WYDZIAŁ ELEKTRONIKI
I TECHNIK INFORMACYJNYCH

Rozproszone systemy operacyjne

Koncepcja testów porównawczych

Autorzy:

Tomasz Adamiec
Piotr Cebulski
Marek Kowalski
Mateusz Rosiewicz
Paweł Sokołowski
Marcin Wnuk

Warszawa, 2013

Wstęp

Testowanie oprogramowania jest to proces związany z wytwarzaniem oprogramowania. Jest to jeden z procesów zapewnienia jakości oprogramowania. Testowanie ma na celu weryfikację oprogramowania oraz walidację oprogramowania. Weryfikacja oprogramowania ma na celu sprawdzenie, czy wytwarzane oprogramowanie jest zgodne ze specyfikacją. Walidacja sprawdza, czy oprogramowanie jest zgodne z oczekiwaniami użytkownika. Testowanie oprogramowania może być wdrożone w dowolnym momencie wytwarzania oprogramowania (w zależności od wybranej metody). W podejściu klasycznym największy wysiłek zespołu testerskiego następuje po definicji wymagań oraz po zaimplementowaniu wszystkich zdefiniowanych funkcjonalności. Nowsze metody wytwarzania oprogramowania (np. Agile), skupiają się bardziej na jednostkowych testach wykonywanych przez członków zespołu programistycznego, zanim oprogramowanie trafi do właściwego zespołu testerów.

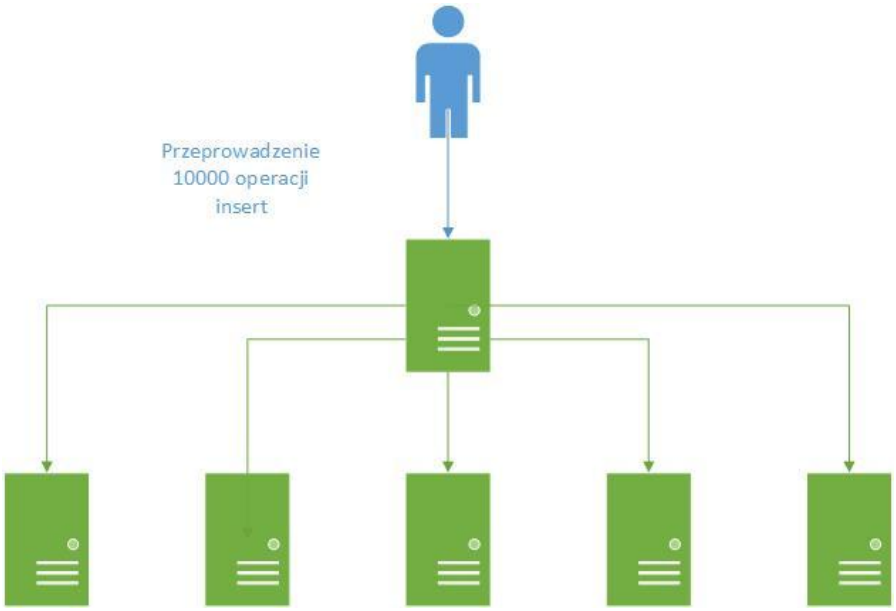
Testowanie nigdy nie jest w stanie wykryć wszystkich błędów danego oprogramowania, jednak może dostarczyć informacji o stabilności oprogramowania, zgodności z wymaganiami klienta, czy też zgodności z oczekiwaniami klienta. Trzeba pamiętać, że testowanie nie sprawdza pod kątem wszelkich możliwych warunków początkowych, lecz jedynie z wybranymi warunkami. Testowanie może we wczesnych fazach projektu wykryć defekty oprogramowania. Wczesne wykrycie defektu jest ważne z ekonomicznego punktu widzenia ponieważ gwarantuje niskie koszty naprawy. Nie wszystkie defekty oprogramowania wynikają z błędów kodowania aplikacji. Duża część defektów jest wynikiem błędów popełnionych podczas definicji wymagań. Tak więc testowanie oprogramowania sprowadza się również do wstępnej analizy wymagań.

Testy porównawcze

Środowisko testowe będzie znajdowało się w laboratorium wydziałowym w Sali 519. Oprogramowanie niezbędne do przeprowadzenia analizy naszego systemu to MongoDB oraz baza stworzona przez nasz zespół. Procedura polega na przeprowadzaniu identycznych testów dla obu rozwiązań. Planowaną metryką jest czas wykonania operacji.

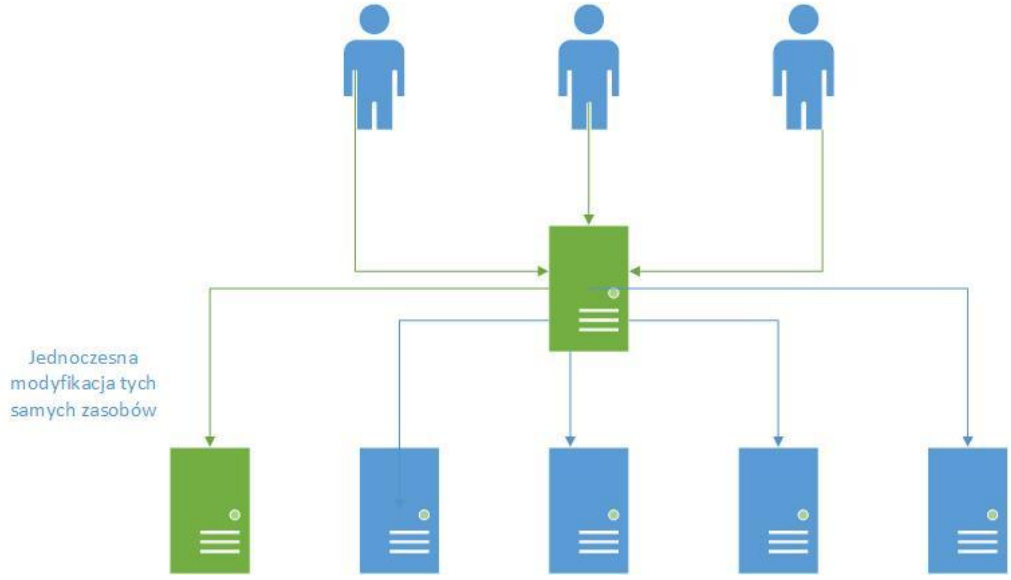
1.1 Przeprowadzenie wielu operacji insert

Test opiera się na wprowadzeniu do baz danych 10000 operacji insert zawierających losowe wartości. Celem jest analiza prędkości wykonania tych operacji oraz reakcji systemów na obciążenie. Jest to najprostszy z wykonywanych testów, jednak jednoznacznie określa wydajność testowanych rozwiązań.



1.2 Weryfikacja wielodostępu

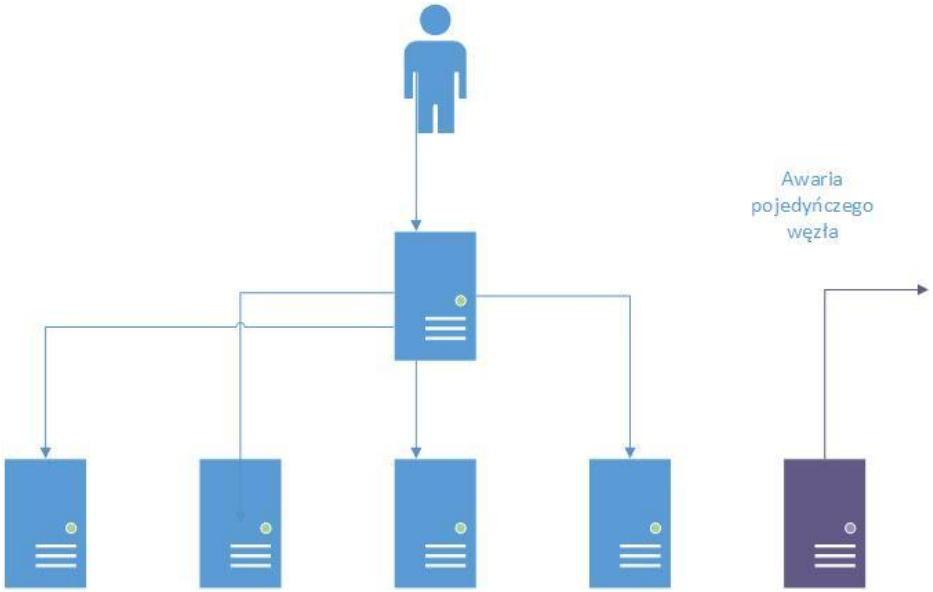
Wielodostęp polega na zabezpieczeniu spójności zasobów w chwili gdy jednocześnie wielu użytkowników próbuje modyfikować tę samą zawartość. Spójność danych jest kluczowym aspektem każdego zbioru informacji. W zależności od zastosowanych rozwiązań wyniki testów mogą być różne. Ocena testów będzie pochodną dwóch czynników: niezawodności oraz komfortu pracy – czasu wykonywania zadań.



1.3 Testowanie systemu w przypadku awarii węzła

Sharding, którego wydajność jest kluczowym punktem naszego systemu, polega na rozkładzie danych na różne węzły. Rozwiązanie to posiada wiele zalet. Przykładowo wykorzystując Sharding zmniejszamy obciążenie pojedynczych maszyn, gdyż zapytania w przypadku sensownego podziału rozkładane są równomiernie na wiele jednostek. Niestety takie rozwiązanie posiada również wady. Jak w każdym systemie rozproszonym mamy większą ilość punktów awarii niż w systemach scentralizowanych.

Ze względu na charakterystykę rozwijanego rozwiązania, przeprowadzimy testy analizujące utrzymanie spójności danych w przypadku awarii pojedynczego węzła. Mierzyć będziemy niezawodność rozwiązań jak i czas reakcji systemów na zaistniałą awarię.

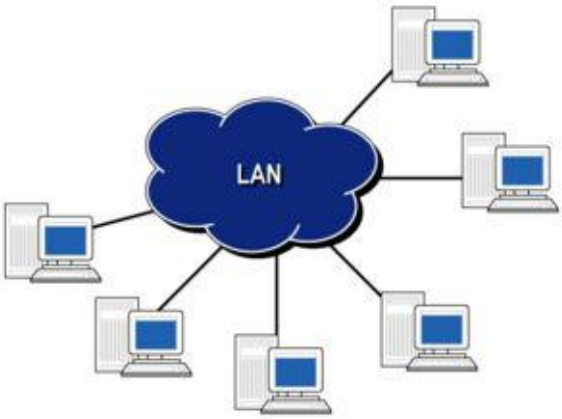


1.4 Testowanie shardingu

Sharding w mongodb pozwala na automatyczny podział danych oraz obciążenia na serwery. Naszym celem będzie poprawa algorytmów za to odpowiedzialnych. Planujemy poprawić wydajność zapisu danych poprzez optymalny rozkład obciążenia na poszczególne serwery.

W naszych testach porównamy czy nasze algorytmy w lepszy sposób dzielą obciążenie na poszczególne serwery podczas procesu shardingu.

Na podstawie jednakowych zbiorów danych zostaną wykonane procesy shardingu na jednakowych stacjach roboczych, dzięki temu wyniki otrzymane podczas testów będą miarodajne i w dużym stopniu pozwolą na porównanie wydajności naszych algorytmów z algorytmami zaimplementowanymi w MongoDB. Wyznacznikiem jakości będzie czas wykonania procesu shardingu i zapisu danych. Mniejszy czas uzyskany w testach będzie wyznacznikiem lepszej jakości wykorzystanego algorytmu. Poniżej znajduję się przybliżony schemat infrastruktury sieciowej planowanej do wykorzystania podczas przeprowadzania testów.



Poniżej przedstawiona jest lista kroków niezbędna do przeprowadzenia testów porównawczych.

1. Uruchomienie systemów operacyjnych na stacjach roboczych.
2. Zweryfikowanie poprawności działania komunikacji poprzez infrastrukturę sieciową.
3. Uruchomienie MongoDB oraz wszystkich pozostałych procesów niezbędnych do przeprowadzenia procesu shardingu.
4. Zaimportowanie zestawu danych testowych.
5. Wykonanie testu w postaci shardowania zaimportowanych danych.
6. Zebranie wyników.
7. Przeprowadzenie kroków od 3 do 6 analogicznie dla rozwiązania autorskiego.
8. Porównanie wyników.

1.5 Podsumowanie

Zaproponowane powyżej testy dostarczają informację na temat stabilności oraz wydajności naszego projektu w porównaniu z rozwiązaniem zastosowanym w MongoDB. W celu uzupełnienia informacji o naszym produkcie planujemy również przeanalizować wielkość shardów i częstotliwość wykorzystania poszczególnych węzłów. Da nam to pogląd na prace systemów.

Ostatnim elementem będzie analiza samego kodu za pomocą specjalistycznego narzędzia Sonar. Pozwoli nam to wyeliminować błędy wynikające ze złych nawyków programistycznych.