Adrian Ioncsov, an IV- IS

Tema 2
Tipuri de concurenta vis-a-vis de modificarea datelor unei aplicatii(optimista, pesimista, ultimul castiga)

Intr-un sistem sau intr-o aplicatie multiuser, concurenta este o problema importanta pe care echipa de dezvoltare ar trebui sa o rezolve. Concurenta vis-a-vis de modificarea datelor unei aplicatii face referire la ce se intampla atunci cand mai multi utilizatori ai aceleiasi aplicatii fac modificari.

Sa consideram urmatorul exemplu:

Mai multi utilizatori au acces la o baza de date in care se fac diferite inregistrari. In cazul in care doi utilizatori(sau mai multi) fac o inregistrare simultan, acest lucru nu ridica nici un fel de problema deoarece inregistrarile noi inserate nu interactioneaza unele cu celelalte. In cazul in care, doi utilizatori vor sa editeze, simultan, o inregistrare deja efectuata, se ridica problema care actiune(de editare) se ia in considerare. Un raspuns la aceasta este dependent de tipul de concurenta care este implementat in cadrul aplicatiei.

Concurenta optimista

In cazul concurentei optimiste, utilizatorii aplicatiei pot vizualiza datele si chiar sa la modifice. In momentul in care un utilizator incearca sa actualizeze o informatie, sistemul verifica daca nu cumva, documentul nu a mai fost actualizat de un alt utilizator din momentul in care acesta a fost preluat de utilizatorul actual. Daca se constata ca documentul a mai fost actualizat, incercarea de actualizare(a utilizatorului curent) este respinsa. Acest tip de concurenta prezinta si un avantaj si un dezavantaj.
Avantajul acestui tip de concurenta ar fi faptul ca toti utilizatorii au acces la documente si pot citi informatia din ele. Dezavantajul concurentei optimiste este faptul ca dupa un timp in care utilizatorul actual a facut modificarile dorite, acestea nu vor fi luate in considerare, nu vor fi salvate. Timpul este irosit, iar procesul trebuie repetat.

Concurenta pesimista

In cazul concurentei pesimiste se realizeaza blocarea datelor. In momentul in care un utilizator doreste sa editeze o inregistrare, acesta are acces la ea daca nici un alt utilizator nu are aceea inregistrare deschisa. Daca un utilizator doreste sa editeze o inregistare care este deschisa(in acel moment) si de un alt utlizator, acesta va primi un mesaj in care i se va transmite ca nu poate accesa inregistrarea deoarece este blocata pentru editare(datele sunt in curs de modificare). Si acest tip de concurenta prezinta un avantaj si un dezavantaj. Avantajul este mare in cadrul aplicatiilor in care sansa de aparitie a unei editari simultane este mare deoarece timpul nu mai este pierdut, asa cum se intampla in cadrul concurentei optimiste. Dezavantajul este faptul ca toti ceilalti utilizatori nu acces nici macar la date, in timp ce acestea sunt actualizate de un utilizator. In cadrul concurentei optimiste, toti utilizatorii aveau acces la date.

Ultimul castiga

In cazul acestui tip, nu mai este facuta nicio verificare in momentul in care un utilizator doreste sa faca o modificare. Actualizarea simultana a acelorasi date de catre utilizatori diferiti, nu mai ridica o problema deoarece acestea(datele) vor fi suprascrise. Sistemul va pastra ceea ce ultimul utilizator a modificat, deci ultimul castiga.

Un exemplu din viata reala

A doreste sa faca un transfer al unei sume din contul lui in contul lui B. In timpul transferului pot aparea urmatoarele scenarii: ce se intampla daca in timpul tranzactiei A sau B doreste sa vada cati bani mai are in cont, ce se intampla daca B doreste sa faca o extragere a unei sume din cont in timp ce tranzactia initiata de A este in desfasurare. Acestea sunt exemple de tranzactii interdependente, tranzactii care sunt dependente unele de celelalte. Strategiile de rezolvare a acestor probleme sunt cele enuntate mai sus, si anume cele trei tipuri de concurenta: optimista, pesimista si ultimul castiga.

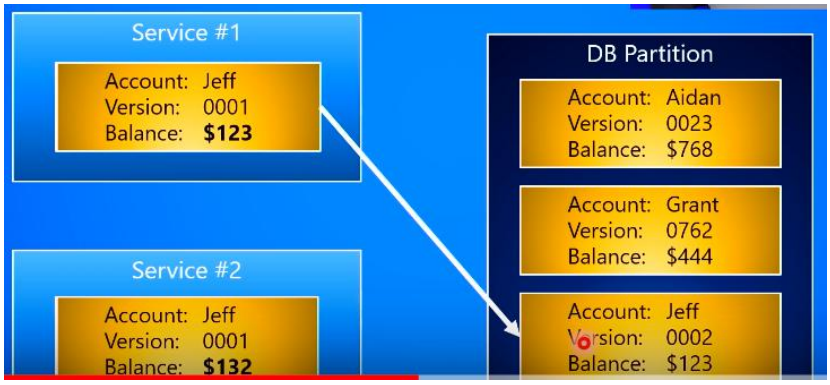
In cazul lucrului cu baze de date, printr-o tranzactie se intelege un set de comenzi SQL(de exemplu). In acest caz poata aparea urmatoarea problema. Sa presupunem ca dorim sa introducem un nou angajat in baza de date si sa ii actualizam salariul. Exista posibilitatea ca inserarea sa nu fie efectuata cu succes, urmand ca nici actualizarea nu are loc. Pentru evitarea problemelor de acest gen, se utilizeaza tranzactii(ori totul, ori nimic).

Exemplu concret in care se utilizeaza concurenta optimista

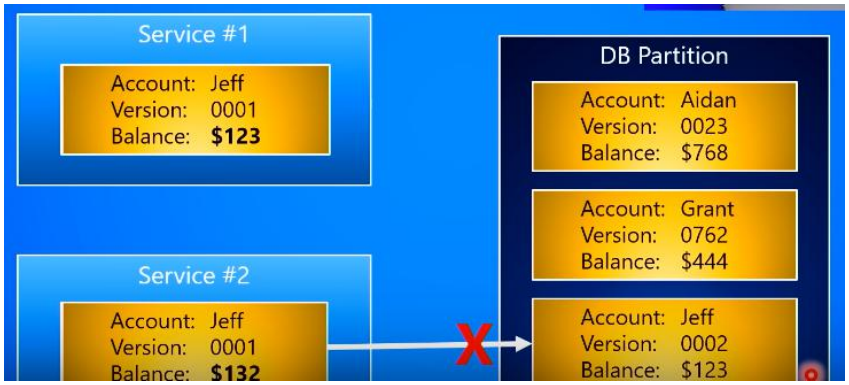
Presupunem ca exista 3 conturi, fiecare cu o suma diferita si fiecare cu o anumita versiune. Jeff are versiunea 0001 pentru ca recent si-a creat contul.
Alex(Service#1) doreste sa transfere in contul lui Jeff o suma, iar Ana(Service #2) doreste sa transfere tot in contul lui Jeff o alta suma.

Se realizeaza un request catre contul lui Jeff in vederea obtinerii informatiilor acestui cont. De remarcat faptul ca informatia esentiala este numele(contul), dar si versiunea.

Service #1 doreste sa transfere 23$ in contul lui Jeff. Acest lucru se realizeaza fara probleme, dar versiunea devine 0002.



In momentul in care Service #2 incearca sa faca transferul, se constata ca versiunile nu mai corespund. Fiind necesar un nou request catre contul lui Jeff.



Versiunea are rolul si de a asigura un transfer sigur, intervenind aici problema securitatii.

1) https://www.youtube.com/watch?v=_NThms8k5k4
2) https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/optimistic-concurrency?fbclid=IwAR0BiO5dMZd5pUQSUtopVB-33dREKRjGnBjEMwjWQd7qEKc8vz5bfGkBpQk
3) https://blogs.msdn.microsoft.com/marcelolr/2010/07/16/optimistic-and-pessimistic-concurrency-a-simple-explanation/