Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

Κατανεμημένα Συστήματα

1η ΟΜΑΔΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη κατανεμημένου συστήματος με RMI

ΑΝΑΦΟΡΑ

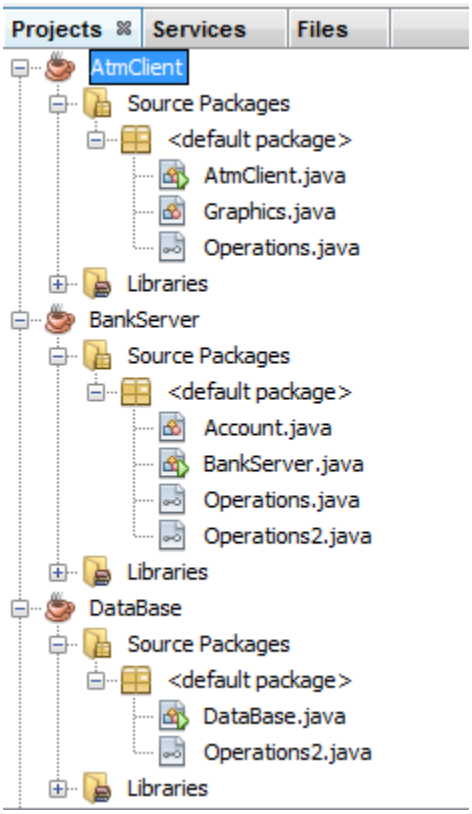
Λίλιος Ανδρέας.....321/2011085

Κοτόρρι Γιόνισον.....321/2011073

Κόσσυφας Θεόδωρος.....321/2012089

Α) Υλοποίηση

ΣΧΕΔΙΑΣΤΙΚΕΣ ΑΠΟΦΑΣΕΙΣ



Αρχικά, φτιάχνουμε τρία project για την υλοποίηση server-server-client. Τα AtmClient, BankServer, DataBase.

AtmClient: Στο συγκεκριμένο project υλοποιείται το κομμάτι του πελάτη μιας τράπεζας.Χρησιμοποιούμε την κλάση της main μεθόδου, την κλάση Graphics με την οποία ο πελάτης έρχεται σε επικοινωνία με την τράπεζα(ATM) για διάφορες λειτουργίες(επικύρωση,ανάληψη,κατάθεση,ενημέρωση).Αυτή η επικοινωνία περαττώνεται μέσω μιας διεπαφής Operations,και ποιο συγκεκριμένα με τη δημιουργία αντικειμένου αυτής καλώντας την εκάστοτε λειτουργία για υλοποίηση της από τον server(BankServer).

BankServer: Σε αυτό το project έχουμε το κομμάτι της τράπεζας, η οποία λειτουργεί ως server για το ATM και ως client για την βάση δεδομένων. Εδώ υπάρχει η main στην οποία γίνονται όλες οι συνδέσεις και η υλοποίηση των μεθόδων-λειτουργιών μέσω δύο διεπαφών(Operations,Operations2).Συγκεκριμένα, με αντικείμενο τύπου Operations υλοποιούμε την επικοινωνία τράπεζα-ATM και με δημιουργία αντικειμένου Operations2

καλούμε την βάση δεδομένων. Οι διεπαφές Operations των δύο project AtmClient,BankServer περιέχουν τις ίδιες μεθόδους.

DataBase: Στο τελευταίο project υλοποιείται η βάση δεδομένων.Εδώ έχουμε μία κλάση που περιέχει την main στην οποία δημιουργούμε server για να συνδεθεί η τράπεζα και στην συνέχεια ανοίγουμε βάση δημιουργώντας τον πίνακα με τα στοιχεία των λογαριασμών.Όλα αυτά γινονται αφού πρώτα έχουμε συμπεριλάβει στις βιβλιοθήκες μας την SQLite. Η κλάση αυτή περιέχει επίσης την υλοποίηση των μεθόδων-λειτουργιών της διεπαφής Operations2 βάσει των οποίων στέλνονται τα δεδομένα στην τράπεζα αλλά και υλοποιούνται(ή όχι) τα αιτήματα του ATM. Οι διεπαφές Operations2 των BankServer,Database περιέχουν τις ίδιες μεθόδους.

Operations: Στις δύο διεπαφές αυτές αρχικοποιούνται-υλοποιούνται οι εξής μεθόδοι-λειτουργίες:

validCard: Καλούμε την validCard2 που υλοποιείται στην βάση και μας επιστρέφει τον έλεγχο της επικύρωσης.Επίσης κάνουμε set το id στο αντικείμενο Account για να το χρησιμοποιήσουμε στις επόμενες μεθόδους.

deposit: Καλούμε την deposit2 που υλοποιείται στην βάση και μας επιστρέφει την κατάσταση της κατάθεσης(επιτυχία ή μη).

analipsi: : Καλούμε την analipsi2 που υλοποιείται στην βάση και μας επιστρέφει την κατάσταση της ανάληψης(επιτυχία ή μη).

info: : Καλούμε την info2 που υλοποιείται στην βάση και μας επιστρέφει την κατάσταση του λογαριασμού.

Operations2: Στις δύο διεπαφές αυτές αρχικοποιούνται-υλοποιούνται οι εξής μεθόδοι-λειτουργίες:

validCard2: Τρέχουμε στην βάση, εαν υπάρχει ο πελάτης που μας έκανε αίτημα, του επιστρέφουμε τον αριθμό λογαριασμού του.

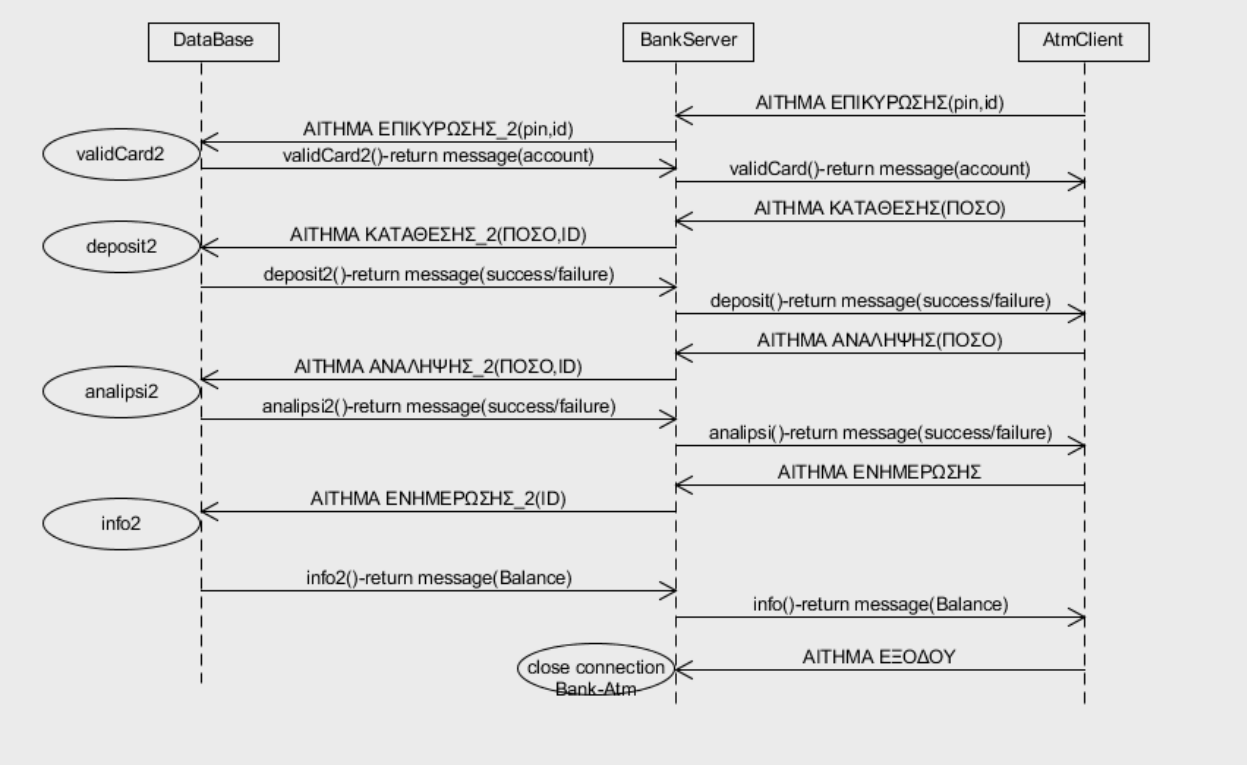
Deposit2: Αφού γίνει η αναζήτηση του λογαριασμού του συγκεκριμένου πελάτη, προσθέτουμε στο υπόλοιπο το ποσό που κατέθεσε ο πελάτης.Επιστρέφουμε ανάλογο μήνυμα.

Analipsi2: Αφού γίνει η αναζήτηση του λογαριασμού του συγκεκριμένου πελάτη, ελέγχουμε εαν μπορεί να γίνει ανάληψη ποσού, αν ναι αφαιρούμε απο το υπόλοιπο το ποσό.Και στις δύο περιπτώσεις επιστρέφουμε το ανάλογο μήνυμα.

Info2 : Αναζητούμε το λογαριασμό του πελάτη και επιστρέφουμε μήνυμα με το υπόλοιπο του λογαριασμού του.

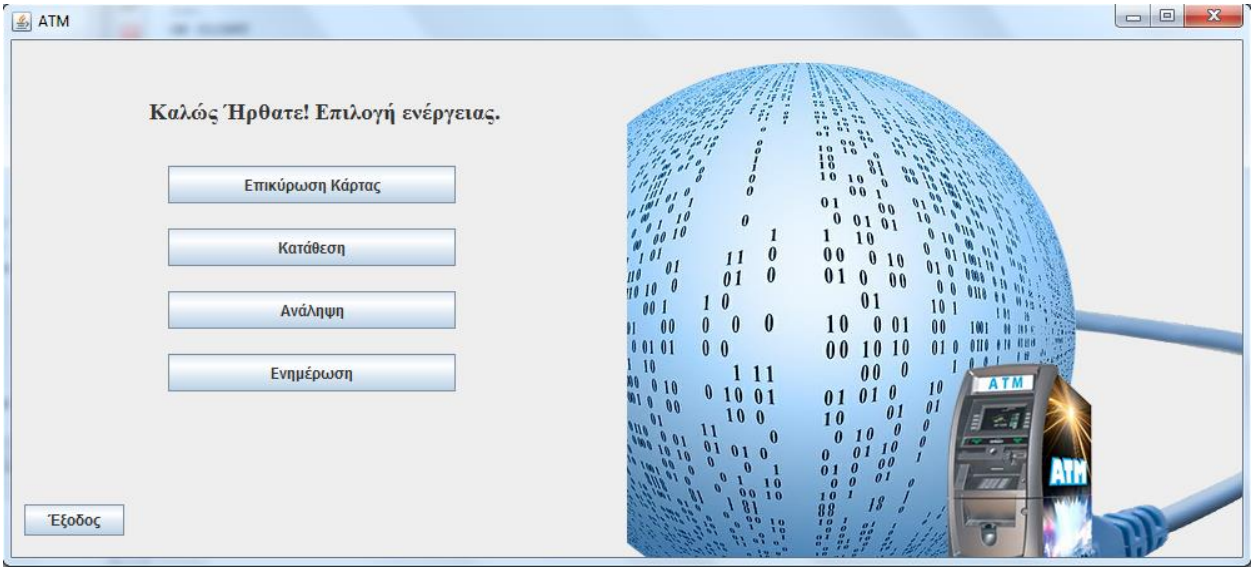
ΔΙΑΓΡΑΜΜΑ UML

Όπου αίτημα επικύρωσης=validCard(),αίτημα επικύρωσης2= validCard2(), κατάθεση=deposit, κατάθεση2=deposit2, ανάληψη=analipsi, ενημέρωση=info (για περισσότερη κατανόηση).

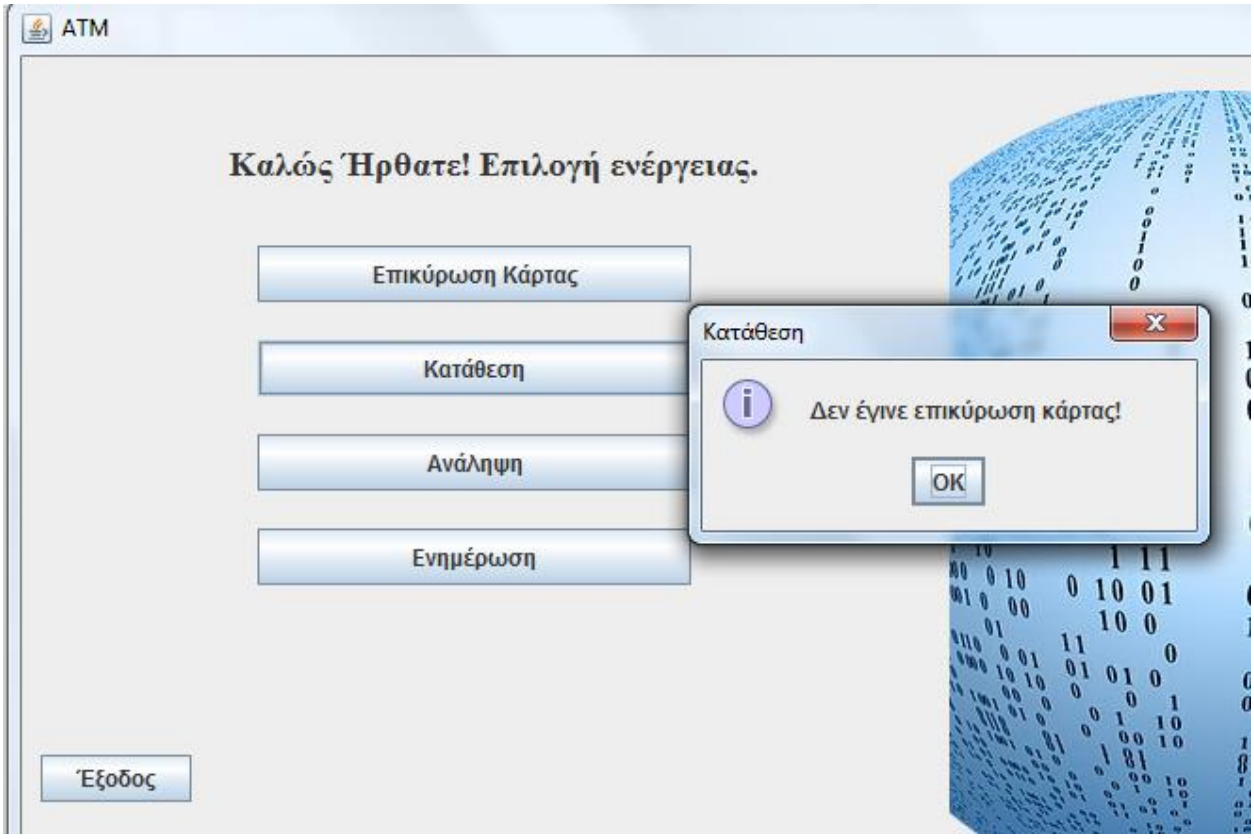


ΟΔΗΓΙΕΣ-SCREENS

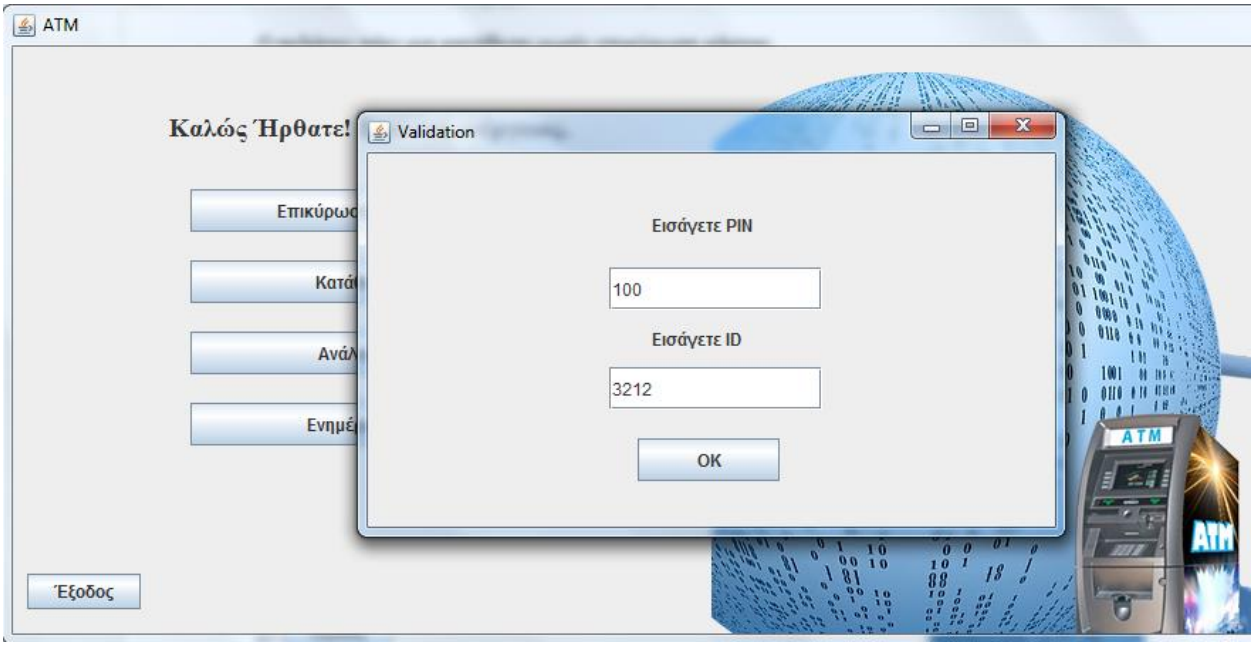
Αρχικό μενού

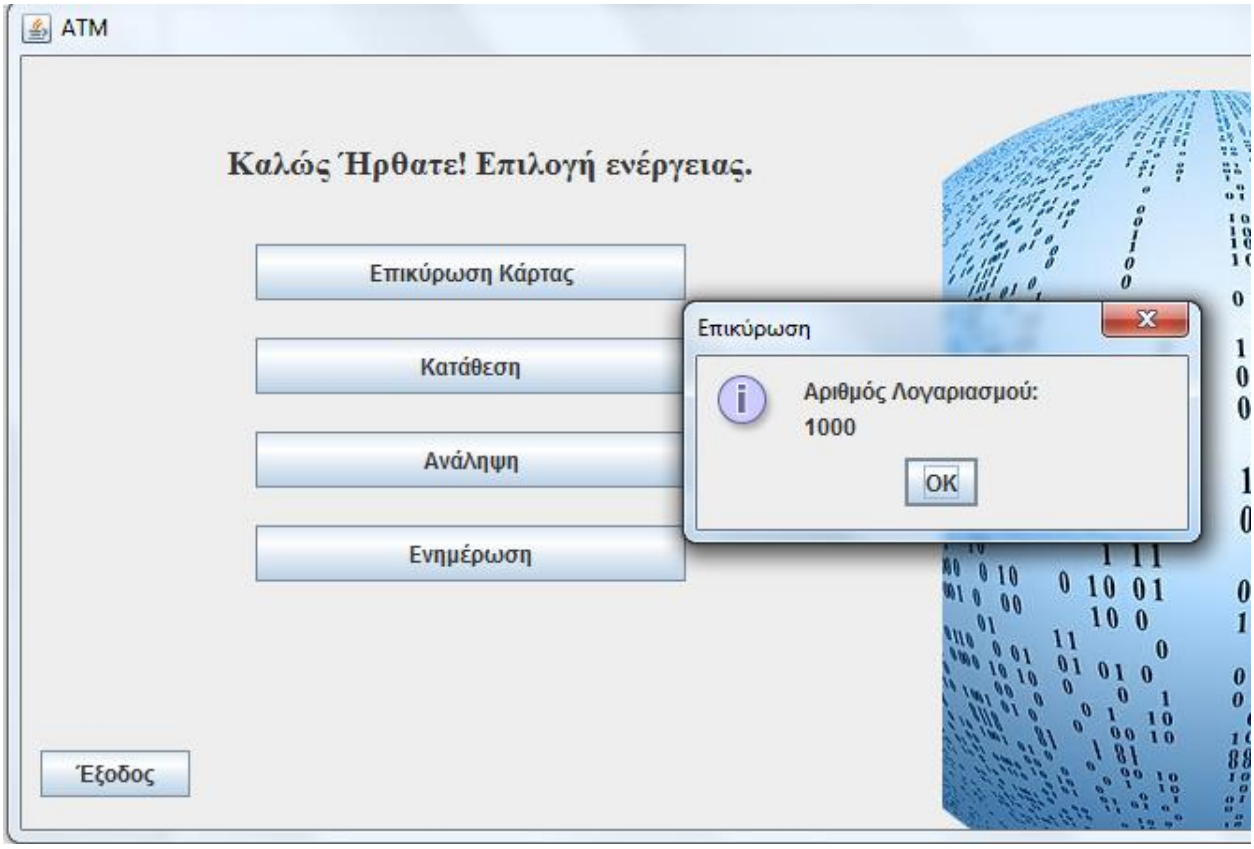


Ο πελάτης πάει για κατάθεση,χωρίς επικύρωση κάρτας.

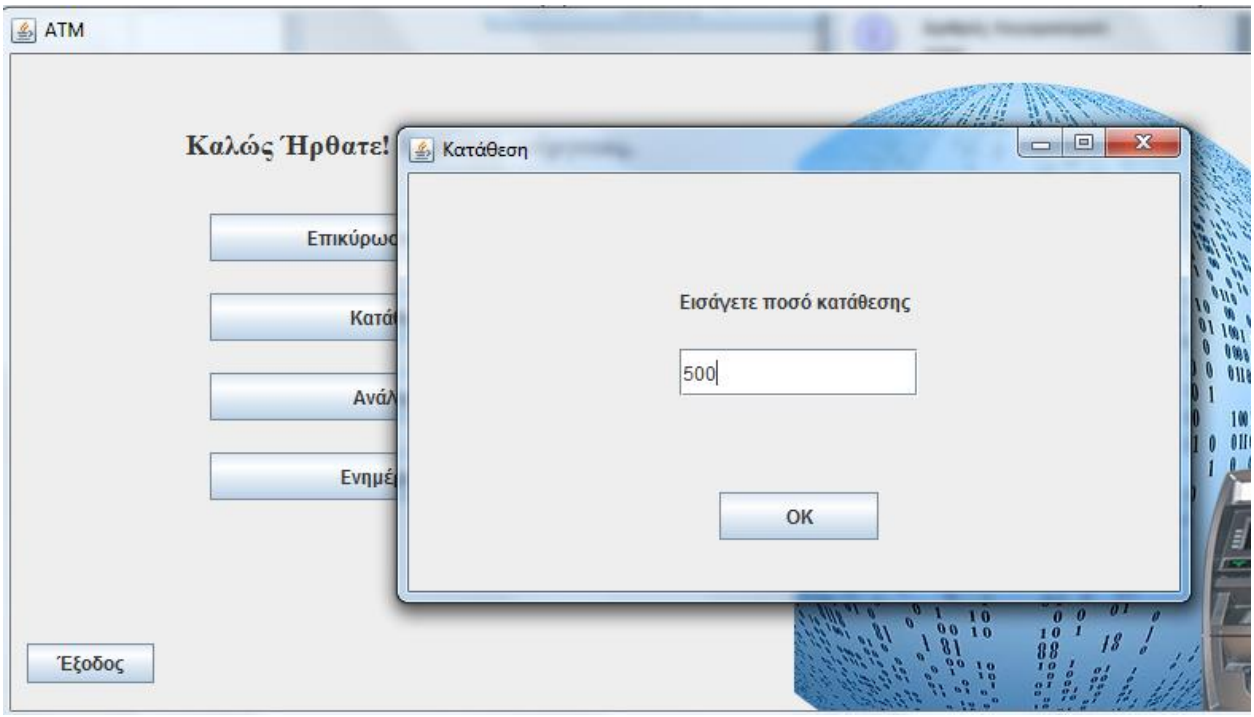


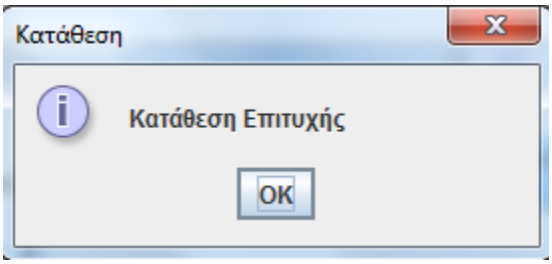
Ο πελάτης κάνει επικύεωση κάρτας



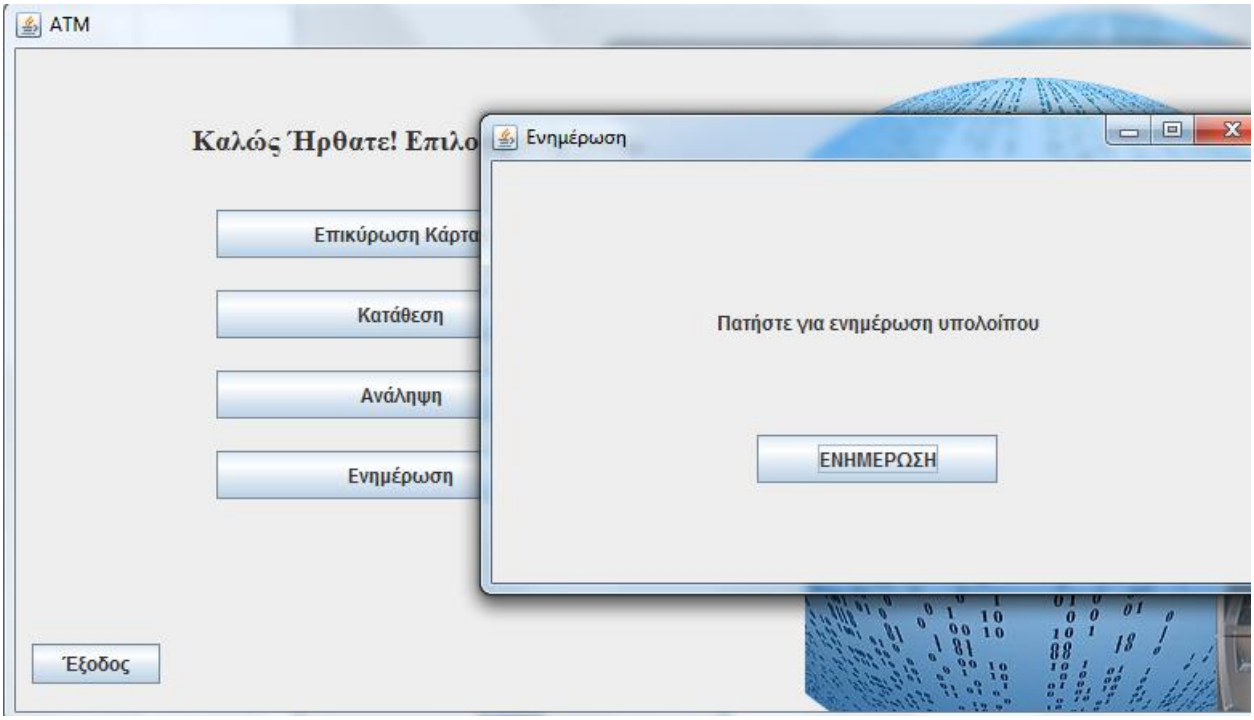


Ο πελάτης καταθέτει 500 ευρώ

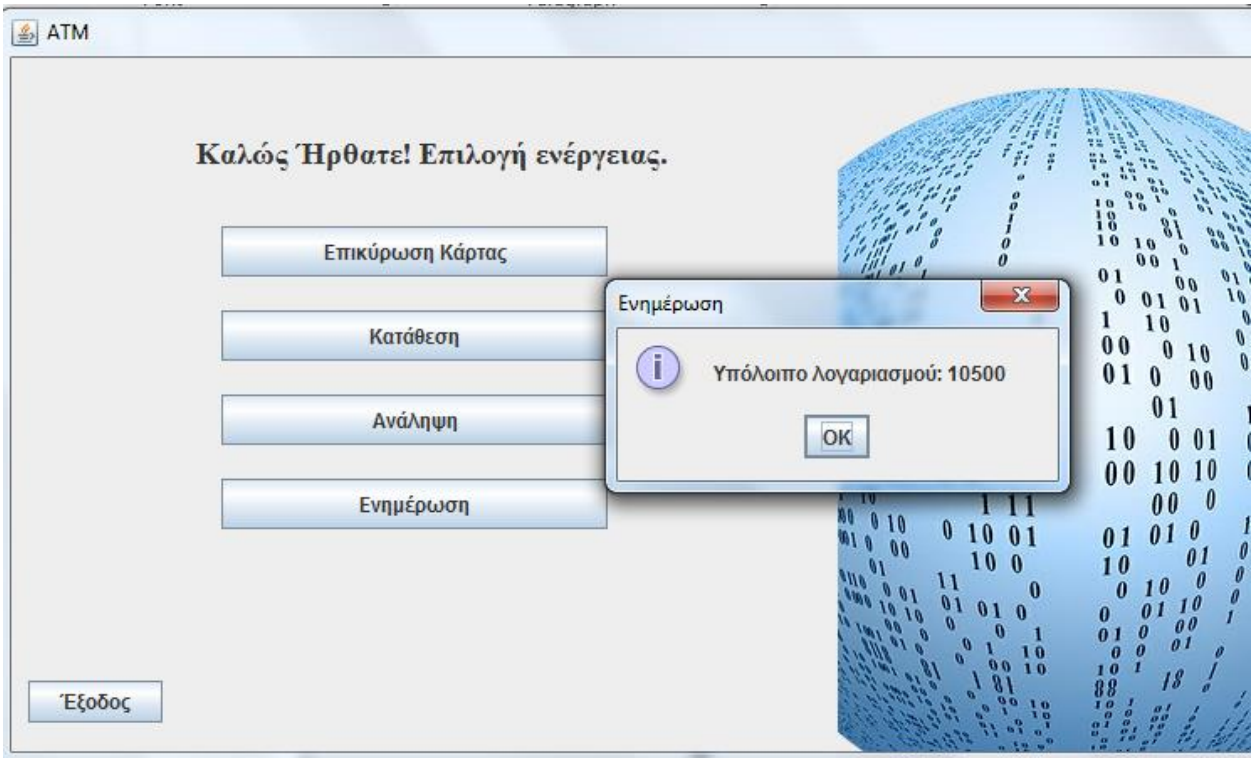




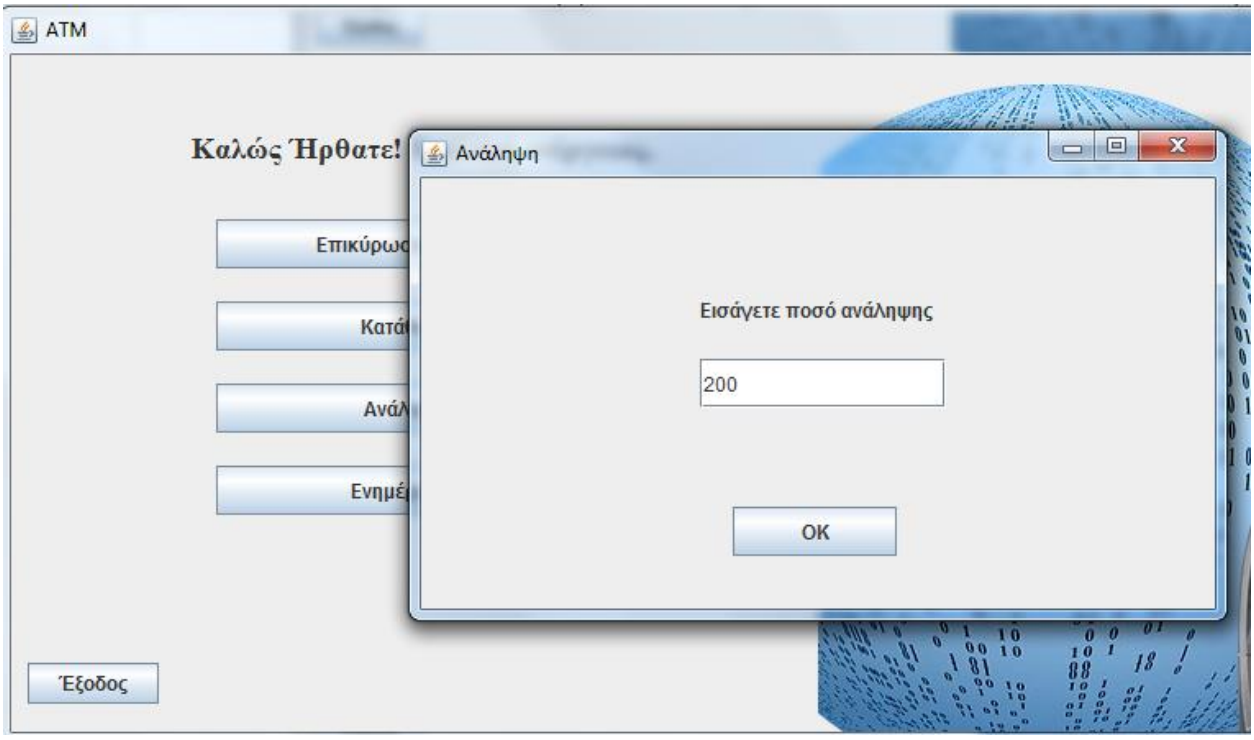
Ο πελάτης κάνει ενημέρωση

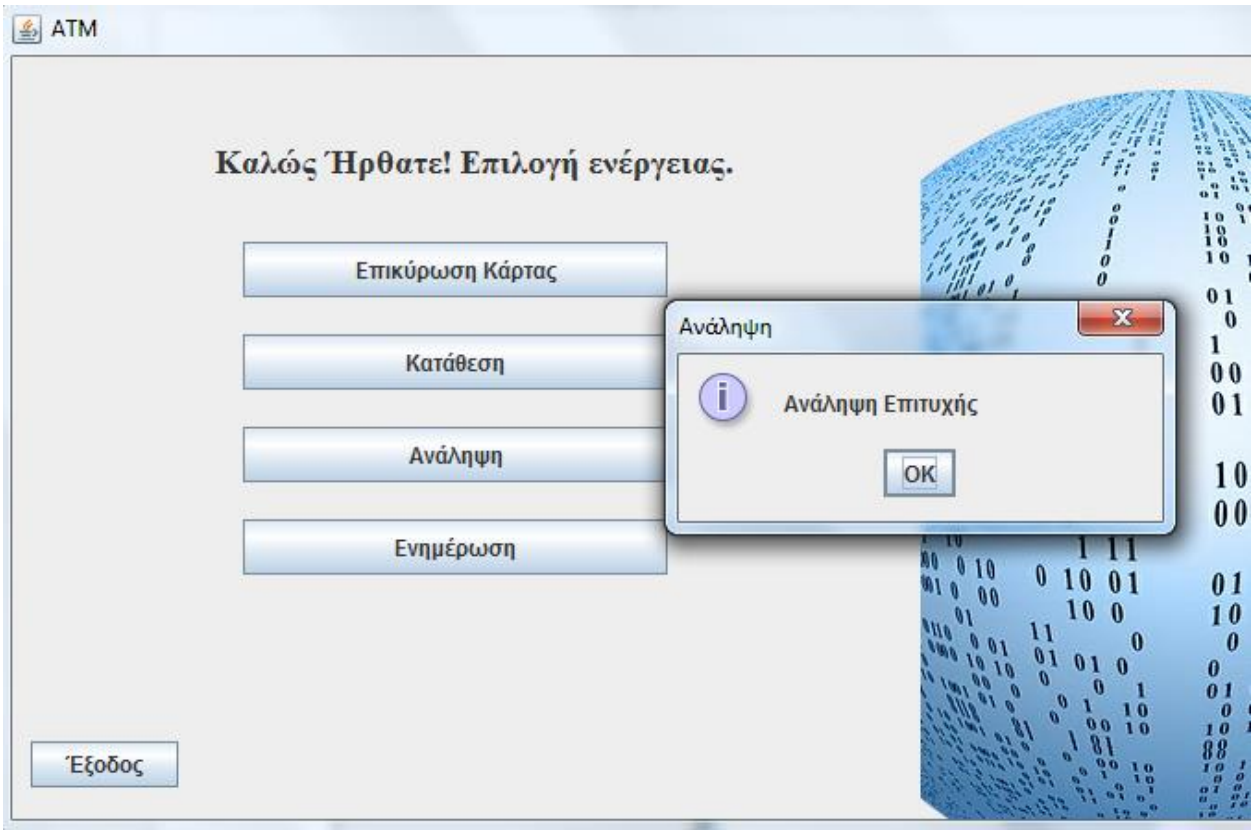


Βλέπουμε ότι όντως προστέθηκαν 500 ευρώ

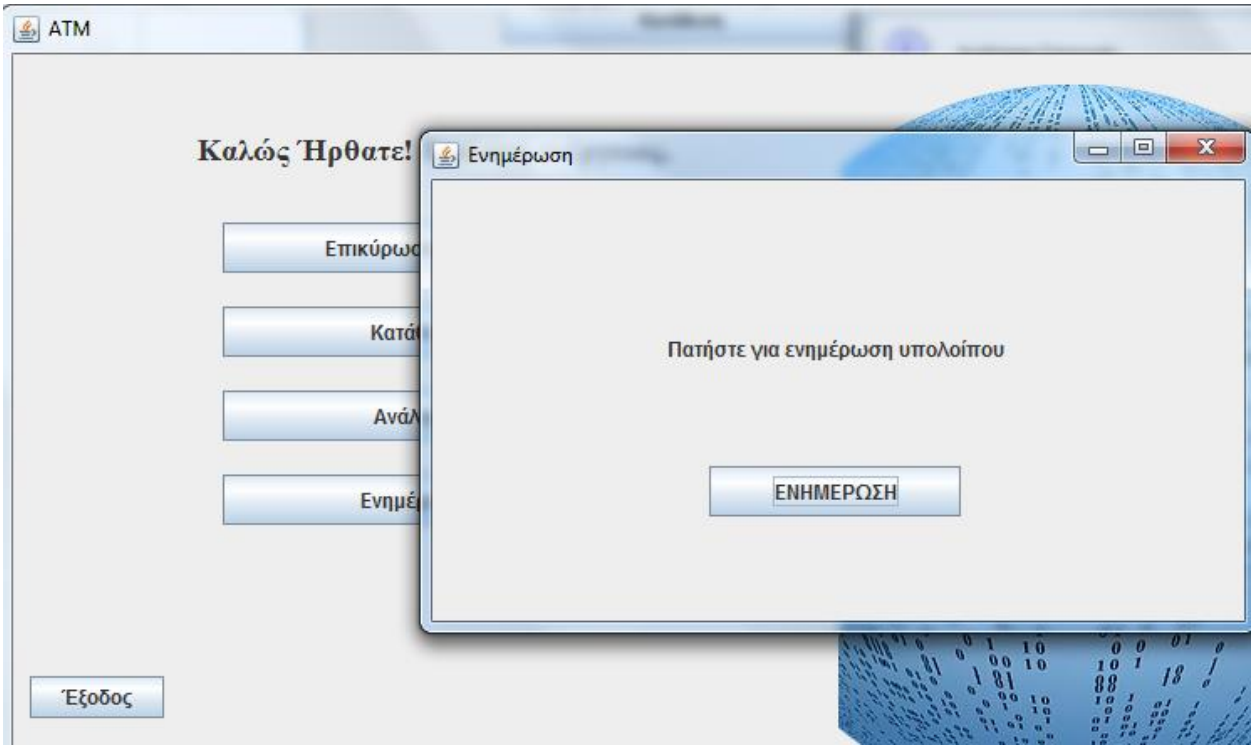


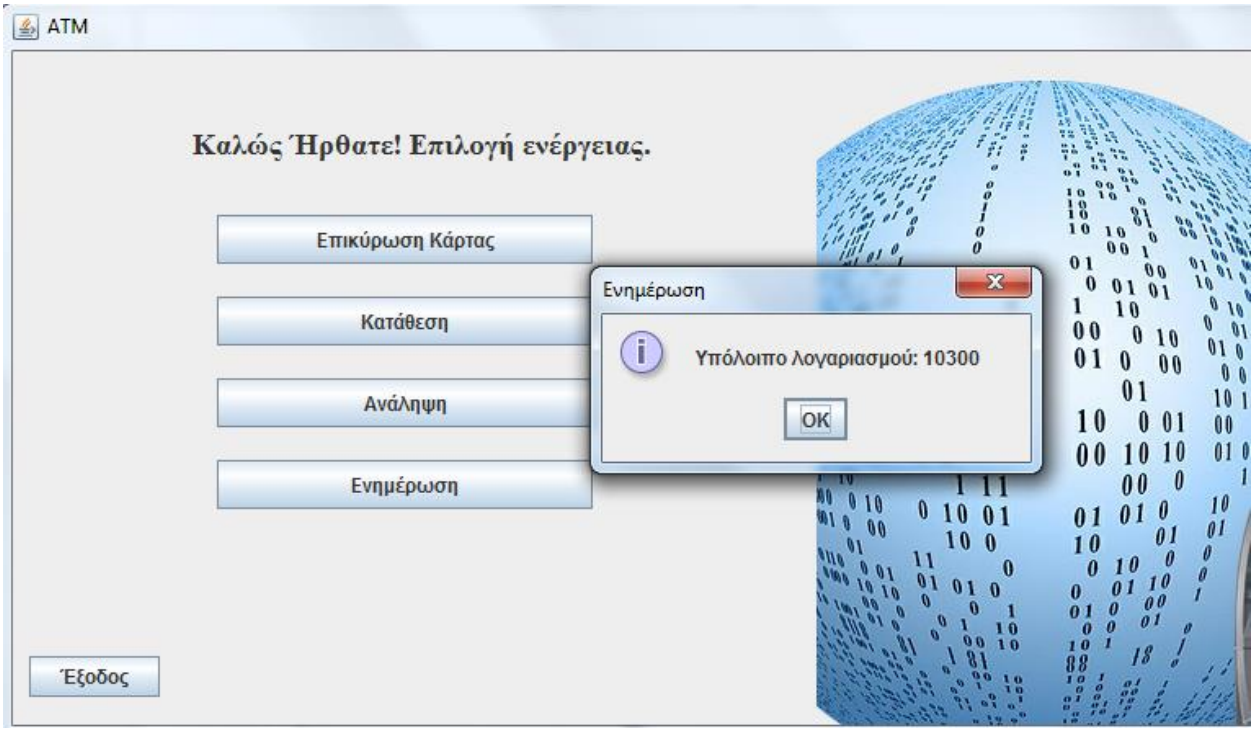
Ο χρήστης κάνει ανάληψη 200 Ευρώ



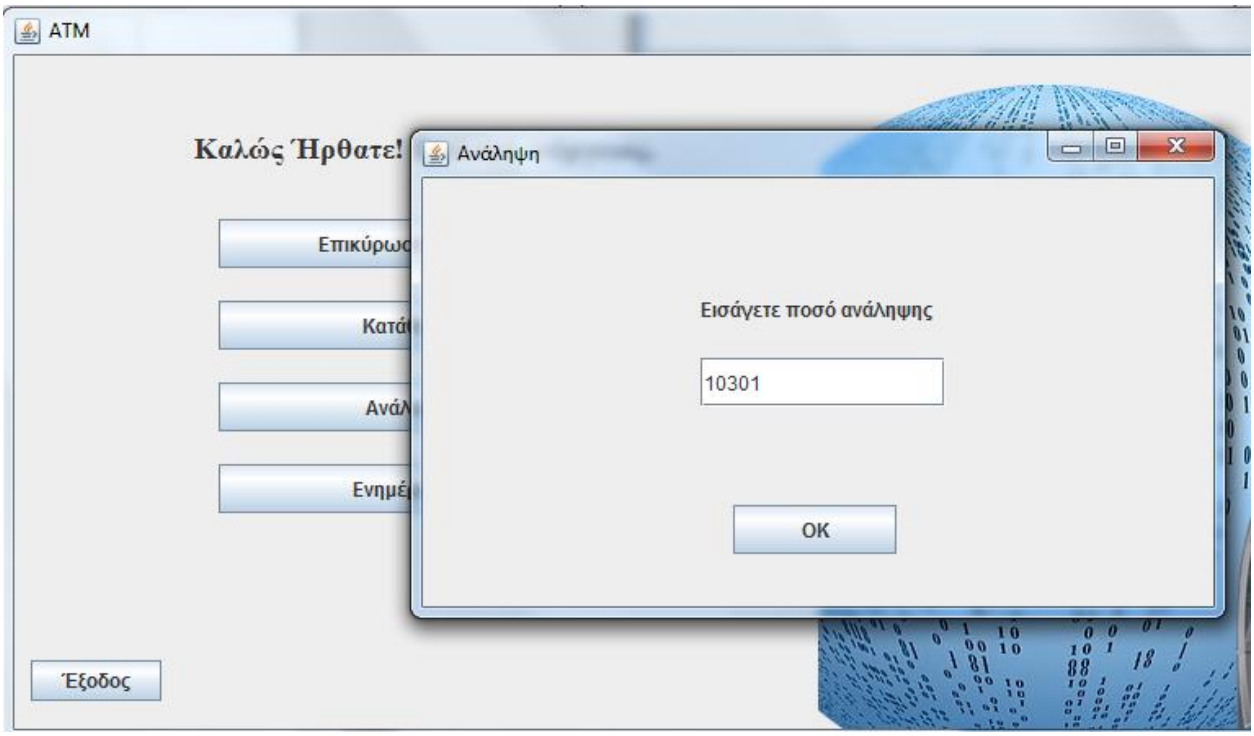


Ο χρήστης ενημερώνεται ξανά

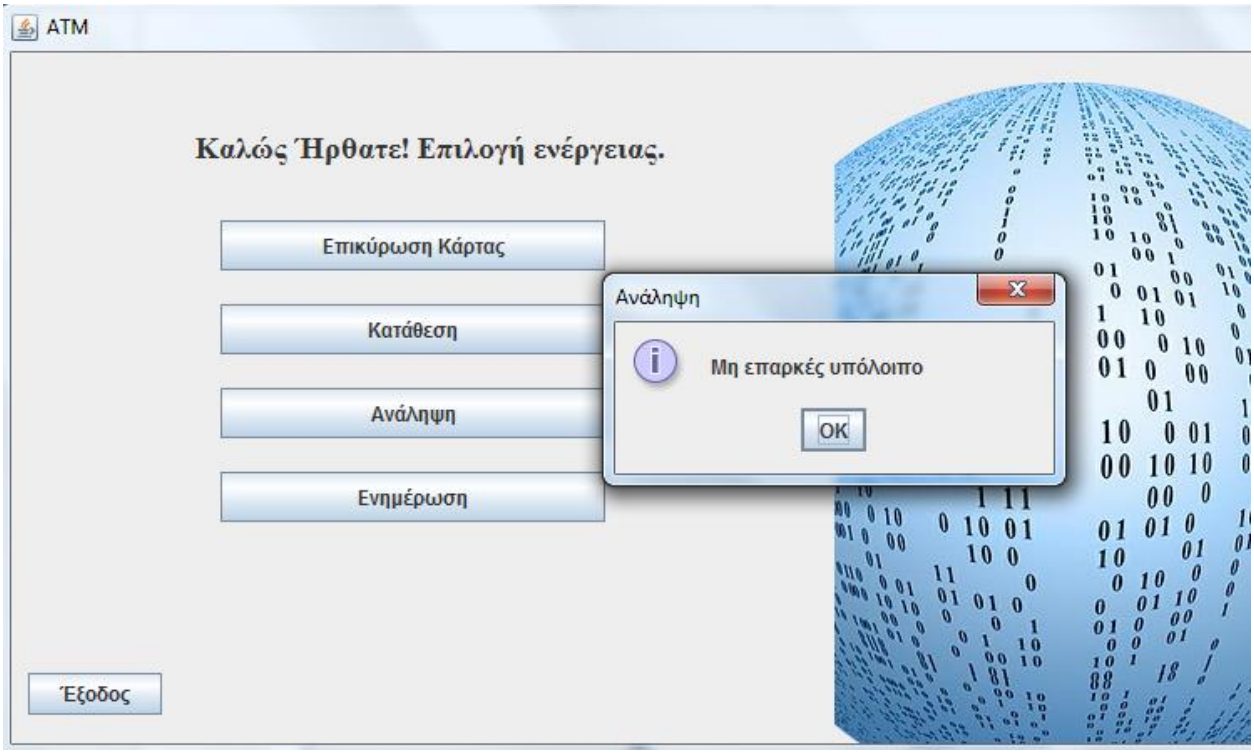




Ο χρήστης κάνει ανάληψη 10301 Ευρώ



Ανάληψη ανεπιτυχής



Ο χρήστης πατάει Έξοδος και κλείνει η σύνδεση.

Β) Συνθήκες ανταγωνισμού (race conditions)

Η σύνδεση στην τράπεζα από πολλούς χρήστες έχει ως αποτέλεσμα να υπάρξουν συνθήκες ανταγωνισμού μεταξύ των λειτουργιών που εκτελούν οι χρήστες.Δηλαδή περιπτώσεις όπου πάνω στον ίδιο λογαριασμό διαφορετικοί χρήστες επιχειρούν διαφορετικές ενέργειες την ίδια στιγμή.
Για παράδειγμα όταν ο client1 κάνει ανάληψη ένα ποσό και ο client2, ενημέρωση απ'τον ίδιο λογαριασμό. Άλλο ένα παράδειγμα είναι όταν κάποιος πραγματοποιεί ανάληψη σε λογαριασμό χωρίς υπόλοιπο και ένας άλλος καταθέτει στον ίδιο λογαριασμό με αποτέλεσμα να μην μπορεί να κάνει ανάληψη ενώ θα μπορούσε μετά από λίγο χρόνο.
Σε τέτοιες περιπτώσεις, η αποτροπή τέτοιων συνθηκών επιτυγχάνεται με τον συγχρονισμό αντικειμένων όπως ένα αντικείμενο Account. Δηλαδή,κάθε φορά που κάποιος χρήστης καλεί κάποια λειτουργία για τον συγκεκριμένο λογαριασμό, θα παίρνει ένα κλειδί(lock). Θα εκτελείται αυτή η μέθοδος-λειτουργία και στη συνέχεια το κλειδί θα είναι διαθέσιμο για το επόμενο αντικείμενο. Γι'αυτό το λόγο αποφασίσαμε να κάνουμε synchronized() τις μεθόδους analipsi2() και info2() που υλοποιούνται στην βάση δεδομένων μας(DataBase).

Γ) Αρχιτεκτονική/Τεχνολογία Υλοποίησης

Χρησιμοποιώντας socket programming πρέπει να ξέρουμε συγκεκριμένα ποιές θύρες θα χρησιμοποιηθούν, πρέπει να ορίσουμε ποιό πρωτόκολλο θα χρησιμοποιηθεί (TCP ή UDP) και μας δίνει τη δυνατότητα να χειριζόμαστε την μορφή των μηνυμάτων μεταξύ του client και του server. Με τα sockets όμως, μπορούμε να στείλουμε μόνο δεδομένα και μόνο σε μια συγκεκριμένη θύρα και τίποτα άλλο.

Από την άλλη πλευρά το RMI (Remote Method Invocation) μπορεί να χρησιμοποιηθεί για τη δημιουργία ενός αντικειμένου, με το οποίο ο client μπορεί να καλέσει διάφορες μεθόδους. Το RMI είναι φτιαγμένο πάνω σε sockets και τα χρησιμοποιεί για τη μεταφορά δεδομένων σε ένα δίκτυο. Επίσης το RMI επιτρέπει stateless συνδέσεις, όπου ο client δεν στέλνει κάποιο μήνυμα επιβεβαίωσης στον server μόλις λάβει κάποιο πακέτο δεδομένων και μπορεί να το λάβει χωρίς να έχουν γίνει κάποιες ρυθμίσεις σύνδεσης εκ των προτέρων.

Αφού αναφέραμε τα παράπανω, παρατηρούμε ότι γενικότερα τα sockets χρησιμοποιούνται συνήθως σε "τοπικές" συνδέσεις. Το RMI όμως φαίνεται πως είναι κατάλληλο για τη δικιά μας υλοποίηση, αφού είναι σχετικά εύκολο στη χρήση, δεν χρειάζεται κάποια προετοιμασία/εγκατάσταση από τη μεριά του client και το βασικότερο είναι ότι μπορούμε να καλούμε μεθόδους του server για την εκτέλεση των λειτουργιών που επιθυμούμε κάθε φορά.

Η εφαρμογή μας χρησιμοποιεί αρχιτεκτονική 3-επιπέδων η οποία αποτελείται από τρία λογικά επίπεδα, τα οποία υλοποιούνται σε διαφορετικές μηχανές. Το υψηλότερο επίπεδο περιλαμβάνει τη διεπαφή του πελάτη (επίπεδο παρουσίασης). Το ενδιάμεσο επίπεδο περιέχει την επιχειρησιακή λογική (επίπεδο εφαρμογής) ενώ το κατώτερο επίπεδο διαχειρίζεται τα δεδομένα που χρησιμοποιούνται από την εφαρμογή (επίπεδο δεδομένων). Όπως φαίνεται παραπάνω, στην αρχιτεκτονική 3-επιπέδων ο καθένας έχει ένα συγκεκριμένο ρόλο, πράγμα το οποίο δεν γίνεται στο μοντέλο 2-επιπέδων (client/server) όπου δεν είναι πάντα σαφής η διάκριση των δύο ρόλων. Κάποια από τα βασικά πλεονεκτήματα της αρχιτεκτονικής 3-επιπέδων είναι ότι δίνει τη δυνατότητα εύκολης εισαγωγής νέων τεχνολογιών Βάσεων Δεδομένων, και η ενίσχυση της ασφάλειας των δεδομένων αφού η προσπέλασή τους γίνεται μόνο μέσω του ενδιάμεσου επιπέδου. Επίσης η ανάπτυξη του μεσαίου επιπέδου με αντικειμενοστρεφής τεχνολογίες ενισχύουν τη δυνατότητα επαναχρησιμοποίησης τμημάτων του λογισμικού. Τέλος, οι διαφορετικοί εξυπηρετητές μπορούν να βελτιστοποιηθούν ανεξάρτητα και ο καθένας με βάση τη λειτουργία του.

Δ) Ασφάλεια

Στις μέρες μας υπάρχουν πάρα πολλών ειδών απειλές για ένα τραπεζικό σύστημα οι οποιές μπορούν να εξαλειφθούν χρησιμοποιώντας κατάλληλους μηχανισμούς για την προστασία των συναλλαγών οι οποίες είναι πάντα ευάλωτες σε επιθέσεις από επίδοξους “χάκερς”. Υπάρχουν πάρα πολλοί περίπλοκοι μηχανισμοί που μπορούν να χρησιμοποιηθούν για την προστασία ενός τραπεζικού συστήματος κατά την αλληλεπίδραση πελάτη-εξυπηρετητή. Παρακάτω αναφέρονται οι πιο αποτελεσματικοί από αυτούς.

Κάποια τραπεζικά συστήματα χρησιμοποιούν μηχανισμούς όπως οι δυναμικοί κωδικοί πρόσβασης. Πρόκειται για μία όχι τόσο δαπανηρή μέθοδο, η οποία παρέχει έναν δεύτερο παράγοντα για την επαλήθευση της ταυτότητας του χρήστη. Η ιδέα πίσω από τη μέθοδο αυτή είναι να χρησιμοποιείται διαφορετικός κωδικός για κάθε συναλλαγή μέσω OPT καρτών (one-time password cards). Με την μέθοδο αυτή ακόμα και αν γίνει κλοπή των στοιχείων αυθεντικοποίησης του χρήστη, τα στοιχεία θα είναι άχρηστα για κάποια επίθεση αφού αυτά θα έχουν αλλάξει μετά την ολοκλήρωση της συναλλαγής του χρήστη.

Η βιομετρική είναι μια από τις υποψήφιες και πολλά υποσχόμενες τεχνολογίες για την βελτίωση της προστασίας των τραπεζικών συστημάτων. Πρόκειται για μια αυτοματοποιημένη μέθοδο για την αυθεντικοποίηση των πελατών μέσω των βιολογικών χαρακτηριστικών τους, όπως τα δακτυλικά αποτυπώματα, ο αμφιβληστροειδής και η αναγνώρηση φωνής. Όλα τα παραπάνω χαρακτηριστικά είναι μοναδικά για τον κάθε άνθρωπο και εξαιρετικά δύσκολο να

πλαστογραφηθούν, γι’αυτό και βιομετρικές τεχνολογίες χρησιμοποιούνται ήδη σε διάφορους τομείς όπου η ασφάλεια έχει πρωταρχικό ρόλο.