Akka - zadanie domowe

Część 1 (5 punktów):

Z użyciem platformy Akka zaimplementuj narzędzie do monitorowania konstelacji satelitów AstraLink:

- konstelacja składa się ze 100 satelitów

- satelity identyfikowane są przez kolejne liczby całkowite od 100 do 199 włącznie

- pobranie statusu satelity realizowane jest przez wywołanie funkcji getStatus z klasy SatelliteAPI:

(tej klasy nie można modyfikować)

(komunikacja z satelitą nie wymaga utrzymywania aktywnego połączenia)

```
import java.util.Random;

public class SatelliteAPI {

    enum Status {

        OK, BATTERY_LOW, PROPULSION_ERROR, NAVIGATION_ERROR

    }


    public static Status getStatus(int satelliteIndex){

        Random rand = new Random();

        try {

            Thread.sleep(100 + rand.nextInt(400));

        } catch (InterruptedException e) {

            e.printStackTrace();

        }

        double p = rand.nextDouble();
```

```
        if (p < 0.8) return Status.OK;
        else if (p < 0.9) return Status.BATTERY_LOW;
        else if (p < 0.95) return Status.NAVIGATION_ERROR;
        return Status.PROPULSION_ERROR;
    }
}
```

Zapytania o status satelit wysyłane są przez stacje monitorujące:

- pojedyncze zapytanie pobiera status n kolejnych satelit i zawiera:

    query_id - identyfikator zapytania (unikalny w obrębie stacji monitorującej)

    first_sat_id - id pierwszego satelity w zakresie wybranym do monitorowania

    range - liczbę kolejnych satelit, dla których ma być pobrany status

    timeout - maksymalny czas oczekiwania na dane z pojedynczego satelity

- przykładowo: zlecenie (1, 120, 60, 300) ma id zlecenia = 1 oraz pobiera statusy satelit 120-179 (włącznie) ustawiając dla każdego z nich timeout = 300ms

- odpowiedź na zlecenie zawiera:

    query_id - identyfikator zapytania którego dotyczy odpowiedź

    mapę id -> status dla satelit, które zwróciły błąd, czyli status inny niż OK (nie przesyłamy tych, które zwróciły OK)

    informację jaki procent satelit zwrócił odpowiedź w wymaganym czasie

- odpowiedź wypisywana jest przez stację monitorującą na konsolę, z podaniem:

    nazwy stacji

    informacji po jakim czasie przyszła odpowiedź (w milisekundach)

    informacji ile błędów jest w odpowiedzi (długość otrzymanej listy)

    listy błędów (osobna linia na każdą parę id + błąd)

- uwaga: timeout liczony jest osobno dla każdego wywołania funkcji getStatus (nie dla całego zapytania o wiele satelit!), tzn. przy timeout=300ms odpowiedź do stacji może przyjść np. po 350ms, przy czym uwzględnia tylko te satelity które odpowiedziały w czasie do 300ms (i zwróciły błąd)

Założenia:

- system w ogólności ma mieć możliwość przyjmowania zapytań z różnych źródeł, stąd wszystkie zlecenia wysyłane są na początku do jednego głównego aktora - Dispatchera - stanowiącego punkt dostępowy do tworzonego systemu

- każda stacja monitorująca jest osobnym aktorem, przy czym zapytania musi wysyłać do Dispatchera, nie może bezpośrednio komunikować się z satelitami

- przy zwracaniu odpowiedzi do stacji dopuszczalne jest pominięcie Dispatchera, ale stacja musi dostać odpowiedź w podanym wcześniej formacie (query_id, mapa z błędami, procent zwróconych na czas odpowiedzi)

- stacja monitorująca przy tworzeniu dostaje referencję do Dispatchera (dla uproszczenia nie stosujemy recepcjonisty)

Wymagania niefunkcjonalne:

- system jest wykorzystywany jednocześnie przez wiele stacji do monitoringu, z których każda może często wysyłać zapytania, stąd w krótkim czasie może pojawiać się wiele zapytań

- system ma zwracać odpowiedzi na zapytania w jak najkrótszym czasie (należy unikać rozwiązań, które wprowadzają dodatkowe opóźnienia)

- system ma działać w trybie ciągłym (proszę zwrócić uwagę na obsługę błędów i czas życia aktorów)

- należy skorzystać z załączonego pliku konfiguracyjnego, który umożliwi jednoczesne uruchomienie liczby aktorów większej niż domyślna

Część 2 (3 punkty):

Dodaj funkcjonalność zapisywania i odczytywania statystyk błędów z użyciem lokalnej bazy danych:

- baza ma przechowywać informację ile łącznie błędów zwrócił dany satelita (bez podziału na typ błędu)

- przy uruchomieniu systemu baza inicjowana jest przez wpisanie 0 błędów dla każdego satelity (id: 100-199 włącznie)

- po każdym zleceniu monitoringu należy zaktualizować bazę

- stacja monitorująca ma możliwość wysłania drugiego typu wiadomości - zapytanie o liczbę błędów dla danego satelity

    w zapytaniu podane jest id satelity, w odpowiedzi podane jest id satelity oraz liczba błędów odczytana z bazy

    dla tych wiadomości nie mierzymy czasu wykonania oraz nie jest potrzebne query_id

- stacja monitorującą wypisuje odpowiedź na konsolę TYLKO jeśli liczba błędów jest większa niż 0

- sugerowane jest wykorzystanie bazy działającej w pojedynczym pliku (np. SQLite)

- proszę zwrócić uwagę, aby zapis do bazy nie spowalniał odpowiedzi na zapytania do satelit

Schemat (2 punkty):

Do zadania należy przygotować schemat, przedstawiający aktorów, przesyłane wiadomości oraz komunikację z bazą danych (w przypadku implementacji części 2.)

Scenariusz testowy:

- trzy stacje (proszę nadać im nazwy) wysyłają po 2 zapytania z parametrami:

first_sat_id = 100 + rand.nextInt(50); range = 50; timeout = 300

- dla części 2. należy następnie odczekać 1 sekundę i wysłać z jednej stacji zapytania o każdą satelitę (łącznie 100 zapytań)