Elemente generale privind sistemul de prelucrare a tranzacţiilor.

Tranzacţii distribuite. Protocolul de comitere în două faze.

(text preliminar)

Vom începe cu prezentarea unei definiţii mai largi a conceptului de tranzacţie, decât cea formulată mai înainte (unitate de prelucrare secvenţială care respectă coerenţa bazei de date). Potrivit înţelegerii adoptate în (carte Gray, Reuter), o tranzacţie este o colecţie de operaţiuni efectuate asupra stării fizice şi stării abstracte a unei aplicaţii. Să remarcăm, în acest context, că prin aplicaţie se înţelege de obicei actul punerii în practică a unor principii sau idei. Definiţii oarecum diferite pentru noţiunea de tranzacţie corespund fie utilizatorilor finali, fie operatorilor sistemului de calcul în care sunt găzduite tranzacţiile.

Sistemul de prelucrarer a tranzacţiilor (prescurtat sistemul PT) pune la dispoziţie mijloacele da facilitare sau de automatizare a programării legate de aplicaţii, a execuţiei şi administrării tranzacţiilor. Aplicaţiile de prelucrare a tranzacţiilor evoluează în interiorul unei reţele de dispozitive, subsisteme şi programe care formulează fraze de interogare şi-sau operaţiuni de actualizare adresate aplicaţiei. Desigur, aplicaţiile pot fi asociate unei **baze de date distribuite.** Pornind de la "intrările" menţionate, aplicaţia întreţine o bază de date ce reprezintă o stare a sistemului din lumea reală care este modelat. Firește, ne putem imagina că ieşirile furnizate de aplicaţie, sau reacţiile acesteia, pot conduce la schimbări ale sistemului real, respectiv a stării acestuia.

Un sistem PT conţine generatoare de aplicaţii, reţele, baze de date, precum şi aplicaţii. Pentru coordonarea şi dirijarea fluxului de tranzacţii prin sistem, este folosit un ansamblu de funcţii- sau servicii-, denumit "monitor de prelucrare a tranzacţiilor", care face parte din sistemul PT.

Este util de reamintit că **sistemele de prelucrare a tranzacţiilor au avut un rol important în evoluţia calculului distribuit şi a calculului tolerant la defecte**. Prin intermediul acestor sisteme a fost iniţiată utilizarea informaţiilor distribuite în vederea asigurării fiabilităţii sistemelor şi aplicaţiilor, a disponibilităţii şi performanţelor acestora. Modelul client-server, foarte răspândit până nu demult, ca şi apelul de la distanţă al procedurilor destinat calculului distribuit, au fost de asemenea propulsate de sistemele PT. Totodată, se consideră că principalele idei unificatoare din domeniul calculului distribuit, respectiv **atomicitatea**, **consistenţa**, **izolarea** şi **durabilitatea**, care împreună poartă numele de **proprietăţi ACID** ale tranzacţiilor, au apărut în procesul de evoluţie a

sistemelor de prelucrare a tranzacțiilor. Datorită însemnătății lor, aceste proprietăți sunt succinct definite mai jos:

Atomicitate: schimbările stării unei tranzacții sunt atomice, în sensul că acestea au loc toate, sau nu are loc nicio schimbare. Schimbările sunt modificări ale bazei de date, mesaje şi acțiuni asupra subsistemelor din mediul extern.

Consistență: o tranzacție constitue o transformare corectă a stării bazei de date. Acțiunile întreprinse în grup nu încalcă niciuna din restricțiile de integritate specifice stării menționate. De aici rezultă că tranzacției trebuie să îi corespundă un program corect.

Izolare: chiar dacă tranzacțiile sunt executate în mod concurent, fiecare tranzacție T "vede" execuția celorlalte tranzacții ca fiind produsă fie anterior față de T, fie după T, dar nu ambele.

Durabilitate: odată ce o tranzacție a fost executată cu succes (deci a fost "completată" sau "a comis"), schimbările pe care ea le-a produs în starea bazei de date nu vor fi afectate de defecțiuni.

Evoluția globală a unei tranzacții, cu respectarea proprietăților ACID, poate fi descrisă după cum urmează. Programul de aplicație declară debutul unei tranzacții noi prin comanda Begin-Work(). În continuare, toate operațiile efectuate de program vor face parte din această tranzacție. În acelaşi timp, toate operațiunile executate de alte programe în legătură cu programul de aplicație sunt parte a tranzacției de bază. Prin inițierea comenzii Commit_Work, programul declară că tranzacția constitue o transformare completă şi corectă. Rezultatele execuției tranzacției – sau efectele acesteia- devin durabile, de îndată ce tranzacția comite.

Pe de altă parte, este posibil ca pe parcursul execuției tranzacției unele operațiuni să fie eronate. În acest caz, aplicația poate „distruge" toate operațiunile, prin comanda Rollback_Work(). Mai observăm că atunci când în timpul execuției unei tranzacții are loc o defecțiune de natură tehnică, sistemul de gestiune poate "trimite înapoi", sau „returna" tranzacția. Pentru încadrarea transformărilor ACID se folosesc comenzile Begin-Commit sau Begin-Rollback.

Elementele de mai sus reprezintă o abordare convenabilă pentru realizarea **aplicațiilor distribuite**. Fiecare modul al aplicației va fi o tranzacție sau o sub-tranzacție. Atunci când execuția se desfăşoară cu succes, tranzacția comite şi toate modulele trec în o stare durabilă nouă.. În situația în care apar erori sau alte incidente, modulele tranzacției vor fi readuse automat în starea inițială, cea de la începutul execuției tranzacției. Întrucât operațiunile de comitere şi de revenire se bazează pe o logică „automată", realizarea modulelor reclamă folosirea de metode cu semantică elementară pentru descrierea defecțiunilor.

Tranzacții distribuite

În cadrul unei **baze de date distribuite în nodurile unei rețele**, accesul la multitudinea de obiecte de date se realizează de obicei prin intermediul tranzacțiilor. Se înțelege că tranzacțiile trebuie să respecte regulile ACID.

Deoarece **tranzacțiile care sunt executate într-un sistem distribuit pot prelucra fie date rezidente într-un singur loc, fie date amplasate în sisteme distincte**, vom distinge **tranzacții locale** şi **tranzacții globale**. Primele au acces la informațiile conținute în o singură bază de date locală şi pot prelucra aceste informații. Tranzacțiile globale realizează accesul la informații amplasate în mai multe baze de date locale şi, desigur, pot prelucra aceste informații.

Respectarea proprietăților ACID în procesul execuției **tranzacțiilor locale** reclamă, practic, participarea tuturor componentelor sistemului de gestiune a bazei de date implicate în gestiunea tranzacțiilor. Asigurarea consistenței unei tranzacții oarecare trebuie dată de programatorul de aplicație, care codifică tranzacția. Pentru respectarea atomicității tranzacției, chiar dacă apar defecțiuni, se poate folosi tehnica de modificare „amânată", sau tehnica modificării imediate a bazei de date. În primul caz, modificările aduse bazei de date sunt înregistrate într-un „jurnal", însă execuția tuturor operațiunilor de înregistrare din tranzacție este amânată până la comiterea parțială a tranzacției (reamintim că prin comitere parțială înțelegem situația în care acțiunea finală a tranzacției a fost executată). Modificarea imediată a bazei de date constă în efectuarea schimbărilor bazei de date atunci când tranzacția este încă în stare activă; schimbările aduse de tranzacții ce se găsesc în stare activă se numesc **modificări ne-comise**. Responsabilitatea operațiunilor de mai sus revine **controlorului** (sau „manager"-ului) **de tranzacții**. În fapt, acest controlor, prin observarea memoriei secundare (disc), urmăreşte valorile vechi ale tuturor obiectelor de date asupra cărora tranzacția a efectuat înregistrări, iar dacă tranzacția nu a completat execuția sa, restaurează vechile valori astfel încât aparent tranzacția nu ar fi fost niciodată executată. Durabilitatea este în sarcina **controlorului de revenire** (sau de recuparare). Prin grja acestui sub-sistem, modificările aduse bazei de date de către o tranzacție sunt înregistrate pe disc înaintea completării tranzacției. De asemenea, este asigurată disponibilitatea unei informații suficiente privind modificările efectuate şi înregistrate pe disc, astfel încât actualizările făcute să poată fi reconstruite atunci când sistemul de baze de date este repornit după o defecțiune. În sfârşit, **controlorul concurenței** este sub-sistemul care, printre altele, asigură respectarea proprietății de izolare. Printre tipurile de planificare ce contribue la sigurarea acestei proprietăți menționăm serializabilitatea de conflict şi serializabilitate de vedere.

În cazul **tranzacțiilor globale**, menținerea proprietăților ACID este sensibil mai complicată, având în vedere faptul că la execuția unei planificări de tranzacții iau parte mai multe noduri ale unei

rețele. Defecțiunile apărute în unul sau mai multe din noduri, ca şi defecțiunile canalelor de comunicație, vor conduce aproape sigur la prelucrări incorecte ale informațiilor.

Din arhitecura unei **baze de date distribuite** fac parte două componente principale: **controlorul de tranzacții** şi **coordonatorul tranzacțiilor.** Aceste sub-sisteme sunt **amplasate în nodurile rețelei de calculatoare**.

Funcțiunea **controlorului de tranzacții** constă în **respectarea proprietăților ACID** ale tranzacțiilor care prelucrează informații memorate **în nodul cu care controlorul este asociat.** Deoarece avem în vedere o bază de date distribuită, se înțelege că în rețea vor fi executate şi tranzacții globale, care efectuează operațiuni asupra datelor depuse în mai multe noduri. În acest proces vor colabora controloare de tranzacții amplasate în nodurile corespunzătoare. Într-un nod dat al rețelei, în care se desfăşoară prelucrarea concurentă a unui număr de tranzacții, **controlorul de tranzacții** ocupă un **loc specific în activitatea algoritmului de control al concurenței**. De asemenea, acest sub-sistem are **responsabilitatea conținutului jurnalului** menținut în vederea revenirii după defecțiuni.

La rândul său, **coordonatorul de tranzacții asociat unui nod**, **dirijează execuția tranzacțiilor locale şi globale ce sunt inițiate şi funcționează în nodul respectiv**. **Funcția de coordonare menționată este specifică numai mediului distribuit**. În sarcina coordonatorului intră **inițierea execuției fiecărei tranzacții**, **segmentarea tranzacțiilor** în scopul distribuirii sub-tranzacțiilor obținute în noduri adecvate, precum şi **coordonarea încheierii activității fiecărei tranzacții**. Subliniem faptul că tranzacțiile globale fie comit în toate nodurile în care sunt executate sub-tranzacțiile din care sunt formate, fie sunt abandonate în toate nodurile.

În legătură cu defectele ce apar uneori în sistemele centralizate, inclusiv în cele care prelucrează baze de date, menționăm erorile de echipament („hardware”), erorile programelor, deterioararea sistemelor de discuri magnetice. Tema defecțiunilor posibile într-un **sistem distribuit** care găzdueşte o bază de date are particularități evidente. Deterioarea componentelor unui nod, defecțiuni ale liniilor de comunicație, perturbarea manierei inițiale de partiționare a rețelei, coruperea mesajelor prin erori sunt tipurile principale de defecte ce au loc în mediu distribuit. Atunci când se defectează mediul de comunicație, mesajele transmise între noduri sunt de obicei re-rutate. În ipoteza dispariției conexiunii între două noduri, rețeaua va fi partiționată din nou. Erorile care apar în mesaje sunt tratate de protocoalele de control al procesului de comunicație, de exemplu TCP/IP.

Într-un paragraf imediat precedent, am afirmat că la sfârşitul execuției unei tranzacții globale- sau distribuite-, aceasta trebuie fie să comită, fie să fie abandonată în toate nodurile rețelei. În o

formulare mai puțin tehnică, rezultatul final al execuției tranzacției trebuie să întrunească „acordul" tuturor nodurilor, fapt în care constă respectarea proprietății de atomicitate a tranzacției.
Pentru asigurarea proprietății de atomicitate, coordonatorul tranzacției menționate este prevăzut cu un **protocol de comitere**. Unul dintre sistemele de programe cele mai frecvent folosite, din această clasă, este **protocolul de comitere în două faze (prescurtat, 2PC).** O variantă mai complexă este reprezentată de protocolul de comitere în trei faze (3PC).

Protocolul de comitere în două faze (2P Commit) - structură, funcționare.

În cele ce urmează, **în contextul unei baze de date distribuite**, va fi descrisă activitatea protocolului 2PC până la finalizarea unei tranzacții, respectiv până la comiterea acesteia. De asemenea, protocolul 2PC îndeplineşte funcții specifice în situațiile de defect, de revenire după defecte şi de control al concurenței tranzacțiilor. Modul de realizare a acestor ultime funcții va fi prezentat ulterior.
Acțiunea de comitere.
Fie o **tranzacție T** care intră în lucru în **nodul Ni** , unde **coordonator este TCi**. Vom distinge două faze în desfăşurarea procesului de încheiere a execuției tranzacției T. Aşa cum se va vedea, procesul se poate sfârşi fie prin comiterea tranzacției, fie prin abandonarea acesteia. Protocolul 2PC intră în lucru atunci când, prin decizia coordonatorului TCi, tranzacția T încheie (sau completează) execuția sa, respectiv atunci când toate nodurile în care a fost prelucrată T anunță coordonatorul TCi că tranzacția a fost completată.
Prima fază. Odată cu intarea în faza de început a procesului, coordonatorul TCi introduce în jurnal înregistrarea cu conținutul <prepare T> şi, totodată, înmagazinează (sau, cu un termen folosit frecevnt, „forțează") conținutul jurnalului în memoria „stabilă" (pe discuri magnetice) a sistemului. În continuare, TCi transmite mesajul privind pregătirea tranzacției tuturor nodurilor în care aceasta a fost prelucrată. În aceste noduri, în urma recepționării mesajului, fiecare controlor de tranzacții asociat decide asupra comiterii porțiunii de tranzacție care îi corespunde. În situația în care răspunsul este afirmativ, controlorul de tranzacții din nod introduce în jurnal înregistrarea cu conținutul <ready T>, iar apoi „forțează" conținutul jurnalului în memoria stabilă, inclusiv toate înregistrările din jurnal care se referă la tranzacția T. Apoi, controlorul de tranzacții transmite coordonatorului TCi mesajul „ready T". Atunci când decizia este negativă, controlorul de tranzacții din nod înscrie înregistrarea <no T > în jurnal, în continuare transmițând coordonatorului TCi mesajul „abort T".
A doua fază. Coordonatorul de tranzacții TCi poate stabili dacă tranzacția T va fi comisă sau abandonată, de îndată ce a primit răspunsuri de la toate nodurile la nesajul său de pregătire a

tranzacției. Decizia respectivă poate fi luată de TCi şi atunci când de la transmiterea mesajului <prepare T> a trecut un timp dat.
Dacă toate nodurile ce participă în proces au răspund prin mesajul „ready T", tranzacția T poate fi comisă, în jurnal se introduce înregistrarea <commit T>, iar conținutul jurnalului este înmagazinat în memoria stabilă. În situația contrară, T este abandonată, urmând ca în jurnal să fie înscrisă informația <abort T>, iar jurnalul- forțat în memoria stabilă. Potrivit exprimării din literatura de specialitate (vezi Silberschatz, pg 216), în acest moment evoluția tranzacției este definitivă. Ca urmare, sistemul coordonator TCi transmite tuturor nodurilor ce au luat parte la execuție mesaju „commit T" sau „abort T". Primind unul din aceste mesaje, nodurile îl înscriu în jurnal.

Să adăugăm câteva comentarii față de elementele de mai sus. În primul rând, de îndată ce controlor de tranzacții emite un mesaj „ready T" către coordonator, tranzacția se va găsi în situația numită „ready state" în nodul corespunzător. În fapt, mesajul transmis coordonatorului este o „promisiune" formulată la nivelul nodului privind invitația de comitere sau de abandonare exprimată de TCi. În acest context, trebuie remarcat că pentru a formula un asemenea răspuns, este necesară înscrierea unei informații corespunzătoare în memoria stabilă. Dacă informația menționată nu va fi disponobilă, este posibil ca în situația în care, în urma apariției unui defect, nodul în cauză iese din funcțiune după ce a transmis mesajul „ready T", intenția de comitere sau abandonare să nu mai poată fi realizată în mod corespunzător.
În al doilea rând, tot aici vom mai observa că drepturile de închidere ale tranzacției rămân valabile până la comiterea acesteia, precum şi faptul că T poate fi abandonată în orice moment, anterior celui în care controlorul de tranzacții din nod trimite coordonatorului mesajul „ready T".
În al treilea rând, vom sublinia faptul că de îndată ce un nod a declarat abandonarea tranzacției T prin răspunsul „abort T", evoluția tranzacției este definitivă, drept urmare a condiției de unanimitate pentru comitere. În legătură cu aceasta, observăm că decizia de abandonare a tranzacției T poate fi luată în mod unilateral de către coordonatorul TCi, al nodului în care s-a produs prelucrarea tranzacției. Decizia finală asupra evoluției tranacției T este luată în momentul în care TCi înscrie hotărârea de comitere sau abandonare adoptată în jurnal, apoi în memoria stabilă. Unele implementări ale protocolului 2PC prevăd ca, la sfârşitul celei de a doua faze, controlorul de tranzacții din nod să transmită coordonatorului mesajul „acknowledge T", urmând ca după recepționarea mesajului din partea tuturor nodurilor, TCi să introducă în jurnal înregistrarea <complete T>.
Funcționarea protocolului de comitere în două faze poate fi descrisă în o manieră alternativă, prezentată succint în cotinuare. Fie o bază de date distribuită în N noduri. Mai jos, vor fi folosite notațiile: <……> pentru reprezentarea înregistrărilor, iar "……." pentru mesaje. Admitem că din

componența unui nod “i” fac parte o unitate centrală de prelucrare UCPi, o memorie centrală MCi, o memorie de tip cache MCHi, în nod fiind înmagazinat codul tranzacțiilor locale şi subtranzacțiilor ce aparțin unor tranzacții globale, precum şi sistemele de programe corespunzătoare controlorului Tctri, respectiv coordonatorului de tranazacți TCi. La elementele de mai sus se adaugă memoria secundară MSECi. În continuare, este prezentată schematic o desfăşurare posibilă a activității protocolului 2P Commit, la momentele de timp t1, t2, etc.

Prima fază

t1 TCi <prepareT> -> jurnal
t2 TCi <jurnal> -> memoria stabilă MSECi
t3 TCi <prepareT> -> noduri
t4 Tctrj decizie privind comiterea porțiunii proprii de tranzacție
t5 decizie = Da, Tctrj <readyT> -> memoria stabilă,
sau
decizie = Nu, Tctrj <noT> -> jurnal
t6 decizie = Da, Tctrj mesaj “readyT” -> TCi,
sau
decizie = Nu, Tctrj mesaj “abort” - > TCi.

A doua fază

t7 TCi analizează răspunsurile Tctrj din momentul t6
t8 toate Tctrj -> mesaj „readyT”, TCi <commitT> -> jurnal,
sau
nu toate Tctrj -> mesaj „ready”, TCi <abort> -> jurnal
t9 TCi <jurnal> -> memoria stabilă MSECj
t10 TCi <commitT> în toate nodurile, sau <abortT> în toate nodurile
t11 fiecare Tctrj participant în prelucrarea tranzacției <commitT> -> jurnal sau
<abortT> -> jurnal.

În ipoteza că unul sau mai multe din nodurile rețelei (respectiv dintre controloarele de tranzacție) nu „votează” favorabil, tranzacția nu poate comite. În această situație, pentru tranzacție trebuie să se ajungă în final la decizia „Abort_Work()”. Jurnalul tranzacției va fi citit înapoi, sub controlul coordonatorului de tranzacții. În acest proces, pentru fiecare înregistrare din jurnaş coordonatorul apelează nodul (controlorul) care a operat ultima actualizare, determinându-l să inițieze o operațiune de „distrugere” a informației respective. Atunci când în procesul inspectării „inverse” a jurnalului se ajunge în punctul de început al tranzacției, respectiv acolo unde trebuie să se afle comanda Begin_Work(), va fi efectuată operațiunea de „distrugere” a acestei ultime comenzi, apoi în jurnal se introduce o înregistrare de abandonare. În sfârşit, în jurnal este introdusă o nouă înregistrare, care consemnează „completarea” abandonării. Mai menționăm că jurnalului tranzacției i se asociază un controlor propriu, a cărui sarcină este de a gestiona înregistrările din jurnal, prin intermediul unei liste înlănțuite, în care fiecărei înregistrări i se atribue o cheie unică. Unul din obiectivele pentru care în procesul de inspectare a conținutului jurnalului se foloseşte o listă înlănțuită este asigurarea eficienței operațiunilor de „distrugere” evocate mai sus.