
JPAXOS

JAVA LIBRARY FOR STATE MACHINE REPLICATION

PRZYGOTOWANE PRZEZ

Jan Kończak

Tomasz Żurkowski

WE WSPÓLPRACY Z

Nuno Santos (EPFL)

KIEROWNIK ZADANIA

dr. hab. Paweł T. Wojciechowski



- 1 Replikacja
- 2 JPaxos
- 3 Wymagania
- 4 API
- 5 DEMO



REPLIKACJA



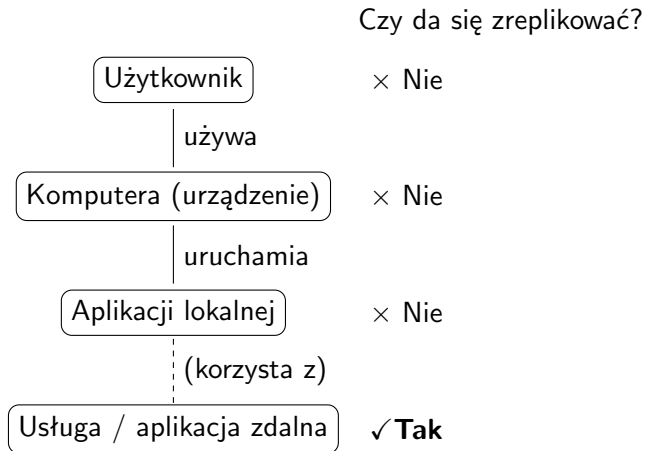
Replikować!

Co replikować?

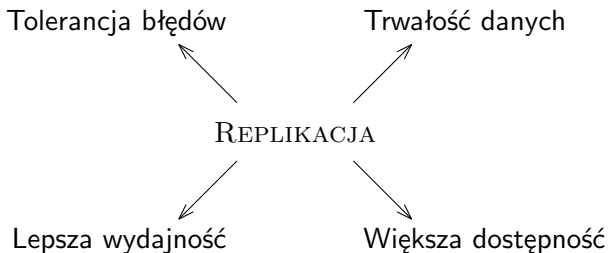
Po co replikować?



JAK WYGLĄDA TYPOWA APLIKACJA?



MOŻLIWE CELE



OCZYWIŚCIE...

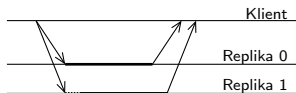
Jednocześnie nie da się spełnić wszystkich
“wybierz sobie dwa...”



Replikacja

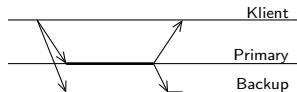
AKTYWNA

- ▶ Każda replika wykonuje każdą operację
- ▶ Transparentna



PASYWNA

- ▶ Jedna replika wykonuje operacje
- ▶ Świadoma



OPERACJA WRITE

Wpływa na następne operacje

OPERACJA READ

Nie zmienia stanu repliki

DLACZEGO LUBIMY OPERACJE READ

Nie trzeba ich uzgadniać

Mogą być wykonane tylko na jednej replice



PROBLEMY!

SPÓJNOŚĆ

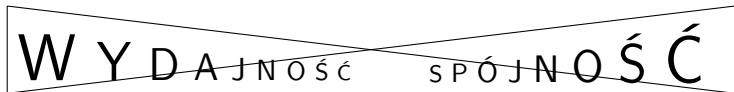
O co chodzi?

PUNKT WIDZENIA UŻYTKOWNIKA

Korzystam z aplikacji.

Chcę dostawać logiczne wyniki!

GŁÓWNA CECHA SPÓJNOŚCI



PRZYKŁADOWY PODZIAŁ NA MODELE SPÓJNOŚCI

atomowa (liniowość) czas rzeczywisty

sekwencyjna globalna kolejność (silna spójność)

przyczynowa uporządkowanie przyczynowe

procesorowa Połączenie FIFO i koherentności

FIFO (PRAM) kolejność z jednego źródła (od klienta)

podręczna (koherentna) kolejność w ramach zasobu

PRAKTYCZNE PRZYKŁADY

Czy na sali jest flipchart?



SILNA SPÓJNOŚĆ

Gwarantuje zachowanie równoważne sekwencyjnemu wykonaniu

CECHY

- ▶ Nie ma potrzeby rollbacku ani kompensacji
- ▶ Programista nie musi się przejmować problemami braku spójności
- ▶ Wymaga dostępności większości replik



Awarie się zdarzają. Oczywiście w najmniej odpowiednim momencie.

MIĘKKIE? TWARDE? BIZANTYJSKIE? JEDNOLITE?

Czy sprzęt może ulec sprzęt?

Czy proces może się zachowywać niepoprawnie?

Czy proces może się zachowywać złośliwie?

ILE AWARII MOŻNA PRZEŻYĆ?

Równoczesnych?

Następujących po sobie?

AUTOMATYCZNE ODTWARZANIE

System może sam wrócić do sprawności



O PROJEKCIE

Projekt dotyczy współczesnych technologii informacyjnych opartych na paradygmacie SOA (ang. Service Oriented Architecture).

PROJEKTY

- ▶ JPaxos - Zreplikowana maszyna stanów
- ▶ PaxosSTM - Rozproszona pamięć transakcyjna
- ▶ RESTGroup - Komunikacja grupowa dla usług sieciowych REST
- ▶ MProxy - Modularny serwer pośredniczący
- ▶ FADE - Failure Detection Service

WIĘCEJ INFORMACJI

<http://www.soa.edu.pl>

<http://www.it-soa.pl>



SUMMER@EPFL

Praktyki letnie dla studentów - na uczelni EPFL (École Polytechnique Fédérale de Lausanne)

LSR

LSR oznacza Laboratorium Systemów Rozproszonych (Laboratoire de Systèmes Répartis).

WSPÓŁPRACA Z:

André Schiper

Nuno Santos

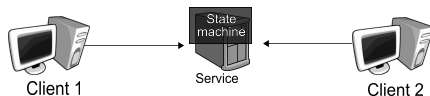


JPAXOS

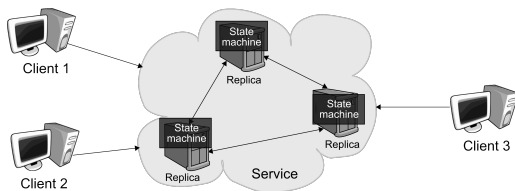


Żądanie, odpowiedź i zrzut stanu traktowane są jako byte []

TYPOWA USŁUGA



USŁUGA ZREPLIKOWANA PRZEZ JPAXOS



USŁUGA ECHO

klient wysyła ciąg znaków

usługa zamienia małe litery na duże

usługa odpowiada klientowi

DEMONSTRACJA



KONSENSUS?

Każdy z procesów wybiera tę samą wartość spośród proponowanych przez wszystkie procesy

REPLIKACJA → TOTAL ORDER RELIABLE BROADCAST

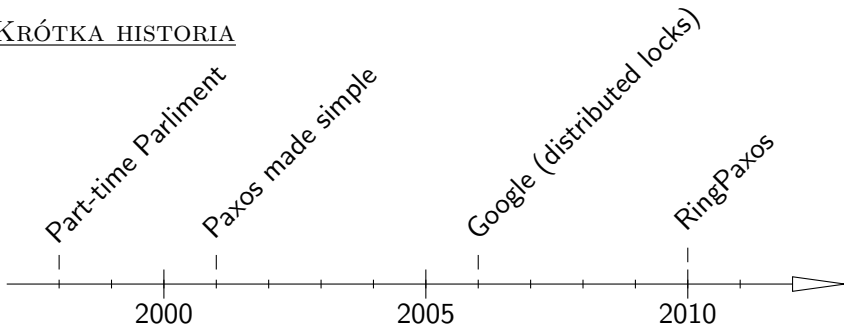
Aby zrealizować replikację, wystarczy mieć dostęp do uporządkowanego b-castu

TOTAL ORDER RELIABLE BROADCAST → KONSENSUS

Aby zrealizować uporządkowany b-cast, wystarczy rozwiązać konsensus



KRÓTKA HISTORIA



KTO ZAINWESTOWAŁ W IMPLEMENTACJĘ PAXOS

- ▶ Google – Chubby distributed lock service
- ▶ IBM – SAN Volume Controller
- ▶ Microsoft - Autopilot cluster management (Bing)



WYMAGANIA



ŚRODOWISKO

Java JRE 1.5 or later

DETERMINIZM

Usługa musi zaczynać od tego samego stanu początkowego generować na dany ciąg zapytań zawsze identyczne odpowiedzi

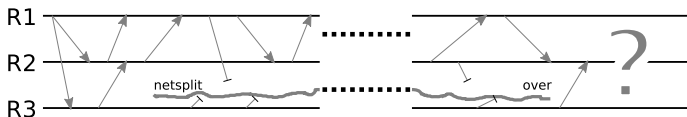
GRUPY STATYCZNE

Lista replik musi być znana podczas uruchamiania systemu

Nie jest możliwe zmienianie liczby ani położenia replik



ZAPIS STANU (SNAPSHOTTING)



MOTYWACJA

Jak uaktualnić stan repliki R3? Wybór:

- ▶ Trzymać wszystkie decyzje
- ▶ Zapewnić mechanizm transferu stanu

WYMAGANIE #3

Usługa musi:

- ▶ Przekazywać regularnie zrzut stanu
- ▶ Umieć odtworzyć się ze zrzutu stanu (innej repliki)



DOSTĘPNOŚĆ

Usługa jest dostępna wtedy i tylko wtedy gdy większość replik jest poprawna

KOMPROMIS WYDAJNOŚĆ – TOLEROWANE AWARIE

Istnieje możliwość tolerowania awarii nawet wszystkich replik kosztem znacznie zmniejszonej wydajności

WYMAGANIE #4

Użytkownik musi wybrać czy system ma tolerować awarię wszystkich replik i działać wolno, czy ma działać szybko tolerując awarię nie więcej niż połowy replik



Żądania (operacje) wysłane przez klientów (użytkowników) będą wykonane na każdej replice w tej samej kolejności



API



API (1)

CLIENT

Client class

```
void connect()
```

```
byte[] execute(byte[])
```

SerializableClient class

```
void connect()
```

```
Object execute(Object)
```

REPLICA

Replica class

```
Replica(Configuration config, int localId,  
          Service service)
```

```
void start()
```



SIMPLIFIEDSERVICE CLASS

- ▶ `byte[] execute (byte[] value)`
- ▶ `byte[] makeSnapshot ()`
- ▶ `void updateToSnapshot (byte[] snapshot)`

SERIALIZABLESERVICE CLASS (EXTENDS SIMPLIFIEDSERVICE)

- ▶ `Object execute (Object value)`
- ▶ `Object makeSnapshot ()`
- ▶ `void updateToSnapshot (Object snapshot)`

ABSTRACTSERVICE CLASS (IMPLEMENTS SERVICE)

- ▶ `byte[] execute (byte[] value, int executeSeqNo)`
- ▶ `void askForSnapshot (int lastSnapshotNextRequestSeqNo)`
- ▶ `void forceSnapshot (int lastSnapshotNextRequestSeqNo)`
- ▶ `void fireSnapshotMade(int nextRequestSeqNo, byte[] snapshot, byte[] response)`
- ▶ `void updateToSnapshot (int requestSeqNo, byte[] snapshot)`
- ▶ `void recoveryFinished()`



PLIK KONFIGURACYJNY PAXOS.PROPERTIES

process.0 = localhost:2021:3001

process.1 = localhost:2022:3002

process.2 = localhost:2023:3003

CrashModel = EpochSS

LogPath = jpaxosLogs

WindowSize = 2

BatchSize = 65507

MaxBatchDelay = 10

Network = TCP



DEMO



PYTANIA I DYSKUSJA

<http://www.it-soa.pl/jpaxos>



KONTAKT

Jan Kończak - jan.konczak@student.put.poznan.pl

Tomasz Żurkowski - tomasz.zurkowski@student.put.poznan.pl

DOWNLOAD

<http://www.it-soa.pl/download/resp/jpaxos/>

