REPLIKACJA W USŁUGACH KATALOGOWYCH X.500 I LDAP A PODSTAWOWE TECHNIKI REPLIKACJI W SYSTEMACH ROZPROSZONYCH

Maja Górecka
Maja.Gorecka@cc.uni.torun.pl

Uniwersyteckie Centrum Technologii Sieciowych, UMK
Naukowa Akademicka Sieć Komputerowa NASK
Zakład Rozproszonych Systemów Informacyjnych

Celem artykułu jest prezentacja technik replikacji, stosowanych w usługach katalogowych. Jak każdy system rozproszony, profesjonalnie prowadzona, operacyjna usługa katalogowa może istotnie poprawić swoją efektywność, jeśli umiejętnie zastosujemy replikację. Pierwszy punkt pracy jest krótkim opisem funkcji usługi katalogowej. Więcej informacji można znaleźć w pracy [1] oraz dostępnej w sieci książce na temat X.500 ([3]). W części 2 jest omówione zagadnienie dystrybucji zasobów w zasobach katalogowych. Część 3 prezentuje popularne techniki replikacji i tendencje związane z ich stosowaniem w systemach rozproszonych. Kolejne części prezentują techniki replikacji implementowane w systemach katalogowych.

1. Wprowadzenie

Większość nowoczesnych systemów informatycznych jest umiejscawiania w rozproszonym środowisku sieciowym i korzysta z infrastruktury komunikacji opartej na sieciach komputerowych. Systemy takie zazwyczaj nie używają wyłącznie danych i urządzeń lokalnych, ale również sięgają do różnego rodzaju zasobów sieciowych, takich jak inne aplikacje, zdalne bazy danych, odległe urządzenia, pliki rezydujące na obcych stacjach komputerowych. Ważnym problemem staje się łatwa lokalizacja oferowanych w sieci zasobów. Oprócz adresu konkretnego obiektu zdalnego, często są potrzebne dodatkowe szczegóły na jego temat. Bywa też, że należy zapewnić właściwą kontrolę dostępu do zasobów, by nie dopuścić do naruszenia prawa własności.

Usługa katalogowa jest mechanizmem umożliwiającym lokalizację, identyfikację i korzystanie z zasobów sieciowych. Jest uporządkowanym sposobem klasyfikacji tych zasobów. Wspomaga wyszukiwanie w zawiłym środowisku sieciowym aplikacji, serwerów, użytkowników. Architektura usługi katalogowej stosuje popularny model klient-serwer. Rolę serwera pełni system zarządzający bazą danych, strona kliencka to niezależny lub wbudowany w konkretną aplikację interfejs, korzystający z zasobów katalogowych, np. w celu zautomatyzowania takich zadań jak znalezienie odpowiedniej drukarki sieciowej, czy określenie członków listy dyskusyjnej. Usługa katalogowa może być silnie specjalizowana, jak np. *Domain Name System* lub przeznaczona do różnorodnych, szeroko pojętych zastosowań — przykłady takich systemów to X.500, *Lightweight Directory Access Protocol* — LDAP oraz *Microsoft Active Directory*.

Technologia korzystania przez systemy informacyjne z globalnych zasobów katalogowych została zdefiniowana w międzynarodowych rekomendacjach, wydanych w 1988r. Były one wynikiem prac grupy roboczej CCITT (*Consultative Commitee on International Telegraphy and Telephony*), obecnie ITU-T (*International Telecommunication Union — telecommunication sector*) oraz ISO (*International Standards Organization*). Kolejna wersja tego standardu, sygnowana X.500'93, pojawiła się w 1995 r. Usługa X.500 jest zaprojektowana w warstwie aplikacji siedmiopoziomowego modelu OSI (*Open System Interconnection*). Ta międzynarodowa definicja stała się nie tylko podstawą do tworzenia produktów implementujących X.500, ale również spowodowała rozwój innych technik obsługi globalnych zasobów sieciowych, takich jak np. protokół LDAP. *Lightweight Directory Access Protocol* początkowo rozwijał się jako protokół dostępowy do zasobów X.500 (oryginalny protokół dostępowy X.500, ang. *Directory Access Protocol* — DAP jest skomplikowany i nie daje łatwych mechanizmów tworzenia aplikacji użytkowych). Z czasem LDAP zyskiwał znaczną popularność i rozszerzył swój

zakres. Obecnie pretenduje do roli globalnej internetowej usługi katalogowej. Jednak szczegółowe specyfikacje LDAP-a są ciągle przedmiotem prac grup roboczych, także replikacja — przedmiot dyskusji w niniejszym artykule, jest w fazie standaryzacji.

Zasoby usługi katalogowej są lokalizowane w specjalnej bazie danych, nazywanej książką informacyjno-adresową lub w skrócie katalogiem. Zakres informacji umieszczanej w takiej bazie jest bardzo różnorodny. Mogą to być mniej lub bardziej rozbudowane informacje o jednostkach organizacyjnych i osobach w nich zatrudnionych (adresy, telefony, faxy, e-maile, tytuły naukowe, stanowiska itp.), a także opisy zasobów sieciowych: urządzeń, aplikacji, struktury domenowej. Podstawową jednostką informacyjną bazy jest **encja** (ang. *entry*), nazywana zamiennie **wpisem**. Utrzymuje ona informację na temat konkretnego egzemplarza obiektu. Obiekty o podobnej charakterystyce są identyfikowane przez wspólną **klasę obiektów** (ang. *object class*). Każdy wpis w bazie X.500 należy do co najmniej jednej klasy obiektów. Jednostką informacji w ramach encji jest **atrybut** (ang. *attribute*). Składa się on z **typu atrybutu** (ang. *attribute type*) oraz jednej lub więcej **wartości atrybutu** (ang. *attribute values*).

Istotną specyfiką usługi X.500 jest hierarchiczna struktura zasobów oraz identyfikowanie obiektów za pomocą **nazwy wyróżnionej** (ang. *distinguished name*, DN), powstającej poprzez dołączenie do nazwy encji macierzystej (nadrzędnej) **relatywnej nazwy wyróżnionej** (ang. *relative distinguished name*, RDN). Zapytania nadchodzące do systemu obsługującego zasoby katalogowe podają **domniemaną nazwę** obiektu (ang. *purported name*), która ma właściwą dla X.500 składnię (sekwencja RDN-ów), chociaż nie musi być nazwą obiektu istniejącego w zasobach. Procedura nazywana **analizą nazwy** (ang. *name resolution*) sprawdza, czy obiekt o zadanej nazwie istnieje. Z logicznego punktu widzenia jest to sekwencyjne dopasowanie kolejnych RDN-ów nazwy wyróżnionej do poszczególnych gałęzi drzewa X.500. Proces analizy nazwy obiektu może mieć, z powodu dystrybucji zasobów, istotny wpływ na efektywność, ponieważ zdarza się, że aby dotrzeć do wskazanego obiektu trzeba wykonać wiele operacji.

Wybór właściwej usługi katalogowej nie jest rzeczą prostą (zob. artykuł [4]). Istnieje kilka produktów ubiegających się o miano globalnej usługi katalogowej. Chociaż podstawą do rozwoju oprogramowania usługi katalogowej stał się standard X.500, często jest on implementowany w ograniczonym zakresie. Prowadzi to do istotnych utrudnień, nieraz nawet uniemożliwia współpracę systemów, używających różnego oprogramowania. Jest to szczególnie istotna sprawa, jeżeli stosowana jest replikacja. Sam standard, zawierający dobrą definicję infrastruktury lokalizacji obiektów w sieci komputerowej, nie jest wystarczający. W celu powszechnej akceptacji prezentowanych rozwiązań konieczna jest dobra dostępność oprogramowania. Omówienie tych problemów można znaleźć w pracy [2].

2. Rozproszony charakter zasobów katalogowych

Ilość danych, które potencjalnie mogą być umieszczone w zasobach katalogowych jest ogromna. Standard X.500 definiuje podstawowe klasy obiektów oraz atrybuty, ale mogą one zostać rozbudowane, w celu dopasowywania schematu bazy do lokalnych potrzeb rodzaju i zakresu gromadzonych danych — istotnie podnosi to możliwości zastosowania prezentowanego systemu do obsługi różnych dziedzin życia. Ze swojej natury usługa X.500 jest globalną, rozproszoną bazą danych zasobów sieciowych, swego rodzaju książką informacyjno-adresową. Utrzymywanie scentralizowanej bazy danych tego typu nie dawałoby dobrych rezultatów. Baza musiałaby być bardzo duża, co zmniejszałoby efektywność pracy, poza tym awaria uniemożliwiałaby całkowicie korzystanie z usługi. Drugim argumentem przeciw takiemu podejściu jest uciążliwość zarządzania danymi. Zasadnicze przyczyny dystrybucji zasobów X.500 są typowe, wspólne dla większości systemów rozproszonych:

- **Względy organizacyjne i ekonomiczne**
  Poszczególne bazy danych, tworzące jeden system globalny są rozdzielone pomiędzy instytucje, które nimi zarządzają. Bardzo często sama natura aplikacji bazodanowych decyduje o ich rozproszonym charakterze. W przypadku lokalnie utrzymywanej bazy jej struktura może zostać rozbudowana stosownie do potrzeb i obsługiwanego środowiska.
- **Połączenie istniejących baz danych**
  Lokalne bazy danych mogą istnieć początkowo niezależnie. Potrzeba korzystania z nich za pomocą jednego rodzaju interfejsu użytkowego wymusza utworzenie globalnej aplikacji, której komponentami są systemy lokalne.

- **Przyrost zasobów**
  Przyczyną dystrybucji zasobów bazy danych może stać się stopniowy wzrost ilości gromadzonej informacji. Scentralizowane systemy operujące na bardzo dużych bazach danych są nie tylko nieefektywne, ale również bardzo trudne w zarządzaniu.
- **Minimalizacja potrzeby zdalnej komunikacji**
  Jeżeli baza danych jest wykorzystywana w sieci komputerowej, to programy klienckie, nazywane interfejsami dostępowymi są zazwyczaj uruchamiane w odległych lokalizacjach. Przybliżenie zasobów do miejsca, z którego są najczęściej używane redukuje nadmiarową komunikację, wprowadza lokalność aplikacji użytkowych.
- **Wymogi efektywnościowe**
  Istnienie kilku autonomicznych procesorów obsługujących system poprawia efektywność dzięki zrównolegleniu działań. Lokalne komponenty obsługiwane są przez własne aplikacje.
- **Niezawodność i dostępność**
  Rozproszone bazy danych, w szczególności takie, które dysponują zdublowanymi kopiami danych, są często wykorzystywane jako metoda zwiększenia niezawodności oraz dostępności sytemu. Wymaga to jednak stosowania różnorodnych technik wspomagających. Awarie w systemie rozproszonym mogą być, ze względu na jego znacznie większą złożoność i rozbudowanie, częstsze niż w środowisku scentralizowanym, ale ich efekt jest związany wyłącznie z aplikacjami wykorzystującymi dane w uszkodzonym węźle. Awaria całego systemu jest sytuacją bardzo rzadką.

Schemat fragmentacji danych w systemach rozproszonych ([5]) definiuje zbiór fragmentów, które gromadzą łącznie wszystkie atrybuty oraz pełne opisy obiektów globalnej bazy danych. Jest odwzorowaniem, przypisującym poszczególne fragmenty węzłom. Sposób rozłożenia fragmentów musi odpowiadać zadanym wymaganiom, zarówno organizacyjnym, jak i funkcjonalnym. Dla jasności i przejrzystości systemu pożądane jest podporządkowanie się zasadzie tworzenia rozłącznych fragmentów danych. Koncepcja fragmentacji drzewa danych X.500 pomiędzy serwery jest bardzo prosta. Globalne drzewo jest podzielone na poddrzewa, zgodnie z potrzebami dotyczącymi lokalnego administrowania. Wielkość poddrzewa jest dowolna, przy czym każda część musi być pełnym poddrzewem, bez „dziur". Poddrzewo rezyduje w jednym serwerze, który zarządza danym fragmentem globalnego drzewa danych, nie może ono być rozproszone między kilka serwerów przed uprzednim podziałem na kolejne poddrzewa. Jeden serwer może przechowywać dowolną liczbę poddrzew, przy czym każde z poddrzew musi być maksymalne — DSA nie może zawierać dwóch uzupełniających się w kierunku pionowym poddrzew. Poddrzewa w zasobach X.500 są identyfikowane poprzez wyróżnioną nazwę encji umieszczonej w korzeniu danego fragmentu jest ona określana **przedrostkiem kontekstu poddrzewa** (ang. *context prefix*). Z każdym poddrzewem jest powiązany zestaw odsyłaczy informacyjnych (ang. *knowledge references*), wskazujących serwery, które przechowują poddrzewa pionowo przylegające do danego fragmentu drzewa. Takie odesłania są utrzymywane przez serwery i stanowią tzw. metainformację o dystrybucji zasobów. Każde poddrzewo, nie zaczynające się od korzenia globalnego drzewa informacyjnego posiada odesłanie do encji **bezpośrednio nadrzędnej**. Takie poddrzewo, które posiada nie tylko encje końcowe, wskazuje serwery przechowujące poddrzewa położone niżej w strukturze za pomocą odsyłaczy do encji **podporządkowanych**. Poddrzewo zawierające encje, przedrostek kontekstu, pełen zestaw odesłań podporządkowanych oraz odesłanie bezpośrednio nadrzędne to zestaw informacji tworzący **kontekst nazewnictwa** (ang. *naming context*) — jednostkę dystrybucji drzewa informacji między serwery DSA.

3. Techniki replikacji danych w systemach rozproszonych

Przed przystąpieniem do prezentacji replikacji w usługach katalogowych, przyjrzyjmy się typowym, najbardziej popularnym technikom replikacji, które są opisywane w literaturze ([5], [6]). **Replikacja** jest strategią stosowaną w systemach rozproszonych, polegającą na przechowywaniu dodatkowych kopii danych. Kopie takie, zwane **replikami** są zazwyczaj umiejscawiane w różnych węzłach systemu rozproszonego, ich lokalizacja wynika z różnorodnych aspektów funkcjonowania konkretnego systemu. Zastosowanie replikacji w systemie rozproszonym daje duże korzyści. Podstawowe zalety to:

- poprawa efektywności realizacji zleceń pobrania danych z zasobów — kopie danych mogą zostać umieszczone bliżej rzeczywistych użytkowników i są używane mniej intensywnie,
- niezawodność — jeżeli jedna kopia danych jest czasowo niedostępna, można w jej miejsce użyć innej.

Nie sposób jednak zyskać nie tracąc czegoś w zamian. Kosztem replikacji jest wzrost złożoności systemu oraz konieczność przechowywania i zarządzania replikami, a nie jest to zagadnienie trywialne. Teoria replikacji jest bardzo rozbudowana. Wiążą się z nią takie problemy jak: transakcyjne przetwarzanie, kontrola współbieżności i niepodzielności, synchronizacja dostępu, spójność zasobów. Na ogół specyfika albo rola systemu narzuca wyraźne żądania odnośnie spójności danych. Niektóre systemy, przede wszystkim te, które pracują w trybie rzeczywistym, wymagają, by były zachowane reguły przetwarzania zadań, gwarantujące synchronizację i właściwe uporządkowanie realizowanych operacji. Opierają się one na pojęciu **transakcji**, jednostki grupującej akcje niezbędne do wykonania zadania, wymagające atomowej obsługi. Przetwarzanie transakcji jest kluczowym pojęciem wielu systemów rozproszonych. Rozwój tej teorii przyczynił się do powstania koncepcji rozproszonych obliczeń i zasobów danych, a także systemów tolerujących uszkodzenia. Każda wykonywana transakcja musi spełniać cztery własności, nazywane w skrócie ACID. Musi ją charakteryzować atomowość (ang. *atomicity*), czyli jest realizowana w całości lub odrzucana, spójność (ang. *consistency*), tj. ma powodować przejście bazy danych z jednego stanu spójnego do drugiego, odseparowanie (ang. *isolation*), czyli jest niezależna od innych zachodzących w systemie działań oraz trwałość (*durability*), co oznacza, że dokonana przez transakcję aktualizacja zasobów nie ulegnie skasowaniu albo wycofaniu.

Jeżeli system musi spełnić ostro postawione wymagania odnośnie spójności, to znacznie komplikuje się obsługa replikacji. Każdej modyfikacji zasobów musi towarzyszyć powielenie nowej informacji do wszystkich replik. Oznacza to, że aktualizacja replikowanych danych musi być transakcją operującą na wszystkich replikach. Jednocześnie replikacja nie może wprowadzać żadnej dwuznaczności z punktu widzenia użytkownika, aspekty użytkowe muszą ukrywać stosowanie replikacji.

Szczegółowa prezentacja typowych technik replikacji zawarta jest w pracy [6].

Oto kilka przykładów technik replikacji, które są stosowane, gdy istotna jest silna spójność zasobów. Najbardziej rygorystyczny, a jednocześnie najprostszy protokół replikacji — metoda **„czytaj jedną kopię, zapisuj do wszystkich kopii”** (ang. *Read one write all*, ROWA) wymaga, by każda operacja zapisu oznaczała, poza modyfikacją oryginalnego elementu danych, aktualizacje wszystkich replik, w których element jest umieszczony. Takie rozwiązanie sprawdza się tylko, gdy przeważają zlecenia odczytu. Daje wówczas dużą niezawodność, dane są dostępne dopóki działa poprawnie co najmniej jeden węzeł posiadający replikę. Wadą takiego podejścia jest konieczność blokowania wszystkich zleceń zapisu, jeśli chociaż jedna z replik nie jest dostępna. Modyfikacją protokołu ROWA jest metoda **„czytaj jedną kopię, zapisuj do wszystkich dostępnych”** (ang. *Read one write all available*, ROWA-A), nazywana algorytmem dostępnych kopii, w której aktualizacje są dokonywane we wszystkich czynnych replikach.

Istnieje sporo rozwiązań stosujących strategię zajmowania większościowego zasobów przed wykonaniem zapisu albo odczytu. Dla potrzeb replikacji wprowadzane jest często pojęcie kopii **podstawowej** (ang. *primary copy*) albo **wyróżnionej** (ang. *distinguished copy*). Według tej teorii, jedna z replik jest traktowana w sposób szczególny. Z nią wiążą się żądania zajęcia elementu danych albo jego zwolnienia. Istnieją metody wykorzystujące ogólne założenia tej strategii, ale różniące się sposobem wyboru kopii wyróżnionych. Można tu wymienić m.in. odmianę protokołu ROWA, korzystającą z kopii podstawowej (ang. *Primary Copy* ROWA). Jedna z replik jest wskazywana jako podstawowa, pozostałe są kopiami **zapasowymi** (ang. *backup*). Operacja zapisu jest wykonywana na kopii podstawowej oraz wszystkich funkcjonujących kopiach zapasowych, natomiast operacja odczytu dotyczy kopii podstawowej. W podejściu tym kopie zapasowe służą do awaryjnego zastąpienia repliki podstawowej. Inna technika jest nazywana po prostu metodą **kopii podstawowej** (ang. *primary copy*). Zasoby są dzielone na jednostki zwane fragmentami, które mogą być replikowane pomiędzy różne węzły. Lokalizacja replik jest znana, dodatkowo wszystkie węzły są uporządkowane w znany sposób. Każdy węzeł przechowuje listę odwołań do swoich partnerów, przechowujących repliki i może ją wykorzystać listę dla określenia, która z kopii jest podstawowa. Kopią podstawową jest ta, która występuje na liście najniżej w określonym uporządkowaniu. Operacja zapisu jest zawsze przekazywana do kopii podstawowej, a potem zmiany wartości elementów są kolejno propagowane do wszystkich replik. Operacja odczytu może, w zależności od decyzji, dotyczyć kopii podstawowej albo jest realizowana w

oparciu o lokalną replikę. Gdy kopia podstawowa ulega uszkodzeniu, kolejna replika, znajdująca się w uporządkowanej liście bezpośrednio powyżej dotychczasowej, staje się bieżącą i w oparciu o nią system kontynuuje swoje działanie. Teoria replikacji określa wiele mechanizmów replikacji opartych na technikach **głosowania** (ang. *voting*) albo **ugody kworum** (ang. *quorum consensous*). Nie stosują one pojęcia kopii wyróżnionej. Zlecenie realizacji określonej operacji, które jest związane, z powodu konieczności nadzoru współbieżności, z zajęciem potrzebnych zasobów, jest przesyłane do wszystkich węzłów zawierających potrzebny element danych. Każda kopia utrzymuje własną pulę reprezentującą zajęte elementy i może przydzielić lub zakazać dostępu do swoich zasobów. Przetwarzanie transakcji musi zostać poprzedzone przegłosowaniem przez większość partnerów prawa do przydzielenia potrzebnych elementów danych. Rodzina protokołów ROWA zdecydowanie faworyzowała operacje odczytu, poza tym nie było tam możliwości tolerowania podziału sieci, pojawiającego się na skutek kłopotów komunikacyjnych. W przypadku strategii opartych na głosowaniu, decyzję o zajęciu zasobów podejmują wszystkie węzły. Mechanizm ten ma swoje wady – występuje tu znaczna ilość komunikatów transmitowanych między poszczególnymi replikami. Dodatkowe utrudnienia wprowadza obsługa awarii, które mogą się zdarzyć w trakcie procesu głosowania. Należy również pamiętać o całej gamie technik mieszanych, które korzystają jednocześnie z metody ROWA lub jej odmian oraz z głosowania i doboru kworum. Istotnym zagadnieniem dla poprawy efektywności opisanych technik silnej spójności jest wprowadzenie możliwości dynamicznego tworzenia reguł obsługi replik — takie rozwiązanie mogłoby uwzględniać specyfikę aktualnej konfiguracji, dostosowywać do typu obsługiwanych operacji i reagować na obciążenie. Przegląd literatury na temat replikacji pokazuje, że jedną z wiodących tendencji jest optymalizowanie podstawowych technik replikacji w celu otrzymania wydajniejszych i bardziej niezawodnych metod.

Zadaniem replikacji jest podniesienie dostępności zasobów oraz poprawa efektywności systemu rozproszonego. Tradycyjne wymagania związane z utrzymywaniem przez cały czas spójnego stanu zasobów w rozproszonych bazach danych są bardzo silne i wprowadzają znaczne obciążenie, zarówno poszczególnych węzłów, jak i sieci komputerowej.

Zaprezentowane powyżej modele replikacji opierają się na transakcyjnym podejściu do zagadnienia replikacji. Narzucają surowe wymagania związane ze spójnością zasobów i niepodzielnością działań na danych. Wśród nowoczesnych strategii, prezentowanych m.in. w pracach [7], [8], [9], [10], coraz częściej pojawiają się techniki, które zezwalają na rozluźnienie potrzeb dotyczących spójności. Takie metody są zazwyczaj dużo bardziej praktyczne, łatwiejsze w implementacji i efektywniejsze w eksploatacji. Wspomniane opracowania zawierają rozbudowane analizy skuteczności i wydajności proponowanych rozwiązań. Należy również zwrócić uwagę, że coraz popularniejsze nowoczesne technologie często uniemożliwiają zapewnienie tradycyjnych warunków pracy w środowisku rozproszonym. Przestaje funkcjonować założenie, że wszystkie węzły są w większości czasu pracy systemu dostępne, a brak łączności z węzłem oznacza jego uszkodzenie. Intensywnie rozwijająca się dziedzina komputerów przenośnych podłączonych do sieci komputerowej (ang. *mobile computing*) powoduje, że zmienia się topologia systemów. Taki węzeł jak np. przenośny komputer może być niedostępny tylko dlatego, że nie jest obecnie włączony, czy nie znajduje się w zasięgu sieci komputerowej. W takiej sytuacji typowe techniki replikacji o charakterze transakcyjnym, realizujące zasadę aktualizacji replik „w każdym miejscu, w każdej chwili, dowolnego elementu danych" nie mogą być wykorzystywane. Aplikacje muszą pozwalać na czasowe występowanie niespójności zasobów. Przykładowo, można dopuścić niezależne wykorzystywanie danych umieszczonych w dwóch częściach sieci, która uległa fragmentacji. W tym kontekście często mówi się o osłabionej, czy ograniczonej spójności replik (ang. *weak consistency*). Najpopularniejszą metodą jest adaptacja przez system żądań z zakresu spójności danych, narzucanych przez konkretną aplikację — z takim właśnie podejściem mamy do czynienia w usługach katalogowych. Natura wielu systemów pozwala na przejściową niespójność danych. Przykładem mogą być systemy, w których gros operacji to odczyt, a operacja zapisu nie musi być natychmiast propagowana do wszystkich replik. Ustala się w takiej sytuacji niezbędny stopień zgodności replik. Dalsze rozróżnienie może dodatkowo podawać, jakiego rodzaju spójność jest zalecana: globalna (z punktu widzenia całego systemu rozproszonego), czy może wystarcza spójność lokalna. Dopuszcza się dwa sposoby aktualizacji replik. Pierwsze podejście, zwane grupowym, ustala, że istnieje zbiór aktywnych replik, a każdy węzeł posiadający kopię danych może aktualizować swoją replikę (strategia ta bywa określana w literaturze metodą wielu kopii głównych, ang. *multimaster*). W drugiej technice, nazywanej metodą jednej kopii głównej (ang. *master*), każdy obiekt danych ma swój węzeł

główny i tylko on może modyfikować kopię podstawową obiektu (ang. *primary copy*). Wszystkie pozostałe repliki mogą być wykorzystywane wyłącznie do operacji odczytu danych i dlatego są nazywane kopiami wtórnymi (ang. *secondary copy*). Będzie o tych metodach mowa w części 4.

Tradycyjny model replikacji, spełniający zasady przetwarzania transakcji w systemie rozproszonym, często nazywany jest **gorliwą propagacją zmian danych** (ang. *eager propagation*). Wymaga się w nim, by transakcja zapisująca nowe dane od razu aktualizowała wszystkie repliki tego elementu, czyli mamy do czynienia z synchroniczną aktualizacją wszystkich replik danego elementu danych w ramach jednej atomowej transakcji. Najnowsze technologie preferują **powolną propagację zmian danych** pomiędzy replikami (ang. *lazy propagation*). Każdy węzeł ma prawo modyfikować lokalne dane. Aktualizacja replik jest dokonywana asynchronicznie, po zatwierdzeniu transakcji zapisu w węźle źródłowym, potem następuje propagacja kopii. Istotną zaletą jest poprawa czasu odpowiedzi systemu. Podejście to stwarza jednak możliwość zatwierdzenia dwóch transakcji aktualizacji , które wprowadzają kolizję. W efekcie konieczne jest zastosowanie transakcji kompensującej, która musi ustalić ostateczny wynik takiej modyfikacji. Z tym zagadnieniem związane są **procedury uzgadniania** (ang. *reconsilation procedures*).

4. Replikacja w usługach katalogowych

Bieżąca część artykułu stanowi przegląd metod replikacji, które są stosowane w usłudze katalogowej opartej na X.500 oraz LDAP na tle technik zaprezentowanych powyżej.

Zajmijmy się najpierw edycją standardu **X.500'88**. Niestety, nie zawierała ona definicji techniki replikacji, ani odpowiednich do potrzeb replikacji rozszerzeń metod realizacji operacji rozproszonych. Już na początku lat 1990 powstało zaimplementowane na podstawie rekomendacji X.500'88 bardzo popularne w środowisku akademicko-naukowym oprogramowanie QUIPU, będące integralną częścią pakietu ISODE. W oparciu o QUIPU został uruchomiony projekt o nazwie PARADISE, który miał służyć przetestowaniu funkcjonowania serwisu X.500 oraz stanowił podłoże do licznych eksperymentów. W konsekwencji prace te przyczyniły się do intensywnego rozwoju kolejnych wersji standardu. Eksploatacja pakietu QUIPU pokazała wiele niedostatków międzynarodowych rekomendacji. W szczególności problemy dotyczyły braku ustaleń dotyczących techniki replikacji zasobów. Mimo że usługa X.500 działała wówczas eksperymentalnie, pojawiła się potrzeba stosowania replikacji. Przykładowo, serwer funkcjonujący w jednostce organizacyjnej dla zwiększenia efektywności powinien mieć kopie danych poziomu kraju albo świata — istnienie tych danych znacznie przyspiesza fazę analizy nazwy encji dostarczonej w zapytaniu. W listopadzie 1991 roku ukazały się dokumenty Internetowe z serii Request for Comments, nr 1275 i 1276, autorstwa S. Hardcastle-Kille z University College London ([11], [12]). Zawierają one zalecenia, dotyczące niezdefiniowanych w standardzie metod realizacji operacji rozproszonych oraz obsługi replikacji. Prezentowane w RFC rozwiązania oparte były na testach efektywności implementacji QUIPU. Autor zastrzegał, że zalecenia te powstały z powodu konieczności wprowadzenia doraźnych rozwiązań, podkreślał wewnętrzny oraz przejściowy charakter proponowanych technik. Fakt, że metody replikacji oraz związane z tym rozszerzenia obsługi operacji rozproszonych nie zostały zestandaryzowane od razu miał znaczny wpływ na trudności współpracy serwisów X.500 zaimplementowanych przez różnych producentów. Stanowiło to również ograniczenie w popularyzacji tej usługi.

Rzeczywiście, propozycja replikacji zawarta w RFC1276 jest modelem bardzo uproszczonym. Zakłada, że w systemie jest możliwe występowanie czasowych niespójności. Takie podejście wynika przede wszystkim z funkcji usługi katalogowej. Zasoby gromadzone w bazach X.500 są przede wszystkim odczytywane i przeszukane. Modyfikacje są znacznie rzadsze. Porównanie z rozważaniami na temat technik replikacji wskazuje, że jest to typowy system o obniżonej, czy wręcz słabej spójności danych.

Mechanizm replikacji stosuje rozszerzony model zasobów X.500, w którym przewiduje się umieszczanie wielowartościowych odesłań, wskazujących serwer zawierający dane podstawowe (ang. *master*) oraz serwery utrzymujące podrzędne kopie danych (ang. *slave*). Aktualizacja dotyczy zawsze danych podstawowych. Zlecenia odczytu mogą korzystać z kopii podrzędnych.

Model danych, rekomendowany w RFC 1276 wprowadza dla potrzeb protokołu replikacji pewne ograniczenia. Określa się mianowicie jednostkę replikacji jako kompletny zbiór encji bezpośrednio podporządkowanych bieżącej encji. Jest to **blok danych encji** (ang. *Entry Data Block*, EDB). EDB

jest nazywany poprzez identyfikację jego encji macierzystej, np. blok danych zawierający wszystkie bezpośrednio podporządkowane dane poziomu c=PL nosi nazwę "c=PL". Blok taki zawiera relatywne nazwy wyróżnione encji bezpośrednio podporządkowanych oraz ich opisy. Główny (ang. *master*) blok EDB składa się z dwóch typów encji. Są to:

1. Encje, którymi DSA zarządza.
2. Kopie encji zarządzanych przez inny serwer, wskazany za pomocą odesłania podporządkowanego. Taka kopia jest utrzymywana automatycznie, dzięki komunikacji z serwerem posiadającym podstawowy blok danych.

Inne serwery mogą przechowywać kopie bloków EDB (tzw. podrzędne EDB, ang. *slave* EDB), które są replikowane w całości bezpośrednio z podstawowego EDB lub z podrzędnego EDB (jeśli mamy do czynienia z wtórnym odzwierciedlaniem danych).

Zastosowanie podejścia, które zaleca dokument RFC1276 i wprowadzenie replikacji w bazie X.500 daje korzyści, które nie są typowe dla wszystkich systemów rozproszonych, lecz wynikają ze specyfiki organizacji zasobów X.500, z ich drzewiastego ustrukturalizowania.

1. Uproszczona jest analiza nazwy encji (ang. *name resolution*), prowadząca do lokalizacji encji w drzewie danych na podstawie jej nazwy, dzięki czemu znacznie podnosi się wydajność.
2. Poprawia się efektywność przeszukiwań, które dotyczą jednego poziomu oraz działań typu „listuj", a także istotnie uproszczona jest implementacja tych operacji. Korzyści wynikają z uniknięcia konieczności kontaktu z licznymi serwerami podporządkowanymi.

Protokół replikacji zdefiniowany w RFC1276 posiada kolejne ograniczenie — zakłada, że stroną inicjującą transfer repliki musi być odbiorca EDB, operacja transmisji ma charakter „ściągania" kopii (*pull-through*). Aktualna replika może być pobierana w określonych odstępach czasowych (domyślnie co 6 godzin) lub jednorazowo, w wyniku jawnego żądania administratora.

Zaletą opisanego mechanizmu replikacji jest prosta implementacja. Podstawową wadą jest niska efektywność, będąca konsekwencją poniższych rozwiązań.

1. Nie jest możliwa obsługa całych poddrzew hierarchicznej bazy informacyjnej, w sytuacji, gdy potrzebne jest powielanie kilku, czy kilkunastu bloków EDB administratorzy muszą wprowadzić odpowiednie, rozbudowane specyfikacje.
2. Nie jest możliwa indywidualna obsługa encji, jeśli dowolny wpis w bloku EDB zmienia się musi zostać odzwierciedlony cały blok EDB.
3. Bloki EDB przesyłane są porcjami tylko w sytuacji serwerów pracujących w domenie Internetu, w innych przypadkach może dochodzić do prób transferu bardzo dużych bloków danych, co często wpływa na obniżenie jakości.
4. Nie jest poprawnie zaimplementowane zabezpieczanie systemu w przypadku pokrywania się operacji transferu replikowanych bloków EDB i modyfikacji danych umieszczonych w tym bloku.

Wyższą wydajność zapewniają techniki inkrementacyjne. Jednak autorzy specyfikacji RFC 1276 uznali, że wykorzystywane metody są wystarczające wobec ówczesnych potrzeb. Zakładali, że kolejne wersje standardu wprowadzą nowe ustalenia i pozwolą na skalowanie modelu.

Rzeczywiście, standard **X.500'93** definiuje w części X.525 ([11]) wyczerpujący, wszechstronny model replikacji. Replikacja jest nazywana w tych rekomendacjach **odzwierciedlaniem, cieniowaniem** (ang. *shadowing*). Standard wymienia również inne sposoby tworzenia dodatkowych kopii informacji, jak np. zastosowanie pamięci podręcznej (ang. *caching*), jednak te metody mogą być realizowane drogą pozaprotokolarną.

Rozwiązanie przyjęte w replikacji w modelu X.500'93 korzysta z osłabionego modelu spójności danych, przedstawionego w drugiej części przeglądu podstawowych technik replikacji. Wśród powodów wyboru takiego modelu wymienia się ponownie argument, że statystyki korzystania z zasobów X.500 wskazują zdecydowaną przewagę zleceń odczytu nad operacjami zapisu. Poza tym, ważną zaletą takiego podejścia jest mniejszy stopień skomplikowania implementacji.

W prezentowanej technice, jest wskazywana kopia główna (ang. *master copy*), czyli miejsce lokalizacji oryginalnych danych. Dane takie są często, może nieco dwuznacznie, nazywane są kopią główną albo kopią podstawową. Inne serwery mogą utrzymywać kopię danych oryginalnych (kopię kopii głównej), czyli kopię wtórną (ang. *shadow copy*). Operacje odczytu mogą (ale nie muszą) korzystać z kopii wtórnych. Operacje zapisu dotyczą zawsze kopii głównej. Następnie zmiany, zgodnie z ustaloną

strategią, są propagowane do kopii wtórnych. Serwer główny (ang. *master DSA*) jest **dostawcą** danych (ang. *supplier DSA*), serwer utrzymujący replikę — **konsumentem** (ang. *consumer DSA*). Informacja podlegająca odzwierciedlaniu to poddrzewo, odpowiadające określonemu kontekstowi nazewnictwa, którym zarządza serwer-dostawca. Można również, poprzez zawarcie specjalnych uzgodnień, zażądać odzwierciedlania wybranych encji z poddrzewa, a nawet w ramach wybranych encji kopiować wskazane atrybuty.

W celu realizacji odzwierciedlania zawierane jest **porozumienie** (ang. *shadowing agreement*) między serwerem-dostawcą a serwerem-konsumentem, które ustala jednostkę replikacji, tryb aktualizacji, nazwę i adres głównego DSA oraz zezwala lub nie na zawieranie porozumień wtórnego odzwierciedlania.

Jednostka replikacji (ang. *unit of replication*), zwana również **obszarem replikacji** (ang. *replicated area*) musi mieścić się w kontekście nazewnictwa, którym zarządza serwer. Jeżeli obszar replikacji kończy się powyżej najniższego poziomu globalnego drzewa danych, to towarzyszy mu zestaw odsyłaczy informacyjnych, po jednym dla każdej encji umieszczonej bezpośrednio poniżej replikowanego obszaru. Odsyłacze takie wskazują serwer, który posiada bliższą informację na temat encji, której dotyczy, niekoniecznie zawiera pełny jej opis. Ta zawężona informacja odnośnie wskazań serwerów przechowujących konteksty nazewnictwa wprowadza pewne opóźnienia w trakcie obsługi zlecenia związane z koniecznością wielokrotnego nawigowania wzdłuż drzewa danych w celu dotarcia do miejsca docelowego. Dla poprawy efektywności rekomendacje dopuszczają stosowanie tzw. **rozbudowanej informacji na temat kontekstu** (ang. *extended knowledge*) poprzez duplikowanie w obszarze replikacji odsyłaczy do encji podporządkowanych w stosunku do danego kontekstu nazewnictwa.

Odzwierciedlanie danych z podstawowego serwera do wskazanego DSA zwane jest **odzwierciedlaniem zasadniczym** (ang. *primary shadowing*). Alternatywna metoda polega na wykorzystaniu kopii danych rezydującej na serwerze, który otrzymał replikę z podstawowego DSA. Serwer ten, jeśli porozumienie z podstawowym DSA na to zezwala, może zawrzeć uzgodnienie z innymi serwerami i przekazywać im wtórną kopię danych, w ten sposób realizowane jest **odzwierciedlanie wtórne** (ang. *secondary shadowing*). Zaletą takiego rozwiązania jest ograniczenie liczby serwerów, do których główny serwer DSA musi przekazać aktualizacją danych, dzięki czemu zostaje rozłożone obciążenie poszczególnych węzłów. Poza tym taka metoda może być bardzo korzystna w konkretnych sytuacjach połączeń między serwerami, np. gdy część serwerów znajduje się w szybkiej sieci LAN, a pozostałe w wolniejszej sieci rozległej. Wadą techniki wtórnego odzwierciedlania jest oddalenie konsumenta od głównego DSA, jest to szczególnie niekorzystne w przypadku konieczności łańcuchowania operacji typu aktualizacja, które muszą zostać wykonane przy użyciu kopii głównej.

Tryb aktualizacji reguluje zasady propagowania zmian. Służy do podania kiedy i w jakich odstępach czasowych ma nastąpić przesłanie nowej repliki oraz kto inicjuje wymianę danych. Do wyboru są trzy możliwości:

1. Inicjuje dostawca po aktualizacji danych podlegających replikacji.
2. Okresowa aktualizacja inicjowana jest przez dostawcę.
3. Okresowa aktualizacja inicjowana jest przez konsumenta.

W przypadku okresowej aktualizacji można wyspecyfikować termin pierwszego transferu, jeśli nie zostanie on podany zakłada się, że transmisja informacji nastąpi zaraz po uaktywnieniu porozumienia dotyczącego odzwierciedlania. Wszystkie dopuszczalne tryby mają charakter asynchroniczny, typowy dla opisanej wcześniej replikacji powolnej, propagacja następuje według ustalonych reguł, po zatwierdzeniu operacji zapisu w węźle utrzymującym kopie główną.

Po pomyślnym zakończeniu fazy uzgodnień między serwerami może nastąpić właściwe przesyłanie danych, które odbywa się zgodnie z **protokołem odzwierciedlania informacji** (ang. *Directory Information Shadowing Protocol*). Jest on zorientowany połączeniowo, podobnie jak DAP (*Directory Access Protocol*) i DSP (*Directory Service Protocol*). Przed właściwą wymianą informacji realizowana jest operacja nawiązania połączenia, która może wiązać się z uwierzytelnieniem stron (jeśli wymagają tego uzgodnienia odzwierciedlania). Protokół DISP zezwala na realizacje replikacji według ustalonego trybu aktualizacji — definiuje dwie możliwości dotyczące strony inicjującej transfer repliki: chęć wysłania kopii zgłasza konsumentowi serwer-dostawca lub serwer-konsument zwraca się do dostawcy repliki ze zleceniem przekazania kopii danych. Operacja aktualizacji repliki może być **pełna** (ang. *total*) lub **przyrostowa** (ang. *incremental*).

Oczywiście replikowana informacja ma służyć poprawie efektywności realizacji zleceń, które są odbierane przez system katalogowy. Zarówno encje jak i odsyłacze informacyjne, występujące w ramach repliki mogą być stosowane do generowania wyniku zapytania wysłanego przez użytkownika bazy X.500. Tylko w poniżej wskazanych sytuacjach nie jest możliwe korzystanie z kopii danych:

- odzwierciedlana informacja nie może zostać poddana operacji modyfikacji,
- użytkownik wystawiający zlecenie może zabronić stosowania repliki w celu tworzenia odpowiedzi poprzez parametr „zakaz stosowania kopii" (ang. *don't use copy*) przesłany w ramach argumentów sterowania usługą.

W środowisku globalnej bazy danych, stosującej techniki replikacji na zapytanie może nadejść wiele odpowiedzi pochodzących z serwera głównego (zarządzającego danymi) oraz serwerów utrzymujących wtórne kopie danych. Taki wynik nie będzie satysfakcjonować użytkownika. Możliwe jest oczywiście usunięcie zdublowanej informacji poprzez serwer obsługujący zapytanie bądź przez inteligentny interfejs obsługujący użytkownika. Nie jest to jednak działanie optymalne, gdyż mamy do czynienia ze zbędnym zużyciem zasobów. Kopie danych istnieją w zasobach po to, by skrócić czas niezbędny do uzyskania odpowiedzi. Jeżeli serwer realizuje zapytanie w technice sekwencyjnego wielokrotnego łańcuchowania, to może na postawie odebranych wyników częściowych odpowiednio ustalać argumenty kolejnych zapytań i żądać wyłączenia określonych obszarów drzewa. Jeżeli pracuje w trybie równoległego wielokrotnego łańcuchowania, jedyną metodą uniknięcia duplikowania wyników jest rezygnacja ze stosowania replik. Ważnym aspektem istnienia replik jest, wspomniana wcześniej, możliwość korzystania z nich w procesie analizy nazwy, czyli lokalizacji obiektu na podstawie jego nazwy, dostarczanej przez interfejs użytkowy.

Podsumowując, można stwierdzić, że zdefiniowana w standardzie X.500'93 technika replikacji zasobów daje skuteczne mechanizmy odzwierciedlania danych katalogowych i pozwala tworzyć jednorodne implementacje. Oczywiście nadal pozostaje problemem używanie replikacji w rozwiązaniach heterogenicznych, czyli łączących węzły korzystające z różnego oprogramowania, które w dowolnym stopniu implementuje przedstawione standardy. Dodatkową trudność, o której nie można zapominać, stanowi niespójna definicja wymagań dotyczących kontroli dostępu do zasobów. Dopiero standard X.500'93 zawiera omówienie zagadnień kontroli dostępu do informacji w bazie(wcześniej systemy katalogowe stosowały własne strategie). Prezentowany model definiowania reguł dostępu do zasobów jest bardzo zawiły, a dodatkowo wprowadzone mechanizmy nie są traktowane jako obowiązkowe. W efekcie nie ma pewności, że dwa systemy działające zgodnie ze standardem X.500'93 identycznie obsługują kwestie dostępu do zasobów. Konsekwencje tego są szczególnie uciążliwe właśnie w technice replikacji, która wymaga dobrego określenia reguł korzystania z zasobów.

Ustalone w standardzie X.500'93 reguły replikacji pozostawiają dużą dowolność administratorom systemów. To oni decydują o trybie aktualizacji, częstotliwości przekazywania replik oraz dobierają stosowną strategię (replikacja pełna bądź przyrostowa). Obszar replikacji może być dopasowywany do konkretnych potrzeb, dzięki filtrowaniu encji i atrybutów. Elementem, który pozostaje najmniej zautomatyzowany, jest faza zawarcia porozumienia replikacji, która wymaga synchronizacji działań administratorów dwóch uczestniczących w wymianie replik węzłów.

Według ukazujących się obecnie dokumentów internetowych, intensywnie rozwijający się **standard LDAP** w dużym stopniu przejmie podstawowe założenia dotyczące replikacji w X.500'93. Dostępne już zalecenia precyzują wymagania odnośnie replikacji w protokole LDAP [14], architekturę replikacji [15] oraz w sposób ogólny określają funkcje procedur uzgadniania aktualizacji [16]. Ponieważ prace standaryzacyjne trwają, nie są dostępne produkty implementujące tę postać replikacji w LDAP-ie. Niemniej, warto przyjrzeć się planom. Replikacja w LDAP-ie różni się od zaprezentowanych wcześniej w tym punkcie technik replikacji zasobów katalogowych przede wszystkim tym, że wprowadza się dwie formy replikacji. Pierwsza to metoda tradycyjna, określana dalej nazwą *single-master*, stosowana również w X.500 — dostępna jest jedna główna kopia danych, wszystkie pozostałe kopie traktowane są jako wtórne. Druga forma replikacji zakłada istnienie wielu kopii głównych, jest nazywana *multi-master*. O regułach korzystania z kopii głównych w technice *multi-master* decyduje typ repliki. LDAP definiuje następujące typy replik:

1. Podstawowa (ang. *primary*) – pełna kopia danych, do której powinny być kierowane wszystkie operacje LDAP wymagające silnej spójności zasobów; wśród replik danego kontekstu nazew-

nictwa tylko jedna może być repliką podstawową (dopuszcza się również brak repliki podstawowej).
2. Modyfikowalna (ang. *updateable*) – replika przyjmująca wszystkie operacje aktualizacji zasobów, ale nie będąca repliką podstawową, kontekst nazewnictwa może nie posiadać tego typu repliki lub dostarczać dowolną ich ilość.
3. Tylko do odczytu (ang. *read-only*) – akceptująca tylko polecenia nie modyfikujące danych.

Dodatkowo z punktu widzenia zakresu encji, biorących udział w procesie replikacji wyróżnia się repliki:

1. Rzadkie (ang. *sparse*) – zawierające wybrane encje z obszaru replikacji (mogą dotyczyć tylko replik modyfikowalnych lub tylko do odczytu).
2. Fragmentaryczne (ang. *fractional*) – gromadzące wybrane atrybuty w ramach encji (muszą zawsze być tylko do odczytu).
3. Częściowe (ang. *partial*) – będące połączeniem metody 1. i 2, repliki częściowe będące fragmentarycznymi (muszą być tylko do odczytu).

W skrajnym przypadku, jeżeli tylko jedna z kopii głównych jest modyfikowalna, replikacja *multi-master* może stać się *single-master*.

Wprowadzenie replikacji, korzystającej z wielu kopii głównych i dopuszczającej aktualizację danych w dowolnej z tych kopii jest istotną zmianą w technologii replikacji zasobów katalogowych. Standaryzacją replikacji *multi-master*, równolegle do prac w ramach LDAP-a, zajmują się grupy robocze X.500. Przygotowywany jest nowy protokół replikacji — *Directory Multi-Master Replication Protocol*, DMRP, który ma zastąpić przedstawiony wcześniej protokół DISP. Technika *multi-master* daje nowe możliwości, ale jednocześnie stawia nowe wymagania. Obsługa replikacji staje się bardziej skomplikowana. Przegląd technik replikacji udowodnił, że im więcej chcemy otrzymać dzięki replikacji, tym trudniejsza implementacja. Strategia *multi-master* korzysta z osłabionego modelu spójności, dopuszcza się czasowe niezgodności w obrębie replik, przy założeniu, że docelowo repliki muszą być identyczne. Ponieważ aktualizacje danych są propagowane między replikami asynchronicznie, mogą wystąpić konflikty aktualizacji, będące efektem zatwierdzenia sprzecznych modyfikacji danych (np. aktualizacja encji, którą właśnie usunięto w innej kopii danych). Ratunkiem w takiej sytuacji jest realizacja wspominanych wcześniej procedur uzgadniania. Zakres działań, które mają wykonywać te procedury jest bardzo rozbudowany, muszą one uwzględniać różnorodne aspekty, przykładowe rozwiązania są zaprezentowane w dokumencie [16]

5. Podsumowanie

Dokonana w niniejszym artykule prezentacja wskazała główne tendencje widoczne w dziedzinach replikacji oraz obsługi zasobów katalogowych. Przegląd podstawowych technik pokazał, że w systemach rozproszonych stosujących replikację warto dopasować formę replikacji do wymagań systemu. Wprowadzenie ostrych wymagań odnośnie spójności zasobów jest bardzo kosztowne i mimo, że wiodącym założeniem replikacji jest poprawa efektywności, może dać efekt odwrotny. Natura zasobów katalogowych nie wymusza wprowadzania silnej spójności. Techniki replikacji powinny być tu jednocześnie skuteczne i proste. Skuteczność oznacza ostateczne doprowadzenie replik do stanu spójności, mimo przejściowego wystąpienia niezgodności danych. Metody replikacji implementowane eksperymentalnie w pierwszych produktach opartych na standardzie X.500 były bardzo proste, wręcz prymitywne. Podstawową ich rolą było ułatwienie wyszukiwania obiektów w rozproszonej, hierarchicznej bazie katalogowej, ograniczenie liczby operacji potrzebnych do odczytania obiektu. Technika wcielona wraz ze standardem X.500'93 jest dużo bardziej zaawansowana, dopuszcza różnorodne konfiguracje, ale główna idee replikacji pozostaje taka sama — dysponujemy jedną kopią główną i opcjonalnie wieloma kopiami wtórnymi — replikami, zlecenia aktualizacji muszą być realizowane w węźle dysponującym kopią główną, kopie wtórne wspomagają operacje odczytu, wyszukiwania oraz analizy nazwy obiektu. Dopiero będąca obecnie w fazie standaryzacji technologia *multi-master* wprowadzi możliwość operowania na kilku kopiach danych. Kierunek działań w zakresie replikacji w usługach katalogowych świadczy o tym, że rozwiązania minimalistyczne nigdy nie są dobre w profesjonalnych systemach. Wymagania użytkowników systemów rozproszonych są różnorodne i muszą zostać udostępnione technologie bardziej zaawansowane. Usługa katalogowa może być stosowana również w systemach, które narzucają potrzebę zagwarantowania wysokiej spójności zasobów. Są przykłady

używania systemów X.500 w takich dziedzinach, jak obronność, ośrodki rządowe, transport, a także w bankowości. Dlatego celem usługi katalogowej jest wprowadzenie elastycznych metod replikacji, które będą pozwalały na dużą dowolność konfiguracji replikacji w systemie.

Bibliografia

[1] M. Górecka, T. Wolniewicz, „Obsługa zasobów informacyjno-adresowych za pomocą protokołu LDAP – przegląd dostępnych narzędzi i porównanie z technologią X.500", Miedzeszyn'98

[2] M. Górecka, T. Wolniewicz, „Europejska usługa katalogowa Paradise-NameFlow: perspektywy i zastosowania", Miedzeszyn'99

[3] D. Chadwick, *Understanding X.500. The Directory.* Chapman & Hall 1994 (`http://www.salford.ac.uk/its024/X500.htm`)

[4] D.S. Linthicum, „Finding your way", artykuł w czasopiśmie DBMS, November 1997

[5] A. Silberschatz, H.F. Korth, S. Sudarshan, „DATABASE SYSTEM CONCEPTS", *WCB/McGraw-Hill*, 1998

[6] A.A. Helal, A.A. Heddaya, B.B. Bhargava, „REPLICATION TECHNIQUES IN DISTRIBUTED SYSTEMS", *The Kluwer International Series on Advances in Database Systems*, 1996

[7] R. Lenz, „The virtual-primary-copy approach compared to other approaches with weak consistent data replication", ECOOP'96, II Workshop on Mobility and Replication

[8] T. Anderson, Y. Breitbart, H. F. Korth, A. Wool, „Replication, consistency, and practicality: are these mutually exclusive?", SIGMOD'98

[9] Y. Breitbart, H. F. Korth, „Replication and Consistency: Being lazy helps sometimes", *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of Database Systems*, 1997

[10] J. Gray, P. Helland, P. O'Neil, D. Shasha, „The danger of replication and a solution", *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, 1996

[11] S.E. Hardcastle-Kille, *„Replication Requirements to provide an Internet Directory using X.500"*, RFC 1275, November 1991

[12] S.E. Hardcastle-Kille, *„Replication and Distributed Operation extensions to provide an Internet Directory using X.500"*, RFC 1276, November 1991

[13] Data Networks and Open System Communications, „Directory; Information Technology – Open Systems Interconnection – The Directory: Replication", ITU-T Recommendation X.525 (11/93)

[14] Russel Weiser, „LDAP Replication Requirements", October 29, 1998, INTERNET-DRAFT, `draft-weiser-replica-req-02.txt`

[15] John Merrells, Ed Reed, „LDAP Replication Architecture", August 5, 1998, INTERNET-DRAFT, `draft-merrells-ldup-model-01.txt`

[16] Steven Legg, „LDAP Update Reconcilation Procedures", December 24, 1998, INTERNET-DRAFT, `draft-legg-ldup-crp-00.txt`