Inżynieria Oprogramowania Systemów Rozproszonych

Laboratoria

Daniel Żmuda daniel.zmuda@agh.edu.pl

Zajęcia - informacje ogólne

W ramach zajęć laboratoryjnych omawiane będą **3 obszary** bezpośrednio powiązane z aspektami inżynierii oprogramowania systemów rozproszonych.

Zajęcia będą podzielone na 3 bloki trwające w okolicach **4 tygodni każdy**, w ramach których zdefiniowane będzie zagadnienie konieczne do realizacji. Każde ze zdefiniowanych zadań można roboczo nazwać **mini-projektem**.

Na **pierwszym** ze spotkań z każdego bloku realizowany będzie wstęp do aktualnie omawianego problemu. Przedstawione zostaną główne założenia projektu, ramy zaliczenia oraz pozostałe elementy bezpośrednio z nim powiązane.

Realizacja każdego z zadań kończy się **krótką prezentacją rozwiązania**, **weryfikacją poprawności działania** oraz **jakości aspektów niefunkcjonalnych** (struktura projektu, jakość kodu, wykorzystane narzędzia itp. [plusy za kreatywność ;-)]).

Na zajęciach prowadzonych przeze mnie możliwa jest praca w grupach składających się maksymalnie z **2 osób** (zakładając, że każdy z członków zespołu pracuje równomiernie).

Metodyka pracy - informacje ogólne

Na zajęciach prowadzonych przeze mnie obowiązuje kilka ważnych zasad:

- **Nie ma** bezpośrednich **wymogów technologicznych**. Projektujecie/implementujecie/dokumentujecie w takich środowiskach i z wykorzystaniem takich narzędzi, z którymi czujecie się komfortowo (jeżeli ktoś będzie potrzebował natchnienia to z dużym prawdopodobieństwem może liczyć na konkrety z “półświatka” Javo’wego).
- Jeżeli temat mini-projektu jest dla was zbyt sztywny możecie **zaproponować coś od siebie**.
- Jeżeli zbytnio przesadzam z zakresem mini-projektu można (a nawet należy) o tym poinformować.
- **Zaangażowanie** w realizację mini-projektu jest **równie ważne** jak jego końcowa postać.
- Jeżeli coś jest niejasne bądź pojawiają się jakieś wątpliwości **nie należy się bać pytań** (nawet tych na pierwszy rzut oka dosyć trywialnych).

Warunki uzyskania zaliczenia

Warunkiem konieczny uzyskania zaliczenia jest zrealizowanie wszystkich projektów zdefiniowanych w ramach laboratorium.

Przekroczenie terminu zaliczenia etapu bez ważnych powodów oraz wcześniejszego ustalenia z prowadzącym jest równoznaczne z natychmiastowym otrzymaniem oceny niedostatecznej (brak zaliczenia) – dalsze uczestnictwo w zajęciach możliwe jest w trybie zaliczenia poprawkowego (odpowiednio drugi lub trzeci termin)

Oceniane elementy projektu

Zrozumienie tematyki problemu:

- Krótka prezentacja opisująca problem i proponowany sposób rozwiązania

Poprawność rozwiązania:

- Zautomatyzowany sposób ewaluacji zaimplementowanego rozwiązania

Obszary około projektowe:

- Struktura projektu, jakość kodu, wykorzystywane narzędzia itp.
- Środowiska Continuous Integration/Continous Delivery
- Aspekty technologiczne (wzorce, paradygmaty itp.)
- Itp. (plusy za kreatywność)

Temat 1
The concept of *concensus* in distributed systems

Wprowadzenie

Daniel Żmuda daniel.zmuda@agh.edu.pl

Wprowadzenie

Jedna z definicja systemu rozproszonego

“A distributed system is a model in which components located on **networked** computers communicate and **coordinate** their actions by passing messages. The components interact with each other in order to achieve a common goal. Three significant characteristics of distributed systems are: **concurrency** of components, lack of a **global clock**, and independent **failure** of components.”

Coulouris, George; Jean Dollimore; Tim Kindberg; Gordon Blair (2011). *Distributed Systems: Concepts and Design (5th Edition)*. Boston: Addison-Wesley.

Problem

Na poziomie technologicznym założenia wyglądają następująco:

- system rozproszony
  - wiele węzłów
  - konieczność komunikacji
- awarie
  - sprzętu
  - infrastruktury sieciowej
  - błędy w oprogramowaniu

Cel:

Zapewnienie niezawodnej (pod względem skończoności) komunikacji pomiędzy rozproszonymi procesami na potrzeby takich operacji jak wybór lidera, synchronizacja stanu czy atomowe rozgłaszanie.

Rozwiązanie

Istnieje wiele algorytmów rozwiązujących problem konsensusu. Do najbardziej popularnych należą:

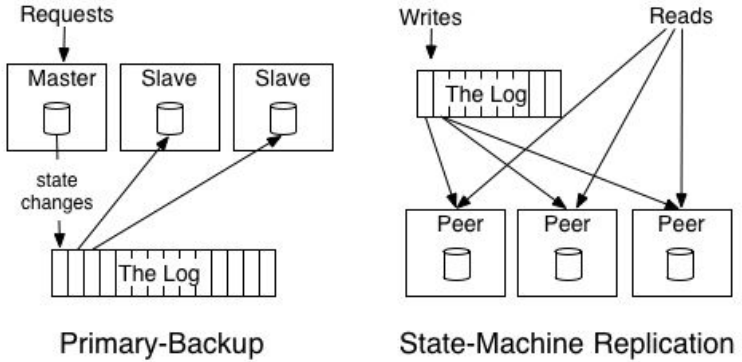
- PAXOS (http://research.microsoft.com/en-us/um/people/lamport/pubs/paxos-simple.pdf) ;
- RAFT (https://raft.github.io/raft.pdf);
- Ripple (https://ripple.com/files/ripple_consensus_whitepaper.pdf);
- i wiele innych.

W większości przypadków każde z rozwiązań, które wykorzystujecie w kontekście systemów rozproszonych wykorzystuje co najmniej jeden z w/w algorytmów.

Rozwiązanie

Implementacja zadanego algorytmu jest zadaniem dosyć nisko-poziomowym. Dodatkowo, jeżeli istnieje konieczność jej zastosowania w rzeczywistości powinna być wydajna i stabilna.

W dużej ilości rozwiązań można wyszczególnić osobną warstwę abstrakcji nazywaną “Distributed Log” bądź “WAL” (Write-Ahead Log).



Tematyka projektu

Zaprojektowanie oraz implementacja replikowalnego rozwiązania np. baza typu key-value, rozproszony log bądź warstwa pośrednicząca.

- można poskładać rozwiązanie z dostępnych na rynku technologii zaznajamiając się z metodą synchronizacji stanu w nim wykorzystywanym.

Czas trwania: 4 tygodnie

Etapy:

- Zestawienie środowiska do rozwoju systemu
- Projekt systemu
- Projekt i implementacja testów (zautomatyzowanych w miarę możliwości) prezentujących zachowanie systemu (duży plus za dostarczenie rozwiązania zgodnego z tzw. “chaos testing”)
- Implementacja rozwiązania
- Prezentacja obejmująca omówienie problemu, zaprojektowany system, oraz testy

Materiały

Distributed Log:

- Jay Kreps, “I Heart Logs”
- https://engineering.linkedin.com/distributed-systems/log-what-every-software-engineer-should-know-about-real-time-datas-unifying

CAP Theorem:

- http://www.glassbeam.com/sites/all/themes/glassbeam/images/blog/10.1.1.67.6951.pdf
- https://people.eecs.berkeley.edu/~brewer/cs262b-2004/PODC-keynote.pdf

Courses:

- [Coursera] Cloud Computing Concepts, Part 1
- [Coursera] Cloud Computing Concepts, Part 2