Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
τομέας επικοινωνιών ηλεκτρονικής και συστημάτων
πληροφορικής

Ανάπτυξη Υπηρεσιών με βάση την θέση του χρήστη, με το λογισμικό Android

Διπλωματική Εργασία

Γεώργιος Πάνου Παπαδημητρίου

Επιβλέπων: Νικόλαος Μήτρου
Καθηγητής Ε.Μ.Π.

Αθήνα, Φεβρουάριος 2010

(Η σελίδα αυτή είναι σκόπιμα λευκή)

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
τομέας επικοινωνιών ηλεκτρονικής και συστημάτων
πληροφορικής

Ανάπτυξη Υπηρεσιών με βάση την θέση του χρήστη, με το λογισμικό Android

Διπλωματική Εργασία

Γεώργιος Πάνου Παπαδημητρίου

Επιβλέπων: Νικόλαος Μήτρου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή επιτροπή την 2 Φεβρουαρίου 2010.

.....	.....	.....
Νικόλας Μήτρου	Μιχαήλ Θεολόγου	Συμεών Παπαβασιελείου
Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π.	Επίκουρος Καθηγητής

Αθήνα, Φεβρουάριος 2010

(Υπογραφή)

.....

Γεώργιος Πάνου Παπαδημητρίου

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Ο σκοπός της διπλωματικής εργασίας ήταν η ανάπτυξη μιας εφαρμογής που προορίζεται για κινητά τηλέφωνα που χρησιμοποιούν το λογισμικό Android της Google, με σκοπό την παροχή υπηρεσιών απευθείας συνομιλίας με άλλους χρήστες, συνομιλία όλων των χρηστών μαζί σε ένα «παράθυρο»-chat και ανταλλαγής αρχείων μεταξύ τους. Το σύστημα που αναπτύχθηκε περιλαμβάνει την εφαρμογή στο κινητό τηλέφωνο του χρήστη και την εφαρμογή στον εξυπηρετητή-Server στον οποίο είναι καταχωρημένοι οι χρήστες και τα στοιχεία που αφορούν τις διαθέσιμες υπηρεσίες. Ο σχεδιασμός του συστήματος έχει γίνει με τρόπο που επιτρέπει την αυτόματη εύρεση των διαθέσιμων υπηρεσιών είτε με κεντρικοποιημένο η με αποκεντρικοποιημένο τρόπο. Ο χρήστης έχει την επιλογή εφόσον ο κεντρικός εξυπηρετητής δεν είναι διαθέσιμος, να αναζητεί μόνος του στο τοπικό δίκτυο άλλους χρήστες και να συνομιλεί μαζί τους είτε ένας προς έναν είτε όλοι μαζί σε ένα κοινό παράθυρο. Υπάρχει η δυνατότητα ο χρήστης να βλέπει που βρίσκονται οι άλλοι χρήστες.

Λέξεις κλειδιά: χρήστες, κινητά τηλέφωνα, υπηρεσίες σε τοπικά δίκτυα, μοντέλο client-server, Android-java, Gps-εντοπισμός στίγματος.

(Η σελίδα αυτή είναι σκόπιμα λευκή)

Abstract

The scope of this thesis was the development of an application for mobile phones which use the Android software of Google, providing services such as contact with other users, chatting with multiple users in a chat-room and exchanging files between the users. The system that was developed consists of two components, the client application and the server application in which the users are registered and choose a service to log in. The design of the system is developed has been a way that it is able to find automatically the available services either in a centralized way or in non-centralized one. If the server is temporary out of service, the user can search in his local network for other users who are using the application so as to talk to them either, one to one or to all of them in a global chat room. It is possible, the user see where the other users are on a map.

Keywords: users, mobile phones, services at local networks, client-server model, Android-java, Gps location

(Η σελίδα αυτή είναι σκόπιμα λευκή)

Πίνακας Περιεχομένων

Εισαγωγή

1 Υπηρεσίες με βάση τη θέση του Χρήστη-Location Based Services

1.1 Επισκόπηση

Το διαδίκτυο και το κινητό τηλέφωνο έγιναν η αιτία μιας ασυνήθιστης «εισβολής» στην αγορά των φορητών συσκευών. Η ένωση αυτών των δύο τεχνολογιών υπόσχεται να φέρει στο κοινό περισσότερες υπηρεσίες. Στην ανερχόμενη εμπορική οικονομία των κινητών τηλεφώνων, η γνώση της θέσης ενός χρήστη αποκτά όλο και περισσότερο ενδιαφέρον για τους παρόχους κινητής τηλεφωνίας που μπορούν με τη σειρά τους να παρέχουν πρωτοποριακές υπηρεσίες βασιζόμενες στη θέση αυτή. Τέτοιες ιδέες δεν είναι κάτι καινούργιο καθώς υπηρεσίες εντοπισμού θέσης ενός οχήματος υπήρχαν από τον 1980. Ωστόσο, μέχρι πρόσφατα, η τελειοποίηση των υπηρεσιών με βάση τη θέση έδωσε έναυσμα στην αγορά τόσο για πελάτες που είναι εταιρείες όσο και σε πελάτες που είναι απλοί χρήστες.

Η πλήρης εμπορευματοποίηση των LBS έγινε δυνατή τα τελευταία χρόνια. Συγκεκριμένα, πρόσφατα τεχνολογικα επιτεύγματα άλλαξαν το τοπίο και η βιομηχανία άρχισε να επενδύει σε αυτό το περιβάλλον. Αποτέλεσμα αυτού, ήταν να ανοιχτεί η πόρτα σε μια πληθώρα εμπορικών υπηρεσιών και εφαρμογών συμπεριλαμβανομένων και αυτών για έκτακτες ανάγκες, πλοήγηση, ενημέρωση χρέωσης με βάση τη τοποθεσία καθώς και πληροφορίες που αφορούν την περιοχή.

1.2 Σημαντικά Χαρακτηριστικά

Στις υπηρεσίες αυτές ο χρήστης αναλόγως του που βρίσκεται, μπορεί να λάβει συγκεκριμένες υπηρεσίες ή δεδομένα. Σημαντικό συστατικό στοιχείο αυτών των υπηρεσιών είναι η δυνατότητα εντοπισμού της θέσης του χρήστη και η ενημέρωση του με βάση μόνο τις πληροφορίες που έχουν ενδιαφέρον για την περιοχή στην οποία βρίσκεται. Οι πληροφορίες αυτές ή τα δεδομένα μπορούν να προέρχονται από κάποιο κεντρικό εξυπηρετητή-Server. Προκειμένου αυτές οι πληροφορίες-δεδομένα να είναι προσεγγίσιμα θα πρέπει ο χρήστης να είναι συνδεδεμένος στο διαδίκτυο μέσω του δικτύου του παρόχου κινητών τηλεπικοινωνιών ή μέσω των διαθέσιμων ασύρματων

υποδομών που υπάρχουν στο περιβάλλον του. Σημαντικό χαρακτηριστικό των υπηρεσιών αυτών είναι ότι οι χρήστες δεν χρειάζεται να υποβάλλουν σε κάποια φόρμα στοιχείων, ταχυδρομικούς κώδικες, διευθύνσεις ή άλλα αναγνωριστικά θέσης όταν περιπλανώνται κατά τη διάρκεια της χρήσης των υπηρεσιών αυτών, καθώς το σύστημα αυτόματα αντιλαμβάνεται που βρίσκεται ο χρήστης με μεθόδους που θα αναπτύξουμε στη συνέχεια.

Προκειμένου να δοθεί μια επιτυχημένη τεχνολογία LBS οι ακόλουθοι παράγοντες πρέπει να πληρούνται όσον αφορά τον εξοπλισμό για τον εντοπισμό θέσης:

- Συντεταγμένες ακρίβειας που καθορίζονται από την αρμόδια υπηρεσία
- Όσο το δυνατό χαμηλότερο κόστος
- Ελάχιστες επιπτώσεις στο δίκτυο και τον εξοπλισμό.

Οι απαιτήσεις που πρέπει να εκπληρώνονται για να είναι δυνατή μια εφαρμογή LBS και εκμεταλλεύσιμη από το κοινό είναι οι εξής:

- Απλό γραφικό περιβάλλον για το χρήστη
- Ελάχιστη χρήση του δικτύου του παρόχου για απόκτηση πληροφοριών
- Το τερματικό να έχει πλήρη ανεξαρτησία
- Χρήση διαθέσιμων διαδικτυακών υπηρεσιών
- Υψηλή διαθεσιμότητα των υπηρεσιών ακόμη και σε περιόδους υψηλής ζήτησης
- Εξελιξιμότητα
- Χαμηλό κόστος

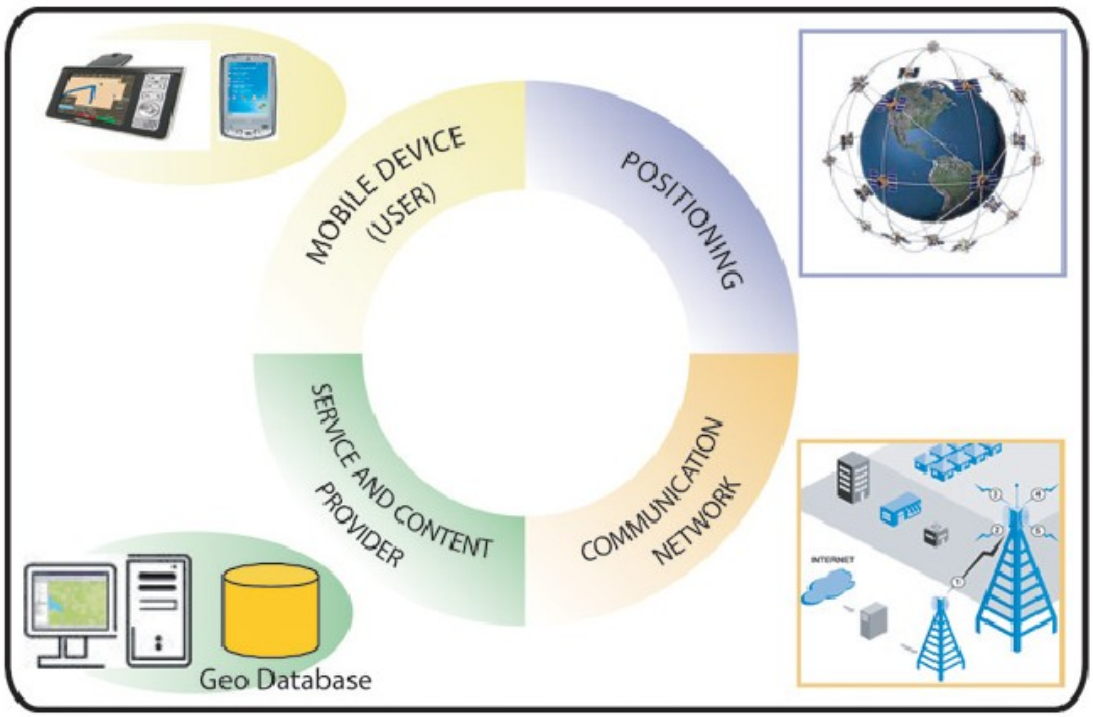
Τέλος σε περιπτώσεις που οι LBS αναφέρονται σε περιοχές εντός κτιρίων (indoors) τότε πρέπει να καλύπτονται και τα παρακάτω κριτήρια:

- Εύκολη καθοδήγηση μέσω του φυλλομετρητή (browser)
- Ελάχιστη γραφική αλληλεπίδραση
- Υποστήριξη προσανατολισμού στο χώρο και ελάχιστη χρήση εντολών

1.3 Δομικά μέρη των Υπηρεσιών με βάση τη θέση του χρήστη (LBS)

Τα δομικά μέρη που απαρτίζουν ένα πλήρες σύστημα βασιζόμενο στη θέση του χρήστη είναι πέντε (4+1) και η σύνδεση μεταξύ τους είναι:

- Κινητό τερματικό-συσκευή: Ένα εργαλείο-συσκευή όπου οι χρήστες μπορούν να ζητήσουν τις πληροφορίες που επιθυμούν. Οι πληροφορίες αυτές δίνονται στο χρήστη με τη μορφή λόγου, εικόνων κειμένου.
- Δίκτυο επικοινωνίας: Το δεύτερο συστατικό στοιχείο είναι ένα δίκτυο κινητής τηλεφωνίας το οποίο χρησιμοποιείται για να μεταφερθούν οι αιτήσεις του χρήστη από το κινητό τερματικό στον εξυπηρετητή και στη συνέχεια τα δεδομένα από αυτόν στο τερματικό.
- Σύστημα εύρεση θέσης: Για να είναι δυνατή η παροχή πληροφοριών που έχουν σχέση με τη θέση του χρήστη θα πρέπει να υπάρχει τρόπος να βρίσκουμε τη θέση του τερματικού εύκολα. Αυτό είναι δυνατό με τη χρήση συστημάτων, όπως το GPS. Άλλος τρόπος είναι η δυνατότητα εύρεσης θέσης μέσω του ασύρματου δικτύου που παρέχει πληροφορίες θέσης.
- Πάροχος υπηρεσιών: Ο πάροχος αυτός παρέχει μια πληθώρα υπηρεσιών στο χρήστη και είναι υπεύθυνος για την επεξεργασία αυτών των αιτήσεων. Τέτοιες υπηρεσίες απαιτούν την δυνατότητα υπολογισμού της θέσης, εύρεσης μια διαδρομής κτλ.
- Πάροχοι δεδομένων: Συνήθως ο πάροχος υπηρεσιών δε μπορεί να διατηρεί στις βάσεις δεδομένων του όλα τα δεδομένα που μπορεί ο κάθε χρήστης να αναζητήσει. Συνεπώς θα πρέπει να υπάρχουν πάροχοι που έχουν τη δυνατότητα να κρατούν χρήσιμες πληροφορίες και δεδομένα και ο πάροχος υπηρεσιών να έχει πρόσβαση σε αυτούς και να παραλαμβάνει κάθε φορά τα δεδομένα που χρειάζεται προκειμένου να εξυπηρετήσει τις αιτήσεις του χρήστη.



1.4 Τεχνολογίες Εντοπισμού Θέσης

Οι τεχνολογίες εντοπισμού θέσης πάνω στις οποίες βασίζονται οι υπηρεσίες αυτές είναι νέες για το ευρύ κοινό, ακόμα και για τις εταιρείες, αλλά παρόλο αυτά είναι πάρα πολύ αξιόπιστες και σε σύντομο χρονικά διάστημα έχουν αναπτυχθεί-προχωρήσει πάρα πολύ και έχουν υψηλή αξιοπιστία. Στη συνέχεια περιγράφονται οι βασικοί τρόποι εντοπισμού θέσης που χρησιμοποιούνται ευρέος στην αγορά:

1.4.1 Εντοπισμός θέσης μέσω GPS

Μία μέθοδος εντοπισμού του εκάστοτε χρήστη είναι το GPS. Αυτή η μέθοδος δίνει τη θέση του χρήστη με μεγάλη ακρίβεια, ωστόσο μπορεί να είναι ακριβή σε κόστος για τον χρήστη καθώς θα πρέπει να επενδύσει σε έναν χειροκίνητο GPS-εξοπλισμό. Το GPS αποτελείται από τρία τμήματα: από 24-32 δορυφόρους που βρίσκονται σε τροχιά γύρω από την Γη, από τέσσερις επίγειους σταθμούς που ελέγχουν και συντονίζουν τους δορυφόρους και από τους δέκτες GPS των χρηστών. Οι δορυφόροι GPS εκπέμπουν σήματα από το διάστημα και χρησιμοποιούνται από το τους δέκτες των χρηστών και παρέχουν τρεις διαστάσεις και το χρόνο.

Οι δέκτες GPS των χρηστών υπολογίζουν τη θέση του χρήστη από την ακρίβεια στο χρόνο των σημάτων που στέλνονται από τους δορυφόρους του συστήματος GPS. Κάθε τέτοιος δορυφόρος μεταδίδει συνεχώς σήματα-μηνύματα που περιλαμβάνουν:

- τη χρονική στιγμή που μεταδίδεται το σήμα
- ακριβείς πληροφορίες τροχιάς
- το γενικό σύστημα "υγείας" και σύστημα τροχιών των δορυφόρων στο σύνολό τους.

Ο δέκτης μέτρα το χρόνο διέλευσης του κάθε μηνύματος και υπολογίζει την απόσταση από κάθε δορυφόρο. Γεωμετρική trilateration χρησιμοποιείται για να συνδυάσει αυτές τις αποστάσεις από τις θέσεις των δορυφόρων ώστε το σύστημα να καθορίσει την θέση του δέκτη. Η θέση αυτή εμφανίζεται στη συνέχεια σε μια οθόνη εμφανίζοντας το χάρτη της περιοχής ή μόνο το γεωγραφικό πλάτος και μήκος. Πληροφορίες υψόμετρου μπορούν να συμπεριληφθούν. Πολλές συσκευές GPS δείχνουν επίσης επιπλέον πληροφορίες, όπως η κατεύθυνση και η ταχύτητα, η οποία υπολογίζονται από τις αλλαγές θέσης του κινητού τερματικού που περιέχει το δέκτη.

Τρεις δορυφόρους μπορεί να είναι επαρκείς για να βρεθεί η θέση, επειδή ο διαθέσιμος χώρος έχει τρεις διαστάσεις. Ωστόσο, ακόμη και ένα πολύ μικρό λάθος στο ρολόι, επηρεαζόμενο από την πολύ μεγάλη ταχύτητα του φωτός, που είναι ίση με τη ταχύτητα με την οποία τα δορυφορικά σήματα-διαδίδονται, έχει ως αποτέλεσμα ένα μεγάλο σφάλμα θέσης. Ως εκ τούτου οι δέκτες χρησιμοποιούν τέσσερις ή περισσότερους δορυφόρους για την επίλυση της τοποθεσία και του χρόνου στο δέκτη.

1.4.2 Εντοπισμός θέσης μέσω GSM

Η εύρεση της θέσης ενός κινητού τηλεφώνου αναλόγως σε ποια κυψέλη βρίσκεται είναι ένας άλλος τρόπος για να βρεθεί η τοποθεσία ενός αντικειμένου ή ενός προσώπου. Στην περίπτωση της χρήσης του εξοπλισμού του παρόχου κινητής τηλεφωνίας χρησιμοποιείται η εξής τεχνική. Το κινητό τηλέφωνο κατά τη διάρκεια της λειτουργίας του επικοινωνεί ανά τακτά χρονικά διαστήματα με την πιο κοντινή κεραία εκπομπής, είτε βρίσκεται σε εξέλιξη μία κλήση είτε όχι. Συνεπώς η τεχνική αυτή, GSM Localization, στηρίζεται στην τεχνική multilateration.

Η τεχνική multilateration βασίζεται στην ακρίβεια υπολογισμού της time difference of arrival (TDOA) ενός σήματος που εκπέμπεται από το κινητό προς τρεις κεραίες εκπομπής του παρόχου. Ο συνδυασμός των χρόνων αυτών με τη βοήθεια του

triangulation μπορεί να προσδώσει στους παρόχους κινητής τηλεφωνίας την θέση του κινητού τηλεφώνου και συνεπώς του χρήστη με μια μικρή πιθανότητα λάθους.

Κάθε σταθμός βάσης αποτελείται από έναν αριθμό κατευθυνόμενων κεραιών και χωρίζουν το χώρο σε κυψέλες. Κάθε κυψέλη αποτελείται από έναν αριθμό καναλιών αναλόγως την κίνηση που περιμένει ο πάροχος να έχει από τους χρήστες. Ένα κανάλι από αυτά χρησιμοποιείται για να μεταδώσει εκτός από πληροφορίες που αφορούν την συγκεκριμένη κυψέλη αλλά και τις ταυτότητες-αναγνωριστικά των γειτονικών κυψελών ώστε να χρησιμοποιηθούν από το κινητό τηλέφωνο για συγκεκριμένους σκοπούς. Το κανάλι αυτό λέγεται Broadcast Control Channel (BCCH). Το κανάλι αυτό εκπέμπει σε τακτά χρονικά διαστήματα σε πλήρης ισχύς σε αντίθεση με τα υπόλοιπα κανάλια που χρησιμοποιούνται στο σταθμό βάσης και το κινητό τηλέφωνο χρησιμοποιεί αυτό το κανάλι για να γίνεται έλεγχος της ενέργειας του σήματος για εξοικονόμηση ρεύματος. Αυτό επιτρέπει στο κινητό τηλέφωνο να συγκρίνει την ισχύ του σήματος από τις γειτονικές κυψέλες και να χρησιμοποιεί την καλύτερη για την επικοινωνία. Συνεπώς γνωρίζοντας την ισχύς του σήματος από το κανάλι BCCH μπορούμε να έχουμε μια εποπτική εικόνα για το που βρίσκεται το κινητό τηλέφωνο σε περίπτωση που δεν έχει επιλεχθεί η μέθοδος multilateration έχοντας αρχικά "εκπαιδεύσει" την εφαρμογή που χρησιμοποιεί αυτή την τεχνική, έτσι ώστε σε επόμενη χρήση της συσκευής να μπορούμε να βρίσκουμε σε ποιο σημείο είμαστε.

1.4.3 Άλλες τεχνολογίες εντοπισμού θέσης

Μία κατηγορία τεχνολογιών εντοπισμού θέσης χρησιμοποιούν τη μέθοδο Near-LBS στην οποία τοπικές σε απόσταση τεχνολογίες, όπως το bluetooth, WLAN και οι υπέρυθρες, χρησιμοποιούνται για να συνδεθούν οι συσκευές-κινητά τηλέφωνα σε τοπικές υπηρεσίες-εξυπηρετητές και αυτοί με τη σειρά τους να παρέχουν στο χρήστη τη τοποθεσία στην οποία βρίσκονται και ενδεχομένως και κάποιες άλλες πληροφορίες δεδομένα σε σχέση με αυτή τη περιοχή.

Μια ενδιαφέρουσα τεχνική εντοπισμού θέσης ενός χρήστη πραγματοποιείται με την εκμετάλλευση μιας καινούργιας τεχνολογίας, του Bluetooth. Η τεχνολογία αυτή αν και σχετικά καινούργια, βρίσκεται σε όλες τις κινητές συσκευές, δεν απαιτεί μεγάλη κατανάλωση μπαταρίας και υλοποιεί ταυτόχρονα πολλές υπηρεσίες δικτύου. Κάθε συσκευή με bluetooth, όταν το χρησιμοποιεί, έχει ένα μοναδικό αναγνωριστικό

και μπορεί να ανιχνευθεί από οποιαδήποτε άλλη συσκευή. Αρχικά βέβαια το bluetooth δεν δημιουργήθηκε για εύρεση θέσης και κατ επέκταση για LBS αλλά για την επικοινωνία χρηστών σε προσωπικά ανοιχτά δίκτυα περιβάλλοντα. Υπάρχουν δύο προσεγγίσεις για την εύρεση θέσης με την τεχνολογία bluetooth:

- Binary location: μια προσέγγιση κατά την οποία κάθε δωμάτιο σε ένα κτίριο έχει εγκαταστημένο ένα Access Point(AP), και στο οποίο το κινητό τηλέφωνο συνδέεται και έτσι γνωρίζουμε σε ποιο χώρο ανήκει.
- Analog location: μια προσέγγιση κατά την οποία τα AP είναι εγκατεστημένα σε κάποια συγκεκριμένα σημεία στο χώρο του κτιρίου, πιο αραιά από την πρώτη περίπτωση, και η απόσταση από κάθε AP μετράται και το σύστημα με βάση τις μετρήσεις αυτές και τη διαδικασία του triangulation (τριγωνισμού) βρίσκει κατά προσέγγιση που βρίσκεται ο χρήστης.

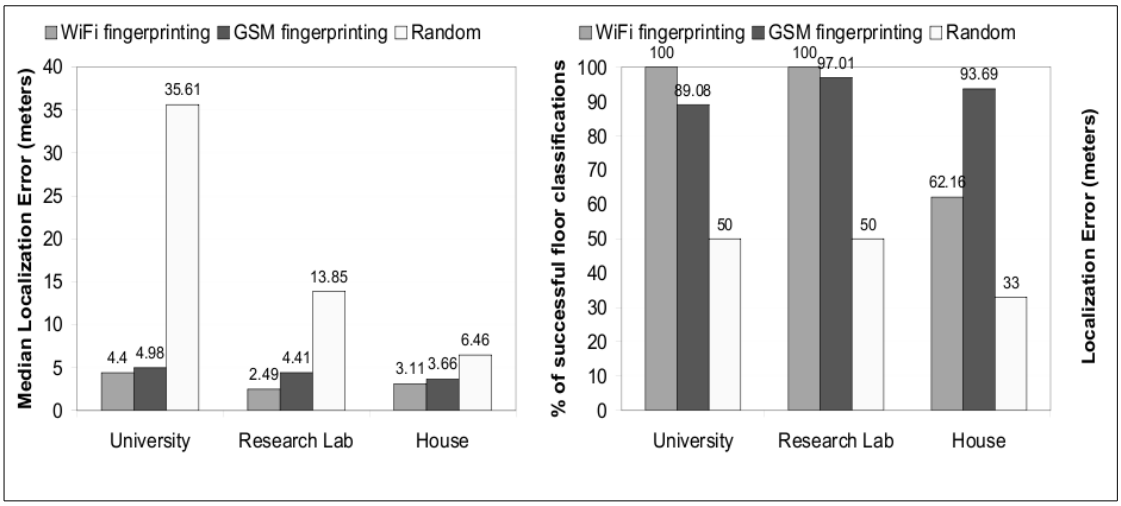
Όπως γίνεται κατανοητό μια τέτοια τεχνολογία που χρησιμοποιείται για την εύρεση της θέσης και συνεπώς την παροχή υπηρεσιών με βάση αυτήν έχει νόημα για τοπικούς χώρους όπως για παράδειγμα μέσα σε ένα συγκρότημα κτιρίων καθώς απαιτείται ένας κεντρικός συντονισμός όλων αυτόν των Access Points.

1.5 Σύγκριση τεχνολογιών εντοπισμού θέσης

Όπως δείχτηκε παραπάνω η δυνατότητα εύρεσης της θέσης ενός χρήστη για την παροχή πληροφοριών και υπηρεσιών με βάση τη θέση αυτή μπορεί να πραγματοποιηθεί με διάφορες τεχνολογίες. Κάθε τεχνολογία έχει πλεονεκτήματα και μειονεκτήματα και συνεπώς καμιά δε μπορεί να θεωρηθεί ότι υπερτερεί αρκετά σε σχέση με τις άλλες. Αυτό όμως που μπορεί να τις διακρίνει και να αξιοποιηθούν κατάλληλα είναι ο χώρος για στον οποίον προτίθεται να χρησιμοποιηθούν.

Σε περιπτώσεις καθαρού ουρανού, δηλαδή για χρήση έξω από κτίρια, η χρήση της τεχνολογίας GPS είναι η ενδεδειγμένη καθώς παρέχει την μεγαλύτερη ακρίβεια στην εύρεση θέσης από οποιαδήποτε άλλη τεχνολογία και είναι προσιτή σε οποιοδήποτε μέρος της Γης είτε πρόκειται σε πεδιάδα, στη θάλασσα είτε σε βραχώδη όρη όπου η κάλυψη από το δίκτυο κινητής τηλεφωνίας μπορεί να μην είναι δυνατή ή το στίγμα που θα δώσει η τεχνολογία GSM Localization να μην είναι ακριβές, να έχει

αρκετά μεγάλη απόκλιση και σε περιπτώσεις που απαιτείται μεγάλη ακρίβεια αυτό να αποτελεί πρόβλημα. Σε περιοχές όπου η δυνατότητα εύρεσης της θέσης μέσω του GPS είναι δύσκολη λόγω φυσικών περιορισμών ή τεχνιτών εμποδίων τότε η τεχνολογία του GSM είναι μια πολύ καλή λύση με αξιόπιστα δεδομένα ειδικά σε περιβάλλοντα αστικά και μέσα σε κτίρια. Σε πρόσφατες μελέτες αποδείχτηκε ότι η χρήση του GSM δικτύου για την εύρεση της θέσης έναντι της χρήσης του wifi όπου ατό είναι δυνατόν να υπάρχει υπερτερεί και φαίνεται στα παρακάτω διαγράμματα.



Τέλος, αν και η τεχνολογία εντοπισμού θέσης με βάση το GSM είναι αξιόπιστη και σε σχέση με το WLAN καλύτερη στο εσωτερικό των κτιρίων όπως αποδεικνύεται παραπάνω, στην περίπτωση ύπαρξης Access Points των Bluetooth τότε ενδείκνυται η χρήση της τεχνολογίας αυτής καθώς μέσα σε κτίρια λόγω των εμποδίων που υπάρχουν η ένδειξη θέσης από το GSM μπορεί να έχει αρκετή απόκλιση από τις πραγματικές τιμές και συνεπώς το σύστημα των AP του δικτύου του bluetooth μπορεί να προσφέρει καλύτερες υπηρεσίες και συνεπώς ο χρήστης να ενημερώνεται με μεγαλύτερη αξιοπιστία και με δεδομένα που αντιστοιχούν στη θέση αυτή.

1.6 Αυτόνομα συστήματα και Υπηρεσίες με βάση τη θέση του χρήστη (LBS)

Συνεπώς από τα παραπάνω προκύπτει ότι απαιτείται ένας συνδυασμός αυτών των τεχνολογιών ώστε να προσφέρουν υπηρεσίες που βασίζονται στη θέση του χρήστη και να καλύπτουν μια μεγάλη περιοχή, χωρίς να "κόβεται" η παροχή υπηρεσιών αλλάζοντας τρόπο επικοινωνίας με τους εξυπηρετητές της εκάστοτε

υπηρεσίας. Αυτή η δυνατότητα αυτόματης εναλλαγής τρόπου εντοπισμού θέσης και ενημέρωσης με πληροφορίες μπορεί να πραγματοποιηθεί με την βοήθεια των Automatic computing techniques. Ουσιαστικά πρόκειται για μία νέα περιοχή στην τεχνολογία και ονομάζεται Autonomic LBS. Σκοπός αυτής της τεχνολογίας είναι η δυνατότητα παροχή υπηρεσιών με βάση τη θέση του χρήστη χωρίς όμως ο ίδιος χρήστης να παρεμβαίνει για να διαχειρίζεται τις αλλαγές που συμβαίνουν στο περιβάλλον του, δηλαδή αλλαγή τρόπου πρόσβασης στο διαδίκτυο, απόκτησης πληροφοριών σχετικά με τη θέση κτλ. Ο στόχος των autonomic LBS μπορεί να διαχωριστεί σε δύο κατηγορίες, πρώτον να κάνει τα LBS να προσαρμόζονται στις αλλαγές του περιβάλλοντος ώστε να δίνουν τις πιο χρήσιμες υπηρεσίες στο χρήστη με τη λιγότερη ενασχόληση του ίδιου και δεύτερον όταν το εσωτερικό η το εξωτερικό περιβάλλον αλλάζει και κάποια ουσιώδη μεταβλητές έχουν ξεπεράσει τα όρια βιωσιμότητας, τα autonomic LBS να διατηρούνται και να πετυχαίνουν την εσωτερική ισορροπία.

Συνεπώς τα autonomic LBS χρειάζονται αισθητήρες για να αντιλαμβάνονται τις αλλαγές στο περιβάλλον τους και εφόσον αυτές οι αλλαγές απαιτούν από την εφαρμογή να κάνει κάτι να ενεργοποιείται και με βάση τις πολιτικές να αποφασίζει για την συμπεριφορά της εφαρμογής.

Τα χαρακτηριστικά που διέπουν τα Autonomic LBS είναι:

- Self-awareness
- Self Optimizing
- Context Aware
- Anticipatory
- Open

1.7 Υπάρχοντες υλοποιήσεις Υπηρεσιών με βάση τη θέση του χρήστη (LBS)

Στα κείμενα (papers) [1],[2],[3] και [4] υπάρχουν αρκετές εφαρμογές για LBS (Cyberguide project [5], GUIDE project [7], CyberAssist project [8], TellMaris project [9], LOL@ System [9, REAL project [10] και Context Sensitive Computing program [11]). Οι τεχνολογίες που χρησιμοποιούνται για να εντοπιστούν οι θέσεις των χρηστών, ακρίβεια και αξιοπιστία που συνδέονται με αυτές τις μετρήσεις και τα

στοιχεία του δικτύου που χρησιμοποιεί κάθε εφαρμογή αναφέρονται στα παραπάνω κείμενα.

Σύμφωνα με το [3] σχεδόν οι μισές εφαρμογές που αφορούν υπηρεσίες βασιζόμενες στη θέση του χρήστη (LBS) χρησιμοποιούν ως σύστημα εντοπισμού το GPS ενώ ένα μεγάλο μέρος αυτών χρησιμοποιούν τις υπέρυθρες ακτίνες. Επιπλέον τα μισά από αυτά τα συστήματα χρειάζονται την παρέμβαση-εμπλοκή του χρήστη για να καθοριστεί η θέση του.

Όσον αφορά την αρχιτεκτονική που χρησιμοποιούν οι εφαρμογές αυτές, μερικά συστήματα στηρίζονται στο μοντέλο πελάτη-εξυπηρετητή (client-server) ενώ άλλα δημιουργήθηκαν χρησιμοποιώντας διαδραστικές εφαρμογές. Η πρώτη περίπτωση[20] έχει σαν πλεονέκτημα ότι εάν η σύνδεση είναι αξιόπιστη μεταξύ του πελάτη και του εξυπηρετητή τότε εύκολα έχουμε την υιοθέτηση πολλών χρηστών-πελατών. Η δεύτερη αρχιτεκτονική είναι μια πιο αποκεντρωμένη προσέγγιση, αλλά στηρίζεται η εφαρμογή σε συγκεκριμένες συσκευές και πλατφόρμες ανάπτυξης. Σύμφωνα με το [21] προτείνεται μία νέα αρχιτεκτονική όπου ενεργό και σημαντικό ρόλο διαδραματίζει το ίδιο το δίκτυο του τηλεπικοινωνιακού παρόχου όταν οι εφαρμογές LBS βρίσκονται πάνω σε τερματικά με δυνατότητα σύνδεσης σε κάποιο τηλεπικοινωνιακό πάροχο, οποίος με τη σειρά του θα αναβαθμίσει τον εξοπλισμό του, τοποθετώντας έναν Geolocation Server (GLS) ο οποίος περιέχει μια Geolocation βάση δεδομένων (GLDB). Ένας τέτοιος εξυπηρετητής θα είναι συνδεδεμένος με κάθε MSC/VLR μέσω υψηλών σε ταχύτητα μεταφοράς δεδομένων ενσύρματων γραμμών. Τέλος σύμφωνα με το κείμενο [19] προτείνεται μια αρχιτεκτονική με χρήση bluetooth όπου σε συγκεκριμένα μέρη έχουμε συσκευές Bluetooth tags που ενημερώνουν τους χρήστες που τα ανιχνεύουν για την ύπαρξή τους και μέσω του διαδικτύου με πρόσβαση σε ένα site ενημερώνεται στη συνέχεια ο χρήστης για την περιοχή ή το σημείο ενδιαφέροντος στο οποίο βρίσκεται. Συνεπώς οι εφαρμογές σε αυτή τη περίπτωση θα απευθύνονται στον πάροχο για να ικανοποιήσουν τα αιτήματα των χρηστών τους.

Στο κείμενο [30], ανάμεσα στα άλλα, προηγμένες ασύρματες υπηρεσίες παρέχονται στο διεθνές αεροδρόμιο της Αθήνας βασιζόμενες στο σύστημα GPS και στις WLAN τεχνολογίες σε μία διανεμημένη αρχιτεκτονική βασιζόμενη σε πράκτορες (agent-based architecture).

Η υπαίθρια εύρεση θέσης ενός χρήστη γίνεται σχεδόν αποκλειστικά με τη χρήση του GPS,ενώ σε εσωτερικούς χώρους αυτό είναι αδύνατο, ωστόσο έχουν αναπτυχθεί

συστήματα εντοπισμού θέσης που επιτυγχάνουν τον εντοπισμό με μεγάλη αξιοπιστία και με μια ευρεία γκάμα τεχνολογιών όπως έχουν αναφερθεί . Το Active Badge [14] σύστημα χρησιμοποιεί υπέρυθρους πομπούς και ανιχνευτές και πετυχαίνει 5-10 μέτρα ακρίβεια, ενώ το Cricket [15] χρησιμοποιεί υπερηχητικά κύματα για να υπολογίσει την απόσταση με μερικά εκατοστά απόκλιση. Επιπλέον, ερευνητικά προγράμματα όπως το RADAR [16] χρησιμοποιεί αποτυπώματα από τέσσερα, 802,11 τύπου, σημεία πρόσβασης για να επιτύχει εντοπισμό θέσης των φορητών υπολογιστών με απόκλιση 3-4 μέτρα. Το PlaceLab [17] έχει πιο φιλόδοξα σχέδια προσπαθώντας να δημιουργήσει μία περιεκτική βάση δεδομένων με θέσεις από σταθερά Wi-Fi, GSM και Bluetooth συσκευές ως αναγνωριστικά. Τέλος στο κείμενο [18] παρουσιάζεται ένα GSM σύστημα εντοπισμού θέσης για εσωτερικούς χώρους όπου επιτυγχάνει ακρίβεια των 5 μέτρων σε μεγάλα πολυώροφα κτίρια.

Αυτά τα συστήματα παρουσιάζουν ένα εύρος ακρίβειας και απαιτήσεων συστήματος ενώ μερικά από αυτά προσφέρουν λύσεις μόνο για υπαίθριους ή εσωτερικούς χώρους, όπως όταν χρησιμοποιείται αποκλειστικά μόνο η τεχνολογία του GPS. Ωστόσο κοινό χαρακτηριστικό αυτών προαναφερθέντων συστημάτων είναι ότι συνδέονται στενά με τη τεχνική εντοπισμού θέσης μιας συσκευής που αφορά έναν χρήστη ή τεχνικές και τεχνολογίες για τον εντοπισμό αυτών.

1.8 Διαχωρισμός εφαρμογών με Υπηρεσίες με βάση τη θέση του χρήστη (LBS)

Στο παρών σημείο γίνεται μια προσπάθεια κατηγοριοποίησης των εφαρμογών που χρησιμοποιούν την εύρεση θέσης για παροχή υπηρεσιών με βάση τις λειτουργικότητες που παρέχουν στους τελικούς χρήστες και της εμπορική κατάσταση των υπηρεσιών αυτών:

- Πλοήγηση: Η ομάδα αυτή αποτελείται από εφαρμογές που βοηθούν στην εύρεση της διαδρομής και παρέχουν υπηρεσίες πλοήγησης για το συγκεκριμένο προορισμό.
- Εύρεση και εντοπισμός: Σκοπός της υπηρεσίας είναι να βρει έναν χρήστη ή ένα συγκεκριμένο αντικείμενο ή τόπο. Η εφαρμογές εντοπισμού έχουν στόχο να εντοπίσουν ένα άνθρωπο, ένα αυτοκίνητο, ένα κατοικίδιο ή κάτι άλλο συγκεκριμένο. Ο συνδυασμός αυτών των λειτουργιών κάνει δυνατή την

επιτυχή ομαδοποίηση διαχείρισης εφαρμογών που προορίζονται να εντοπίζουν και να βρίσκουν ένα σύνολο από αντικείμενα και τον αποτελεσματικό διαχειρισμό τους.

- Location based content delivery: Αυτό μπορεί να πραγματοποιηθεί κυρίως με δύο τρόπους: Directory search (πληροφορίες με βάση τη θέση στη πλευρά του client) ή Push based delivery.
- Geotagged content making: Μερικές υπηρεσίες βασίζονται στη θέση του χρήστη και στις πληροφορίες που ο ίδιος δημοσιεύει κατά επιθυμία (φωτογραφίες σχετικά με κάποιες περιοχές, σχόλια για τουρίστες, εγκαταστάσεις στο χάρτη κτλ)
- Location enchanced Communications and social networking: Εκτεταμένοι τρόποι επικοινωνίας μεταξύ χρηστών με την προσθήκη δυνατοτήτων location awareness στις κοινές υπηρεσίες, όπως είναι instant messaging ή push-to-talk. Επιπλέον οι LBS βοηθούν στην υποστήριξη στις κινητές συσκευές και κυρίως στα τηλέφωνα την ιδέα του community όπως αυτό επιτυχημένα επιτεύχθηκε στο διαδίκτυο.
- Location Based charging: Διάφορα συστήματα χρέωσης, που ποικίλουν ανάλογα με την τοποθεσία του χρήστη, προτείνουν οι πάροχοι για να ανταγωνιστούν τις ενσύρματες συνδέσεις στο διαδίκτυο, όπως τα DSL συστήματα.

2 Λογισμικό Android

2.1 Πακέτα

Τα τελευταία χρόνια δίνεται ιδιαίτερη έμφαση από πλευράς τεχνολογίας στα κινητά τηλέφωνα και συγκεκριμένα στις ανάγκες που μπορούν να καλύψουν. Η εποχή της χρήσης του κινητού τηλεφώνου για απλή συνομιλία και αποστολή γραπτών μηνυμάτων(sms,mms,ems) έχει περάσει ανεπιστρεπτί. Οι καταναλωτές απαιτούν από τις συσκευές αυτές όλο και περισσότερες δυνατότητες. Αποτέλεσμα αυτών, είναι η ανάπτυξη ισχυρών τηλεφώνων από άποψη hardware και συνεπώς η αναγκαιότητα δημιουργίας λογισμικού που θα εκμεταλλεύεται αυτήν την επεξεργαστική ισχύ γίνεται ολοένα και μεγαλύτερη. Το τελευταίο χρόνο κυκλοφόρησε στην αγορά το λογισμικό της Google για τα κινητά τηλέφωνα, γνωστό ως Android. Το εγχείρημα αυτό υποστηρίχτηκε από την Open Handset Alliance έναν συνεταιρισμό 48 hardware, software και telecom εταιρειών αποφασισμένες να στηρίξουν τον ανοιχτό κώδικα λογισμικού για τις συσκευές. Η συμβολή αυτού του λογισμικού στην ανάπτυξη των υπηρεσιών LBS είναι πάρα πολύ σημαντική καθώς τόσο οι δυνατότητες ανάκτησης θέσης είναι αρκετές όσο και σε επίπεδο λογισμικού με τις εφαρμογές που παρέχει στους χρήστες όσο και με τα πακέτα-βιβλιοθήκες για ανάπτυξη εφαρμογών από τους προγραμματιστές-developers που εκμεταλλεύονται τις τεχνολογίες για εντοπισμό της θέσης του χρήστη.

Συγκεκριμένα το λογισμικό Android παρέχει στους προγραμματιστές τη βιβλιοθήκη com.google.android.maps η οποία δίνει τη δυνατότητα σε κάθε εφαρμογή να δείχνει στο γραφικό της περιβάλλον-οθόνη, έναν χάρτη με μία ευρεία γκάμα εστίασης. Το πλεονέκτημα αυτού του πακέτου είναι το γεγονός πως ο χρήστης δε χρειάζεται να ενεργοποιήσει τον explorer ή κάποιο άλλο πρόγραμμα του κινητού προκειμένου να έχει πρόσβαση σε χάρτες, αλλά απευθείας μπορεί να δει στην εφαρμογή του το χάρτη αρκεί βέβαια να έχει πρόσβαση στο διαδίκτυο για να μπορέσει να κατεβάσει το τμήμα του χάρτη που θέλει. Επίσης μπορεί να μετακινήσει το χάρτη προς οποιαδήποτε κατεύθυνση ή να αλλάξει επίπεδο εστίασης προκειμένου να έχει μια πιο εποπτική εικόνα.

Επίσης εκτός από το πακέτο com.google.android.maps το Android προσφέρει το πακέτο android.location το οποίο δίνει στο προγραμματιστή τη δυνατότητα να χρησιμοποιήσει στις εφαρμογές του το δέκτη gps, το σύστημα του παρόχου για

απόκτηση πληροφοριών θέσης από τις κεραίες κινητής τηλεφωνίας και από το ασύρματο δίκτυο στο οποίο είναι συνδεδεμένο το κινητό με τη βοήθεια της google για την απόκτηση μιας περιοχής σύγκλισης για το που βρίσκεται ο χρήστης. Από τις παραπάνω τρεις δυνατότητες η πιο ακριβής είναι ο δέκτης gps και ο λιγότερο ακριβής η χρήση του ασύρματου διαδικτύου. Με το πακέτο αυτό ο προγραμματιστής έχει τη δυνατότητα να ενημερώνει την εφαρμογή του για το πού βρίσκεται ο χρήστης και αν απαιτείται να προβάλει τη θέση του με το com.google.android.maps πακέτο όπως προαναφέρθηκε. Σημαντικό χαρακτηριστικό του πακέτου android.location είναι πως δεν απαιτείται αναγκαστικά οι πληροφορίες θέσης να ενημερωθούν με το που χρησιμοποιείται το πακέτο αυτό αλλά μπορεί να χρησιμοποιηθούν πληροφορίες θέσης που είναι out of date και όμως μπορούν να προσδώσουν στην εφαρμογή τις πληροφορίες που ίσως απαιτεί.

2.2 Αρχιτεκτονική

Μια νέα πλατφόρμα όπου βασίζεται στο Linux Kernel και η καινοτομία του είναι πως είναι ανοιχτό στο κόσμο, είναι ελεύθερο, και ο οποιοσδήποτε μπορεί να το τροποποιήσει και να φτιάξει μια δικιά του εκδοχή του λειτουργικού. Τα δομικά μέρη της αρχιτεκτονικής του Android είναι ο Linux Kernel και ο προγραμματισμός σε java περιβάλλον, αρκετά διαδεδομένη γλώσσα και συνεπώς προσιτή σε έναν μεγάλο αριθμό προγραμματιστών που θέλουν να πάρουν μέρος στην εξέλιξη του λογισμικού. Τα βασικά χαρακτηριστικά του είναι τα εξής:

- Application framework το οποίο επιτρέπει την επαναχρησιμοποίηση και αντικατάσταση των components
- Davlik virtual machine optimized για κινητές συσκευές
- Integrated browser βασισμένος στην open source μηχανή WebKit
- Optimized graphics (custom 2D library) (3D βασισμένο στο OpenGL ES 1.0 specification (hardware accelaration optional)
- SQLite for structured data storage
- Multimedia υποστήριξη για σχεδόν όλα τα διάσημα formats video ήχου και εικόνας
- GSM telephony
- Bluettoth, 3G,EDGE και WIFI
- Camera, GPS, compass, και accelometer

Το λογισμικό αυτό βασίζεται στον πυρήνα του Linux έκδοση 2.6 για τις κύριες λειτουργίες όπως:

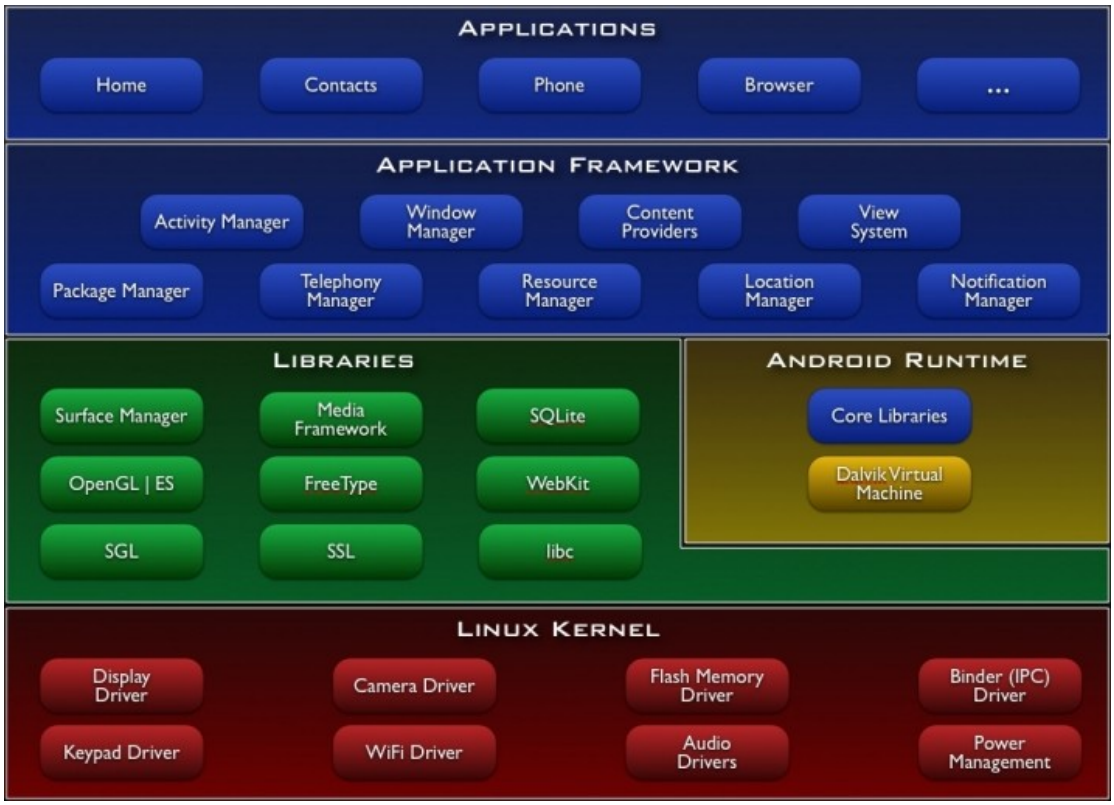
- Security
- Memory Management
- Process Management
- Nework Stack
- Driver Model

Παρόλο που το Android είναι χτισμένο πάνω στο πυρήνα του Linux δεν είναι Linux. Ο πυρήνας δρα σαν abstraction layer μεταξύ του hardware και του υπόλοιπου software stack.

Οι βιβλιοθήκες του Android περιλαμβάνουν ένα σετ από C/C++ βιβλιοθήκες που χρησιμοποιούνται από διάφορα components του συστήματος. Αυτές διατίθενται στους προγραμματιστές μέσω του Android application framework. Μερικές από τις βιβλιοθήκες είναι οι παρακάτω:

- System C library - a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices
- Media Libraries - based on PacketVideo's OpenCORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
- Surface Manager - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications
- LibWebCore - a modern web browser engine which powers both the Android browser and an embeddable web view
- SGL - the underlying 2D graphics engine
- 3D libraries - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D
- software rasterizer
- FreeType - bitmap and vector font rendering
- SQLite - a powerful and lightweight relational database engine available to all applications

Παρακάτω δίνεται η αρχιτεκτονική του Android με τα components του συστήματος:



2.3 Δυνατότητες Ανάπτυξης Εφαρμογών

Οι εφαρμογές έχουν την ίδια ισχύ, καμία δεν προωθείται περισσότερο από κάποια άλλη ούτε αποκλείει η μία την άλλη. Αυτό πρακτικά σημαίνει πως οι εφαρμογές που έρχονται μαζί με το Android δεν υπερτερούν σε τίποτα σε σχέση με αυτές που υλοποιεί ένας προγραμματιστής καθώς και αυτός έχει πρόσβαση στα δομικά μέρη της συσκευής. Επίσης το Android υπερτερεί στο γεγονός ότι έχει τη δυνατότητα του multi-tasking δηλαδή πολλαπλές εφαρμογές τρέχουν ταυτόχρονα, ενώ δεν απαιτείται να κλείσει καμιά εφαρμογή, όταν ο χρήστης επιλέξει να βγει από μία εφαρμογή αυτή συνεχίζει να εκτελείται στο background. Η υλοποίηση νέων εφαρμογών είναι εύκολη, χρησιμοποιώντας το ανοιχτό λογισμικό-πρόγραμμα eclipse μαζί με το Android SDK. Η δυνατότητα δημιουργίας εφαρμογών δεν είναι δύσκολη, μόλις ο προγραμματιστής αντιληφθεί τη λογική ανάπτυξής της. Οι εφαρμογές στο Android αποτελούνται από τα παρακάτω Components:

- Activities Κυρίως παράθυρα, GUI-συγκροτούνται από Views
- Services Υπηρεσίες που εκτελούνται στο background

- Broadcast Receivers Components που περιμένουν να ενεργοποιηθούν από ένα συμβάν
- Content Providers Αποθηκεύουν τα δεδομένα και τα κρατούν διαθέσιμα σε άλλες εφαρμογές

Ένα σημαντικό γεγονός που ευνοεί την ανάπτυξη εφαρμογών είναι πως το πακέτο Android SDK συνεργάζεται με το eclipse και συνεπώς ο προγραμματιστής μπορεί εύκολα και γρήγορα να βλέπει τις αλλαγές στο κώδικα στον emulator που του παρέχει το Android SDK χωρίς να χρειάζεται να εξάγει κάθε φορά την εφαρμογή και να την εγκαθιστά σε κινητό. Επίσης ο emulator είναι πολύ αξιόπιστος καθώς έχει ακριβώς την ίδια συμπεριφορά αν η εφαρμογή εγκατασταθεί σε ένα κινητό τηλέφωνο Android. Τέλος ένα άλλο σημαντικό πλεονέκτημα είναι το γεγονός πως σε αναβαθμίσεις του λογισμικού η εφαρμογή εξακολουθεί να δουλεύει χωρίς την ανάγκη επανασχεδιασμού κάποιων σημαντικών κομματιών του κώδικα που αφορά την αλληλεπίδραση της εφαρμογής με τα δομικά μέρη-hardware- του κινητού τηλεφώνου.

2.4 Υπάρχουσες υλοποιήσεις-εφαρμογές

Όπως προαναφέρθηκε το λογισμικό Android είναι μια νέα πλατφόρμα στην αγορά, ωστόσο η απήχησή του στο καταναλωτικό κοινό και στην κοινότητα των προγραμματιστών τόσο σε επίπεδο ερασιτεχνικής ενασχόλησης όσο και επαγγελματικά μεμονωμένα ατομικά ή σε επίπεδο εταιρειών είναι μεγάλη και αυτό φαίνεται από την πληθώρα των εφαρμογών που δημιουργούνται κάθε μέρα. Οι εφαρμογές είναι διαθέσιμες στους καταναλωτές-χρήστες κινητών τηλεφώνων Android μέσω της εφαρμογής Market που παρέχεται εγκατεστημένη στο κινητό τηλέφωνο. Εκεί ο κάθε χρήστης μπορεί εύκολα και απλά να αναζητήσει και να βρει εφαρμογές που χρειάζεται ή απλά θέλει να έχει στη συσκευή του. Μπορεί είτε να αναζητήσει μια εφαρμογή περιγράφοντας περιεκτικά σε λίγες λέξεις τι θέλει η εφαρμογή να κάνει είτε αν γνωρίζει το όνομα της εφαρμογής που αναζητεί να το πληκτρολογήσει και στη συνέχεια να την κατεβάσει και να την εγκαταστήσει το σύστημα αυτόματα. Άλλος τρόπος είναι να χρησιμοποιήσει το ήδη διαχωρισμένο σε κατηγορίες menu της εφαρμογής που χωρίζει τις εφαρμογές με βάση το περιεχόμενό τους, δηλαδή με βάση τη κατηγορία εφαρμογής είναι. Οι διαθέσιμες κατηγορίες όπως μέχρι τώρα είναι διαμορφωμένες από την εταιρεία Google είναι:

Αρχικά σε δυο μεγάλες κατηγορίες:

- Applications
- Games

Και η κάθε μία κατηγορία ομαδοποιεί τις εφαρμογές-προγράμματα με βάση τις ανάγκες που καλύπτουν:

Applications	**Games**
Comics	Arcade & Action
Communication	Brain &Puzzle
Entertaiment	Cards & Casino
Finance	Casual
Health	
LifeStyle	
Multimedia	
New & Weather	
Productivity	
Reference	
Shopping	
Social	
Sports	
Themes	
Tools	
Travel	
Demo	

Στις κατηγορίες αυτές οι εφαρμογές εμφανίζονται και με βάση αν ο χρήστης θέλει να χρησιμοποιήσει εφαρμογές που είναι διαθέσιμες επί πληρωμή ή είναι δωρεάν. Σε μερικές περιπτώσεις ανάλογα ποιος πάροχος εξυπηρετεί έναν χρήστη είναι δυνατόν οι επί πληρωμή εφαρμογές να μην είναι διαθέσιμες γιατί δεν έχουν οριστικοποιηθεί οι συμφωνίες μεταξύ παρόχου και Google για το διαμοιρασμό των εσόδων από αυτές. Υπάρχει όμως η δυνατότητα να αγοράσει ο χρήστης την εφαρμογή από το διαδίκτυο και να την εγκαταστήσει στο κινητό του τηλέφωνο.

Σύμφωνα με έρευνες, οι συνολικές εφαρμογές-προγράμματα που υπάρχουν στο Market φθάνουν τις 10,000 περίπου. Η έρευνα αυτή πραγματοποιήθηκε στα τέλη του περασμένου Σεπτέμβρη ενώ στις αρχές Μαΐου οι εφαρμογές ήταν περίπου 4900. Φαίνεται ξεκάθαρα πόσο γρήγορα αναπτύσσεται αυτή η κοινότητα και αυτό είναι δυνατόν καθώς το λογισμικό είναι ανοιχτό και ο καθένας μπορεί να δημιουργήσει μια εφαρμογή. Τέλος από αυτές τις εφαρμογές το 64,2% είναι δωρεάν ενώ μόλις το 35,8% είναι επί πληρωμή. Το σημαντικό είναι πως το 81% των προγραμμάτων που διατίθενται στο Market είναι εφαρμογές και μόλις το 19% αφορούν παιχνίδια.

2.4.1 Υπάρχουσες υλοποιήσεις για Υπηρεσίες με βάση τη θέση του χρήστη (LBS)

Οι περισσότερες υλοποιήσεις για LBS καλύπτουν την ανάγκη για πλοήγηση από σημείο σε σημείο χωρίς προηγουμένως να γνωρίζει ο χρήστης τη διαδρομή αλλά μέσω της εφαρμογής να βλέπει και να ακολουθεί τη διαδρομή που του προτείνει. Τέτοιες εφαρμογές είναι το google maps, το ndrive, AndNav και TeleNav.

Άλλη κατηγορία υλοποιήσεων με χρήση της θέσης του χρήστη είναι οι εφαρμογές εκείνες που στόχο έχουν την ενημέρωση για το καιρό που επικρατεί ή πρόβλεψη του για το κοντινό μέλλον στη περιοχή του χρήστη. Στη κατηγορία αυτή ανήκουν εφαρμογές όπως το weatherbug και weather widget.

Τέλος άλλες εφαρμογές αξιοποιούν τις δυνατότητες των LBS για την ενημέρωση του χρήστη για σημεία ενδιαφέροντος κοντά στη περιοχή που βρίσκεται, και μπορεί να αφορά πρατήρια βενζίνης super market κινηματογράφους κτλ. Μία τέτοια εφαρμογή είναι το Google Places.

3 Περιγραφή Εφαρμογής

3.1 Αντικείμενο της διπλωματικής

Στην παρούσα διπλωματική αναπτύχθηκε μια εφαρμογή που επιτρέπει στους χρήστες να αλληλεπιδρούν με άλλους χρήστες που βρίσκονται στο κοντινό τους σχετικά περιβάλλον. Οι χρήστες έχουν την δυνατότητα να επικοινωνούν με άλλους χρήστες με ανταλλαγή μηνυμάτων σε πραγματικό χρόνο και σε μορφή κανονικού chat. Επίσης μπορούν να συνδεθούν στο global chat room και εκεί να ανταλλάξουν και πάλι σε πραγματικό χρόνο μηνύματα. Δίνεται η δυνατότητα εφόσον το επιθυμούν οι χρήστες να διαθέσουν στους υπόλοιπους χρήστες διάφορα αρχεία που είναι αποθηκευμένα στη μνήμη του κινητού τηλεφώνου τους και εν συνεχεία αν κάποιο από τα αρχεία που διαμοιράζουν άλλοι χρήστες θελήσουν να το αποκτήσουν μπορούν να το «κατεβάσουν» στο κινητό τους τηλέφωνο. Τέλος έγινε μια προσπάθεια, η εφαρμογή όσον αφορά στην συνομιλία των χρηστών, να ανεξαρτητοποιηθεί από την επικοινωνία με τον τοπικό εξυπηρετητή ώστε να έχουμε αυτόνομες επικοινωνίες μεταξύ των χρηστών. Στη περίπτωση που ο εξυπηρετητής δεν είναι διαθέσιμος, ή ο χρήστης δε θελήσει να συνδεθεί στον εξυπηρετητή, μπορεί να αναζητήσει χρήστες που έχουν κάνει την ίδια επιλογή και να επικοινωνήσει μαζί τους αναλόγως τι υπηρεσία έχουν αποφασίσει να χρησιμοποιήσουν(global chat ή private chatting). Η εφαρμογή μπορεί να συγκριθεί με το γνωστό σύστημα Live Messenger της Microsoft αλλά η ειδοποιός διαφορά έγκειται στην προσπάθεια απεξάρτησης όσο γίνεται από την ανάμειξη του εξυπηρετητή στην επικοινωνία των χρηστών. Αυτό έχει βέβαια το κόστος ο χρήστης να μπορεί να επικοινωνεί μόνο με χρήστες που βρίσκονται στο ίδιο δίκτυο με αυτόν, όπως είναι το ασύρματο δίκτυο του Πολυτεχνείου.

3.2 Περιγραφή Προβλήματος

Η επικοινωνία ανάμεσα στους χρήστες, ανά πάσα στιγμή, είναι ένα φαινόμενο που όλο και γίνεται πιο αναγκαίο να επιτευχθεί καθώς ο άνθρωπος είναι ένα κοινωνικό ων και καθώς η τεχνολογία προχωρά θέλει όπου βρίσκεται να επικοινωνεί όταν υπάρχουν φυσικά ή τεχνικά εμπόδια που αποθαρρύνουν την φυσική επικοινωνία.

- Κάποιος βρίσκεται στο χώρο του πολυτεχνείου και θέλει να μάθει ποιος άλλος είναι στον ίδιο χώρο ώστε να κανονίσουν να συναντηθούν και να μιλήσουν από κοντά.
- Ένας φοιτητής βρίσκεται σε μια διάλεξη ή σεμινάριο και κρατά σημειώσεις είτε χειρόγραφα είτε σε αρχείο στο κινητό του και στη συνέχεια θέλει να διαμοιράσει αυτές τις σημειώσεις σε άλλους συμφοιτητές του, ειδικά όταν χρησιμοποιούν Android κινητά τηλέφωνα όπου η ανταλλαγή αρχείων μέσω Bluetooth είναι απενεργοποιημένη.
- Χρήστες βρίσκονται σε έναν κοινό χώρο όπου απαιτείται ησυχία ή δεν είναι δυνατή η άμεση επαφή και παρόλα αυτά πρέπει να διαμοιράζονται μεταξύ τους κάποιες πληροφορίες ή να κάνουν σχόλια.
- Ένα άλλο σημαντικό θέμα είναι η ανάγκη του ανθρώπου να είναι σε συνεχή επαφή με τους ανθρώπους και να συνομιλεί μαζί τους.

3.3 Επίλυση Προβλήματος

Η εφαρμογή που αναπτύχθηκε μπορεί να βοηθήσει στις παραπάνω περιπτώσεις και να τις επιλύσει. Ο χρήστης είτε κάνοντας εγγραφή στον εξυπηρετητή το τοπικό που έχει την ευθύνη του συντονισμού των χρηστών με την ίδια εφαρμογή είτε χρησιμοποιώντας την αναζήτηση στο τοπικό δίκτυο, να εγγραφεί στην υπηρεσία private chatting και εν συνεχεία να πάρει μία λίστα με όλους τους χρήστες που είναι συνδεδεμένοι στην ίδια υπηρεσία και να ανταλλάσσει με τον καθέναν από αυτούς μηνύματα, και άρα να αποκτήσει επικοινωνία σε πραγματικό χρόνο. Αν επιλέξει να εγγραφεί στο global chat-room τότε θα μπορεί να αποστέλλει και να παραλαμβάνει γενικά μηνύματα. Τέλος αν επιλέξει την υπηρεσία ανταλλαγής δεδομένων ο χρήστης μπορεί να κάνει γνωστό τα αρχεία που έχει διαθέσιμα στο κινητό του τηλέφωνο και ταυτόχρονα να παραλάβει αρχεία από άλλους χρήστες.

3.4 Ανάλυση Απαιτήσεων Συστημάτων

Ακολουθεί η περιγραφή της αρχιτεκτονικής του συστήματος και η ανάλυση των απαιτήσεων για την λειτουργία του.

3.4.1 Απαιτήσεις Συστήματος

Το σύστημα που υλοποιήθηκε ακολουθεί το μοντέλο client-server, τόσο μεταξύ του κινητού τηλεφώνου και εξυπηρετητή όσο και μεταξύ των κινητών τηλεφώνων. Υπάρχει ένας κεντρικός-τοπικός εξυπηρετητής(Server) με τον οποίο όλα τα κινητά τηλέφωνα επικοινωνούν εφόσον το επιθυμούν και τα ενημερώνει για τις διαθέσιμες υπηρεσίες. Είτε ο χρήστης επιλέξει να συνδεθεί στον εξυπηρετητή είτε όχι, το κάθε τερματικό είναι ταυτόχρονα ένας τοπικός εξυπηρετητής και τερματικό, όπου στη περίπτωση του εξυπηρετητή ενημερώνει σε ποια υπηρεσία είναι εγγεγραμμένος ο χρήστης και δέχεται αναλόγως της υπηρεσίας ανάλογα αιτήματα, πχ αίτηση για συνομιλία ή αποστολή αρχείων. Στη περίπτωση του τερματικού, ο χρήστης στέλνει αιτήματα για συνομιλία ή λήψη αρχείων.

3.4.2 Το ιδανικό σύστημα

Παραπάνω αναφέρθηκε ότι το σύστημα που υλοποιήθηκε αποτελείται από δύο υποσυστήματα, τον κεντρικό-τοπικό εξυπηρετητή και το τερματικό-κινητό τηλέφωνο.

Σε ιδανικές συνθήκες ο εξυπηρετητής είναι πάντα ενεργός και προσπελάσιμος από όλα τα τερματικά που ανήκουν στο ίδιο τοπικό δίκτυο, ώστε να μπορούν τα κινητά τηλέφωνα να παίρνουν άμεσα και γρήγορα όλες τις απαραίτητες πληροφορίες που χρειάζονται. Απαιτείται να έχει μεγάλη διαθέσιμη μνήμη και το απαιτούμενο εύρος ζώνης να είναι αρκετό ώστε να ικανοποιούνται όλες οι συνδέσεις που ζητούνται από τους χρήστες-κινητά τηλέφωνα. Τα δεδομένα που ανταλλάσσονται μεταξύ του εξυπηρετητή και του εκάστοτε χρήστη, δεν είναι ογκοβόρα οπότε αυτό που απαιτείται να ικανοποιείται, είναι η δυνατότητα γρήγορων συνδέσεων-επικοινωνία, η γρήγορη ανταλλαγή δεδομένων από τον εξυπηρετητή στο κινητό τηλέφωνο. Το σύστημα του εξυπηρετητή-Server θα πρέπει να παρέχει στους διαχειριστές τη δυνατότητα προβολής στατιστικών στοιχείων με βάση τις προτιμήσεις των χρηστών ώστε σε μελλοντικές αναβαθμίσεις του λογισμικού της εφαρμογής στην πλευρά των κινητών τηλεφώνων να δίνεται έμφαση σε αυτές τις υπηρεσίες που είναι δημοφιλείς.

Τα κινητά τηλέφωνα που έχουν την εφαρμογή του client, και ουσιαστικά την ίδια την εφαρμογή που αυτή είναι υπεύθυνη για την αλληλεπίδραση με τους χρήστες και με τα άλλα κινητά τηλέφωνα απαιτούν επαρκή μνήμη και εν συνεχεία υπολογιστική

ισχύ ικανή να αντεπεξέλθει στις πολλαπλές διεργασίες που εκτελούνται κατά την διάρκεια της εφαρμογής. Τα κινητά τηλέφωνα που κυκλοφορούν στην αγορά καλύπτουν αυτές τις απαιτήσεις.

Ο κάθε χρήστης μπορεί να είναι συνδεδεμένος σε υπηρεσίες, όπως είναι το private chatting, ανταλλαγή μηνυμάτων με άλλους χρήστες όπως είναι το γνωστό Live Messenger. Άλλη υπηρεσία είναι η δυνατότητα ανταλλαγής αρχείων μεταξύ των χρηστών και η δυνατότητα συμμετοχής σε public chat-rooms όπου όλοι βλέπουν τα μηνύματα που αποστέλλουν οι άλλοι και μπορεί να αποστείλει το δικό του μήνυμα, έχει ακόμη τη δυνατότητα να επιλέγει σε ποιο chat room θα μπει κατά την είσοδό του στην υπηρεσία ή να δημιουργήσει ένα νέο chat room.

Για να μπορεί να λειτουργήσει το σύστημα απαιτείται το κινητό τηλέφωνο να είναι συνδεδεμένο σε κάποιο δίκτυο, είτε αυτό είναι το τοπικό δίκτυο μέσω του wifi, είτε το δίκτυο που μας παρέχει ο εκάστοτε provider κινητής τηλεφωνίας με το λεγόμενο 3G δίκτυο. Σε κάθε τέτοιο δίκτυο θα πρέπει να υπάρχει και ο αντίστοιχος εξυπηρετητής εφόσον αυτό είναι δυνατόν αλλιώς ο χρήστης όταν διαπιστώσει την απουσία του Server τότε έχει τη δυνατότητα να αναζητήσει άλλους χρήστες. Το πλεονέκτημα αυτή της εφαρμογής είναι η δυνατότητα να λειτουργεί απουσία εξυπηρετητή και internet, απαιτεί μόνο σύνδεση σε ένα τοπικό δίκτυο-router στην πιο απλή περίπτωση.

Το πρόβλημα των πολλαπλών εξυπηρετητών μπορεί να λυθεί αν έχουμε έναν κεντρικό εξυπηρετητή ο οποίος θα αναλαμβάνει την «ομαδοποίηση» των χρηστών με βάση σε ποιο δίκτυο βρίσκονται και φυσικά θα πρέπει τα κινητά τηλέφωνα-τερματικά να έχουν πρόσβαση σε αυτόν μέσω του διαδικτύου.

3.4.3 Το σύστημα που υλοποιήθηκε

Στο πλαίσιο της συγκεκριμένης διπλωματικής υλοποιήθηκε το παρακάτω σύστημα: Η εφαρμογή εξυπηρετητή-Server είναι υλοποιημένη σε java SE. Παρέχει τη δυνατότητα πολλαπλών συνδέσεων από χρήστες και την παροχή τριών υπηρεσιών, private chatting, public chat και ανταλλαγή δεδομένων που είναι υλοποιημένες και στην εφαρμογή που είναι εγκατεστημένη στα κινητά τηλέφωνα. Η εφαρμογή του εξυπηρετητή δείχνει το ιστορικό των συνδιαλέξεων που έχει από τη στιγμή της ενεργοποίησης της και επιπλέον παρέχει δυνατότητα απλής παρουσίασης των χρηστών που χρησιμοποιούν την εφαρμογή με τη βοήθεια του εξυπηρετητή και σε

ποια υπηρεσία είναι εγγεγραμμένοι. Οι χρήστες που είναι εγγεγραμμένοι κατά τη διάρκεια της χρήσης της εφαρμογής καθώς και ποιες υπηρεσίες παρέχονται από το σύστημα αποθηκεύονται σε μια βάση δεδομένων υλοποιημένη με mySQL. Η εφαρμογή στα κινητά τηλέφωνα-τερματικά υλοποιήθηκε σε java με χρήση Android βιβλιοθηκών ώστε να μπορεί να εγκατασταθεί σε όλα τα κινητά τηλέφωνα που χρησιμοποιούν Android λογισμικό και έχουν φυσικά τη δυνατότητα σύνδεσης σε wifi δίκτυα ή σε δίκτυο του παρόχου. Ο χρήστης έχει τη δυνατότητα να επιλέξει σύνδεση με τον εξυπηρετητή και εν συνεχεία να εγγραφεί σε μια από τις τρεις υλοποιημένες υπηρεσίες που του παρέχει ο εξυπηρετητής ή να αναζητήσει αυτόματα η εφαρμογή άλλους χρήστες στο τοπικό δίκτυο που είναι συνδεδεμένο εφόσον πρώτα επιλέξει μια από τις δύο διαθέσιμες υπηρεσίες, private chatting ή public chat. Σημαντικό είναι να επισημάνουμε ότι για την αποθήκευση των χρηστών που είναι διαθέσιμοι στο σύστημα χρησιμοποιήθηκε η βάση δεδομένων SQLite database που είναι διαθέσιμη από το Android και για κάθε εφαρμογή αποθηκεύεται σε ξεχωριστό φάκελο μέσα στην εφαρμογή. Τέλος ο χρήστης έχει τη δυνατότητα να βλέπει, σε όποια υπηρεσία και αν είναι εγγεγραμμένος, που βρίσκονται και οι υπόλοιποι χρήστες στο χάρτη αρκεί να ανήκουν στην ίδια υπηρεσία με αυτόν και έχουν συνδεθεί σε αυτήν με τον ίδιο τρόπο,δηλαδή είτε με τη βοήθεια του εξυπηρετητή είτε με τοπική αναζήτηση στο δίκτυο.

3.5 Περιγραφή Παρεχόμενων Υπηρεσιών

3.5.1 Εφαρμογή κινητού τηλεφώνου

Η εφαρμογή επιτρέπει στους χρήστες να επικοινωνούν με άλλους χρήστες εφόσον πρώτα επιλέξουν πως θα ενημερωθούν για την ύπαρξη άλλων χρηστών που χρησιμοποιούν την υπηρεσία στην οποία θέλουν να εγγραφούν. Συγκεκριμένα έχουν τις εξής επιλογές:

- Δυνατότητα να δουν εφόσον έχουν καθαρό ουρανό-clear sky που ακριβώς βρίσκονται στο χάρτη
- Μπορούν να εισάγουν και να τροποποιούν τα προσωπικά τους στοιχεία
- Επιλογή τρόπου ενημέρωσης ύπαρξης άλλων χρηστών
- Επιλογή με βάση τον παραπάνω τρόπο ενημέρωσης σε ποια υπηρεσία επιθυμούν να συνδεθούν

- Εφόσον έχουν επιλέξει το private chatting μπορούν να επικοινωνήσουν με έναν άλλο χρήστη και να ανταλλάξουν μηνύματα
- Μπορούν μέσω του public chat να αποστέλλουν στα άλλα μέλη του chat μηνύματα και να παίρνουν από αυτά μηνύματα
- Εφόσον έχουν επιλέξει να ενημερωθούν για την ύπαρξη άλλων χρηστών μέσω του εξυπηρετητή να δουν ποια αρχεία είναι διαθέσιμα για αποστολή και ταυτόχρονα να δημοσιεύσουν στους υπολοίπους χρήστες καθώς και να ενημερώσουν τον εξυπηρετητή με τα αρχεία που έχουν προς διάθεση.
- Σε οποιαδήποτε υπηρεσία μπορούν να δουν που βρίσκονται οι υπόλοιποι χρήστες με του οποίους μπορούν να ανταλλάξουν μηνύματα ή δεδομένα.

3.5.1.1 Εκκίνηση Εφαρμογής

Όταν ξεκινά η εφαρμογή ο χρήστης βλέπει ένα χάρτη με default τοποθεσία το κέντρο του Εθνικού Μετσόβιου Πολυτεχνείου εάν υπάρχει σύνδεση με το διαδίκτυο, καθώς το κινητό τηλέφωνο δεν έχει τη δυνατότητα προ-εγκατεστημένων χαρτών. Όταν επιλέξει το πλήκτρο menu από τη συσκευή έχει τη δυνατότητα να επιλέξει μια από τις 3 επιλογές:

- Location Button
- My Info
- Process Login

Κατά τη διάρκεια της παραμονής σε αυτό το περιβάλλον, ο χρήστης ενημερώνεται από το σύστημα σε τακτά χρονικά διαστήματα εάν η σύνδεση με το δορυφόρο και συνεπώς η εύρεση στο χάρτη της τοποθεσίας του είναι εφικτή. Εάν δεν είναι η σύνδεση με το δορυφόρο εφικτή εμφανίζεται μήνυμα “gps unavailable”, σε περίπτωση που έχουμε σήμα από το δορυφόρο εμφανίζεται το μήνυμα “gps available” και ο χρήστης με την επιλογή Location Button μπορεί να δει στην οθόνη

της συσκευής που βρίσκεται στο χάρτη. Εάν όμως η σύνδεση με το δορυφόρο χαθεί τότε σε επιλογή του συγκεκριμένου menu εμφανίζεται μήνυμα “sorry try again later”.

Επιλέγοντας την επιλογή My Info, αν ο χρήστης πρώτη φορά χρησιμοποιεί την εφαρμογή τότε εμφανίζονται τα πεδία για τις προσωπικές πληροφορίες, username, name, lastname κενά και καλείται ο χρήστης να τα συμπληρώσει ώστε να καταχωρηθούν τα προσωπικά του δεδομένα ώστε να μπορεί να εγγραφεί στον εξυπηρετητή και να είναι δυνατόν οι υπόλοιποι χρήστες να τον αναγνωρίζουν. Σε περίπτωση που ο χρήστης έχει ήδη δηλώσει σε προηγούμενη χρήση της επιλογής My Info τα στοιχεία του, τότε στα αντίστοιχα πεδία εμφανίζονται τα καταχωρημένα του στοιχεία, και αν επιθυμεί μπορεί να τα τροποποιήσει.

Τέλος με την επιλογή του Process Login ο χρήστης προχωρά στον τρόπο με τον οποίο θα αναζητήσει τους χρήστες και σε ποια υπηρεσία εν συνεχεία θα εγγραφεί. Αν ο χρήστης εισέρχεται για πρώτη φορά στην εφαρμογή και δεν έχει δηλώσει τα στοιχεία του τότε η διαδικασία για εγγραφή δε θα προχωρήσει αλλά θα εμφανιστεί ένα σύντομο σε διάρκεια μήνυμα που θα ενημερώνει το χρήστη ότι πρέπει πρώτα να εισάγει τα προσωπικά του δεδομένα αν θέλει να χρησιμοποιήσει τις υπόλοιπες δυνατότητες της εφαρμογής.

3.5.1.2 Επιλογή τρόπου Σύνδεσης

Στο menu αυτό ο χρήστης εφόσον έχει καταχωρήσει τα στοιχεία του στην εφαρμογή, έχει τη δυνατότητα να επιλέξει έναν από τους δύο τρόπους για να μπορέσει να αναζητήσει χρήστες.

- Server

Ο χρήστης σε αυτήν την περίπτωση, θα συνδεθεί με το τοπικό κεντρικό εξυπηρετητή προκειμένου να ενημερωθεί για τις υπηρεσίες που διατίθενται από αυτόν τον Server και εν συνεχεία να επιλέξει σε ποια υπηρεσία θα εγγραφεί.

- Auto_Scan

Ο χρήστης επιλέγοντας το Auto_scan θα προσπαθήσει να αναζητήσει με βάση σε ποια υπηρεσία θέλει να πάρει μέρος στο τοπικό δίκτυο στο οποίο είναι συνδεδεμένος άλλους χρήστες-κινητά τηλέφωνα.

3.5.1.3 Χρήση Server

Επιλέγοντας ο χρήστης από την διαδικασία Process Login την επιλογή Server, αυτόματα ο χρήστης μεταβαίνει σε ένα νέο παράθυρο όπου εάν για τον οποιοδήποτε λόγο ο εξυπηρετητής δεν είναι διαθέσιμος, πχ είναι σε συντήρηση από τους διαχειριστές ή προσωρινά για τεχνικούς λόγους είναι αδύνατη η σύνδεση με αυτόν εμφανίζεται ένα μήνυμα που μας ενημερώνει για την αδυναμία να μας εξυπηρετήσει ο Server.

Σε περίπτωση που εμφανιστεί το παραπάνω μήνυμα ο χρήστης πρέπει γυρίσει στο προηγούμενο παράθυρο, να περιμένει για λίγη ώρα και να προσπαθήσει να συνδεθεί ξανά, αλλιώς να επιλέξει την άλλη δυνατότητα.

Εάν ο χρήστης συνδεθεί με επιτυχία στον εξυπηρετητή τότε του εμφανίζονται στο παράθυρο οι υπηρεσίες που είναι διαθέσιμες και επιλέγοντας μία από αυτές μπορεί να εγγραφεί σε αυτές και να ενημερωθεί αναλόγως το τι έχει επιλέξει ποιοι χρήστες είναι εγγεγραμμένοι εκεί ή ποια αρχεία είναι διαθέσιμα ώστε να τα χρησιμοποιήσει στο επόμενο στάδιο της διαδικασίας. Εφόσον ο τοπικός εξυπηρετητής είναι σε «καλή» κατάσταση και δεν γίνεται συντήρηση σε κάποιο μέρος του, ο χρήστης θα δει στην οθόνη του τις τρεις υλοποιημένες, για την διπλωματική αυτή, υπηρεσίες που έχουν αναφερθεί και προγενέστερα (private chatting, public chat, data-exchange).

Μόλις ο χρήστης διαλέξει την υπηρεσία που επιθυμεί τότε εμφανίζεται ένα μήνυμα στην κορυφή του παραθύρου που ζητά από τον χρήστη να περιμένει σε περίπτωση που η διαδικασία ενημέρωσης του κινητού τηλεφώνου για τους χρήστες που είναι στην ίδια υπηρεσία αργεί αρκετά, λόγω χαμηλής ταχύτητας του δικτύου. Σε αντίθετη περίπτωση ο χρήστης θα μεταβεί αμέσως στο παράθυρο της υπηρεσίας την οποία έχει επιλέξει.

3.5.1.4 Χρήση του Private Chatting

Ο χρήστης επιλέγοντας αυτή την υπηρεσία έχει τη δυνατότητα να συνομιλεί με προσωπικά μηνύματα σε διάλογο με οποιοδήποτε χρήστη έχει την ίδια υπηρεσία και έχει επιλέξει προγενέστερα τον ίδιο, με αυτόν, τρόπο σύνδεσης στην υπηρεσία, δηλαδή είτε με τη βοήθεια του εξυπηρετητή-Server είτε Auto_Scan. Το παράθυρο που εμφανίζεται περιέχει όλους τους χρήστες με τους οποίους μπορεί να έρθει σε επικοινωνία και ταυτόχρονα βλέπει τα προσωπικά του δεδομένα, ονοματεπώνυμο προκειμένου να αποφεύγονται μπερδέματα σε περιπτώσεις όπου κάποιοι χρήστες έχουν το ίδιο username-ψευδώνυμο. Η λίστα αυτή με την εισαγωγή ή την αποσύνδεση κάθε χρήστη από την υπηρεσία ανανεώνεται αυτόματα ώστε ο χρήστης να γνωρίζει ποιοι είναι σε πραγματικό χρόνο διαθέσιμοι.

Ο χρήστης όταν το επιθυμεί μπορεί να επιλέξει έναν άλλο χρήστη από τη λίστα και να επικοινωνήσει μαζί του. Στη περίπτωση αυτή ενεργοποιείται ένα νέο παράθυρο όπου εκεί μπορούν να συνομιλήσουν. Το σημαντικό σε αυτή την περίπτωση είναι πως ο καλών χρήστης μπορεί να στέλνει μηνύματα ακόμη και αν ο καλούμενος δεν έχει αποδεχτεί την συνομιλία καθώς μπορεί να είναι απασχολημένος με μια άλλη συνομιλία. Παρόλα' αυτά ο καλών μπορεί να στέλνει μηνύματα στο χρήστη. Σε περίπτωση που ο χρήστης εισέλθει στην συνομιλία στέλνονται μηνύματα αυτόματα και στους δύο συνομιλητές "username:Start Chatting…" και μπορούν να συνομιλήσουν. Αξιοσημείωτο είναι πως ο καλούμενος, σε περίπτωση που ο καλών έχει ήδη αποστείλει μηνύματα, θα τα δει τη στιγμή που θα εισέλθει στην συνομιλία.

Αν ο καλών δε θελήσει να περιμένει μετά από κάποιο χρονικό διάστημα τον συνομιλητή του να εισέλθει στη κουβέντα τότε μπορεί να κλείσει το παράθυρο. Όταν ο καλούμενος θα είναι στην αναμονή της υπηρεσίας, στο παράθυρο με τη λίστα των διαθέσιμων χρηστών, θα δει στο username του χρήστη που του έστελνε μηνύματα, μια ειδοποίηση τύπου /sent message και έτσι όταν θελήσει μπορεί να επιλέξει το χρήστη αυτόν και να δει τι του έστειλε και εν συνεχεία αν ο χρήστης αυτός είναι ακόμη διαθέσιμος να ξεκινήσει μαζί του μια συνομιλία, αλλιώς θα δει στο message log της συνομιλίας στο τέλος ένα μήνυμα "Could not Start Chat" που θα σημάνει ότι ο χρήστης πλέον δεν είναι διαθέσιμος γιατί αποσυνδέθηκε από την υπηρεσία.

Επίσης κατά τη διάρκεια της συνομιλίας μπορεί ένας εκ των ομιλούντων να διακόψει την επικοινωνία κλείνοντας από το παράθυρο. Τότε ο άλλος χρήστης βλέπει στο παράθυρο του ότι έχει απενεργοποιηθεί η δυνατότητα αποστολής άλλου μηνύματος και ταυτόχρονα εμφανίζεται στο πεδίο εισαγωγής μηνύματος προς

αποστολή “Chat Ended…”. Εάν θελήσει να αποστείλει και άλλο μήνυμα θα πρέπει να κλείσει και αυτός το παράθυρο και εν συνεχεία να τον επιλέξει από τη λίστα και να ξεκινήσει μια νέα συνομιλία η απλά να του στείλει το μήνυμα που επιθυμεί και να κλείσει το παράθυρο.

Ο χρήστης ανά πάσα στιγμή μπορεί να βρίσκεται ή σε κατάσταση αναμονής ή να συνομιλεί με έναν άλλο χρήστη. Το σημαντικό είναι πως δε μπορεί να έχει παραπάνω από μία συνομιλία ενεργή, ωστόσο μπορεί να δεχτεί μηνύματα κατά τη διάρκεια που συνομιλεί με κάποιον άλλον από οποιοδήποτε και να επικοινωνήσει μαζί του στη συνέχεια. Δε καταργείται η δυνατότητα πολλαπλών συνομιλιών αλλά λόγο περιορισμένου χώρου στην οθόνη έχει γίνει αυτή η παραδοχή ότι μια συνομιλία θα γίνεται κάθε φορά.

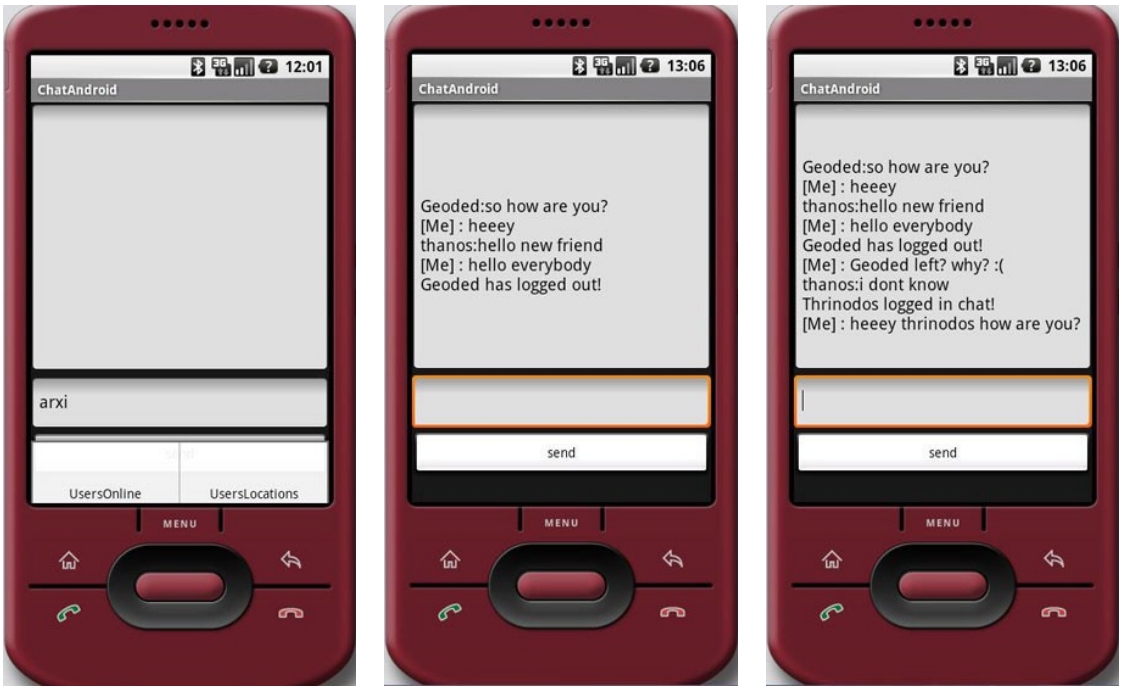
Στις παραπάνω εικόνες βλέπουμε στην πρώτη σειρά εικόνων μια λίστα με διαθέσιμους χρήστες, στη συνέχεια επικοινωνούμε με τον χρήστη Geoded και εν το μεταξύ ο χρήστης thanos επεχείρησε να μας μιλήσει. Όταν κλείσαμε τη συνομιλία με το χρήστη Geoded στην λίστα χρηστών βλέπουμε δίπλα στο όνομα του χρήστη thanos το μήνυμα "/sent message" και όταν επιλέγουμε να επικοινωνήσουμε μαζί του, αρχικά βλέπουμε στην οθόνη τα μηνύματα που μας έστειλε και εν συνέχεια συνεχίζουμε την επικοινωνία μαζί του.

3.5.1.5 Χρήση Public Chat

Ο χρήστης εισερχόμενος σε αυτή την υπηρεσία έχει τη δυνατότητα να βλέπει τα μηνύματα που στέλνουν οι υπόλοιποι χρήστες σε αυτή την υπηρεσία και με τον ίδιο τρόπο σύνδεσης στο σύστημα. Τα μηνύματα αποστέλλονται από το χρήστη προς όλους τους άλλους χωρίς διακρίσεις. Όταν ένας χρήστης εισέρχεται στο chat-room όλοι οι χρήστες ενημερώνονται για την είσοδό του με μήνυμα που εμφανίζεται στο chat log "User George logged in". Αντίστοιχο μήνυμα εμφανίζεται στο chat log όταν ένας χρήστης αποχωρεί: "User George has logged out" .

Προκειμένου ο χρήστης να γνωρίζει ποιος άλλος είναι συνδεδεμένος σε αυτή την υπηρεσία μπορεί να επιλέξει από το menu την επιλογή Online Users και να οδηγηθεί

σε μία λίστα όπου εμφανίζονται όλοι οι χρήστες με το ψευδώνυμό τους και το ονοματεπώνυμό τους.



3.5.1.6 Χρήση Data-Exchange

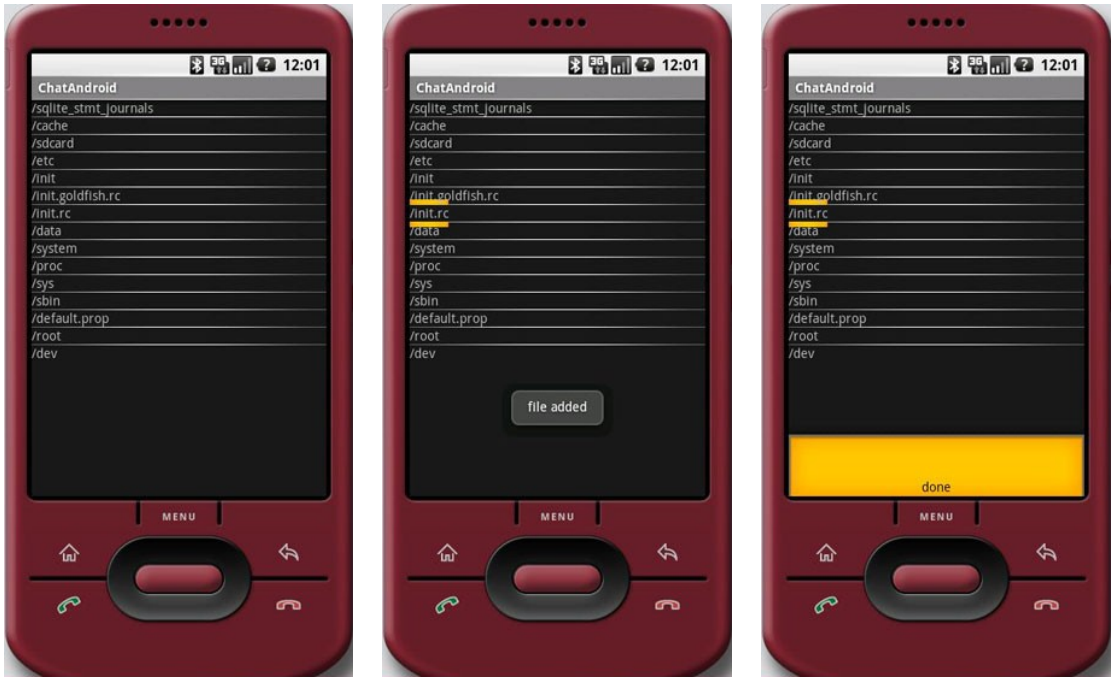
Η υπηρεσία αυτή είναι διαθέσιμη μόνο από τους χρήστες που έχουν επιλέξει να συνδεθούν στο σύστημα με τη χρήση του εξυπηρετητή. Στην υπηρεσία αυτή ο χρήστης έχει τη δυνατότητα να ανταλλάσσει αρχεία με άλλους χρήστες. Αρχικά όταν ο χρήστης επιλέξει αυτή την υπηρεσία προωθείται σε ένα περιβάλλον όπου μπορεί να αναζητήσει μέσα στη συσκευή του τα αρχεία που επιθυμεί να δημοσιεύσει και συνεπώς οι υπόλοιποι χρήστες να «κατεβάσουν» στο κινητό τους. Κάθε φορά που ο χρήστης επιλέγει ένα αρχείο για δημοσίευση, εμφανίζεται ένα στιγμιαίο μήνυμα στο κάτω μέρος της οθόνης όπου ενημερώνει το χρήστη για την ενέργεια του.

Όταν ο χρήστης τελειώσει με τα αρχεία που επιθυμεί να διαμοιράσει, επιλέγει από το menu την επιλογή Done για να συνεχίσει η διαδικασία. Στο νέο παράθυρο που δημιουργείται ο χρήστης παίρνει από τον εξυπηρετητή τα αρχεία που έχουν διαθέσιμα οι υπόλοιποι χρήστες και εμφανίζονται σε μια λίστα.

Κάθε φορά που ο χρήστης επιλέγει από τη λίστα αυτή ένα αρχείο για να κατεβάσει στο κινητό του, επικοινωνεί με τον αντίστοιχο χρήστη και εφόσον ο χρήστης είναι ακόμη online στέλνει το αρχείο και εμφανίζεται στο τέλος της λήψης ένα μήνυμα με την επιτυχή αποθήκευσή του. Αν αποτύχει διότι ο χρήστης δεν

μπορούσε να προσεγγισθεί το αρχείο που θέλαμε να πάρουμε καθώς και οποιοδήποτε άλλο αρχείο που ανήκει στον ίδιο χρήστη διαγράφεται από τη λίστα και δεν είναι πλέον ορατό στο χρήστη. Αν ο χρήστης είναι προσεγγίσιμος αλλά το αρχείο δε μπορούσε να σταλεί τότε εμφανίζεται ένα μήνυμα στο χρήστη που το ζήτησε «not able to receive file…».

Ο χρήστης όσο είναι στην υπηρεσία αυτήν, και βλέπει τη λίστα με τα διαθέσιμα αρχεία για παραλαβή, τότε ο οποιοσδήποτε άλλος χρήστης μπορεί να επικοινωνήσει μαζί του, εφόσον ο πρώτος χρηστής έχει διαθέσιμα αρχεία για αποστολή, για να παραλάβει κάποιο από τα αρχεία που διαθέτει ο πρώτος. Αν για κάποιο λόγο δεν είναι δυνατή η αποστολή ενός αρχείου θα εμφανιστεί στο πρώτο χρήστη ένα μήνυμα «not able to send file…».



Στις παραπάνω εικόνες που παρατίθενται βλέπουμε τη διαδικασία αλλά και τα μηνύματα που εμφανίζονται κατά τη διάρκεια της χρήσης της υπηρεσίας. Αρχικά εμφανίζεται το σύστημα για επιλογή των αρχείων που θέλουμε να διαθέσιμου και αφού πλοηγηθούμε στο φάκελο όπου βρίσκονται τα επιλέγουμε και για κάθε επιλογή εμφανίζεται ένα σύντομο μήνυμα "file added". Εν συνεχεία εμφανίζονται τα αρχεία που είναι διαθέσιμα από άλλους χρήστες και σε περιπτώσεις όπου ένα αρχείο δε μπορούμε να το λάβουμε η δε μπορούμε να το αποστείλουμε η εφαρμογή μας ενημερώνει για την αδυναμία αυτή.

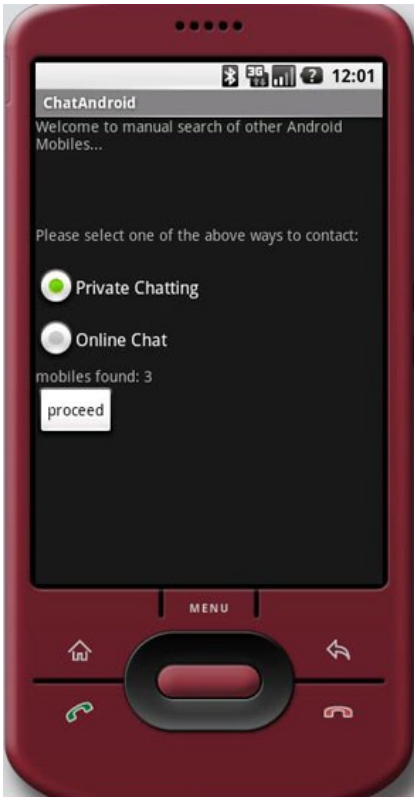
3.5.1.7 Χρήση Auto_Scan

Επιλέγοντας ο χρήστης αυτό τον τρόπο σύνδεσης, ο χρήστης μεταβαίνει σε ένα νέο παράθυρο όπου ενημερώνεται για την επιλογή του και εν συνεχεία του προσφέρονται δύο υπηρεσίες:

- Private chatting
- Online chat

Ο χρήστης αφού επιλέξει πρώτα μια υπηρεσία το σύστημα ψάχνει για άλλους διαθέσιμους χρήστες που έχουν επιλέξει τον ίδιο τρόπο σύνδεσης και την ίδια υπηρεσία και όταν η αναζήτηση τελειώσει, που συνήθως απαιτεί αρκετό χρόνο καθώς

ψάχνει σε όλο το τοπικό δίκτυο, εμφανίζει πόσα τερματικά-κινητά τηλέφωνα έχουν βρεθεί. Στη συνέχεια ο χρήστης μεταβαίνει σε ένα νέο παράθυρο ανάλογα με την υπηρεσία που επέλεξε και έχει ακριβώς τα ίδια χαρακτηριστικά σε εμφάνιση και λειτουργία με τις υπηρεσίες που έχουν οριστεί για τη σύνδεση μέσω εξυπηρετητή-Server με μόνο μερικές διαφορές εσωτερικές που θα αναλυθούν αργότερα.



3.5.1.8 Εμφάνιση χρηστών στο χάρτη

Τέλος μια επιπλέον λειτουργία που προσφέρει η εφαρμογή στους χρήστες είναι η δυνατότητα να βλέπουν στο χάρτη εφόσον το επιθυμούν τη θέση τους και τις θέσεις των άλλων χρηστών ώστε να έχουν μια σχετική εικόνα που βρίσκονται σε σχέση με αυτόν. Η επιλογή αυτή είναι διαθέσιμες σε όλες τις υπηρεσίες όπως φαίνεται από τις εικόνες που ακολουθούν. Είναι σημαντικό να τονίσουμε ότι σε περίπτωση που κάποιος χρήστης δεν είχε τη δυνατότητα να ενημερωθεί με το στίγμα του ή να ενημερώσει την εφαρμογή με την ακριβή του τοποθεσία τότε αυτόματα από την εφαρμογή θεωρείται ως θέση του το κέντρο του Εθνικού Μετσόβιου Πολυτεχνείου. Η κουκκίδα στο χάρτη η οποία στο πάνω μέρος της δε συνοδεύεται από κάποιο όνομα ανήκει στο χρήστη που χρησιμοποιεί την εφαρμογή(δηλαδή σε εμάς) ενώ στις υπόλοιπες κουκκίδες αναφέρονται τα usernames των χρηστών στις οποίες ανήκουν.

3.5.2 Εφαρμογή του Εξυπηρετητή

Η Εφαρμογή του τοπικού εξυπηρετητή-Server ουσιαστικά κάνει πιο εύκολη τη δυνατότητα αναζήτησης άλλων χρηστών που χρησιμοποιούν την ίδια εφαρμογή σε ένα τοπικό δίκτυο με δυνατότητες όπως αυτό των πανεπιστημίων και συγκεκριμένα του Εθνικού Μετσόβιου Πολυτεχνείου. Ο χρήστης μπορεί εύκολα άμεσα και γρήγορα να ενημερωθεί για όλους τους χρήστες που χρησιμοποιούν την ίδια με αυτόν

υπηρεσία σε ένα δίκτυο που είναι μεγαλύτερο από το τοπικό δίκτυο του εκάστοτε wifi access point καθώς στο Auto_Scan δεν είναι δυνατόν από άποψη χρόνου και σπατάλης πόρων του δικτύου να αναζητούμε χρήστες γενικά σε ένα χαώδες δίκτυο. Παρακάτω δίνεται μια εικόνα-στιγμιότυπο του εξυπηρετητή.

Στο παράθυρο αυτό υπάρχει το Log του εξυπηρετητή. Εδώ καταγράφονται όσο ο εξυπηρετητής είναι ενεργός όλες οι συνδιαλέξεις που έχει με τους χρήστες της εφαρμογής. Κάθε μήνυμα αναφέρει αναλυτικά τι ζητά ο χρήστης από τον εξυπηρετητή και φυσικά τις συνδέσεις και αποσυνδέσεις των χρηστών. Όταν επιλέγουμε το κουμπί Start Server ο εξυπηρετητής αρχικά αποκτά σύνδεση με τη βάση δεδομένων mySql και συνδέεται στη βάση με όνομα androidbase. Εφόσον δεν παρουσιαστεί πρόβλημα ο εξυπηρετητής ενεργοποιεί τη δυνατότητα να αποδέχεται συνδέσεις στην πόρτα 8189. Τα μηνύματα που παρουσιάζονται στο Log είναι τα εξής:

- User George Papadimitriou logged in Server as Georgios

  Το μήνυμα αυτό εμφανίζεται όταν ένας χρήστης θέλει να εγγραφεί στον εξυπηρετητή. Εφόσον έχει συμπληρωμένα τα απαιτούμενα πεδία, όνομα, επώνυμο, ψευδώνυμο και διεύθυνση IP τότε καταχωρείται στη βάση δεδομένων και εμφανίζεται το παραπάνω μήνυμα που μας ενημερώνει ποιος χρήστης και με ποιο ψευδώνυμο εισήλθε.
- User Georgios requested the available services

Ο χρήστης που στέλνει αυτό το μήνυμα ζητά από τον εξυπηρετητή εφόσον εγγράφει χωρίς πρόβλημα, τις διαθέσιμες υπηρεσίες και το σύστημα ενημερώνει κάθε φορά το Log με το ποιος ζήτησε την ενημέρωση αυτή.

- User Georgios signed in private chatting

  Το μήνυμα αυτό στο Log υποδηλώνει ότι ο χρήστης με το ψευδώνυμο Georgios αποφάσισε να εισέλθει στην υπηρεσία private chatting και ο Server το επέτρεψε αποστέλλοντας του τους διαθέσιμους χρήστες και εγγράφοντάς τον στη συνέχεια στην υπηρεσία αυτή.

- User Georgios signed in public chat

  Ο χρήστης Georgios ζήτησε να εγγραφεί στην υπηρεσία public chat. Ο εξυπηρετητής αποδέχτηκε το αίτημα του αναφέρει όλους τους χρήστες που είναι εγγεγραμμένοι στην υπηρεσία και εγγραφεί εν συνεχεία τον ίδιο.

- User Georgios reuested uploaded files

  Στη περίπτωση αυτή ο χρήστης Georgios ζητά να ενημερωθεί για τα αρχεία που είναι διαθέσιμα και που μπορεί να τα βρει, δηλαδή ποιοι χρήστες τα έχουν ώστε σε περίπτωση που τα θέλει να μπορέσει να τους τα ζητήσει.

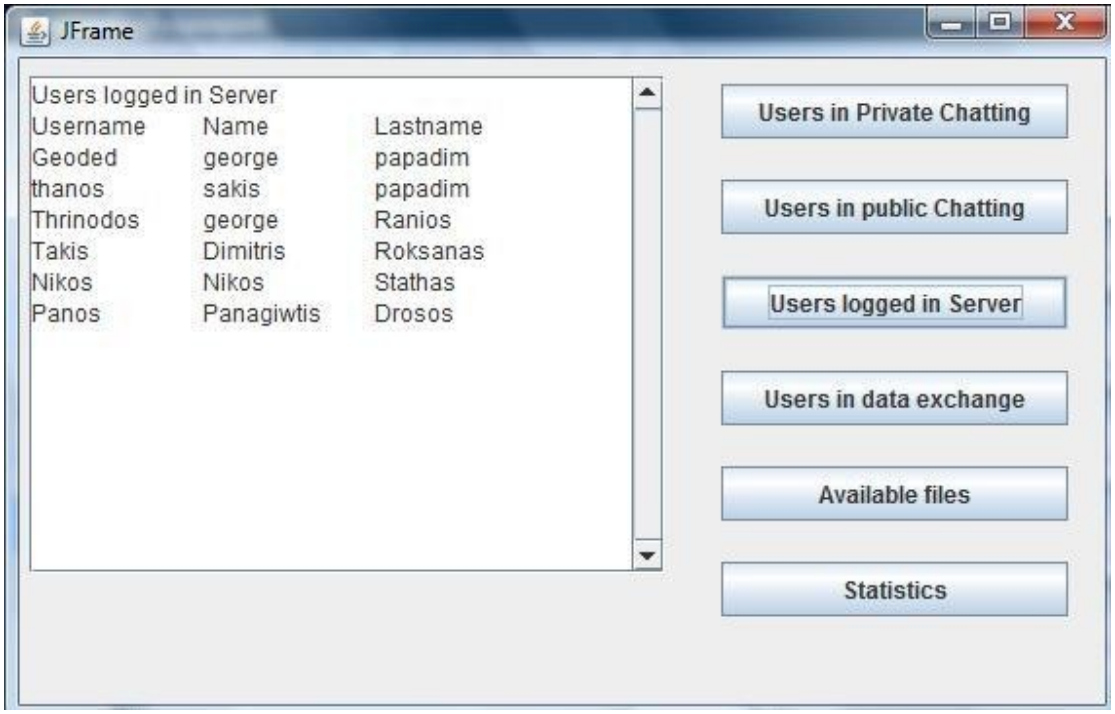
- User Georgios uploaded a new file to spread

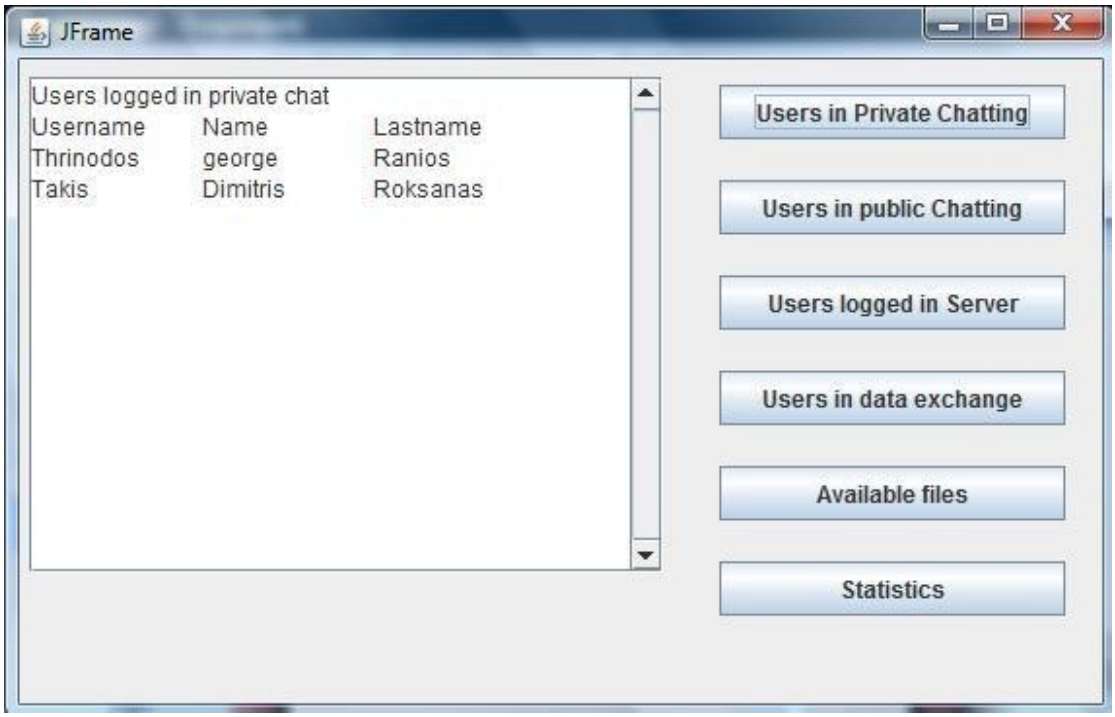
  Ο χρήστης εφόσον θελήσει μπορεί να κάνει γνωστό το αρχείο που θέλει να διαμοιράσει στους υπόλοιπους χρήστες, και ενημερώνει τον εξυπηρετητή ποιο αρχείο είναι αυτό.

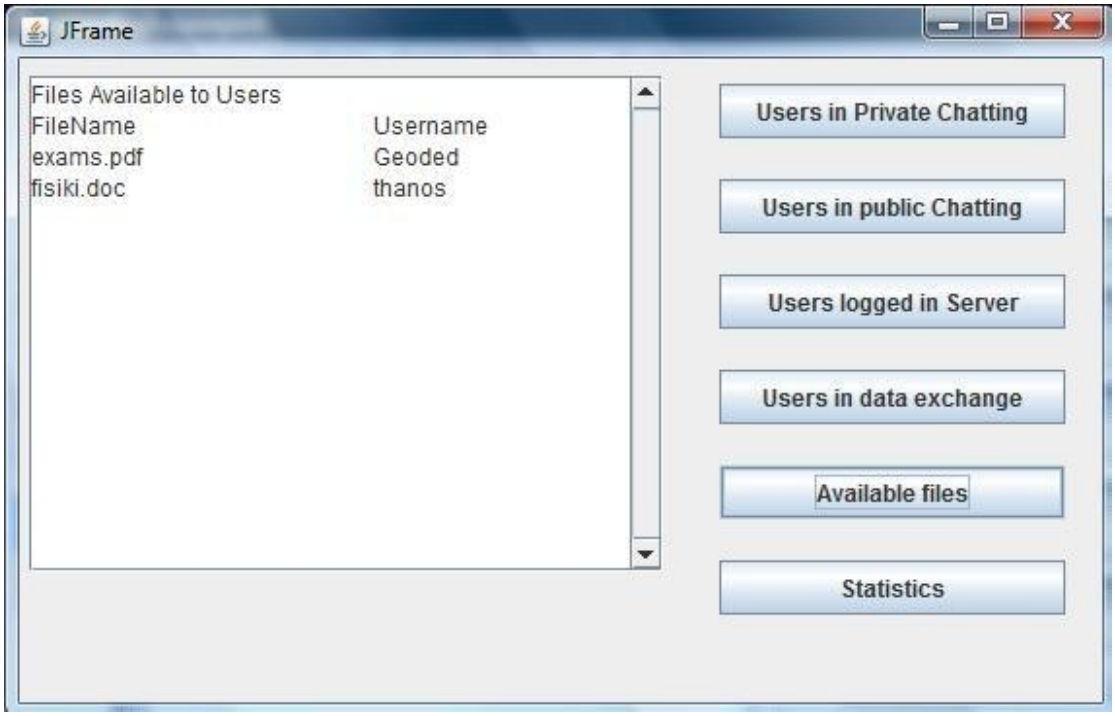
- User Georgios has logout from Server

  Τέλος ο χρήστης Georgios όταν θελήσεις να κλείσει την εφαρμογή ή να βγει από μια υπηρεσία στέλνει ένα μήνυμα στον εξυπηρετητή και αυτός τον «βγάζει» από τη βάση και ότι είναι μαζί του συνδεδεμένο, και τέλος ενημερώνει το Log του με αυτό το μήνυμα.

Ο χρήστης επιλέγοντας το κουμπί Users in Server μπορεί να αποκτήσει μια πιο λεπτομερή αναφορά με το τι κάνουν οι χρήστες αυτή τη στιγμή. Συγκεκριμένα έχουν τη δυνατότητα οι διαχειριστές του εξυπηρετητή να δουν σε κάθε διαθέσιμη υπηρεσία πόσοι είναι συνδεδεμένοι και να δουν κάποια στατιστικά στοιχεία σχετικά με τη χρήση των υπηρεσιών και πως κατανέμονται οι χρήστες στις διάφορες υπηρεσίες.


JFrame
Users logged in Server
Username Name Lastname
Geoded george papadim
thanos sakis papadim
Thrinodos george Ranios
Takis Dimitris Roksanas
Nikos Nikos Stathas
Panos Panagiwtis Drosos
Users in Private Chatting
Users in public Chatting
Users logged in Server
Users in data exchange
Available files
Statistics


JFrame
Users logged in private chat
Username Name Lastname
Thrinodos george Ranios
Takis Dimitris Roksanas
Users in Private Chatting
Users in public Chatting
Users logged in Server
Users in data exchange
Available files
Statistics


JFrame
Files Available to Users
FileName
Username
exams.pdf
Geoded
fisiki.doc
thanos
Users in Private Chatting
Users in public Chatting
Users logged in Server
Users in data exchange
Available files
Statistics


JFrame
Show statistics for Server:
Users logged in: 6
Users logged in private chatting: 2
Users logged in online chat: 2
Total files available: 2
Users in Private Chatting
Users in public Chatting
Users logged in Server
Users in data exchange
Available files
Statistics

4 Υλοποίηση

4.1 Περιγραφή Υλοποίησης

Στο παρών κεφάλαιο ασχολούμαστε με το κυρίως σώμα της εφαρμογής, με τον τρόπο με τον οποίο έχει δομηθεί η εφαρμογή τόσο σε σχέση με τα προτοτυποποιημένα μηνύματα που αποστέλλονται ανάμεσα στα κινητά ο και ανάμεσα στο κινητό και τον εξυπηρετητή όσο και τις βάσεις δεδομένων που δημιουργήθηκαν στη πλευρά του εξυπηρετητή και του τερματικού προκειμένου να αποθηκεύονται τα δεδομένα που απαιτούνται για την λειτουργία της.

4.1.1 Περιγραφή Βάσεων Δεδομένων

4.1.1.1 Βάση Δεδομένων του Εξυπηρετητή

Η εφαρμογή του εξυπηρετητή χρησιμοποιεί μία βάση δεδομένων για την αποθήκευση προσωρινά όσο είναι ενεργοποιημένος ο εξυπηρετητής των προσωπικών δεδομένων των χρηστών καθώς και σε ποιες υπηρεσίες είναι εγγεγραμμένοι, αν αφορά τις υπηρεσίες private chatting και public chats καθώς και ποιων αρχείων έχουνε αποφασίσει οι χρήστες να διαμοιράσουν στους υπόλοιπους χρήστες. Τέλος χρησιμοποιείται και για την αποθήκευση των υπηρεσιών που το σύστημα εξυπηρετητής και κινητό τηλέφωνο μπορεί να εξυπηρετήσει-υποστηρίξει.

Οι οντότητες και σχέσεις που περιγράφουν τη βάση δεδομένων είναι η εξής:

- Η οντότητα users αντιπροσωπεύει τους χρήστες που είναι συνδεδεμένος στο σύστημα και έχει τις ακόλουθες ιδιότητες:
    - userID ένα μοναδικό κλειδί για κάθε χρήστη
    - username το ψευδώνυμο του χρήστη
    - name το όνομα του χρήστη
    - lastname το επώνυμο του χρήστη
    - ip την διεύθυνση ip του χρήστη
    - coordinateX την πρώτη συντεταγμένη του χρήστη
    - coordinateY την δεύτερη συντεταγμένη του χρήστη

- Η οντότητα services περιέχει τις διαθέσιμες υπηρεσίες που υποστηρίζει το σύστημα:
    - servicesID ένα μοναδικό κλειδί για κάθε υπηρεσία
    - name το όνομα κάθε υπηρεσίας

- Η οντότητα chatting περιέχει τους χρήστες οι οποίοι είναι εγγεγραμμένοι στην υπηρεσία private chatting:
    - userID ένα μοναδικό ξένο-κλειδί για κάθε χρήστη

- Η οντότητα onlinechat περιέχει τους χρήστες οι οποίοι είναι εγγεγραμμένοι στην υπηρεσία public chat:
    - userID ένα μοναδικό ξένο-κλειδί για κάθε χρήστη

- Η οντότητα dataexchange περιέχει τους χρήστες οι οποίοι έχουν αποφασίσει και διαθέτουν στους υπόλοιπους χρήστες :
    - userID ένα μοναδικό ξένο-κλειδί για κάθε χρήστη
    - file το μονοπάτι σε κάθε κινητό τηλέφωνο όπου βρίσκεται το κάθε αρχείο

4.1.1.2 Βάση Δεδομένων του τεραμτικού

Από την πλευρά του τερματικού-κινητό τηλέφωνο η βάση δεδομένων χρησιμοποιείται για την αποθήκευση των στοιχείων του χρήστη καθώς η αποθήκευση σε xml αρχεία δεν επιτρέπεται από το λογισμικό καθώς τα xml αρχεία είναι βασικά συστατικά της δομής των εφαρμογών και για αποφυγή κακόβουλων παρεμβάσεων έχει απενεργοποιηθεί αυτή η δυνατότητα. Η βάση δεδομένων χρησιμοποιείται επίσης για την προσωρινή αποθήκευση των στοιχείων των χρηστών που είναι εγγεγραμμένοι σε μια υπηρεσία ώστε να είναι διαθέσιμα άμεσα τόσο σε περίπτωση ανάγκης επικοινωνίας ή ανταλλαγής αρχείων ή για εμφάνιση στο χάρτη της τοποθεσίας τους. Τέλος η χρήση της βάσης είναι αναγκαία και στην περίπτωση που ο χρήστης επιλέγει να χρησιμοποιήσει την υπηρεσία ανταλλαγής αρχείων όπου απαιτείται να αποθηκεύουμε τα αρχεία που του είναι διαθέσιμα από άλλους χρήστες.

Οι οντότητες και σχέσεις που περιγράφουν τη βάση δεδομένων είναι η εξής:

- Η οντότητα users περιέχει τους χρήστες που είναι κάθε φορά εγγεγραμμένοι σε μια υπηρεσία:
  - _ID ένα μοναδικό κλειδί για κάθε χρήστη
  - username το ψευδώνυμο του χρήστη
  - name το όνομα του χρήστη
  - lastname το επώνυμο του χρήστη
  - ip την διεύθυνση ip του χρήστη
  - coordinateX την πρώτη συντεταγμένη του χρήστη
  - coordinateY την δεύτερη συντεταγμένη του χρήστη
  - messages πεδίο για την αποθήκευση των μηνυμάτων που στέλνει ο χρήστης αυτός και δεν είναι δυνατόν να τα δει άμεσα ο χρήστης που τα δέχεται

- Η οντότητα userinfo περιέχει τα στοιχεία του χρήστη που χρησιμοποιεί το κινητό τηλέφωνο και την εφαρμογή:
  - _ID ένα μοναδικό κλειδί για τη κάθε καταχώρηση
  - username το ψευδώνυμο που χρησιμοποιεί στις υπηρεσίες ο χρήστης
  - name το όνομα του χρήστη
  - lastname το επώνυμο του χρήστη
  - ip η διεύθυνση ip του χρήστη

- Η οντότητα data περιέχει τα στοιχεία των αρχείων που είναι διαθέσιμα για "κατέβασμα":
  - _ID ένα μοναδικό κλειδί για κάθε αρχείο
  - name το όνομα του αρχείου
  - path το μονοπάτι μέσω του οποίου βρίσκεται το αρχείο στο εκάστοτε κινητό τηλέφωνο που το έχει προς διάθεση
  - ip η διεύθυνση στην οποία μπορεί να βρεθεί το αρχείο

4.2 Πρότυπα μηνύματα-πρωτόκολλα Επικοινωνίας

Στη παράγραφο αυτή γίνεται μια παρουσίαση των μηνυμάτων που ανταλλάσσονται μεταξύ του εξυπηρετητή και του κινητού τηλεφώνου καθώς και ανάμεσα στα κινητά τηλέφωνα. Τα μηνύματα αυτά αποστέλλονται χωρίς κρυπτογράφηση και είναι αναγκαία για την ορθή λειτουργία του συστήματος της εφαρμογής καθώς και για την επικοινωνία των χρηστών μεταξύ τους.

4.2.1 Μηνύματα μεταξύ εξυπηρετητή κινητού τηλεφώνου

Παρακάτω περιγράφονται τα μηνύματα και η σειρά με την οποία αυτά ανταλλάσσονται μεταξύ του εξυπηρετητή και του κινητού τηλεφώνου-χρήστη ώστε αυτός να εγγραφεί στο σύστημα, να ενημερωθεί για τις διαθέσιμες υπηρεσίες, να εγγραφεί σε μία από αυτές και να ενημερωθεί για τους χρήστες που είναι εγγεγραμμένοι σε αυτήν ή αν πρόκειται για την υπηρεσία ανταλλαγής δεδομένων-αρχείων να ενημερωθεί για τα αρχεία που είναι διαθέσιμα προς download.

4.2.1.1 Εγγραφή χρήστη στον εξυπηρετητή

Αρχικά ο χρήστης εγκαθιστά μια επικοινωνία με TCP connection στην IP Address η οποία φιλοξενεί τον εξυπηρετητή και στη πόρτα 8189 στην οποία “ακούει” ο εξυπηρετητής. Εν συνεχεία εφόσον η επικοινωνία εγκαθίσταται τότε ο χρήστης στέλνει ένα μήνυμα τύπου String με την εξής δομή για να μπορέσει αν εγγραφεί στο σύστημα:

login	username	name	lastname	IP Address	Latitude	Longtitude

Το παραπάνω μήνυμα όπως βλέπουμε αποτελείται από 7 επιμέρους String-λέξεις. Κάθε String-λέξη διαχωρίζεται από την επόμενη αφήνοντας μεταξύ τους ένα κενό διάστημα. Η πρώτη λέξη είναι “login”, και υποδηλώνει στον εξυπηρετητή ότι ο χρήστης προτίθεται να συνδεθεί στο σύστημα και να εγγραφεί σε αυτό και στο ίδιο μήνυμα παραθέτονται τα προσωπικά στοιχεία του καθώς και την διεύθυνση IP Address η οποία έχει αποδοθεί από το δίκτυο στο κινητό τηλέφωνο κατά την παρούσα σύνδεση. Τέλος το μήνυμα αυτό ολοκληρώνεται με την παροχή των δύο γεωγραφικών συντεταγμένων, το στίγμα δηλαδή του χρήστη ώστε να είναι δυνατόν οι άλλοι χρήστες όταν το επιθυμούν να βλέπουν στο χάρτη που βρίσκονται οι άλλοι

χρήστες. Μόλις ο εξυπηρετητής παραλάβει το μήνυμα αυτό ξεκινά τη διαδικασία εγγραφής του χρήστη στο σύστημα και εν συνεχεία μόλις ολοκληρωθεί η εγγραφή του στέλνει μήνυμα:

login_OK

Η εφαρμογή όταν παραλάβει το μήνυμα αυτό τότε γνωρίζει ότι η εγγραφή στο σύστημα έγινε επιτυχώς και μπορεί να συνεχίσει στο επόμενο βήμα για τη συλλογή πληροφοριών για τις διαθέσιμες υπηρεσίες και τέλος την εγγραφή σε μία από αυτές και την ενημέρωση της για τα δεδομένα που χρειάζεται.

4.2.1.2 Ενημέρωση διαθέσιμων υπηρεσιών

Ο χρήστης ύστερα από την επιτυχή εγγραφή του στο σύστημα, πρέπει να ενημερωθεί για τις διαθέσιμες υπηρεσίες. Συνεπώς στέλνει το ακόλουθο μήνυμα στον εξυπηρετητή:

services_request

Ο εξυπηρετητής όταν λάβει το μήνυμα αυτό, αναζητά στη βάση δεδομένων του τις διαθέσιμες υπηρεσίες που υπάρχουν και εν συνεχεία στέλνει τα ακόλουθα μηνύματα. Αρχικά στέλνει ένα μήνυμα με έναν αριθμό, οποίος υποδηλώνει το σύνολο των υπηρεσιών που είναι διαθέσιμες για το χρήστη, ώστε να γνωρίζει πόσα μηνύματα θα δεχτεί που αφορούν την περιγραφή των υπηρεσιών τις οποίες μπορεί να χρησιμοποιήσει:

counter

Αν ο αριθμός αυτός,το counter, είναι ίσο με μηδέν (0),τότε η εφαρμογή δεν αναμένει μηνύματα που να αφορούν περιγραφή των υπηρεσιών, αφού δεν υπάρχουν

και ενημερώνει τον χρήστη μέσω του γραφικού περιβάλλοντος-οθόνη ότι δεν υπάρχουν διαθέσιμες υπηρεσίες. Συγκεκριμένα εμφανίζεται το μήνυμα:

“No services available”

Σε περίπτωση που δεν ισχύει η παραπάνω περίπτωση, δηλαδή ο αριθμός, counter, είναι μεγαλύτερος ή ίσος με ένα (1) τότε ο εξυπηρετητής στέλνει συνεχόμενα μηνύματα, που το κάθε ένα περιγράφει ποία υπηρεσία είναι διαθέσιμη από τον εξυπηρετητή και η εφαρμογή στο τερματικό την εμφανίζει στην οθόνη του κινητού:

num service_name

Τα μηνύματα αυτά είναι ένα απλό String-λέξη που αποτελούνται από έναν αριθμό στην αρχή, που είναι η αρίθμηση της υπηρεσίας, από ένα κενό διάστημα και τέλος το όνομα της υπηρεσίας. Με την παραλαβή του κάθε μηνύματος η εφαρμογή εμφανίζει στην οθόνη την διαθέσιμη υπηρεσία.

4.2.1.3 Εγγραφή σε υπηρεσία

Στη παράγραφο αυτή περιγράφουμε τα μηνύματα που ανταλλάσσονται μεταξύ του εξυπηρετητή και του τερματικού προκειμένου ο χρήστης να μπορέσει να ενημερωθεί για τους χρήστες ή τα αρχεία που είναι διαθέσιμα από άλλους χρήστες με βάση την υπηρεσία που έχει επιλέξει ο χρήστης. Λόγω ομοιότητας της διαδικασίας στις υπηρεσίες private chat και public chat θα περιγραφούν μια φορά, ενώ η υπηρεσία ανταλλαγής αρχείων ξεχωριστά.

4.2.1.3.1 Εγγραφή σε private chat & public chat

Ο χρήστης ύστερα από την εμφάνιση στην οθόνη των διαθέσιμων υπηρεσιών έχει τη δυνατότητα να επιλέξει μια υπηρεσία και αν εγγραφεί σε αυτήν καθώς και αν ενημερωθεί για τους χρήστες που συμμετέχουν.

Η εφαρμογή, όταν ο χρήστης επιλέγει να εγγραφεί στην υπηρεσία private chat στέλνει στον εξυπηρετητή το ακόλουθο μήνυμα:

chatting_login

Στην περίπτωση που ο χρήστης επιλέξει την υπηρεσία online chat τότε στέλνεται το ακόλουθο μήνυμα:

onlinechat_login

Όταν ο εξυπηρετητής παραλάβει τα παραπάνω μηνύματα, ένα από τα δύο σε κάθε σύνδεση με κάποιο χρήστη, τότε ενημερώνεται με βάση το μήνυμα που παρέλαβε πόσοι χρήστες είναι εγγεγραμμένοι στην αντίστοιχη υπηρεσία, μέσω της βάσης δεδομένων του εξυπηρετητή και αποστέλλει ένα νέο μήνυμα που περιλαμβάνει τον αριθμό των χρηστών σε αυτήν την υπηρεσία, ώστε να προετοιμαστεί η εφαρμογή στην πλευρά του χρήστη να δεχτεί τα αντίστοιχα μηνύματα που περιέχουν τα προσωπικά στοιχεία και πληροφορίες θέσης για τους άλλους χρήστες.

counter

Σε περίπτωση που ο αριθμός που στάλθηκε είναι ίσος με μηδέν (0), τότε η διαδικασία δε προχωρά άλλο καθώς είναι κατανοητό πως δεν υπάρχουν εγγεγραμμένοι χρήστες στην υπηρεσία αυτή, είτε πρόκειται για την υπηρεσία private chat είτε για την public chat.

Σε αντίθετη περίπτωση τότε, ύστερα από το παραπάνω μήνυμα που ενημερώνει την εφαρμογή στο τερματικό πόσα μηνύματα με πληροφορίες χριστών θα δεχτεί στέλνει μια σειρά από μηνύματα ο εξυπηρετητής με το ακόλουθο μοτίβο:

username	name	lastname	IP Address	Latitude	Longtitude

Τα μηνύματα αυτά περιέχουν τα προσωπικά στοιχεία του χρήστη, όνομα και επώνυμο, καθώς και το ψευδώνυμο που χρησιμοποιεί στην διάρκεια της εφαρμογής και ειδικότερα κατά τη διάρκεια συμμετοχής στις υπηρεσίες καθώς και την IP Address που του έχει δοθεί από το ασύρματο δίκτυο που τον φιλοξενεί και τέλος τις γεωγραφικές συντεταγμένες του για χρήση από τους άλλους χρήστες.

4.2.1.3.2 Εγγραφή στην υπηρεσία ανταλλαγής δεδομένων

Τέλος η τρίτη υλοποιημένη υπηρεσία στην εφαρμογή αυτή, είναι η ανταλλαγή αρχείων μεταξύ των χρηστών και η υπηρεσία αυτή παρέχεται εφόσον ο χρήστης επιλέξει να συνδεθεί με τον εξυπηρετητή. Ο χρήστης επιλέγοντας να χρησιμοποιήσει την υπηρεσία αυτή, για ένα διάστημα αποσυνδέεται από τον εξυπηρετητή, κάνει logout και εν συνεχεία σε επόμενο βήμα ξανά-συνδέεται ώστε να συνεχίσει η εγγραφή του κανονικά. Συνεπώς στέλνει ένα μήνυμα το τερματικό-κινητό τηλέφωνο στον εξυπηρετητή που τον ενημερώνει για αυτή την ενέργεια. Το μήνυμα που αποστέλλεται έχει τη παρακάτω δομή:

logout	username	lastname

Το μήνυμα αυτό απαρτίζεται από τρία επιμέρους String-λέξεις οι οποίες χωρίζονται μεταξύ τους με ένα κενό διάστημα. Η πρώτη λέξη “logout” υποδηλώνει ότι ο χρήστης προτίθεται να αποσυνδεθεί από το σύστημα και εν συνεχεία για επιβεβαίωση ποιος χρήστης είναι που στέλνει το μήνυμα αποστέλλονται για ταυτοποίηση το ψευδώνυμο και το επώνυμό του.

Ο χρήστης για ένα διάστημα επιλέγει τα αρχεία που θέλει να διαμοιράσει και εν συνεχεία μεταβαίνει στο επόμενο βήμα για να ολοκληρωθεί η διαδικασία εγγραφής στην υπηρεσία ανταλλαγής δεδομένων-αρχείων. Αρχικά η εφαρμογή στο κινητό τηλέφωνο ξεκινά εκ νέου την εγγραφή στο σύστημα, όπως περιγράφηκε παραπάνω. Όταν λάβει το μήνυμα επιβεβαίωσης εγγραφής, “login_OK” τότε στέλνει ένα νέο μήνυμα με περιεχόμενο “dataexchange_request”:

dataexchange_request

Ο εξυπηρετητής δεχόμενος αυτό το μήνυμα αντιλαμβάνεται την πρόθεση του χρήστη και αναζητεί στην βάση δεδομένων του αν υπάρχουν διαθέσιμα αρχεία προς

download από χρήστες και στέλνει έναν αριθμό που δηλώνει το πλήθος αυτών των αρχείων:

counter

Αν ο αριθμός αυτός δεν είναι μηδέν (0), τότε η διαδικασία, συνεχίζεται στέλνοντας εν συνεχεία ο εξυπηρετητής στον χρήστη συνεχόμενα μηνύματα με την εξής δομή:

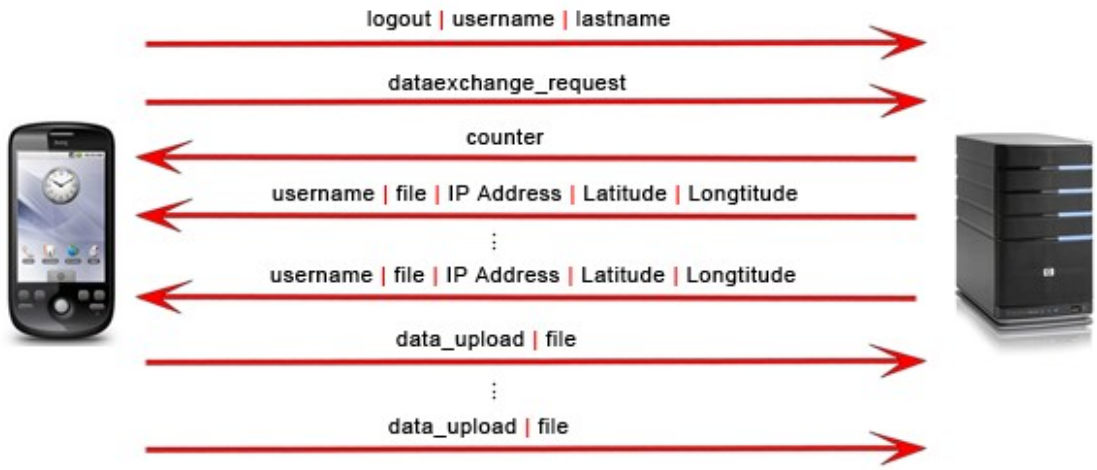
username	file	IP Address	Longtitude	Latitude

Όπως φαίνεται από το παραπάνω υπόμνημα το μήνυμα αυτό αποτελείται από πέντε επιμέρους στοιχεία-λέξεις και χωρίζονται μεταξύ τους με ένα κενό διάστημα. Η πρώτη λέξη αφορά στο ψευδώνυμο του χρήστη ο οποίος παρέχει το αρχείο, η δεύτερη το μονοπάτι και το όνομα του αρχείου που είναι προς διάθεση από τον χρήστη αυτόν, την IP Address του ώστε να μπορούμε να απευθυνθούμε σε αυτόν και να πάρουμε το αρχείο και τέλος από τις δύο γεωγραφικές συντεταγμένες. Είναι προφανές ότι κάθε χρήστης μπορεί να έχει προς διάθεση παραπάνω από ένα αρχεία και συνεπώς θα αποστέλλονται πολλαπλά τέτοια μηνύματα που θα διαφέρει κάθε φορά μόνο το σημείο-λέξη με την ονομασία του αρχείου “file”.

Όταν τελειώσει η παραπάνω διαδικασία, τότε εφόσον ο χρήστης έχει επιλέξει από προηγούμενο βήμα να διαθέσει και αυτός αρχεία στους άλλους χρήστες τότε αρχικά ενημερώνει τον εξυπηρετητή στέλνοντας μηνύματα με την εξής δομή:

data_upload	file

Αυτά τα μηνύματα μπορεί να είναι παραπάνω από ένα, δηλαδή να είναι τόσα όσα και τα αρχεία που έχει ο χρήστης προς διάθεση. Το μήνυμα αποτελείται ουσιαστικά από δύο επιμέρους λέξεις, πρώτα υπάρχει η λέξη “data_upload” όπου ο εξυπηρετητής αντιλαμβάνεται πως ο συγκεκριμένος χρήστης έχει σκοπό να διαθέσει ένα αρχείο και στη δεύτερη λέξη δίνει το μονοπάτι μέσα στο κινητό όπου υπάρχει το αρχείο μαζί με το όνομα του αρχείου.



4.2.1.3.3 Διαγραφή από υπηρεσία

Τέλος η τελευταία συνδιαλλαγή που έχουνε ο εξυπηρετητής και η εφαρμογή του τερματικού-κινητό τηλέφωνο είναι η διαδικασία της διαγραφής από μία υπηρεσία η την διαγραφή γενικότερα από το σύστημα αν ο χρήστης δεν έχει εγγραφεί σε κάποια διαθέσιμη υπηρεσία. Το μήνυμα που αποστέλλει η εφαρμογή του χρήστη έχει την εξής δομή:

logout	username	lastname

Το μήνυμα αυτό απαρτίζεται από τρία επιμέρους String-λέξεις οι οποίες χωρίζονται μεταξύ τους με ένα κενό διάστημα. Η πρώτη λέξη “logout” υποδηλώνει ότι ο χρήστης προτίθεται να αποσυνδεθεί από το σύστημα και εν συνεχεία για επιβεβαίωση ποιος χρήστης είναι που στέλνει το μήνυμα αποστέλλονται για ταυτοποίηση το ψευδώνυμο και το επώνυμό του.

4.2.2 Μηνύματα μεταξύ κινητών τηλεφώνων

Στη παράγραφο αυτή περιγράφουμε τα μηνύματα που ανταλλάσσονται μεταξύ των χρηστών ύστερα από πρωτοβουλία των χρηστών,όπως όταν θέλουν να συνομιλήσουν με άλλους χρήστες,είτε μηνύματα που η εφαρμογή αυτόματα στέλνει όταν συγκεκριμένα γεγονότα συμβαίνουν.

4.2.2.1 Ενημέρωση χρηστών για νέο χρήστη

Στην περίπτωση που ένας χρήστης εισέρχεται στην υπηρεσία και έχει ενημερωθεί για τους υπόλοιπους χρήστες που συμμετέχουν στην ίδια υπηρεσία, πρέπει να ειδοποιήσει τους χρήστες αυτούς για την είσοδό του στην υπηρεσία.

4.2.2.1.1 Εισαγωγή χρήστη στην υπηρεσία private chat

Στην υπηρεσία αυτή, είτε ο χρήστης έχει χρησιμοποιήσει τον εξυπηρετητή και συνεπώς έχει εγγραφεί σε αυτόν είτε έχει χρησιμοποιήσει την αυτόματη αναζήτηση αποστέλλει το παρακάτω μήνυμα σε κάθε έναν από αυτούς ξεχωριστά εφόσον πρώτα εγκαταστήσει μια TCP connection, εφόσον πρώτα οι χρήστες που είναι ήδη συνδεδεμένοι έχουν αποστείλει το παρακάτω μήνυμα:

cause_of_talk?

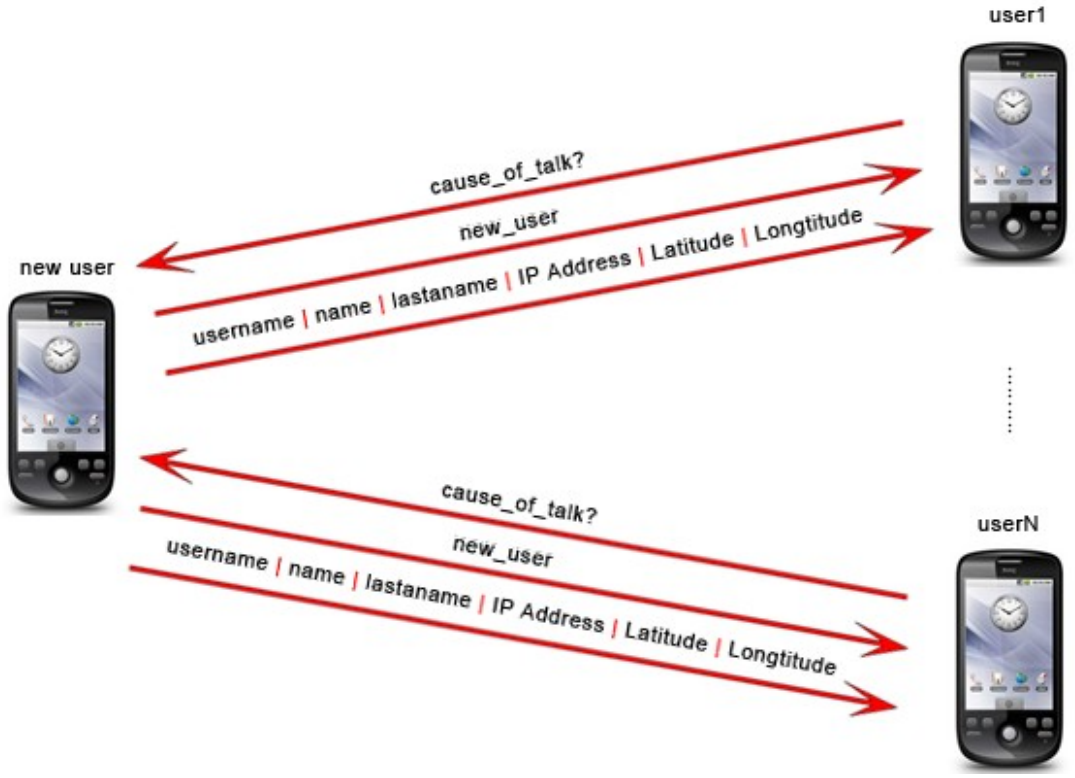
Το παραπάνω μήνυμα προέρχεται όπως ειπώθηκε από τους χρήστες που είναι ήδη συνδεδεμένοι στην υπηρεσία ενώ μετά από αυτό το μήνυμα ο χρήστης προς εγγραφή στην υπηρεσία στέλνει τα ακόλουθα μηνύματα:

new_user

Το παραπάνω μήνυμα προετοιμάζει την εφαρμογή των χρηστών που το λαμβάνουν ότι πρόκειται ένας νέος χρήστης να συνδεθεί στην υπηρεσία και θα στείλει σε επόμενο μήνυμα τα προσωπικά του στοιχεία και επιπλέον χρήσιμες πληροφορίες όπως την IP Address και τις γεωγραφικές συντεταγμένες. Ύστερα από αυτό το μήνυμα στέλνει τα στοιχεία του χρήστη ώστε να είναι διαθέσιμα, στους υπάρχοντες χρήστες της υπηρεσίας:

username	name	lastname	IP Address	Latitude	Longtitude

Με την αποστολή των παραπάνω μηνυμάτων σε κάθε χρήστη οι συνδέσεις τερματίζονται και η ενημέρωση για την είσοδο του νέου χρήστη έχει πραγματοποιηθεί.



4.2.2.1.2 Εισαγωγή χρήστη στην υπηρεσία public chat

Στην υπηρεσία αυτή, είτε ο χρήστης έχει χρησιμοποιήσει τον εξυπηρετητή και συνεπώς έχει εγγραφεί σε αυτόν είτε έχει χρησιμοποιήσει την αυτόματη αναζήτηση αποστέλλει το παρακάτω μήνυμα σε κάθε έναν από αυτούς ξεχωριστά εφόσον πρώτα εγκαταστήσει μια TCP connection, εφόσον πρώτα ο χρήστης που είναι ήδη συνδεδεμένος έχει αποστείλει το παρακάτω μήνυμα:

Send...

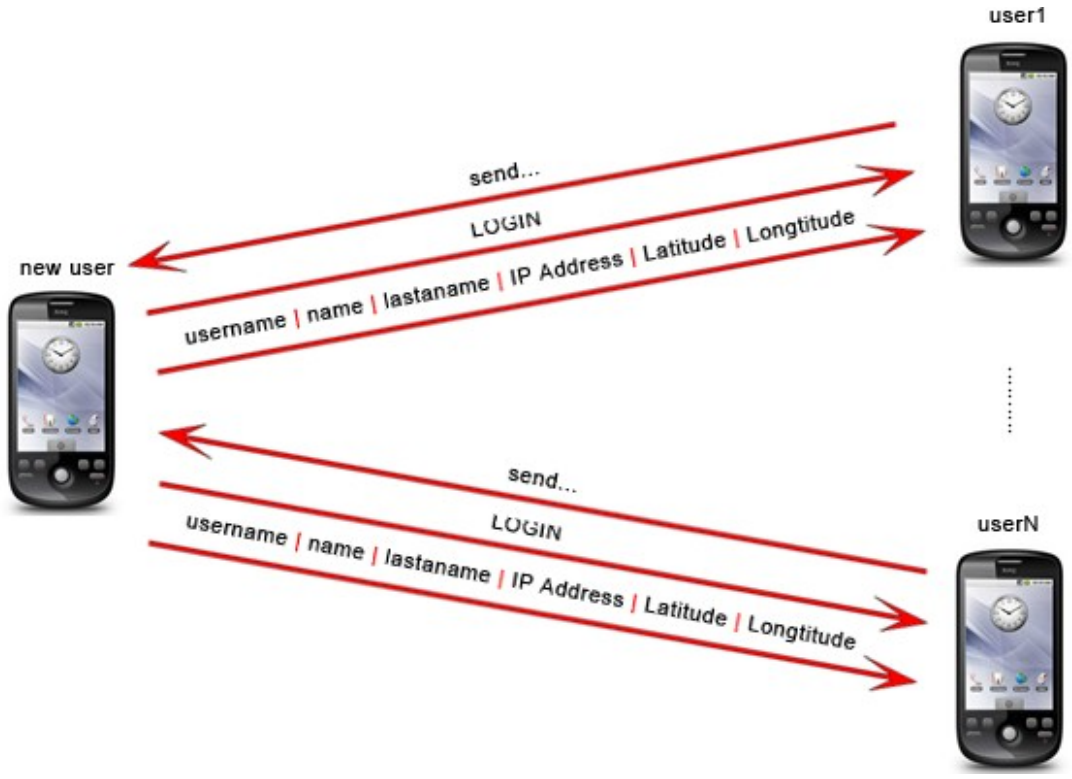
Το παραπάνω μήνυμα προέρχεται όπως ειπώθηκε από τους χρήστες που είναι ήδη συνδεδεμένοι στην υπηρεσία ενώ μετά από αυτό το μήνυμα ο χρήστης προς εγγραφή στην υπηρεσία στέλνει τα ακόλουθα μηνύματα:

LOGIN

Το παραπάνω μήνυμα προετοιμάζει την εφαρμογή των χρηστών που το λαμβάνουν ότι πρόκειται ένας νέος χρήστης να συνδεθεί στην υπηρεσία και θα στείλει σε επόμενο μήνυμα τα προσωπικά του στοιχεία και επιπλέον χρήσιμες πληροφορίες όπως την IP Address και τις γεωγραφικές συντεταγμένες. Ύστερα από αυτό το μήνυμα στέλνει τα στοιχεία του χρήστη ώστε να είναι διαθέσιμα, στους υπάρχοντες χρήστες της υπηρεσίας:

username	name	lastname	IP Address	Latitude	Longtitude

Με την αποστολή των παραπάνω μηνυμάτων σε κάθε χρήστη οι συνδέσεις τερματίζονται και η ενημέρωση για την είσοδο του νέου χρήστη έχει πραγματοποιηθεί.



4.2.2.2 Ενημέρωση χρηστών για νέα διαθέσιμα αρχεία

Στην υπηρεσία αυτή όταν ένας νέος χρήστης επιθυμεί να διαθέσει κάποια αρχεία σε άλλου χρήστες εκτός από τον εξυπηρετητή που έχει ενημερώσει, θα πρέπει ν ενημερώσει τους υπάρχοντες χρήστες για τα αρχεία αυτά που έχει προς διάθεση. Γνωρίζοντας της διευθύνσεις IP Address του καθενός, πραγματοποιεί συνδέσεις TCP

connection προς τον καθένα και ανταλλάσσουν τα παρακάτω μηνύματα. Οι χρήστες που είναι ήδη συνδεδεμένοι στην υπηρεσία αποστέλλουν αρχικά το παρακάτω μήνυμα:

causeofcontact?

Ο νέος χρήστης τότε στέλνει το ακόλουθο μήνυμα:

new_user

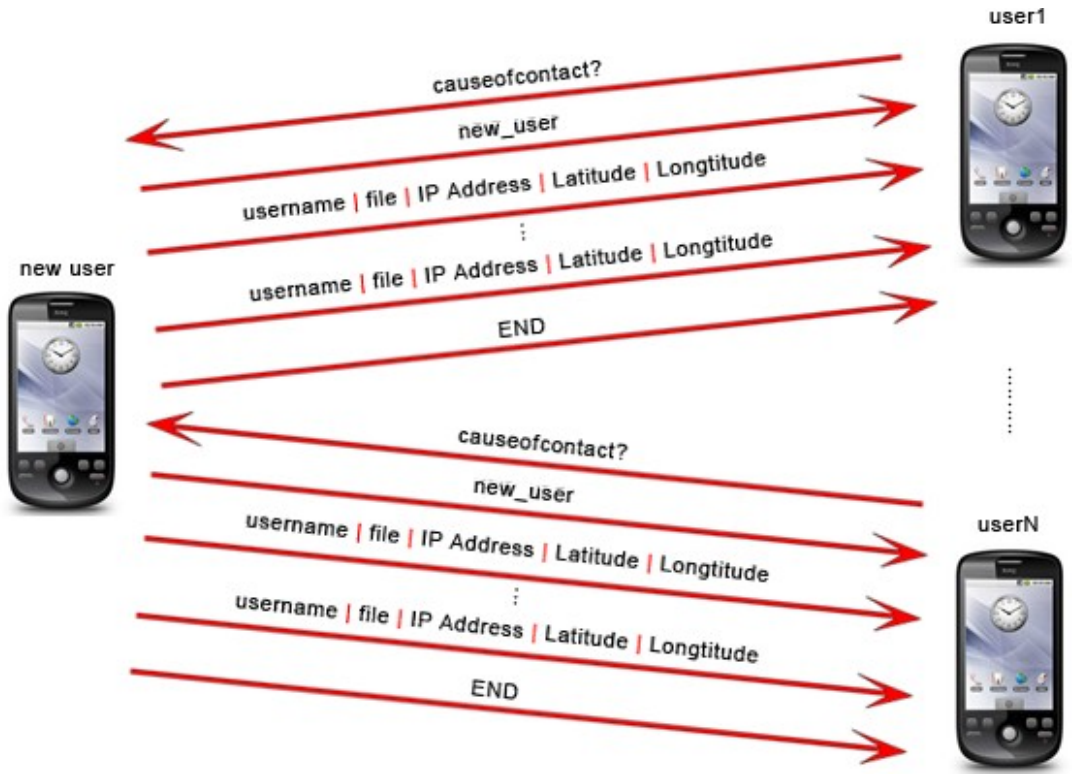
Το παραπάνω μήνυμα προετοιμάζει την εφαρμογή των χρηστών που το λαμβάνουν ότι πρόκειται ένας νέος χρήστης να συνδεθεί στην υπηρεσία και θα στείλει σε συνεχόμενα μηνύματα το ψευδώνυμό του το μονοπάτι με το όνομα του αρχείου μαζί, την IP Address και τις γεωγραφικές συντεταγμένες:

username	file	IP Address	Latitude	Longtitude

Όταν ο χρήστης δεν έχει να στείλει άλλα μηνύματα που να ενημερώνουν με τα διαθέσιμα αρχεία τότε στέλνει ένα ακόμα μήνυμα με το εξής περιεχόμενο:

END

Με την αποστολή του παραπάνω μηνύματος η σύνδεση με τον κάθε χρήστη τερματίζεται και όλοι οι χρήστες πλέον είναι ενημερωμένοι για τα νέα διαθέσιμα αρχεία.



4.2.2.3 Ενημέρωση χρηστών για αποχώρηση χρήστη

4.2.2.3.1 Εξαγωγή χρήστη από την υπηρεσία private chat

Στην υπηρεσία αυτή, είτε ο χρήστης έχει χρησιμοποιήσει τον εξυπηρετητή και συνεπώς έχει εγγραφεί σε αυτόν είτε έχει χρησιμοποιήσει την αυτόματη αναζήτηση αποστέλλει το παρακάτω μήνυμα σε κάθε έναν από αυτούς ξεχωριστά εφόσον πρώτα εγκαταστήσει μια TCP connection, εφόσον πρώτα οι χρήστες που είναι ήδη συνδεδεμένοι έχουν αποστείλει το παρακάτω μήνυμα:

cause_of_talk?

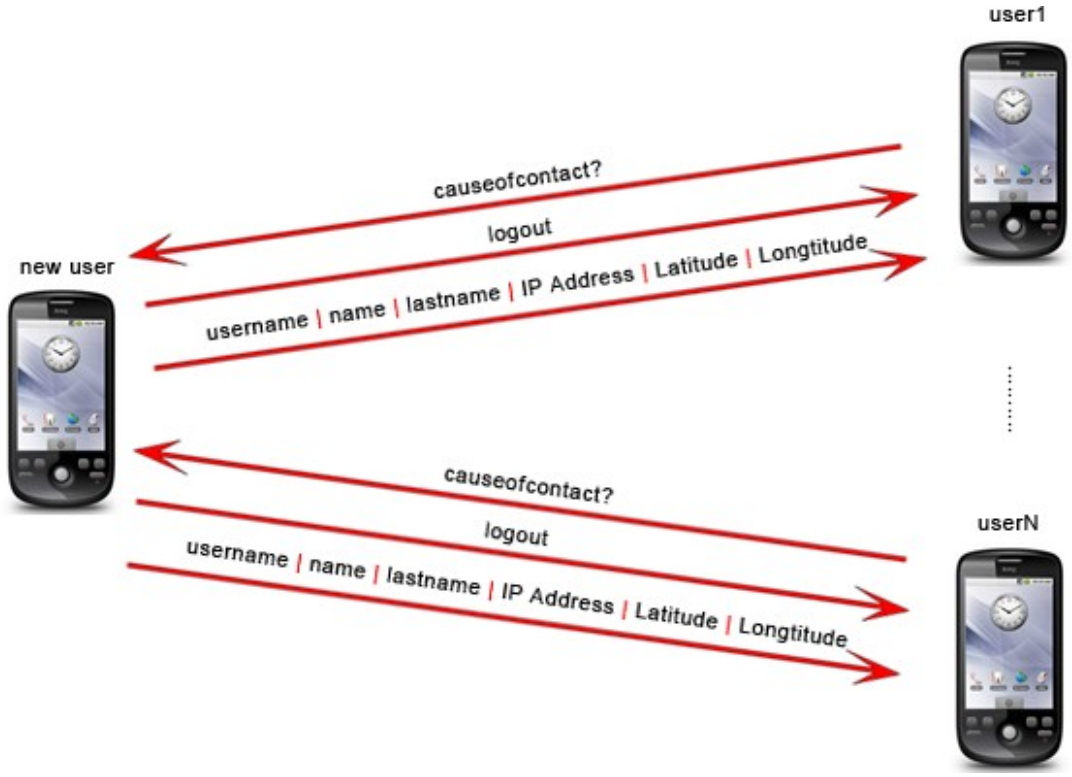
Το παραπάνω μήνυμα προέρχεται όπως ειπώθηκε από τους χρήστες που είναι ήδη συνδεδεμένοι στην υπηρεσία ενώ μετά από αυτό το μήνυμα ο χρήστης προς διαγραφή από την υπηρεσία στέλνει τα ακόλουθα μηνύματα:

logout

Το παραπάνω μήνυμα προετοιμάζει την εφαρμογή των χρηστών που το λαμβάνουν ότι πρόκειται ένας χρήστης να αποσυνδεθεί από την υπηρεσία και θα στείλει σε επόμενο μήνυμα τα προσωπικά του στοιχεία ώστε να είναι δυνατή η ταυτοποίηση του στην βάση δεδομένων του κάθε χρήστη και να τον διαγράψει από αυτήν:

username	name	lastname	IP Address	Latitude	Longtitude

Με την αποστολή των παραπάνω μηνυμάτων σε κάθε χρήστη οι συνδέσεις τερματίζονται και η ενημέρωση για την έξοδο του χρήστη έχει πραγματοποιηθεί.



4.2.2.3.2 Εξαγωγή χρήστη από την υπηρεσία online chat

Ο χρήστης σε περίπτωση που έχει διαλέξει να συνδεθεί σε αυτή την υπηρεσία,είτε έχει χρησιμοποιήσει τον εξυπηρετητή και συνεπώς έχει εγγραφεί σε αυτόν είτε έχει χρησιμοποιήσει την αυτόματη αναζήτηση, δημιουργεί μια TCP connection, και ο κάθε χρήστης που είναι ήδη συνδεδεμένος αποστέλλει το παρακάτω μήνυμα:

send..

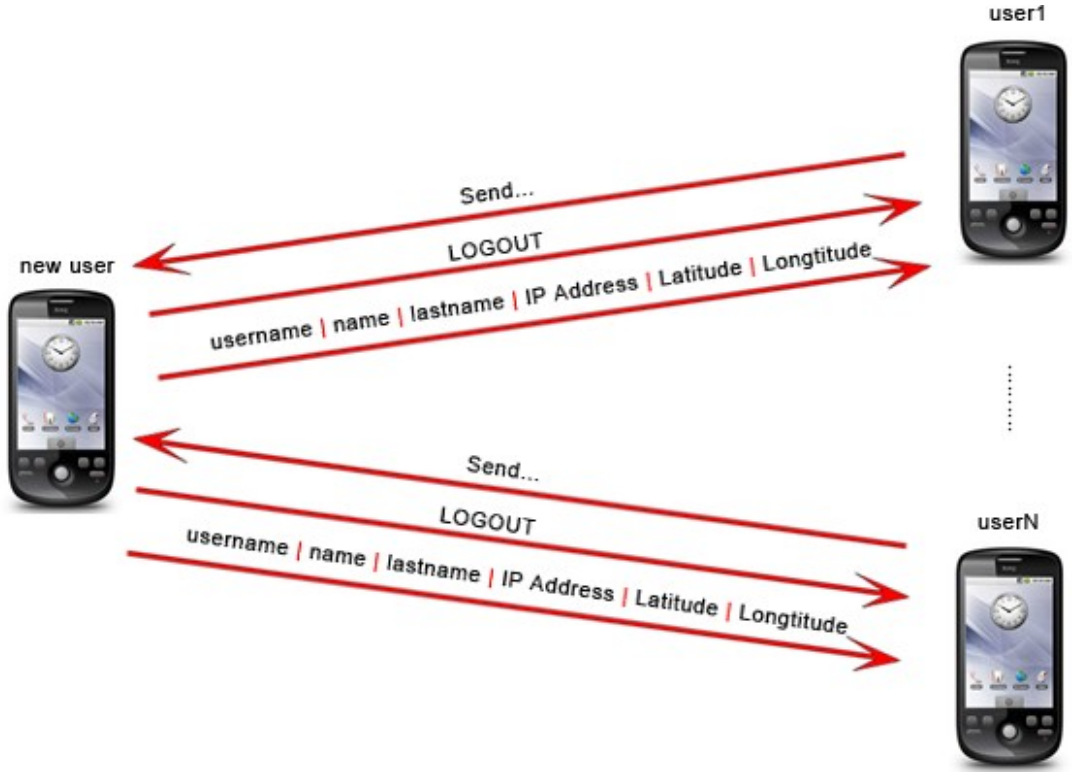
Το παραπάνω μήνυμα προέρχεται όπως ειπώθηκε από τους χρήστες που είναι ήδη συνδεδεμένοι στην υπηρεσία ενώ μετά από αυτό το μήνυμα ο χρήστης προς διαγραφή από την υπηρεσία στέλνει τα ακόλουθα μηνύματα:

LOGOUT

Το παραπάνω μήνυμα προετοιμάζει την εφαρμογή των χρηστών που το λαμβάνουν ότι πρόκειται ένας χρήστης να αποσυνδεθεί από την υπηρεσία και θα στείλει σε επόμενο μήνυμα τα προσωπικά του στοιχεία ώστε να είναι δυνατή η ταυτοποίηση του στην βάση δεδομένων του κάθε χρήστη και να τον διαγράψει από αυτήν:

username	name	lastname	IP Address	Latitude	Longtitude

Με την αποστολή των παραπάνω μηνυμάτων σε κάθε χρήστη οι συνδέσεις τερματίζονται και η ενημέρωση για την έξοδο του χρήστη έχει πραγματοποιηθεί.



4.2.2.4 *Συνομιλία χρηστών στην υπηρεσία Private Chat*

Όταν ένας χρήστης θελήσει να συνομιλήσει με έναν άλλο χρήστη στην υπηρεσία αυτή,ύστερα από την επιλογή του χρήστη από την λίστα με τους διαθέσιμους χρήστες

η εφαρμογή ξεκινά την παρακάτω ανταλλαγή μηνυμάτων με την εφαρμογή του άλλου χρήστη. Αρχικά δημιουργείται μια σύνδεση TCP με διεύθυνση IP την IP Address του χρήστη που θέλει να συνομιλήσει και στην "πόρτα" 8189. Εάν η σύνδεση δεν είναι εφικτή, διότι ο χρήστης δεν βρίσκεται στην υπηρεσία είτε διότι έχει αφήσει κάποιο μήνυμα στο χρήστη που ήθελε να επικοινωνήσει μαζί του και αποσυνδέθηκε είτε διότι κόπηκε η σύνδεσή του ξαφνικά από το δίκτυο και δεν είχε την δυνατότητα να ενημερώσει το σύστημα. Εφόσον ο χρήστης είναι ακόμη συνδεδεμένος τότε στέλνει πρώτος ένα μήνυμα με περιεχόμενο το παρακάτω:

cause_of_talk?

Ο χρήστης Α (ο χρήστης που ξεκίνησε την διαδικασία για συνομιλία),στέλνει το ακόλουθο μήνυμα:

chat

Ο χρήστης Β (ο χρήστης που δέχεται την συνομιλία),εάν είναι ήδη απασχολημένος σε μια άλλη συνομιλία αποστέλλει μήνυμα:

Busy..

Αυτό όμως το μήνυμα δεν αποτρέπει την εφαρμογή του χρήστη Α και τον ίδιο να αποστείλει μηνύματα στον χρήστη Β.

Αν ο χρήστης Β δεν είναι απασχολημένος τότε αποδέχεται τη συνομιλία και στέλνει το μήνυμα:

yes!

Το οποίο ακολουθείται από ένα μήνυμα που περιλαμβάνει την πόρτα στην οποία θα του μιλήσει ο χρήστης Α.

port_number

Η παρούσα σύνδεση ανάμεσα στον χρήστη Α και τον χρήστη Β τερματίζεται και ο χρήστης Α αφού δεχτεί τα παραπάνω δύο μηνύματα δημιουργεί μία νέα σύνδεση με

τον χρήστη Β με νέα σύνδεση στην ίδια IP Address αλλά σε νέα πόρτα που από default στην εφαρμογή είναι η 8188.

Κατά τη διάρκεια της συνομιλίας τα μηνύματα που ανταλλάσσονται μεταξύ των δύο χρηστών έχουν την παρακάτω δομή:

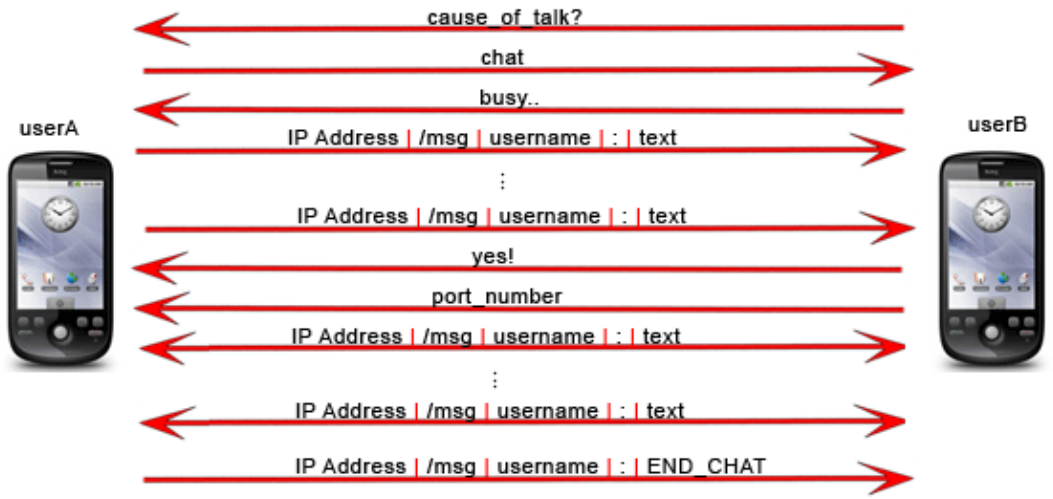
IP Address	/msg	username	:	text

Τα μηνύματα αυτά αποτελούνται από την διεύθυνση IP Address του αποστολέα, από το "συνθηματικό" /msg που χρησιμοποιείται στο κώδικα για τον διαχωρισμό στην πλευρά του παραλήπτη του μηνύματος σε δύο μέρη και προβολή του δεύτερου. Ακολούθως αποτελείται από το ψευδώνυμο του αποστολέα, ένα σύμβολο ":" και τέλος από το text, που είναι το μήνυμα που στέλνει ο αποστολέας στον παραλήπτη.

Όταν ο ένας από τους δύο χρήστες θελήσουν να εξέλθουν από την συνομιλία τότε αποστέλλεται το παρακάτω μήνυμα:

IP Address	/msg	username	:	END_CHAT

Δηλαδή στέλνεται ένα συγκεκριμένο μήνυμα σαν text με το περιεχόμενο "END_CHAT". Με την αποστολή αυτού του μηνύματος η συνομιλία τερματίζεται και ο χρήστης που το παραλαμβάνει ενημερώνεται από την εφαρμογή για την λήξη της συνομιλίας.



4.2.2.5 *Συνομιλία χρηστών στην υπηρεσία Public Chat*

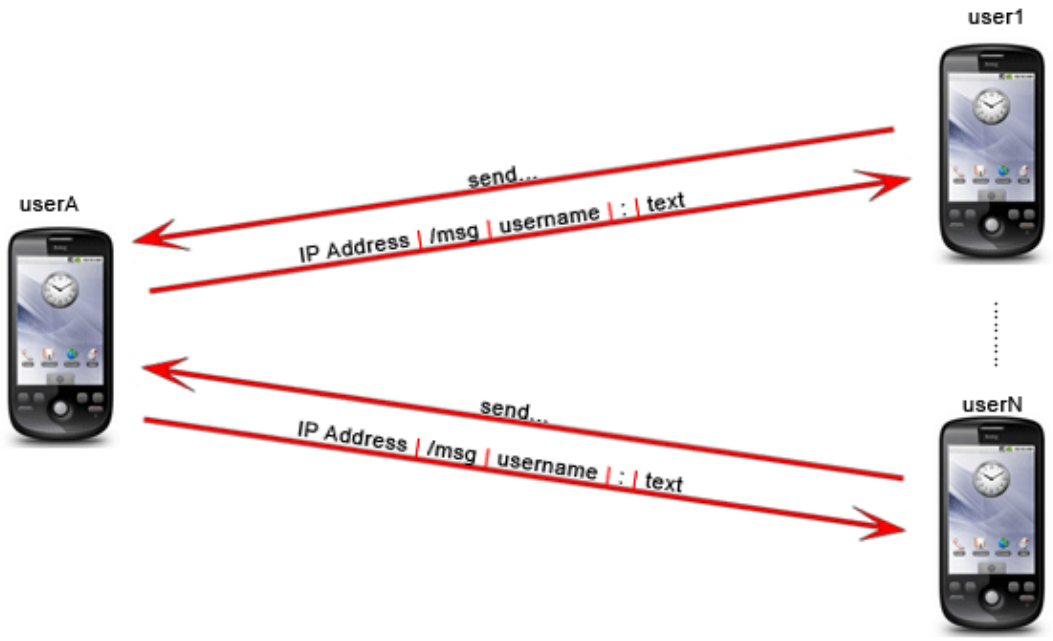
Στην υπηρεσία αυτή, ο κάθε χρήστης όταν θέλει να στείλει ένα μήνυμα προς τους υπόλοιπους χρήστες αυτό που ουσιαστικά κάνει, είναι η δημιουργία με τον καθένα TCP connection στην "πόρτα" 8188. Με το που εγκαθίστανται οι συνδέσεις αυτές, οι χρήστες που πρόκειται να λάβουν το μήνυμα του χρήστη A (ο χρήστης που θέλει να στείλει ένα μήνυμα στους υπόλοιπους χρήστες) στέλνουν ένα μήνυμα με περιεχόμενο:

send...

Εν συνεχεία ο χρήστης A στέλνει το κείμενο που επιθυμεί και το μήνυμα αυτό έχει την εξής δομή:

IP Address	/msg	username	:	text

Με την αποστολή του μηνύματος αυτού η σύνδεση τερματίζεται και η εφαρμογή είναι σε αναμονή για αποστολή ή παραλαβή άλλου μηνύματος.



4.2.2.6 Ανταλλαγή αρχείων

Στη παράγραφο αυτή περιγράφουμε τον τρόπο με τον οποίο το σύστημα ανάμεσα στους χρήστες επικοινωνεί για να ανταλλάξουν αρχεία. Έστω ότι ο χρήστης Α θέλει να παραλάβει ένα από τα αρχεία που έχει προς διάθεση ο χρήστη Β. Αρχικά ο χρήστης Α εγκαθιστά μια σύνδεση με τον χρήστη Β χρησιμοποιώντας την IP Address που έχει στη διάθεση του για το χρήστη Β και στην "πόρτα" 8189. Με την δημιουργία της εγκατάστασης της σύνδεσης ο χρήστη Β στέλνει το ακόλουθο μήνυμα:

causeofcontact?

Ο χρήστης Α τότε, ύστερα από το μήνυμα αυτό αποστέλλει το παρακάτω μήνυμα που ενημερώνει τον χρήστη Β ότι προτίθεται να παραλάβει ένα αρχείο από αυτόν:

receivefile

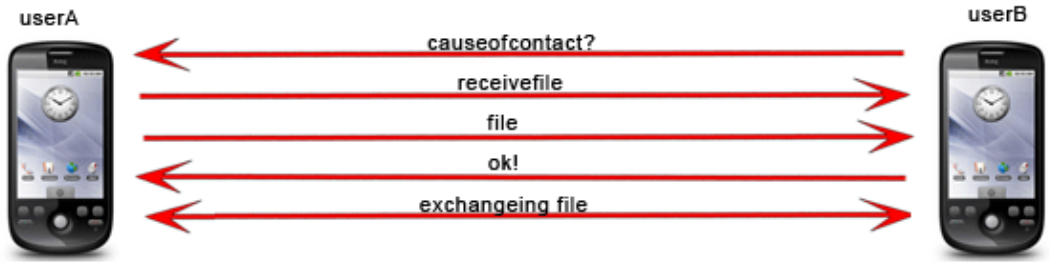
Ακολούθως ο χρήστης Α στέλνει άλλον ένα μήνυμα με το μονοπάτι και το όνομα του αρχείου που επιθυμεί να παραλάβει και αναμένει την έγκριση από τον χρήστη Β:

file

Ο χρήστης Β εφόσον δεν υπάρχει κάποια εμπλοκή και είναι δυνατόν να εξυπηρετήσει το αίτημα του χρήστη Α στέλνει το ακόλουθο μήνυμα και ταυτόχρονα ενεργοποιεί τη διαδικασία αποστολής αρχείου:

ok!

Η παρούσα σύνδεση μεταξύ των δύο χρηστών τερματίζεται ενώ μία νέα σύνδεση μεταξύ τους με πρωτοβουλία του χρήστη Α ξεκινά και συνδέονται στην "πόρτα" 1500 για να γίνει η ανταλλαγή του αρχείου. Όταν τελειώσει η μεταφορά του αρχείου από τον χρήστη Β στον χρήστη Α η σύνδεση αυτή τερματίζεται και συνεπώς η ανταλλαγή του αρχείου έχει πραγματοποιηθεί.


userA
userB
causeofcontact?
receivefile
file
ok!
exchangeing file

5 Σύνοψη

Στο κεφάλαιο αυτό συνοψίζουμε σύντομα κάποια χρήσιμα συμπεράσματα της εφαρμογής που αναπτύχθηκε καθώς και κάποιες κατευθύνσεις για μελλοντικές επεκτάσεις της διπλωματικής.

5.1 Συμπεράσματα

Στην παρούσα διπλωματική ασχοληθήκαμε με την ανάπτυξη ενός συστήματος που παρέχει υπηρεσίες στους χρήστες βασιζόμενες εν μέρει στην θέση του χρήστη (Location Based Services). Τέτοια συστήματα αποκτούν όλο και πιο μεγάλο ενδιαφέρον καθώς η τεχνολογία αναπτύσσεται και οι πληροφορίες είναι πλέον τεράστιες σε όγκο και απαιτείται συνεπώς ένα φιλτράρισμα ώστε ο χρήστης να έχει πρόσβαση μόνο σε αυτές που τον ενδιαφέρουν, ανάλογα στη θέση που είναι.

Αναπτύχθηκε τόσο το κομμάτι του κεντρικού τοπικού εξυπηρετητή ο οποίος αναλαμβάνει να κρατά τα στοιχεία των χρηστών που επιλέγουν να χρησιμοποιήσουν την εφαρμογή με τη βοήθεια του εξυπηρετητή, όσο και το τμήμα της εφαρμογής στα κινητά τηλέφωνα με λογισμικό Android, τα οποία όσο περνά ο καιρός γίνονται ολοένα και πιο δημοφιλή και οι προγραμματιστές δημιουργούν με υψηλό ρυθμό συνέχεια εφαρμογές,πράγμα που ενθαρρύνει τους επίδοξους-μελλοντικούς χρήστες τέτοιων κινητών τηλεφώνων-pda να τα αγοράσουν και να αξιοποιήσουν όλες τις δυνατότητες τους. Η εφαρμογή στο κινητό τηλέφωνο έχει τη δυνατότητα να αναζητήσει χρήστες που συμμετέχουν σε διάφορες υπηρεσίες είτε με τη βοήθεια του τοπικού κεντρικού εξυπηρετητή και να συνδεθούν εν συνεχεία σε μία από τις υπηρεσίες , private chat, public chat και υπηρεσία ανταλλαγής αρχείων, είτε απευθείας η εφαρμογή μόνη της να αναζητήσει, στο τοπικό δίκτυο που καλύπτεται από το ίδιο ασύρματο δίκτυο, χρήστες που ενδέχεται να χρησιμοποιούν μία από τις δύο υπηρεσίες private chat ή public chat.

Η προσπάθεια σε αυτή τη διπλωματική επικεντρώθηκε σε δύο άξονες. Αρχικά τη δημιουργία μιας εφαρμογής σε περιβάλλον Android που θα παρέχει υπηρεσίες επικοινωνίας και ανταλλαγής αρχείων στους χρήστες με όσο το δυνατόν η επικοινωνία των των χρηστών, να μην ελέγχεται-καθορίζεται από κάποιον εξυπηρετητή. Αυτό επιτεύχθηκε σε μεγάλο βαθμό όπου οι χρήστες τη μόνη

συνδιαλλαγή που έχουν με τον τοπικό εξυπηρετητή είναι η εγγραφή σε αυτόν, η εγγραφή σε μία από τις υπηρεσίες του και η ενημέρωση της εφαρμογής του χρήστη για τους χρήστες που είναι ήδη συνδεδεμένοι στην υπηρεσία που έχει εγγραφεί και ο ίδιος. Τέλος επικοινωνεί με τον εξυπηρετητή για να τον ενημερώσει όταν ο χρήστης αποσυνδέεταικ και πρέπει να διαγραφεί από τη βάση δεδομένων του εξυπηρετητή. Η εφαρμογή των χρηστών στα κινητά τους τηλέφωνα αναλαμβάνει τον κύριο όγκο του συντονισμού με τους υπόλοιπους χρήστες ενημερώνοντας για την είσοδο του χρήστη σε μία από τις υπηρεσίες, για τα προσωπικά στοιχεία του χρήστη και τα στοιχεία επιμοινωνίας μαζί του, δηλαδή τη διεύθυνση IP Address και τις γεωγραφικές συντεταγμένες. Η εφαρμογή σε συνεργασία με την ίδια εφαρμογή στους άλλους χρήστες αναλαμβάνει τον συντονισμό για την επικοινωνία των χρηστών είτε στην υπηρεσία private chat είτε public chat. Το ίδιο συμβαίνει και στην υπηρεσία ανταλλαγής δεδομένων-αρχείων όπου η εφαρμογή των χρηστών στα κινητά τους τηλέφωνα αναλαμβάνει τον συντονισμό για να ενημερώνονται για νέα διαθέσιμα αρχεία και για να παραλάβουν ή να στείλουν αρχεία ανάλογα με την επιθυμία των χρηστών κάθε φορά. Ουσιαστικά ο εξυπηρετητής το μόνο που κάνει είναι να παρέχει γρήγορα στους χρήστες που εισέρχονται στο σύστημα τα προσωπικά στοιχεία και πληροφορίες των άλλων χρηστών. Στον δεύτερο άξονα έγινε μια προσπάθεια σε περίπτωση που δεν υπάρχει τοπικός εξυπηρετητής να μπορούν οι χρήστες να επικοινωνούν απευθείας, δηλαδή να είναι ικανοί να αναζητήσουν και να εντοπίσουν άλλους χρήστες που χρησιμοποιούν την ίδια εφαρμογή, και αυτό επιτεύχθηκε στο βαθμό που οι χρήστες αυτοί βρίσκονται στο ίδιο τοπικό δίκτυο-ασύρματο δίκτυο δηλαδή εξυπηρετούνται “δικτυακά” από τον ίδιο σταθμό βάσης.

5.2 Μελλοντικές επεκτάσεις

Παρακάτω αναφέρονται μερικές ιδέες για επεκτάσεις της παρούσας εργασίας:

5.2.1 Εναλλακτικοί τρόποι εύρεσης θέσης

Στην παρούσα διπλωματική το κινητό τηλέφωνο για να εντοπίσει τη θέση του χρήστη χρησιμοποιεί αποκλειστικά το GPS δέκτη, το οποίο είναι σε όλα τα κινητά τηλέφωνα που χρησιμοποιούν το λογισμικό Android.

Μελλοντικά θα ήταν καλό να υποστηρίζονται από την εφαρμογή και άλλοι τρόποι εύρεσης της θέσης του χρήστη, όπως είναι η αξιοποίηση των κεραιών κινητής

τηλεφωνίας όπου χρησιμοποιώντας την καθυστέρηση της λήψης του σήματος από τις κοντινές κεραίες να μπορεί να βρεθεί με απόκλιση μικρή το στίγμα του χρήστη. Ένας άλλος τρόπος εύρεσης θέσης είναι με την αξιοποίηση των κυψελών της κινητής τηλεφωνίας και συνεπώς να έχουμε μια απόκλιση έστω μικρή για τη θέση του χρήστη και τέλος με την αξιοποίηση τοπικών ασυρμάτων συστημάτων που μπορούν να παρέχουν πληροφορίες για τη θέση που βρισκόμαστε. Τέλος ένας ακόμη τρόπος εύρεσης της θέσης του κινητού τερματικού μπορεί να πραγματοποιηθεί με την αξιοποίηση της τεχνολογίας Bluetooth η οποία ήταν "κλειδωμένη" από τους προγραμματιστές λόγο κενών ασφαλείας. Στην έκδοση Android 2.0 γίνεται πλέον εφικτή η χρήση του Bluetooth και συνεπώς η αξιοποίηση του για τον εντοπισμό της θέσης του κινητού τηλεφώνου.

Οι παραπάνω τρόποι εύρεσης θέσης είναι αυτοματοποιημένοι και δεν ελέγχονται από το χρήστη. Θα μπορούσε μελλοντικά ο χρήστης να εισάγει τη διεύθυνση στην οποία είναι, μαζί με το όνομα της πόλης, και με την σύνδεσή του στο διαδίκτυο και την αξιοποίηση πακέτων που έχει το android να βρίσκει το στίγμα του χρήστη το σύστημα.

5.2.2 ΄Υπαρξη ή μη εξυπηρετητή

Όπως αναφέρθηκε, σε προηγούμενη παράγραφο, αλλά και κατά τη διάρκεια του κειμένου αυτού, η εφαρμογή χρησιμοποιεί ως ένα μικρό βαθμό έναν τοπικό εξυπηρετητή, εφόσον το επιλέγουν οι χρήστες ώστε αυτοί να συντονίζονται σε αρχικό στάδιο και ειδικότερα να ενημερώνονται για την ύπαρξη άλλων χρηστών από τον εξυπηρετητή.

Σε μια μελλοντική υλοποίηση θα μπορούσε να δοθεί βάση στην αυτονόμηση του συστήματος, δηλαδή να μην υπάρχει κάποιος τοπικός εξυπηρετητής που να βοηθά στην ενημέρωση χρηστών που βρίσκονται στο ίδιο δίκτυο και χρησιμοποιούν την ίδια υπηρεσία με τον εισερχόμενο στο σύστημα χρήστη. Αυτό απαιτεί την επέκταση της αυτόματης αναζήτησης χρηστών όχι μόνο στο τοπικό δίκτυο που καλύπτεται ενδεχομένως από έναν σταθμό βάσης αλλά από ένα ευρύτερο τοπικό δίκτυο, όπου σταθμοί βάσης συνεργάζονται μεταξύ τους για την κάλυψη ενός μεγάλου χώρου όπως αυτού του Πολυτεχνείου για παράδειγμα. Στην υπάρχουσα υλοποίηση χρήστες που βρίσκονται σε αυτόν τον ευρύτερο χώρο μπορούν να επικοινωνήσουν εφόσον έχουν επιλέξει να εγγραφούν στον τοπικό εξυπηρετητή.

Μία άλλη μελλοντική υλοποίηση θα μπορούσε να δώσει βάρος στο να επεκτείνει τον τοπικό εξυπηρετητή και να αντικαταστήσει τους εκάστοτε τοπικού εξυπηρετητές με έναν κεντρικό εξυπηρετητή ο οποίος θα αναλάβει να ομαδοποιεί τους χρήστες με βάση τη θέση τους. Η ομαδοποίηση θα μπορεί να γίνεται είτε μέσω της γεωγραφικής τους θέσης είτε με βάση την εξωτερική IP Address του δικτύου στο όποιο φιλοξενούνται. Ο κεντρικός εξυπηρετής θα μπορεί να συντονίζει την όλη διαδικασία σε περιπτώσεις επικοινωνίας χρηστών που βρίσκονται σε άλλα υποδίκτυα και δεν έχουν απευθείας δυνατότητα επικοινωνίας .

5.2.3 Εμπλουτισμός με νέες Υπηρεσίες

Στην παρούσα εργασία οι υπηρεσίες που είναι διαθέσιμες είναι τρεις. Private chat, public chat και εφόσον, οι χρήστες χρησιμοποιούν τον τοπικό εξυπηρετητή, η υπηρεσία ανταλλαγής δεδομένων.

Σε μια μελλοντική επέκταση της διπλωματικής θα ήταν θετικό να μπορεί ο χρήστης να χρησιμοποιεί και τις τρεις υπηρεσίες ταυτόχρονα, να μην υπάρχει περιορισμός του χρήστη να είναι εγγεγραμμένος κάθε φορά μόνο σε μία από τις τρεις υπηρεσίες.

Μελλοντικά θα μπορούσαν και οι χρήστες που χρησιμοποιούν την αυτόματη αναζήτηση άλλων χρηστών να έχουν την δυνατότητα ανταλλαγής αρχείων μεταξύ τους και να μην είναι αναγκασμένοι να συνδεθούν σε τοπικό εξυπηρετητή για να έχουν αυτή τη δυνατότητα.

Θα μπορούσαν να εισαχθούν δυνατότητες αποκλεισμού και απόκρυψης χρηστών με βάση την επιθυμία τους, δηλαδή ο χρήστης κατά την είσοδό του στο σύστημα και συγκεκριμένα σε μια υπηρεσία να μπορεί να αποκλείσει κάποιους χρήστες από το να τον βλέπουν ή να εισέρχεται αλλά να μην ενημερώνονται οι υπόλοιποι χρήστες για την είσοδό του.

6 Βιβλιογραφία

[1] Carmine Ciavarella & Fabio Paterno , "Design Criteria for Location-aware, Indoor, PDA applications", Proceedings of Fifth International Symposium on Human Computer Interaction with Mobile Devices and Services, Udine, Italy, 2003, επ. L. Chittaro, Lecture Notes in Computer Science, Vol. 2795, 131-144. Berlin: Springer-Verlag.

[2] Dillip Mohapatra, Suma S.B, "Survey of Location Based Wireless Services", Personal Wireless Communications, 2005. ICPWC 2005. IEEE International Conference, 2005.

[3] Christian Kray, Jorg Baus, "A Survey of Mobile Guides", ICPWC 2005.

[4] Todd Simcock, Stephen Peter Hillenbrand & Bruce H. Thomas, "Developing a Location Based Tourist Guide Application", 2003

[5] D. A. Abowd, C. G. Atkeson, J. Hong, S. Long, K. R., and M. Pinkerton. Cyperguide: A Mobile Context-Aware Tour Guide.Wireless Networks, 3(5):421–433, 1996

[6] H. Anegg, H. Kunczier, E. Michlmayr, G. Pospischil, and M. Umlauft. LoL@: designing a location based UMTS application. Elektrotechnik-und-Informationstechnik, 119(2):48–51, 2002

[7] K. Cheverst, N. Davies, K. Mitchell, A. Friday, and C. Efstratiou. Developing a Context-aware Electronic Tourist Guide: Some Issues and Experiences. In Proceedings of the 2000 Conference on Human Factors in Computing Systems (CHI-00), pages 17–24, New York, 2000. ACM Press.

[8] CARC. Web page of the Cyber Assist Research Center (CARC). http://www.carc.aist.go.jp, 2002.

[9] C. Kray, K. Laakso, C. Elting, and V. Coors. Presenting route instructions on mobile devices. In W. L. Johnson, E. Andr´e, and J.Domingue, editors, Proceedings of IUI 03, pages 117–124, MiamiBeach, FL, 2003. ACM Press.

[10] J. Baus, C. Kray and A. Kruger. Visualization of route descriptions in a

resource-adaptive navigation aid. Congnitive Processing, 2(2-3):323-345. 2001.

[11] Dey, A. K. and Abowd, G. D. (2000) In Proceedings of the 2nd application to dynamically select localization technology, the International Symposium on Handheld and Ubiquitous Computing (HUC2K) Bristol, UK, pp. 172-186

[12] Anastasios Zafeiropoulos, Emmanuel Solidakis, Stavroula Zoi, Nikolaos Konstantinou, Panagiotis Papageorgiou, Panagiotis Stathopoulos Nikolas Mitrou. A Lightweight approach forproviding location based contnet retieval. NTUA Athens, Greece

[13] Yanying, G., Anthony, L. and Ignas, N. (2009) 'A survey of indoor positioning systems for wireless personal networks', IEEE Communications Surveys & Tutorials, Vol. 11, No. 1.

[14] Hopper, A., Harter, A. and Blackie, T. (1993) 'The active badge system', Proc. of INTERCHI-93, Amsterdam.

[15] Priyantha, B., Chakraborty, A. and Balakrishnan, H. (2000) 'The cricket location support system',Mobile Computing and Networking, pp.32–43.

[16] Bahl, P. and Padmanabhan, N. (2000) 'RADAR: an in-building RF-based user location and tracking system',INFOCOM,Tel-Aviv,Israel, pp.775–784.

[17] Schilit, B., LaMarca, A., Borriello, G., William Griswold, M., Lazowska, E., Balachandran, A. and Iverson, V. (2003) 'Challenge: ubiquitous location-aware computing and the Place Lab initiative', First ACM International Workshop of Wireless Mobile Applications and Services on WLAN.

[18] Otsason, V., Varshavsky, A., LaMarca, A. and Eyal de Lara (2005) 'Accurate GSM indoor localization', 7th International Conference on Ubiquitous Computing (UbiComp), Tokyo, Japan.

[19] Anastasios Zafeiropoulos, Ioannis Papaioannou, Emmanuel Solidakis, Nikolaos Konstantinou, Panagiotis Stathopoulos and Nikolas Mitrou NTUA Greece, Athens. Exploiting Bluetooth for deploying indoor LBS over a

localisation infrastructure independent architecture.

[20] A mobile location-based architecture for intelligent selecting multi-dimension position data over interent. Sheng HU , Jun-Wei GE , Zheng-Wu YUAN , Hae-Young BAE

[21] An Architecture for Delivering Location-Based Services. Ronald Beaubrun, Bernard Moulin and Nafaa Jabeur. university Laval, Pavillon Adrien-Pouliot, Québec (Québec), Canada 2007

[22] Διαδίκτυο και Εφαρμογές Σημειώσεις μαθήματος Διαδίκτυο και εφαρμογές της ροής Δ Επικοινωνίες και Δίκτυα Επικοινωνιών Εθνικού Μετσόβιου Πολυτεχνείου

[23] Συστήματα Βάσεων Δεδομένων, Η πλήρης Θεωρία των Βάσεων Δεδομένων, 4η έκδοση , Silberschatz, Korth, Sudarshan Εκδόσεις Μ. Γκιούρδας Αθήνα 2004

[24] Autonomic LBS based on context: Preview, LI Fang LiXiaolei, BIAN Fuling Research Center of Spatial Information and Digital Engineering, Wuhan University

[25] Indoor Location, Dr. Zeev Weissman Chief Scientist Tadlys Ltd

[26] Building a framework to characterize location-based services, Ana M. Bernados, Jose R. Casar and Paula Tarrio, Universidad Politecnica de Madrid

[27] Location-Based Services, The State of the Art, Stuart J. Barnes Victoria University of Wellington

[28] A Survey of Indoor Positioning Systems for Wireless Personal Networks, Yanying Gu, Anthony Lo, Senior Member, IEEE, and Igans Niemegeers

[29] Are GSM phones THE solution for localization? Alex Varshavsky, Eyal de Lara Computer Science University of Toronto, Jon Froehlich, Fred Potter Computer Science and Engineering University of Washington, Karen Tang Computer Science Carnegie Mellon University, Timothy Sohn Computer Science and Engineering Univeristy of California, Mike Y. Chen, Dirk

Haehnel, Jeffrey Hightower, Anthony LaMarca, Ian Smith Inter Research Seattle

[30] Y. Wang, L. Cuthbert, Francis J. Mullany, P. Stathopoulos, V. Tountopoulos, M. Senis, "Exploring Agent-based Wireless Business Models and Decision Support Applications in an Airport Environment", Journal of Telecommunications and Information Technology, no 3/2004.

[31] Getting Android Emulator working with Google Maps API Key, http://informationideas.com/news/2008/11/06/getting-android-emulator-working-with-google-maps-api-key/ , Last access December 2009

[32] Using Google Maps in Android http://www.devx.com/wireless/Article/39145/1954 ,Last access December 2009

[33] Modifying Google Mpas appearence in Android http://blogoscoped.com/archive/2008-12-15-n14.html ,Last access December 2009

[34] Getting you IP Anddress on your device http://www.droidnova.com/tag/localhost ,Last access December 2009

[35] Control the Back Button in Android http://stackoverflow.com/questions/1265095/control-the-back-button-in-android ,Last access December 2009

[36] Usefull code examples for Google Android http://www.anddev.org ,Last access December 2009

[37] The developer Guid from Android http://developer.android.com/guide/tutorials ,Last access December 2009

[38] Work with Google Android Databases http://www.brighthub.com/mobile/google-android/articles/25881.aspx ,Last access December 2009

[39] Search files in sd-card in Adnroid http://www.coderanch.com/t/442062/Android/Mobile/search-files-presnt-

SD-Card#1966618 ,Last access December 2009

[40] Gps Status recognize http://groups.google.com/group/android-developers/browse_thread/thread/ba32f9f7759be150 ,Last access December 2009

[41] Android Market Statistics http://androinica.com/2009/09/07/android-market-has-10000-apps-many-of-which-dont-appeal-to-users/ ,Last access December 2009

7 Παραρτήματα

Στο παρών παράρτημα παρατίθενται συγκεκριμένα κομμάτια κώδικα, τα οποία είναι σημαντικά για την λειτουργία της εφαρμογής και καλύπτουν το κομμάτι της επικοινωνίας μεταξύ του εξυπηρετητή και του κινητού τερματικού. Ο κώδικας είναι χωρισμένος με βάση στα αρχεία java τα οποία ανήκει και με βάση τη λειτουργία που κάθε μπλοκ κώδικα υλοποιεί.

7.1 Chatting

Ο κώδικας αυτός χρησιμοποιείται για την επικοινωνία του χρήστη Α με τον χρήστη Β όταν ο χρήστης Β έχει προσκαλέσει σε συνομιλία τον πρώτο χρήστη και αυτός είναι διαθέσιμος.

```
public class Comm extends Thread
        {
            public void run()
            {
                try
                    {
                            Bundle extras = getIntent().getExtras();
                            int port=extras.getInt("port");

                      s = new ServerSocket(port);
                      incoming = s.accept ( ); //socket obj. incoming is constructed
                      in = new BufferedReader (new InputStreamReader(incoming.getInputStream()));
                      out = new PrintWriter (incoming.getOutputStream(),true);
                      out.println( myip+"/msg"+myname+":"+"Start Chatting..." );

                     Boolean done=false;
                     while(!done)
                     {
                            if(in.ready()){

                                        inmessage=in.readLine();
                                        String[] cutip=inmessage.split("/msg");
                                        inmessage=cutip[1];
                                        String[] messagetemp=inmessage.split(":");

                                        if(messagetemp[1].equals("END_CHAT")){
                                                done=true;
                                        runOnUiThread(new InAction2("end"));
                                        }
                                        else if (inmessage.equals("\n"));
                                        else{
                                        //inputlog="\n"+user+" : "+inmessage;
                                        inputlog="\n"+inmessage;
                                        runOnUiThread(new InAction(inputlog));
                                        }
                                }
                            }
                    out.println("End discussion...");
                        incoming.close();
                        s.close();
                    }catch (Exception e) { e.printStackTrace();runOnUiThread(new InAction("skaata")); }

}}
```

7.2 ChattingMe

Ο κώδικας αυτός χρησιμοποιείται για την επικοινωνία του χρήστη Α με τον χρήστη Β όταν ο χρήστης Α προσκαλεί σε συνομιλία τον χρήστη Β.

```
public class Conn2 extends Thread
{public void run(){
                        try {
                        boolean stillinchat=false;
                        while(!stillinchat){

                        if(stillsending){
                                if(!connected){
                                        outemp = new Socket();
                                        outemp.connect(new InetSocketAddress(ip, port), 500);
                                out = new PrintWriter( new BufferedWriter( new
                                                                OutputStreamWriter(outemp.getOutputStream())),true);
                                in = new BufferedReader(new InputStreamReader(outemp.getInputStream()));
                           connected=true;
                           out.println("chat");
                                }
                                if(in.ready()){
                                        if(in.readLine().equals("yes!")) {
                                                port=Integer.parseInt(in.readLine());
                                                outemp.close();
                                                connected=false;
                                                stillsending=false;
                                                try {
                                                        Thread.sleep(1000);
                                                } catch (InterruptedException e) {
                                                        e.printStackTrace();
                                                }}
                                }
                        }

                        if(!stillsending){
                                if(!connected){outgoing = new Socket(ip, port);
                        out = new PrintWriter( new BufferedWriter( new
                                                        OutputStreamWriter(outgoing.getOutputStream())),true);
                            in = new BufferedReader(new InputStreamReader(outgoing.getInputStream()));
                            out.println(myip+"/msg"+myname+":"+"Start Chatting...");
                            connected=true;
                        }boolean done=false;
                while(!done){
                            if(in.ready()){
                                        String connResp = in.readLine();
                                        String[] cutip=connResp.split("/msg");
                                        connResp=cutip[1];
                                        String[] tempmessage=connResp.split(":");
                                        if(tempmessage[1].equals("END_CHAT")){
                                                done=true;
                                                stillinchat=true;
                                                runOnUiThread(new InAction2("end"));
                                         }else if (connResp.equals("\n"));
                                                else{
                                           inputlog="\n"+connResp;
                                           runOnUiThread(new InAction(inputlog));
                                                }
                                        }
                    }outgoing.close();
                        }
                        }
                }catch(IOException exception) {
                        found=false;
                        runOnUiThread(new useroffline("could not start Chat"));
                        }
}}
```

7.3 Manualsearch

Ο κώδικας αυτός είναι υπεύθυνος για την εύρεση άλλων χρηστών που χρησιμοποιούν την ίδια υπηρεσία με τον χρήστη και δεν χρησιμοποιεί τον εξυπηρετητή.

```
void searchip(){
                found=false;
        counter=3;

                        myip=getLocalIpAddress();
                        String[] tempip=myip.split( "\\." );
                        String initialip=tempip[0]+"."+tempip[1]+"."+tempip[2]+".";
                        int minIP=2;
                        int maxIP=10;
                        for(int i=minIP;i<maxIP;i++){
                                String randomIp=initialip+i;
                                connectToServer(randomIp);
                                String count=Integer.toString(counter);
                                 runOnUiThread(new InAction(count));

                        }
                        if(counter==0)
                        runOnUiThread(new InAction2("empty"));

    }


    private void connectToServer(String host)
      { try {
                long timeStart=System.currentTimeMillis();
                        Socket outgoing = new Socket();
                        outgoing.connect(new InetSocketAddress(host, 8189), 200);
                    PrintWriter out = new PrintWriter(outgoing.getOutputStream(),true);
        BufferedReader in = new BufferedReader(new InputStreamReader(outgoing.getInputStream()));
        out.println(service);
        boolean done=false;
        while(!done){
            long timeEnd=System.currentTimeMillis();
            if(in.ready()){
        String connResp = in.readLine();

        if (connResp.equals("yes!")){
            connResp = in.readLine();
            String[] userinfo=connResp.split(" ");
            adduser(userinfo[0],userinfo[1],userinfo[2],userinfo[3],userinfo[4],userinfo[5]);
            counter++;
            found=true;
        }
        out.println("BYE");
        done=true;
       }if((timeEnd-timeStart)>2000){done=true;
            out.println("BYE");}
        }
        outgoing.close();
      }
       catch(IOException exception) {

       }
      }
```

7.4 Onineusersmanual

Ο παρακάτω κώδικας υλοποιεί την υπηρεσία public chat σε περίπτωση που ο χρήστης έχει επιλέξει να μην συνδεθεί με εξυπηρετητή.

```
public class Comm extends Thread
{
      public void run()
      {
            try
            { int i = 1;     s = new ServerSocket(8189);
          for (;;)
            { Socket incoming = s.accept( );
           Thread t = new ThreadedEchoHandler(incoming, i);
            t.start();
             i++;     }
       }  catch (Exception e) { e.printStackTrace();System.out.println("skata"); }
     }}



class ThreadedEchoHandler extends Thread
 {
    public ThreadedEchoHandler(Socket i, int c)         {  incoming = i; counter = c;  }
    public void run()
      {  try
         {
           PrintWriter out = new PrintWriter (incoming.getOutputStream(), true /* autoFlush*/);
           BufferedReader in = new BufferedReader(new InputStreamReader(incoming.getInputStream()));
           out.println("send...");
           String message=in.readLine();


           if(message.equals("LOGOUT")){
                String[] userout=in.readLine().split(" ");
                removeuser(userout[0],userout[3]);
           }else if(message.equals("LOGIN")){
                String userintemp=in.readLine();
                String[] userin=userintemp.split(" ");
                adduser(userin[0],userin[1],userin[2],userin[3],userin[4],userin[5]);
                runOnUiThread(new InAction(userin[0]+" logged in chat!"));
           }else if (message.equals("Android_Msn?onlinechat")){
                out.println("yes!");
                out.println(myinformation);
           }else{
                String[] cutip=message.split("/msg");
                message=cutip[1];
                runOnUiThread(new InAction(message));
           }
           incoming.close();


         } catch (Exception e){  e.printStackTrace();  }
      }
    private Socket incoming;    private int counter;
}


void sendmessage(String message){
            int counter=1;
            int temport=8189;
            UsersDbAdapter mDbHelper;
            mDbHelper = new UsersDbAdapter(this);
mDbHelper.open();

Cursor users=mDbHelper.fetchAllusers();
if (users != null) {
   users.moveToFirst();


while (!users.isLast()){
      if(counter!=1){
            String tempusername=users.getString(
     users.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));
```

```
String tempip=users.getString(
      users.getColumnIndexOrThrow(UsersDbAdapter.KEY_IP));


        try {
   Socket outgoing = new Socket();
                              outgoing.connect(new InetSocketAddress(tempip, temport), 500); //na allaksw tin port se 8189
                        out = new PrintWriter( new BufferedWriter( new
                                                        OutputStreamWriter(outgoing.getOutputStream())),true);
    in = new BufferedReader(new InputStreamReader(outgoing.getInputStream()));

    out.println(message);
    outgoing.close();
                    }catch(IOException exception) {}



}users.moveToNext();counter++; //temport--; //profanws stin teliki ilopoiisi de xreiazetai to temport--;
}
if(counter!=1){
String tempusername=users.getString(
      users.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));
String tempip=users.getString(
      users.getColumnIndexOrThrow(UsersDbAdapter.KEY_IP));

try {
   Socket outgoing = new Socket();
   outgoing.connect(new InetSocketAddress(tempip, temport), 500);
   out = new PrintWriter( new BufferedWriter( new OutputStreamWriter(outgoing.getOutputStream())),true);
   in = new BufferedReader(new InputStreamReader(outgoing.getInputStream()));

    out.println(message);
    outgoing.close();
                    }catch(IOException exception) {}

}
       }mDbHelper.close();

       }
       void sendnotification(String reason, String info){
                    int counter=1;
                    int temport=8189;
                    UsersDbAdapter mDbHelper;
                    mDbHelper = new UsersDbAdapter(this);
mDbHelper.open();

Cursor users=mDbHelper.fetchAllusers();
if (users != null) {
   users.moveToFirst();
   while (!users.isLast()){
       if(counter!=1){
                    String tempusername=users.getString(
      users.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));
String tempip=users.getString(
      users.getColumnIndexOrThrow(UsersDbAdapter.KEY_IP));


        try {
   Socket outgoing = new Socket();
   outgoing.connect(new InetSocketAddress(tempip, temport), 500); //na allaksw tin port se 8189
    out = new PrintWriter( new BufferedWriter( new OutputStreamWriter(outgoing.getOutputStream())),true);
    in = new BufferedReader(new InputStreamReader(outgoing.getInputStream()));
    out.println(reason);
    out.println(info);
    outgoing.close();
     }catch(IOException exception) {}

}users.moveToNext();counter++;  //temport--; //profanws stin teliki ilopoiisi de xreiazetai to temport--;
}
if(counter!=1){
String tempusername=users.getString(
      users.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));
String tempip=users.getString(
      users.getColumnIndexOrThrow(UsersDbAdapter.KEY_IP));
```

```
try {
   Socket outgoing = new Socket();
   outgoing.connect(new InetSocketAddress(tempip, temport), 500); //na allaksw tin port se 8189
   out = new PrintWriter( new BufferedWriter( new  OutputStreamWriter(outgoing.getOutputStream())),true);
   in = new BufferedReader(new InputStreamReader(outgoing.getInputStream()));

    out.println(reason);
    out.println(info);
    outgoing.close();
                }catch(IOException exception) {}
    }
    }mDbHelper.close();
}
```

7.5 Onlineusers

Ο παρακάτω κώδικας υλοποιεί την υπηρεσία public chat σε περίπτωση που ο χρήστης έχει επιλέξει να συνδεθεί με τη βοήθεια του εξυπηρετητή.

```
public class Comm extends Thread
{
      public void run()
      {
            try
            { int i = 1;    s = new ServerSocket(8189);
        for (;;)
            { Socket incoming = s.accept( );
              Thread t = new ThreadedEchoHandler(incoming, i);
              t.start();
              i++;      }
      }  catch (Exception e) { e.printStackTrace();System.out.println("skata"); }
     }}



class ThreadedEchoHandler extends Thread
 {
    public ThreadedEchoHandler(Socket i, int c)         {  incoming = i; counter = c;  }
    public void run()
      {  try
         {
           PrintWriter out = new PrintWriter (incoming.getOutputStream(), true /* autoFlush*/);
           BufferedReader in = new BufferedReader(new InputStreamReader(incoming.getInputStream()));
           out.println("send...");
           String message=in.readLine();

           if(message.equals("LOGOUT")){
                String[] userout=in.readLine().split(" ");
                removeuser(userout[0],userout[3]);
           }else if(message.equals("LOGIN")){
                String userintemp=in.readLine();
                String[] userin=userintemp.split(" ");
                adduser(userin[0],userin[1],userin[2],userin[3],userin[4],userin[5]);
                runOnUiThread(new InAction(userin[0]+" logged in chat!"));
           }else
           {
                String[] cutip=message.split("/msg");
              message=cutip[1];
                runOnUiThread(new InAction(message));
           }
              incoming.close();


       } catch (Exception e){  e.printStackTrace();  }
     }
    private Socket incoming;    private int counter;
}

void sendmessage(String message){
            int counter=1;
            int temport=8189;
```

```
                        UsersDbAdapter mDbHelper;
                        mDbHelper = new UsersDbAdapter(this);
      mDbHelper.open();

      Cursor users=mDbHelper.fetchAllusers();
      if (users != null) {
         users.moveToFirst();

      while (!users.isLast()){
            if(counter!=1){
                        String tempusername=users.getString(
            users.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));
      String tempip=users.getString(
            users.getColumnIndexOrThrow(UsersDbAdapter.KEY_IP));

             try {
         Socket outgoing = new Socket();
         outgoing.connect(new InetSocketAddress(tempip, temport), 500); //na allaksw tin port se 8189
         out = new PrintWriter( new BufferedWriter( new OutputStreamWriter(outgoing.getOutputStream())),true);
          in = new BufferedReader(new InputStreamReader(outgoing.getInputStream()));
          out.println(message);
          outgoing.close();
                        }catch(IOException exception) {}

      }users.moveToNext();counter++;
      }
      if(counter!=1){
      String tempusername=users.getString(
            users.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));
      String tempip=users.getString(
            users.getColumnIndexOrThrow(UsersDbAdapter.KEY_IP));

      try {
         Socket outgoing = new Socket();
                                    outgoing.connect(new InetSocketAddress(tempip, temport), 500); //na allaksw tin port se 8189
                              out = new PrintWriter( new BufferedWriter( new
                                                      OutputStreamWriter(outgoing.getOutputStream())),true);
          in = new BufferedReader(new InputStreamReader(outgoing.getInputStream()));
         out.println(message);
          outgoing.close();
                        }catch(IOException exception) {}
      }
     }mDbHelper.close();
     }

void sendnotification(String reason, String info){
                        int counter=1;
                        int temport=8189;
                        UsersDbAdapter mDbHelper;
                        mDbHelper = new UsersDbAdapter(this);
      mDbHelper.open();

      Cursor users=mDbHelper.fetchAllusers();
      if (users != null) {
         users.moveToFirst();

      while (!users.isLast()){
            if(counter!=1){
                        String tempusername=users.getString(
            users.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));
      String tempip=users.getString(
            users.getColumnIndexOrThrow(UsersDbAdapter.KEY_IP));

             try {
         Socket outgoing = new Socket();
         outgoing.connect(new InetSocketAddress(tempip, temport), 500);
         out = new PrintWriter( new BufferedWriter( new OutputStreamWriter(outgoing.getOutputStream())),true);
          in = new BufferedReader(new InputStreamReader(outgoing.getInputStream()));
          out.println(reason);
          out.println(info);
          outgoing.close();
}catch(IOException exception) {}

      }users.moveToNext();counter++; // temport--; //profanws stin teliki ilopoiisi de xreiazetai to temport--;
      }
```

```
      if(counter!=1){
      String tempusername=users.getString(
            users.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));
      String tempip=users.getString(
            users.getColumnIndexOrThrow(UsersDbAdapter.KEY_IP));

      try {
         Socket outgoing = new Socket();
         outgoing.connect(new InetSocketAddress(tempip, temport), 500);
         out = new PrintWriter( new BufferedWriter( new OutputStreamWriter(outgoing.getOutputStream())),true);
          in = new BufferedReader(new InputStreamReader(outgoing.getInputStream()));
          out.println(reason);
          out.println(info);
          outgoing.close();
                        }catch(IOException exception) {}

      }
}mDbHelper.close();
  }
```

7.6 Serverconnect

Στο παρών κώδικα υλοποιείται η λειτουργία όπου ο χρήστης επιλέγει και ενημερώνεται από τον εξυπηρετητή για τις υπηρεσίες και τους χρήστες που είναι συνδεδεμένοι στις υπηρεσίες.

```
InetAddress serverAddr = InetAddress.getByName(serverip);
socket = new Socket();
 socket.connect(new InetSocketAddress(serverAddr, 8189), 1000);

              String messageOut = "login "+username+" "+name+" "+lastname+" "+myip+" "+Lat+" "+Lng;
               out = new PrintWriter( new BufferedWriter( new OutputStreamWriter(socket.getOutputStream())),true);
              in=new BufferedReader(new InputStreamReader(socket.getInputStream()));
              out.println(messageOut);
              String messageIn=in.readLine();
              messageIn=in.readLine();
              if (messageIn.equals("login_OK")){
                            messageOut="services_request";
                            out.println(messageOut);
                            messageIn=in.readLine();
                            if (messageIn.equals("0"))servermessages.setText("No services available");
                            else {
                            servermessages.setText("Services available "+messageIn);
                            int temp=Integer.parseInt(messageIn);
                           for (int i=1;i<temp+1;i++){
                            try{
                            messageIn=in.readLine();
                            if(messageIn.equals("1 chatting"))
                            chatting.setText(messageIn);
                            else if(messageIn.equals("2 dataexchange"))
                            dataexchange.setText(messageIn);
                            else if(messageIn.equals("3 onlinechat"));
                            onlinechat.setText(messageIn);
                            }
                            catch(IOException exception) {
                            servermessages.setText("No services available");
                            }
                            }
              chatting.setOnClickListener(new View.OnClickListener(){
                            public void onClick(View view) {
                            servermessages.setTextColor(Color.RED);
                            servermessages.setText("You have choosen Chatting..Please wait");
                            userprofile("chatting_login");
                            servermessages.setText("Online Users retrieved, please wait to process");
                            chatlogin();
                            finish();
                            }
                            });
              dataexchange.setOnClickListener(new View.OnClickListener(){
                            public void onClick(View view) {
                            servermessages.setTextColor(Color.GREEN);
                            servermessages.setText("You have choosen dataexchange..Please wait");
```

```
                        out.println("logout "+username+" "+lastname);
                        out.println("BYE");
                        dataexchangelogin();
                        finish();
                        }
                        });

                onlinechat.setOnClickListener(new View.OnClickListener(){
                        public void onClick(View view) {
                        servermessages.setTextColor(Color.YELLOW);
                        servermessages.setText("You have choosen onlinechat..Please wait");
                        userprofile("onlinechat_login");
                        onlinelogin();
                        finish();
                        }
                       });
                        }}
                else {
                       servermessages.setTextColor(Color.RED);
                       servermessages.setTextSize(15);
                       servermessages.setText("No Valid Data sent..!");
                }
}

   catch(IOException exception) {
                servermessages.setTextColor(Color.RED);
                        servermessages.setTextSize(15);
                servermessages.setText("No server connection! Please choose another way to communicate");
   }
  }
```

7.7 Showreceivingfiles

Παρακάτω δίνεται ο κώδικας οποίος υλοποιεί την υπηρεσία ανταλλαγής δεδομένων στο κινητό τηλέφωνο.

```
try {
                InetAddress serverAddr = InetAddress.getByName(serverip);
                Socket socket2 = new Socket();
                socket2.connect(new InetSocketAddress(serverAddr, 8189), 1000);
                String messageOut="login "+myinformation;
                out2 = new PrintWriter( new BufferedWriter( new  OutputStreamWriter(socket2.getOutputStream())),true);
                in2=new BufferedReader(new InputStreamReader(socket2.getInputStream()));
                out2.println(messageOut);
                String messageIn2=in2.readLine();
                messageIn2=in2.readLine();
                if (messageIn2.equals("login_OK")){
                messageOut="dataexchange_request";
                out2.println(messageOut);
                messageIn2=in2.readLine();
                int total=Integer.parseInt(messageIn2);
                String prevuserip=" ";
                for(int i=0;i<total;i++){
                messageIn2=in2.readLine();
                String[] data=messageIn2.split(" ");
                String[] path=data[1].split("/");
                int count=path.length;
                String filename=path[count-1];
                mDbHelper.addreceivefile(filename,data[1],data[2]);
                if(!(prevuserip.equals(data[2]))){
                prevuserip=data[2];
                mDbHelper.createUser(data[0]," "," ",data[2]," ",data[3],data[4]);
                 }}}

                 if(addedfile)
                  try{
                 mNotesCursor = mDbHelper.fetchAllsendfiles();
                 if (mNotesCursor != null) {
                 mNotesCursor.moveToFirst();
                 while(!mNotesCursor.isLast()){
                 messageOut="data_upload "+mNotesCursor.getString(mNotesCursor.getColumnIndex("name"));
                 out2.println(messageOut);
```

```
                mNotesCursor.moveToNext();
                }
                messageOut="data_upload "+mNotesCursor.getString(mNotesCursor.getColumnIndex("name"));
                out2.println(messageOut);
                }
               }catch( SQLException exception) {}
                out2.println("BYE");
                 out2.close();
                 in2.close();
                socket2.close();
                 }catch(IOException exception) {
              int duration = Toast.LENGTH_SHORT;
              Toast t = new Toast(showreceivingfiles.this);
              t.makeText(showreceivingfiles.this, "No Server Available",duration).show();
                }

if(addedfile)informusers();
fillData();
new Comm().start();
}

@Override
protected void onListItemClick(ListView l, View v, int position, long id) {
super.onListItemClick(l, v, position, id);
Cursor c = mNotesCursor;
c.moveToPosition(position);
String message="busy..";
String file=mNotesCursor.getString(mNotesCursor.getColumnIndex("path"));
if(!file.equals("Available files to receive")){
 String ip=mNotesCursor.getString(mNotesCursor.getColumnIndex("ip"));
String[] path=file.split("/");
int count=path.length;
String filename=path[count-1];
String save="/sdcard/"+filename;

 try{
      Socket socketreceiv = new Socket();
      socketreceiv.connect(new InetSocketAddress(ip, 8189), 1000);
      PrintWriter out1 = new PrintWriter (socketreceiv.getOutputStream(), true /* autoFlush*/);
      BufferedReader in1 = new BufferedReader(new InputStreamReader(socketreceiv.getInputStream()));
      out1.println("receivefile");
      out1.println(file);
      message=in1.readLine();
      out1.close();
      socketreceiv.close();
   }catch(IOException exception) {
mDbHelper.deletereceivefile(ip);
mDbHelper.deleteUserbyip(ip);
runOnUiThread(new InAction(" "));
   }
  try {
      Thread.sleep(1000);
       } catch (InterruptedException e1) {
                        e1.printStackTrace();
                        }

if(message.equals("ok!"))
                    try{
                     new Client(ip,1500, save).connect();
                    } catch (IOException e) {
                        int duration = Toast.LENGTH_LONG;
                        Toast t = new Toast(showreceivingfiles.this);
                        t.makeText(showreceivingfiles.this, "not able to receive file..",duration).show();
                                }
                    }
}

public class Comm extends Thread
            {
                public void run()
                {
                    try
                    { int i = 1;    ServerSocket s = new ServerSocket(8189);
                   for (;;)
                    { Socket incoming = s.accept( );
```

```
                    Thread t = new ThreadedEchoHandler(incoming, i);
                    t.start();
                     i++;     }
                  }  catch (Exception e) { e.printStackTrace();System.out.println("skata"); }
                }}


class ThreadedEchoHandler extends Thread
            {
               public ThreadedEchoHandler(Socket i, int c)        {  incoming = i; counter = c;  }
               public void run()
                {  try
                   {
                    PrintWriter out = new PrintWriter (incoming.getOutputStream(), true /* autoFlush*/);
                    BufferedReader in = new BufferedReader(new InputStreamReader(incoming.getInputStream()));
                    out.println("causeofcontact?");
                     String inmessage= in.readLine();
                     if(inmessage.equals("receivefile")){
                         inmessage= in.readLine();
                         out.println("ok!");
                         out.close();
                         in.close();
                     try {
                         new Server(1500,inmessage).listen();
                         }
                     catch (IOException e) {
                                    int duration = Toast.LENGTH_LONG;
                                    Toast t = new Toast(showreceivingfiles.this);
                                    t.makeText(showreceivingfiles.this, "not able to send file",duration).show();
                                            }
                   }else if(inmessage.equals("new_user")){
                            boolean firstfile=true;
                            inmessage= in.readLine();
                            while(!inmessage.equals("END")){
                                    String[] data=inmessage.split(" ");
                                    String[] path=data[1].split("/");
                                    int count=path.length;
                                    String filename=path[count-1];
                                    mDbHelper.addreceivefile(filename,data[1],data[2]);
                                    if(firstfile){mDbHelper.createUser(data[0]," "," ",data[2]," ",data[3],data[4]);
                                    firstfile=false;
                                    }
                                    inmessage=in.readLine();
                                    runOnUiThread(new InAction(" "));
                                    }
                           }
                     incoming.close();
                  } catch (Exception e){  e.printStackTrace();  }
                }
               private Socket incoming;    private int counter;
            }


private void fillData() {

      mNotesCursor = mDbHelper.fetchAllreceivefiles();
      startManagingCursor(mNotesCursor);
      String[] from = new String[]{"name"};
      int[] to = new int[]{R.id.reveivefilesrow};
      SimpleCursorAdapter notes =  new SimpleCursorAdapter(this, R.layout.receivefile_row, mNotesCursor, from, to);
     setListAdapter(notes);
   }

public class Client {

            private String ip;
            private int port;
            private String fileName;
            public Client(String ip,int port, String fileName) {
            this.ip=ip;
            this.port = port;
```

```
		this.fileName = fileName;
		}
		private void connect() throws UnknownHostException, IOException {
		BufferedInputStream in = null;
		BufferedOutputStream out = null;
		Socket socket = null;
		try {
		socket = new Socket();
		socket.connect(new InetSocketAddress(ip, port), 500);
		in = new BufferedInputStream(socket.getInputStream());
		File f = new File(fileName);
		out = new BufferedOutputStream(
		new FileOutputStream(f));
		byte[] b = new byte[256];
		int read = -1;
		while ((read = in.read(b)) >= 0) {
		out.write(b, 0, read);
		}
		received();
		} finally {
		try {
		in.close();
		} catch (Exception e) {
		}
		try {
		out.close();
		} catch (Exception e) {
		}
		try {
		socket.close();
		} catch (Exception e) {
		}
		}
		}
}

void received(){

		int duration = Toast.LENGTH_LONG;
		Toast t = new Toast(showreceivingfiles.this);
		t.makeText(showreceivingfiles.this, "file received!",duration).show();
		}

public class Server {

		private int port;
		private String fileName;
		public Server(int port, String fileName) {
		this.port = port;
		this.fileName = fileName;
		}
		private void listen() throws IOException {
		BufferedInputStream in = null;
		BufferedOutputStream out = null;
		Socket client = null;
		try {
		ServerSocket socket = new ServerSocket(port);
		client = socket.accept();
		out = new BufferedOutputStream(client.getOutputStream());
		File f = new File(fileName);
		in = new BufferedInputStream(new FileInputStream(f));
		byte[] b = new byte[256];
		int read = -1;
		while ((read = in.read(b)) >= 0) {
		out.write(b, 0, read);
		}
		socket.close();
		} finally {
		try {
		in.close();
		} catch (Exception e) {}
		try {
		out.close();
		} catch (Exception e) {}
		try {
```

```
            client.close();
            } catch (Exception e) {}
            }
            }

}

void informusers(){
            int temport=8189;
            String tempmessageout;
Cursor users=mDbHelper.fetchAllusers();

            if (users != null) {
            users.moveToFirst();

            while(!users.isLast()){
            String usersip=users.getString(users.getColumnIndex(UsersDbAdapter.KEY_IP));
            try{
            Socket tempsocket = new Socket();
            tempsocket.connect(new InetSocketAddress(usersip, temport), 500);
            PrintWriter tempout = new PrintWriter( new BufferedWriter( new
                                                            OutputStreamWriter(tempsocket.getOutputStream())),true);
            tempout.println("new_user");
            Cursor mNotesCursor2 = mDbHelper.fetchAllsendfiles();
            if (mNotesCursor2 != null) {
            mNotesCursor2.moveToFirst();
            while(!mNotesCursor2.isLast()){
            tempmessageout=myusername+" "+mNotesCursor2.getString(mNotesCursor2.getColumnIndex("name"))+" "+myip+"
"+Lat+" "+Lng;
            tempout.println(tempmessageout);
            mNotesCursor2.moveToNext();
            }
            tempmessageout=myusername+" "+mNotesCursor2.getString(mNotesCursor2.getColumnIndex("name"))+" "+myip+"
"+Lat+" "+Lng;
            tempout.println(tempmessageout);
            tempout.println("END");
            }
            tempsocket.close();
            }catch(IOException exception) {}
            users.moveToNext();
            }
            try{
            String usersip=users.getString(users.getColumnIndex(UsersDbAdapter.KEY_IP));
            Socket tempsocket = new Socket();
            tempsocket.connect(new InetSocketAddress(usersip, temport), 500);
            PrintWriter tempout = new PrintWriter( new BufferedWriter( new
                                                            OutputStreamWriter(tempsocket.getOutputStream())),true);
            BufferedReader tempin = new BufferedReader(new InputStreamReader(tempsocket.getInputStream()));
            tempout.println("new_user");
            Cursor mNotesCursor2 = mDbHelper.fetchAllsendfiles();
            if (mNotesCursor2 != null) {
                        mNotesCursor2.moveToFirst();
                        while(!mNotesCursor2.isLast()){
                        tempmessageout=myusername+" "+mNotesCursor2.getString(mNotesCursor2.getColumnIndex("name"))+"
"+myip+" "+Lat+" "+Lng;
                        tempout.println(tempmessageout);
                        mNotesCursor2.moveToNext();
                        }
                        tempmessageout=myusername+" "+mNotesCursor2.getString(mNotesCursor2.getColumnIndex("name"))+"
"+myip+" "+Lat+" "+Lng;
                        tempout.println(tempmessageout);
                        tempout.println("END");
                }
               tempsocket.close();
            }catch(IOException exception) {}
            }
            }
```

7.8 Userlist

Ο κώδικας που υλοποιεί την υπηρεσία του private chatting όταν ο χρήστης είναι συνδεδεμένος στον εξυπηρετητή.

```
@Override
  protected void onListItemClick(ListView l, View v, int position, long id) {
            super.onListItemClick(l, v, position, id);
     Cursor c = mUsersCursor;
     c.moveToPosition(position);
     String usercheck=c.getString(
          c.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));
      if(!usercheck.equals("Users...")){
     Intent i = new Intent(this, chattingMe.class);
     i.putExtra(UsersDbAdapter.KEY_ROWID, id);

     i.putExtra(UsersDbAdapter.KEY_IP, c.getString(
          c.getColumnIndexOrThrow(UsersDbAdapter.KEY_IP)));

     String user=c.getString(
          c.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));

     String datastore=mDbHelper.messages(user);
     if(!datastore.equals(" ")){
             i.putExtra("data", datastore);
             String[] changenametonormal=user.split( "/" );
             Long userid= mDbHelper.fetchuser3(user);
             String tempname=mDbHelper.fetchuser4(userid);
        String templastname=mDbHelper.fetchuser5(userid);
        String tempip=mDbHelper.fetchuser6(userid);

             mDbHelper.updateUser(id, changenametonormal[0],tempname,templastname,tempip," "," "," ");
     }
     else i.putExtra("data", " ");
     i.putExtra("myname", myusername);
     i.putExtra("myip", myip);
     onchat=true;
     startActivityForResult(i, CHAT_END);
      }
  }

            public class Comm extends Thread
            {
                  public void run()
                  {
                        try
                        { int i = 1;     s2 = new ServerSocket(8189);
                     for (;;)
                        { Socket incoming = s2.accept( );
                         System.out.println("Spawning " + i);


                         Thread t = new ThreadedEchoHandler(incoming, i);
                         t.start();
                          i++;      }
                    }  catch (Exception e) { e.printStackTrace();System.out.println("skata"); }
                  }}



            class ThreadedEchoHandler extends Thread
             {
                public ThreadedEchoHandler(Socket i, int c)         {  incoming = i; counter = c;  }
                public void run()
                  {  try
                     {
                      PrintWriter out = new PrintWriter (incoming.getOutputStream(), true /* autoFlush*/);
                      BufferedReader in = new BufferedReader(new InputStreamReader(incoming.getInputStream()));
                     out.println("causeoftalk?");
                      String contype= in.readLine();
                      if(contype.equals("new_user")){
                          String[] info=in.readLine().split(" ");
                           mDbHelper.createUser(info[0], info[1], info[2], info[3], " ",info[4],info[5]);
                           incoming.close();
```

```
                        runOnUiThread(new InAction(" "));
                    }else if (contype.equals("logout")){
                        String[] info=in.readLine().split(" ");
                         mDbHelper.removeUser(info[0], info[3]);
                        incoming.close();
                        runOnUiThread(new InAction(" "));
                    }
                    else{
                    data[counter]=" ";
                    boolean done=false;
                    boolean firstsending=true;
                    while(!done){
                    if(!onchat){
                        out.println( "yes!" );
                        out.println(port);
                        incoming.close();
                        onchat=true;
                        done=true;
                        chattinglog(counter);
                    }else{if (firstsending)
                        out.println("busy..");
                        firstsending=false;
                        if(in.ready()){
                        String temp=in.readLine();
                        String[] cutip=temp.split("/msg");
                        temp=cutip[1];
                        String[] user=temp.split(":");
                        if(user[1].equals("END_CHAT")){
                        done=true;
                        String[] findip=cutip[0].split("/");
                        Long userid=mDbHelper.retrieveIDuser(user[0],findip[0]);
                        String tempname=mDbHelper.fetchuser4(userid);
                        String templastname=mDbHelper.fetchuser5(userid);
                        String tempip=mDbHelper.fetchuser6(userid);
                        mDbHelper.updateUser(userid, user[0]+"/sent message", tempname,templastname,tempip,data[counter]," ","
");
                        runOnUiThread(new InAction(" "));
                        fillData();
                         incoming.close();
                        }
                        else{
                        data[counter]=data[counter]+"\n"+user[1];
                        }}
                     }
                    } }
                   } catch (Exception e){  e.printStackTrace();  }
                 }
                private Socket incoming;    private int counter;
            }


void informusers(String reason, String info){
            int counter=1;
            String usersip;
            int temport=8189;
            Cursor users=mDbHelper.fetchAllusers();
            if (users != null) {
      users.moveToFirst();

       while(!users.isLast()){
            if(counter!=1){
            usersip=users.getString(users.getColumnIndex(UsersDbAdapter.KEY_IP));

            try{
            Socket tempsocket = new Socket();
            tempsocket.connect(new InetSocketAddress(usersip, temport), 500);
            PrintWriter out2 = new PrintWriter( new BufferedWriter( new
                                                    OutputStreamWriter(tempsocket.getOutputStream())),true);
            out2.println(reason);
            out2.println(info);
            tempsocket.close();
            }catch(IOException exception) {}
            }users.moveToNext();counter++;
            }
```

```
            if(counter!=1){
            try{
            usersip=users.getString(users.getColumnIndex(UsersDbAdapter.KEY_IP));
            Socket tempsocket = new Socket();
            tempsocket.connect(new InetSocketAddress(usersip, temport), 500);
            PrintWriter out2 = new PrintWriter( new BufferedWriter( new
                                                    OutputStreamWriter(tempsocket.getOutputStream())),true);
            out2.println(reason);
            out2.println(info);
            tempsocket.close();
            }catch(IOException exception) {}
            }
            }

}

public boolean onKeyDown(int keyCode, KeyEvent event) {
            if(keyCode == KeyEvent.KEYCODE_BACK){
            informusers("logout",myinformation);
            try {
            InetAddress serverAddr = InetAddress.getByName(serverip);
            Socket socket = new Socket();
            socket.connect(new InetSocketAddress(serverAddr, 8189), 1000);
            String messageOut = "logout "+myusername+" "+mylastname;
            PrintWriter out3 = new PrintWriter( new BufferedWriter( new OutputStreamWriter(socket.getOutputStream())),true);
             out3.println(messageOut);
             socket.close();
            }catch(IOException exception) {}

            try {
            s2.close();
            } catch (IOException e) {
            e.printStackTrace();
            }
            Intent mIntent = new Intent();
            setResult(RESULT_OK, mIntent);
            finish();
            }return super.onKeyDown(keyCode, event);
}
```

7.9 Userlistmanual

Ο κώδικας που υλοποιεί την υπηρεσία του private chatting όταν ο χρήστης δεν έχει επιλέξει να συνδεθεί στον εξυπηρετητή.

```
@Override
   protected void onListItemClick(ListView l, View v, int position, long id) {
            super.onListItemClick(l, v, position, id);
      Cursor c = mUsersCursor;
      c.moveToPosition(position);

      String usercheck=c.getString(
            c.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));
      if(!usercheck.equals("Users...")){

      Intent i = new Intent(this, chattingMe.class);
      i.putExtra(UsersDbAdapter.KEY_ROWID, id);

      i.putExtra(UsersDbAdapter.KEY_IP, c.getString(
            c.getColumnIndexOrThrow(UsersDbAdapter.KEY_IP)));

      String user=c.getString(
            c.getColumnIndexOrThrow(UsersDbAdapter.KEY_USERNAME));

      String datastore=mDbHelper.messages(user);
      if(!datastore.equals(" ")){
            i.putExtra("data", datastore);
            String[] changenametonormal=user.split( "/" );
            Long userid= mDbHelper.fetchuser3(user);
            String tempname=mDbHelper.fetchuser4(userid);
         String templastname=mDbHelper.fetchuser5(userid);
         String tempip=mDbHelper.fetchuser6(userid);
```

```
            mDbHelper.updateUser(id, changenametonormal[0],tempname,templastname,tempip," "," "," ");
            }
      else i.putExtra("data", " ");
      i.putExtra("myname", myusername);
      i.putExtra("myip", myip);
      onchat=true;
      startActivityForResult(i, CHAT_END);
      }
   }
public class Comm extends Thread
            {
                  public void run()
                  {
                        try
                        { int i = 1;     s2 = new ServerSocket(8189);
                     for (;;)
                        { Socket incoming = s2.accept( );

                         System.out.println("Spawning " + i);


                         Thread t = new ThreadedEchoHandler(incoming, i);
                         t.start();
                          i++;     }
                    }  catch (Exception e) { e.printStackTrace();System.out.println("skata"); }
                  }}



            class ThreadedEchoHandler extends Thread
             {
                public ThreadedEchoHandler(Socket i, int c)         {  incoming = i; counter = c;  }
                public void run()
                  {  try
                     {
                       PrintWriter out = new PrintWriter (incoming.getOutputStream(), true /* autoFlush*/);
                       BufferedReader in = new BufferedReader(new InputStreamReader(incoming.getInputStream()));

                      out.println("causeoftalk?");
                       String contype= in.readLine();
                       if(contype.equals("new_user")){
                            String[] info=in.readLine().split(" ");
                            mDbHelper.createUser(info[0], info[1], info[2], info[3]," ",info[4],info[5]);
                            incoming.close();
                            runOnUiThread(new InAction(" "));
                       }else if (contype.equals("logout")){
                            String[] info=in.readLine().split(" ");
                            mDbHelper.removeUser(info[0], info[3]);
                            incoming.close();
                            runOnUiThread(new InAction(" "));
                       }else if (contype.equals("Android_Msn?chatting")){
                            out.println("yes!");
                            out.println(myinformation);
                            contype= in.readLine();
                            incoming.close();
                            }
                       else{
                       data[counter]=" ";
                       boolean done=false;
                       boolean firstsending=true;
                       while(!done){
                       if(!onchat){
                            out.println( "yes!" );
                            out.println(port);
                            incoming.close();
                            onchat=true;
                            done=true;
                            chattinglog(counter);
                       }else{if (firstsending){
                            out.println("busy..");
                       }
                       firstsending=false;
                            if(in.ready()){
```

```
                        String temp=in.readLine();
                        String[] cutip=temp.split("/msg");
                        temp=cutip[1];
                        String[] user=temp.split(":");
                        if(user[1].equals("END_CHAT")){
                        done=true;
                        String[] findip=cutip[0].split("/");
                        Long userid=mDbHelper.retrieveIDuser(user[0],findip[0]);
                        String tempname=mDbHelper.fetchuser4(userid);
                        String templastname=mDbHelper.fetchuser5(userid);
                        String tempip=mDbHelper.fetchuser6(userid);
                        mDbHelper.updateUser(userid, user[0]+"/sent message", tempname,templastname,tempip,data[counter]," ","
");
                        runOnUiThread(new InAction(" "));
                        fillData();
                        incoming.close();
                        }
                        else{
                        data[counter]=data[counter]+"\n"+user[1];
                        }}
                    }
                     } }
                  } catch (Exception e){  e.printStackTrace();  }
                }
                private Socket incoming;    private int counter;
        }


void informusers(String reason, String info){
        int counter=1;
        String usersip;
        int temport=8189;
        Cursor users=mDbHelper.fetchAllusers();
        if (users != null) {
        users.moveToFirst();

        while(!users.isLast()){
                if(counter!=1){
                usersip=users.getString(users.getColumnIndex(UsersDbAdapter.KEY_IP));

                try{
                Socket tempsocket = new Socket();
                tempsocket.connect(new InetSocketAddress(usersip, temport), 500);
                PrintWriter out2 = new PrintWriter( new BufferedWriter( new
                                                        OutputStreamWriter(tempsocket.getOutputStream())),true);
                out2.println(reason);
                out2.println(info);
                tempsocket.close();
                }catch(IOException exception) {}
                }users.moveToNext();counter++;  //temport--; //profanws stin teliki ilopoiisi de xreiazetai to temport--;
                }

        if(counter!=1){
                try{
                usersip=users.getString(users.getColumnIndex(UsersDbAdapter.KEY_IP));
                Socket tempsocket = new Socket();
                tempsocket.connect(new InetSocketAddress(usersip, temport), 500);
                PrintWriter out2 = new PrintWriter( new BufferedWriter( new
                                                        OutputStreamWriter(tempsocket.getOutputStream())),true);
                out2.println(reason);
                out2.println(info);
                tempsocket.close();
                }catch(IOException exception) {}
        }
        }
}

public boolean onKeyDown(int keyCode, KeyEvent event) {
        if(keyCode == KeyEvent.KEYCODE_BACK){
                informusers("logout",myinformation);
                try {
                s2.close();
                } catch (IOException e) {
                        e.printStackTrace();
                }
```

```
                    Intent mIntent = new Intent();
                    setResult(RESULT_OK, mIntent);
                    finish();
            }return super.onKeyDown(keyCode, event);
            }
```

7.10 getLocalIpAddress

Η παρακάτω διαδικασία-procedure μας δίνει την IP-Address του κινητού τηλεφώνου ανεξαρτήτου του τρόπου σύνδεσης στο διαδίκτυο(3G,wifi)

```
public String getLocalIpAddress() {
            try {
                for (Enumeration<NetworkInterface> en = NetworkInterface.getNetworkInterfaces(); en.hasMoreElements();) {
                    NetworkInterface intf = en.nextElement();
                    for (Enumeration<InetAddress> enumIpAddr = intf.getInetAddresses(); enumIpAddr.hasMoreElements();) {
                        InetAddress inetAddress = enumIpAddr.nextElement();
                        if (!inetAddress.isLoopbackAddress()) {
                            return inetAddress.getHostAddress().toString();
                        }
                    }
                }
            } catch (SocketException ex) {

            }
            return null;
        }
```