Rozproszone systemy operacyjne

Koncepcja testów porównawczych

Autorzy:

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

Warszawa, 2013

[1.1 Testowanie 4](#_Toc356293711)

[1.2 Koncepcja testów porównawczych 4](#_Toc356293712)

* 1. Testowanie

Testowanie oprogramowania  jest to proces związany z wytwarzaniem [oprogramowania](http://pl.wikipedia.org/wiki/Oprogramowanie). Jest to jeden z procesów zapewnienia [jakości oprogramowania](http://pl.wikipedia.org/wiki/Jako%C5%9B%C4%87_oprogramowania). Testowanie ma na celu [weryfikację](http://pl.wikipedia.org/wiki/Weryfikacja) oprogramowania oraz [walidację](http://pl.wikipedia.org/wiki/Walidacja_(technika)) oprogramowania. Weryfikacja oprogramowania ma na celu sprawdzenie, czy wytwarzane oprogramowanie jest zgodne ze specyfikacją. Walidacja sprawdza, czy oprogramowanie jest zgodne z oczekiwaniami [użytkownika](http://pl.wikipedia.org/wiki/U%C5%BCytkownik_komputera). 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.

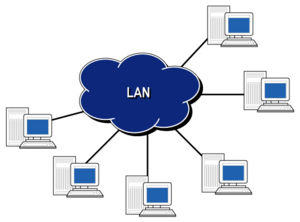
Testowanie nigdy nie jest w stanie wykryć wszystkich błędów danego oprogramowania, jednak może dostarczyć informacji o stabilności oprogramowania, zgodności z wymaganiami klienta, czy też zgodności z oczekiwaniami klienta. Trzeba pamiętać, że testowanie nie sprawdza pod kątem wszelkich możliwych warunków 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.

* 1. Koncepcja testów porównawczych

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.