Comunicarea asincrona prin schimb de mesaje

Tolas Ramona - 30243

Arhitectura conceptuala

Schimbul de mesaje este o modalitate asincrona de comunicare intre procese. Principaul avantaj pentru care s-a gandit implemntarea unei solutii prin transmitere de mesaje este capacitatea sistemului astfel conceput de a fi asincron. Aceasta legatura asincrona intre procese (unul din procese nu trebuie sa fie activ cand celalalt trimite mesaje si invers, procesul care trimite mesaje nu este conditionat de disponibilitatea celui care primeste) se realizeaza defapt prin doua legaturi sincrone intre producer si o coada de mesaje si consumer si o coada de mesaje. Pentru a asigura aceasta comunicare s-a folosit framework-ul RabbitMQ care face posibila aceasta comunicare intre procese si ofera un loc unde mesajele pot sa se pastreze pana vor fi consumate. Pentru a-l folosi se acceseaza o coada de mesaje cu numele cunoscut de ambii colaboratori, adica atat de proucer cat si de catre consumer.

Mai jos se prezinta diagrama conceptual pentru o aplicatie care este un sistem de gestiune a unui magazin de DVD-uri (vanzare DVD). O scurta descriere a sistemului ar include use case-uri precum : gestionarul poate sa introduca in sistem un DVD nou(DVD-ul este salvat intr-o baza de date), clientii vor primi o notificare despre introducere acestui DVD in sistem prim email. Exista de asemenea un fisier de logare unde se logheaza toate intrarile noi in sistem (toate DVD-urile adaugate) .

Conceptual sistemul este alcatuit din 2 actori principali : producer si consumer. Producer-ul dupa cum sugereaza si numele produce mesaje (adica introduce un DVD nou in sistem). In aplicatie acest lucru este realizat prin View-ul facut cu javafx care ofera o interfata utilizator care permite utilizatorul sa introduca titlul, anul aparitiei si pretul pentru un dvd si sa apeze butonul de validare a datelor introduse. Odata cu pasarea butonului se trimite un mesaj spre cealalta parte importanta a aplicatiei care este consumer-ul. Acesta trebuie sa satisfaca trei responsabilitati: sa asigure partea de persistance (sa perimita inserarea in baza de date a unui DVD nou), sa logheze intr-un fisier detaliile referitoare la noul DVD si sa trimita un email tuturor abonatilor cu datele referitoare la noul DVD. Deoarce sunt trei lucruri de facut, care urmeaza unul dupa altul s-a implementat aceasta solutie folosid design pattern-ul chain of responsabilities. In cazul de fata chain-ul are lungimea 3 si este format din trei tipuri de Task-uri : PersistanceTask, LogginTask,EmailTask.

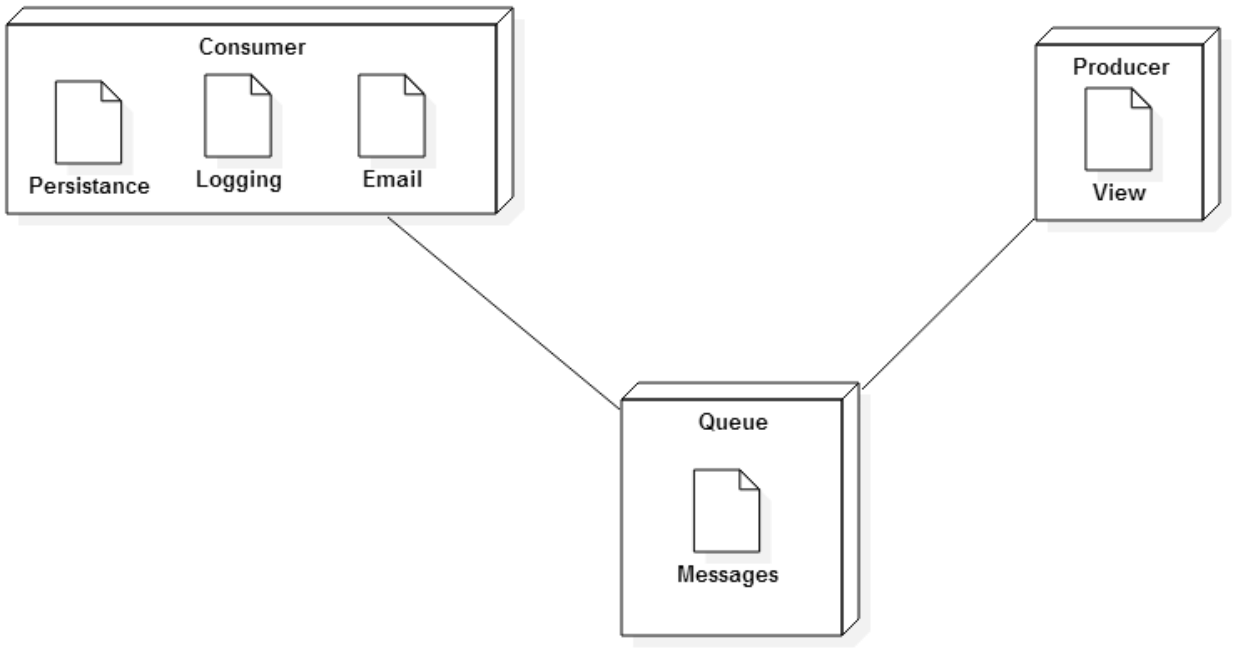
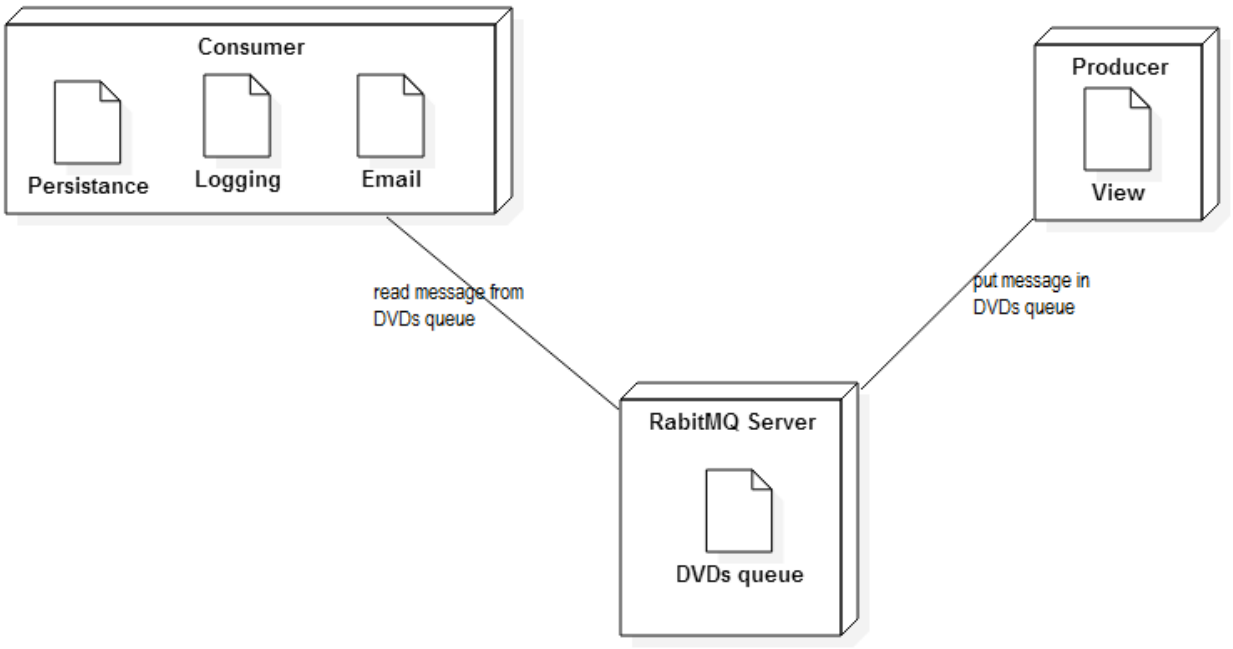


Diagrama de deployment

Continand modalitati mai concrete de comunicare, diagrama de deployment are in plus fata de cea conceptuala detalii privind serverul folosit pentru pasarea de mesaje si coada prin care cele doua entitati comunica. Ar mai fi de mentionat faptul ca mesajele transmise sunt de tipul String. Astfel, pentur a trimite un object, spre exemplu in cadrul aplicatiei, un DVD a fost nevoie de implementarea unui serializator care converteste un obiect intr-un String , apoi stie din mesajul de tip String sa scoata din nou obiectul initial. S-a implementat un serializator generic care converteste orice obiect intr-un String si returneaza un obiect de tip de tipul respectiv, folosind tipuri generice si informatii obtinute din caracteristicile de tipul Class al oricarui obiect de tipul Object (folosind caracteristica Java ca orice obiect mosteneste din clasa Object si se poate astfel apela metoda care returneaza obiectul de tip Class la runtime.



Manual de utilizare

1. Se deascarca aplicatia DVDApplication si se dezarhiveaza.
2. Se porneste un IDE (spre exemplu Eclipse) dupa care se importa proiectul (care este Maven based) in IDE.
3. Se descarca Serverul RabbitMQ si se ruleaza scriptul RAbbitMQ start.
4. Se ruleaza clasa ConsumerStart din pachetul consumer.
5. Se ruleaza clasa ProducerStart din pachetul Producer.
6. Se introduc datele unui nou DVD in interfata grafica care apare (anul aparitiei trebuie sa fie un intreg si pretul trebuie sa fie un numar real)
7. Se apasa butonul Trimite.
8. La apasarea butonului a fost trimis un email tuturor abonatilor si a fost actualizat sistemul de logare a aplicatiei precum si baza de date.