Πανεπιστήμιο Αιγαίου
Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

Κατανεμημένα Συστήματα - 1η Ομαδική Εργασία
Ημερομηνία Παράδοσης : Πέμπτη 16/05/2019

Εργαστηριακοί Διδάσκοντες: Δούμα Αναστασία

1η ΟΜΑΔΙΚΗ ΕΡΓΑΣΙΑ

Θέμα – Ανάπτυξη κατανεμημένου συστήματος με Java Sockets και RMI

ΠΕΡΙΓΡΑΦΗ ΠΡΟΒΛΗΜΑΤΟΣ

Το κατανεμημένο σύστημα που θέλουμε να αναπτύξουμε ακολουθεί το μοντέλο πελάτη - εξυπηρετητή - εξυπηρετητή και αφορά μια απλοποιημένη θεώρηση ενός κατανεμημένου συστήματος για κράτηση αεροπορικών εισιτηρίων. Το σύστημα αποτελείται από δύο εξυπηρετητές (ο 1ος εξυπηρετητής διαχειρίζεται το ***Κεντρικό Σύστημα Κρατήσεων*** και ο 2ος εξυπηρετητής την ***Κεντρική Βάση Δεδομένων***) και πολλούς πελάτες που αντιπροσωπεύουν του πιθανούς αγοραστές αεροπορικών εισιτηρίων.
Πιο συγκεκριμένα, οι πελάτες θα υποβάλουν την αίτηση τους μέσω μιας γραφικής διεπαφής (GUI) στον 1ο εξυπηρετητή. Ο εξυπηρετητής αυτός είναι υπεύθυνος για να δέχεται όλα τα αιτήματα από τους υποψήφιους πελάτες, και αφού πρώτα επικοινωνήσει με τον 2ο εξυπηρετητή, ο οποίος διαθέτει μια βάση με όλες τις πληροφορίες για τις διαθέσιμες πτήσεις, θα τους ενημερώνει για την διαθεσιμότητα των πτήσεων. Σε περίπτωση που γίνει επιλογή για κράτηση εισιτηρίων από κάποιο πελάτη ο 1ος εξυπηρετητής είναι υπεύθυνος για να ενημερώσει τον πελάτη για το εάν μπόρεσε να γίνει η κράτηση ή όχι και ταυτόχρονα να ενημερώσει και τον 2ο εξυπηρετητή.
Οι τεχνολογίες που θα πρέπει να χρησιμοποιηθούν για την δημιουργία του συστήματος είναι **Java RMI** για την σύνδεση των πελατών με τον 1ο εξυπηρετητή και **sockets ή datagrams** για την σύνδεση του 1ου εξυπηρετητή με τον 2ο εξυπηρετητή.

Πελάτες

Οι πελάτες θα πρέπει να έχουν τη δυνατότητα να αναζητούν για διαθέσιμες πτήσεις. Για αναζήτηση πτήσεων θα πρέπει να εισάγουν την *τοποθεσία αναχώρησης*, την *τοποθεσία προορισμού*, την *ημερομηνία αναχώρησης* και την *ημερομηνία επιστροφής* καθώς και τον *αριθμό των επιβατών*.

Οι πελάτες αφού θα καταχωρήσουν τις πληροφορίες για το ταξίδι που θέλουν να αναζητήσουν, θα ενημερώνονται από τον 1ο εξυπηρετητή εάν υπάρχουν διαθέσιμες πτήσεις ή όχι. Σε περίπτωση που υπάρχουν, θα εμφανίζονται στον χρήστη και θα του δίνεται η δυνατότητα να επιλέξει την πτήση που επιθυμεί να κάνει κράτηση ή να ξανά κάνει αναζήτηση. Για την ίδια μέρα αναχώρησης ή επιστροφής, θα μπορούν να υπάρχουν παραπάνω από μια πτήσεις για τον ίδιο προορισμό, αλλά σε διαφορετικές ώρες. Σε περίπτωση που ο χρήστης επιλέξει να κάνει κράτηση, θα πρέπει να περιμένει από τον 1ο εξυπηρετητή να τον ενημερώσει εάν τελικά έγινε ή όχι η κράτηση. Σε περίπτωση που δεν υπάρχουν διαθέσιμες πτήσεις ή δεν γίνει η κράτηση θα δίνεται η επιλογή στον χρήστη να ξανά κάνει αναζήτηση.

1ος Εξυπηρετητής

Ο 1ος εξυπηρετητής θα μπορεί να εξυπηρετεί παραπάνω από έναν πελάτες ταυτόχρονα. Είναι υπεύθυνος για να δέχεται τα ερωτήματα για όλα τα διαθέσιμα δρομολόγια από τους πελάτες, και αφού πρώτα επικοινωνήσει με τον 2ο εξυπηρετητή, να τους ενημερώνει για τις διαθέσιμες πτήσεις.
Επιπλέον είναι υπεύθυνος για να δεχτεί αίτημα κράτησης από τους πελάτες, να αποφασίσει εάν μπορεί να γίνει η κράτηση σε συνεργασία με τον 2ο εξυπηρετητή και να ενημερώνει τους πελάτες για την εξέλιξη της κράτησης τους.

2ος Εξυπηρετητής

Ο 2ος εξυπηρετητής διαθέτει μια συλλογή με όλες τις πληροφορίες για τις διαθέσιμες πτήσεις. Κάποιες από τις πληροφορίες που θα διαθέτει είναι:

Ημερομηνία, Ώρα, Από, Προς, Αριθμός Πτήσης, Αριθμός θέσεων, Τιμή Εισιτηρίου

Ο 2ος εξυπηρετητής αφού λάβει ένα ερώτημα από τον 1ο εξυπηρετητή (σύμφωνα με σχετικό αίτημα του πελάτη) θα κάνει αναζήτηση στις διαθέσιμες πτήσεις και θα του απαντήσει με τα αποτελέσματα. Επίσης είναι υπεύθυνος να ενημερώνει τη βάση του σε περίπτωση που υπάρχει κράτηση εισιτηρίου. Η συλλογή με τις πληροφορίες θα μπορούσε να είναι αποθηκευμένη είτε σε μια δομή δεδομένων όπως πίνακας, Array List, HashTable, είτε σε αρχείο (π.χ. αρχείο κειμένου) είτε σε βάση δεδομένων (π.χ. SQLite database). Για την εκτέλεση της εφαρμογής θα πρέπει να εισάγετε πληροφορίες πτήσης στη δομή που επιλέξετε, κατά την έναρξη του προγράμματος.

ΖΗΤΟΥΜΕΝΑ

Α) Υλοποίηση [60 μονάδες]

Να υλοποιηθεί το παραπάνω σύστημα σε Java χρησιμοποιώντας RMI και Sockets (ή Datagrams) για την διαδιεργασιακή επικοινωνία. Στην αναφορά σας θα πρέπει να περιγράψετε σύντομα:

α) τις βασικές σχεδιαστικές αποφάσεις που έχετε λάβει για την υλοποίηση της εφαρμογής,

β) τυχόν παραδοχές που έχετε κάνει, πέραν των προδιαγραφών που σας έχουν δοθεί,

γ) οδηγίες για την εκτέλεση τόσο της εφαρμογής του πελάτη όσο και των εξυπηρετητών και

δ) οθόνες εκτέλεσης της εφαρμογής με διαφορετικά σενάρια.

Β) Συνθήκες ανταγωνισμού (race conditions) [20 μονάδες]

Στην περιγραφή του παραπάνω σεναρίου, μπορείτε να εντοπίσετε κάποιες καταστάσεις που μπορούν να οδηγήσουν σε συνθήκες ανταγωνισμού; Περιγράψτε σενάρια εκτέλεσης της εφαρμογής που να δείχνουν την εμφάνιση συνθηκών ανταγωνισμού. Φροντίστε για την αντιμετώπιση αυτών των καταστάσεων στην υλοποίηση της εφαρμογής σας, με μηχανισμούς που διαθέτει η Java.

Γ) Αρχιτεκτονική/Τεχνολογία Υλοποίησης [15 μονάδες]

Ποια θεωρείται ότι είναι τα πλεονεκτήματα της αρχιτεκτονικής (πελάτης - εξυπηρετητής - εξυπηρετητής) με την οποία υλοποιείται η συγκεκριμένη κατανεμημένη εφαρμογή έναντι του απλού μοντέλου πελάτη/εξυπηρετητή; Στην παρούσα εργασία χρησιμοποιείται RMI και Sockets για την διασύνδεση των οντοτήτων του συστήματος σας. Ποια πιστεύετε είναι τα πλεονεκτήματα της υλοποίησης της εφαρμογής με RMI έναντι της υλοποίησης με Sockets;

Δ) Ασφάλεια [5 μονάδες]

Ένα ζήτημα σε μια ρεαλιστική υλοποίηση της συγκεκριμένης εφαρμογής είναι η διαχείριση της ασφάλειας του συστήματος κατά την αλληλεπίδραση πελάτη – εξυπηρετητή αλλά και στην επικοινωνία μεταξύ των δύο εξυπηρετητών. Περιγράψτε μηχανισμούς που μπορούν να χρησιμοποιηθούν για την προστασία του συστήματος. Το ζητούμενο αυτό δεν θα πρέπει να υλοποιηθεί στην τρέχουσα έκδοση της εφαρμογής.

Υποδείξεις για την υλοποίηση

- Η υλοποίηση θα περιλαμβάνει **τρία διαφορετικά projects**, το πρόγραμμα του πελάτη, το πρόγραμμα του Κεντρικού Συστήματος Κρατήσεων (1ος εξυπηρετητής) και της Κεντρικής Βάσης Δεδομένων (2ος εξυπηρετητής).
- Οι πελάτες επικοινωνούν κάνοντας χρήση Java RMI, ενώ οι εξυπηρετητές χρησιμοποιώντας Java sockets για την διαδιεργασιακή επικοινωνία.
- Ο 1ος εξυπηρετητής μπορεί να εξυπηρετεί ταυτόχρονα πολλούς διαφορετικούς πελάτες.
- Θα χρειαστεί να ορίσετε ένα request/reply πρωτόκολλο επικοινωνίας μεταξύ του πελάτη, του 1ου εξυπηρετητή 1 και του 2ου εξυπηρετητή και τη μορφή των μηνυμάτων που ανταλλάσσονται.
- Οι λειτουργίες που θα υλοποιήσετε για την επικοινωνία του 1ου εξυπηρετητή με τον 2ο θα πρέπει να υλοποιηθούν κατάλληλα τόσο στην πλευρά του 1ου όσο και στην πλευρά του 2ου εξυπηρετητή.
- Ο χρήστης επιλέγει την λειτουργία/ που επιθυμεί μέσα από ένα απλό μενού που παρουσιάζεται στην οθόνη. Μόνο για τους πελάτες θα πρέπει υλοποιήσετε γραφική διεπαφή.
- Εφόσον δεν έχετε δίκτυο υπολογιστών ώστε να τρέξετε τις διεργασίες πελάτη και εξυπηρετητή σε διαφορετικούς κόμβους μπορείτε να χρησιμοποιήσετε την διεύθυνση localhost (IP address 127.0.0.1) η οποία δίνει την ψευδαίσθηση ενός δικτύου με έναν κόμβο. Ανοίξτε διαφορετικά παράθυρα για τους πελάτες και τερματικά παράθυρα για τους δύο εξυπηρετητές.
- Διασφαλίστε ότι κάθε φορά που μια σύνδεση με τον εξυπηρετητή τερματίζεται απελευθερώνονται οι πόροι του συστήματος (δυναμική μνήμη, socket descriptors, κ.α.).

Γενικές υποδείξεις

1) Η εργασία μπορεί να εκπονηθεί από ομάδα φοιτητών (3 το πολύ), όπως έχει ήδη δηλωθεί στο e-class.

2) Η αποστολή της εργασίας θα πρέπει να γίνει μέχρι την προκαθορισμένη ημερομηνία με χρήση της ηλεκτρονικής πλατφόρμας e-class. Εκπρόθεσμη υποβολή μέχρι μια εβδομάδα χάνει το 25% της βαθμολογίας, ενώ πάνω από μια εβδομάδα δεν γίνεται δεκτή.

3) Να αναφέρετε τις βιβλιογραφικές πηγές που τυχόν έχετε χρησιμοποιήσει.

4) Η υποβολή κοινών απαντήσεων από διαφορετικούς φοιτητές ή διαφορετικές ομάδες φοιτητών δεν επιτρέπεται και θεωρείται ως **ΑΝΤΙΓΡΑΦΗ**. Η αντιγραφή έχει ως αποτέλεσμα το ΜΗΔΕΝΙΣΜΟ ΤΗΣ ΕΡΓΑΣΙΑΣ ΣΥΝΟΛΙΚΑ.

Υποδείξεις/κανόνες για τη συγγραφή και υποβολή της εργασίας

5) Μόνο ο «υπεύθυνος» της ομάδας θα πρέπει να στείλει την εργασία με μορφή συμπιεσμένου αρχείου zip ή rar. Το όνομα του αρχείου θα είναι: DSPROECT_AM.<rar|zip>. Να γίνει χρήση λατινικών χαρακτήρων για την αποφυγή προβλημάτων. Το zip|rar αρχείο θα περιλαμβάνει το pdf αρχείο της αναφοράς σας και τα project της υλοποίησης σας.

6) Ο πηγαίος κώδικας θα αξιολογηθεί ως προς το αν υλοποιεί τα βασικά ζητούμενα της εκφώνησης, εκτελείται χωρίς να προκύπτουν σφάλματα λογισμικού (bugs), ακολουθεί «καλές αρχές» προγραμματισμού (π.χ. σχολιασμό, στοίχιση, εύγλωττη ονοματοδοσία μεταβλητών, επαναχρησιμοποίηση κώδικα, κλπ).

Η εφαρμογή των παραπάνω κανόνων είναι **ΥΠΟΧΡΕΩΤΙΚΗ** και βαθμολογείται σύμφωνα με το αντίστοιχο κριτήριο αξιολόγησης.