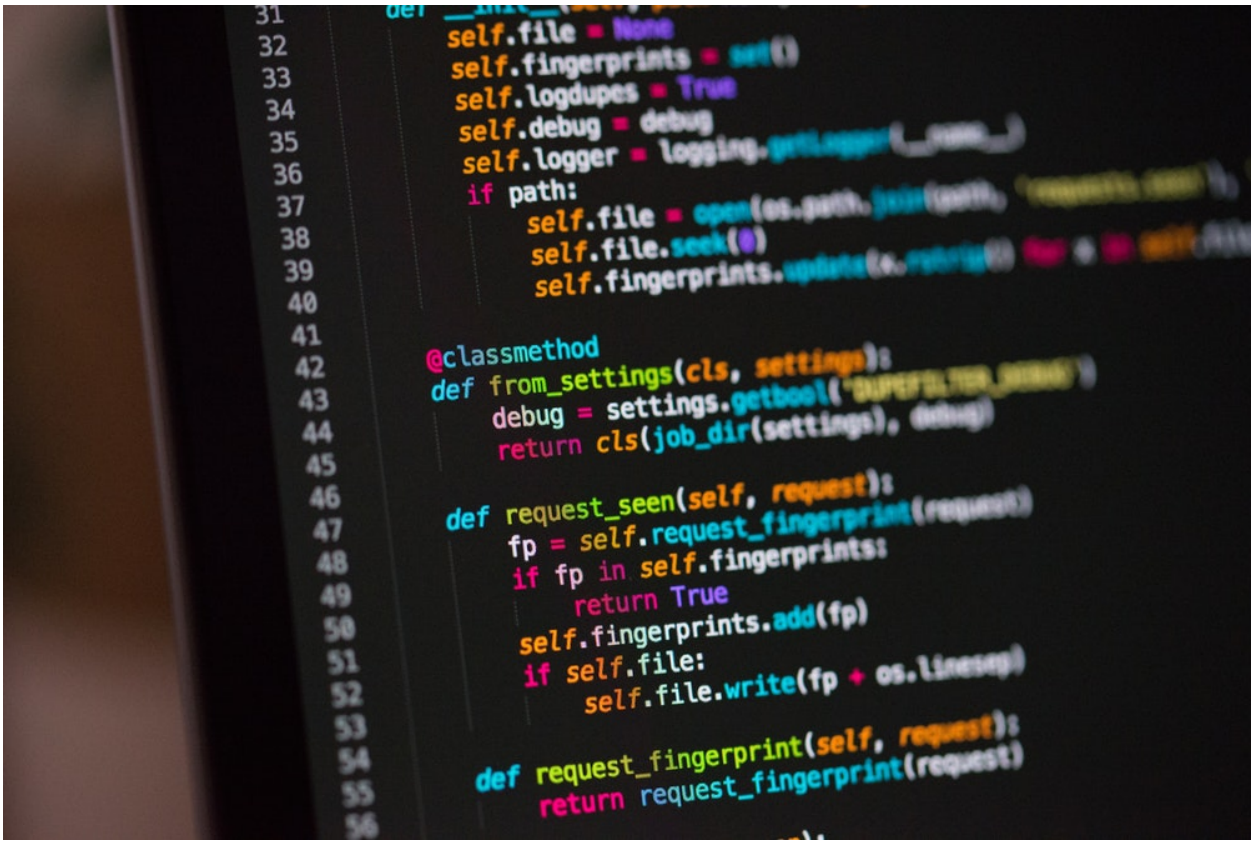
TP7 : Transactions et mémoire transactionnelle



Hamza Mouddene

21 décembre 2020

1 Exemples et scénarios d'exécution

Soit le script/scénario suivant :

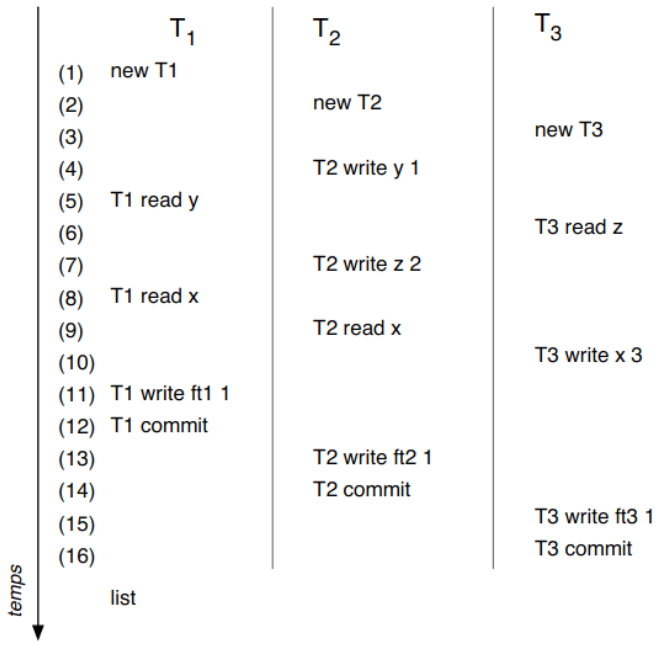


FIGURE 1 – Premier script

— TMPP : Les transactions abandonnées seront T1 et T2, la transaction validée serait T3.
TMPC : La transaction abandonnée sera T3, les transactions validées seront T1 et T2.
TM2PL : Les transactions validées seront T1, T2 et T3. T2 sera bloqué par T3 car il essaiera d'écrire dans z alors que T3 l'a lu juste avant. T1 sera bloqué par T2 car T2 écrit une valeur dans y donc T1 est bloqué quand il essaye de lire la valeur de y.
— L'ordre série équivalent obtenu sera comme le suivant :
TMPP : 6 - 10 - 15 - 16
TMPC : 5 - 8 - 11 - 12 - 4 - 7 - 9 - 13 - 14
TM2PL : 4 - 6 - 10 - 15 - 16 - 7 - 9 - 13 - 14 - 5 - 8 - 11 - 12
— Les valeurs de variables affichées par la commande list comme le suivant :
TMPP : x = 3 ; y = 0 ; z = 0 ; ft1 = 0 ; ft2 = 0 ; ft3 = 1
TMPC : x = 0 ; y = 1 ; z = 2 ; ft1 = 1 ; ft2 = 1 ; ft3 = 0
TM2PL : x = 3 ; y = 1 ; z = 2 ; ft1 = 1 ; ft2 = 1 ; ft3 = 1

Soit le script/scénario suivant :
— TMPP : Les transactions abandonnées seront T2 et T3, la transaction validée serait T1.
TMPC : La transaction abandonnée sera T3, les transactions validées seront T1 et T2.
TM2PL : Les transactions T1, T2 et T3 seront bloquées car chacune veut accéder à une valeur lue ou écrite par une autre transaction. Il y a donc interblocage.
— L'ordre série équivalent obtenu sera comme le suivant :
TMPP : 5 - 8 - 11 - 12
TMPC : 5 - 8 - 11 - 12 - 4 - 7 - 9 - 13 - 14
TM2PL : Néant
— Les valeurs de variables affichées par la commande list seront :
TMPP : x = 0 ; y = 0 ; z = 0 ; ft1 = 1 ; ft2 = 0 ; ft3 = 0 TMPC : x = 0 ; y = 1 ; z = 2 ; ft1 = 1 ; ft2 = 1 ; ft3 = 0 TM2PL : x = 0 ; y = 1 ; z = 0 ; ft1 = 0 ; ft2 = 0 ; ft3 = 0

On utilise maintenant une politique d'accès sans contrôle de concurrence (TMNoCC), avec propagation-

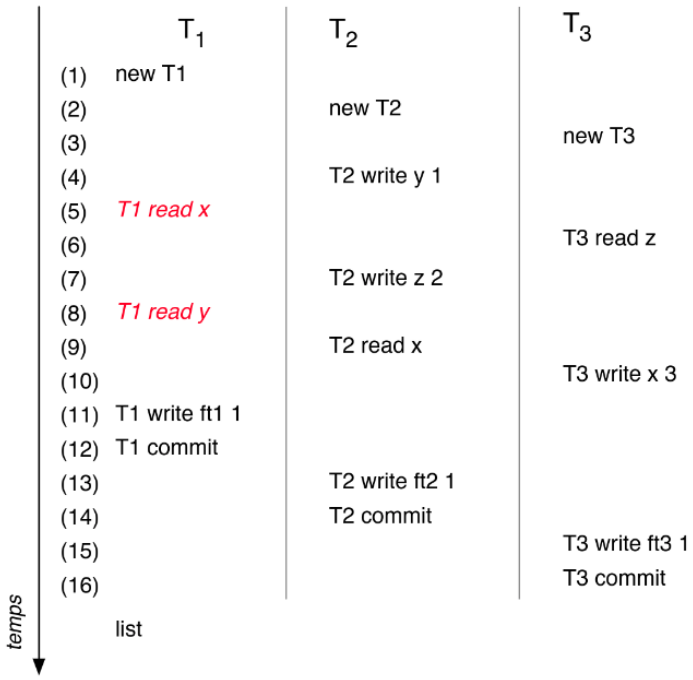


FIGURE 2 – Deuxième script

directe. Si T1 écrit une valeur dans x, T2 lit la valeur de x puis T1 annule, on a T2 qui aura lu une mauvaise valeur.

2 Évaluation de protocoles

TM2PL : T1, T2 et T3 sont validées mais on est loin du temps optimal. Les unités de temps ne sont pas perdus mais la libération des verros produit pas mal de temps d'attente . TMPC avec T : T1 et T2 sont validées, T3 est abandonnée. Si cette fois ci le temps optimal était atteint, il y'avait malheureusement des unités de temps perdues à cause de la transaction T3 abandonnée. TMPC avec S : T1, T2 et T3 sont validées. Le temps optimal n'est pas atteint mais on est plus proche que le premier cas.

TM2PL : Cette politique est plus efficace quand il y a peu d'accès concurrents à la mémoire transactionnelle, car un verrou est posé à chaque accès à une variable, que ce soit en lecture ou écriture, jusqu'à ce que la transaction soit validée. TMPC : Cette politique est plus efficace lorsque les transactions sont sérialisables puisque c'est une politique optimiste donc la sérialisabilité n'est testée qu'à chaque commit. TMPP : Cette politique est plus performante quand une transaction n'est plus sérialisable avant sa validation car TMPP est pessimiste et vérifie la sérialisabilité des transactions à chaque exécution.