THESE

présentée

à l'Université des Sciences et de la technologie
HOUARI BOUMEDIENE

POUR OBTENIR LE GRADE DE
MAGISTER
EN
INFORMATIQUE

PAR

Mohand Amokrane AITOUN

ÉTUDE ET ÉVALUATION DES TECHNIQUES DE CONTROLE D'ACCÉS CONCURRENTS DANS LES S.G.B.D. RÉPARTIS - DÉVELOPPEMENT DE CONTROLEURS PRÉVENANT LES INTERBLOCAGES

SOUTENUE LE **9 SEPTEMBRE 1984** DEVANT LA COMMISSION D'EXAMEN

B. SANSAL : Professeur à l'U.S.T.H.B. — PRESIDENT

F. ANDRE : Professeur à l'I.R.I.S.A. (Rennes I)
S.A. LARIBI : Professeur à l'U.S.T.H.B.
A. AINOUCHE : Maître de Conférences à l'U.S.T.H.B. — EXAMINATEURS

- REMERCIEMENTS -

Je tiens à exprimer mes remerciements à Monsieur B. SANSAL, Professeur à l'U.S.T.H.B. qui malgré ses nombreuses occupations me fait l'honneur de présider ce Jury.

Monsieur A. AINOUCHE, Maître de Conférences à l'U.S.T.H.B. a bien voulu examiner ce travail, qu'il trouve ici l'expression de mes sincères remerciements.

Madame F. ANDRE, Professeur à l'Université de RENNES I a, avec l'amabilité qui la caractérise, mis sa compétence à ma disposition. L'objet de mon étude n'aurait pu être maîtrisé efficacement sans les stages que j'ai effectués à RENNES au sein de l'IRISA. Je lui exprime ma gratitude pour les conseils et les encouragements qu'elle n'a cessés de me prodiguer et sans lesquels ce travail n'aurait pu être mené à terme.

J'ai toujours trouvé à mes côtés Monsieur S.A. LARIBI, Professeur à l'U.S.T.H.B. et Directeur de Recherche, dont j'ai, tout au long de ce travail, apprécié l'aide efficace et cordiale. Pour cela, ainsi que pour m'avoir initié à la pratique de la recherche, m'avoir fait bénéficier de son expérience, de sa compétence et de ses appréciations constructives, je tiens à lui exprimer ma profonde gratitude et ma reconnaissance la plus sincère.

Les moyens de l'ENSI ont été mis à notre disposition, grâce à la compréhension que nous avons rencontré auprès de Monsieur HASSANI, ancien Directeur Général, et l'amabilité de mon ami A. BENHEDOUGA, Chef de Projet au sein de l'UPMSI, qu'ils en soient remerciés vivement et trouvent ici l'expression de ma sincère reconnaissance.

Que tous ceux, collègues et amis m'ont encouragé et soutenu le long de la réalisation de ce travail, soient assurés de ma gratitude la plus profonde.

S O M M A I R E

Pages

Pages

Pages

Pages

INTRODUCTION

L'évolution des applications vers la manipulation de bases de données, l'essor des traitements de type transactionnel, font du partage et de la protection des problèmes essentiels des systèmes actuels. Ceci, joint au besoin de plus en plus ressenti de distribuer les traitements, rend fondamental le contrôle des accès concurrents dans un système de gestion de bases de données réparties.

Dans un système transactionnel accédé par des processus concurrents, une des fonctions primordiales est d'assurer la concordance des informations avec le monde réel. Ainsi, les accès simultanés d'un grand nombre d'utilisateurs à une base de données, qu'elle soit centralisée ou distribuée, provoquent des interactions pouvant nuire à la cohérence.

Le contrôle des accès concurrents, objet de notre étude et partie d'un système de gestion de bases de données, préserve la cohérence et l'intégrité d'une base de données. Il offre un partage des données complètement transparent à l'utilisateur et permet un haut niveau de parallélisme et une synchronisation des accès simultanés conflictuels.

Les nécessités de performances ont motivé beaucoup de travaux récents sur le contrôle. Cependant, le nombre de propositions ne cesse de croître, mais chacune porte en soi son contexte propre, sa mise en exergue de certains points particuliers et sa négligence pour d'autres, ses hypothèses et sa terminologie. Tous ces facteurs rendent fort mal aisée une comparaison des différentes études.

L'étude que nous développons dans cette thèse, effectue une analyse synthétique des techniques de contrôle, propose un mode de description unique et précise le champ d'action de la méthode, afin de guider l'ensemble des choix possibles pour une application donnée et enfin, met en oeuvre un système de Contrôle des Accès Concurrents (CAC).
CAC réalisé au Département d'Informatique et Logique Mathématique, se situe dans un environnement transactionnel centralisé avec pour tâche la multi-utilisation du système de gestion de bases de données, EASY, en voie de développement. L'ensemble des contrôleurs CAC est fondé sur 2 grandes

classes de techniques de contrôle : l'ordonnancement par estampillage et le verrouillage à deux phases. La première classe fournit des stratégies laissant s'exécuter les demandes "usager" (transactions), en vérifiant que les accès à une donnée sont sérialisés suivant un ordre prédéfini au lancement des transactions. La seconde classe consiste à éviter la génération d'exécution incorrecte en faisant attendre les opérations conflictuelles; cette classe soulève le phénomène d'interblocage (verrou mortel) dû à l'attente réciproque entre transactions.

Ci-dessous, nous présentons sommairement le contenu de notre étude :

En premier lieu, nous définissons quelques concepts de base fondamentaux; nous analysons les problèmes posés par les accès conflictuels, puis nous donnons les solutions apportées. Nous précisons enfin la classification des mécanismes de résolution en faisant apparaître deux types de résolution du verrou mortel : la détection-guérison et la prévention.

Dans le chapitre 2, nous effectuons une synthèse des principaux travaux sur le contrôle d'accès concurrents dans les systèmes de gestion de bases de données réparties en fournissant de façon unifiée un mode de fonctionnement de chaque technique et en précisant les domaines d'application.

Nous avons dressé au chapitre 3, une liste de critères qualitatifs de jugement de ces techniques. Pour chacun de ces critères, nous examinons le comportement des différentes techniques et nous établissons une classification générale des mécanismes étudiés.

Le chapitre 4 aborde un aspect complémentaire du précédent : l'aspect quantitatif. Pour chacun des critères quantitatifs définis, nous évaluons les performances des divers mécanismes.

En deuxième lieu, nous présentons le développement du système de Contrôle des Accès Concurrents - CAC -.

b - Aucun nouveau verrouillage ne sera effectué par une transaction sur un granule déjà vérrouillé par elle, sauf pour le verrouiller en mode exclusif s'il était déjà verrouillé en mode partagé.

c - Aucun granule ne reste verrouillé par une transaction après sa terminaison.

2.2.3 - Description du modèle BA

La figure 2.2.3 décrit le système BA.

INITIAL, TBF, EXECUTION, ATTENTE et VALIDATION ont un rôle identique à celui des stations du contrôleur AR.

BLESSEE : est un module qui contient les transactions 'blessées' : les transactions 'blessées' sont les transactions détruites par celle détectrice du conflit. Le mécanisme obéit aux règles suivantes :

a - une transaction 'blessée' est autorisée à poursuivre son exécution tant qu'elle ne se met pas en attente d'un autre granule.

b - si la transaction 'blessée' est en attente au moment de la blessure, elle est rejetée.

c - si la transaction 'blessée' entre en conflit avec une autre transaction, le même mécanisme est réappliqué, à savoir :
- . si la transaction est plus ancienne que sa concurrente, la transaction 'blessée' attend et la concurrente est 'blessée' à son tour.
- . si la transaction 'blessée' est plus jeune, elle est rejetée, libérant immédiatement la transaction qui l'avait blessée.

RECHGRANU : a le même rôle que dans 2.2.2, mais la gestion des conflits dans cette station est différente :

```
Procédure 'BLESSEE-ATTENTE' BA;
    Si T2 < T1 (T2 plus vieille que T1)
        alors  T1 est blessée et mise dans 'BLESSEE'
        sinon  T2 attend dans la file 'ATTENTE'
    fsi
fin BA;
```

où T1 et T2 ont la même signification que dans le paragraphe 2.2.2.

Le processus ou transaction détecteur du conflit peut tenter une action de destruction du processus concurrent. Cette tentative de destruction (ou blessée) du processus concurrent a lieu s'il a l'estampille la plus ancienne.

Dans cette méthode, une transaction 'ancienne' ne se met jamais en attente, si ce n'est de la réponse à la tentative de destruction; tout risque d'attente infinie est évité, car la priorité d'un processus croît avec sa durée d'existence.

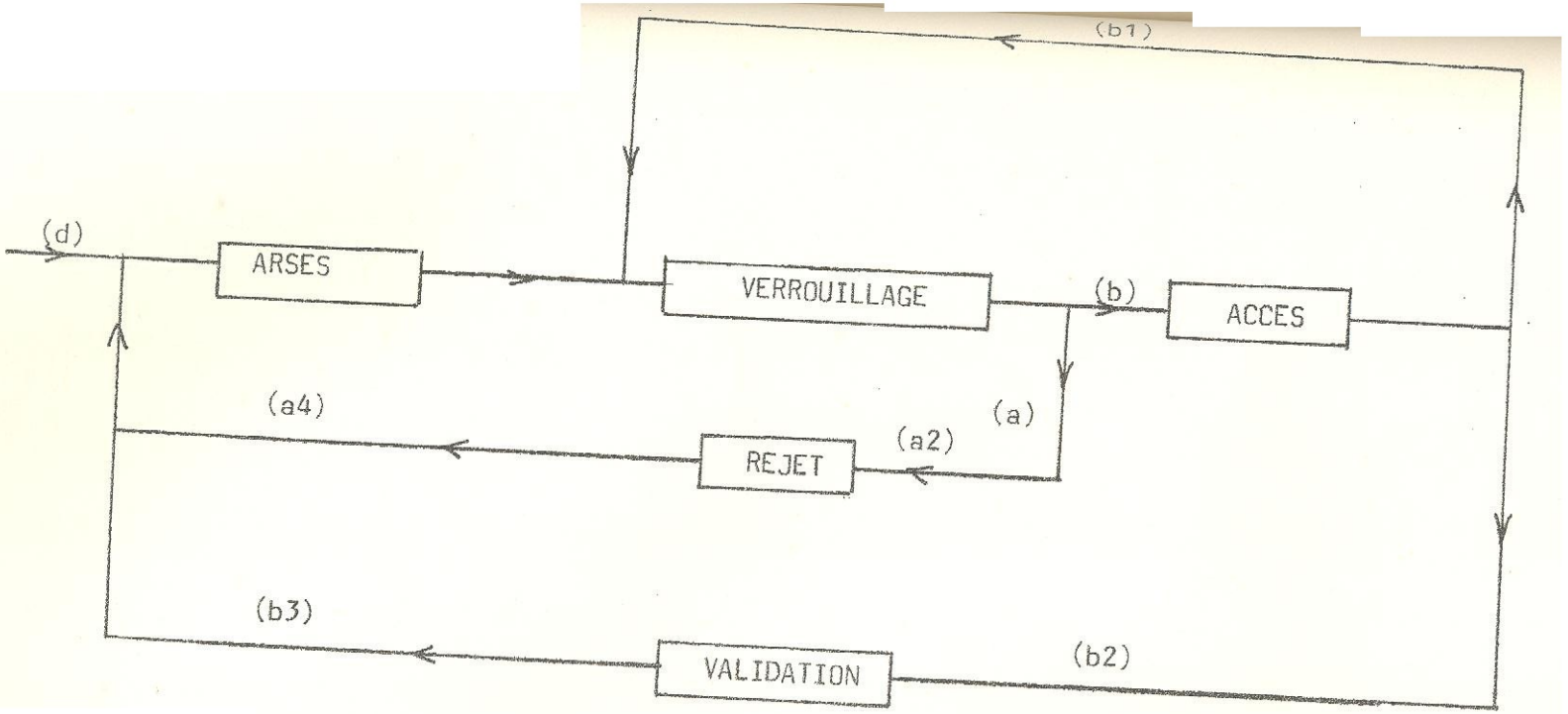

Figure 2.1.2 : Système de contrôle OE

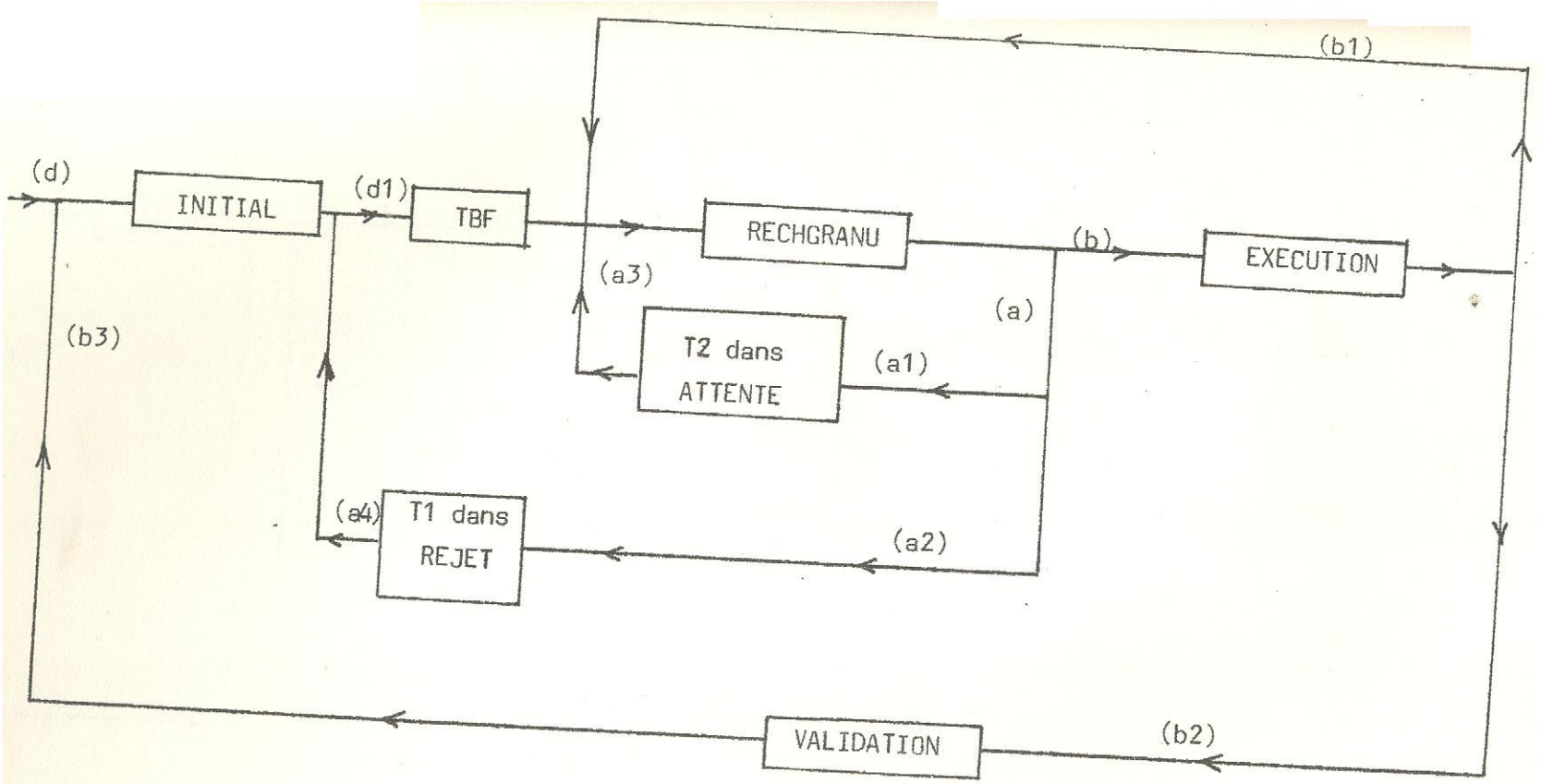

Figure 2.2.2 : Système de contrôle 2PLP - AR

Légende des figures 2.1.2, 2.2.2 et 2.2.3

(d) : arrivée des transactions

(a) : granule occupé

(b) : granule libre

(a1) : attente de granule pour le compte d'une transaction

(a2) : rejet ou attente après avortement de la transaction

(a3) : les transsctions en attente de granules sont libérées de 'ATTENTE' pour continuer leur exécution

(a4) : celles en attente de la transaction sont libérées de 'REJET' pour reprendre leur exécution à leur début

(b1) : accès au granule suivant par la transaction

(b2) : fin de transaction, la transaction rentre dans la phase de validation

(b3) : libération dans la session de transations

(d1) : les transactions deviennent bien-formées dans TBF

(a5) : la transaction conflictuelle autre que celle détectrice du conflit est dirigée dans BLESSEE.

Les résultats de comparaison obtenus après la mise en oeuvre des différents contrôleurs prévenant l'interblocage, nous montrent que la technique AR autorise la formation de chaînes d'attente de la forme :

$$T_i \xrightarrow{\text{Attente}} T_j \xrightarrow{\text{Attente}} T_{j+k} \xrightarrow{\text{Attente}} \cdots \xrightarrow{\text{Attente}} T_n$$

avec $T_i < T_j < T_{j+k} < \cdots < T_n$

La technique BA autorise la formation de chaînes d'attente de la forme :

$$T_i \xrightarrow{\text{Attente}} T_j \xrightarrow{\text{Attente}} T_{j+k} \xrightarrow{\text{Attente}} T_j \xrightarrow{\text{Blessée}} T_l \cdots \xrightarrow{\text{Blessée}} T_n$$

avec $T_i > T_j > T_{j+k} < T_l < \cdots < T_n$

Ces deux techniques interdisent donc la formation de cycle dans la station 'ATTENTE'; la technique BA donne la priorité à la transaction la plus ancienne et limite le nombre de transactions rejetées à tort; par contre, AR, donne la priorité à la transaction la plus jeune et provoque trop de rejets pour être perfomante.

La limitation du nombre de transactions rejetées apporte à la technique BA un gain de performance par rapport à AR dans les sessions à faibles charges (nombres de transactions faibles), mais induit en contrepartie un accroissement de la longueur des chaînes d'attente, donc du nombre de granules inutilisés : la méthode BA n'autorise un parallélisme qu'entre les transactions non concurrentes.

La technique BA donne donc un meilleur temps de réponse que la technique AR si la probabilité de conflit est faible.

L'implémentation des techniques de prévention avec verrouillage deux phases AR - BA qui sont moins coûteuses en messages que les techniques de détection, rend ces approches de contrôle de concurrence d'accès beaucoup plus comparables sur ce critère avec la technique d'ordonnancement par estampillage OE :

Les techniques AR - BA donnent un meilleur temps moyen de réponse qu
la technique OE, si la probabilité de conflit est très forte /⁻PARTIE I_7,
ou si le nombre moyen d'actions par transaction est petit.

Par contre, la technique OE devient meilleure si le nombre moyen
d'actions par transactions augmente ou si sa loi de distribution est géo-
métrique.

Dans la première partie, nous avons analysé plusieurs techniques de contrôle d'accès concurrents par rapport à des paramètres pris individuellement.

Cependant, nous avons précisé qu'il n'est pas possible de trouver une technique optimale pour un ensemble de critères qualitatifs et quantitatifs simultanés. Cette étude ne met pas en évidence l'adéquation d'une approche de contrôle à une situation précise et concrète.

Le développement du système transactionnel CAC, variété de contrôleurs des accès concurrents aux informations prévenant l'interblocage, constitue une couche au-dessus des méthodes d'accès aux systèmes de gestion de bases de données augmentant le parallélisme d'exécution tout en préservant la cohérence des données en dépit des pannes /PARTIE II/.

La mise en oeuvre de CAC n'a couvert qu'une partie des techniques de contrôle d'accès, vu la grande diversité des mécanismes présentant des avantages et des inconvénients suivant le type d'application.

La cohabitation de ces contrôleurs dans le système de gestion de bases de données EASY offrira la possibilité de choix de l'un d'eux pour une application spécifique.

La contribution de CAC dans EASY est certaine dans la mesure où il permet une multi-utilisation et fournit aussi des résultats préliminaires à l'analyse du comportement de l'accès concurrent en environnement réparti.

La conception et la réalisation du système CAC résument l'étude des performances à un critère de base : le débit du système.

Lorsque le système est faiblement chargé (niveau de cohérence bas) les contrôleurs détectant les conflits /BERG 84/ sont meilleurs. Lorsque la charge devient forte, l'impact d'une régulation effective (limitation de chaînes d'attente) ou d'une auto-régulation (abandons provoqués par une technique d'accès concurrents) est considérable sur le comportement du système.

Les abandons à 'tort' provoqués par les contrôleurs prévenant l'interblocage /~PARTIE II_7 ont un effet bénéfique sur le débit du système.

CAC prévoit aussi le risque de 'famine' : construits sur l'estampillage des transactions, les contrôleurs préviennent ce risque. Par la suite, l'âge des transactions est modulé par un système de priorité qui permet de favoriser certaines transactions.

Une étude intéressante reste toutefois à développer en vue d'élargir ce travail, les axes d'investigation suivants restent à développer :

A court terme :

Coordonner les différentes fonctions du système de gestion de bases de données EASY, à savoir :

- Développement d'un contrôleur de terminaux qui permettra d'établir un ordre de lancement des transactions utilisateurs et d'adresser les résultats d'exécution obtenus par le système vers l'utilisateur approprié.
- Décomposition des transactions en actions élémentaires en tenant compte de certaines contraintes liées au langage d'interrogation disponible sur la base de données.
- Recherche d'une granularité optimale permettant un niveau de parallélisme élevé en n'exigeant pas une gestion de verrous coûteuse.
- Etude du domaine d'application nécessitant un examen approfondi des types de bases de données, en précisant la nature de l'application, les objectifs du concepteur de l'application et les contraintes liées à l'intégrité des données.

L'ensemble de ces axes de travail permettrait d'affiner l'implémentation effective du contrôleur.

A long terme :

- Recherche d'outils destinés d'une part à valider les différents types de contrôleurs à l'aide des types abstraits et d'autre part, à choisir automatiquement uncontrôleur suivant l'application traitée.
- Extension du travail aux bases de données réparties avec l'apport de quelques micro-ordinateurs interconnectés à l'aide d'un réseau local.
- Répartition du contrôleur des terminaux sur les différents sites à l'aide du séquenceur circulant (anneau virtuel).

PARTIE I - *ETUDE ET EVALUATION DES TECHNIQUES DE CONTROLE D'ACCES CONCURRENTS DANS LES SYSTEMES DE GESTION DE BASES DE DONNEES REPARTIES*

/ATRO 84/ : M.A. ATROUN, F. BERGHEUL, S.A. LARIBI
Concurrence d'accès dans les systèmes de gestion de bases de données.
5ème JTEA, ENSET, mai 1984

/BAGH 84/ : L. BAGHDALI, N. BENSAOU, S.A. LARIBI, N. MADANI
EASY - Système de gestion de bases de données construit à partir du modèle Entités/Associations.
3ème CAPM, Tunis, avril 1984.

/BENM 82/ : F. BENMAKROUHA
Evaluation de performances de systèmes transactionnels répartis.
Thèse de Docteur Ingénieur, Rennes, Mars 1982.

/BERG 84/ : F. BERGHEUL
Etude et évaluation des techniques de contrôle d'accès concurrents dans les SGBD répartis.
Développement de contrôleurs détectant les interblocages.
Thèse de Magister, USTHB, septembre 1984.

/BERN 78/ : P.A. BERNSTEIN, J.B. ROTHNIE, N. GOODMAN, C.A. PAPADIMITRIOU
The concurrency control mechanism of SDD-1. A system for distributed databases. (the fully redundant case).
IEEE Trans. on software Engineering, vol.4, n° 3, mai 1978.

/BERN 79/ : P.A. BERNSTEIN, D. SHIPMAN, W.WONG
Formal aspects of serializability in database concurrency control.
IEEE Trans. on software engineering, vol. 5, n° 3, mai 1979

/BERN 80/ : P.A. BERNSTEIN, W. DAVID, J.B. ROTHNIE
Concurrency control in a system for distributed databases (SDD-1).
ACM Trans on database systems, Vol.5, n° 1, mars 1980

/BERN 81/ : P.A. BERNSTEIN, N. GOODMAN
Concurrency control in distributed database systems.
ACM Computing surveys, Vol. 13, n° 2, juin 1981, pp. 185-222

/ELLI 77/ : C.A. ELLIS
Consistency and correct ness of duplicate databases.
System proceeding of 6 th ACM. Symposium on OSP, novembre 1977.

/ESWA 76/ : K. ESWARAN, J. GRAY, R. LORIE, I. TRAIGER
The notion of consistency and predicate locks in a database system.
CACM, vol.19, nº 11, novembre 1976.

/GARC 79/ : H. GARCIA-MOLINA
Performance of update algorithms for replicated data in a distributed database.
Report nº STAN-CS-79-744, Department of Computer Science, Stanford University.

/GARD 81/ : G. GARDARIN, M. MELKANOFF
Concurrency control principles in distributed and centralized databases.
Rapport de Recherche INRIA, nº 113, janvier 1981.

/GARD 83/ : G. GARDARIN
Les systèmes et leurs langages.
Eyrolles, 1983, pp. 137-213.

/GRAY 76/ : J.N. GRAY, R.A. LORIE, G.R. PUTZOLU, I. TRAIGER
Granularity of locks and degrees of consistency in a shared database.
CACM, vol.19, nº 11, novembre 1976, pp. 365-394.

/GRAY 80/ : J.N. GRAY
A transactionnel Model
8 Th International conference on language and programming, 1980.

/HABE 69/ : A.N. HABERMANN
Prévention of system deadlocks
ACM Communications, Vol. 12, nº 7, juillet 1969.

[HAVE 68] : J.W. HAVENDER
Avoiding deadlock in multitasking systems.
IBM, Syst. J. n° 2, 1968.

[HERM 79] : D. HERMAN, J.P. VERJUS
An algorithm for maintening the consistency of multiples copies.
1ère Conférence internationale sur DCS.
Huntsville, Alabama, octobre 1979.

[LAMP 76] : L. LAMPORT
Time, clocks and the ordering of events in a distributed system.
Massachussetts Computer Associates Report CA-7603-2911,
mars 1976.

[LEBI 80] : LE BIHAN et Coll.
Sirius Delta, a distributed database management system prototype.
International Symposium on distributed databases, IRIA, march 80.

[LELA 77] : G. LE LANN
Distributed systems, towards a formal approach.
IFIP, Congress Proceedings, 1977.

[LELA 78a] : G. LE LANN
Coherent data-sharing in failure-tolerant distributed processing systems.
IRIA, Sirius Project, janvier 1978.

[LELA 78b] : G. LE LANN
Algorithms for distributed data-sharing systems which use tickets.
Proceeding of the third Berkeley Workshop on distributed data.
Management and computer Networks. San Francisco, août 1978.

[LELA 79] : G. LE LANN
Le contrôle dans les systèmes informatiques répartis : nature du problème et quelques solutions.
Journées Bigre, Nancy, France, janvier 1979.

CHAPITRE 1 - LE CONTROLE DES ACCES CONCURRENTS

1.1 - CONCEPTS DE BASE

1.1.1 - Multi-utilisation des données

1.1.1.1 - Bases de données centralisées (BD)

a - Définition

Une BD est un ensemble structuré de données enregistrées sur des supports accessibles par l'ordinateur de façon sélective et en un temps opportun.

La notion de BD a introduit des structures de données facilitant les accès partagés entre plusieurs usagers et rend possible la centralisation, la coordination, l'intégration et la diffusion de l'information archivée.

b - Systèmes de gestion de BD (SGBD)

Le SGBD logiciel qui permet d'inter-agir avec la BD, doit assurer les principaux objectifs suivants : l'indépendance physique, l'indépendance logique, la manipulation des données, l'efficacité des accès aux données... Ce sont là des fonctions premières complétées par des fonctions plus complexes afin de maintenir la cohérence des données, de les protéger contre tout incident ou accident et d'obtenir des performances acceptables de la part du système /GARD 83/.

Pour atteindre ces objectifs, les concepteurs de SGBD ont distingué trois niveaux de représentation d'une BD :

1 - Le niveau interne

Le schéma physique va spécifier la façon de stocker les données sur les organes périphériques.

2 - Le niveau conceptuel

Le schéma conceptuel est une partie fondamentale dans l'architecture d'un SGBD; il décrit en termes abstraits une certaine réalité pour la mise en oeuvre d'une BD.

Plusieurs classes de modèles sont apparues : nous distinguons les modèles hiérarchique, réseau, relationnel et entités/associations.

Le modèle hiérarchique est basé sur la notion de graphe dont les noeuds représentent les classes d'objets décrits et les arcs sont les associations relatives.

Le modèle réseau élargit le modèle hiérarchique en incluant des relations dans le graphe entre père-fils et inversement.

Le modèle relationnel est fondé sur la théorie mathématique des relations; les relations sont représentées sous forme de tables. Ce modèle offre une grande simplicité de description et de manipulation d'une BD. L'utilisateur manipule un ensemble de tableaux de valeurs sans se soucier de la manière dont ces données ont été stockées.

Le modèle entités/associations, adopté par la SGBD mono-utilisateur EASY et développé au Département d'Informatique et Logique Mathématique (DILM) [BAGH 84_7], est basé sur la théorie des ensembles et la théorie des relations. Il utilise comme éléments de base deux catégories d'objets :

. l'ensemble d'entités
. l'ensemble d'associations

L'ensemble d'entités regroupe toutes les entités ayant un certain nombre de propriétés communes. Une entité est définie comme un objet distinctement identifié.

L'ensemble d'associations correspond aux liens sémantiques entre les entités appartenant aux ensembles correspondants.

L'information concernant une entité ou une association est exprimée par un ensemble de paires attributs-valeurs. Les valeurs sont classées dans différents ensembles de valeurs. Un prédicat est associé à chaque ensemble pour tester l'appartenance de la valeur à l'ensemble de valeurs.

Un attribut est une fonction qui part de l'ensemble d'entités ou d'associations vers l'ensemble de valeurs ou vers le produit cartésien d'ensembles de valeurs. La notion d'attribut est l'une des principales caractéristiques du modèle entités/associations.

Plusieurs classes de modèles sont apparues : nous distinguons les modèles hiérarchique, réseau, relationnel et entités/associations.

Le modèle hiérarchique est basé sur la notion de graphe dont les noeuds représentent les classes d'objets décrits et les arcs sont les associations relatives.

Le modèle réseau élargit le modèle hiérarchique en incluant des relations dans le graphe entre père-fils et inversement.

Le modèle relationnel est fondé sur la théorie mathématique des relations; les relations sont représentées sous forme de tables. Ce modèle offre une grande simplicité de description et de manipulation d'une BD. L'utilisateur manipule un ensemble de tableaux de valeurs sans se soucier de la manière dont ces données ont été stockées.

Le modèle entités/associations, adopté par la SGBD mono-utilisateur EASY et développé au Département d'Informatique et Logique Mathématique (DILM) [BAGH 84_7], est basé sur la théorie des ensembles et la théorie des relations. Il utilise comme éléments de base deux catégories d'objets :

- l'ensemble d'entités
- l'ensemble d'associations

L'ensemble d'entités regroupe toutes les entités ayant un certain nombre de propriétés communes. Une entité est définie comme un objet distinctement identifié.

L'ensemble d'associations correspond aux liens sémantiques entre les entités appartenant aux ensembles correspondants.

L'information concernant une entité ou une association est exprimée par un ensemble de paires attributs-valeurs. Les valeurs sont classées dans différents ensembles de valeurs. Un prédicat est associé à chaque ensemble pour tester l'appartenance de la valeur à l'ensemble de valeurs.

Un attribut est une fonction qui part de l'ensemble d'entités ou d'associations vers l'ensemble de valeurs ou vers le produit cartésien d'ensembles de valeurs. La notion d'attribut est l'une des principales caractéristiques du modèle entités/associations.

3 - Le niveau externe

Le schéma externe ou "vue" décrit la façon dont seront perçues les données par un programme d'application. Ce niveau correspond à la vision de tout ou partie du schéma conceptuel par un groupe d'utilisateurs concerné par une application.

c - *Mise en oeuvre du SGBD*

Généralement, la réalisation et l'utilisation d'un SGBD nécessite :

- la construction du schéma conceptuel,
- la construction du ou des schémas externes,
- la définition des droits d'accès,
- la spécification des organisations physiques des données, ainsi que les méthodes d'accès qui seront utilisées,
- la construction d'un mécanisme de contrôle de concurrence d'accès aux données lorsque plusieurs utilisateurs demandent l'accès simultané à une même donnée,
- la mise en oeuvre de procédure assurant un certain niveau de sécurité,
- la construction d'un langage afin de créer, modifier, interroger et effectuer toute manipulation sur la BD.

L'architecture fonctionnelle du SGBD EASY peut être décrite au moyen de schémas modulaires suivants :

. le traducteur transforme les requêtes écrites dans un langage de haut niveau, non procédural, en arborescences algébriques,
. l'optimiseur de ces requêtes traduites, mettant en oeuvre des méthodes d'optimisation des chemins d'accès,
. le système de stockage permet une organisation efficace des données,
. l'interpréteur de requêtes exécute l'arborescence optimisée.

La première étape du projet EASY restreint l'utilisation du SGBD à un seul initiateur.

Le contrôleur de concurrence /ATRO 84_7, mécanisme offrant la multi-utilisation de EASY, permet à plusieurs usagers le lancement de requêtes, indépendantes les unes des autres, et le maintien de la cohérence de la BD (Fig.1).

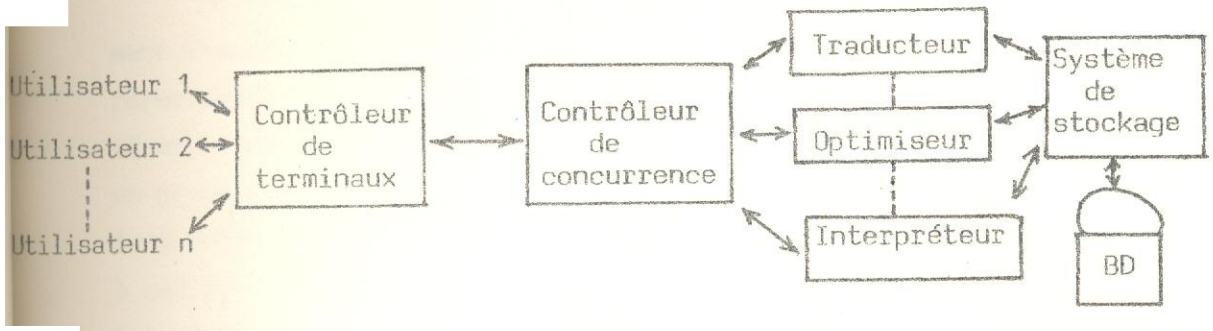

Figure 1 : Architecture fonctionnelle

Le contrôleur de terminaux gèrera les entrées de requêtes ainsi que la sortie des résultats.

1.1.1.2 - Bases de données réparties (BDR)

a - Définition

Une BDR est une partition physique d'une BD sur des installations informatiques (sites) éventuellement différentes, connectées par des lignes de communication (réseau) et permettant aux utilisateurs d'accèder aux données comme si la BDR était une base centralisée.

Une BDR est manipulée par une multiplicité d'usagers distribués sur le réseau sans qu'il y ait de relation entre la répartition des utilisateurs et des données.

Une BD peut être distribuée de trois manières :

- soit la BD est décomposée en plusieurs partitions disjointes sur des sites distincts : BD partitionnées
- soit la BD est dupliquée totalement, chaque copie se trouve sur un site : BD dupliquées ou copies multiples
- soit la duplication de la BD est partielle : BD partiellement dupliquée.

b - Apport des BDR par rapport aux BD

L'arrivée des réseaux a permis de décentraliser l'information afin d'assurer une gestion plus souple et plus efficace.

Un réseau permet à un certain nombre d'ordinateurs géographiquement dispersés (sites), de communiquer entre eux pour échanger des informations. Dès que ces échanges se font avec une fiabilité suffisante, il est immédiatement tentant d'oublier l'existence d'un réseau et par un processus d'abstraction de considérer l'ensemble des ordinateurs reliés entre eux comme une seule installation informatique. En appliquant à celle-ci le concept de BD et de SGBD, nous voilà aux portes des BDR. Les réseaux ont donné l'idée aux concepteurs de SGBD :

- d'une part, de relier les BD individuelles : c'est l'approche ascendante,
- d'autre part, d'éclater la BD centralisée en la spécialisant : c'est l'approche descendante.

Les BDR présentent un autre avantage : celui de la réduction des pannes au niveau global; en effet, si un SGBD situé sur un site donné tombe en panne, l'un des autres SGBD prend la relève; la BD du site défaillant est sauvegardée.

c - *Système de gestion de bases de données réparties (SGBDR)*

Un SGBDR est constitué par l'ensemble des logiciels existants sur les sites du réseau et qui concourent à la gestion de la BDR.

d - *Mise en oeuvre d'un SGBDR*

Dans un SGBDR, le contrôleur de concurrence d'accès a une tâche plus complexe qu'en centralisé, puisque la synchronisation n'est pas instantanée pour l'ensemble des sites.

Dans la hiérarchie des niveaux de protocoles-réseau, le niveau que nous étudions, constitue une interface entre le niveau des protocoles d'application comportant notamment les SGBD et le niveau qui permet une exécution. Il fournit un ensemble de techniques [Chapitre 2] destiné à :

. assurer une exécution en environnement réparti,
. contrôler l'exécution répartie,
. résoudre les problèmes causés par des accès concurrents conflictuels,
. permettre un fonctionnement en dépit de certaines pannes.

Ce niveau est destiné à rendre transparent à l'utilisateur l'existence de la répartition.

e - *Les différents types de SGBDR*

Avant de définit les types de SGBDR existants, nous allons donner certaines notions nécessaires à leur établissemant.

La présence d'utilisateurs sur un site correspond à ce que nous appelons un "point d'accès", celle de données à un "point de stockage"; nous appellerons "point d'échange" tout site offrant les deux fonctions précédentes (accès et stockage).

Nous considérons trois éléments dans la composition d'un système réparti :

- les utilisateurs : u_i
- les données : D
- les systèmes de gestion de données : SGD

Les différents types de SGBDR sont caractérisés par deux catégories : les SGBDR assurant un contrôle de concurrence d'accès centralisé et les SGBDR assurant un contrôle de concurrence réparti.

1 - SGBDR à contrôle centralisé

Dans la classe des systèmes centralisés dans un contexte réparti, la gestion de données est centralisée sur un site assurant le service réparti.

La SGD peut être aussi bien un SGBD classique ou un système de gestion de fichiers (SGF).

Dans ce type (Fig.2), les données sont sur un site; le SGD gère toutes les manipulations sur les données et, en particulier, il assure le contrôle d'accès aux informations communes aux différents utilisateurs.

Dans ce cas, le mécanisme de contrôle de concurrence est basé sur les techniques à aspect centralisé.

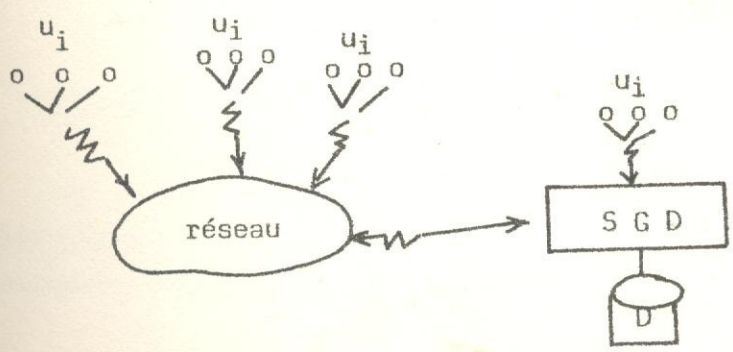

Figure 2 : Système centralisé à accès réparti

La répartition ne rentre en ligne de compte qu'au niveau des points d'accès (Fig.2), ou au niveau des points de stockage (Fig.3). Dans ce dernier cas, les données sont réparties, mais leur gestion se fait comme dans le cas précédent sur un site central (Fig.3).

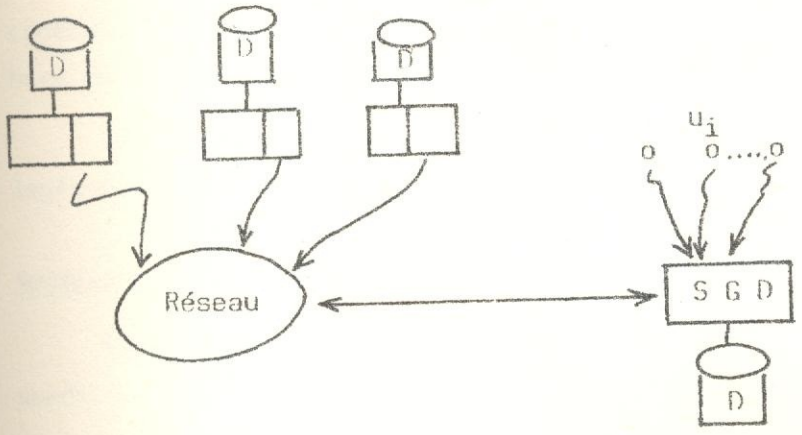

Figure 3 : Système centralisé à stockage réparti

2 - SGBDR à contrôle réparti

Réseau de systèmes centralisés indépendants

Dans cette deuxième classe de systèmes répartis, nous allons définir la décentralisation du SGD (Fig.4).

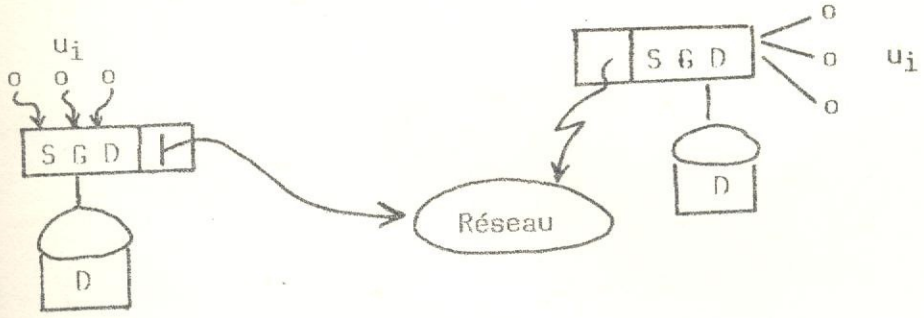

Figure 4 : Réseau multi-SGD

Sur chaque site du réseau, il existe un SGD centralisé qui gère de façon autonome ses propres données. L'utilisateur s'adresse à l'un ou l'autre des SGD en spécifiant la localisation de ses données.

Le mécanisme de contrôle de concurrence peut être implanté sur un site particulier, contrôle centralisé, ou réparti sur tous les sites, contrôle réparti.

Réseau de systèmes centralisé correlés

Dans cette classe, la seule différence par rapport aux systèmes centralisés indépendants, réside dans le fait que l'utilisateur est libéré de la connaissance de la localisation de ses données.

Ce résultat est obtenu en confiant à un logiciel système réparti, implanté sur chaque site, la recherche des sites à accéder.

Système de gestion de fichiers répartis

Par contre, dans cette classe, ce que nous appelons SGF réparti, va au-delà de cette simple extension du pouvoir d'accès; il réalise une intégration de l'ensemble des fichiers du réseau.

Nous illustrons cette classe par un exemple :

Soit une requête d'un utilisateur portant sur les données d_1, d_2, ...,d_n; le SGF réparti pourra y répondre en accèdant par exemple aux données d_1, d_2,... , d_i situées sur un site et aux autres d_1, d_{i+1}, ..., d_n, situées sur un autre site; d_1 étant la donnée dupliquée permettant d'identifier la réquête relative à un utilisateur sur les deux sites.

L'analyse et la conception de ces types de systèmes ne peuvent être conduites en faisant abstraction du contexte dans lequel ils vont évoluer.

Dans un SGBD, les moyens de traitement et de stockage des informations sont concentrés en un point d'accumulation. Les limitations aux BD centralisées entraînent une croissance importante des responsabilités, privant les usagers de l'accès aux données en cas de panne. En conséquence, il est nécessaire de concevoir des outils permettant d'administrer et de manipuler des ensembles de données hétérogènes répartis sur des sites hôtes d'un réseau et gérés localement par des SGBD classiques.

Au cours de la définition des notions de BD, BDR, SGBD et SGBDR, nous avons introduit les principales fonctions qu'ils assurent; nous allons nous occuper plus particulièrement de la fonction de contrôle des accès simultanés à une même information.

Les accès concurrents conflictuels, des usagers aux mêmes informations, ne doivent en aucun cas nuire aux contraintes d'intégrité de la BD; ces contraites définissent les relations de dépendance liant sémantiquement les informations contenues dans la base : elles doivent être vérifiées, que la BD soit centralisée, partitionnée ou dupliquée.

1.1.2 - Contraintes d'intégrité (CI) et Cohérence

Un ensemble d'informations accédé par des requêtes est supposé satisfaire certaines règles de cohérence appelées Contraintes d'Intégrité (CI).

Une contrainte d'intégrité est une assertion que doivent vérifier, à certaines instants privilégiés, les données d'une BD.

Les instants choisis sont généralement les états stables de la BD. La transformation d'un état e_i de la BD sous l'influence de l'information contenue dans une mise à jour n'a pas lieu instantanément : la période d'exécution de toute mise à jour est délimitée par les deux bornes d_m^i et f_m^i qui représentent respectivement l'instant d'initialisation et de terminaison de l'exécution de la mise à jour "m" sur la copie c_i. Dans l'intervalle (d_m^i, f_m^i), c_i se trouve dans un état stable.

Une BD dont l'ensemble des CI est respecté est cohérente.

Deux types de cohérence sont à distinguer dans les SGBDR :

- La cohérence faible : un système est faiblement cohérent si en l'absence d'initialisation de nouvelles mises à jour après un instant t, il existe un instant t' fini tel que la distance entre les copies est nulle par une remise à niveau des copies.

- La cohérence forte : un système est fortement cohérent si l'exécution d'une mise à jour sur un site entraîne la mise à jour simultanée sur l'ensemble des copies situées sur les différents sites.

1.1.3 - Transaction

Une transaction, unité de traitement séquentiel exécutée pour le compte d'un usager, appliquée à une BD cohérente, restitue une BD cohérente. La transaction est une fonction faisant passer la BD d'un état à un autre; pour cela, les données permettant d'effectuer ce changement d'états sont mémorisées sur un support temporaire de travail et à l'aide de la terminaison de la transaction, le résultat est enregistré par une ou plusieurs opérations de recopie sur la BD : c'est la phase de validation de la BD.

Un système transactionnel contrôle l'exécution de transactions en garantissant la propriété d'atomicité [GRAY 80] : la séquence d'actions élémentaires formant la transaction s'exécute de manière indivisible, préservant la cohérence des données. Ainsi, l'utilisation du concept de transaction permet de définir un premier niveau de cohérence.

En environnement réparti, la liaison de plusieurs systèmes transactionnels étend le concept de transaction en transaction globale : une transaction globale est composée de plusieurs agents coopérants, situés sur des sites distincts. L'atomicité s'applique à la transaction globale, c'est-à-dire pour l'ensemble des actions exécutées par chaque agent sous le contrôle du système transactionnel correspondant.

Une transaction est modélisée comme une suite d'actions portant sur les données de la BD. D'une manière générale, nous représentons la transaction comme suit :

$$T = \left\{ (t, A_i, O_i) \ / \ i = 1, \ldots, n \right\} \quad \text{où}$$

t identifie la transaction

A_i correspond à l'opération sur la BD; A_i est l'action, elle peut prendre comme valeur :

- . début-transaction : initialisation d'une nouvelle transaction
- . lire-donnée : lecture d'une donnée de la base puis stockage dans un espace de travail.

. écrire-donnée : écriture dans la BD d'une donnée à partir d'une valeur stockée dans un espace de travail

. fin-transaction : terminaison de la transaction.

O_i est l'objet de la BD ou donnée accédée par l'action A_i.

Une transaction sera dite bien formée ou bien spécifiée si elle transforme tout état cohérent de la BD en un état cohérent.

1.1.4 - Verrouillage et granularité des verrous

D'une manière générale, l'exécution d'une transaction requiert un ensemble de ressources (ou données). Ces ressources peuvent être demandées de manière exclusive ou partagée par les transactions concurrentes. Le contrôle des accès concurrents est le contrôle d'allocation/désallocation de l'ensemble des ressources. L'acquisition des ressources par une transaction nécessite le blocage de ces ressources par dépôt d'un verrou, qui peut être une valeur entière, sur chacune d'elles.

La "granularité" est la taille minimale de l'ensemble de données (relation, tuple, ...) que l'on alloue.

Il existe plusieurs études sur la granularité optimale [GRAY 76, RIES 77, RIES 79]; le choix de la taille du granule à verrouiller est un problème fondamental dans le développement d'un SGBD multi-usagers.

Plus le granule est 'petit' et plus la concurrence pourra être élevée; la notion de concurrence élevée se traduit par un grand nombre de transactions prises en compte au même instant par un contrôleur de concurrence.

Mais si le granule est 'petit', la tâche du contrôleur sera complexe, puisqu'il aura un plus grand nombre de verrous à gérer. A l'inverse, si les verrous peuvent être placés sur de 'gros' granules, cela diminuera le nombre de transactions en accèdant à des composantes plus élémentaires. Par exemple, la granularité est 'grosse' si l'élément d'allocation et de verrouillage est le fichier (relation, base de données); elle sera 'fine' si la demande d'allocation est l'enregistrement logique (tuple d'une relation) ou la rubrique (attribut).

En organisant hiérarchiquement les objets et en fournissant un protocole de verrouillage dynamique, il est possible d'offrir différentes tailles d'objets à verrouiller dans un même système.

1.2 - CONFLITS D'ACCES CONCURRENTS

Le partage d'informations par plusieurs transactions soulève de nombreux problèmes [ESWA 76]. Ainsi, si l'on autorise un grand nombre d'usagers à manipuler de manière simultanée les mêmes données, il est nécessaire de s'assurer que les accès conflictuels ne perturbent pas la cohérence de la base de données.

1.2.1 - Perte d'opération

Un des problèmes de la simultanéité d'exécution de transactions concerne la perte d'opération, qui survient dès que le résultat d'une action est détérioré par une action d'une autre transaction simultanée. La BD est vue, dans le cas de la perte d'opération, comme un ensemble d'objets indépendants sans contraintes d'intégrité entre eux. L'accès concurrent à un tel ensemble d'objets ne va pas sans problème. Le cas le plus typique est la perte de mise à jour.

La perte de mise à jour apparaît lorsqu'une transaction T1 lit puis écrit une donnée A, mais entre les actions de lecture et d'écriture de T1, une autre transaction T2 écrit (ou modifie) la donnée A; ainsi la mise à jour effectuée par T2 est perdue par celle de T1.

Exemple :

Soit T1 une transaction qui effectue une lecture de A, suivie d'une écriture

```
                début T1
                lire  A
        T1  :
                écrire A
                fin T1

  soit  T2      une transaction qui modifie la donnée A

                début T2
        T2  :   écrire A
                fin T2
```

Lorsque T1 est exécutée seule, toutes ses actions sont bien exécutées; de même pour T2. Mais les transactions T1 et T2 sont exécutées simultanément.

Considérons le recouvrement suivant de T1 et T2 :

T1 : début T1
T1 : lire A

T2 : début T2
T2 : écrire A
T2 : fin T2

T1 : écrire A
T1 : fin T1

Ce recouvrement (ou exécution simultanée) conduit à perdre la mise à jour effectuée par T2.

1.2.2 - Incohérence

Un second type de problème est soulevé par l'existence de contraintes d'intégrité.

Nous distinguons trois cas d'incohérence :

1.2.2.1 - Observation d'incohérence

Ce problème apparaît quand une transaction effectue des actions sur un état transitoire incohérent de la BD. Les contraintes d'intégrité non vérifiées par l'état intermédiaire peuvent impliquer des sorties incohérentes.

Exemple :

Soit T3
Début T3, modifie B (B = B + 100), lit B, modifie (B=B-100), Fin T3
T3 est bien spécifiée.

Si T1 et T3 s'exécutent simultanément, les données A et B doivent être égales à tout instant et B initialement positive, ne doit pas dépasser la valeur 99.

Considérons le recouvrement suivant de T1 et T3 :

T1 : début T1
T1 : lire A
T1 : écrire A

T3 : début T3
T3 : écrire (B = B + 100)
T3 : lire B
T3 : écrire (B = B - 100)

T1 : fin T1

T3 : fin T3

A la terminaison de l'exécution de T1 et T3, les contraintes A = B et B < 99 sont vérifiées, mais au cours des modifications de T3, nous observons une incohérence : B > 99.

1.2.2.2 - Perte de cohérence

Un tel conflit peut survenir quand une transaction modifie un état transitoire incohérent de la BD en cours de modification par une autre transaction. Les données modifiées peuvent alors devenir définitivement incohérentes.

Exemple :

Considérons la transaction T4 suivante :

T4 : début T4
écrire B (B = B + 100)
fin T4

T4 est bien spécifiée.

Soit le recouvrement de T1 et T4 :

T1 : début T1
T1 : lire A

T4 : début T4
T4 : écrire B (B = B + 100)

T1 : écrire A
T1 : fin T1

T4 : fin T4

Un tel recouvrement conduit à A ≠ B.

1.2.2.3 - Non-reproductibilité des lectures

Une transaction peut lire deux fois une même donnée et trouver deux valeurs différentes du fait que la donnée a été modifiée par une transaction concurrente entre les deux lectures. Les deux ensembles de valeurs obtenues sont alors différents. Ceci est sans importance si aucune contrainte d'intégrité portant sur les sorties de la transaction n'induit l'égalité des deux ensembles de valeurs, mais conduit à détruire la cohérence de la BD dans le cas contraire.

Exemple :

soit T5 :
début T5
lire A
écrire A
lire A
écrire A
fin T5

T5 est une transaction bien formée.

Considérons une transaction bien spécifiée T6, telle que :

T6 :
début T6
modifier A (A = A + 1)
fin T6

et le recouvrement suivant de T5 et T6 :

T5 : début T5
T5 : lire A
T5 : écrire A

T6 : début T6
T6 : écrire (A = A + 1)
T6 : fin T6

T5 : lire A
T5 : écrire A
T5 : fin T5

Du fait que T6 modifie A, T5 lira deux valeurs différentes pour A.

1.2.3 - Difficultés propres aux SGBDR

Le contrôle d'accès concurrent an environnement réparti a le même rôle qu'en centralisé. Cependant, la répartition d'un système de gestion de ressources sur plusieurs sites indépendants pose les problèmes spécifiques suivants :

1.2.3.1 - Communication par message

Le contrôleur d'accès concurrent n'est plus réalisé par un seul programme s'exécutant sur un processeur, mais par plusieurs programmes coopérants par échange de messages. Il n'y a pas de relation d'ordre évidente entre les événements traités par chacun des contrôleurs. De plus, chacun des contrôleurs ne voit qu'une partie des données et des transactions qui y ont accès.

1.2.3.2 - Indépendance des contrôleurs

Il est souhaitable que le SGBDR dans son ensemble survive à la panne d'un des systèmes coopérants le constituant. Ce fonctionnement dégradé impose que chaque contrôleur soit capable de prendre la décision d'allouer ou de libérer des ressources, indépendamment des autres contrôleurs s'ils sont devenus inaccessibles. Cette indépendance n'est généralement jamais totale. Certaines pannes obligent un contrôleur à attendre le redémarrage d'un autre contrôleur pour pouvoir décider de l'action à entreprendre sur une donnée. La difficulté principale vient de ce que des pannes du réseau peuvent isoler des contrôleurs tout en les laissant en "vie". Les contrôleurs isolés doivent autoriser l'exécution de transactions qui ne font pas appel aux sites inaccessibles.

1.3 - RESOLUTION DES CONFLITS D'ACCES CONCURRENT

1.3.1 - Sérialisation

L'objectif du contrôle des accès concurrents est de maintenir la cohérence du SGBD ou du SGBDR. BERNSTEIN et GOODMAN [BERN 81] ont montré que le contrôle d'accès concurrent est fondé sur la SERIALISATION :

elle consiste à ne laisser s'exécuter que les transactions provoquant les mêmes effets sur les données qu'une exécution en séquence des mêmes transactions.

De nombreuses techniques de contrôle d'accès concurrent ont été proposées. Dans le chapitre suivant, nous analysons les principales techniques d'accès simultanés. Une technique d'accès concurrent est correcte si elle n'autorise que des exécutions sérialisables : la technique garantit ainsi la sérialisation des accès simultanés.

1.3.2 - Résolution des pertes d'opération

- Un recouvrement (ou simultanéité) de transactions dont l'exécution ne provoque pas de pertes d'opération est dit légal.
- Des opérations sont compatibles si tout recouvrement donne le même résultat qu'une exécution séquentielle dans un ordre quelconque de ces opérations.

Théorème 1 : SERIALISABILITE

Une condition suffisante pour qu'un recouvrement soit légal est qu'il ne contienne pas d'opérations simultanées incompatibles effectuées par des transactions différentes.

1.3.3 - Résolution des incohérences

- Un recouvrement dont l'exécution ne provoque pas d'incohérence est dit consistant.
- Nous dirons qu'un recouvrement à la fois légal et consistant est correct.
- Une exécution E de (T1, T2, ..., TN) est une succession s'il existe une permutation Π de (1, 2, ..., N) telle que :

$$E = < T_{\Pi(1)}, T_{\Pi(2)}, \ldots, T_{\Pi(N)} >$$

Théorème 2 :

Toute recouvrement R équivalent à une succession est correct

1.3.4 - Détection du conflit dû à la répartition des données : le graphe de précédence

Nous pouvons formaliser les interférences entre accès aux données en construisant un "graphe de précédence". Les noeuds de ce graphe représentent les transactions. Les arêtes représentent les dépendances entre transactions. Nous traçons une arête entre deux transactions si des données lues par une transaction sont aussi mises à jour par l'autre ou si des données modifiées par une transaction sont aussi mises à jour par l'autre. La première arête représente une dépendance "lecture-écriture", la deuxième une dépendance "écriture-écriture". Les arêtes sont orientées par la précédence dans le temps, d'où le nom du graphe. Une arête est orientée de T vers U si la transaction U a manipulé postérieurement la même donnée que T.

BERNSTEIN et Coll. [BERN 79, BERN 81] démontrent qu'un ordonnancement conduit à une exécution sérialisable des transactions s'il n'y a pas de cycles dans le graphe de dépendance.

1.3.5 - Classification des méthodes de sérialisation

Toutes les techniques de contrôle d'accès concurrents [Chapitre 2] peuvent être obtenues par combinaison de deux méthodes de sérialisation [BERN 81] :

- ordonnancement initial par estampillage
- et le verrouillage à deux phases.

1.3.5.1 - Ordonnancement initial par estampillage

L'ordonnancement initial des transactions est une méthode de base fondée sur la génération d'estampilles associées aux transactions dès leur lancement. Les estampilles sont des nombres générés, de telle sorte que chaque nombre soit unique dans le système de contrôle.

Les techniques d'ordonnancement par estampillage ordonnent les transactions en conflit en les forçant à s'exécuter dans l'ordre de leur estampille : cette méthode garantit la sérialisation. En effet, il ne

peut y avoir de cycle dans le graphe de précédence puisque chaque arête part d'un noeud ayant une estampille strictement inférieure à l'estampille du noeud d'arrivée.

Selon BERNSTEIN et Coll. [BERN 80], les règles de contrôle de la méthode de base sont les suivantes :

a - Synchronisation Lecture-Ecriture

Une lecture est acceptée si l'estampille de la transaction est supérieure à l'estampille en écriture de la donnée. L'estampille en lecture est alors mise à jour (maximum de l'ancienne estampille en lecture et de l'estampille de la transaction). Une écriture est acceptée si l'estampille de la transaction est supérieure à l'estampille en lecture de la donnée. L'estampille en écriture de la donnée est alors mise à jour (maximum de l'estampille de la transaction et de l'ancienne estampille en écriture de la donnée).

b - Synchronisation Ecriture-Ecriture

Une écriture est acceptée si l'estampille de la transaction est supérieure à l'estampille en écriture de la donnée. L'estampille en écriture est alors mise à jour avec la valeur de l'estampille de la transaction.

1.3.5.2 - Verrouillage à deux phases (2PL)

Dans cette seconde méthode, à chaque ressource est associée :

- une variable indiquant son état : libre, réservée en accès partagé ou exclusif
- une file d'attente des transactions ayant demandé cette ressource sans pouvoir l'obtenir
- une liste des transactions qui possèdent la ressource en mode partagé.

Quand une transaction requiert une ressource, le contrôleur vérifie l'état du verrou et éventuellement alloue la ressource à la transaction. Si nécessaire, il change l'état du verrou associé. Si le verrou est dans un état incompatible avec l'action, la transaction est mise en attente et prend place dans la file d'attente de la ressource. Ces attentes introduisent des possibilités d'interblocage.

Les techniques basées sur le verrouillage garantissent la sérialisation des exécutions en imposant des contraintes sur les séquences de verrouillage/déverrouillage effectuées par les transactions.

Nous trouvons généralement les deux contraintes suivantes :

a - Transaction bien formée

- aucun accès à une donnée n'est autorisé sans verrouillage préalable compatible avec l'accès envisagé,
- aucune donnée ne reste allouée en fin de transaction.

b - Transaction à deux phases (acquisition-libération)

- aucune nouvelle demande de ressource n'est autorisée après la première libération de ressource.

Les techniques qui satisfont ces deux contraintes sont appelées techniques "2 PL" (two-phases Locking), et introduisent le phénomène d'interblocage.

L'interblocage ou verrou mortel est la situation à laquelle on aboutit lorsque des granules ont été verrouillés dans un ordre tel qu'un groupe de transactions, demandant toutes les mêmes granules, vérifie les propositions suivantes :

- chaque transaction est bloquée en attente d'un granule verrouillé par l'une des transactions du groupe
- l'exécution de toutes les transactions n'appartenant pas au groupe ne permet pas de débloquer une des transactions du groupe.

Le verrou mortel est un problème primordial à résoudre par le contrôle d'accès concurrent.

1.3.5.3 - Résolution des interblocages

Deux méthodes de résolution de l'interblocage sont proposées par différents auteurs :

- la prévention des interblocages
- la détection des interblocages.

a - La prévention [HABE 69, ROSE 78]

La solution préventive garantit qu'aucun interblocage ne se produit; elle consiste à éviter l'apparition de conflits en différant les transactions. La prévention conduit au problème suivant : il arrive qu'une transaction soit reprise plusieurs fois de suite sans finalement avoir la possibilité de s'exécuter. Pour résoudre ce problème de "famine", des techniques d'estampillage sont utilisées. L'estampille augmente la priorité d'exécution de la transaction [ROSE 78].

b - La détection [HAVE 68, OBER 82, MENA 79]

La détection laisse se créer des interblocages et les résoud après qu'ils aient été détectés. Nous construisons à cette fin le graphe des attentes représentant les dépendances dans le temps entre les transactions. Chaque noeud représente une transaction. Une arête est tracée d'un noeud vers un autre si la transaction représentée par le premier noeud est en attente d'une ressource détenue par la transaction représentée par le deuxième noeud. Ce graphe est un sous-graphe de précédence introduit précédemment (sous réserve que le graphe de précédence ne contienne pas de cycle).

Il y a interblocage dans le système transactionnel s'il existe un cycle dans le graphe des attentes.

La construction du graphe des attentes est aisée en environnement centralisé. Elle est plus délicate en environnement distribué car les informations nécessaires à sa construction sont réparties, chaque site n'ayant la visibilité que d'une partie des transactions.

1.3.6 - Conclusion

BERNSTEIN et GOODMAN ont montré que toutes les techniques pouvaient être obtenues par des combinaisons de ces deux mécanismes de bases [BERN 80].

L'ensemble de contrôle des accès concurrents (CAC), développé au DILM, nous a permis de prouver que l'ordonnancement initial des transactions nécessitait une seule mise en oeuvre [Partie II], par contre les techniques à deux phases de verrouillage sont nombreuses [Partie II, BERG 84].

CHAPITRE 2 - ETUDE DES TECHNIQUES DE CONTROLE DES ACCES CONCURRENTS

2.1 - INTRODUCTION

De nombreux travaux portant sur le contrôle des accès concurrents ont motivé l'existence de plusieurs techniques.

Bien que le but recherché par les diverses techniques ne diffère guère d'une approche à une autre, il apparaît cependant que l'étude de ces différentes techniques reste complexe pour les raisons suivantes :

- la description imprécise de certaines techniques,
- les modes de description différents rendant toute comparaison ardue,
- le manque de précision quant au domaine d'application,
- l'abondance de la littérature.

Ces constatations nous ont amené à faire une étude systématique des différentes techniques afin de sélectionner les approches les plus adéquates à la mise en oeuvre et l'évaluation de contrôleurs d'accès simultanés dans les sytèèmes de gestion de BD centralisés et répartis [PARTIE II, BERG 84].

Dans ce chapitre, nous décrivons de façon homogène, les principales approches d'accès concurrents. Pour chaque approche, nous donnons la description de son fonctionnement et ses domaines d'application. Nous proposons ainsi un modèle de description ouvert pour les techniques récentes que nous n'avons pas étudiées.

2.2 - APPROCHE DE BERNSTEIN

2.2.1 - Description du fonctionnement

SDD-1 [BERN 80, ROTH 80], maquette de base de données réparties, forme un vaste projet de recherche ayant pour but la gestion des informations dupliquées et partitionnées.

Le principe de la technique de contrôle utilisée dans SDD-1 est l'ordonnancement sans reprise, consistant à refuser une opération sur une donnée si elle est susceptible d'empêcher une action ultérieure sur la même donnée.

Pour s'assurer qu'aucune opération "en retard" ne viendra perturber les opérations déjà exécutées, chaque contrôleur attend d'avoir reçu de tous les contrôleurs un message sur leur état d'avancement. Chaque site possède une file d'actions reçues des sites distants. Les actions sont rangées par ordre d'arrivée. Les estampilles de ces actions sont croissantes car chaque contrôleur attend la fin de l'émission des actions d'une transaction pour émettre les premières actions de la suivante.

SDD-1 améliore le degré de parallélisme en introduisant des "classes de transactions". Les transactions sont préanalysées de manière à détecter les données qui seront accédées au cours de leur exécution. Elles sont ensuite réparties en plusieurs classes; ces classes décèlent le type de protocole de synchronisation à utiliser en vue de maintenir une cohérence faible de l'ensemble des copies tout en assurant un niveau de parallèlisme élevé; quatre protocoles de synchronisation, notés de p1 à p4, sont définis :

- Le protocole p1 : empêche que les messages de lecture d'une transaction, entrant en conflit avec des messages d'écriture émanant d'une autre transaction, ne soient pas pris en compte dans un ordre différent par certains sites.

  Ce protocole garantit la possibilité de sérialisation des transactions si le graphe des conflits ne comporte pas de cycle.

- Le protocole p2 : ce protocole, utilisé en cas d'existence d'un cycle dans le graphe des conflits, permet de résoudre les interférences du type observation d'incohérence.

- Le protocole p3 : assure en plus que la BD à l'instant d'observation, est à jour. Ce protocole exige que les actions de lecture et d'écriture relatives aux transactions i et j, soient exécutées dans l'ordre de leurs estampilles respectives sur chacun des sites.

Ces trois protocoles suffisent à garantir la sérialisation si toutes les transactions initialisées correspondent à l'une des classes prédéfinies.

Cependant, si une transaction ne rentre dans aucune des classes, le protocole p4 est utilisé.

- Le protocole p4 : exige que toutes les transactions en cours soient achevées avant que la transaction nécessitant la mise en oeuvre de ce protocole ne puisse être exécutée. Ce type de protocole nécessite un verrouillage très strict et engendre dès lors une dégradation dans le niveau de parallèlisme possible.

2.2.2 - Conclusion

SDD-1 utilise la technique d'ordonnancement des transactions sans reprise, fondée sur l'estampillage. Cette approche est l'une des rares à étudier les problèmes posés par les accès en écriture, mais aussi par les accès en lecture et à analyser les interférences entre ces deux types d'accès, ainsi que les interférences au niveau de la cohérence. Elle présente un mécanisme de fonctionnement en cas de pannes, en utilisant une procédure de validation des transactions. Néanmoins, l'approche de BERNSTEIN risque d'être trop générale pour des applications particulières : la quantité d'informations à gérer est importante lors de l'établissement du graphe des attentes sur l'ensemble des sites, et son examen pour détecter les interblocages.

2.3 - APPROCHE D'ELLIS

2.3.1 - Description du fonctionnement

Dans [ELLI 77] est décrit une technique aux caractéristiques essentielles suivantes :

- L'algorithme de base tire profit d'une structure en anneau virtuel du réseau : les moniteurs du système situés sur les sites différents sont reliés logiquement entre eux par des messages circulant en anneau successivement d'un moniteur au moniteur qui le suit logiquement sur cet anneau [LELA 78 a, MOSS 82].

- Toute transaction de mise à jour est exécutée en deux étapes :

  . La première étape, appelée étape de synchronisation nécessite une révolution complète du message de synchronisation sur l'anneau; cette étape est destinée à la résolution des conflits.

  . Lorsque le message de synchronisation revient au site d'origine, la transaction est automatiquement acceptée : une seconde révolution sur l'anneau est nécessaire afin que chaque moniteur effectue la mise à jour.

- En cas de conflits, il n'y a pas de rejet de l'une des transactions : une technique de mise en attente est adoptée pour la transaction la moins prioritaire.

2.3.2 - Conclusion

Nous pouvons souligner plusieurs avantages pour cette technique :

. Chaque site n'est logiquement connecté qu'à ses deux voisins.

. La connaissance des entités à modifier n'est pas nécessaire durant l'étape de synchronisation.

. Le nombre de messages circulant sur le réseau est réduit.

Par contre, cette technique comporte certains désavantages, en particulier :

- La technique n'est pas adaptée aux BD partitionnées.
- La suppression de toute possibilité de parallèlisme, vu le verrouillage global de la totalité de la base.
- Le temps de réponse assez long du fait qu'une mise à jour nécessite deux parcours sur l'anneau.
- Le fonctionnement de la technique dans un environnement sujet aux pannes n'est pas pris en compte.

2.4 - APPROCHE D'HERMAN ET VERJUS

2.4.1 - Description du fonctionnement

Dans l'approche de [HERM 79], chaque transaction est estampillée par un numéro formé à partir d'une horloge logique à chaque site et du numéro d'identification du site d'initialisation de la transaction. L'exécution des transactions suivant un ordre total sur chaque site : le site possède des files d'attente contenant chacune des messages provenant d'un site distinct.

Pour qu'une transaction puisse être traitée, deux conditions doivent être remplies :

. Aucune file ne peut être vide,

. la transaction à traiter doit être la plus ancienne.

Cette technique ne nécessite pas d'étape de synchronisation des différents sites : la synchronisation est vérifiée localement par l'examen des différentes files et des dates associées aux transactions situées dans chacune des files.

2.4.2 - Conclusion

Cette technique d'une conception très simple, paraît surtout appropriée si les BD sont entièrement dupliquées; elle maintient la cohérence faible d'un ensemble de copies tout en assurant un haut niveau de parallélisme. La quantité d'informations de contrôle nécessaire à l'exécution des transactions est réduite. Enfin, la technique est capable de fonctionner dans un environnement sujet aux pannes.

2.5 - APPROCHE DE LE LANN

2.5.1 - Description du fonctionnement

Dans cette approche, le contrôle de la concurrence dans les bases de données partitionnées ou dupliquées repose sur plusieurs notions et sur la mise en oeuvre des mécanismes suivants :

- Un anneau virtuel relie logiquement les différents contrôleurs : les messages de contrôle transmis sur le réseau circulent sur cet anneau virtuel.

Un schéma d'exclusion mutuelle [MOSS 82] adapté aux systèmes distribués dans lesquels surviennent des défaillances, est élaboré à l'aide du mécanisme de jeton [LELA 79]. La solution qui est suggérée ici, est de disposer sur l'anneau virtuel d'un message de format particulier, appelé le jeton de contrôle, il est unique et circule sur l'anneau. Le jeton matérialise la possession d'un privilège de contrôle. Dans son article [LELA 79], LE LANN démontre que le jeton n'est jamais définitivement perdu; en cas de disparition, il est regénéré et que, à tout moment, il existe un jeton et un seul sur l'anneau virtuel.

- Afin de garantir la sérialisation des transactions, un ticket [LELA 78 b] unique est attribué à chaque transaction. Les tickets sont délivrés par le jeton circulant sur le réseau. Les transactions sont traitées selon un ordonnancement séquentiel des tickets par chacun des contrôleurs.

La mise à jour d'une transaction se décompose en deux phases :

. La première étape consiste en une diffusion de la mise à jour à tous les contrôleurs gérant une copie, ces contrôleurs ne modifient pas leur copie, mais effectuent momentanément la modification sur un double de la copie.

. Lorsque le contrôleur qui a initialisé la transaction a reçu tous les acquittements, il expédie un ordre de validation pour que les contrôleurs distants mettent à jour la copie et incrémentent la valeur du ticket local qui indique la dernière transaction prise en compte.

2.5.2 - Conclusion

La technique de LE LANN paraît bien adaptée :

- pour résoudre les problèmes de maintien de la cohérence dans les BD dupliquées ou partitionnées,

- si le taux d'arrivée des transactions est important,
- si le système est sujet aux pannes,
- si le réseau permet une diffusion de messages.

Cette technique est basée sur une synchronisation à deux phases. Cependant, le nombre de messages circulant sur le réseau est important.

2.6 - APPROCHE DE ROSENKRANTZ

2.6.1 - Description du fonctionnement

ROSENKRANTZ a présenté différents modèles /ROSE 78_7 qui assurent un contrôle distribué d'accès concurrent à des bases de données partitionnées ou dupliquées.

Une cohérence faible est maintenue entre les différentes copies. L'exécution d'une transaction passe par deux phases :

- La première phase consiste à acquérir les ressources en les verrouillant, puis à résoudre les conflits en mettant en attente les transactions concurrentes.
- La deuxième phase consiste à exécuter la transaction et déverrouiller les ressources.

Dans cette approche, comme toute transaction n'implique pas le verrouillage de la totalité de la base, mais seulement les entités manipulées, le parallélisme est possible entre différentes transactions.

La gestion des accès à la BD est fondée sur le principe suivant : à toute transaction est associée un processus. Ce processus unique pour la transaction visite suivant un ordre quelconque l'ensemble des sites concernés par la transaction. Il est important de remarquer que :

. Chaque processus associé à une transaction n'est actif à un instant donné que sur un seul site et donc, aucun parallélisme n'a lieu au niveau de l'exécution d'une transaction.

. Le parallélisme a lieu entre des transactions différentes.

Plusieurs techniques de résolution des conflits sont proposées par ROSENKRANTZ :

- WAIT-DIE system
- WOUND-WAIT system
- Les autres techniques combinent certaines propriétés des approches précédentes et des mécanismes de résolution des conflits propres aux systèmes centralisés.

Les deux techniques WAIT-DIE et WOUND-WAIT sont basées sur la prévention des interblocages : les transactions sont estampillées à l'aide d'un compteur incrémenté à chaque activation d'une nouvelle transaction. Cet estampillage est utilisé en cas de conflit pour décider si une transaction peut attendre ou si elle doit être abandonnée.

2.6.2 - Conclusion

Ces techniques de l'approche de ROSENKRANTZ sont simples à mettre en oeuvre. Elles sont basées sur deux phases de synchronisation et une prévention des interblocages. Elles ne requièrent qu'un nombre restreint de messages en vue de l'exécution d'une mise à jour en l'absence de conflit. Cependant, elles ne sont pas adaptées aux systèmes sujets aux pannes.

Etant donné l'association d'un processus unique à une transaction, il nous semble que l'extension de ces techniques au cas des systèmes défaillants ne devrait point poser de problèmes particuliers. En effet, en renvoyant un acquittement, il devrait être possible de détecter si le processus progresse ou s'il ne peut poursuivre son activité du fait de la panne ou de l'isolement du site sur lequel il était en train de s'exécuter.

2.7 - APPROCHE DE STONEBRAKER

2.7.1 - Description du fonctionnement

Dans [STON 79], plusieurs techniques sont décrites. Elles gèrent le contrôle des accès concurrents aux bases de données dupliquées ou partitionnées. Une notion de base sur laquelle reposent les algorithmes de

cette approche est celle de site primaire : à chaque transaction est associé un site primaire. Toute mise à jour d'une donnée est d'abord effectuée sur le site "primaire" de la transaction.

Le verrouillage des seules entités manipulées est réalisé de manière distribuée; par contre, la détection et la résolution des conflits sont centralisés sur un site appelé site "arbitre".

Cette approche comporte une synchronisation à deux phases : durant la première phase, chaque site "esclave" (autre que le site "maître" ou "primaire") renvoie directement un acquittement au site maître si la transaction n'est pas en conflit avec une autre transaction en cours; s'il y a conflit, ce site esclave doit avertir l'arbitre, ce qui accroît le nombre de communications et le temps de réponse.

2.7.2 - Conclusion

Cette approche maintient une cohérence faible entre les copies.

Un caractère centralisé dans la gestion de la cohérence est donné par la présence du site "arbitre"; cette solution comporte l'avantage de ne devoir maintenir le graphe des attentes que sur un seul site, et de nécessiter donc moins de communications, par contre, en cas de panne de l'arbitre, toute cette information recueillie par l'arbitre est perdue.

2.8 - APPROCHE DE THOMAS

2.8.1 - Description du fonctionnement

Dans cette approche /THOM 79/, au cours du premier pas de synchronisation, l'exécution d'une mise à jour nécessite l'obtention d'un consensus majoritaire : le consensus ne peut être obtenu que si une majorité de moniteurs "vote" l'acceptabilité de la transaction.

La transaction en cours d'examen est résolue et acceptée au premier pas de synchronisation, passe au deuxième où elle atteint son point de validation et le site qui détecte le consensus diffuse alors l'ordre de mise à jour à l'ensemble des différents sites.

Les règles de vote sont établies afin de résoudre les problèmes posés par les transactions concurrentes; chaque contrôleur, lors de l'examen d'une transaction, prend l'une des quatre décisions suivantes :

. Vote OK : si aucune variable ne possède une estampille périmée et si la transaction n'entre pas en conflit avec une autre.

. Vote REJ : le rejet est prononcé du moment qu'il existe une variable dont l'estampille est périmée.

. Vote PASSE : le contrôleur ne se prononce pas si la requête entre en conflit avec une requête en cours d'exécution plus prioritaire.

. Vote ATTEND : dans les autres cas, la requête reste momentanément bloquée sur ce site en attendant que les transactions simultanées conflictuelles soient résolues.

2.8.2 - Conclusion

Un mécanisme d'estampillage est mis en oeuvre afin de classer les transactions utilisant les données communes (identification du site et horloge physique relative au site). Une cohérence faible est assurée entre les copies multiples. Cette technique résiste aux pannes dans le cas où un grand nombre de sites est interconnecté.

Cependant, le grand nombre d'informations de contrôle rend cette approche complexe dans le mécanisme de consultation.

2.9 - CONCLUSION

Il est montré dans [BERN 81] que les techniques de contrôle de concurrence respectent la sérialisation, il reste à trouver la(les) meilleure(s) du point de vue performances.

Les approches basées sur l'ordonnancement par estampillage vérifient que les accès aux données suivent l'ordre des transactions. Si cet ordre n'est pas respecté, la transaction est avortée pour être reprise ultérieurement.

Les approches fondées sur le verrouillage deux phases évitent les conflits en posant des verrous sur les données utilisées de façon à garantir la cohérence; le verrouillage doit être en deux phases, une première phase de verrouillage de toutes les données et une deuxième phase de déverrouillage, c'est-à-dire libération des données en fin de transactions.

Cette deuxième pouvant amener une situation de verrous mortels, doit être complétée par une technique éliminant cette possibilité.

Dans ce chapitre, nous avons donc analysé les principales techniques de contrôle d'accès concurrents; parmi celles que nous n'avons pas présenté, notons l'étude de /BERN 81/ qui aboutit à identifier 48 techniques différentes basées sur ces classes : l'ordonnancement par estampillage et le verrouillage deux phases.

La validité de ces techniques étant prouvée, le problème est de faire une évaluation qualitative et quantitative des performances du système.

Dans les deux chapitres suivants, nous comparons les performances en définissant deux classes de critères :

- les critères qualitatifs
- et les critères quantitatifs.

CHAPITRE 3 - EVALUATION QUALITATIVE DES TECHNIQUES DE CONTROLE DES ACCES CONCURRENTS

3.1 - INTRODUCTION

Dans ce chapitre, nous étudions les critères qualitatifs qui permettent d'examiner une caractéristique fondamentale d'une technique de contrôle étudiée, dès lors de comparer différentes approches sous un angle particulier.

Ces critères ont été obtenus à partir d'une analyse des techniques gérant la concurrence d'accès simultanés et des bilans établis antérieurement. Certains de ces critères ont été évalués lors du développement du système de contrôle d'accès concurrents /BERG 84, PARTIE II/.

A la fin de ce chapitre, nous dressons un tableau récapitulatif où nous mentionnons les principales techniques.

3.2 - DEFINITION DES CRITERES

Il existe un nombre important de critères qualitatifs de comparaison; certains sont communs à toutes les techniques et sont destinés à préciser l'approche utilisée en vue de contrôler et de maintenir la cohérence, d'autres permettent de juger le niveau de performance des techniques, ce qui permet de comparer leur efficacité respective.

Les différents critères retenus sont :

- *Le mécanisme de contrôle de concurrence*

Il a pour tâche la résolution des problèmes posés par les accès concurrents. On distingue deux types :

. le mécanisme de contrôle de concurrence centralisé
. et le mécanisme de contrôle de concurrence réparti.

- *Les outils de désignation unique*

Ils permettent d'établir un ordre entre les transactions et de sérialiser leur exécution sur chacun des sites afin d'obtenir une version de la base analogue sur chaque site; cet ordre peut être défini en associant une date à toute transaction, cette date est fournie par une horloge physique ou une horloge logique ou par l'attribution de tickets.

- *Les critères de résolution des interblocages*

Le problème de l'interblocage se pose d'une manière aigüe dans les applications réparties. Quatre solutions sont apportées : le séquencement initial, l'ordonnancement initial, la réclamation à l'avance des ressources et le graphe des attentes.

- *Le type de verrouillage et la granularité des verrous*

Le type de verrouillage consiste au verrouillage de toutes les copies, de certaines copies ou absence de verrouillage.

Par contre, la granularité des verrous consiste en l'affectation des verrous à la totalité de la base ou à une portion de celle-ci.

- *Le niveau de parallélisme*

Il consiste à exécuter une transaction ou plusieurs transactions sur un ensemble de sites.

- *La résilience* ou niveau de résistance aux pannes.

- *Type de transactions* ou type d'actions des transactions prises en compte tels que un ensemble d'actions de lecture ou d'écriture ou de lecture-écriture, ...

- *Type d'allocation* ou type d'allocation de ressources nécessitée par l'ensemble des transactions.
On distingue : l'allocation dynamique, l'allocation pseudo-dynamique et l'allocation statique des ressources à l'initialisation des transactions.

- *Le niveau de cohérence* ou cohérence fournie :
cohérence faible, cohérence forte des copies de la base.

3.3 - ETUDE DES CRITERES

3.3.1 - Mécanisme de contrôle de la concurrence

Le mécanisme de contrôle de la concurrence ou contrôleur, peut être réalisé :

- de manière centrale et unique pour l'ensemble des sites par un contrôleur spécialisé,
- de manière répartie par un ensemble de contrôleurs qui collaborent à l'exécution des transactions.

3.3.1.1 - Contrôleur central

Ce type de mécanisme consiste à choisir un des sites comme site central chargé de la gestion de contrôle de concurrence; le contrôleur n'est pas nécessairement affecté statiquement à un site déterminé, mais il peut être affecté à tout site en cours d'exécution d'une transaction. Ainsi nous distinguons deux types de contrôleur central :

Contrôleur central fixe ou site primaire

Toute transaction émanant d'un site quelconque est dirigée après initialisation vers le site primaire qui est chargé de résoudre tous les problèmes de concurrence. L'un des inconvénients est le risque d'engorgement sur ce site.

Contrôleur central non fixe

Le contrôle est également centralisé, mais il n'est pas assigné à un site déterminé; pour qu'un site soit autorisé à lancer une transaction pour exécution, il doit acquérir le droit de contrôle. Ce droit unique sur le réseau est détenu par le site qui a initialisé la dernière transaction. Lorsqu'un site a l'intention de lancer une transaction, il le signale au site possesseur de ce droit appelé aussi administrateur.

Ce mécanisme nécessite la gestion de la file des sites en attente de lancement des transactions et le transfert de cette file d'un administrateur à un autre.

3.3.1.2 - Contrôleur réparti

Pour aboutir à une décision commune, l'acceptation ou le rejet d'une transaction peut nécessiter un échange de messages de contrôle entre plusieurs sites; nous parlerons du contrôle de concurrence réparti.

Ce type de mécanisme requiert une étape de synchronisation durant laquelle un protocole d'accord est déroulé.

On distingue deux types de contrôleur réparti :

Contrôleur réparti sans exécutions concurrentes

Ce mécanisme consiste en la diffusion, par le site initialisant une transaction, d'un message de synchronisation à l'ensemble des sites; ce message est destiné à avertir l'ensemble de ces sites d'une exécution prochaine de cette transaction. Après l'achèvement de l'étape de synchronisation, si la transaction est acceptée par l'ensemble des sites, sa validation aura lieu et toute autre transaction ne peut être traitée ni acceptée. Dans ce type de contrôleur, il n'y a qu'une seule transaction acceptée globalement : c'est la transaction possédant la plus haute priorité.

Dans [ELLI 77, LELA 78 a], ce message est transmis de proche en proche par un anneau virtuel, par contre, dans [ROSE 78], il est transmis suivant un ordre quelconque avec possibilité de passages multiples pendant cette étape.

Ce mécanisme est contraignant, car il n'y a pas d'exécutions concurrentes même s'il n'existe aucune entité commune entre les transactions.

Contrôleur réparti avec exécutions concurrentes

Dans ce mécanisme, l'étape de synchronisation consiste en un vote effectué par l'ensemble des sites sur l'acceptation ou le rejet d'une transaction [LELA 78 a, BERN 80, ROSE 78].

L'acceptation ne peut être choisie que si aucune entité manipulée par la transaction courante n'intervient dans une transaction concurrente en cours; un certain niveau de parallélisme est donc admis dans l'exécution des transactions et la résolution des conflits d'accès est prise en compte à l'aide des priorités des transactions.

Nous concluons, que le nombre de communications et la densité de communications sont élevés dans le contrôleur central fixe et faibles s'il est non fixe; par contre, dans le contrôleur réparti, le nombre de communications est élevé et la densité est équi-répartie.

3.3.2 - Outils de désignation unique

Il est nécessaire de définir les outils permettant d'identifier de manière unique les sites initiateurs et les transactions.

Nous avons vu que l'exécution concurrente d'un ensemble de transactions doit être munie d'un mécanisme permettant de sérialiser ces différentes transactions.

De ce fait, il est important d'établir une relation d'ordre entre ces transactions en leur associant une date lors de leur initialisation; cette date peut être obtenue, soit par un mécanisme d'horloge logique, soit en consultant une horloge physique, soit par l'attribution de tickets.

3.3.2.1 - Utilisation d'horloges physiques

L'utilisation d'une horloge physique unique à tous les sites permet de dater les événements survenus en différents endroits du réseau [LAMP 76]. Cet outil pose des problèmes dûs aux temps de transmission nécessaires à la détection des événements [LELA 78 b].

La résolution de ce problème est de conserver une horloge individuelle (ou estampille) sur chaque site qui est réajustée périodiquement; chaque horloge dérive l'une par rapport à l'autre.

Dans [ROSE 78, BERN 80, PARTIE II], il est à remarquer que la reinitialisation d'une transaction rejetée garde la même estampille déjà assignée, on dit alors qu'une transaction est d'autant plus ancienne que l'estampille associée est petite, par contre, dans [THOM 79], une nouvelle estampille est assignée à la transaction rejetée lors de sa reinitialisation.

3.3.2.2 - Utilisation d'horloges logiques

Dans cet outil, l'estampille est remplacée par un compteur dont les valeurs sont des entiers strictement croissants.

Le compteur local à chaque site, est réajusté à la valeur du numéro de la transaction traitée si sa valeur est supérieure à celle du site.

Ces horloges formant un temps logique appelé aussi temps virtuel, sont utilisées dans /ELLI 77, HERM 79/.

Dans /ELLI 77/, une transaction avance en parcourant un anneau virtuel, et n'est bloquée sur un site que si ce site a initialisé une transaction plus prioritaire et non encore validée; par contre, dans /HERM 79/, une transaction n'est autorisée à s'exécuter sur un site que si les deux conditions suivantes sont réalisées :

- la transaction est la plus ancienne des transactions à traiter sur ce site,
- ce site possède au moins une transaction en attente de traitement provenant de chacun des sites.

Ces compteurs n'établissent qu'un ordre partiel pour l'ensemble des transactions, mais il serait intéressant d'obtenir un ordre total sans adjoindre à l'horloge physique ou logique l'identification du site initiateur.

3.3.2.3 - Attribution de tickets

La technique de /LELA 77/ remplit la remarque ci-dessus, les transactions sont identifiées par des numéros appelés tickets; ces tickets sélectionnés par les différents sites, sont alloués d'une manière distribuée.

Sur l'anneau virtuel circule un jeton de contrôle unique qui contient la première valeur disponible du ticket, lorsqu'un site est en possession de ce jeton, il est autorisé à s'allouer un certain nombre de tickets consécutifs à partir de la valeur fixée dans le jeton; lorsque ces tickets sont octroyés à un site, la valeur du jeton est égale à

l'ancienne, augmentée du nombre de tickets alloués; après l'obtention de cette nouvelle valeur, le jeton est transféré au site voisin.

3.3.2.4 - Utilisation du degré d'actualité

Le degré d'actualité consiste à identifier séquentiellement des transactions. Chaque site conserve un numéro correspondant à l'identification de la dernière transaction validée par ce site.

Ce numéro est attribué par le site administrateur de la base qui est le seul apte à donner une valeur d'identification à une transaction.

Le degré d'actualité d'une transaction est comparé à celui de la dernière transaction exécutée sur ce site; cette vérification garantit que tous les sites exécutent les transactions dans le même ordre. La défaillance de l'administrateur (site central) nécessite la recherche de la dernière "écriture" pour identifier le degré d'actualité le plus élevé.

3.3.2.5 - Conclusion

Les horloges physiques garantissant une relation d'ordre partiel, utilisent une information supplémentaire, le numéro d'identification du site, et provoquent une surcharge du réseau en manipulant des messages de resynchronisation.

Les horloges logiques, garantissant aussi une relation d'ordre partiel, utilisent l'information supplémentaire mais n'utilisent pas de message particulier de resynchronisation.

L'utilisation des tickets garantit une relation d'ordre total, évite l'identification du site, mais nécessite la circulation du jeton sur le réseau, donc la présence d'un réseau en anneau virtuel.

Le degré d'actualité garantit aussi une relation d'ordre total, mais c'est une méthode centralisée.

L'outil de synchronisation (ou d'identification) conseillé est l'horloge logique avec l'utilisation des files d'attentes multiples sur chaque site; les horloges physiques provoquent un retard, les tickets et le degré d'actualité augmentent le nombre de transactions de décalage.

3.3.3 - Critères de résolution de l'interblocage

Dans cette section, nous allons étudier les solutions possibles utilisées dans les différentes techniques de contrôle. Les trois premières sont des méthodes de prévention de l'interblocage et la suivante de détection de l'interblocage.

1 - Le séquencement initial : il consiste à ordonner les transactions à l'initialisation, ce qui permet de les traiter ultérieurement dans l'ordre sans risque d'interblocage; l'assignation des estampilles aux transactions conduit également à un pré-séquencement /BERN 80, HERM 79/.

2 - L'ordonnancement initial des sites permet aussi un traitement des transactions suivant l'ordre prédéfini des sites /ELLI 77/.

3 - La réclamation à l'avance des ressources nécessaires sur l'ensemble des sites; elle permet de rejeter ou de faire attendre les transactions conflictuelles non prioritaires. Cette solution adoptée dans l'un des contrôleurs de type prévention /PARTIE II/, et dans /ROSE 78, GARD 79, THOM 79/ est adaptée pour les BD dupliquées et pour les BD partitionnées.

4 - Graphe des attentes

La détection dynamique des interblocages est basée sur l'établissement d'un graphe sur lequel un cycle est recherché; l'existence d'un cycle symbolise un interblocage sur le réseau entre les transactions en cours de traitement sur les sites.

. Dans /STON 79/ la détection des interblocages n'est pas réalisée de manière distribuée; lorsqu'un site s'aperçoit qu'une transaction entre en conflit avec une transaction en cours, un site unique sur le réseau appelé 'arbitre' est averti du conflit existant; il a pour tâche de détecter les interblocages grâce à un graphe global des transactions en attente.

. Dans le système SDD-1 [BERN 80], un graphe local géré sur chacun des sites du réseau, associe à chaque classe de transactions une paire de noeuds représentant les opérations de lecture et d'écriture effectuées sur chaque site.

. Dans d'autres techniques de contrôle, le graphe de conflits a une autre signification. A chaque transaction est associé un noeud et un arc orienté est tracé de la transaction détectant le conflit vers la transaction conflictuelle [BERG 84].

Dans les applications réparties, la dernière technique est plus coûteuse que celle décrite dans SDD-1, car elle nécessite un nombre important de messages sur le réseau si l'ensemble de transactions en blocage est important.

La solution de [STON 79] est centralisée, car le stockage des informations et toute vérification de l'état du graphe a lieu sur un même site; ceci a l'avantage de donner une vue globale de l'état du système, mais une panne a de lourdes conséquences : tous les renseignements sauvegardés par le site défaillant doivent être recherchés, ce qui augmente le temps d'attente des transactions en attente.

Conclusion

Nous représentons sur le tableau 3.3.3 les différentes solutions possibles pour la résolution de l'interblocage, nous mentionnons le domaine d'application et les conséquences qui en découlent et enfin, nous spécifions pour chaque solution, les techniques de contrôle de la concurrence l'utilisant.

Technique	Domaine d'application	Conséquences	Techniques
PREVENTION	B D R	Etape de synchronisation	THOM 77, BERN 78 LELA 78, ATRO 84 ELLI 77 ROSE 78
DETECTION	B D R	- nombre de transmissions élevé - tenue du graphe	BERG 84 BERN 80 STON 79

Tableau 3.3.3 : Résolution de l'interblocage

3.3.4 - Granularité des verrous et type de verrouillage

Le type de verrouillage et la granularité influent de manière déterminante sur les performances du système, sur le type de cohérence assurée et essentiellement sur le degré de parallélisme entre les transactions.

3.3.4.1 - Granularité des verrous

Dans la plupart des techniques de contrôle de la concurrence d'accès, la notion de granularité n'est guère explicitée alors que la taille des granules joue un rôle considérable sur le degré de parallélisme.

Nous distinguons différentes tailles du granule :

a - La totalité de la base

Dans le cas le plus défavorable, l'exécution d'une transaction requiert un verrouillage de la base entière, quelque soit le nombre d'entités manipulées par la transaction : dans ce cas le granule s'identifie à la copie et provoque la dégradation des performances en fournissant un degré de parallélisme nul.

Cependant une telle technique est intéressante si le nombre de transactions initialisées par unité de temps sur l'ensemble des sites est réduit, car elle implique une gestion facile des verrous : il suffit de tester sur chaque copie l'état d'un seul verrou.

b - Une partie de la base

D'autres approches utilisent une granularité fine des verrous (partie de la base), cette nature ou taille de granule autorise un parallélisme qui peut croître avec la finesse de verrouillage.

Plus le nombre d'entités logiquement indépendants est élevé plus le nombre de verrous est élevé, il en résulte que sur chaque site une zone mémoire importante de taille proportionnelle au nombre de verrous soit prévue et que la gestion de ces verrous peut rapidement devenir lourde : le temps nécessaire au balayage, lors de la vérification sur chaque site, des verrous libres pour chaque transaction peut devenir non négligeable.

c - Une entité de la base

Les approches de THOMAS et ROSENKRANTZ utilisent la variable comme unité de verrouillage.

Dans le système SDD-1 /BERN 80/, le granule est défini par les classes de transactions, les interblocages entre classes sont détectés par l'élaboration de conflits.

Plusieurs autres tailles de granules pourraient être définies. Ces entités peuvent être des entités logiques tels que les variables, les tableaux, les segments ou des entités logiques, tels que les pages, les pistes, les cylindres, tous les verrous associés à ces derniers objets sont des verrous physiques. On peut avoir des verrous à prédicats /ESWA 76/ pouvant s'appliquer à une relation entre les différentes entités d'une BD, ou un tuple particulier.

Conclusion

Avant de dresser un tableau récapitulatif (tableau 3.3.4.1), nous dirons que le problème crucial réside dans le choix d'un granule 'bien adapté' et que le degré de granularité n'est pas déterminé par les approches, cette tâche revient aux administrateurs de la base.

Granularité	Verrouillage	Degré de parallélisme	Cohérence	Performance
Copie	Global	Nul	Forte	Mauvaise si nombre de transactions augmente
Variable d'une transaction	Partiel	Elevé	Faible	Bonne
Classe de transactions	Partiel	Elevé	Faible	Bonne
Tuple d'une transaction	Partiel	Faible	Faible	Variable

Tableau 3.3.4.1 : Granularité des verrous

3.3.4.2 - Type de verrouillage

Nous distinguons deux types de verrouillage :

- le verrouillage global

Il consiste au verrouillage global de chacune des copies de la base; ce type de verrouillage maintient une cohérence forte entre les copies, mais ne permet aucun parallélisme dans l'exécution des transactions.

- le verrouillage local

D'autres techniques garantissent le maintien de la cohérence sans exiger le verrouillage global, elles utilisent un verrou logique local. Ce verrou est obtenu dans [BERN 80] et [HERM 79] à l'aide d'un ensemble de files d'attente des transactions sur chaque site, une par site distant.

Dans [LELA 78 b], le verrou est obtenu par l'attribution de tickets aux transactions qui peuvent s'exécuter dans un ordre séquentiel sur chaque site.

L'existence de ce verrou logique local contribue à l'augmentation du degré de parallélisme dans le traitement des transactions.

Conclusion

Dans le tableau 3.3.4.2, nous indiquons les relations entre le type de verrou, le type de cohérence et le niveau de parallélisme, nous mentionnons enfin le nombre de verrous à gérer par chacun des types.

Type de verrou	Nombre de verrous	Parallélisme	Type de cohérence
Global pour l'ensemble des copies	1 seul verrou	Nul	Forte
Local	N files d'attentes	Elevé	Faible

Tableau 3.3.4.2 : Type de verrouillage

3.3.5 - Niveau de parallélisme

Le niveau de parallélisme est examiné sous deux aspects différents :
- le parallélisme interne à l'exécution d'une transaction,
- le parallélisme dans l'exécution de plusieurs transactions.

3.3.5.1 - Parallélisme intra-transactionnel

Nous parlons de parallélisme interne à l'exécution d'une transaction si plusieurs processus physiques situés sur des sites distants collaborent à l'exécution d'un ensemble d'actions d'une même transaction.

Dans les techniques de contrôle [HERM 79, BERN 80, GARC 79], après initialisation d'une transaction, un message est envoyé à chaque site afin que chacun d'eux participe à l'étape de verrouillage et renvoie à l'initiateur un message d'acceptation ou de rejet de la transaction en cours : dans ce cas, un sous-ensemble de processus (ou sites) participe donc simultanément à l'exécution d'une transaction; avec une telle solution le temps de réponse n'est guère influencé par le nombre de sites participant à son exécution.

3.3.5.2 - Parallélisme inter-transactionnel

L'étude du parallélisme à l'exécution simultanée de plusieurs transactions n'a pas de sens si la granularité est réduite à la base toute entière, l'existence d'un seul verrou pour la copie entière (verrouillage global) empêche toute exécution simultanée entre transactions.

L'exécution de l'étape de synchronisation peut être exécutée en parallèle pour plusieurs transactions, mais à la détection d'un conflit une des transactions est acceptée alors que les autres sont rejetées.

Une granularité fine permet un haut niveau de parallélisme qui est limité par le nombre de processus distants traitant l'ensemble des transactions sur le réseau. Le degré de parallélisme dans l'exécution des transactions ne diminue pas généralement le temps de réponse, car il nécessite une lourde gestion des accès concurrents.

Conclusion

Nous donnons un tableau récapitulatif mettant en relief les caractéristiques liées au parallélisme inter-transactionnel. (Tableau 3.3.5)

Verrouillage	Granularité	Niveau de parallélisme	Temps de réponse	Cohérence
Global total	La copie	Nul	Elevé	Forte
Global partiel	Une entité	Important suivant les entités distinctes	Réduit	Faible
Local	/	Total entre site	Très réduit	Faible

Tableau 3.3.5 : Parallélisme inter-transactionnel

3.3.6 - Résilience (Résistance aux pannes)

3.3.6.1 - Description des pannes

Dans les systèmes répartis, il est nécessaire de tenir compte du facteur 'panne'.

Nous mentionnons dans cette partie, les principales pannes perturbant le bon fonctionnement du système, puis nous étudions le comportement de certaines techniques dans un milieu défaillant.

Les différentes pannes sont :

- panne d'un noeud : elle est définie par l'impossibilité pour l'ensemble des autres noeuds de collaborer avec ce noeud,
- panne d'une ligne logique : toute panne provoquant la rupture de communication entre deux ou plusieurs noeuds est appelée "Panne de ligne logique"; la rupture de lignes logiques provoquant le partitionnement du réseau en sous-réseaux peut ne pas garantir la cohérence de la BDR.

D'autres pannes sont provoquées :

. par une combinaison quelconque des pannes précédemment citées,
. par la transformation volontaire des messages entre l'émission et la réception,
. par la production de messages parasites.

La panne d'un processeur situé sur un site quelconque empêche le contrôleur associé de contribuer à la stratégie de mise à jour : si un des contrôleurs attend un message de tous les autres sites, la défaillance de l'un d'eux peut provoquer l'interruption du protocole de gestion du contrôle des accès concurrents des copies pendant toute la durée de la panne.

Toutes ces pannes sont résolues par la "reprise"; le redémarrage des noeuds en panne nécessite la remise à niveau de leurs copies, c'est-à-dire que l'ensemble des sites qui sont restés en activité doivent être capables de réstituer aux sites en panne des informations stockées dans les 'journaux'.

Un service est dit résilient, s'il est capable d'être assuré malgré l'indisponibilité d'une partie matérielle du réseau. On dit aussi qu'un service est résilient d'ordre N s'il peut être assuré tant que le nombre de sites en panne n'atteint pas N.

3.3.6.2 - Influence des pannes

Une panne surgissant durant l'exécution d'une transaction peut être une source d'incohérences si certains sites ont validé une transaction alors que d'autres n'en ont pas eu connaissance. De ce fait, deux phases s'avèrent indispensables à l'exécution de toute transaction.

- A la réception d'une transaction, tout contrôleur doit mémoriser l'effet de cette transaction sur un support fixe, mais les modifications ne sont faites qu'à la deuxième phase : au moment de la validation.
- Après la mémorisation de la transaction, soit le contrôleur passe la transaction au successeur /ELLI 77, LELA 79/, soit il renvoie un acquittement /BERN 78/. Dans le premier cas, lorsque la transaction a fait un tour complet sur l'anneau virtuel et que le site initiateur la récupère, ou dans le deuxième lorsque le site maître a reçu les acquittements de tous les contrôleurs présents : on dit que le point de validation est atteint. Ce point représente l'instant auquel la transaction est effectivement prise en compte par l'ensemble des copies.

Nous remarquons que cette même technique n'est pas à l'abri de tout risque d'incohérences; nous illustrons cette incohérence par l'exemple suivant :

Supposons qu'au moment où le point de validation vient d'être atteint, une défaillance sur le réseau se produit, un seul contrôleur autre que l'initiateur a reçu la commande de validation, dans ce cas il n'y aura que deux qui vont prendre en compte cette transaction. Ainsi, pour combler ce risque d'incohérences, une autre technique de validation à trois phases a été proposée dans SDD-1 [BERN 80], elle consiste à partager l'étape de validation en deux parties séparant l'annonce de la validation de son opération à l'aide d'un délai fixé.

3.3.7 - Allocation des ressources

3.3.7.1 - Introduction

La ressource est définie comme une entité d'une BD à laquelle sont associées des règles d'utilisation et de partage : accès exclusif, accès partage avec ou sans limitation du nombre d'utilisateurs.

Dans un système réparti, la notion de transactions, définie comme unité utilisatrice de ressources, peut impliquer l'exécution d'un processus unique ou de plusieurs processus localisés sur des sites distincts.

Un processus demandant une ressource doit envoyer un message à l'allocateur de celle-ci.

La charge d'un système,désignant l'ensemble des messages soumis à à un allocateur, est caractérisée par le nombre, la nature et la répartition dans le temps des messages qui le constitue.

L'allocateur assure plusieurs fonctions de régulation :

- une fonction de distribution de charge : répartit les processus entre un ensemble de ressources,
- une fonction de régulation de charge globale : limite le nombre de processus autorisés pour certaines ressources.

3.3.7.2 - Mécanismes d'allocation

Le problème de l'allocation des ressources peut être résolu localement si la base de données est entièrement redondante, un seul contrôleur sur n'importe quel site peut vérifier si les contraintes d'intégrité sont respectées et si toutes les ressources sont disponibles sur le même site. Dans le cas où les ressources nécessaires à l'exécution de cette transaction se trouvent dispersées sur plusieurs sites, un sous-ensemble de contrôleurs doit collaborer lors de l'allocation des ressources.

Dans [LELA 79], l'allocation est pseudo-dynamique, la première phase consiste à allouer les ressources qui restent préemptibles tant que cette étape n'est pas terminée; à la deuxième étape, les ressources sont bloquées et ne sont plus préemptibles, durant cette étape, la transaction est exécutée puis les ressources sont libérées. Ce mécanisme ne nécessite pas la connaissance de toutes les ressources à l'initialisation de la transaction; c'est un protocole qui permet aux différents contrôleurs d'acquérir dynamiquement les ressources [BERN 80, BERG 84].

Les techniques décrites dans les chapitres suivants et dans [ROSE 78] autorisent aussi une allocation dynamique des ressources puisqu'un seul processus associé à chaque transaction visite l'ensemble des sites concernés et acquiert sur chacun d'eux les ressources nécessaires, sans les modifier avant de les avoir toutes acquises. Les techniques de [THOM 79, STON 79] utilisent une allocation dynamique des ressources à l'initialisation des transactions.

3.3.8 - Types de transactions

Jusqu'à présent, nous avons défini un certain nombre de critères qualitatifs de comparaison qui permettent de classer les différentes approches.

Cependant il reste une notion très importante qui peut influencer le choix de telle ou telle approche; il s'agit du genre de transactions qui peut simplifier certaines parties des mécanismes proposés :

- Aucun verrouillage global n'est nécessaire si les transactions consistent uniquement en des affectations de valeurs à certaines variables sans modification; un verrouillage local en mode partagé est nécessaire afin qu'une transaction consistant uniquement en des lectures ne soit pas interrompue, en cours d'exécution, par une transaction modifiant ces mêmes valeurs. En général, l'absence de verrouillage est possible si toutes les transactions suivent la règle :

  f (g (x)) = f (x), pour toutes opérations f, g sur une variable x.

  Dans ce cas, le mécanisme de contrôle de concurrence peut être réduit à une simple vérification du numéro d'identification de la transaction.

- Pour les applications dont les opérations sur les transactions consistent en des additions et des soustractions, aucun verrouillage n'est requis et la linéarisation dans l'exécution des transactions n'est pas non plus nécessaire, mais cette absence de verrous empêche d'effectuer des lectures cohérentes.

Dans le système SDD-1, le type de transactions a été étudié en détail, le partitionnement pré-spécifié en classes de transactions permet de définir celles qui n'entrent pas en conflit; l'exécution des transactions appartenant à des classes différentes est autorisée.

Pour une application donnée, une étude approfondie de type de transaction doit être menée afin de guider le choix de la technique appropriée.

3.3.9 - Niveau de cohérence

Nous avons déjà introduit dans le chapitre 2 les notions de cohérence forte et de cohérence faible. Ainsi, une cohérence forte n'est assurée que si les techniques sont centralisées et qu'un verrouillage global est appliqué à la totalité de la base [ELLI 77]; ces techniques interdisent tout parallélisme et maintiennent la cohérence forte des copies.

Un verrouillage local assure une cohérence faible des copies et permet un certain parallélisme entre les transactions.

Les dépendances entres le verrouillage, le parallélisme et la cohérence sont résumées dans le tableau 3.3.5.

3.4 - COMPARAISON DES TECHNIQUES DE CONTROLE

3.4.1 - Domaine d'application

Une première classification peut être effectuée sur base du domaine d'application de ces différentes techniques de contrôle. Si la plupart des techniques traitent le problème de maintien de la cohérence des BD dupliquées, le domaine d'application de certaines d'entre elles s'étend au maintien de la cohérence dans les BD partitionnées.

Le problème de maintien de la cohérence des informations partitionnées constitue un problème aussi crucial que celui des informations totalement dupliquées; ils se rejoignent couramment dans les BDR.

Ainsi, il est intéressant de s'orienter de préférence vers les techniques qui ont des mécanismes permettant de gérer ces deux types de problèmes [ROSE 78, LELA 79, BERN 80].

Cependant, un certain nombre de problèmes posés par les BD partitionnées ne doivent pas être pris en compte par les BD dupliquées et vice versa, notamment :

- l'allocation dynamique des ressources sur un ensemble de sites distants,
- le problème de maintien de cohérence et de l'intégrité ne peut pas être vérifié localement.

Certaines techniques basées sur la gestion des BD dupliquées ne sont pas appropriées pour gérer les BD partitionnées, il en est ainsi que le verrouillage global de l'ensemble des copies lors de l'exécution de chaque transaction empêche tout parallélisme, alors que le verrouillage ne doit être appliqué qu'à certaines partitions.

3.4.2 - Classification des techniques

Les différentes techniques de contrôle étudiées seront classées (Tableau 3.4.2) sur la base des critères que nous avons précédemment énoncés.

Les critères prédominants pris en compte sont :

- le type de BD gérées,
- le mécanisme de contrôle de la concurrence,
- la technique de résolution des conflits,
- les outils de synchronisation,
- la possibilité de verrouillage,
- le niveau de parallélisme,
- le type d'allocation des ressources,
- le type de cohérence assurée.

D'autres critères non négligeables, tels que :

- le mécanisme physique et logique de transmission de messages : diffusion ou anneau virtuel,
- le niveau de résilience,
- le mécanisme de reprise en cas de pannes,

peuvent compléter cette classification.

3.4.3 - Conclusion

Toutes les approches étudiées présentent une caractéristique commune : le contrôle de la concurrence assurent la sérialisation des transactions et le traitement des interblocages.

Suivant le type de mécanisme de contrôle de la concurrence, nous pouvons distinguer quatre grandes classes :

. La classe des techniques à contrôle centralisé,
. La classe des techniques à contrôle local [HERM 79],
. La classe des techniques à contrôle partiellement réparti [BERN 80, LELA 79],
. La classe des techniques à contrôle totalement réparti [ROSE 78, STON 79].

- Les deux premières classes ne sont pas appropriées :
  - . aux bases de données dupliquées,
  - . au réseau sujet à des pannes fréquentes : l'absence du contrôleur primaire provoque l'interruption du fonctionnement et nécessite l'élection d'un nouveau site primaire. Elles n'offrent qu'une allocation statique des ressources.

- Les deux dernières classes, choisies pour le développement d'une variété de contrôleurs [PARTIE II, BERG 84], possèdent un domaine d'application plus vaste en gérant les bases de données dupliquées et les bases de données partitionnées, en allouant dynamiquement des ressources. En effet, la politique d'acquisition dynamique des ressources, amenant une immobilisation des ressources moins importante, sont plus avantageuses que les politiques d'acquisition statique [BENM 82].

Critères \ Approches		BERNSTEIN	ROSEN-KRANTZ	LELANN	HERMAN VERJUS	THOMAS	ELLIS	STONE-BRAKER
Type de BD	BDD	oui	oui	oui	oui	oui	oui	oui
	BDP	oui	oui	oui	non	non	non	non
Contrôle de concurrence	SM	non	non	non	non	non	non	oui
	V	non	non	non	non	oui	non	non
	CR	oui	oui	oui	oui	non	oui	non
Résolution des conflits	R	oui	oui	non	non	oui	non	oui
	ATT	oui	oui	oui	oui	oui	oui	non
	VG	non	non	non	non	non	non	non
Synchronisation	E	oui	oui	non	non	oui	non	non
	C	non	non	non	oui	non	oui	non
	T	non	non	oui	non	non	non	non
	F	non	non	non	oui	non	non	oui
Cohérence		faible	faible	forte	faible	faible	forte	faible
Verrouillage partiel		oui	oui	oui	non	oui	non	oui
Type d'allocation		Dynamique	D	S,D	Statique	S	S	S
Parallélisme	Inter	oui	oui	oui	oui	oui	oui	oui
	Intra	oui	non	oui	oui	non	non	oui
Transmission logique de messages		Diffusion	Anneau virtuel	Anneau virtuel	Diffusion	Diffusion ou anneau virtuel	Anneau virtuel	Diffusion
Transmission physique de messages		Diffusion non fiable	Point à Point	Diffusion ou point à point	Diffusion fiable	Diffusion non fiable	Point à point	Diffusion ou point à point
Résilience		Paramétable	Non significative	N Sites	N Sites	N/2 Sites	Extension à N Sites	N Sites
Reprise en cas de panne pannes		oui	non significative	oui	oui	oui	Extension	oui

Tableau 3.4.2 : Classification des techniques

4.1 - INTRODUCTION

Une classification qualitative des techniques n'est pas suffisante pour pouvoir juger leurs efficacités respectives; ainsi nous nous sommes penchés, dans ce chapitre, sur la définition de certaines paramètres permettant d'obtenir une idée plus précise des performances, en particulier :

- le temps d'attente moyen,
- le nombre moyen de transactions rejetées ou mises en attente,
- le temps nécessaire à l'exécution d'une transaction,
- la taille moyenne de files d'attente.

Nous terminons ce chapitre avec une comparaison des différentes techniques sur la base de ces critères.

4.2 - DEFINITION DES CRITERES

Dans le cas général, les différents paramètres qui influent sur la simulation [BENM 82] ou l'étude numérique d'une technique sont subdivisés en plusieurs catégories :

. le taux d'arrivée des transactions,
. le contenu d'une transaction,
. les caractéristiques du réseau,
. la vitesse d'exécution des processeurs et d'E/S

4.2.1 - Taux d'arrivée des transactions

Le taux d'arrivée des transactions est défini comme le nombre total de transactions arrivées sur un site par unité de temps. Dans une simulation, comme dans une étude numérique, il s'avère indispensable de séparer les transactions de lecture, qui n'auront d'influence significative que sur le taux de charge de l'unité centrale traitant ces transactions, des transactions d'écriture (de mise à jour).

Temps moyen écoulé entre deux transactions d'écriture

Les transactions en écriture nécessitent un verrouillage global et des transmissions de messages sur le réseau.

Le nombre d'arrivées des transactions d'écriture peut être représenté par une loi de Poisson de moyenne (ME); il est évident que le temps écoulé entre l'arrivée de deux transactions suit une loi exponentielle :

la probabilité P (arrivée de la prochaine Ti $<$ t) = $1 - e^{-t/ME}$

Le taux d'arrivée des transactions (λE) représente le nombre moyen de transactions émanant d'un site local, c'est-à-dire initialisés sur ce site et celles provenant des sites distants.

Temps moyen écoulé entre deux transactions de lecture

Dans la plupart des applications, de nombreuses transactions se réduisent à des opérations de lecture; pour ne pas effectuer des lectures sur des entités en cours de modification, ces transactions doivent être verrouillées localement.

Une distribution exponentielle analogue, de moyenne ML, est adoptée pour exprimer le temps écoulé entre deux transactions consécutives de lecture :

$$\lambda L = \lambda L, i = \frac{1}{ML}$$

$$ML = ML, i$$

A partir des expressions précédentes, nous déduisons le trafic sur un site (ou le taux d'arrivée de transactions)

$$\lambda = \lambda L + \lambda M$$

4.2.2 - Taille de la BD et des transactions

Taille de la base de données : TB

La taille de la base de données (TB) peut être exprimée de diverses manières : le nombre d'entités contenues dans la base, le nombre de pages occupées en mémoire, le nombre d'enregistrements, le nombre de piste occupées, etc...

Taille d'une transaction : NE

La taille d'une transaction est définie comme le nombre d'entités (NE) manipulées par l'ensemble des actions de la transaction : ce nombre suit une distribution exponentielle de moyenne NE et est évidemment limité par la taille TB de la base de données.

De nouveau, ces entités peuvent être les variables de la base ou des pages mémoire.

Il serait utile aussi de définir la loi régissant la distribution des accès sur l'ensemble des entités, car généralement, elles ne sont pas utilisées avec la même fréquence. Il a été montré que cette loi est fortement liée à l'application constituée.

Nombre de granules : NG

L'évaluation de la probabilité que deux transactions concurrentes entrent en conflit, nécessite en plus de la taille de la base et de la taille de la transaction, le nombre de granules manipulés par la transaction; en effet, l'exécution d'une transaction nécessite au préalable le verrouillage des entités à manipuler car un verrou n'est pas toujours associé à toute entité. Le traitement d'une transaction peut être accordé, si tous les granules nécessaires à la transaction sont verrouillés.

On définit à l'aide de NG et de TB, la finesse de verrouillage comme le rapport suivant :

$$TNE = NG/TB$$

Si $TNE \approx 0$, c'est-à-dire $NG \ll TB$, on dit que la granularité est "grande", par contre, elle est appelée "fine", si $NG \equiv TB$ et dans ce cas, on associe à chaque entité un verrou ($TNE = 1$).

Nombre de copies de la base : NC

En raison des pannes, nous distinguons deux sortes de copies : les copies présentes (CPR) qui participent à la gestion du contrôle de la concurrence et les copies absentes (CAB) qui sont rattachées à des sites en panne.

Le nombre de copies NC = CPR + CAB

Remarque :

Dans le cas des BDD, nous faisons l'hypothèse qu'un site possède au plus une copie de la base; par contre, dans les BDP, le nombre de copies présentes CPR représente l'ensemble des partitions impliquées dans une transaction donnée, tandis que le nombre de copies absentes CAB est l'ensemble des partitions absentes et des partitions non impliquées dans la transaction.

4.2.3 - Paramètres du réseau

L'existence d'un réseau de communication nécessite l'évaluation du taux de pannes; pour cela un certain nombre de facteurs doivent être analysés :

Nombre de sites : NS

A partir de l'expression analytique ci-dessus, nous déduirons les relations suivantes :

$$NS \quad CPR + CAB = NC$$

Le nombre de sites présents : $NSP \geqslant CPR$

Le nombre de sites absents : $NSA \geqslant CAB$

avec $NS = NSP + NSA$

Taux de pannes : TP

Le calcul du taux de pannes fait appel aux deux paramètres suivants :

TFM : durée moyenne d'activité (ou de fonctionnement) entre deux pannes consécutives

TPM : durée moyenne d'une panne pour un site donné.

$$TP = \frac{TPM}{TPM + TFM}$$

Probabilité de panne d'un sous-ensemble de sites : PP

La valeur du taux de pannes permet de calculer la probabilité que NSA sites d'un sous-ensemble soient simultanément en panne :

P (site présent) = TP

P (site absent) = 1 - TP

$$P \text{ (NSA sites absents)} = C_N^{NSA} (1 - TP)^{NSP} \cdot TP^{NSA}$$

Cette dernière formule n'est valable que dans le cas où les pannes sur les différents sites se produisent indépendamment.

Durée de transmission : DT

Un autre paramètre du réseau, est le temps nécessaire à la transmission d'un message d'un site à un autre.

Ce paramètre est constant si le réseau est faiblement chargé où si la charge est uniformément distribuée sur les différents sites, par contre, si le réseau est composé de sites distants, il faut tenir compte des différentes valeurs de DT entre les couples de sites.

4.2.4 - Temps d'Exécution et d'Entrée/Sortie : TEES

Il consiste au calcul du temps d'exécution d'une transaction sur un site donné, ainsi que du temps prévu pour les opérations d'entrée/sortie.

Si nous disposons d'un réseau de communication de haute performance, ou si le réseau ne compte pas de sites géographiquement dispersés, le temps d'exécution et d'E/S devient non négligeable en comparaison de la durée de transmission DT.

Quantum de temps d'exécution : QTE

La durée d'exécution d'une opération élémentaire est appelée quantum de temps d'exécution : cette opération peut consister en une insertion d'éléments dans une file, à la recherche dans la table des verrous, à la comparaison d'estampilles, etc...

Temps d'entrée/sortie : TES

En raison du manque d'espace de la mémoire centrale et de la taille de l'information à stocker, les données ne sont pas directement accessibles en mémoire : pour ce faire, ces opérations d'E/S sont nécessaires.

De plus, si l'information de contrôle nécessaire à la gestion des copiées est volumineuse, elle sera elle aussi stockée en mémoire secondaire.

L'exemple d'une approche associant un verrou distinct et une estampille montre qu'il est impossible de stocker cette information en mémoire centrale.

Temps de calcul d'une nouvelle entité : TCE

C'est le temps requis pour la modification d'une entité par une action d'une transaction; si une transaction modifie x entités sur un même site, le temps total de calcul sera x . TCE.

Dans les BDD, un seul processeur est en activité sur un site, ainsi, il n'y a pas de parallélisme sur ce site; par contre, dans les BDP, il est réalisé en parallèle par les différents sites.

Temps d'attente avant reinitialisation : TAR

Dans certaines approches de contrôle de concurrence, un conflit peut conduire au rejet de la transaction le provoquant; lorsque celle-ci est rejetée, la reinitialisation est inutile durant un certain temps (attente de l'exécution de la transaction concurrente soit terminée).

La transaction est reinitialisée après une période fixe TAR.

4.3 - COMPARAISON QUANTITATIVE DES TECHNIQUES DE CONTROLE

4.3.1 - Introduction

A partir des paramètres que nous venons de définir, il est possible de calculer les performances d'un système et d'aboutir à des résultats pour la comparaison quantitative de plusieurs techniques.

Nous présentons dans cette section :

- le temps de réponse,
- le nombre de messages,
- la charge d'un site,
- la probabilité de conflit,
- l'impact des pannes et
- la mesure de la cohérence des copies multiples.

4.3.2 - Temps de réponse d'une transaction : TM

Le temps de réponse à une transaction de mise à jour est l'intervalle de temps délimité par les deux extrémités : début de transaction et fin de transaction.

Le temps de réponse se décompose en plusieurs périodes :

La période d'initialisation : PI

Elle correspond à la formation de transaction à sa compilation, à la vérification des contraintes d'intégrité, à la validation de la transaction et à la recherche des verrous associés aux entités accédées.

La période d'attente : PA

L'apparition d'un conflit provoque une durée d'attente pour chaque transaction conflictuelle; si les conflits sont résolus par rejet, cette période est remplacée par le temps d'attente avant reinitialisation TAR.

Le temps total d'exécution locale : TTEL

C'est la durée totale passée sur chaque site au verrouillage des entités, au calcul des nouvelles entités et aux opérations d'entrées/sorties.

Le temps total de transmission : TTT

Cette valeur correspond à la durée totale des transmissions.

Le temps total perdu à cause des rejets : TTR

Cette valeur correspond au temps passé au traitement des transactions rejetées au moment du rejet.

Ces différentes valeurs nous permettent de calculer le temps de réponse T_M pour une transaction donnée :

1 - Si les conflits sont résolus par attente :

$$T_M = PI + PA + TTT + TTEL$$

2 - Si les conflits sont résolus par rejet :

$$T_M = PI + TTR + TAR + TTT + TTEL$$

4.3.2.1 - Comparaison des techniques suivant le temps de réponse

Le temps de réponse dépend essentiellement de deux facteurs :

- la technique de transmission (séquentielle ou à diffusion)
- la probabilité de conflit provoquant le rejet ou l'attente de la transaction.

a - Temps de réponse en absence de conflits

En absence de conflits, le temps de réponse se décompose en :

. Période d'initialisation : PI
. Temps total d'exécution locale : TTEL
. et temps total de transmission : TTT.

Les valeurs de PI et de TTEL varient en fonction de la granularité des verrous, par contre, celle de TTT dépend :

- de la technique de transmission
- du nombre de pas de synchronisation
- et de la nécessité ou non d'attendre un acquittement après l'envoi d'un message.

Généralement, la valeur du temps de réponse sans conflit nous amène à choisir la classe des techniques pour lesquelles le temps de transmission est indépendant du nombre de sites et dont la finesse de verrouillage est faible.

b - Temps de réponse en cas de conflits

Dans ce cas, le temps de réponse compte un terme supplémentaire qui décrit le temps nécessaire à la résolution des conflits.

Les techniques peuvent être divisées en trois catégories :

- Catégorie où toute détection de conflit conduit à l'attente d'une transaction [HERM 79, ELLI 77].
- Catégorie où toute détection de conflit entraîne le rejet d'une transaction [BERN 80, THOM 79, BERG 84].
- Catégorie pour laquelle les deux précédentes peuvent être adaptées [ROSE 78, STON 79, PARTIE II].

Les temps supplémentaires de TM, fonction de plusieurs facteurs, rend la classification délicate.

4.3.3 - Nombre de messages : NMM

Le nombre de messages échangés par une transaction de mise à jour varie suivant les techniques de contrôle. Cependant, deux valeurs sont significatives pour le calcul du nombre moyen de messages :

. le nombre de messages circulant sur le réseau pour une transaction en absence de conflits : NMSC

. le nombre de messages causés par une situation de conflit : NMAC.

Nous aurons alors NMM = NMSC + K.NMAC où K est un facteur dépendant du nombre de rejets.

Il faut remarquer que les messages échangés entre l'utilisateur et le site initiateur ne sont pas pris en compte.

4.3.3.1 - Comparaison quantitative des techniques de contrôle suivant NMM

Nous avons vu que ces différentes techniques sont classifiées suivant les valeurs de NMSC et NMAC.

Nous dirons alors qu'une technique est performante, si en absence de conflits, elle ne nécessite que l'envoi d'un seul message sur chacun des sites distants. Le nombre de messages transmis sur le réseau pour l'exécution d'une transaction croît linéairement avec le nombre de copies présentes sur le réseau et varie suivant le type de mécanisme de contrôle de la concurrence.

La solution fournie par [HERM 79] est très performante dans le cas où la charge est importante ou uniforme.

La solution de [STON 79] fournit une valeur de NMSC très élevée dans le cas où chaque message doit être acquitté.

Les autres techniques ont une valeur de NMSC constante qui peut être réduite à l'aide de la phase de pré-validation [BERN 80, ROSE 78].

4.3.4 - Charge d'un site

La charge d'un site dépend du type de contrôle et de la loi de distribution des transactions sur les sites. Des techniques distinctes peuvent influer différemment sur la charge des sites : si le contrôle de la concurrence est centralisé, la charge du site "maître" est très importante par rapport à celle des sites "esclaves"; si le contrôle est réparti, la charge est pratiquement uniforme sur l'ensemble des sites.

Il peut être intéressant de mesurer le nombre de transactions validées par unité de temps (débit moyen) :

$$DBT = \frac{1}{KE.TE + KL.TL} \qquad \text{où}$$

TE et TL sont les temps de réponse de transactions d'écriture et de lecture
KE la proportion de transactions en écriture,
et KL la proportion en lecture.

Nous remarquons que si le type de contrôle est centralisé, il y risque d'engorgement, ainsi il faut prévoir un espace de stockage important dont la taille dépend principalement du nombre de transactions initialisées par unité de temps et de la quantité d'informations reprises.

4.3.4.1 - Comparaison quantitative des techniques de contrôle suivant la charge

La charge des différente sites est déterminée par l'espace de stockage des informations relatives au contrôle de la concurrence, par la taille des files d'attentes comportant les transactions en attente d'exécution et par le temps passé à l'exécution d'une transaction sur chaque site.

La charge des sites n'est pas toujours également répartie, un rapport entre la charge minimum et la charge maximum pour une transaction est définie de façon à déterminer si la répartition est uniforme.

L'espace de stockage est important pour les techniques de contrôle qui détectent les conflits [BERN 80, BERG 84].

La taille des files d'attente dépend du nombre moyen de transactions qui peuvent entrer en conflit, ce nombre de transactions est lui-même fonction du taux d'arrivée des transactions.

4.3.5 - Probabilité de conflit: Pc

La probabilité de conflit dépend de trois facteurs :

- le nombre de granules (NG)
- le nombre de verrous utilisés par chaque transaction (NV)
- la probabilité de concurrence sur un site (Pcc).

1 - *Nombre de granules : NG*

correspond au nombre de granules manipulés par les transactions conflictuelles. Ce nombre est un paramètre qui est déterminé par simulation. Nous remarquons que le nombre de granules n'est pas le nombre total d'objets dans la base.

2 - *Nombre de verrous : NV*

Le nombre de verrous utilisés par chaque transaction est proportionnel au nombre d'objets manipulés par la transaction; ce nombre dépend donc de la 'finesse' de verrouillage.

Nous en déduisons que la probabilité Pc pour que deux transactions T1 et T2 entrent en conflit, dépend du nombre de verrous utilisés par T1 et du nombre de verrous de T2.

3 - *Probabilité de concurrence : Pcc*

La probabilité de concurrence permet d'atteindre une situation de conflit.

Cette probabilité est étudiée suivant deux cas :

- probabilité d'entrée en conflit d'une transaction d'écriture et d'une transaction quelconque en cours,
- probabilité d'entrée en conflit d'une transaction de lecture et d'une transaction d'écriture en cours.

4.3.6 - Impact des pannes

La plupart des approches de contrôle de concurrence d'accès simultanés tiennent compte du facteur "panne"; ainsi, il est utile de définir quelques critères pouvant être évalués :

- le taux de résilience,
- le temps nécessaire à la reinitialisation complète du système,
- le temps de réponse pour la remise à niveau des copies multiples,
- le nombre de messages requis pour la remise à niveau d'une copie,
- le nombre de messages requis pour la reinitialisation complète du système,
- le nombre de copies concernées pour la remise à niveau,
- et la durée d'interruption dans le fonctionnement normal.

4.3.6.1 - Comparaison quantitative des différentes techniques de contrôle suivant le facteur panne

Afin de conserver la cohérence des informations lors des pannes, un échange de messages supplémentaires en fonctionnement normal du site est nécessaire; cet échange provoque une dégradation des performances pour certaines techniques [HERM 79, LELA 79, THOM 79, STON 79].

Le redémarrage du système dans [BERN 80] nécessite la mise en place du graphe des conflits.

Les techniques de [LELA 78, HERM 79, BERN 80] ne peuvent pas faire appel à des procédures spéciales de redémarrage, car aucun site ne joue le rôle de privilégié.

Dans d'autres techniques, un des sites joue ce rôle; la panne de ce site provoque une dégradation du système; il en résulte une interruption du service normal dont la durée est connue à l'aide du délai de redémarrage.

4.3.7 - Mesure de la cohérence des copies multiples

Nous distinguons :

1 - Le niveau de version des copies

A partir d'une variable booléenne, nous pouvons caractériser à un instant donné, le niveau de version des copies, c'est-à-dire que la variable est vraie ou fausse suivant l'identité des copies.

2 - Le nombre de mises à jour séparant deux copies

Un compteur associé à chaque copie permet d'évaluer le nombre de mises à jour effectuées entre deux copies.

3 - La distance basée sur les estampilles

Dans certaines applications, des mises à jour peuvent être insérées, alors que des mises à jour antérieures affectant les mêmes variables ne sont pas prises en compte, ainsi les deux premières notions citées ci-dessus ne sont plus valables. D'autres distances sont proposées : elles consistent à comparer à un instant donné les estampilles d'une même variable sur des sites différents.

4 - La distance temporelle

Elle permet d'évaluer le retard entre les différentes copies; ce retard est obtenu en comparant les estampilles correspondant à la transaction la plus récemment validée sur chacun des sites.

4.4 - CONCLUSION

L'évaluation en fonction d'un critère donné ne fournit pas un choix décisif sur une technique de contrôle : une technique peut être performante pour un paramètre sans pour autant l'être pour un autre.

En effet, suivant le taux d'arrivée de transactions, il est utile de choisir les techniques de contrôle qui fournissent un temps de réponse long en l'absence de conflit, mais en compensation minimisent la probabilité de conflit. En régime peu chargé, le nombre moyen de messages échangés par les transactions est proche de celui sans conflit, la technique d'HERMAN n'est pas appropriée; par contre, en régime saturé, il est nécessaire de rechercher les approches dont la surchage due au conflit est faible et la granularité fine.

Dans sa première phase de développement, le système de gestion de base de données EASY, en voie de développement, est mono-utilisateur, le but de cette deuxième partie est la conception et la réalisation d'un système de contrôle d'accès concurrents CAC, le rendant multi-utilisateur.

Dans un SGBD, le contrôle de la concurrence d'accès doit permettre à plusieurs utilisateurs de manipuler une base de données tout en garantissant la cohérence de celle-ci.

De nombreuses techniques permettent d'arriver à ce but [PARTIE I 7], mais la diversité et le type d'application imprécis de ces techniques de contrôle d'accès concurrents décrites par divers auteurs, nous ont amenés à développer une variété de contrôleurs, mettant en oeuvre ces différentes techniques, qui seront évalués puis utilisés suivant le type d'application.

Parmi elles, deux classes de techniques de prévention dynamique ont été implémentées :

- la technique basée sur l'ordonnancement par estampillage - OE -,
- la technique basée sur le verrouillage deux phases avec prévention des interblocages - 2PLP -.

Les deux méthodes de la première classe choisie (2PLP) évitent les conflits en posant des verrous sur les informations utilisées, de manière à éviter les incohérences des données; ce verrouillage doit être à deux phases c'est-à-dire que les verrous ne peuvent être relâchés qu'à la terminaison de la transaction. Cette méthode, pouvant amener une situation de verrou mortel, doit être complétée par une technique prévenant l'interblocage. La méthode de la deuxième classe développée, est l'ordonnancement par estampillage des transactions, cette méthode vérifie que l'accès aux informations par les transactions suit l'ordre des estampilles; si cet ordre n'est pas respecté, une des transactions est rejetée pour être reprise ultérieurement.

Les deux classes de prévention dynamique des interblocages consistent à mettre en oeuvre des techniques d'allocation de ressources évitant toute formation d'interblocages. La formation des interblocages peut être aussi évitée en ordonnant de façon unique les transactions et en utilisant cet ordre pour bloquer ou avorter une transaction en cas de conflit [MIRA 79].

Dans le premier chapitre nous présentons le système transactionnel, dans le second, nous donnons la modélisation des deux classes de contrôleurs prévenant les interblocages :

- classe de l'ordonnancement par estampillage,
- classe du verrouillage à deux phases.

Nous terminons le chapitre en comparant les contrôleurs.

1.1 - SYSTEME D'EXPLOITATION UTILISE

Le système transactionnel CAC, système de contrôle d'accès concurrents dans un SGBD centralisé (s'exécutant sur un seul site), est réalisé sur le VAX 11/730.

La série VAX 11 désigne la famille des mini-ordinateurs universels 32 bits.

Le VAX 11/730 comprend un processeur micro-programmé très rapide, qui exécute un vaste jeu d'instructions de longueur variable. Son architecture 32 bits présente un jeu d'instructions étendu, de nombreux types de données, un ensemble efficace de modes d'adressage, une gestion mémoire avec pagination à la demande et un espace d'adressage virtuel de quatre milliards d'octets (mémoire principale 1 MO + deux disques RL0 2 et R 80). Le système d'exploitation utilisé sur cette machine est VMS (Virtual Memory System).

Il clarifie les travaux à effectuer dans le système, supervise et coordonne les activités et gère la mémoire d'une manière la plus efficace possible.

VMS divise le programme en pages de taille égale à 512 octets; les pages actives sont placées en mémoire, tandis que les autres sont en mémoir auxiliaire.

Les avantages d'un tel système sont :

- la taille des programmes n'est pas limitée par la mémoire centrale,
- la segmentation des programmes n'est pas nécessaire.

Il est à remarquer que les contraintes dues à la gestion physique de l'unité centrale et des périphériques ne sont pas considérées dans les modèles mis en oeuvre.

1.2 - STRUCTURE DE DONNEES

1.2.1 - Organisation des données

Les différents concepts qu'utilise le modèle ENTITE/ASSOCIATION sont :

- l'ensemble d'entités,
- l'ensemble d'associations,

- l'attribut,
- l'ensemble des valeurs.

Pour conserver une représentation identique de ces différents concepts, la translation du diagramme d'Entités/Associations en diagramme de structures de données est faite à l'aide d'une correspondance entre les différents ensembles et les types d'enregistrement associés (Table ou fichier).

Les différents ensembles sont représentés sous forme de tableaux d'attributs et chaque t-uple est stocké comme une suite contigüe des valeurs de ses attributs.

Les différents types de tableaux qui ont été conçus dans EASY sont :

- VALEUR : la table VALEUR contient les informations concernant les ensembles de valeurs existants dans le schéma conceptuel.
- ATTRIBUT : contient les informations concernant les attributs des ensembles d'entités ou d'associations.
- CATALOGUE : en plus des tables d'entités, associations, ensembles de valeurs et attributs, une table CATALOGUE permet d'avoir les informations sur toutes ces tables et aussi sur d'autres tables spéciales qui permettent d'accéder aux tables définies par l'utilisateur (entité, association).
- LISTE : contient les valeurs d'un ensemble énuméré du type Pascal.
- INDEX : contient les informations sur les index primaires et secondaires des ensembles d'entités et d'associations.
- LIEN : contient les informations sur les associations d'ensembles d'entités.
- ATTRIBUTS-CLE : contient les informations sur les clés.
- INVERSE : permet d'accéder aux tables définies par l'utilisateur (ensembles d'entités ou associations).
- ENSEMBLE : contient tous les t-uples d'un ensemble d'entités ou d'associations.

L'espace secondaire alloué à une base de données est composé d'un ensemble de pages chaînées, de taille égale.

L'aspect dynamique du modèle Entités/Associations, à savoir, l'évolution de la base de données au cours des applications, est résolue comme suit :

Une longueur initiale est réservée au t-uple dont la taille maximale est la page; lorsque la page du t-uple dépasse la page qui lui est assignée initialement, une page 'suite' est indiquée à l'aide d'un pointeur.

Le schéma général des pages de stockage est représenté sur la figure 1.

1.2.2 - Modes d'accès

Les différentes opérations offertes sur un fichier sont :

a - Insertion d'un t-uple : l'insertion d'un nouveau t-uple se fait dans la dernière page.

b - Suppression, modification d'un t-uple : elles sont réalisées grâce à l'indicateur d'existence de t-uple sité en en-tête (fig.1) de la page.

c - Recherche d'un t-uple : demande un balayage séquentiel de tous les enregistrements.

1.2.3 - Méthodes d'accès

Les méthodes d'accès mises en oeuvre dans EASY sont :

- La méthode séquentielle
- Les B*-arbres
- Le hachage virtuel

a - Méthode séquentielle

Tout ensemble ou toute table est stocké comme une suite séquentielle de t-uples et un accès séquentiel sur une table consiste en un parcours de cette dernière telle qu'elle est stockée physiquement.

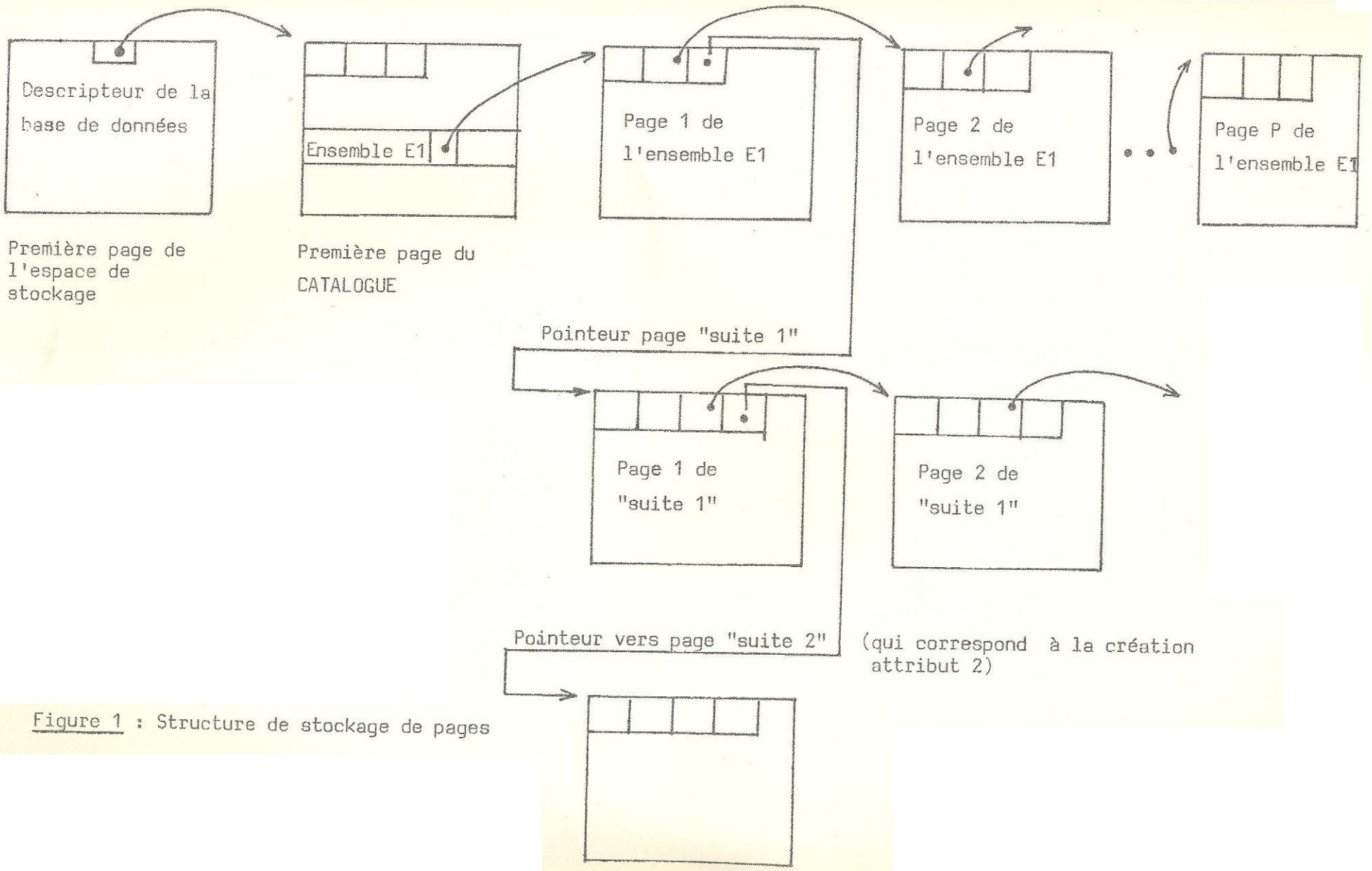

Figure 1 : Structure de stockage de pages

b - B*-arbres

Un B*-arbre d'ordre q et de profondeur p est un arbre où :

1 - chaque noeud a au plus q fils ($q \geqslant 2$)
2 - chaque noeuf, excepté la racine et les feuilles, a au moins $\left[\frac{q}{2}\right]$ fils.
3 - la racine a au moins deux fils (à moins qu'elle ne soit une feuille).
4 - toutes les feuilles apparaissent au même niveau (le niveau p)
5 - un noeud ayant K fils contient K-1 clés.

La recherche d'une clé dans un B*-arbre est réalisée comme suit : de chaque noeud, les résultats de comparaisons suivantes indiquent le chemin à suivre :

- si la clé est plus petite que celle examinée, alors on prend le chemin gauche,
- si elle est plus grande et la clé examinée n'est pas la dernière du noeud, alors on examine la clé suivante,
- on va à droitre, s'il s'agit de la dernière clé du noeud,
- on répète cette procédure jusqu'à ce qu'un noeud feuille soit atteint.

La propriété - 4 - du B*-arbre garantit l'équilibre de l'arbre, c'est-à-dire le nombre d'accès à n'importe quel t-uple est toujours le même. L'équilibre peut être maintenu, en cas d'insertion, en faisant éclater le noeud en deux et on remonte la clé médiane au niveau supérieur, la nouvelle clé est insérée dans le noeud correspondant.

c - Hachage virtuel

Dans le hachage, l'idée de base est de répartir les enregistrements dans des cases qui se composent d'une ou plusieurs pages.

On dispose d'une fonction de hachage qui fait correspondre à une valeur de clé un entier, représentant le numéro de case où se trouve l'enregistrement.

Dans cette technique, les insertions peuvent engendrer des collisions et les suppressions peuvent engendrer une perte d'espace secondaire puisqu' il est alloué statiquement, ce qui peut détériorer les performances d'accès.

Pour y remédier, l'opération de rehachage, s'avérant nécessaire, peut provoquer l'indisponibilité du système.

Cette insuffisance est surmontée par le hachage virtuel; le problème de la collision est résolu en appliquant des modifications de la fonction de hachage et seulement quelques cases du fichier sont réorganisées.

Ces deux techniques d'accès B*-arbres et Hachage virtuel ont été choisies dans EASY.

1.2.4 - Concepts de CAC

1.2.4.1 - Bases de données

Dans le système CAC, ensemble de contrôleurs, la base de données est représentée comme un fichier de tuples.

1.2.4.2 - Granule

Le tuple forme l'unité de verrouillage ou granule /¯PARTIE I_7.

1.2.4.3 - Transaction

Une transaction est une séquence atomique d'actions ou d'opérations sur les données de la base et les différents types d'actions sont :

- l'action 'début de transaction' : elle représente l'initialisation de la transaction dans le système,
- l'action 'fin de transaction' : elle identifie la terminaison de la transaction,
- l'action 'lecture' : signifie l'accès au granule en mode lecture,
- l'action 'écriture' : signifie l'accès au granule en mode écriture.

L'exemple suivant illustre la notion de transaction : soit la transaction T1 qui lit un tuple et écrit sur un autre tuple,

```
début de transaction    1
lire    A
écrire  B
fin de transaction      1
```

le numéro 1 est l'estampille de la transaction T1 [Paragraphe 2.2.2.7]; A et B sont les adresses des tuples qui se trouvent dans la table des verrous (voir ci-dessous).

1.2.4.5 - Mode de verrouillage

Les deux modes de verrouillages ou primitives utilisés par une transaction lors de la demande d'un granule sont :

- V-partage (X) = verrou utilisé pour acquérir le droit d'utilisation du granule X en mode lecture seule (mode partagé),
- V-exclusif (X) = verrou utilisé pour acquérir le droit d'utilisation du granule X en mode écriture et/ou lecture (mode exclusif).

Chaque accès à la base concerne un granule et nécessite un accès à la table des verrous.

1.2.4.6 - Table des verrous

Le système de contrôle maintient une table des granules verrouillés.

Cette table est modifiée dans deux circonstances :

1 - Demande d'accès à un granule par une transaction suivant l'état du granule. Cette demande se traduit pour la transaction, par l'allocation dynamique, l'attente ou l'avortement.

2 - Libération de tous les granules en fin de transaction (terminaison ou retour-arrière après avortement).

1.2.4.7 - Table de Journalisation

La journalisation est basée sur l'utilisation d'un Journal 'avant'. L'écriture sur le journal des images avant la modification et l'écriture sur la base des granules modifiés sont différées jusqu'à la fin de la transaction.

Le journal sert exclusivement lorsqu'une panne interrompt la transaction en écriture des images modifiées sur la base.

Le système de contrôle ne prenant pas en compte les pannes actuellement, la représentation de la journalisation reste superflue.

2.1 - MODELE 'ORDONNANCEMENT PAR ESTAMPILLAGE' OE

2.1.1 - Fonctionnement de OE

Cette stratégie consiste d'abord à définir un ordre initial des transactions en leur attribuant une estampille, et à vérifier à l'aide des estampilles des données que les accès conflictuels sont effectués dans l'ordre donné.

Si l'une des transactions conflictuelles ne respectent pas cet ordre, alors l'une d'elles est avortée pour être reprise ultérieurement.

Cette méthode utilise deux estampilles de données : une estampille de lecture (L) et une d'écriture (E).

Les procédures de lecture et d'écriture seront données dans la section suivante.

2.1.2 - Description du modèle OE

La figure 2.1.2 décrit le système OE.

Les stations (rectangles) ou modules de ce système sont :

REJET : contient les transactions répétées. La transaction détectant le conflit est tuée ou mise en attente d'une transaction /Paragraphe 2.2/.

ARSES : ayant un nombre constant d'actions pour l'ensemble des transactions, la station ARSES est la session de transactions concurrentes.
Les transactions arrivées au contrôleur ont des actions entrelacées, dont chaque ensemble d'actions d'une même transaction suit l'ordre interne de la transaction.
'ARSES' ordonne les transactions suivant leur ordre d'arrivée (début de transation) et à l'aide des estampilles les identifiant.

VERROUILLAGE : la vérification de l'état du granule se fait dans ce module.
On distingue deux principales procédures /GARD 81/ :

1 - Procédure de lecture d'un granule dont l'algorithme est :

```
Procédure Lire (T_i, d);
  Si E (d) ≤ i
    alors 'Exécuter la lecture';
          L (d) := max (L (d), i)
    sinon Rejet de la transaction T_i
  fsi
fin lire;
```

2 - Procédure d'écriture d'un granule dont l'algorithme est :

```
Procédure Ecrire (T_i, d);
  Si L (d) ≤ i
    alors Si E (d) ≤ i
            alors 'Exécuter écriture';
                  E (d) := i
          fsi
          Sinon Rejet de la transaction T_i
  fsi
fin écrire;
```

Ces deux procédures utilisent deux estampilles de données : une estampille de lecture 'L' et une d'écriture 'E', 'T_i' est une transaction d'estampille 'i' et 'd' un granule qu'elle désire lire ou écrire.

ACCES : est une file où la transaction a acquis tout ou partie de ses ressources.

VALIDATION : est la phase de validation de la transaction, c'est-à-dire que la transaction a atteint sa terminaison et que la (ou les) mise(s) à jour doit(vent) être recopiées sur la base de données cohérente (support physique).

2.2 - MODELES 'ATTENTE-REJET' AR et 'BLESSEE-ATTENTE' BA

2.2.1 - Fonctionnement de AR et BA

Le verrouillage, utilisé pour une classe de techniques de contrôle, consiste à éviter l'exécution incorrecte de transactions en faisant attendre celles voulant effectuer des accès conflictuels sur une même donnée.

Pour avoir un contrôle de concurrence d'accès correct,cette technique comparable à la méthode classique d'allocation de ressources à des processus, doit restreindre l'utilisation des actions 'verrouiller' et 'déverrouiller'. Deux phases sont ainsi obtenues :

- une première phase de verrouillage ou d'expansion :
  pendant cette phase, tous les granules demandés par une transaction doivent être bloqués au préalable et sont préemptibles.
- une deuxième phase de déverrouillage ou de réduction :
  en fin de transaction, nous avons la phase de validation et tous les verrous acquis doivent être libérés.

Une transaction est à deux phases, si après libération d'un granule, elle ne peut plus faire de demande de verrouillage d'un nouveau granule.

La technique de verrouillage pose le problème de verrous mortels, pour le résoudre, la prévention de leurs formations est possible si nous définissons un ordre initial des transactions à l'aide des estampilles et si nous interdisons les attentes entre les transactions conflictuelles ne respectant pas cet ordre.

C'est selon ce principe, inspiré des stratégies décrites dans ROSENKRANTZ /ROSE 78/ et dans LEBIHAN /LEBI 80/, que fonctionne les deux contrôleurs de prévention des interblocages AR et BA.

2.2.2 - Description du modèle AR

La figure 2.2.2 donne le schéma du système AR. Les stations (rectangles) ou modules de ce système sont :

RECHGRANU : est un module qui recherche la disponibilité d'un granule donné dans la table des verrous.

Si ens-read désigne l'ensemble des granules en lecture et ens-write celui en écriture, l'exécution concurrente de deux transactions T1 et T2 peut aboutir à un conflit si nous avons :

- ens-write (T1) $\cap$ ens-read (T2) $\neq \emptyset$
- ens-read (T1) $\cap$ ens-write (T2) $\neq \emptyset$
- ens-write (T2) $\cap$ ens-write (T2) $\neq \emptyset$

A la détection de ces conflits, réalisée dans ce module, soit la transaction attend dans la file 'ATTENTE', soit l'une des deux transactions est tuée, et dans ce cas les granules sont libérés.

La transaction tuée peut être reinitialisée immédiatement ou attendre la fin de la transaction concurrente dans la file 'REJET'.

Remarque : l'accès en lecture de deux transactions sur un même granule n'implique pas un conflit, mais nous indique qu'un granule peut être lu par plusieurs transactions pourvu qu'elles ne fassent pas d'écriture.

La gestion des conflits dans la station 'RECHGRANU' est la suivante :

Procédure 'ATTENTE-REJET' AR

Si T2 < T1 (T2 plus ancienne que T1)
alors T2 attend dans la station 'ATTENTE'
sinon T1 est tuée et mise dans la station 'REJET'
fsi
fin AR

où T1 est une transaction (estampille 1) possède le granule d

T2 est une transaction (estampille 2) demande le granule d.

Dans la technique AR, seul le processus (ou transaction) détecteur du conflit effectue un traitement particulier en vue de la résolution du conflit; le processus avec lequel il entre en conflit poursuit normalement son activité.

Si le processus détecteur a un numéro d'identification (estampille) plus ancien, il est mis en attente dans 'ATTENTE', soit de la fin d'exécution normale, soit de l'avortement du processus concurrent.

Par contre, si ce processus est plus jeune, il se 'tue' dans 'REJET', l'activité qu'il avait entamée est annulée; dans un SGBDR, les sites déjà visités sont avertis. L'activité du processus tué doit être reprise à son début.

INITIAL : ayant pour rôle l'estampillage des transactions, le module'INITIAL' maintient un compteur qui est incrémenté à chaque initialisation de transaction. Ainsi, l'ensemble de transactions simultanées, formant une session, sont estampillées chacune de manière unique.

ATTENTE : les transactions en attente de granules sont mise dans la station 'ATTENTE', pour continuer leurs exécutions après libération des granules verrouillés par les transactions conflictuelles plus prioritaires.

REJET :la station 'REJET' contient les transactions rejetées. Une transaction tuée ou en attente d'une transaction perd ses granules (ressources) et attend la fin de la transaction conflictuelle pour reprendre son exécution à son début.

EXECUTION : contient le délai d'exécution et d'utilisation des granules pour une transaction; la boucle EXECUTION-RECHGRANU est répétée tant que toutes les actions d'une même transaction ne sont pas épuisées.

VALIDATION : la transaction, atteignant sa fin, exécute la phase de validation dans cette station; cette étape consiste à recopier les modifications, se trouvant dans la zone de travail de la transaction (tampon) dans la base de données.

TBF : est un module qui rend les transactions bien-formées dès leurs initialisations.
Une transaction est bien-formée si elle obéit aux règles suivantes :

a - Aucune transaction ne pourra effectuer un accès en écriture ou en lecture d'un granule si elle n'a pas acquis un contrôle au moyen des verrous.