AKADEMIA GÓRNICZO-HUTNICZA
im. STANISŁAWA STASZICA W KRAKOWIE

Wzorce projektowe

Projekt oraz implementacja programowego mechanizmu load-balancing na warstwie mapowania O-R.

Autorzy

Łukasz Stanik

Michał Worsowicz

Kamil Tomaszewski

Radosław Bielecki

Opiekun projektu

Dr inż. Rafał Mrówka

9 grudnia 2020

1 Wstęp

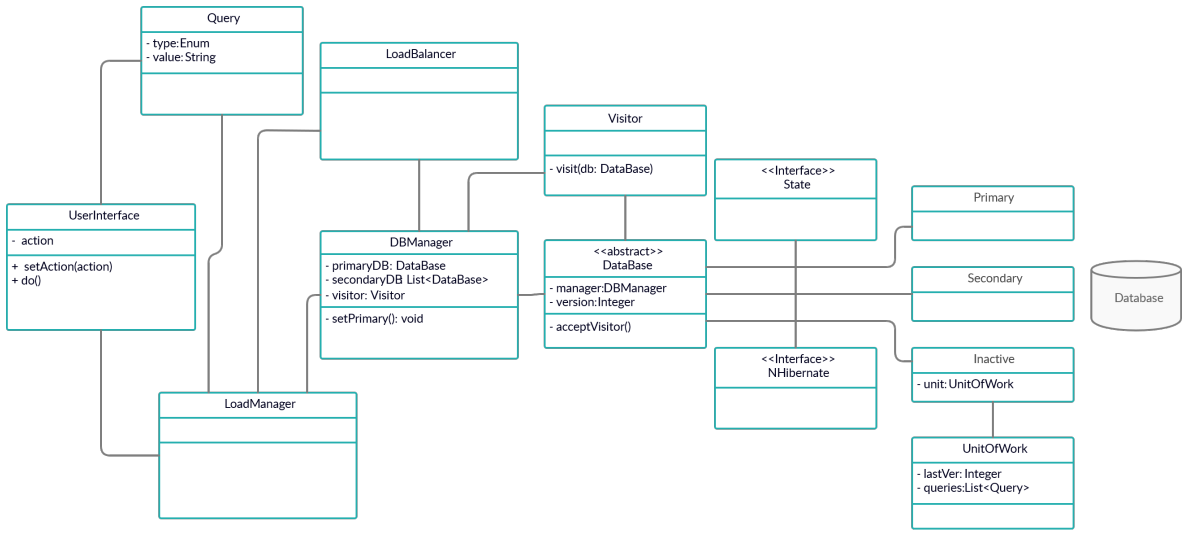
Celem naszego projektu jest implementacja programowgo mechanizmu load-balancing'u na warstwie mapowania OR.

Load-balancing odnosi się do procesu dystrybucji zestawu zadań na zbiór jednostek obliczeniowych, w celu zwiększenia wydajności ich działania. Technika ta pozwala zrównoważyć obciążenie, a co za tym idzie zoptymalizować czas odpowiedzi dla każdego zadania, jednocześnie unikając nierównomiernego przeciążenia węzłów. Load-balancing podzielić można na algorytmy statyczne i dynamiczne, gdzie w tych drugich uwzględniany jest stan maszyn oraz występuje wymiana informacji pomiędzy jednostkami.

Mapowanie O-R (obiektowo-relacyjne) jest za to techniką programowania służącą do konwersji danych pomiędzy niekompatybilnymi systemami przy użyciu obiektowych języków programowania. W efekcie powstaje "wirtualna baza danych", z której korzystać można przy użyciu danego języka programowania. Poszczególne wpisy do bazy danych traktowane są jako obiekty, a ich atrybuty - jako zmienne tego obiektu. Z takim obiektem można również skojarzyć różne funkcje, zapewniając nam dodatkową funkcjonalność.

W celu implemetacji opisanego mechanizmu wykorzystamy technologię C#. Do translacji danych pomiędzy relacyjną bazą danych a językiem obiektowym wykorzystamy narzędzie NHibernate, ze względu na jego kompatybliność z językiem C#.

2 Architektura przyjętego rozwiązania

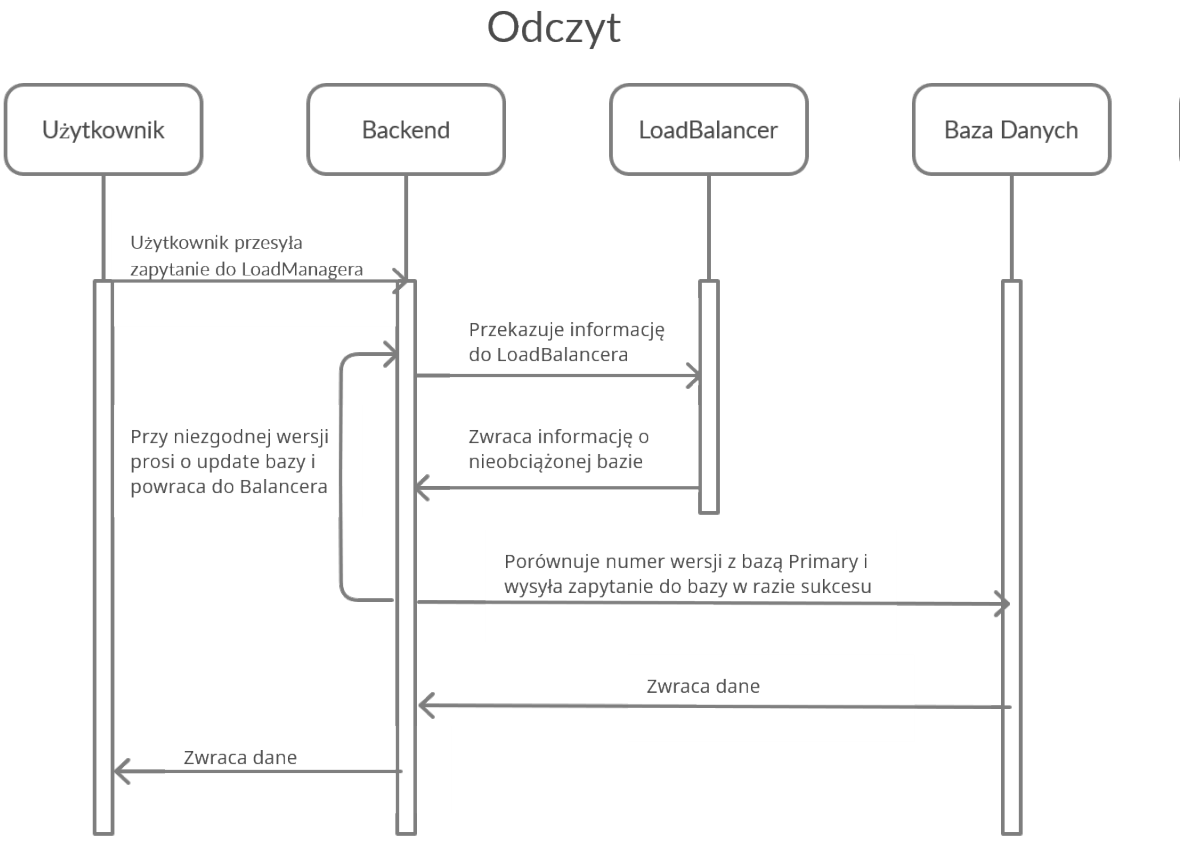


3 Architektura logiczna i specyfikacja dynamiki systemu

W tej sekcji opisany zostanie sposób działania programu i jego poszczególnych funkcjonalności.

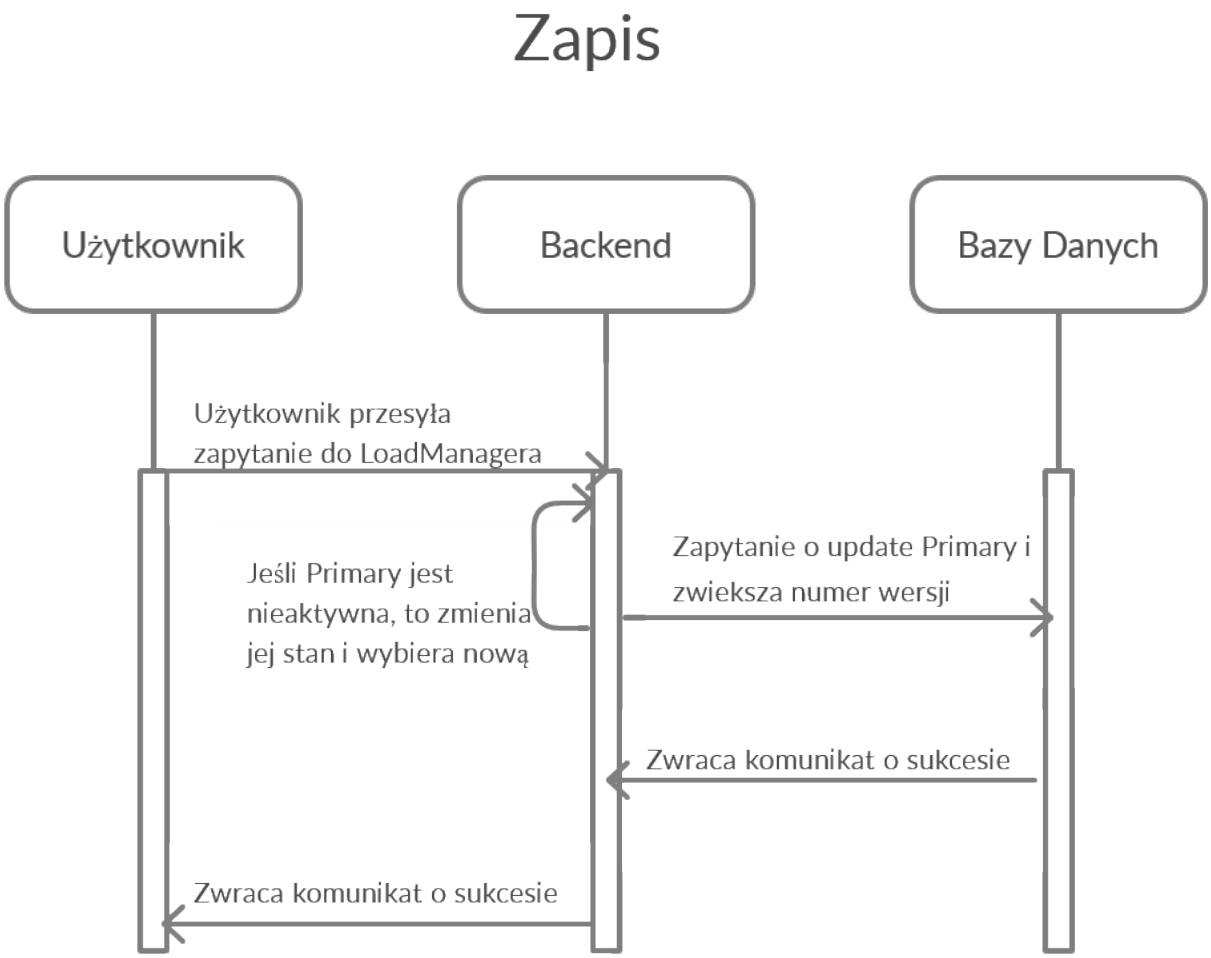
3.1 Odczyt

Funkcję Odczytu można w najproszy sposób opisać jako czynność w której użytkownik otrzymuje dane z bazy danych. Proces ten rozpoczyna się w momencie wysłania przez użytkownika zapytania do LoadManagera, który to następnie przekazuje informacje do LoadBalancera. LoadBalancer zwraca informacje o najmniej obiążonej bazie danych z powrotem do LoadManagera, który to porównuje wersje danej bazy z bazą Primary. W przypadku niezgodnej wersji wysyłana jest prośba o aktualizację bazy, a proces wraca do LoadBalancera. W przypadku sukcesu wysłane zostaje zapytanie do bazy. Na końcu baza danych zwraca dane, o której pytał użytkownik, do beckendu, a ten następnie przekazuje je do użytkownika.



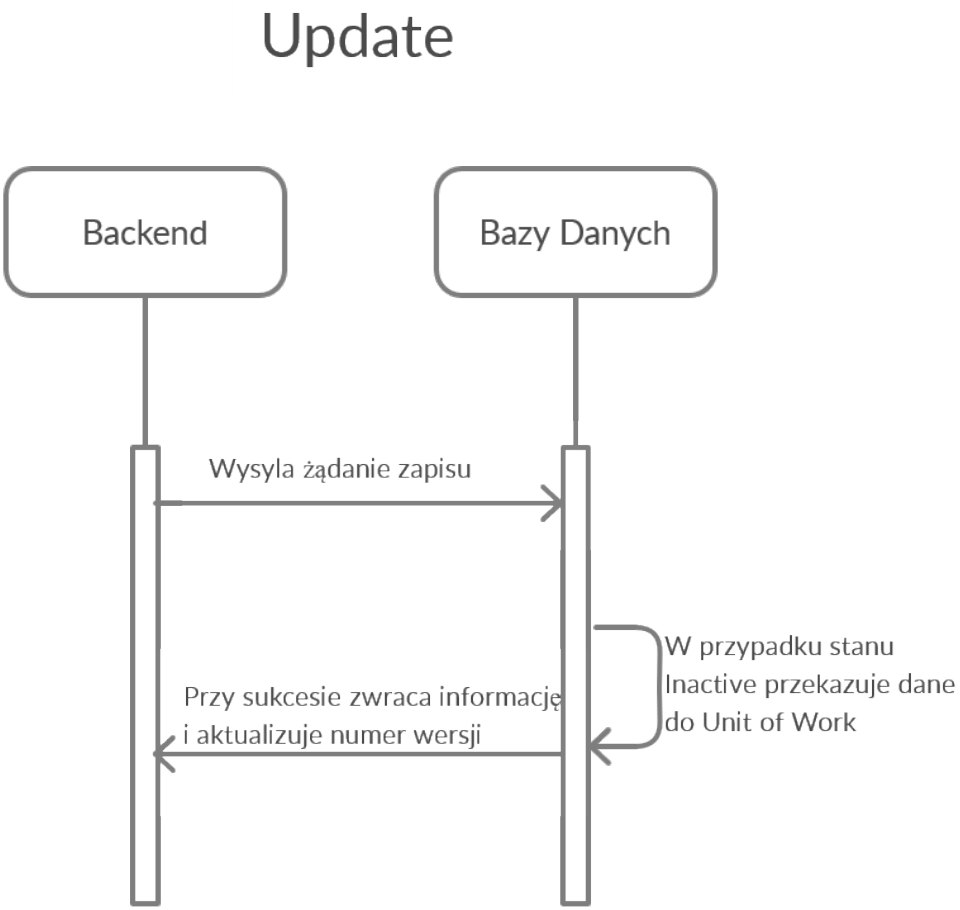
3.2 Zapis

Zapis zdefiniować można jako przekazanie przez użytkownika danych do bazy danych w celu ich zapamiętania. Początkowo użytkownik wysyła zapytanie do LoadManagera. W kolejnym kroku zapytanie o aktualizacje zostaje przesłane do bazy Primary, a numer wersji zostaje zwiększony. W przypadku gdy Primary jest nieaktywna jej stan zostaje zmieniony na Inactive, a następnie wybrana zostaje nowa baza Primary. W przypadku sukcesu informacja o tym zostaje zwrócona, wpierw do backendu, a następnie do samego użytkownika.



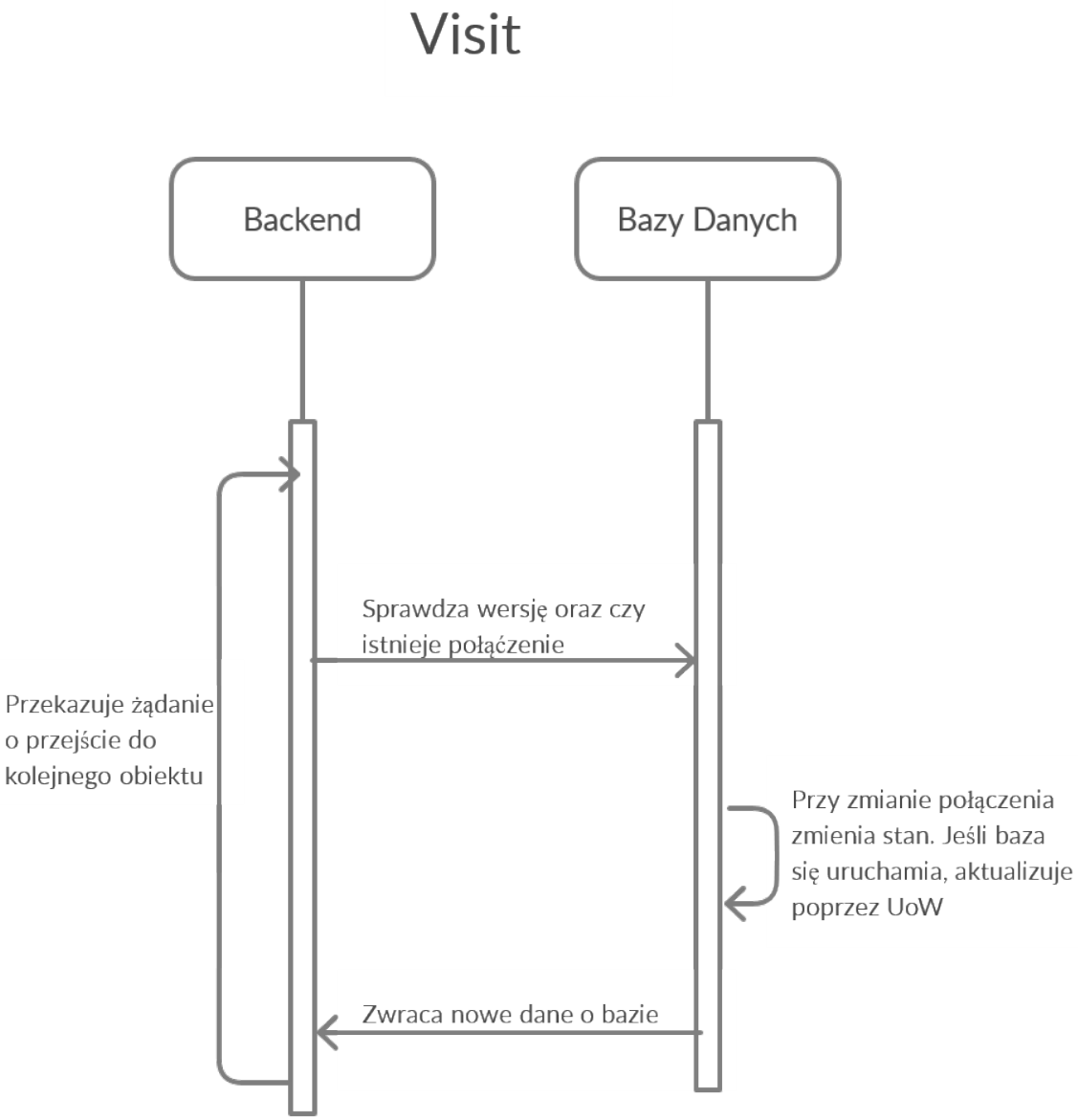
3.3 Update

Update to proces mający miejsce po wywołaniu zapisu do bazy danych, a jego celem jest synchronizacja danych we wszystkich bazach z bazą Primary. Całość zaczyna się w momencie, gdy backend wysyła żądanie zapisu do wszystkich baz danych. Jeżeli dana baza jest w stanie Inactive to odpowiednie dane przekazywane są do Unit of Work. W przypadku sukcesu zwraca na jest odpowiednia informacja, a numer wersji zostaje zaktualizowany tak aby odpowiadał temu z bazy Primary.



3.4 Visit

Zadaniem Visit jest monitorowanie stanu baz danych i podejmowanie odpowiednich kroków na podstawie uzyskanych informacji. W pierwszym kroku sprawdzana zostaje wersja oraz stan połączenia z daną bazą. Jeżeli zajdzie zmiana w połączeniu adekwatnie dochodzi do zmiany stanu bazy. W przypadku, gdy doszło do uruchomienia bazy danych zostaje ona zaktualizowana przy użyciu Unit of Work.W ostatnim kroku zwracane zostają nowe informacje o bazie i następuje przejście do kolejnego obiektu.



4 Lista wzorców

4.1 Visitor

Wzorzec Visitor pozwala na oddzielenie funkcji od obiektu na którym jest ona wykonywana, co służy do sprawdzenia stanu bazy danych bez jej modyfikacji. Dzięki temu możemy pilnować tego, czy bazy danych są aktywne i czy są zaaktualizowane do wersji bazy Primary. Zaimplementowany z użyciem algorytmu round-robin stale odwiedza kolejne bazy danych aby jak najszybciej aktualizować ich stan oraz wprowadzać ewentualne zmiany.

4.2 Unit of work

Unit of work to wzorzec, który śledzi wszystko, co zostaje zrobione i może mieć wpływ na bazę danych. Dowiaduje się i zapamiętuje on również wszystko, co należy zrobić, aby zmienić bazę danych w wyniku działań użytkownika. Zastosowanie w naszym projekcie znajduje przy zapisie do baz danych - w przypadku gdy zapis do którejkolwiek z nich się nie powiedzie to chcemy aby dane zostały zsynchronizowane w momencie, gdy dana baza wróci do pracy.

4.3 State

Stan to behawioralny wzorzec, umożliwiający obiektowi zmianę zachowania wraz ze zmianą jego stanu wewnętrznego. Stanowi on bardziej przejrzysty sposób zmiany zachowania obiektu w czasie działania programu, bez uciekania się do instrukcji warunkowych. Wzorzec ten posłuży nam do zmiany zachowania obiektów baz danych w zależności od ich stanu (primary, secondary, inactive).