La cohérence dans les traitements coopératifs

Gérôme Canals, Jean Ferrié

LA COHÉRENCE DANS LES TRAITEMENTS COOPÉRATIFS

Gérôme Canals* et Jean Ferrié**
(*) LORIA - UMR 7503
Gerome.Canals@loria.fr
(**) LSI Montpellier
ferrié@lsi.fr

INTRODUCTION

Les applications et systèmes coopératifs, où plusieurs utilisateurs accèdent simultanément à des données communes, offrent des problèmes particuliers vis-à-vis de la cohérence des données qu'ils manipulent. En particulier, les approches classiques issues des systèmes d'exploitation ou des systèmes de bases de données sont prises en défaut puisque basées sur la mise en oeuvre d'accès exclusifs aux données (moniteurs, transactions isolées).
On classe généralement les applications coopératives en deux catégories : les applications *synchrones*, où les utilisateurs accèdent à des objets communs au même instant, et les applications *asynchrones*, où les utilisateurs accèdent aux objets partagés à des moments différents. Chaque catégorie voit des solutions spécifiques au problème de la cohérence des données. Nous présentons ici deux approches, l'une tournée vers les systèmes synchrones et l'autre vers les systèmes asynchrones.

LA COHÉRENCE DANS LES SYSTÈMES SYNCHRONES

Le problème est de synchroniser des personnes qui modifient en même temps le même objet (même texte, même graphique) pour rendre visible aussi rapidement que possible les modifications faites par l'un aux autres. Une façon de faire est d'établir un ordre total entre toutes les opérations et d'exécuter toutes les opérations dans cet ordre sur tous les sites, mais nous visons le développement d'outils plus adaptés aux besoins (effet immédiat des opérations locales) et pouvant se suffire dans certains cas d'un ordre partiel. On peut decomposer le problème en 3 sous-problèmes:

- le respect de la causalité entre les opérations,
- le respect de l'intention de l'utilisateur,
- la convergence des copies de l'objet.

Concernant le respect de la causalité, on peut utiliser une technique classique de vecteur d'état.
Concernant le respect de l'intention de l'utilisateur, la solution que nous proposons utilise une technique de transposition des opérations en avant et en arrière. La transposition en avant résoud les problèmes de concurrence en respectant l'intention de l'utilisateur. Si deux opérations o1 et o2 s'exécutent sur le même objet ob respectivemet sur deux site s1 et s2 à partir du même état, on exécutera sur s2 l'opération o1', la transposée en avant de o1, qui tient compte du fait que o2 a été exécutée sur ce site alors qu'elle ne l'avait pas été sur le site 1 au moment où o1 a été exécutée (resp. o'2 sue le site 1). La transposition en avant fonctionne bien lorsque les opérations s'exécutent dans le même état de l'objet. Pour assurer cette hypothèse, il peut être nécessaire de réordonner des opérations déjà exécutées. Pour cela, on s'appuie sur des opérations de transposition en arrière symétriques des opérations de transposition en avant.
Le respect de la causalité et de l'intention ne sont pas suffisants pour assurer que les copies en cours de modification sur les différents sites convergent vers les mêmes valeurs.

TRANSACTIONS ET SYSTÈMES ASYNCHRONES

Notre approche pour le maintient de la cohérence des données dans un système coopératif asynchrone est issue d'une étude sur le support à la coopération dans le développement de logiciels. Elle fait actuellement l'objet d'une généralisation vers différents domaines de l'ingénierie coopérative distribuée. Dans ce type d'applications, les acteurs coopèrent en menant en parallèle des activités modifiant des objets communs. L'aspect asynchrone de la collaboration est ici du au fait que ces acteurs ne travaillent pas forcément au même instant (les activités durent longtemps), et surtout au fait qu'ils ont besoin d'alterner périodes de travail isolé et période d'échange. Une modification ne peut donc pas être rendue visible immédiatement.

L'approche se base sur un mécanisme de transactions mettant en oeuvre un critère de correction (la *COO-sérialisabilité)* n'imposant pas l'isolation des traitements pour permettre le partage d'objets, et utilise un modèle de coopération basé sur la *copie / modification / fusion* des objets partagés : les transactions partagent un objet en possédant chacune une copie de cet objet. Ces copies sont modifiées librement, et les divergences sont ensuite réconciliées par fusion des copies.

La cohérence offerte par le critère de correction se fonde sur les points suivants :

- on suppose que les objets validés par une transactions au moment où elle commet sont cohérents individuellement et entre eux,
- une transaction peut effectuer des lectures sales à condition qu'elles soient suivies de lectures propres sur les mêmes objets avant de valider ses propres modifications.

En d'autres termes, une transaction peut rendre visible une ou plusieurs valeurs successives et provisoires d'un même objet avant de commettre. Ces valeurs provisoires (ou intermédiaires) peuvent être lues par d'autres transactions qui devront alors lire les valeurs validés (donc cohérentes) correspondantes avant de pouvoir elles-mêmes valider leurs modifications.

Des verrous mortels peuvent apparaître lorsque deux (ou plus) transactions lisent, modifient et s'échangent des valeurs provisoires et créent ainsi un cycle de dépendance. Ces verrous sont traités en groupant les transactions interbloquées. Un groupe de transaction peut alors terminer en réalisant un consensus sur la valeur finale des objets communs. Cette terminaison doit bien entendu respecter les conditions énoncées ci-dessus vis-à-vis des lectures sales faites hors du groupe.

Le protocole mis au point pour maintenir ce critère est sans verrous et n'annule jamais de transactions. Lorsque les conditions de validation ne sont pas réunies, la terminaison est refusée et la transaction reste active. Ces propriétés sont particulièrement intéressantes dans le cas d'activités à durée longue puisque les objets sont ainsi toujours disponibles et le travail n'est jamais perdu.

L'approche est complétée par un mécanisme de contraintes de transition basées sur la logique temporelle qui permet de spécifier et de garantir la cohérence des états validés par une transaction. Ce mécanisme est en particulier utilisé pour spécifier le comportement d'une transaction face à la relecture d'un objet déjà lu et éventuellement modifié localement : fusion éventuelle, propagation des modifications à d'autres objets, ré-exécution de certaines opérations.

Cette approche a été mise en oeuvre au sein d'un modèle de transactions emboîtées. Elle est actuellement généralisée pour prendre en compte la distribution et une plus grande flexibilité du modèle afin de s'appliquer au domaine élargi de l'ingénierie de conception à distance.