XI Seminarium PLOUG
Warszawa
Czerwiec 2005

# RAC, GRID, Data Guard – technologie zwiększające wydajność i niezawodność bazy danych Oracle 10g

Mariusz Masewicz

*Politechnika Poznańska*

Artykuł omawia podstawowe rozwiązania techniczne serwera bazy danych Oracle 10g umożliwiające konstrukcję środowisk odpornych na awarie. Technologie RAC i GRID wykorzystują redundancję platform obliczeniowych w celu umożliwienia kontynuacji przetwarzania w sytuacji, gdy dochodzi do awarii lub utraty sprzętu. Technologia Data Guard pozwala chronić zawartość baz danych przed utratą w wyniku uszkodzenia lub utraty urządzeń pamięci masowej. Autor przedstawi również założenia Oracle Maximum Availability Architecture – promowanej przez Oracle listy „dobrych praktyk" z zakresu konstrukcji systemów o zwiększonej niezawodności.

## Wstęp

Jedną z wielu miar pozwalających na ocenę jakości współczesnych systemów informatycznych jest ich niezawodność. Pozostałe miary to między innymi funkcjonalność, wydajność i oczywiście cena. W przypadku wielu systemów informatycznych o sukcesie przedsięwzięcia można mówić dopiero po spełnieniu szeregu wygórowanych kryteriów, które często powoduje znaczne przekroczenie zakładanego budżetu. Z tego też powodu tak ważnym zadaniem stojącym przed projektantami i administratorami systemów jest wybór takich rozwiązań, które sprawią, że docelowi użytkownicy otrzymają produkt maksymalnie dostosowany do ich potrzeb.
Artykuł niniejszy jest wprowadzeniem do kilku technologii, dostarczanych przez firmę Oracle, pozwalających na budowę systemów cechujących się przede wszystkim wysoka niezawodnością, a także – niejako przy okazji stosowania tych rozwiązań – wysoka wydajnością, przy jednoczesnym zmniejszeniu nakładów potrzebnych do osiągnięcia takiego stanu rzeczy.
Oracle RAC i GRID, to technologie pozwalające na zapewnienie ciągłej pracy serwerów bazy danych nawet w przypadku awarii prawie wszystkich maszyn, na których działają instancje danej bazy danych, natomiast DATA GUARD, to mechanizm pozwalający na utrzymywanie zupełnie niezależnej, zapasowej bazy danych, która – w zależności od konfiguracji – jest w stanie przejąć funkcjonalność systemu uszkodzonego przez awarię krytyczną dla systemów RAC i GRID.

## Niezawodność systemów informatycznych

W związku z tym, że większość tworzonych obecnie systemów to systemy projektowane z myślą o zastosowaniach, w których ich dostępności i niezawodność to podstawowe kryteria decydujące o powodzeniu lub klęsce przedsięwzięcia kluczowym elementem jest wyspecyfikowanie dokładnych oczekiwań co do tego aspektu tworzonego staremu informatycznego. Na co dzień przyjmuje się, że dobrą miarą niezawodności takiego systemu jest wartość odpowiadająca procentowej ilości czasu, kiedy to system ma działać poprawnie w stosunku do całego czasu użytkowania systemu. Na ogół za jednostkę miary przyjmuje się rok. W tabeli pierwszej zestawiono przykładowe wymagania czasu dostępności systemów.

| Dostępność systemu w procentach | Akceptowany czas niedostępności systemu (w ciągu roku) |
|---|---|
| 95% | 18 dni |
| 99% | 4 dni |
| 99.9% | 9 godzin |
| 99.99% | 1 godzina |
| 99.999% | 5 minut |

**Tab. 1 Przykładowe wartości miary dostępności systemów informatycznych**

Od wielu lat z powodzeniem stosuje się najprostszą z technik zwiększania niezawodności systemów informatycznych – zwielokrotnianie (redundancja) jego najistotniejszych, lub też najbardziej podatnych na awarię elementów. W tym podejściu dość często przyjmuje się, że tymi częściami systemu są rozwiązania sprzętowe, toteż przeważnie można spotkać się z redundancją na poziomie:

- poszczególnych elementów maszyny – w tym podejściu zwielokrotnia się ilość zasilaczy, urządzeń dyskowych, czy innych komponentów istotnych z punktu widzenia zapewniania właściwego (akceptowalnego) poziomu niezawodnej pracy serwera. Bardzo często zwielokrotnienie uzyskuje się tu metodami sprzętowymi na przykład macierz dyskowa, z kontrolerem wspierającym „mirroring" dysków, chociaż obecnie podobną funkcjonalność można uzyskiwać także programowo – obecnie większość systemów operacyjnych potrafi obsługiwać dyski w sposób zbliżony do sprzętowych kontrolerów macierzy.

- maszyn – w tym podejściu mamy do czynienia z drugim (i kolejnymi) serwerami, które podczas normalnej pracy systemu są wyłączone, bądź też działają w specjalnym trybie synchronizowania się z „głównym" serwerem.
- kompleksowych rozwiązań – kiedy to zwielokrotnieniu ulegają niejednokrotnie całe serwerownie, przy czym lokalizacja serwerów zapasowych może być dowolnie oddalona od głównych maszyn.

Oczywiście, czym wyższym poziom redundancji, tym większa niezawodność systemu informatycznego, ale z drugiej strony większe są też koszty potrzebne na zapewnienie takiego zwielokrotnienia. Dodatkowa wadą takiego rozwiązania jest fakt, iż podstawowym założeniem tego podejścia jest to, że w pełni wykorzystuje się tylko „oryginalny" element systemu, natomiast wszystkie jego kopie czekają „bezużytecznie" na ewentualną awarię swoich odpowiedników. Dodatkowo, gdyby udało się wykorzystać te powielone części do normalnej pracy systemu, to na pewno spowodowałyby one znaczy wzrost jego wydajności.

Powyższe rozumowanie przyświeca także rozwiązaniom firmy Oracle. Zarówno klaster aplikacyjny (RAC), GRID, czy też DATA GUARD to mechanizmy pozwalające na jednoczesne zwiększanie niezawodności systemu i sprawiające, że podczas jego normalnej pracy wydajność takiego systemu znacznie wzrasta. Niebagatelne znaczenie ma tu także argument finansowy – w klasycznym podejściu dwukrotne zwiększenie odporności serwera na wykpienie ewentualnej awarii wiąże się z zakupem identycznej maszyny pełniącej rolę serwera zapasowego. Podobnie ma się sprawa z dwukrotnym zwiększeniem wydajności systemu – najprościej jest dokupić drugi serwer i rozłożyć obciążenie pomiędzy dwie maszyny. Czyli jeżeli chcemy dwukrotnie zwiększyć zarówno wydajność, jak i niezawodność, to jesteśmy zmuszeni do zakupu aż czterech maszyn – dwie aby osiągnąć pożądaną wydajność i dwie stanowiące ich kopie zapasowe. (W rzeczywistości trudno jest jednak spotkać systemy, w których skalowalność wydajności i niezawodności jest liniowa, na ogół jest ona opisana bardziej skomplikowaną funkcja wykładniczą, a co za tym idzie – dwukrotna poprawa jednej z omawianych miar wiąże się nakładami znacznie większymi niż dwukrotne.)

## Niezawodność systemów według Oracle

Firma Oracle zdefiniowała system niezawodny, jako system, posiadający następujące cechy:

- wiarygodność – Oracle zakłada, że zarówno sprzęt wykorzystywany do uruchamiania jej produktów, jak i oprogramowanie na nim zainstalowane (system operacyjny, narzędzia, …) zostały dokładnie wyselekcjonowane i przetestowane, gdyż to także od nich zależy niezawodność produktów Oracle. Dotyczy to wszystkich komponentów architektury systemu informatycznego, począwszy od bazy danych, poprzez warstwy pośrednie oprogramowania, na serwerach aplikacji skończywszy
- odtwarzalność – cecha systemu, pozwalająca na przywrócenie stanu sprzed dowolnej awarii – począwszy od przypadkowego usunięcia wiersza z tabeli, usunięcia całej tabeli, uszkodzenia dowolnego komponentu serwera bazodanowego, poprzez awarie krytyczne. System jest uznawany za odtwarzalny, jeżeli przewidziano procedury definiujące zachowanie w tego typu sytuacjach.
- wykrywanie uszkodzeń – niezawodny system powinien posiadać mechanizmy pozwalające na czesne wykrywanie potencjalnych zagrożeń i informowanie o wykrytych nieprawidłowościach. Pozwala to przede wszystkim na podjęcie działań mających nie dopuścić do zaistnienia awarii, a w razie jej ewentualnego wystąpienia – ułatwić diagnostykę i usuwanie skutków.
- ciągłość pracy – .system niezawodny powinien zapewniać odpowiedni poziom dostępności, zdefiniowany w wymaganiach systemowych. Podobnie zresztą działania administracyjne wykonywane na takim systemie powinny być przeprowadzane w sposób jak najmniej uciążliwy dla jego użytkowników.

System o podwyższonej niezawodności powinien więc:

- być odporny na najczęściej występujące awarie (odporny jest system, a nie jego poszczególne komponenty)
- posiadać mechanizmy monitorowania i ostrzegania przed awariami
- posiadać mechanizmy ułatwiające szybkie odtwarzanie systemu
- pozwalać automatyzować proces odtwarzania systemu po awarii
- posiadać mechanizmy zapobiegające utracie danych nawet w przypadku poważnych awarii

W związku z tym, że utrzymywanie wielu systemów spełniających powyższe wymagania jest bardzo kosztowne, Oracle zaleca:

- rezygnację z utrzymywania dotychczasowych systemów, jeżeli ich koszt ich utrzymania znacznie przekracza koszt utrzymania systemu o podwyższonej niezawodności
- inwestycję w systemy zapewniające pożądany stopień odporności
- stworzenie nowej architektury systemów IT, ułatwiającej zapewnienie wysokiej niezawodności
- przeprojektowanie procesów biznesowych w przedsiębiorstwie
- i wreszcie zatrudnienie odpowiednio wykwalifikowanego personelu do obsługi systemów informatycznych.

Analiza wymagań niezawodności systemu powinna za każdym razem dotyczyć:

- rzeczywistych wymagań narzucanych przez procesy biznesowe – czy faktycznie system musi być dostępny przez 365 dni w roku, i 24 godziny na dobę, czy być może wystarczy dostępność systemu tylko w dni robocze między godzina 8 a 16
- faktyczne straty ponoszone przez przedsiębiorstwo w wyniku przestoju systemu informatycznego (np. kosz godziny przestoju)
- wymagania dotyczące czasu ewentualnego odtwarzania danych, oraz ilości danych, które będzie trzeba odtwarzać, bądź które zostaną być może bezpowrotnie utracone w wyniku awarii

# Zasada działania omawianych rozwiązań

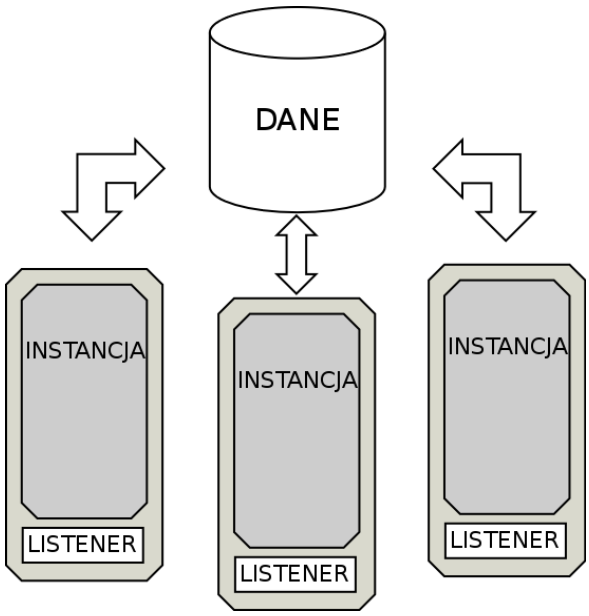
## Real Application Cluster

RAC to technologia pozwalająca na uruchamianie wielu instancji jednej bazy danych na różnych maszynach. Oznacza to, że na jednym zestawie danych, składowanych na jednym urządzeniu dyskowym, może jednocześnie pracować wiele niezależnych serwerów bazodanowych. Naturalna konsekwencją takiego rozwiązania jest więc jego skalowalność – aby zwiększyć wydajność systemu po prostu dołączamy do klastra kolejna maszynę oraz uruchamiamy na niej instancję bazy danych.

W bazie danych Oracle 10g rozwiązania RAC zostały udoskonalone w stosunku do poprzednich wersji, oraz dodano do ich narzędzia pozwalające na kontrolę konfiguracji klastra i pełne zarządzanie wszystkimi aspektami jego pracy. W tej wersji bazy danych RAC stanowi już w pełni samodzielny produkt, do którego uruchomienia i nadzorowania nie potrzeba wykorzystywać żadnych narzędzi firm trzecich.

W architekturze klastra aplikacyjnego mamy do czynienia ze zbiorem niezależnych maszyn współdzielących urządzenie dyskowe do składowania danych (urządzenie sieciowe – NAS, sieć urządzeń dyskowych – SAN, lub urządzenie podłączone do wspólnej magistrali SCSI). O wyborze właściwego typu urządzenia dyskowego decydują planowane wymagania wydajnościowe, lub architektura sprzętowa tworzonego klastra. Kolejnym komponentem klastra aplikacyjnego jest dedykowana sieć komputerowa (interconnect), służąca do wymiany informacji pomiędzy węzłami klastra (równoważenie obciążeń, zapobieganie i łagodzenie skutków awarii, itp.).. Klaster aplikacyjny może tworzyć od 1 do 100 maszyn połączonych wspominaną wyżej dedykowaną siecią komputerową, z których każda ma dostęp do urządzenia dyskowego przechowującego dane. Ma-

szyny tworzące węzły klastra mogą się różnic między sobą – warunkiem jest jednak uruchomienie na nich tego samego systemu operacyjnego i tej samej wersji serwera bazy danych Oracle.




Rys. 1. Ogólna architektura RAC.

RAC posiada zaimplementowany mechanizm wymiany informacji pomiędzy buforami poszczególnych instancji - „cache fusion". Pozwala on na wyeliminowanie zbędnych operacji dyskowych, które ze względu na czas ich wykonania mają duży wpływ na wydajność systemu. Mechanizm ten pozwala na współdzielenie pomiędzy instancjami bloków bazodanowych, które zostały już odczytany przez jedną z instancji i znajduje się w jej buforze.
Rozwiązanie oparte o Real Application Cluster pozwala na wdrożenie dwóch mechanizmów zwiększających niezawodność ich pracy:

- Load Balancig zapewniający równomierne obciążenie wszystkich serwerów w klastrze. Mechanizm ten kontroluje aktualne obciążenie poszczególnych instancji i pozwala na przejecie lub przekierowanie żądań od użytkowników.
- Transparent Application Failover (TAF) przenoszący w sposób całkowicie transparentny sesje użytkowników z serwera, który uległ awarii na inny, sprawny, serwer w klastrze.

### Zalety korzystania z mechanizmu RAC:

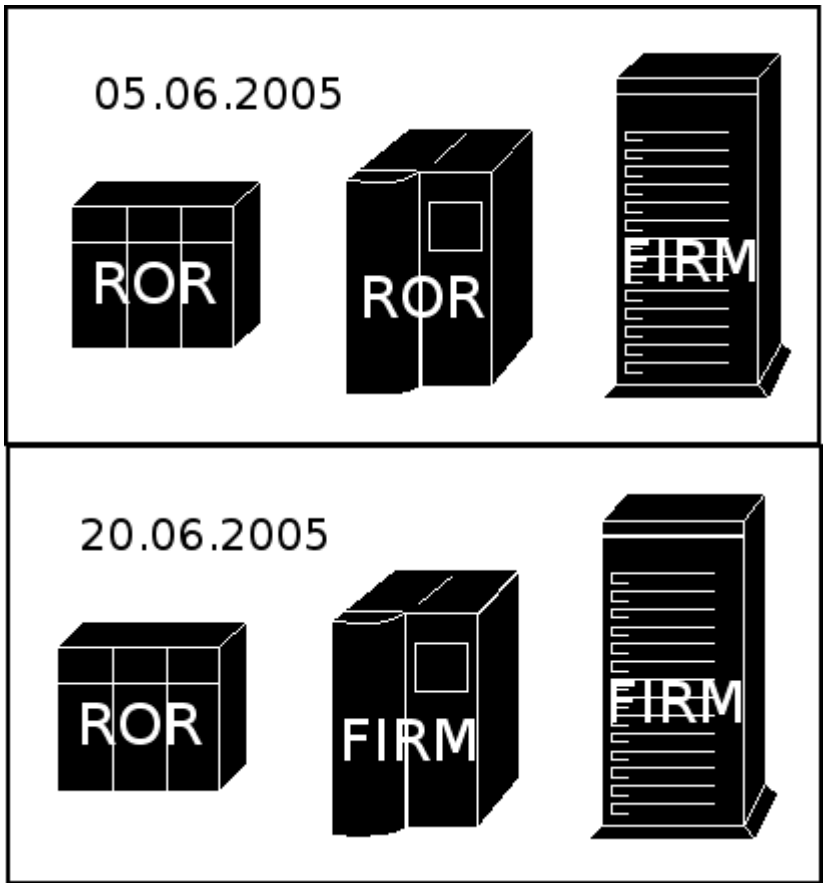
- wykrycie i naprawienie błędów wynikających z awarii węzła lub pojedynczej instancji bazy danych trwa pojedyncze sekundy
- w przypadku wykrycia awarii instancji lub węzła użytkownicy są automatycznie przełączani do innej, działającej instancji (mechanizm niewidoczny dla użytkownika)
- możliwość zaplanowania wyłączenia poszczególnych węzłów lub instancji bez większego wpływu na dostępność całego systemu
- możliwość scentralizowanego zarządzania uaktualnieniami
- możliwość przełączania dostępnych instancji pomiędzy poszczególnymi węzłami pozwalająca na uzyskanie efektu dowolnej skalowalności systemu
- możliwość wykorzystania wspólnych narzędzi do zarządzania bazą danych oraz elementami klastra aplikacyjnego (Oracle Enterprise Manager)

### Wady korzystania z mechanizmu RAC:

- Awaria współdzielonego urządzenia dyskowego sprawie, że wszystkie instancje klastra zostają pozbawione dostępu do danych, a tym samym SA unieruchomione do czasu odtworzenia bazy danych.

## GRID

Grid jest rozwinięciem idei klastra aplikacyjnego. Podobnie, jak to miało miejsce przy mechanizmie RAC, tak i tutaj z daną bazą danych związanych jest wiele działających instancji. Podstawową różnicą jest tutaj sposób widzenia tego środowiska przez użytkownika końcowego – dla niego wszystkie te instancje bazy danych tworzą jedną wydajną i niezawodną wirtualną instancję bazy danych. Z punktu widzenia administracji takim środowiskiem zagadnienie to jest bardziej złożone. Działający Grid korzysta z następujących mechanizmów dostępnych w bazie danych Oracle 10g: automatyczne strojenie instancji, możliwość automatycznej alokacji instancji i możliwość migracji danych pomiędzy działającymi instancjami. Dzięki temu podsystem zarządzający Gridem na bieżąco śledzi aktualne obciążenie poszczególnych instancji i serwerów bazodanowych i może podejmować decyzje o zmianie parametrów pracy dowolnej instancji, o wyłączeniu części instancji obsługujących bazę danych o niewielkim obciążeniu, czy tez o włączeniu instancji w przypadku wzrostu obciążenia bazy danych.
Przykładem może być tu bank posiadający trzy serwery bazodanowe, i dwie aplikacje: **ROR** – do obsługi klientów indywidualnych i **FIRM** do obsługi przedsiębiorców. Na początku każdego miesiąca w banku tym można zaobserwować znaczy ruch na rachunkach klientów indywidualnych (pensje, opłaty, wypłaty), natomiast w okolicach dwudziestego przedsiębiorcy rozliczają się z podatków. W tych „gorących" okresach odpowiednie aplikacje wymagają odpowiednio dużej mocy obliczeniowej, natomiast poza nimi zapotrzebowanie na moc serwera jest raczej niewielkie. Zanim wdrożono w tym banku system zarządzany przez Grid to administratorzy musieli podejmować decyzje o przełączaniu odpowiednich instancji pomiędzy serwerami. Robili to znając historyczne obciążenie serwerów w poprzednich miesiącach. Czasami jednak zdarzało się, że ich decyzje nie były do końca trafne i w czasie kiedy najsilniejszy serwer nie był w pełni wykorzystany – dwie mniejsze maszyny pracowały na granicy swoich możliwości. Obecnie to zadanie zostało przerzucone na warstwę zarządzającą Gridem, która analizuje obciążenie każdego z serwerów i instancji i podejmuje decyzje o tym, że właśnie nadszedł odpowiedni moment na rekonfiguracje systemu. Na poniższym rysunku wyraźnie pokazano, że zastosowanie mechanizmu Grid pozwoliło uprościć i zoptymalizować proces zarządzania środowiskiem o zmiennej charakterystyce obciążenia. Gdyby nie wdrożono tego rozwiązania, to pewnie w niedługim czasie bank ten zakupiłby odpowiednią ilość serwerów, zdolną do obsługi każdej z tych aplikacji w okresie wzmożonego zapotrzebowania na zasoby systemowe, bez konieczności ciągłego „ręcznego" optymalizowania pracy tego systemu.




Rys. 2. Przykład zarządzania instancjami bazy danych przez Grid.

Mechanizmy zarządzania instancjami są dodatkowo wspierane przez mechanizmy migracji danych (Streams, Data Pump, Transpotrable Tablespaces), pozwalających na przesyłanie i składowanie danych tam, gdzie w danym momencie są one najintensywniej wykorzystywane.

### Zalety korzystania z mechanizmu GRID:

- Grid posiada wszystkie pozytywne cechy systemu opartego o klastry aplikacji
- Pozwala na dynamiczną kontrolę obciążenia każdej z maszyn
- Pozwala na migrację danych bliżej miejsca ich przetwarzania
- Odporny na ewentualną awarię jednego z urządzeń dyskowych (dzięki wykorzystaniu mechanizmu ASM – Automatic Storage Management)

### Wady korzystania z mechanizmu GRID:

- Mechanizm stosunkowo młody i z tego powodu dość niechętnie stosowany przez projektantów systemów i administratorów

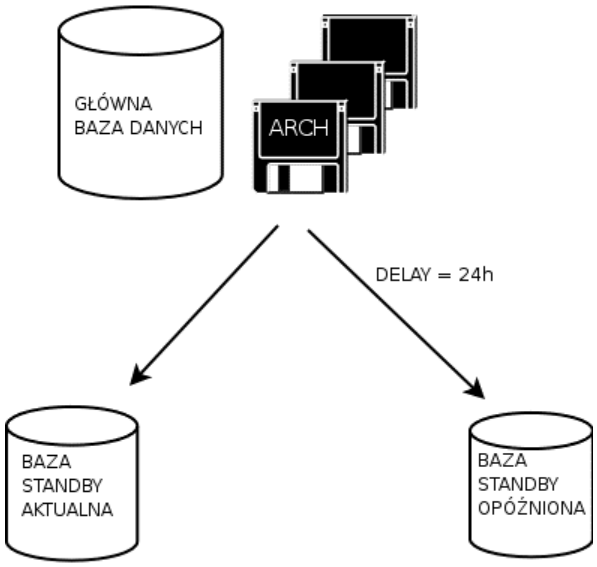
## DATA GUARD

Funkcja Data Guard, została wprowadzona po raz pierwszy w bazie danych Oracle8i i zapewnia ochronę przed awariami, utrzymując zapasowe bazy danych w stanie gotowości. W 10g funkcję tę rozbudowano, wprowadzając opcjonalną możliwość kompresji i szyfrowania zapisów w dzienniku oraz obsługę dodatkowych typów danych w trybie logicznych baz standby (Apply SQL).
Problem instalowania uaktualnień, jednej z głównych przyczyn planowanych przestojów, rozwiązano poprzez dodanie obsługi tego procesu w czasie pracy systemu (rolling upgrade). Możliwość ta obejmuje zarówno sprzęt i system operacyjny, jak i wersje bazy danych

W rzeczywistości Oracle Data Guard jest kompletnym zestawem narzędzi pozwalających na tworzenie i zarządzanie specjalnych kopii bazy danych – zwanych bazami standby. Bazy zapasowe są to regularnie uaktualniane kopie bazy głównej, gotowe w każdej chwili do przejęcia jej funkcjonalności. Sytuacja taka może mieć miejsce albo na skutek awarii głównej bazy danych, albo w wyniku jej planowego wyłączenia. Funkcjonalność taka sprawia, że czas niedostępności bazy danych jest bardzo krótki – na tyle, że cześć użytkowników może się nie zorientować, że takie przełączenia miało miejsce. Bazę standby można też traktować jako element strategii backupu głównej bazy danych.
Działający system Data Guard składa się z jednej głównej bazy danych i przynajmniej jednej bazy typu standby. Bazy standby mogą odzwierciedlać fizyczną, bądź logiczna strukturę bazy źródłowej.
W przypadku tak zwanej fizycznej bazy standby – serwer główny przesyła do takiej bazy danych zarchiwizowane pliki dziennika powtórzeń, a ta, działając w trybie ciągłego odtwarzania wykonuje operacje sprawiające, że struktura fizyczna każdego z bloków dyskowych bazy docelowej jest identyczna z blokami bazy źródłowej. W przypadku logicznej bazy standby – główna baza danych także przesyła do docelowej lokalizacji zarchiwizowane pliki dziennika powtórzeń, tylko tym razem przy pomocy narzędzia Log Miner z tych plików jest wydobywana treść poleceń SQL, które wprowadzały zmiany w bazie źródłowej. Dość często spotyka się takie konfiguracje bazy standby, w których zmiany z bazy źródłowej są wprowadzane do niej z celowym opóźnieniem (rzędu kilku godzin, czy jednego dnia). Rozwiązanie takie, choć z jednej strony znacznie wydłuża czas przełączenia systemu po awarii głównej bazy danych ma jednak tę zaletę, że w razie wystąpienia błędu użytkownika (przypadkowa modyfikacja /usunięcie krotki) w każdej chwili jest dostępna aktywna kopia danych z przed zadanego okresu.

### Fizyczna baza standby

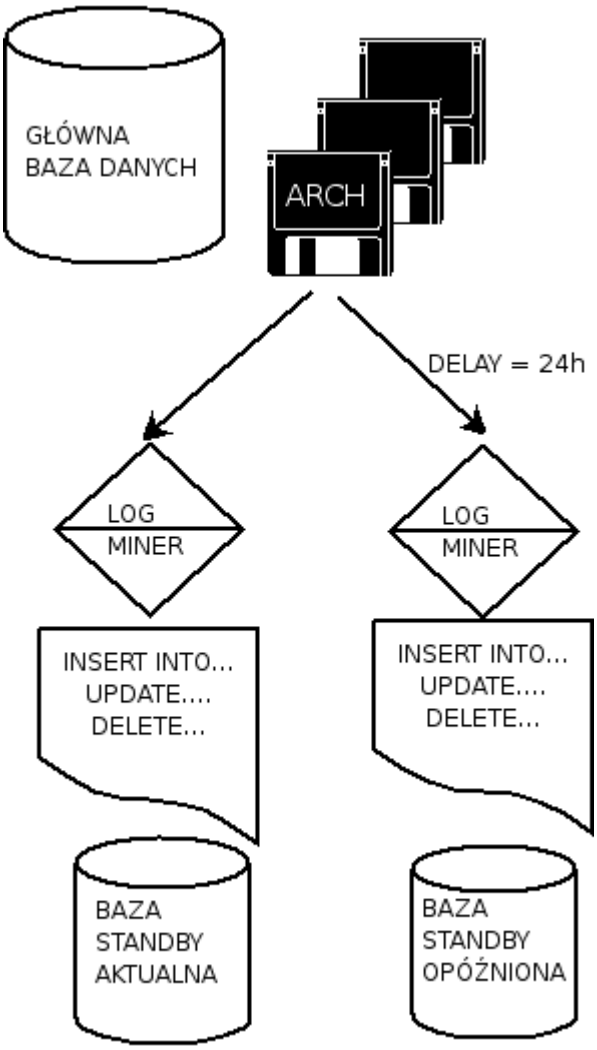
Fizyczną bazę standby tworzy się z kopii archiwalnej bazy źródłowej, uruchamiając ją w trybie odtwarzania po awarii i przesyłając do niej zarchiwizowane pliku dziennika powtórzeń celem przetworzenia ich w taki sposób, jak by to były pliki właśnie tej instancji (standby). Z tego powodu fizyczna struktura takiej bazy jest identyczna ze struktura bazy źródłowej.
Istnieje możliwość otwarcia takiej bazy w trybie „tylko do odczytu”.




Rys. 3. Idea działania fizycznej bazy standby.

### Logiczna baza standby

Logiczna baza danych standby powstaje w wyniku aplikowania do niej operacji SQL wydobywanych przy pomocy narzędzia Log Miner z przesyłanych logów transakcyjnych głównej bazy danych. Oznacza to, że jej zawartość jest identyczna z zawartością głównej bazy danych (przynajmniej z dokładnością do powtórzenia wykonywanych w niej operacji). Jednocześnie taka baza danych standby może być otwarta w trybie pozwalającym na odczyt i modyfikacje danych. Oznacza to, że jednak z pewnych powodów zawartość tej bazy danych ulegnie zmianie w stosunku do danych źródłowych.




Rys. 4. Idea działania logicznej bazy standby.

Tego typu bazy danych standby oprócz pełnienia funkcji serwerów zapasowych – redundantnych w stosunku do źródłowej bazy danych pozwalają na implementowanie rozwiązań typu hurtownie (magazyny) danych. Możliwość zapisywania w takiej bazie danych użytkowników może być wykorzystana w celu stworzenia metadanych opisujących dane pochodzące z bazy źródłowej.
Bazy tworzące architekturę Data Guard mogą znajdować się w dowolnych lokalizacjach geograficznych, gwarantujących im jedynie niezawodną transmisje zarchiwizowanych plików dziennika powtórzeń. Takie rozproszenie chroni bazę danych przed skutkami ewentualnych lokalnych kataklizmów typu trzęsienie ziemi, powódź, czy pożar.

### Zalety korzystania z mechanizmu DATA GUARD:

- Upraszcza procesy odtwarzania po awarii, jest jednym z mechanizmów ochrony (archiwizacji) danych oraz zapewnia wysoka dostępność tak skonfigurowanego systemu bazodanowego. System chroniony w ten sposób jest odporny zarówno na planowane, jak i nieplanowane wyłączenia głównej bazy danych.
- Data Guard zapewnia kompleksowa ochronę danych składowanych w bazie danych. Istnieją sposoby zapewniania, że nawet w przypadku awarii centralnej bazy danych nie nastąpi utrata żadnych informacji. Jest to o tyle istotne, ze fizyczne uszkodzenie plików z danymi w bazie źródłowej nie propaguje się do baz zapasowych. Także błędy użytkownika mogą być w prosty sposób wykrywane. Dodatkowo mechanizmy Data Guard dokonują kontroli i walidacjo każdego dostarczonego zarchiwizowanego pliku dziennika powtórzeń.
- Efektywne wykorzystanie zasobów – zapasowe bazy danych aktualizowane przy pomocy zarchiwizowanych plików dziennika powtórzeń mogą być otwarte przynajmniej w trybie umożliwiającym odczyt zawartych w nich danych. Oznacza to, że można je z powodzeniem wykorzystywać jako bazy danych, na których są wykonywane różnego rodzaju analizy. Pozwala to odciążyć system źródłowy, zwłaszcza, biorąc pod uwagę to, że przeważnie zapytania analityczne SA zapytaniami, których wykonanie angażuję znaczne ilości pamięci operacyjnej serwera, czasu procesora i innych zasobów serwera. Możliwość otwarcia logicznej bazy standby w trybie do odczytu i zapisu jeszcze zwiększa jej użyteczność dla tego typu systemów, gdyż umożliwia ona tworzenie dodatkowych struktur, takich jak indeksy, czy perspektywy materializowane.
- Elastyczność pozwalająca na połączenie mechanizmów zapewniających wzrost dostępności systemu bazodanowego z mechanizmami zwiększającymi jego wydajność przy stosunkowo niewielkich nakładach potrzebnych na poprawę wartości obu tych cech.
- Automatyczne wykrywanie problemów z połączeniem między komponentami architektury – Data Guard potrafi wykryć i prawidłowo zdiagnozować brak połączenia pomiędzy bazą źródłową a docelowymi. W takiej sytuacji nie mam możliwości przesłania zarchiwizowanych plików dziennika powtórzeń bezpośrednio do baz docelowych. Pliki te są odpowiednio buforowane i po nawiązaniu połączenia trafiają w końcu do bazy docelowej. Powtórna synchronizacja z wykorzystaniem tych plików logu następuje automatycznie i nie wymaga interwencji administratora systemu.
- Scentralizowane i proste zarządzanie – Data Guard posiada graficzny interfejs oraz narzędzia wywoływane z linii poleceń powalające na automatyzację zarządzania wszystkimi bazami wchodzącymi w skład tej architektury. Narzędzia te pozwalają tez na niezależne monitorowanie pracy każdej z tych baz danych.
- Pełna integracja z mechanizmami bazy danych Oracle 10g

### Wady korzystania z mechanizmu DATA GUARD:

- Tylko jedna baza danych jest bazą w pełni otwartą, umożliwiająca wykonywanie operacji na danych. Ewentualne zmiany zawartości logicznej bazy danych standby nie zostaną przesłane do głównej bazy danych.
- Istnieje pewne opóźnienie pomiędzy bazą źródłową a docelowymi bazami danych i awaria źródłowej bazy danych może doprowadzić do utraty niewielkiej ilości informacji (od administratora systemu zależy jak wielkie będzie to opóźnienie).

## Inne mechanizmy

### Oracle Enterprise Manager

OEM jest zintegrowana konsola pozwalającą skupić w jednym miejscu wszystkie aspekty zarządzania produktami Oracle dostępnymi w przedsiębiorstwie. Działanie tego narzędzia opiera się na wynikach pracy szeregu *agentów* dedykowanych dla każdego produktu Oracle, składowanych w centralnym repozytorium OEM. Dostarczana wraz z baza danych Oracle 10g wersja OEM Grid

Control pozwala na zarządzanie miedzy innymi bazą danych, serwerami aplikacji, środowiskiem sieciowym, serwerem aplikacji, także, co nas najbardziej interesuje instancjami klastra aplikacyjnego oraz GRIDu.
OEM powala także na zdefiniowanie szeregu progów, dla poszczególnych aspektów działania kontrolowanych komponentów i podejmowania odpowiednich akcji (np. powiadamianie administratora) po przekroczeniu wartości krytycznych dla tych parametrów.

### SQL*Net

Systemy o podwyższonej niezawodności nie miałyby szansy powodzenia, gdyby nie były wspierane przez pozostałe komponenty architektury Oracle. Dotyczy to przede wszystkim środowiska sieciowego SQL*Net, które w swoich założeniach pozwala na taka konfiguracje systemu, aby ukryć wszystkie omawiane mechanizmy przed użytkownikiem końcowym. W większości wypadków wystarczy, że użytkownik ten wie, iż ma się połączyć z baza danych, a o wybór właściwej instancji RAC, czy GRID lub tez bazy w mechanizmie DATA GUARD zatroszczy się właśnie SQL*Net. Środowisko to pozwala na elastyczny wybór sposobów konfiguracji oraz podejmowania decyzji co do dołączenia się do konkretniej działającej instancji. W najprostszym rozwiązaniu wystarczą odpowiednie wpisy w plikach konfiguracyjnych klienta bazy danych (*tnsnames.ora*):

```
baza.com=
  (DESCRIPTION=
    (ADDRESS_LIST=
      (ADDRESS=(PROTOCOL=tcp)(HOST=baza1-server)(PORT=1521))
      (ADDRESS=(PROTOCOL=tcp)(HOST=baza2-server)(PORT=1521)))
    (CONNECT_DATA=
    (SERVICE_NAME=baza.com)))
```

W przypadku rozwiązań bardziej skomplikowanych można posłużyć się mechanizmem connection managera, czy też mechanizmami równoważenia obciążeń zaimplementowanymi w listenerach baz danych.

### RMAN

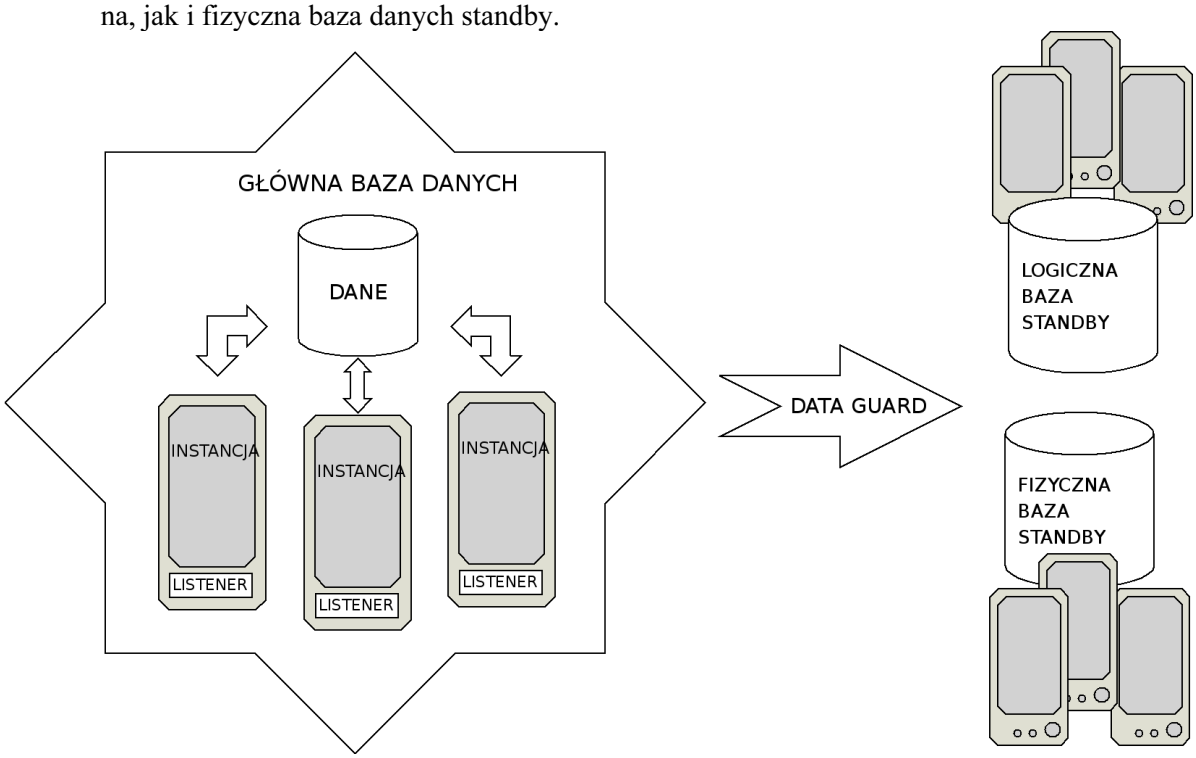
Recovery Manager – narzędzie służące do zarządzania i automatyzacji procesu archiwizacjo bazy danych, jej ewentualnego odtwarzania po awarii oraz zarządzania zapasowymi nośnikami danych.
Przykładem wykorzystania tego narzędzia w środowiskach o podwyższonej niezawodności jest możliwość wykonywania szeregu pełnych i przyrostowych kopii zapasowych bazy danych zarówno otwartej przez instancje, jak i zamkniętej. Dodatkowo można to narzędzie skonfigurować w taki sposób, aby optymalizowało ono ewentualne odtwarzanie bazy danych po awarii, tak planując strategię archiwizacji, aby estymowany proces odtwarzania był nie dłuższy niż pewien założony czas.
Szczególnie ciekawe wydaje się też wykorzystanie tego narzędzia do archiwizowania baz standby zamiast archiwizacji bazy źródłowej w mechanizmie Data Guard. Pozwala to na uzyskanie w pełni funkcjonalnej kopii bezpieczeństwa bez konieczności wyłączania, czy tez dodatkowego obciążania głównego serwera bazy danych.

## Dobra praktyka projektowania systemów o podwyższonej niezawodności

Firma Oracle definiuje szereg klas systemów o podwyższonej niezawodności:

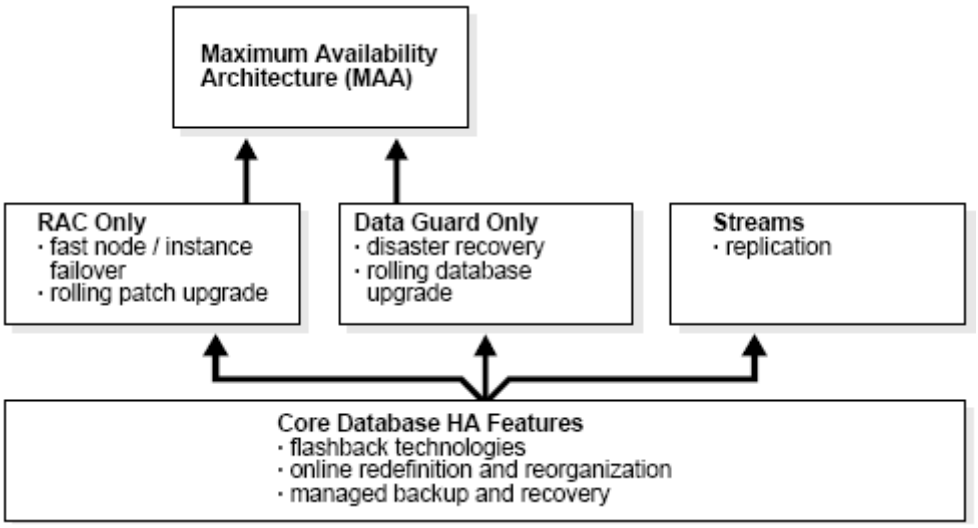
- **Database only** – to system oparty o jedną instancję bazy danych – uruchomioną na jednej maszynie. Niezawodność systemu jest tu zapewniana przez podstawowe mechanizmy bazy danych Oracle 10g – flashback (pozwalający na usuwanie błędów użytkowników bazy danych), mechanizmy automatycznego zarządzania bazą danych, RMAN (mechanizm zarządzania archiwizowaniem i odtwarzaniem bazy danych)
- **RAC only** – system korzystający ze wszystkich możliwości oferowanych przez klaster aplikacyjny, a tym samym odporny na ewentualną awarię instancji, czy węzła

- **Data Guard only** – system korzystający z mechanizmu DATA GUARD, skonfigurowanego w ten sposób, że ewentualna awaria głównej bazy danych powoduje automatyczne przełączenie do pracy z rezerwową bazą danych.
- **Maximum Availability Architecture (MAA)** – Architektura systemu składa się z dwóch części (podobnie jak to ma miejsce w DATA GUARD), przy czym przynajmniej główną bazą danych jest baza danych działająca w architekturze klastra aplikacyjnego. W części zapasowej systemu skonfigurowanego jako MAA powinna znajdować się zarówno logiczna, jak i fizyczna baza danych standby.




Rys. 5. Architektura systemu MAA.

- **Streams** – Architektura systemu oparta jest o dwie (lub więcej) niezależne bazy danych w których za replikację danych odpowiada mechanizm Oracle Streams. Oznacza to, że wszystkie bazy danych umożliwiają modyfikowanie danych a transakcje w każdej z nich są propagowane do pozostałych. W tym rozwiązaniu dopuszczalne są systemy działające na różnych platformach sprzętowych, czy programowych.




Rys. 6. Zależności pomiędzy systemami o podwyższonej niezawodności.

Wybór właściwej architektury systemu następuje po dokładnej analizie wymagań. W dużym uproszczeniu można przedstawić taka analizę, jako próbę odpowiedzi na następujące pytania:

- Czy system ma być dostępny 24h na dobę, 7 dni w tygodniu, 365 dni w roku?
- Czy będą wykonywane zaplanowane zadania administracyjne wymagające wyłączenia systemu?
- Czy jest potrzebne zapewnienie pełnej odtwarzalności danych po awarii?

Poniższy rysunek przedstawia sposób wyboru właściwej architektury o podwyższonej niezawodności:

| Answer 1 (Local Site HA) | Answer 2 (Rolling Maintenance) | Answer 3 (Disaster Recovery) | Recommended Architecture |
|---|---|---|---|
| No | No | No | Database only |
| Yes | No | No | RAC only |
| No | Yes | No | Data Guard only |
| No | No | Yes | Data Guard only |
| Yes | Yes | No | MAA |
| Yes | No | Yes | MAA |
| No | Yes | Yes | Data Guard only |
| Yes | Yes | Yes | MAA |

Rys. 7. Wybór właściwej architektury o podwyższonej niezawodności.

## Planowanie działań związanych z utrzymaniem systemów o podwyższonej niezawodności

Wszelkie działania wykonywane na systemach MAA powinny być dokładnie zaplanowane i opisane w odpowiednich procedurach. W szczególności dotyczy to takich działań, jak:

- planowanie i zapewnianie odpowiedniego poziomu obsługi – zgodnie z wcześniej ustalonym SLA
- planowanie, dokumentowani i na koniec wykonywanie wszelkich działań mających na celu zwiększenie (a zwłaszcza obniżenie) poziomu niezawodności

- planowanie i wykonywanie archiwizacji systemów o podwyższonej niezawodności (a także okresowe wykonywanie testów odtwarzania systemu po awarii!!!)
- weryfikacja i ewentualna modyfikacja procedur postępowania w przypadku wystąpienia awarii
- planowanie działań administracyjnych i powiadamianie o nich wszystkich zainteresowanych użytkowników
- szkolenia pracowników
- prowadzenia dokumentacji wszelkich zdarzeń mogących wpłynąć na poziom niezawodności systemu
- fizyczne i logiczne zabezpieczanie systemu o podwyższonej niezawodności

## Konfigurowanie systemów o podwyższonej niezawodności

System o podwyższonej niezawodności powinien składać się z komponentów (zarówno sprzęt, jak i oprogramowanie) spełniających następujące założenia:

- wszystkie komponenty istotne z punktu widzenia integralności systemu powinny być zwielokrotnione
- należy używać komponentów o dużej skali integracji i najlepiej jednolitych w ramach całego systemu (np. takie same karty sieciowe, oraz takie same systemy operacyjne z zainstalowanymi tymi samymi poprawkami) – co pozwala na wyeliminowanie szeregu problemów związanych z ich ewentualną niekompatybilnością, czy też wadliwie działającymi interfejsami pomiędzy komponentami
- system powinien umożliwiać ewentualna późniejszą rozbudowę i modernizację
- komponenty powinny umożliwiać zdalne zarządzanie zarówno poszczególnymi składowymi systemu, jak i całym systemem, dodatkowo należy preferować te komponenty, które pozwalają na automatyczne wykrywanie zbliżającej się awarii
- poszczególne składniki systemu powinny być od siebie odseparowane w taki sposób, aby awaria jednego powodowała możliwie małe straty w innych jego komponentach (np. rozdzielanie oprogramowania od właściwych danych)
- i przede wszystkim – użyte komponenty powinny zostać zatwierdzone jako „poprawnie współpracujące" z oprogramowaniem Oracle

## Konfiguracja bazy danych Oracle wspierająca podwyższona niezawodność

Podobnie jak to miało miejsce w przypadku wyboru sprzętu i dodatkowego oprogramowania wspierającego architekturę o podwyższonej niezawodności, tak i baza danych Oracle powinna być odpowiednio skonfigurowana tak, aby móc w pełni korzystać możliwości oferowanych przez ta architekturę. Podstawowe zalecenia dotyczą:

- wykorzystywania przynajmniej dwóch kopii pliku kontrolnego, oraz plików dziennika powtórzeń przełączających się przynajmniej raz na około 20 minut (autor zaleca wręcz tworzenie po jednej kopii tych plików na każdym dostępnym urządzeniu dyskowy)
- ustawienia odpowiednio dużej wartości parametru konfiguracyjnego: CONTROL_FILE_RECORD_KEEP_TIME –co najmniej 3 razy większej niż częstotliwość wykonywanie kipi zapasowej bazy danych
- włączenie trybu: ARCHIVELOG
- włączenie mechanizmu zliczania sum kontrolnych dla bloków danych
- włączenie logowania punktów kontrolnych a pliku Alert Log
- wykorzystanie parametru Fast-Start Checkpointing w celu kontrolowania czasu ewentualnego automatycznego odtwarzania bazy danych
- zbierania i analizowania statystyk dotyczących wydajności bazy danych
- wykorzystania mechanizmów: Automatic Undo Management, Locally Managed Tablespaces, Automatic Segment Space Management

- właściwej konfiguracji tymczasowych przestrzeni tabel, a także wyspecyfikowania domyślnej tymczasowej przestrzeni tabel
- włączenia mechanizmu Flash Recovery, oraz Flashback Database
- wykorzystania narzędzia Resource Manager
- wykorzystywania dynamicznego pliku konfiguracyjnego – spfile
- wykorzystanie mechanizmu dynamicznej rejestracji instancji baz danych do działających procesów nasłuchu (listener)

## Podsumowanie

Przedstawiony powyżej krótki opis mechanizmów bazy danych Oracle 10g – tych najbardziej istotnych z punktu widzenia procesu zapewniania odpowiedniej niezawodności systemu – powinien wprowadzić czytelnika w skomplikowane zagadnienia projektowania i administrowania systemów o zwiększonej odporności na awarie. Większość poruszanych tu zagadnień jest dokładniej omówiona w dokumentacji dostarczanej wraz z oprogramowaniem bazy danych. Szczególnej uwadze czytelnika polecam dokument: **High Availability Architecture and Best Practices 10g Release 1 (10.1).** Dodatkowo zachęcam do zapoznania się z szeregiem publikowanych przez Oracle dokumentów mających w tytule wspólny człon: „Best Practices".

Wszystkich prezentowane tu mechanizmy pozwalające na zwiększenia niezawodności i niejako przy okazji wydajności systemu nie wymagają od ich użytkowników konieczności wymiany serwerów na nowsze, silniejsze, bardziej niezawodne maszyny. Wystarczy tylko dodać kolejny komputer do klastra aplikacyjnego, czy też odpowiednio skonfigurować GRID ewentualnie wykorzystać możliwości istniejącego już w firmie mechanizmu DATA GUARD.

Oprogramowanie Oracle stwarza złudzenie istnienia jednego, ogromnego, niezawodnego superkomputera, kiedy w rzeczywistości jednak za każdym razem mamy pewien zbiór mniejszych, być może także zawodnych maszyn.

Na koniec pragnę jeszcze raz zaznaczyć, że każdy system jest tak „powolny", jak jego najwolniejszy element i tak podatny na awarię, jak jego najbardziej zawodna część. Stosowanie omówionych tu rozwiązań firmy Oracle z pewnością pozwala zwiększać wydajność i niezawodność systemów informatycznych (przy zachowaniu umiarkowanych kosztów tego wzrostu), ale tak naprawdę kluczowe znaczenie ma tu zawsze właściwe rozpoznanie wymagań użytkowników tego systemu i dobranie rozwiązań właściwych dla danego problemu.

## Literatura

Dokumentacja Oracle

- **High Availability Architecture and Best Practices 10g Release 1 (10.1)**
- Oracle® Real Application Clusters Administrator's Guide 10g Release 1 (10.1)
- Oracle® Real Application Clusters Deployment and Performance Guide 10g Release 1 (10.1)

Oracle Technical White Paper:

- Best Practices for Using XA with RAC
- Configuring Enterprise Manager for High Availability
- Oracle9i Data Guard: Primary Site and Network Configuration Best Practices
- Oracle Database 10g Services
- Oracle RAC Best Practices on Linux
- Oracle RAC 10g Overview
- Oracle Real Application Clusters 10g