IX Konferencja PLOUG
Kościelisko
Październik 2003

Grid Computing: wprowadzenie do przetwarzania danych

Maciej Zakrzewicz

Politechnika Poznańska

Przetwarzanie siatkowe (grid computing) jest nowoczesną koncepcją wykorzystania rozproszonych zasobów komputerowych jako logicznej jednostki przetwarzania danych. Od niedawna pojawiają się na rynku coraz liczniejsze propozycje rozwiązań systemowych opartych na tej idei. Celem niniejszego artykułu jest omówienie podstaw Grid Computing, przedstawienie kierunków standaryzacji oraz charakterystyka jego zastosowań w dziedzinie systemów baz danych, w szczególności na tle funkcjonalności systemu zarządzania bazą danych Oracle10*G*.

Informacja o autorze:
dr inż. Maciej Zakrzewicz jest pracownikiem naukowym Instytutu Informatyki Politechniki Poznańskiej, prezesem Stowarzyszenia Polskiej Grupy Użytkowników Systemu Oracle. Specjalista w dziedzinach systemów baz/hurtowni danych, eksploracji danych oraz rozwiązań internetowych. Dla krajowych i zagranicznych (RFN, Wielka Brytania, USA) uczelni i przedsiębiorstw prowadzi wykłady z zakresu projektowania i budowy systemów informatycznych.

1. Wprowadzenie

Ogromny sukces zastosowań Internetu w dziedzinie globalnego upowszechniania informacji spowodował wzrost zainteresowania wykorzystaniem jego infrastruktury jako platformy dla realizacji aplikacji rozproszonych. W ciągu ostatnich kilkunastu lat zaproponowano szereg technologii, umożliwiających implementację komponentowych aplikacji rozproszonych, których elementy mogą być rozlokowane na wielu heterogenicznych węzłach połączonych siecią typu internet lub intranet: CORBA, DCOM, Enterprise JavaBeans (EJB), WebServices. Aplikacje realizowane z pomocą tych technologii składają się zwykle z pojedynczego modułu klienta, pracującego na komputerze użytkownika końcowego i odpowiadającego za obsługę graficznego interfejsu użytkownika oraz za sterowanie przetwarzaniem, oraz z wielu modułów zdalnych, osadzonych na serwerach aplikacji, odpowiadających za realizację właściwego przetwarzania danych. Charakterystyczne różnice pomiędzy wspomnianymi technologiami sprowadzają się do zakresu wspieranych języków programowania (np. EJB współpracuje wyłącznie z językiem Java), platform systemu operacyjnego (np. DCOM przeznaczony jest dla środowisk Microsoft Windows) oraz stosowanych protokołów komunikacji klienta z modułami zdalnymi (np. HTTP w WebServices, nastomiast CORBA i EJB korzystają z IIOP).

Pewnego rodzaju wadą wyżej wymienionych technologii budowy rozproszonych aplikacji komponentowych jest sztywność powiązania zdalnych modułów ze sprzętem lub serwerami aplikacji, do których zostały przydzielone. Osoby wdrażające rozproszone aplikacje komponentowe podejmują wiążące decyzje o fizycznej lokalizacji wszystkich elementów aplikacji. Działanie takie powoduje, że każdy system aplikacyjny posługuje się zbiorem dedykowanych komputerów, nazywanym często „wyspą przetwarzania danych" (island of computing), a zasoby sprzętowe przedsiębiorstwa są podzielone (najczęściej rozłącznie) pomiędzy posiadane systemy aplikacyjne. W codziennej eksploatacji może się zdarzać, że dochodzi do przeciążenia zasobów wykorzystywanych przez jeden system aplikacyjny, podczas gdy pozostałe systemy nie są w ogóle wykorzystywane przez użytkowników.

Problem sztywnego przydziału oprogramowania do sprzętu może zostać złagodzony dzięki zastosowaniu koncepcji przetwarzania Grid Computing (przetwarzanie siatkowe, przetwarzanie gridowe, przetwarzanie sieciowe). Grid Computing postuluje, aby traktować wszystkie posiadane zasoby sprzętowe jako jeden wielki komputer wirtualny, zdolny wykonywać wszystkie dotychczasowe aplikacje, automatycznie alokując je do poszczególnych maszyn w taki sposób, aby równoważyć obciążenie maszyn oraz redukować wpływ awarii na dostępność systemów aplikacji. Najczęściej mówi się o Computational Grids [FK99], które wirtualizują zasoby obliczeniowe (procesory, pamięci operacyjne) oraz o Data Grids [CFK+01], które wirtualizują pamięć masową (dyski twarde, napędy CD/DVD, streamery).

Często pisze się o Grid Computing w kontekście E-utilities, gdzie zwraca się uwagę na to, że zasoby służące do przechowywania i przetwarzania danych powinny być traktowane podobnie jak energia elektryczna, gaz i woda – użytkownik nie musi być świadomy tego, skąd pochodzą i jaka jest architektura ich sieci dystrybucyjnych, natomiast powinien oczekiwać gwarancji dostaw oraz zgodności parametrów z ustalonymi standardami (napięciem sieciowym, kalorycznością lub ciśnieniem). Użytkownik wykorzystywałby E-utilities powierzając do wykonania swoje zadania przetwarzania danych i otrzymując odpowiedzi w czasie zgodnym z ustaloną umową świadczenia usługi. Komunikacja ze śro-

dowiskiem E-utilities odbywałaby się poprzez ustalone standardowe interfejsy, natomiast usługi przetwarzania danych byłyby świadczone przez podmioty gospodarcze na zasadach podobnych do dzisiejszych Application Service Providers.

Artykuł ten stanowi wprowadzenie do dziedziny przetwarzania Grid Computing. Struktura artykułu jest następująca. W rozdziale 2 scharakteryzowano podstawowe własności przetwarzania typu Grid Computing. Rozdział 3 omawia pierwszą dużą realizację środowiska Grid Computing dla potrzeb analizy sygnałów astronomicznych. W rozdziale 4 dokonano przeglądu własności narzędzi pozwalających budować środowiska Grid Computing. Rozdział 5 przedstawia rozszerzenia funkcjonalności systemu zarządzania bazą danych Oracle 10g, umożliwiające konstrukcję środowisk typu Enterprise Grid Computing. Rozdział 6 zawiera podsumowanie.

2. Grid Computing

Przetwarzanie Grid Computing jest nową generacją przetwarzania rozproszonego, polegającą połączeniu dużej ilości heterogenicznych zasobów komputerowych (węzłów) w celu stworzenia iluzji posiadania komputera wirtualnego o bardzo dużej mocy obliczeniowej [FKN+02][FKT+01].

Najprostszym przykładem wykorzystania środowiska Grid Computing może być uruchamianie standardowej aplikacji na komputerze innym niż zwykle. Jeżeli komputer, na którym aplikacja jest zwykle uruchamiana, zostanie przeciążony, wtedy aplikacja może zostać uruchomiona na innym, nieobciążonym w danym momencie komputerze. Aby taki scenariusz mógł zostać zrealizowany, spełnione muszą być dwa warunki: (1) natura aplikacji musi pozwalać na wykonanie zdalne, np. z punktu widzenia interakcji z użytkownikiem i (2) zdalny komputer musi spełniać wymagania sprzętowe i systemowe stawiane przez uruchamianą aplikację.

W wielu przedsiębiorstwach występują ogromne ilości niewykorzystanych zasobów obliczeniowych. Większość stacji roboczych jest obciążana zaledwie w 5%. W niektórych przypadkach niedociążone są również serwery – dzieje się tak bądź za sprawą nadmiaru mocy w stosunku do rzeczywistego obciążenia, bądź ze względu na czasowe fluktuacje aktywności użytkowników. Budowa środowiska Grid Computing umożliwia wykorzystanie niedociążonych zasobów jako wsparcia dla tych zasobów komputerowych, które są chwilowo bądź permanentnie przeciążone. Reguły alokacji aplikacji lub zadań na węzłach środowiska Grid Computing są definiowane przez administratora systemu i mogą być przez niego dynamicznie zmieniane. Często dopuszcza się możliwość przenoszenia pomiędzy węzłami nawet tych zadań, które znajdują się w trakcie realizacji. Dzięki temu możliwe jest dynamiczne równoważenie obciążenia pomiędzy wszystkimi węzłami.

Uzyskanie ogromnej równoległej mocy przetwarzania jest jednym z najważniejszych atrybutów Grid Computing. Jest to odpowiedź na zapotrzebowanie formułowane nie tylko w ramach projektów naukowych, ale również przez technologie biomedyczne, modelowanie finansowe, technologie wydobycia ropy naftowej, animację komputerową, itd. Należy zwrócić uwagę na specyfikę aplikacji, które potrafią wykorzystać potencjał dużej liczby równoległych komputerów. Są to aplikacje wysoce aktywne obliczeniowo, ale podzielne na mniejsze fragmenty-podzadania, które mogą być niezależnie wykonywane na oddzielnych komputerach i nie muszą realizować intensywnej komunikacji z innymi podzadaniami. Potocznie o takich aplikacjach mówi się, że są "skalowalne", co w optymalnym przypadku oznacza, że np. dziesięciokrotne zwiększenie liczby procesorów spowoduje dziesięciokrotne skrócenie czasu pracy aplikacji. W praktyce tak dobra skalowalność nie

występuje, głównie w związku z niemożliwością podziału algorytmów na dowolnie dużą liczbę podzadań lub w związku z koniecznością synchronizacji dostępu do współdzielonych zasobów. Oczywiście nie każdy rodzaj aplikacji może zostać efektywnie dostosowany do pracy w środowisku Grid Computing (grid-enabled applications). Przekształcenie istniejącej aplikacji w aplikację zdolną wykorzystywać własności środowiska Grid Computing musi zostać wykonane manualnie przez jej projektantów i programistów. Pomoc w realizacji tego procesu mogą stanowić ogólnodostępne biblioteki i środowiska programowania, np. Globus Toolkit.

Środowiska Grid Computing oferują także możliwości podnoszenia niezawodności aplikacji. Jeżeli awarii ulega węzeł realizujący jedno z podzadań aplikacji, wówczas podzadanie to może zostać ponownie uruchomione na innym, sprawnym węźle. Realizacja zadań może wręcz z założenia przebiegać z pewną redundancją tak, aby ewentualne awarie pojedynczych węzłów nie powodowały wzrostu czasu odpowiedzi krytycznych aplikacji czasu rzeczywistego. Jeżeli do awarii nie dojdzie, wtedy nadmiarowe wyniki zostaną „zniweczone".

Agregacji mogą podlegać nie tylko zasoby obliczeniowe, ale również zasoby dyskowe. Często serwery lub stacje robocze dysponują bardzo dużymi, w znacznym stopniu niewykorzystanymi, zasobami pamięci masowej. Połączenie tych zasobów w jeden duży wirtualny zbiornik danych, nazywany Data Grid, umożliwia zwykle uzyskanie nie tylko zwiększonej pojemności, ale również pozwala na poprawę wydajności (striping) i niezawodności (mirroring i replikacja). Jeżeli zadanie obliczeniowe uruchamiane na jednym z węzłów środowiska Grid Computing wymaga intensywnego dostępu do danych, to dane te mogą zostać uprzednio skopiowane do pamięci masowej lokalnej w stosunku do komputera, któremu zostanie powierzone to zadanie (automatyczna replikacja). Zabieg taki spowoduje nie tylko poprawę wydajności aplikacji, ale także częściowo zabezpieczy dane na wypadek awarii (zniszczenia lub niedostępności) danych źródłowych.

Praktyczne środowiska Grid Computing są zwykle konstruowane z trzech głównych typów komponentów:

- Stacje robocze (scavenging grids): zakłada się, że użytkownicy komputerów osobistych nie wykorzystują permanentnie ich pełnej mocy obliczeniowej, w związku z czym może ona zostać "zagospodarowana" na potrzeby rozproszonego przetwarzania danych; zadania obliczeniowe są najczęściej realizowane w formie wygaszaczy ekranu, aktywowanych przez system operacyjny w czasie nieaktywności użytkownika; aplikacje uruchamiane w środowisku zbudowanym z dużej liczby stacji roboczych cechują się bardzo wysokim stopniem równoległości.
- Serwery (computational grids): dedykowane środowiska połączonych komputerów o dużej mocy obliczeniowej umożliwiają efektywną realizację bardzo złożonych zadań przetwarzania danych; zadania obliczeniowe pochodzące ze środowiska Grid Computing nie muszą rywalizować o dostęp do procesora z zadaniami "lokalnymi"; istnieją możliwości udostępniania specjalizowanych urządzeń na potrzeby wybranych zadań obliczeniowych.
- Dane (data grids): współdzielenie danych jest koniecznym warunkiem realizacji przetwarzania Grid Computing; aplikacje korzystają z wirtualnej bazy danych lub wirtualnego systemu plików, których lokalizacje są przezroczyste dla aplikacji; stosowane są mechanizmy automatycznej replikacji lub transferu danych w celu podniesienia wydajności operacji wejścia-wyjścia

Istotną funkcją środowiska Grid Computing jest alokacja aplikacji lub zadań na dostępnych zasobach. W najprostszych systemach, odpowiedzialność przydziału komputera do konkretnego zadania spoczywa na użytkowniku, który uruchamia zadanie. Bardziej

zaawansowane systemy posiadają dedykowany moduł rozdziału zadań (scheduler, resource broker), który automatycznie odnajduje węzeł najbardziej odpowiedni do realizacji powierzonego zadania. Moduły rozdziału zadań potrafią adaptować się do aktualnej dostępności węzłów w środowisku. W środowiskach Grid Computing konstruowanych ze stacji roboczych, aktywną stroną są stacje robocze w stanie bezczynności, które zgłaszają się do modułu rozdziału zadań i pobierają nowe zadanie do wykonania. Niestety, nagłe pojawienie się na takich stacjach aktywności "lokalnej" zwykle powoduje zawieszenie lub opóźnienie realizacji powierzonego zadania, a to przekłada się na nieprzewidywalność czasu odpowiedzi całej aplikacji.

Poza dynamicznym rozdziałem zadań, środowisko Grid Computing powinno umożliwiać również wstępną rezerwację zasobów przez aplikację w celu zagwarantowania zadanej jakości usługi (quality of service). Przydział innych zadań do zasobów zarezerwowanych przez aplikację jest wówczas ograniczony i zwykle możliwy wyłącznie dla zadań o niskich priorytetach.

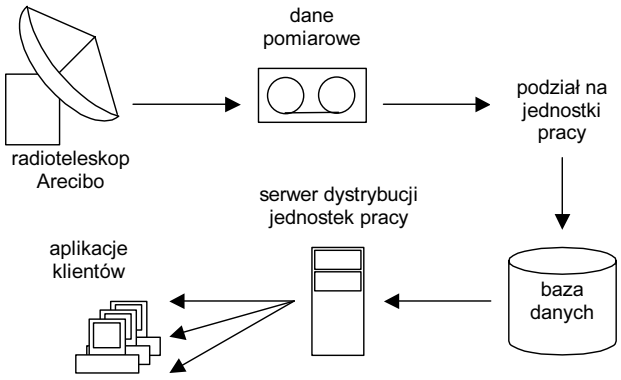
3. Początki Grid Computing: SETI@home

Jedną z pierwszych praktycznych realizacji wizji przetwarzania Grid Computing był projekt SETI@home, który umożliwił konstrukcję ogromnego, darmowego superkomputera dla prowadzenia obliczeń astronomicznych dla potrzeb SETI. SETI (Search for Extraterrestial Intelligence) jest dziedziną badań naukowych, której celem jest poszukiwanie inteligentnego życia pozaziemskiego [SHO98]. Jedno ze stosowanych w tej dziedzinie podejść, nazywane Radio SETI, wykorzystuje radioteleskopy astronomiczne do prowadzenia nasłuchu wąskiego pasma sygnałów radiowych pochodzących z przestrzeni kosmicznej. Odebranie takiego sygnału byłoby traktowane jako dowód istnienia technologii pozaziemskiej.

Niestety, radioteleskopy astronomiczne odbierają sygnały składające się w większości z szumów generowanych przez ziemskie urządzenia elektroniczne, a także z sygnałów telewizyjnych, satelitarnych i radarowych. W związku z tym konieczna jest cyfrowa analiza odbieranych sygnałów w celu wyeliminowania tych, które są pochodzenia ziemskiego. Analiza taka przebiega zwykle w trzech fazach: (1) dokonywany jest rozkład sygnału na częstotliwości składowe, (2) przy pomocy algorytmów rozpoznawania wzorców, znajdowane są sygnały-kandydaci do dalszej analizy, (3) eliminowane są te sygnały-kandydaci, które są najprawdopodobniej pochodzenia naturalnego lub ziemskiego. Cyfrowe przetwarzanie sygnałów odbieranych przez radioteleskopy jest bardzo kosztowne obliczeniowo, proporcjonalnie do szerokości i czułości nasłuchiwanego pasma radiowego oraz proporcjonalnie do stopnia pokrycia nieba przez radioteleskop. Powoduje to stale rosnące i niezaspokojone zapotrzebowanie na ogromną moc obliczeniową komputerów wykorzystywanych w projektach Radio SETI.

We wczesnych badaniach Radio SETI, do masowej analizy danych wykorzystywano superkomputery zlokalizowane w bliskości teleskopu. W roku 1995, David Gedye zaproponował realizację badań Radio SETI przy pomocy wirtualnego superkomputera zbudowanego z dużej liczby pojedynczych maszyn (głównie stacji roboczych) połączonych siecią Internet [ACK+02]. Projekt został nazwany SETI@home, a dzięki jego spopularyzowaniu wśród użytkowników sieci Internet, do października roku 2003 dobrowolnie i nieodpłatnie zaangażowało się w niego ponad 4.5 miliona komputerów, realizując łącznie ponad 1.5 miliona lat obliczeń, odpowiadających realizacji 3.7×10^{21} operacji zmiennoprzecinkowych [SETI].

Fundament projektu SETI@home stanowi największy i najczulszy na świecie radioteleskop astronomiczny znajdujący się w Arecibo w Puerto Rico, którego antena pomocnicza jest źródłem danych pomiarowych przetwarzanych przez aplikacje SETI@home. Dane źródłowe, zbierane z szybkością 5Mbps w paśmie 1.42 GHz (pasmo wodoru), są zapisywane na taśmach i transportowane do Uniwersytetu Berkeley w Kalifornii (rys. 1). Tam sygnał jest dzielony na fragmenty podpasma szerokości 1 kHz i długości 20s, nazywane jednostkami pracy (work unit), które są zapisywane w bazie danych, a następnie dystrybuowane wśród komputerów biorących udział w projekcie. Zainstalowana wcześniej na każdym z tych komputerów aplikacja klienta SETI@home pobiera jednostkę pracy, przetwarza ją, a wyniki odsyła przez Internet do Uniwersytetu Berkeley. Następnie aplikacja klienta może pobrać i przetwarzać kolejną jednostkę pracy. Aplikacje klientów nie komunikują się między sobą. Są zaimplementowane dla 175 platform systemowych w formie wygaszacza ekranu, procesu tła lub aplikacji graficznej. W przetwarzaniu jednostek pracy stosuje się także niewielką redundancję, w wyniku której pojedyncza jednostka jest przetwarzana dwa lub trzy razy przez różnych klientów. Celem redundancji jest wykrycie i wyeliminowanie wyników generowanych przez uszkodzonych lub fałszywych klientów.



Rys. 1. Dystrybucja danych pomiarowych SETI@home

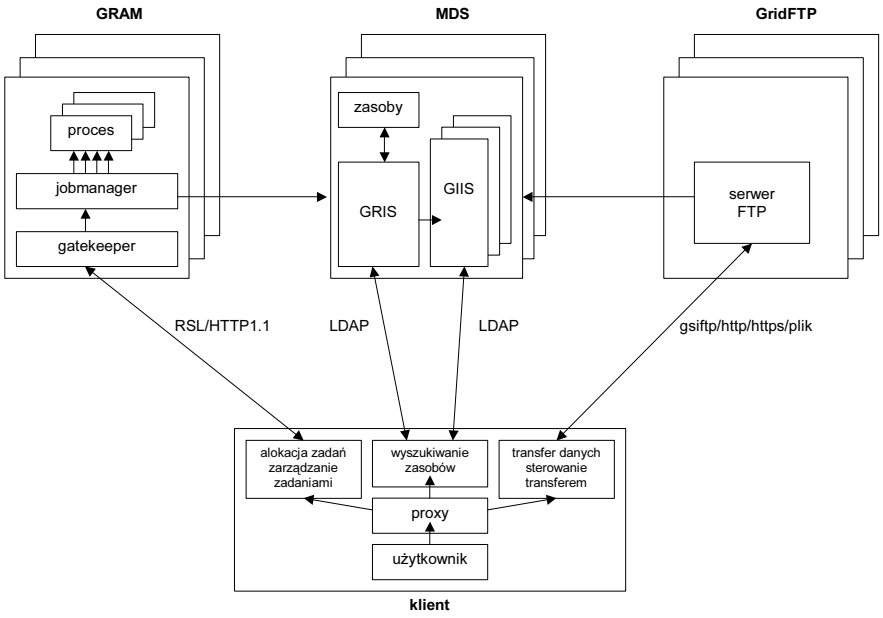
Przetwarzanie sygnału przez aplikację klienta SETI@home ma na celu wykrycie jednego z czterech rodzajów wzorców: impulsów wzrostu mocy sygnału, wąskopasmowych sygnałów gaussowskich, sygnałów pulsujących lub potrójnych impulsów wzrostu mocy sygnału o tej samej częstotliwości (trypletów). Wzorce odkryte przez aplikacje klientów, po dostarczeniu do odbiorczej bazy danych Uniwersytetu Berkeley, poddawane są dalszej analizie, której celem jest eliminacja sygnałów pochodzenia ziemskiego. Przez wszystkie lata realizacji projektu nie znaleziono żadnego wzorca pochodzenia pozaziemskiego.

4. Narzędzia dla konstrukcji i programowania środowisk Grid Computing

Rynek oferuje bardzo dużą liczbę narzędzi wspomagających konstrukcję, konfigurację oraz programowanie środowisk Grid Computing. Najpopularniejszymi z nich są: Globus Toolkit [Globus], Oracle Globus Development Kit (oparty na Globus) [Oracle], IBM Grid Toolbox (oparty na Globus) [IBM], Avaki [Avaki], DataSynapse LiveCluster [Synapse], Entropia DCGrid [Entropia], Platform LSF, ActiveCluster oraz MultiCluster [Platform],

United Devices MetaProcessor Platform [United]. Z punktu widzenia funkcjonalności i liczby zastosowań, najbardziej godny uwagi jest pakiet Globus Toolkit.

Globus Toolkit [FBA+03][FK97] zawiera trzy podstawowe komponenty do budowy środowisk Grid Computing: (1) moduł zarządzania zasobami, (2) moduł usług informacyjnych i (3) moduł zarządzania danymi (rys. 2). *Moduł zarządzania zasobami* odpowiada m.in. za alokację zadań do węzłów, zdalne uruchamianie zadań, zarządzanie stanem i postępem ich pracy. Zawiera wewnętrzne komponenty GRAM (Grid Resource Allocation Manager) i GASS (Global Access to Secondary Storage). Nie zawiera wbudowanego komponentu rozdzielania zadań, lecz oferuje interfejs pozwalający wykorzystywać zewnętrzny komponent odpowiedzialny za taką funkcjonalność. Żądania uruchomienia zadań (komenda `globusrun`), zapisane w języku RSL (Resource Specification Language), są przekazywane przez użytkowników do elementu Gatekeeper. Gatekeeper jest demonem służącym do autoryzacji użytkownika, powoływania procesu Jobmanager i przekazywania jemu dalszej komunikacji z klientem. Proces Jobmanager parsuje żądania wyrażone w języku RSL, alokuje zadania do lokalnych zasobów, a po zakończeniu zadań przekazuje klientom wyniki ich pracy. *Moduł usług informacyjnych* jest prostym serwerem LDAP, służącym do przechowywania informacji o strukturze całego środowiska Grid Computing (adresy IP, rozmiary pamięci, itp.) oraz statystyk opisujących obciążenie poszczególnych węzłów. Swoją funkcjonalność realizuje przy pomocy komponentów GRIS (Grid Resource Information Service) i GIIS (Grid Index Information Service), wspólnie nazywanych MDS (Monitoring and Discovery Service). Użytkownik uzyskuje dostęp do modułu usług informacyjnych przy pomocy komendy `ldapsearch`. *Moduł zarządzania danymi* oferuje mechanizmy transferu plików i zarządzania transferem plików pomiędzy węzłami Grid Computing. Jego kluczowym elementem jest komponent GridFTP, bazujący na standardowym FTP, lecz wzbogacający je o transmisję wielokanałową, automatyczne strojenie i zintegrowane mechanizmy bezpieczeństwa. Pełna funkcjonalność Globus Toolkit jest dostępna w formie komend wiersza poleceń oraz API dla popularnych języków programowania.



Rys. 2. Architektura środowiska Globus Toolkit

5. Grid Computing w Oracle 10g

System zarządzania bazą danych Oracle 10g zawiera pewne mechanizmy wspomagające implementację idei Grid Computing w środowisku intranetu (tzw. Enterprise Grid Computing) [Oracle03]. Mechanizmy te umożliwiają konstrukcję wirtualnej instancji bazy danych, składającej się z wielu rzeczywistych instancji rozlokowanych na różnych węzłach sieci, oraz konstrukcję wirtualnej bazy danych, składającej się z plików rozmieszczonych na dyskach dołączonych do wielu węzłów sieci. Realizacja większości funkcji została oparta na pakiecie Oracle Globus Development Kit. Trzy główne rozwiązania, stanowiące filary dla realizacji tego typu środowisk to: (1) automatyczne strojenie instancji bazy danych, (2) automatyczna migracja danych i (3) automatyczna alokacja instancji.

Funkcje automatycznego strojenia instancji bazy danych wyręczają administratorów z zadań regularnego monitorowania wydajności systemu oraz dostosowywania konfiguracji instancji do specyfiki aplikacji i zapytań realizowanych przez instancje. Na podstawie regularnie zbieranych parametrów obciążenia systemu następuje dynamiczna reorganizacja pamięci SGA w celu poprawy współczynników wydajności pracy użytkowników. Reorganizacja SGA sprowadza się do dostosowywania rozmiarów zbiornika współdzielonego (shared pool) oraz obszarów buforowych (buffer cache) do specyfiki realizowanego przetwarzania. Mimo iż taka funkcjonalność jest pożądana nie tylko w środowiskach Enterprise Grid Computing, to jednak właśnie w takich rozwiązaniach bardzo zyskuje na znaczeniu. W przypadku środowisk składających się z dużej liczby dynamicznych węzłów, manualna realizacja strojenia instancji przez administratora byłaby niezwykle kosztowna.

Automatyczna migracja danych polega na sporządzaniu kopii fragmentu bazy danych i przeniesieniu jej do węzła, na którym pracuje instancja intensywnie z tych danych korzystająca. Dzięki temu, dane lokalizowane są jak najbliżej procesów, które z nich korzystają, umożliwiając efektywną realizację operacji wejścia-wyjścia. Migracja danych może następować także pomiędzy instancjami pracującymi pod kontrolą nieidentycznych systemów operacyjnych. Funkcje automatycznej migracji danych są realizowane przy pomocy mechanizmów Oracle Streams, Data Pump i Transportable Tablespaces.

Automatyczna alokacja instancji jest jednym z najciekawszych rozwiązań wprowadzonych do Oracle10g. Załóżmy, że dysponujemy środowiskiem Enterprise Grid Computing zawierającym 6 serwerów, początkowo równo podzielonych pomiędzy dwa niezależne systemy informatyczne A i B. Jeżeli aktywność użytkowników każdego z systemów będzie niewielka, wtedy na trzech serwerach będą pracować instancje systemu A, a na pozostałych trzech – instancje systemu B. Jeżeli jednak aktywność użytkowników systemu A gwałtownie wzrośnie, wtedy może dojść do automatycznego zatrzymania instancji systemu B na dwóch z trzech serwerów i zamiast tego uruchomienia na nich instancji systemu A. Dotychczasowe sesje użytkowników podłączonych do zatrzymanych instancji zostaną w przezroczysty sposób przeniesione do innych instancji. W ten sposób pięć serwerów będzie mogło równocześnie obsługiwać intensywną pracę użytkowników systemu A, podczas gdy mało aktywni użytkownicy systemu B będą realizować swoją pracę przy użyciu instancji pracującej wyłącznie na jednym serwerze. Jeżeli w przyszłości ulegnie zmianie charakterystyka obciążenia systemów, wtedy może dojść do dalszych zmian w sposobie alokacji instancji na serwerach środowiska Enterprise Grid Computing. Alokacja instancji odbywa się według reguł zdefiniowanych przez administratora środowiska i jest przez niego w pełni kontrolowana. Warto nadmienić, że automatyczna alokacja instancji jest stosowana również podczas awarii węzłów, kiedy dokonuje rozłożenia dotych-

czasowego obciążenia systemu pomiędzy dostępne sprawne węzły. Podstawowa funkcjonalność mechanizmów automatycznej alokacji instancji i równoważenia ich obciążenia pochodzi z Oracle Real Application Clusters.

6. Podsumowanie

Przetwarzanie Grid Computing umożliwia efektywne wykorzystanie posiadanych zasobów komputerowych oraz pozwala na konstrukcję systemów o podwyższonej niezawodności. Dostępność gotowych, standardowych narzędzi programistycznych stanowi dla projektantów zachętę do rozważenia architektur Grid Computing w nowotworzonych systemach. Implementacja idei Grid Computing w systemie zarządzania bazą danych Oracle 10g umożliwia łatwą reorganizację posiadanego środowiska serwerów dedykowanych w celu budowy wielkich wirtualnych serwerów baz danych, wykorzystywanych przez wszystkie systemy informatyczne przedsiębiorstwa.

7. Literatura

1. [ACK+02] David P. Anderson, Jeff Cobb, Eric Korpela, Matt Lebofsky, Dan Werthimer, *SETI@home: An Experiment in Public-Resource Computing*, Communications of the ACM, Vol. 45 No. 11, November 2002
2. [Avaki] http://www.avaki.com
3. [CFK+01] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, S. Tuecke , *The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets*, Journal of Network and Computer Applications, 23:187-200, 2001
4. [Entropia] http://www.entropia.com
5. [FBA+03] Fereira, L., et. al*., Introduction to Grid Computing With Globus*, IBM Redbook, December 2002
6. [FK97] I. Foster, C. Kesselman, *Globus: A Metacomputing Infrastructure Toolkit*. Intl J. Supercomputer Applications, 11(2):115-128, 1997
7. [FK99] I. Foster, C. Kesselman, *Computational Grids*, Chapter 2 of "The Grid: Blueprint for a New Computing Infrastructure", Morgan-Kaufman, 1999
8. [FKN+02] I. Foster, C. Kesselman, J. Nick, S. Tuecke, *The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration*, Open Grid Service Infrastructure WG, Global Grid Forum, June 22, 2002
9. [FKT+01] I. Foster, C. Kesselman, S. Tuecke, *The Anatomy of the Grid: Enabling Scalable Virtual Organizations*, International J. Supercomputer Applications, 15(3), 2001.
10. [Globus] http://www.globus.org
11. [IBM] http://www.ibm.com
12. [Oracle] http://otn.oracle.com
13. [Oracle03] *Oracle 10g: Infrastructure for Grid Computing*, An Oracle White Paper, September 2003
14. [Platform] http://www.platform.com
15. [SETI] http://setiathome.ssl.berkeley.edu/totals.html
16. [SHO98] Shostak, Seth, *Sharing the Universe: Perspectives on Extraterrestrial Life*, Berkeley Hills Books, 1998
17. [Synapse] http://www.datasynapse.com
18. [United] http://www.ud.com