Παράλληλος και Κατανεμημένος Υπολογισμός

Εργαστήριο 2

Πελάτης - διακομιστής και Αναγνώστες - συγγραφείς

Ο διακομιστής επιτρέπει πολλαπλές ταυτόχρονες συνδέσεις πελατών.

Ο διακομιστής διαχειρίζεται μια απλή δομή δεδομένων που αντιστοιχεί (χονδρικά) σε ένα πίνακα δρομολογίων ενός αεροδρομίου.

Κάθε σειρά του πίνακα έχει τη μορφή

<Κωδικός Δρομολογίου>	<Κατάσταση>	<Ώρα>
XY1234	Άφιξη/Αναχώρηση	HH:MM

Προφανώς μπορούμε να έχουμε πολύ περισσότερες λεπτομέρειες στη κατάσταση αλλά για το σκοπό του εργαστηρίου δεν είναι σημαντικό.

Υπάρχουν δύο ειδών πελάτες: οι Αναγνώστες και οι Συγγραφείς.

Οι αναγνώστες είναι πολλοί και εκτελούν συχνά απλές αναγνώσεις. Η κάθε ανάγνωση διαρκεί ελάχιστα. Μπορείτε να δημιουργήσετε ένα πελάτη αναγνώστη που να εκτελεί μεγάλο αριθμό διαφορετικών αναγνώσεων, αλλά καλύτερα να έχετε 2 ή 3 επαναληπτικούς πελάτες τουλάχιστο για να υπάρχει περισσότερος ταυτοχρονισμός.

Το μήνυμα αιτήματος του πελάτη αναγνώστη έχει τη μορφή

READ <Κωδικός Δρομολογίου>

Ο διακομιστής άπαντα στο αίτημα με ένα μήνυμα απάντησης της μορφὴς

ROK <Κωδικός Δρομολογίου> <Κατάσταση> <Ώρα>
ή
RERR

αν η αναζήτηση αποτύχει.

Οι συγγραφείς είναι λίγοι και εκτελούν λίγες εγγραφές. Όμως η κάθε εγγραφή διαρκεί πολύ. Μπορείτε να δημιουργήσετε ένα πελάτη συγγραφέα που να εκτελεί μια εγγραφή σε αραιά χρονικά διαστήματα, αλλά καλύτερα να έχετε 2 ή 3 επαναληπτικούς πελάτες τουλάχιστο για να υπάρχει περισσότερος ταυτοχρονισμός.

Το μήνυμα αιτήματος του πελάτη συγγραφέα έχει τη μορφή

WRITE <Κωδικός Δρομολογίου> <Κατάσταση> <Ὡρα>

Ο διακομιστής άπαντα στο αίτημα με ένα μήνυμα απάντησης της μορφὴς

WOK
ή
WERR

αν η αναζήτηση αποτύχει.

Τι πρέπει να κάνετε:

1. Γράψτε μια εφαρμογή πελάτη διακομιστή με βάση το κώδικα που έχετε (για υποδοχές TCP) . Αρχικά μπορείτε να έχετε στο διακομιστή ένα πολύ απλό πρωτόκολλο που απλά να επιστρέφει ως ηχώ το αίτημα του κάθε πελάτη ώστε να είστε σίγουροι ότι η επικοινωνία είναι σωστή.

2. Επιλέξτε μια κατάλληλη δομή δεδομένων και μοιράστε την στα νήματα του διακομιστή. Μπορείτε να χρησιμοποιήσετε ταυτόχρονες δομές δεδομένων αν θέλετε. Αν όχι τότε πρέπει να τη προστατεύσετε με μηχανισμούς ταυτοχρονισμού. Εδώ έχετε μια πρώτη επιλογή, σχετική με τη διακριτότητα του κλειδώματος της δομής: θα κλειδώσετε όλη τη δομή ή κάθε γραμμή ξεχωριστά; Πώς μπορεί να γίνει αυτό;

3. Επεκτείνετε το πρωτόκολλο του πελάτη συγγραφέα και του διακομιστή ώστε να έχει μηνύματα και λειτουργίες Τροποποίησης, Διαγραφής στη μοιραζόμενη δομή δεδομένων.

4. Το πείραμα αυτό μπορεί να εκτελεστεί και χωρίς να έχει υλοποιηθεί το ερώτημα 3.

Αρχικοποιείστε τη δομή και ελέγξτε την ασφαλή λειτουργία του συστήματος με το κανονικό πρωτόκολλο και πρόσβαση στη μοιραζόμενη δομή δεδομένων. Υπάρχει καθυστέρηση στους αναγνώστες όταν ένας αργός συγγραφέας εξυπηρετείται από τον διακομιστή; Αν έχετε επιλέξει κλείδωμα με υψηλή διακριτότητα τότε το πρόβλημα θα εμφανιστεί μόνο στους αναγνώστες που διαβάζουν την ίδια γραμμή με αυτή που τροποποιεί ο συγγραφέας.

Εκτελέστε σχετικά πειράματα και τεκμηριώστε τους χρόνους απόκρισης.

5. **Προαιρετικά** μελετήστε λύσεις που μπορούν να αμβλύνουν το πρόβλημα της καθυστέρησης των αναγνωστών όπως τα ReadWriteLocks ή τα StampedLocks στη Java. Δείτε το γενικότερο πρόβλημα Αναγνωστών Συγγραφὲων στο
http://www.it.uom.gr/teaching/python/TheLittleBookOfSemaphores2eGR.pdf ενότητα 4.2
και στις διαφάνειες ReadWriteDraft2019.pdf στο Compus.

Παραδοτέα:
Κώδικας (source) και απλό αρχείο αποτελεσμάτων από την εκτέλεση πειραμάτων.

Ισχύουν οι γενικές οδηγίες της πρώτης εργασίας.

Ημερομηνία υποβολής εργασίας 12/05/2019
Καταληκτική ημερομηνία προφορικής εξέτασης 06/05/2019 ώρα 13:00.