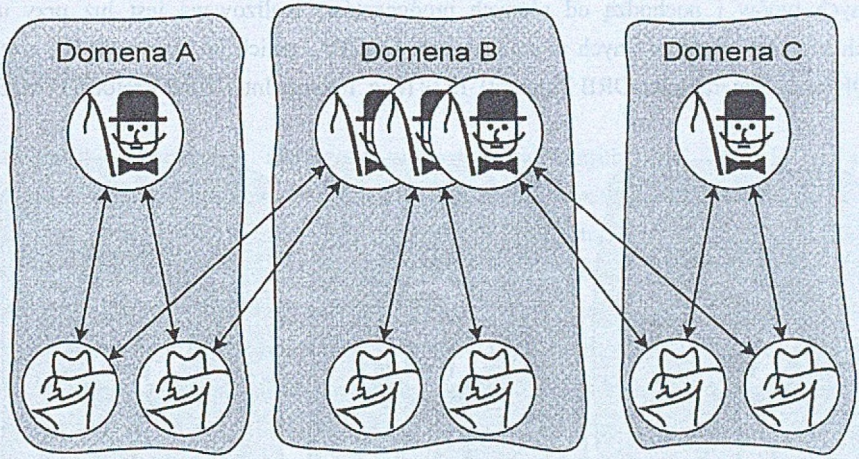
Rys. 3. Integracja na poziomie zasobów
Fig. 3. Resources level integration

4.2. Integracja na poziomie zarządcy

Część tych problemów rozwiązuje model alternatywny przerzucający obowiązki integracyjne na zarządcę (rys. 4): taki rozbudowany zarządca realizuje wiele protokołów zarządzania i posiada wbudowane narzędzia translacji poszczególnych mechanizmów zarządzania. Główną przesłanką tego podejścia jest fakt, iż łatwiej jest przeprowadzić reorganizację po stronie zarządcy, niż przebudować wszystkie zasoby zarządzane. Wadą jest natomiast pogłębianie centralizacji platform zarządzających (skutkiem ich nadmiernej rozbudowy) oraz wbudowanie nowych funkcji w sposób sztywny, trudny do modyfikacji i specyficzny dla konkretnego produktu.

4.3. Model integracyjny architektury CORBA

Systemy zarządzania cechują się rosnącą złożonością, a integracja drogą ingerencji w konstrukcję wewnętrzną istniejących systemów powoduje dodatkową komplikację ich struktury. Organizacją współpracy z innymi środowiskami preferowaną ze względu na cechy obiektowe i zalety rozproszenia jest enkapsulacja tych środowisk mechanizmami architektury CORBA. Dzięki technologii obiektowej możliwe jest uzyskanie pełnej przezroczystości zarówno informacji zarządzania, jak i usług.

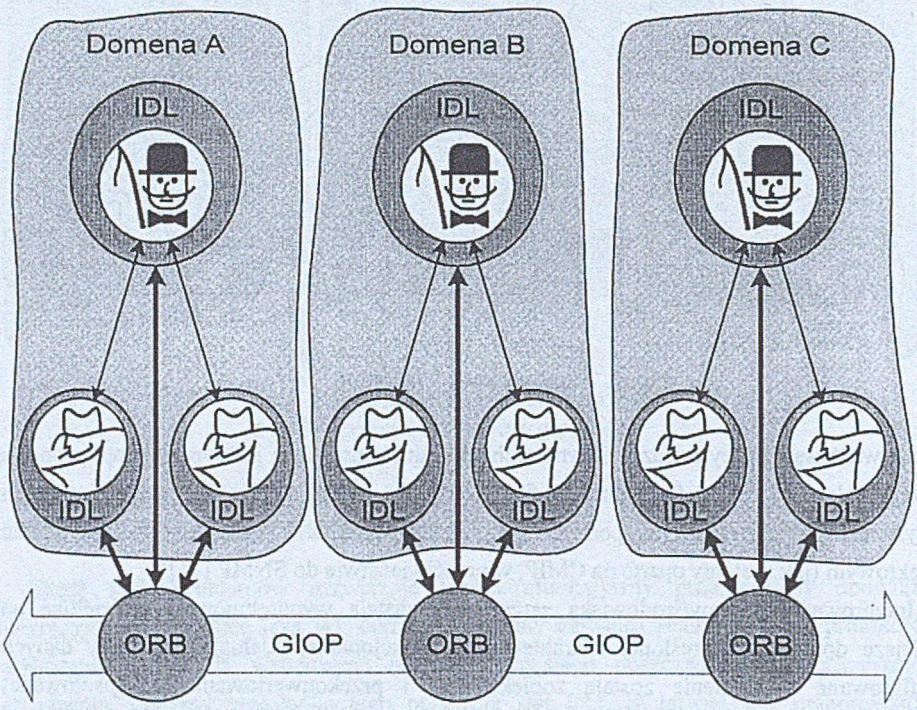


Rys. 4. Integracja na poziomie zarządcy
Fig. 4. Manager level integration

Łatwość współpracy między różnymi architekturami zarządzania jest wprost proporcjonalna do tego, jak podobnie zdefiniowano ich funkcjonalność. Zatem najdogodniejsze do integracji z systemem zbudowanym na podstawie architektury CORBA są systemy o charakterze obiektowym (jak systemy oparte na CMIP, w przeciwieństwie do SNMP [11]).

Integrowane systemy/środowiska zarządzania zostają wyodrębnione i podzielone na mniejsze domeny o określonej charakterystyce funkcjonalnej. Usługi i struktury danych zdefiniowane w domenie zostają zobiektowane i przekonwertowane (enkapsulowane) poprzez tzw. bramy zarządzania (Management Gateways) do postaci zgodnej z architekturą CORBA [12], [13]. Zadaniem bram zarządzania jest tłumaczenie modeli komunikacyjnych przez odwzorowanie wiadomości protokołu zarządzania na żądania i odpowiedzi ORB, a także konwersja struktury informacji zarządzania (modeli informacyjnych). Osłonięcie usług i struktur informacyjnych językiem IDL ujednolica technikę dostępu ukrywając szczegóły implementacyjne i architekturalne. Dzięki temu w każdą domenę może być wbudowany moduł

komunikacyjny ORB, scalający ją w indywidualne, pełnowartościowe środowisko w architekturze CORBA. W tak skonstruowanych domenach nie jest potrzebny specjalny protokół, ponieważ obiekty współpracują ze sobą poprzez wzajemne wywoływanie swoich metod. Cały ciężar komunikacji przejęty jest przez moduł ORB (ang. Object Request Broker). Transmituje on wywołania klienta do obiektu bez względu na ich wzajemne zależności związane z lokalizacją czy specyficznymi cechami implementacji. Połączenie tak zorganizowanych domen zarządzania sprowadza się więc do połączenia ze sobą poszczególnych modułów ORB (rys. 5). Komunikacja pomiędzy ORB (nawet jeśli są one różnych typów i pochodzą od różnych producentów) realizowana jest już przy użyciu mechanizmów zdefiniowanych w architekturze CORBA, takich jak: mosty ORB, protokoły GIOP (ang. General Inter-ORB Protocol), IIOP (ang. Internet Inter-ORB Protocol) i inne.



Rys. 5. Model integracyjny architektury CORBA
Fig. 5. CORBA architecture integration model

Użycie ORB może być całkowicie przezroczyste dla integrowanych systemów zarządzania. Jest wtedy możliwe swobodne użycie dotychczas stosowanych protokołów

zarządzania do wymiany informacji z systemami zarządzanymi. Enkapsulacja istniejących środowisk zarządzania w architekturę CORBA może być uważana jako krok pośredni w kierunku realizacji zarządzania opartego w całości na tej architekturze.

Ważną zaletą jest tu fakt, iż funkcjonalność zdefiniowana w architekturach poddanych integracji może być również użyta w środowisku architektury CORBA. Aby uzyskać dostęp do tych usług, wystarczy, by miały one interfejsy stworzone w języku IDL. Nie ma już znaczenia, w jakim stopniu integrowana architektura ma charakter obiektowy, jeżeli interfejsy jej komponentów sprawiają, że wydają się one obiektami.

5. Podsumowanie

Środowiska zarządzane, zwłaszcza w przypadku dużych sieci, zwykle są silnie heterogeniczne. Poszczególni producenci sprzętu zazwyczaj wykazują dość wąską specjalizację, co sprawia, że do różnych zadań wykorzystuje się często skrajnie różne technologie. Ponadto zbiór obiektów zarządzanych może podlegać ciągłym modyfikacjom, za którymi często muszą podążać również modyfikacje systemu zarządzania. Uwzględnić należy też rozwój technologiczny i powstawanie zupełnie nowych potrzeb, nowych typów obiektów, nowych aspektów zarządzania. Istotnym więc wymaganiem jest, by narzędzia zarządzające były elastyczne, łatwe w rozbudowie i konserwacji. Muszą też zapewniać jak największe możliwości integracyjne z różnorodnym sprzętem i oprogramowaniem.

Ponieważ architektura CORBA została stworzona w odpowiedzi na potrzeby integracyjne środowisk heterogenicznych, doskonale rozwiązuje wiele problemów tak scharakteryzowanego zarządzania. Przede wszystkim jest to architektura otwarta, ukierunkowana na akceptowanie różnorodnych standardów, a więc bardzo rozwojowa. Umożliwia współpracę elementów opartych na różnych technikach i językach programowania, zarówno elementów tworzonych, dedykowanych dla środowiska CORBA, jak i elementów istniejących już wcześniej.

Łatwość integrowania środowisk programowania ma znaczenie proporcjonalne do złożoności i rozmiarów tworzonego systemu. Różne komponenty takiego systemu mogą być wówczas tworzone w różnych technologiach w zależności od potrzeb specyficznych dla danego zadania i możliwości odpowiednich technik programowania. Tu głównie uwidaczniają się zalety architektury CORBA jako dobrze opisanej i zestandaryzowanej, zwłaszcza jeśli chodzi o specyfikacje języków i interfejsy programowania. Ilość i dostępność z nią związanych narzędzi ciągle wzrasta wpływając na upowszechnienie tej technologii.

Bardzo ważna dla projektantów dużych systemów jest również możliwość przenoszenia architektury CORBA na różne platformy, zarówno systemowe, jak i sprzętowe. Aplikacje

klienckie bez potrzeby zmian w kodzie mogą całkowicie przezroczyście współpracować z dowolnym modułem ORB, o ile tylko realizuje on odwzorowanie odpowiedniego języka. Również współpraca z różnymi implementacjami obiektów jest przezroczysta, ponieważ odwołania do nich realizowane są przez wskaźniki do obiektów, maskujące wszelkie szczegóły implementacyjne.

Cechy naturalnie wynikające z koncepcji architektury to łatwość uzyskania redundancji, równoważenia obciążenia (ang. load balancing) i dostępności zarówno do informacji, jak i do funkcjonalności. Są to cechy szczególnie ważne w dużych systemach zarządzania.

LITERATURA

1. The Common Object Request Broker: Architecture and Specification, Revision 2.0. Object Management Group, July 1995.
2. The Common Object Request Broker: Architecture and Specification, Revision 2.2. Object Management Group, February 1998.
3. ITU-T Rec. X.208, Information Technology – Open Systems Interconnection, Specification of Abstract Syntax Notation One (ASN.1). 1988.
4. Sloman N.: Network and Distributed Systems Management. Addison-Wesley, June 1994.
5. Solstice Enterprise Manager Reference Manual, Revision A. Sun Microsystems, Inc., August 1996.
6. HP Openview IT/Operations Concept Guide. Hewlett Packard, 1997.
7. Wellens C., Auerbach K.: Towards Useful Management. "The Simple Times", 4(3), July 1996
8. Goldszmidt G., Yemini Y.: Distributed management by delegation. Proceedings of the 15th International Conference on Distributed Computing Systems, June 1995.
9. Goldszmidt G.: Distributed Management by Delegation. PhD thesis, Columbia University, 1996.
10. Kalyanasundaram P., Sethi A.: Interoperability Issues in Heterogeneous Network Management. Journal of Network and Systems Management, vol. 2, Plemym Publishing Corporation, June 1994.
11. RFC 1157, A Simple Network Management Protocol (SNMP). May 1990.
12. Mazumdar S.: Inter-Domain Management between CORBA and SNMP. Proceedings of the IFIP/IEEE International Workshop on Distributed Systems: Operations & Management, L'Aquila, Italy, October 1996.

13. Soukouti N., Hollberg U.: Joint Inter-Domain Management: CORBA, CMIP and SNMP. Proceedings of the 5th International IFIP/IEEE Symposium on Integrated Management (IM), San Diego, USA, May 1997.

Recenzent: Dr inż. Rafał Cupek

Wpłynęło do Redakcji 27 marca 2001 r.

Abstract

This article discusses application of CORBA architecture to modern network management systems. It describes object features of large scale management systems (Fig. 1) and consequeces of advanced simplification of object model. The agent and manager uniformity paradigm is also presented (Fig. 2) as well as its good and bad points. The article also discusses how the use of CORBA mechanisms influences management information model making it dynamic and allowing automatic configuration of information concerning managed objects. Highly centralized management platforms are depicted as a potential bottleneck of future network technologies. Distributed management systems are shown as a solution of this problem. The article discusses some aspects of distributed management such as mid-level management systems, encapsulation or implementing applications as CORBA objects, application distribution and dynamic functionality delegation to management agents (a kind of network self-management). Integration of different management domains is discussed herein. Two typical integration models are shown: resources level integration (Fig. 3) using multiarchitectural agents, and manager level integration (Fig. 4) using multiarchitectural managers. Fig. 5 presents integration model using CORBA mechanisms such as IDL encapsulation, which allows standard management environments to operate without significant modifications. The summary presents some CORBA architecture features, which make it an attractive solution for modern management in heterogenous environments.