Ατομική Διπλωματική Εργασία

ΥΛΟΠΟΙΗΣΗ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ ΓΙΑ ΤΗΝ ΕΥΡΕΣΗ ΚΟΝΤΙΝΩΝ ΚΙΝΗΤΩΝ ΣΥΣΚΕΥΩΝ

Θεόδωρος Ιωάννου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2011

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΥΛΟΠΟΙΗΣΗ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ ΓΙΑ ΤΗΝ ΕΥΡΕΣΗ ΚΟΝΤΙΝΩΝ ΚΙΝΗΤΩΝ ΣΥΣΚΕΥΩΝ

Θεόδωρος Ιωάννου

Επιβλέπων Καθηγητής
Δρ. Γιώργος Σαμάρας

Συν-επιβλέπων Καθηγητής
Δρ. Δημήτρης Ζεϊναλιπούρ

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2011

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τους επιβλέποντες καθηγητές μου Δρ. Γιώργο Σαμάρα και Δρ. Δημήτρη Ζεϊναλιπούρ, οι οποίοι μου έδωσαν την ευκαιρία να εργαστώ πάνω στο αντικείμενο της παρούσας εργασίας, το θέμα της οποίας είναι πολλά υποσχόμενο και ίσως κάποια μέρα να το δούμε να χρησιμοποιείται ευρέως. Με την καθοδήγηση και τη πολύτιμη βοήθεια τους, κατάφερα να ολοκληρώσω την εργασία αυτή.

Θέλω να τους ευχαριστήσω επίσης, για την εμπιστοσύνη που έδειξαν προς το πρόσωπο μου, σε κάποιες περιόδους αδυναμίας που υπήρξαν εκ μέρους μου, καθ' όλη την διάρκεια της χρονιάς, συνεχίζοντας να με στηρίζουν για να προχωρήσω με την εκπόνηση της εργασίας μου.

Θα ήθελα ακόμη να ευχαριστήσω τον Γεώργιο Χατζημιλιούδη για την συνεργασία μας και τη μεγάλη του βοήθεια που μου προσέφερε, μέσω πολλών συμβουλών αλλά και μέσω ενός προγράμματος που μου έδωσε του οποίου μερικός αλγόριθμος και κάποια αποτελέσματα χρησιμοποιούνται στο δικό μου πρόγραμμα.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια μου, η οποίοι ήταν πάντα δίπλα μου, έστω και νοερά, αφού όλους αυτούς τους μήνες τους στέρησα την παρουσία μου στο σπίτι, μιας βρισκόμουν σχεδόν επί μονίμου βάσεως στο αγαπημένο μου εργαστήριο 216, το οποίο έγινε το δεύτερο μου σπίτι.

Περίληψη

Ο αυξανόμενος αριθμός των αγορών σε κινητές συσκευές τα τελευταία χρόνια, αλλά και η εξέλιξη που είχαν, την τελευταία πενταετία, ωθεί και τους ερευνητές στο να στρέψουν το ενδιαφέρον τους προς τέτοιες συσκευές. Η αλήθεια είναι ότι τα κινητά τηλέφωνα έχουν γίνει ένα αναγκαίο κακό στις μέρες μας. Σχεδόν ο κάθε άνθρωπος, περισσότερο στα νεανικά πλήθη και λιγότερο στις μεγαλύτερες ηλικίες, έχει στην κατοχή του ένα κινητό τηλέφωνο. Αυτός ο παράγοντας, σε συνδυασμό με τις ανεπτυγμένες δυνατότητες που προσφέρουν τα σημερινά προηγμένα κινητά, τα λεγόμενα “Smart phones”, έχει καταστήσει τα κινητά την πρώτη σε ζήτηση πλατφόρμα για να φτιάξει κάποιος μία εφαρμογή, η οποία θα σχετίζεται με τις πράξεις του χρήστη του κινητού τηλεφώνου, κάθε στιγμή της μέρας του. Μία τέτοια υπηρεσία, είναι και η εύρεση κοντινών κινητών συσκευών, την οποία θα αναλύσουμε και θα δείξουμε την δική μας προτεινόμενη αρχιτεκτονική, για την παροχή της προς τους χρήστες.

Η υπηρεσία που θα παρέχουμε, η οποία αναπτύσσεται σε αυτή την διπλωματική εργασία, θα παρέχει στους χρήστες της την πληροφορία με τους κοντινότερους Κ άλλους χρήστες κινητών συσκευών, οι οποίοι βρίσκονται γεωγραφικά κοντά στη δική μας συσκευή. Εκτός από την εφαρμογή η οποία θα συμπεριφέρεται σαν εξυπηρετητής, αναπτύξαμε και μια εφαρμογή για κινητά τηλέφωνα Android, η οποία παρουσίαζε αυτά τα δεδομένα για τις γειτονικές κινητές συσκευές, αποσκοπώντας στην αποστολή μηνυμάτων στις γειτονικές συσκευές οι οποίες μας περιβάλλουν.

Επίσης, στο τέλος της εργασίας αυτής, έχουμε πειραματική μελέτη η οποία βάζει σε δοκιμή τον εξυπηρετητή μας, συγκρίνοντας δύο τεχνικές οι οποίες μελετήθηκαν και υλοποιήθηκαν με σκοπό να συγκριθούν. Ακόμα, έχουν αναλυθεί δύο διαφορετικοί αλγόριθμοι οι οποίοι μπορούν να προσφέρουν την υπηρεσία της γειτνίασης, εκ των οποίων υλοποιήσαμε τον ένα και παραθέτουμε τα πειραματικά αποτελέσματα του, όσο αφορά τον χρόνο εκτέλεσης του αλγορίθμου.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Υποκίνηση Εργασίας

Η ευρεία εξάπλωση των «έξυπνων» κινητών συσκευών, οι οποίες υποστηρίζουν εξεύρεση της γεωγραφικής τους θέσης (π.χ. AGPS, WLAN και Cellular tower τοποθέτηση), αλλά επίσης περιέχουν και άλλων ειδών αισθητήρες, με τους οποίους μπορούν να μάθουν στοιχεία για τον χώρο που τις περιβάλλει (π.χ. κάμερα, μικρόφωνο, επιταχυνσιόμετρο, αισθητήρες φωτισμού κ.α.), συνδυαζόμενα με τις δυνατότητες συνδεσιμότητας προς το Διαδίκτυο τις οποίες προσφέρουν, οι οποίες μπορεί να είναι οποιασδήποτε τεχνολογίας 3G, 3.5G, προτεινόμενες 4G τεχνολογίες ή ακόμα και μέσω Wi-Fi, έκαναν όλους, προγραμματιστές, εταιρίες παιχνιδιών, εταιρίες παροχής τηλεφωνίας και Διαδικτύου, αλλά ακόμα και ερευνητές, να ασχολούνται συνεχώς μαζί τους.

Όλοι γνωρίζουμε την μεγάλη επιτυχία της γνωστής εταιρίας Apple, όταν το 2007 έδωσε στην αγορά το γνωστό σε όλους iPhone [3]. Τότε, όλοι οι αντίπαλοί της είχαν παραδεχτεί ότι η Apple τους είχε πιάσει στον ύπνο. Τέσσερα χρόνια μετά, έχουμε μία πληθώρα από εταιρίες (π.χ. Apple, Google, Microsoft, Palm και RIM), οι οποίες έχουν

αναπτύξει το δικό τους λειτουργικό σύστημα για κινητές συσκευές, που είναι εφάμιλλο αλλά ίσως και σε κάποια σημεία καλύτερο από διάφορους υπέρ-υπολογιστές του παρελθόντος.

Έχοντας αναφέρει αυτό, βλέπουμε την εξέλιξη που είχαμε τα τελευταία χρόνια στην τεχνολογία. Από προσωπική εμπειρία, αλλά είναι κάτι που συμβαίνει στους περισσότερους πιστεύω, το κινητό που έχω τώρα στην κατοχή μου, έχει περισσότερη υπολογιστική ισχύ, κύρια μνήμη, αλλά και δυνατότητες, από τον πρώτο υπολογιστή που είχα αγοράσει, 10 - 15 χρόνια περίπου πριν. Επομένως βλέπουμε για ακόμη μία φορά τον λόγο που πολλές εταιρίες στις μέρες μας στρέφονται προς την ανάπτυξη εφαρμογών και παιχνιδιών για τέτοιες κονσόλες, όπως τις «έξυπνες» κινητές συσκευές. Υπήρξαν πολλές έρευνες στο παρελθόν από διάφορους ερευνητές, οι οποίοι έχουν ασχοληθεί με την ανάπτυξη εφαρμογών και προσφορά υπηρεσιών για κινητές συσκευές. Αυτό, σε συνδυασμό με την εξέλιξη που είχαμε τα τελευταία τρία με τέσσερα χρόνια, δηλαδή κυρίως μετά την εμφάνιση του iPhone από την Apple, αλλά και τις υπόλοιπες εταιρίες οι οποίες ακολούθησαν το παράδειγμά τους, ώθησε και εμάς στο να ασχοληθούμε με την ανάπτυξη και προσφορά κάποιας υπηρεσίας σε τέτοιες συσκευές.

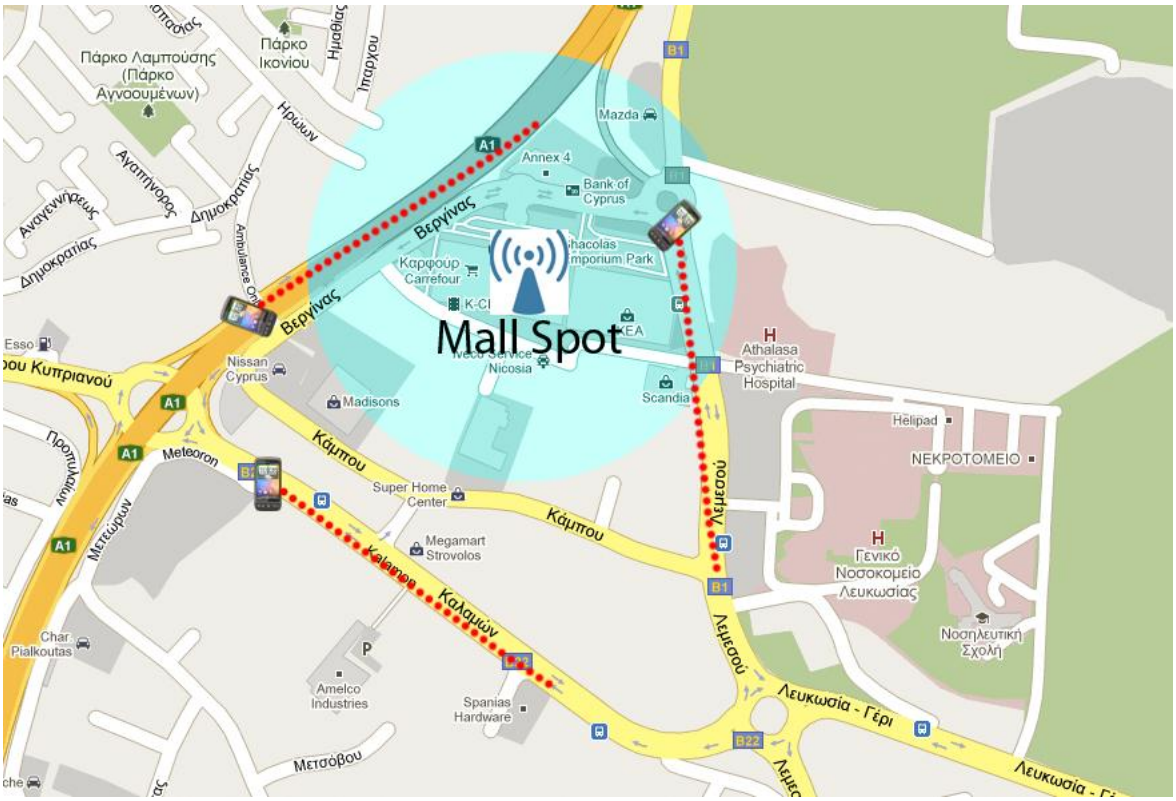
Συγκεκριμένα, η υπηρεσία που αποφασίσαμε να προσφέρουμε σε εφαρμογές στις κινητές συσκευές, είναι αυτή της «Εύρεσης Κοντινών Κινητών Συσκευών». Πιο ειδικά, αυτό που θα έχει ως αποτέλεσμα η χρήση της υπηρεσίας μας, θα είναι η παροχή μίας λίστας από τις Κ κοντινότερες κινητές συσκευές, σε σχέση με την δική μας τοποθεσία. Αυτή η πληροφορία, θα είναι το αποτέλεσμα του αλγορίθμου μας, ο οποίος θα επεξεργάζεται τα χωρικά-γεωγραφικά δεδομένα τα οποία θα λαμβάνει από τους χρήστες κινητών συσκευών, οι οποίοι με αυτό τον τρόπο θα κάνουν επερωτήσεις στην υπηρεσία μας. Δηλαδή, μία επερώτηση από κάποιο χρήστη κινητής συσκευής, θα περιέχει τα δικά του γεωγραφικά δεδομένα, κυρίως το γεωγραφικό μήκος και γεωγραφικό πλάτος, τα οποία μπορούν να συλλεχτούν με τρείς διαφορετικούς τρόπους. Αυτοί οι τρείς τρόποι είναι (α) Δια μέσου δορυφόρου (GPS), (β) Δια μέσου Ασύρματων δικτύων (Wi-Fi) και τέλος (γ) δια μέσου κυψέλων της κινητής τηλεφωνίας (Cellular Towers) [2]. Στη συνέχεια, θα γίνεται επεξεργασία όλων των επερωτήσεων που αποστέλλονται από τους χρήστες που είναι συνδεδεμένοι, για να απαντηθεί το

ερώτημα. Έπειτα, τα αποτελέσματα θα στέλνονται στην κάθε κινητή συσκευή, για να μπορεί να χρησιμοποιήσει την πληροφορία αυτή, με τον τρόπο τον οποίο θέλει.

1.2 Παραδείγματα χρήσης της γειτνίασης

Εάν θέλαμε να αναφέρουμε κάποια παραδείγματα με τα οποία θα μπορούσαμε να αναδείξουμε την σημαντικότητα της χρήσης της πληροφορίας ότι κάποιος άλλος βρίσκεται σε κοντινή απόσταση από εσένα, θα ήταν αμέτρητα [4]. Όμως, θα αναφέρουμε μερικά, για να δούμε ποιες είναι οι νέες δυνατότητες που μπορούμε να έχουμε σε μία κινητή συσκευή, εάν κάνουμε χρήση της υπηρεσίας της εύρεσης κοντινών κινητών συσκευών. Ακολουθούν μερικά παραδείγματα, από διαφορετικά περιβάλλοντα/τομείς υλοποίησης κάποιας εφαρμογής που κάνει χρήση της δικής μας υπηρεσίας, έτσι για να φανεί επίσης και η προσαρμοστικότητα της υπηρεσίας που προσφέρουμε, στα πλαίσια διαφόρων τομέων της ζωής και της κοινωνίας μας.

Πρώτο παράδειγμα, θα ήταν να χρησιμοποιηθεί η υπηρεσία μας, για λόγους διαφημιστικούς, στα πλαίσια λειτουργίας ενός εμπορικού κέντρου. Συγκεκριμένα, θα μπορούσε να υπάρχει ένα κεντρικό σημείο, όπου τα διάφορα καταστήματα του εμπορικού κέντρου θα μπορούσαν να καταχωρούν τις διαφημίσεις τους, οι οποίες θα διαδίδονται στους χρήστες οι οποίοι είναι σε κάποια ακτίνα από το εμπορικό κέντρο και επίσης έχουν στην κατοχή τους κάποια «έξυπνη» κινητή συσκευή, στην οποία έχουν εγκατεστημένη την εφαρμογή για τις διαφημίσεις. Με αυτό τον τρόπο, θα μπορούν να μεταδίδονται οι διαφημίσεις σε χρήστες που πραγματικά είναι κοντά στο εμπορικό κέντρο, επομένως οι πιθανότητες να επισκεφτούν το κατάστημα που διαφημίζει μια προσφορά, είναι αυξημένες. Άρα, με αυτό τον τρόπο, δεν έχει συμφέρον μόνο η εταιρία από την χρήση της υπηρεσίας, αφού θα μπορεί να πληρώσει μόνο ένα μικρό αντίτιμο, για να διαφημιστεί άμεσα σε πελάτες οι οποίοι είναι πιο πιθανό να επισκεφτούν την επιχείρησή τους, αλλά επίσης και για τον πελάτη, ο οποίος αφού βρίσκεται κοντά στην περιοχή που είναι οι εγκαταστάσεις της επιχείρησης, θα μπορεί να επισκεφτεί την επιχείρηση, χωρίς να χρειάζεται για παράδειγμα να οδηγήσει ξανά για να πάει στην συγκεκριμένη περιοχή, αλλά μπορεί να συνδυάσει αυτή του την έξοδο, που προφανώς είχε βγει για κάποιο άλλο λόγο, με την επίσκεψη του στην διαφημιζόμενη επιχείρηση.



Σχήμα 1.1 : Εφαρμογή Mall Spot

Το επόμενο παράδειγμα χρήσης της υπηρεσίας της εύρεσης κοντινότερων κινητών συσκευών, είναι το πλέον χαρακτηριστικό, αλλά συνάμα και αναμενόμενο. Στην απλή του περίπτωση, μπορεί να είναι απλό, αλλά εάν κάποιος βάλει φαντασία, μπορεί να προσθέσει πολλές λεπτομέρειες, οι οποίες θα το κάνουν πολύ χρήσιμο. Η εφαρμογή που αναφέρουμε, είναι η ειδοποίηση με το πάτημα ενός κουμπιού για έκτακτες ανάγκες, περιπτώσεις κινδύνου κ.α., όπου ένα άτομο μπορεί να βρεθεί σε κάποια δυσάρεστη κατάσταση και να θέλει να ειδοποιήσει με μία κίνηση, το πάτημα δηλαδή ενός κουμπιού, κάποιον για να προσέλθει να τον βοηθήσει. Όταν ο χρήστης της κινητής συσκευής, ο οποίος βρίσκεται σε κάποια κατάσταση, η οποία χρίζει βοήθειας, κάνει χρήση της εφαρμογής, τότε το αποτέλεσμα θα είναι να ειδοποιηθούν οι κοντινότεροι χρήστες «έξυπνων» κινητών συσκευών, για να προσέλθουν προς βοήθεια του. Η εφαρμογή από μόνη της, μπορεί να συλλέξει στοιχεία που αφορούν την συγκεκριμένη θέση που βρίσκεται ο χρήστης, διάφορες πληροφορίες που μπορεί να λάβει με τους αισθητήρες της συσκευής, όπως θερμοκρασία, φωτισμό, ή ακόμα και να υπολογίσει τις οδηγίες που πρέπει να ακολουθήσει κάποιος για να σπεύσει να βοηθήσει. Με αυτό τον τρόπο, όταν κάποιος από αυτούς που θα παραλάβουν το μήνυμα, θελήσει, εθελοντικά πάντα, να βοηθήσει τον συνάνθρωπο του, τότε θα μπορεί να ακολουθήσει τις οδηγίες

για να τον προσεγγίσει άμεσα και με ταχύτητα, χωρίς να χάνονται κρίσιμα δευτερόλεπτα, τα οποία πολλές φορές μπορούν να αποβούν και μοιραία για την υγεία, αλλά και την ζωή του ατόμου που χρειάστηκε βοήθεια.

Σχήμα 1.2 : Εφαρμογή Push S-O-S

Τέλος, ένα τρίτο παράδειγμα, το οποίο μπορεί να αναδείξει την χρήση της υπηρεσίας μας, είναι μία εφαρμογή για ανταλλαγή μηνυμάτων, μεταξύ μιας «έξυπνης» κινητής συσκευής και των υπολοίπων, οι οποίες βρίσκονται σε κοντινή απόσταση από αυτήν. Με αυτό τον τρόπο, θα μπορούσαμε να παρομοιάσουμε αυτή την εφαρμογή, σαν τοπικό ραδιόφωνο μέσω Διαδικτύου (κείμενο και όχι φωνή).

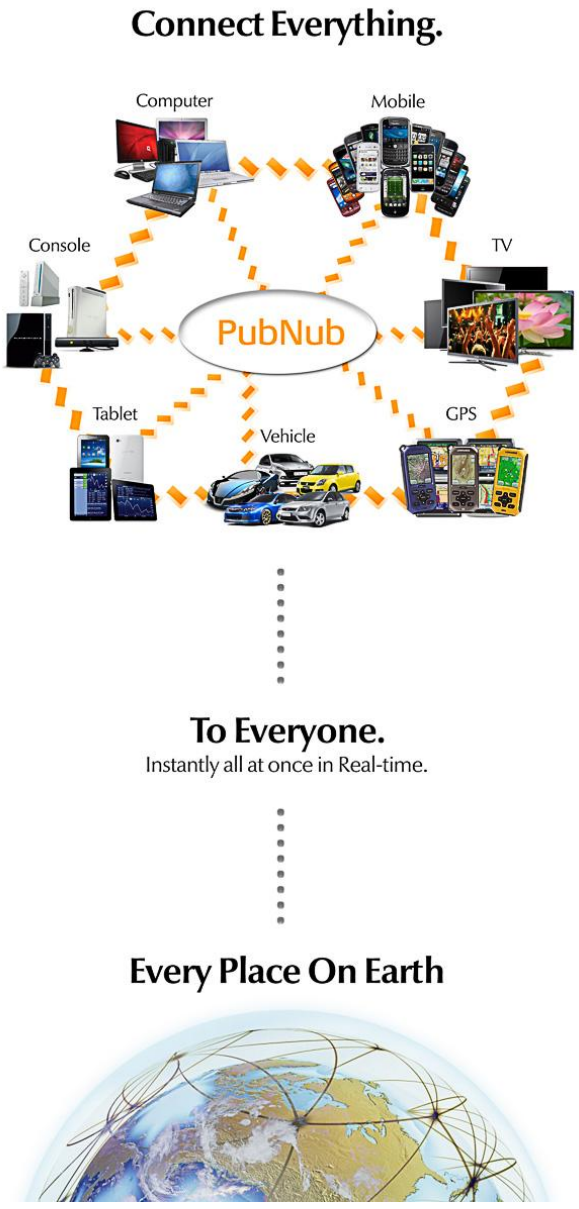
Σχήμα 1.3 : Εφαρμογή ανταλλαγής μηνυμάτων

1.3 Σχετική Δουλειά

Υπάρχουν πολλές καινοτόμες εφαρμογές που μπορούμε να βρούμε σε δίκτυο έξυπνων τηλεφώνων. Ένα παράδειγμα είναι η καιροσκοπική και η συμμετοχική αίσθηση [11], [12], [13], [15] οπού εφαρμογές μπορούν να ενεργούν ως κινητοί κόμβοι σε μια δεδομένη περιοχή για να παρέχουν πληροφορίες για την γειτνίαση χρησιμοποιώντας τις ικανότητες των αισθήσεων. Ακόμα ένα παράδειγμα είναι ο υπολογισμός της καθυστέρησης λόγο των κυκλοφοριακής συμφόρησης [15] χρησιμοποιώντας Wi-Fi σήματα που συλλέχτηκαν από έξυπνες κινητές συσκευές έναντι της υψηλής σε κόστος χρησιμοποίηση του GPS. Στην κοινωνική σελίδα Google Latitude ενεργοποιεί χρήστες για να εντοπίζει αυτούς και τα κοινωνικά δίχτυα που έχουν επισκεφτεί. Η Δεδομένη υπηρεσία ανέφερε πάνω από 3 εκατομμύρια εγγεγραμμένους χρήστες και πάνω από 1 εκατομμύριο ενεργούς χρήστες., πάρα την αμφιλεγόμενη ανησυχία για τα ιδιωτικά δεδομένα. Παρομοίως υπάρχουν πολλές εφαρμογές όπως το Foursquare, Gowalla και Loopt που έχουν την ανάλογη επιτυχία στον κόσμο των έξυπνων τηλεφώνων. Επίσης η ακαδημαϊκή έρευνα και πρόοδος σε αυτό τον τομέα βρίσκεται σε εξέλιξη.

Αυτό φυσικά, μας φέρνει στο γνωστό κοινωνικό δίκτυο, Twitter, το οποίο χαρακτηρίζεται ως SMS του Διαδικτύου. Το Twitter [1,22,23,24] είναι μία ιστοσελίδα κοινωνικής δικτύωσης, η οποία προσφέρει υπηρεσία ανάρτησης μικρού μεγέθους μηνυμάτων, συγκεκριμένα μέχρι 140 χαρακτήρων. Το Twitter προϋποθέτει να έχεις κάποιο λογαριασμό για να μπορείς να χρησιμοποιείς την υπηρεσία που προσφέρει. Επίσης, για να μπορείς να διαβάσεις αυτά τα μηνύματα μικρού μεγέθους που γράφονται από τους χρήστες, πρέπει να τους ακολουθάς. Επίσης, το ίδιο ισχύει εάν αυτοί θέλουν να διαβάσουν τα μηνύματα που εσύ γράφεις, πρέπει να σε ακολουθούν οι ίδιοι. Δηλαδή δεν ισχύει η αμφίδρομη σχέση και κατά κύριο λόγο οι χρήστες του Twitter δεν ακολουθούν αυτούς οι οποίοι είναι ακόλουθοι τους. Στο Twitter, επίσης, έχει προστεθεί εκ των υστέρων, η δυνατότητα επισύναψης πληροφορίας που αφορά την θέση του συγγραφέα του μηνύματος, την στιγμή που το κοινοποιεί στο χρονοδιάγραμμα (timeline) του. Με αυτό τον τρόπο, κάποιος θα μπορούσε να κάνει επερωτήσεις, ψάχνοντας μηνύματα, από την υπηρεσία αναζήτησης που προσφέρει το Twitter, περιορίζοντας τα αποτελέσματα στα μηνύματα που αναρτήθηκαν στην γύρω περιοχή από αυτή που βρίσκεται τη συγκεκριμένη στιγμή ο χρήστης.

Μία υπηρεσία η οποία είναι παρόμοια με το προαναφερθέν Twitter, είναι το PubNub [6], το οποίο χαρακτηρίζεται από τους κατασκευαστές του ως «το Twitter για τις μηχανές». Συγκεκριμένα, κάποιος με την χρήση της υπηρεσίας αυτής, μπορεί να ανταλλάζει πληροφορίες και διάφορα μηνύματα ανάμεσα στις διάφορες συσκευές, οι οποίες λειτουργούν σαν τους ακολούθους που υπάρχουν στην υπηρεσία του Twitter. Ένα αρκετά περιγραφικό σχήμα, είναι το σχήμα 1.4, το οποίο απεικονίζει την χρήση του PubNub.



Σχήμα 1.4 : Περιγραφή PubNub [6]

Κάτι πιο κοντά σε αυτό που κάνουμε, είναι οι δεκάδες, ίσως και εκατοντάδες εφαρμογές σε iOS και Android λειτουργικά συστήματα για «έξυπνες» κινητές

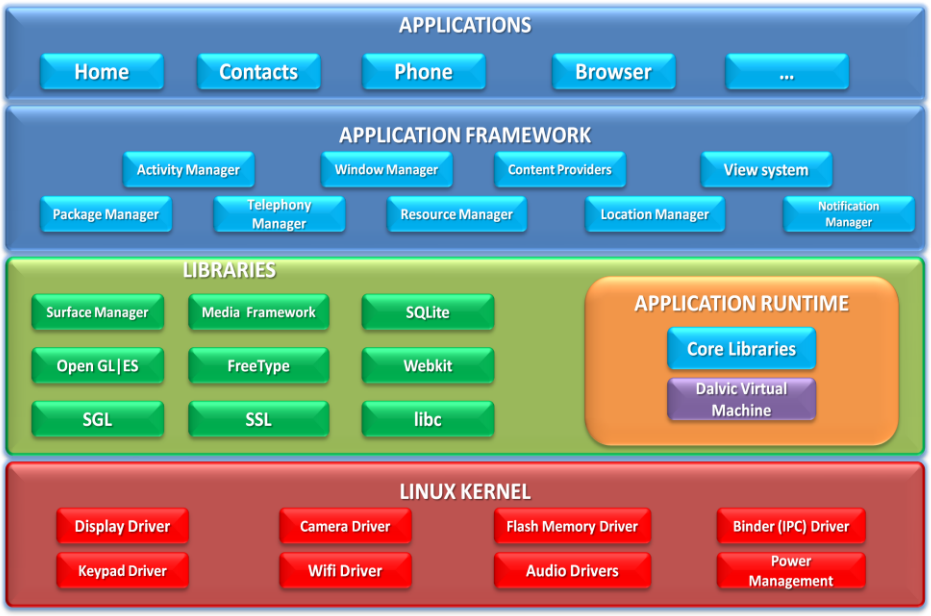
συσκευές, από την Apple και Google αντίστοιχα. Μερικά ονόματα τέτοιων εφαρμογών είναι το Blockboard, Yobongo, Foursquare, Gowalla, χρήση του Google Buzz στο κινητό, χρήση του Twitter για κινητό, χρήση του Google Latitude και άλλες πολλές εφαρμογές και υπηρεσίες. Στην πλειονότητα τους αυτές οι εφαρμογές, προσφέρουν κάποιου είδους εγγραφή στην υπηρεσία, έτσι δημιουργείται ένα προφίλ του ατόμου που θέλει να χρησιμοποιήσει τις εφαρμογές αυτές. Στη συνέχεια, σε μερικές από αυτές τις εφαρμογές, πρέπει ο χρήστης να προσθέσει στους φίλους του, τα άτομα με τα οποία θέλει να μοιράζεται πληροφορίες, τόσο για την γεωγραφική του τοποθεσία, όσο και για τα μηνύματα που θα αποστέλλονται. Κάποιες δεν χρειάζονται την διαδικασία της προσθήκης φίλων για την χρήση της υπηρεσίας, όμως αυτές συνήθως είναι περιορισμένες σε κάποιες μεμονωμένες γεωγραφικές περιοχές. Το πιο κοντινό στην υπηρεσία που θα φτιάξουμε εμείς, είναι το Proximity 360 [5], το οποίο προσφέρει την δυνατότητα διαφήμισης, την δυνατότητα ανακοινώσεων μέσω μικρού μεγέθους μηνύματα, αλλά επίσης και εξεύρεση εστιατορίων και επιχειρήσεων κοντινών στο σημείο που βρισκόμαστε.

Αυτό που θα κάνει την δική μας υπηρεσία για ανεύρεση των κοντινότερων χρηστών με κινητές συσκευές να ξεχωρίζει, είναι το γεγονός ότι οι χρήστες μας δεν θα χρειάζεται να εγγράφονται με οποιοδήποτε τρόπο στο σύστημα μας, η υπηρεσία θα προσφέρεται ανώνυμα, χωρίς να μπορεί να ταυτιστεί με οποιοδήποτε τρόπο η πληροφορία της γεωγραφικής θέσης μίας συσκευής, με οποιαδήποτε άλλα στοιχεία και η υπηρεσία μας θα δουλεύει σε όλο τον κόσμο, χωρίς να έχει περιορισμούς από γεωγραφικές περιοχές και συγκεκριμένες πόλεις ή χώρες στις οποίες θα δουλεύει. Το σημαντικότερο είναι ότι η υπηρεσία που θα φτιάξουμε θα μπορεί να χρησιμοποιείται και από άλλους προγραμματιστές, για να εφαρμόζουν την δική τους ιδέα, στο πώς μπορεί κάποιος να εκμεταλλευτεί την φυσική εγγύτητα κάποιων κινητών συσκευών και να αναπτύξει κάτι ενδιαφέρον, με φαντασία, το οποίο θα εξυπηρετεί αρκετό κόσμο.

1.4 Αρχιτεκτονική Android

Ο ορισμός Android είναι μια στοίβα λογισμικού για κινητές συσκευές που περιλαμβάνει ένα λειτουργικό σύστημα, ενδιάμεσο λογισμικό και τις βασικές

εφαρμογές. Το Android SDK παρέχει τα εργαλεία και τα APIs για να αρχίσουν να αναπτύσσονται εφαρμογές για την πλατφόρμα Android χρησιμοποιώντας τη γλώσσα προγραμματισμού Java.



Σχήμα 1.5 : Περιγραφή PubNub

Το λειτουργικό σύστημα Android αποτελείται από πολλά αναγκαία και ανεξάρτητα μέρη συμπεριλαμβανομένων των ακόλουθων [21]:

- Το σχέδιο ή υπόδειγμα αναφοράς υλικού που περιγράφει τις ικανότητες που απαιτούνται από μια κινητή συσκευή, ώστε να υποστηρίξει το λογισμικό στοίβα
- Ένας Linux πυρήνας του λειτουργικού συστήματος που παρέχει τη διασύνδεση χαμηλού επιπέδου με το υλικό, τη διαχείριση μνήμης, και ελέγχου διεργασιών, όλα βελτιστοποιημένα για φορητές συσκευές
- Βιβλιοθήκες ανοιχτού κώδικα για την ανάπτυξη εφαρμογών, συμπεριλαμβανομένων SQLite, WebKit, OpenGL, και Media Manager
- Ο χρόνος εκτέλεσης είναι ο χρόνος που χρησιμοποιείται για να εκτελέσει και να φιλοξενήσει τις Android εφαρμογές, συμπεριλαμβανομένης της Dalvik εικονικής μηχανής και τις βιβλιοθήκες των βασικών ικανοτήτων που παρέχουν στο Android συγκεκριμένη λειτουργικότητα. Ο χρόνος εκτέλεσης είναι σχεδιασμένος για να είναι μικρός και αποδοτικός για να χρησιμοποιείται από κινητές συσκευές.

- Ένα πλαίσιο εφαρμογών που εκθέτει πραγματικά σύστημα υπηρεσιών για το επίπεδο εφαρμογών, συμπεριλαμβανομένου των διαχειριστή παραθύρων, content providers, τηλεφωνία και peer-to-peer υπηρεσίες.
- Ένα πλαίσιο για διεπαφή του χρήστη που χρησιμοποιείται για να φιλοξενήσει και να εκκινήσετε εφαρμογές.
- Προεγκατεστημένες εφαρμογές που αποστέλλονται ως μέρος της στοίβας.
- Ένα πακέτο ανάπτυξης λογισμικού που χρησιμοποιείται για τη δημιουργία εφαρμογών, συμπεριλαμβανομένων των εργαλείων, plug-ins, και τεκμηρίωσης.

1.5 Περίγραμμα Εργασίας

Το υπόλοιπο της εργασίας οργανώνεται ως εξής: Στο δεύτερο κεφάλαιο το οποίο περιλαμβάνει το μοντέλο συστήματος και συμπεριλαμβάνει επίσης την περιγραφή του προβλήματος, αναφορά στους αλγορίθμους που χρησιμοποιήθηκαν και αναφορά σε σχετική δουλειά η οποία έγινε από άλλους ερευνητές. Στο τρίτο κεφάλαιο το οποίο περιλαμβάνει την αρχιτεκτονική του συστήματος, τόσο του εξυπηρετητή, όσο και του πελάτη για την υπηρεσία της Εύρεσης κοντινών κινητών συσκευών, καθώς επίσης και ένα παράδειγμα εφαρμογής σε έξυπνη κινητή συσκευή Android, με κάποια στιγμιότυπα οθόνης. Επίσης θα περιγραφούν μικρά βοηθητικά προγράμματα, τα οποία ήταν σχετικά με επιμέρους θέματα της διπλωματικής εργασίας, τα οποία και αναπτύχθηκαν. Στο τέταρτο κεφάλαιο το οποίο περιλαμβάνει την πειραματική μελέτη που έγινε στα πλαίσια αυτής της διπλωματικής εργασίας. Συγκεκριμένα έχουμε την περιγραφή των δεδομένων που χρησιμοποιήθηκαν, το σενάριο στο οποίο βασιστήκαμε και έπειτα τα πειραματικά αποτελέσματα. Τέλος το πέμπτο και τελευταίο κεφάλαιο, περιλαμβάνει τα συμπεράσματα που έχουμε από την χρήση της αρχιτεκτονικής μας για τη προσφορά της υπηρεσίας της γειτνίασης. Επίσης, εκεί θα παραθέσουμε κάποιες ιδέες οι οποίες θα μπορούσαν να αποτελέσουν μελλοντική δουλειά, η οποία να βασίζεται στη συγκεκριμένη αρχιτεκτονική, ή και ιδέες για την βελτιστοποίηση της και παραπέρα ανάπτυξη της. Στο Παράρτημα Α παρουσιάζουμε τον πηγαίο κώδικα που αφορά τον εξυπηρετητή του συστήματος μας, στο Β τον πηγαίο κώδικα της εφαρμογής για επίδειξη που είχαμε αναπτύξει, στο Γ τον πηγαίο κώδικα των βοηθητικών εφαρμογών που έχουμε αναπτύξει και τέλος τον συνοδευτικό ψηφιακό δίσκο.

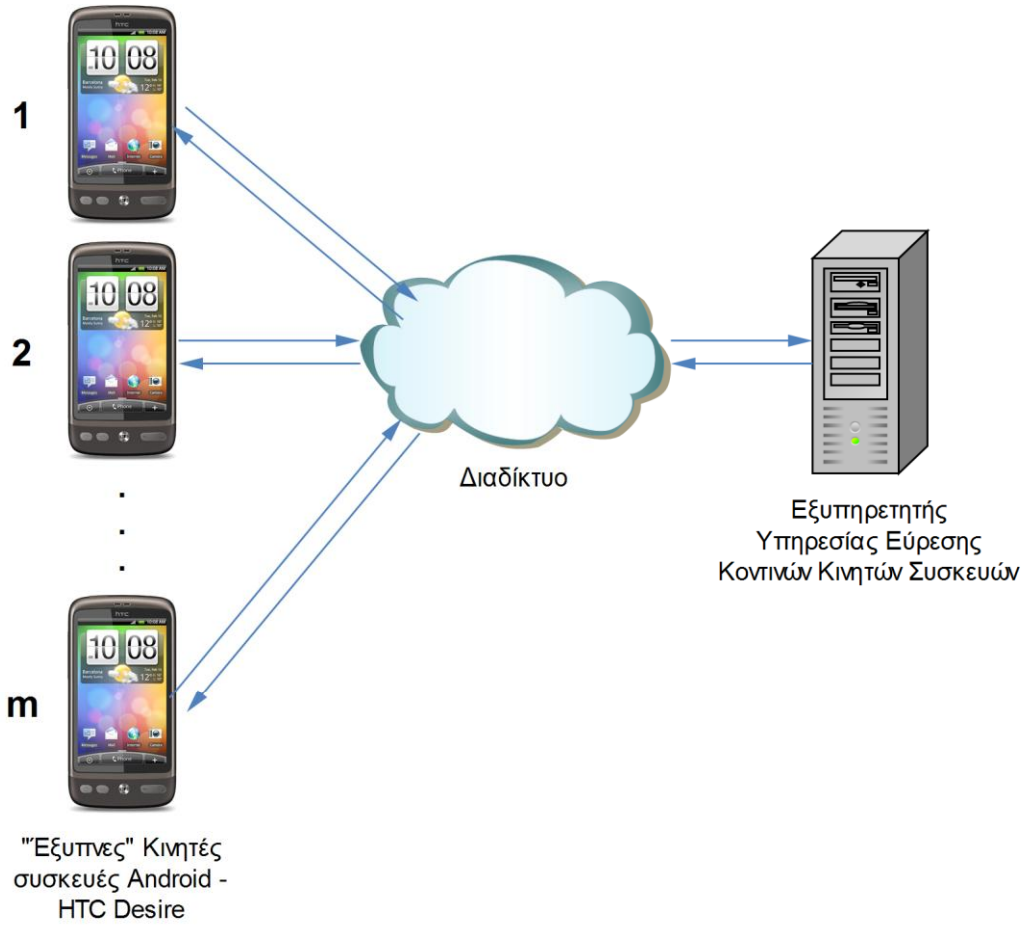
Κεφάλαιο 2

Υπόβαθρο και σχετική δουλειά

Σε αυτό το κεφάλαιο, θα καθορίσουμε την σημειογραφία την οποία θα ακολουθήσουμε σε όλη την εργασία, καθώς επίσης θα καθορίσουμε το μοντέλο συστήματος που έχουμε.

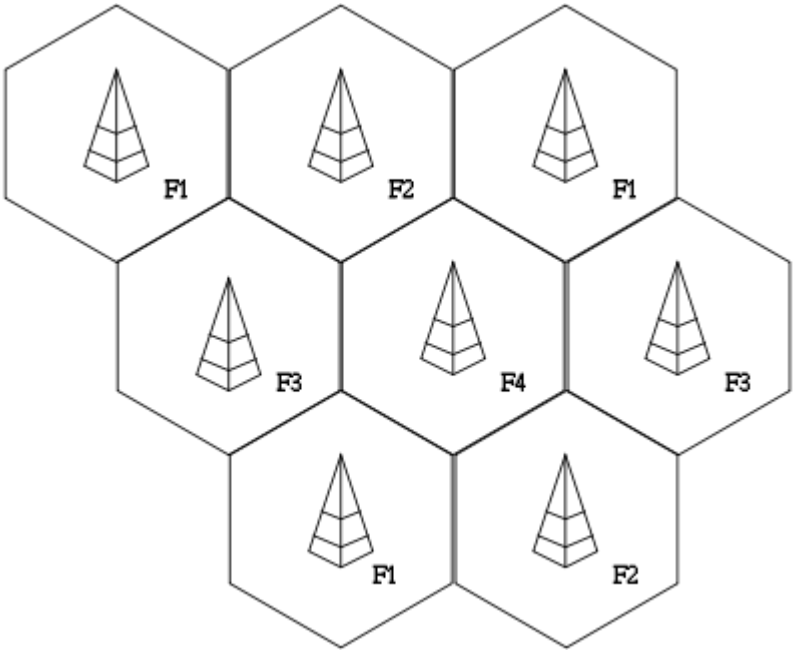
2.1 Μοντέλο Συστήματος

Έστω ότι έχουμε ένα σύνολο από χρήστες κινητών τηλεφώνων, {A1, A2, …, Am} , οι οποίοι κινούνται στο XY-επίπεδο. Κάθε δεδομένη στιγμή, το αντικείμενο Ai (i<=m), συλλέγει δεδομένα που το αφορούν. Αυτά τα δεδομένα, είναι πληροφορίες που συλλέγει από το περιβάλλον του, ή αφορούν το αντικείμενο συγκεκριμένα. Οι εγγραφές των αντικειμένων αυτών είναι του τύπου {X, Y, Acc, Cid, nCid1, …,nCidn }. Το X και Y αναφέρονται στην θέση του αντικειμένου στο επίπεδο. Συγκεκριμένα, αφορούν το γεωγραφικό μήκος και γεωγραφικό πλάτος. Το «Acc» αφορά την ακρίβεια που έχουμε την μέτρηση της γεωγραφικής θέσης του αντικειμένου. Αυτή την ακρίβεια έχει ως αποτέλεσμα ο τρόπος που χρησιμοποιεί η κινητή συσκευή για να βρει την γεωγραφική της θέση (GPS, WiFi ή Cellular tower triangulation). Όσο πιο ακριβής είναι ο τρόπος που έγινε η μέτρηση, τόσο πιο μικρή είναι τιμή της ακρίβειας. Στο σχήμα 2.1, φαίνεται το σχεδιάγραμμα του συστήματος μας, το οποίο αποτελείται βασικά από πολλούς πελάτες, τις «έξυπνες» κινητές συσκευές, οι οποίες στην περίπτωση μας είναι τα HTC Desire, και ένα εξυπηρετητή, ο οποίος ακούει για επερωτήσεις από τους πελάτες, μέσω Διαδικτύου.



Σχήμα 2.1 : Απλό σχεδιάγραμμα συστήματος

Το Cid που ακολουθεί, αφορά το Cellular tower ID, δηλαδή τον αριθμό καταχώρησης της κεραίας της κινητής τηλεφωνίας, στην οποία είναι εκχωρημένη η κινητή συσκευή. Τέλος, τα { nCid1, …,nCidn }, αφορούν τους αριθμούς καταχώρησης από τις γειτονικές κεραίες κινητής τηλεφωνίας, τις οποίες η κινητή συσκευή μπορεί να δει, αλλά δεν είναι καταχωρημένη σε αυτές. Στο σχήμα 2.2, φαίνεται το μοντέλο μίας περιοχής και η κάλυψη που προσφέρεται από της κυψέλες κινητής τηλεφωνίας στη συγκεκριμένη περιοχή. Η αναπαράσταση των κυψελών με εξάγωνο σχήμα, είναι η ιδανική, η οποία βοηθά στις μαθηματικές πράξεις και υπολογισμούς που χρειάζεται να γίνουν για την ανεύρεση της ραδιοκάλυψης ενός σταθμού βάσης κυψέλης. Στην πραγματικότητα, το σχήμα που μπορεί να έχει η ραδιοκάλυψη μιας κυψέλης, είναι οποιοδήποτε άμορφο σχήμα, με υπερκαλύψεις με τις ραδιοκαλύψεις των γειτονικών του κυψελών.



Σχήμα 2.2 : Παράδειγμα Cellular Towers

Μόλις συλλεχθούν αυτές οι πληροφορίες από το περιβάλλον της κινητής συσκευής, τότε γίνεται επικοινωνία με τον εξυπηρετητή ο οποίος θα δεχτεί τις πληροφορίες αυτές. Η κινητή συσκευή, αν και είναι «έξυπνη», όπως ονομάζονται οι τελευταίου τύπου κινητές συσκευές με αρκετά αυξημένες δυνατότητες, δεν έχει τις αρκετές δυνατότητες για να μπορέσει να επεξεργαστεί τις πληροφορίες αυτές και να τρέξει αλγορίθμους απαιτητικούς τόσο σε μνήμη, όσο και σε πόρους επεξεργαστή. Επομένως η αποστολή των πληροφοριών στον εξυπηρετητή, ήταν αναπόφευκτη. Ακόμη και σε αυτή την περίπτωση, η αποστολή των δεδομένων, αποτελεί και αυτή από μόνη της, αρκετά ακριβή πράξη, λόγω της αποστολής μέσω ασύρματης σύνδεσης, αφού είναι πιο επιρρεπής σε αναμεταδόσεις πακέτων, λόγω λαθών στο μέσο μετάδοσης.

2.2 Ποσοτικοποίηση του μοντέλου συστήματος

Σε αυτό το υποκεφάλαιο, θα παρουσιάσουμε συγκεκριμένα, αυτά που αφορούν το μοντέλο συστήματος. Συγκεκριμένα, θα αναφέρουμε σε μερικές πραγματικές μετρήσεις και ποσότητες που αφορούν το μοντέλο μας. Οι «έξυπνες» κινητές συσκευές που αναφέρονται, είναι ουσιαστικά το κινητό από την εταιρία HTC, με όνομα μοντέλου «Desire». Αυτή η κινητή συσκευή, έχει εγκατεστημένο το λειτουργικό Android, έκδοση 2.2 (κωδική ονομασία: Froyo), το οποίο αναπτύσσει η γνωστή εταιρία Google. Αυτή η

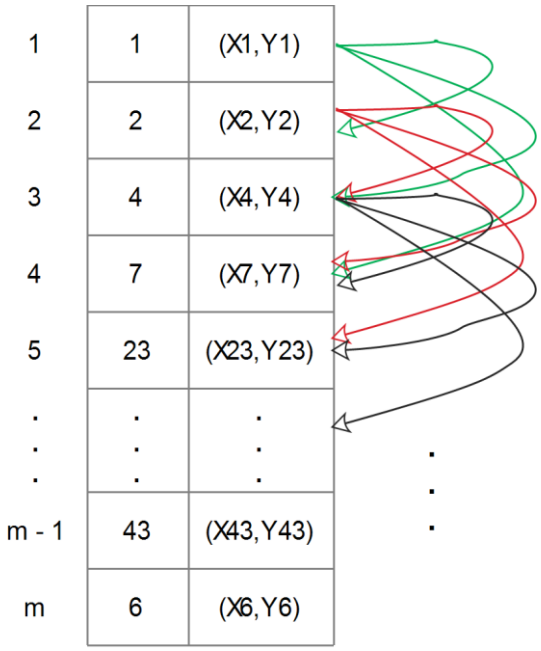
κινητή συσκευή, έχει δυνατότητα σύνδεσης με ασύρματα δίκτυα, τύπου 802.11b/g. Έχει επίσης επεξεργαστή Qualcomm QSD8250 στα 1000 MHz, 512MB μνήμη ROM και 576MB κύριας μνήμης (RAM). Αυτή την κινητή συσκευή χρησιμοποιήσαμε για την ανάπτυξη της εφαρμογής μας σε κινητή συσκευή, η οποία επικοινωνεί με τον εξυπηρετητή που τρέχει τον αλγόριθμο για την εύρεση των Κ κοντινών κινητών συσκευών. Είναι γραμμένη η εφαρμογή σε γλώσσα Java, η οποία είναι η επίσημη γλώσσα για ανάπτυξη εφαρμογών στην πλατφόρμα του Android. Αυτή η εφαρμογή χρησιμοποιήθηκε για σκοπούς επίδειξης στα πλαίσια των δραστηριοτήτων του τμήματος Πληροφορικής, αλλά επίσης και για να αποδείξει, με διαδραστικό τρόπο, την χρήση της πληροφορίας της εγγύτητας κινητών συσκευών μεταξύ τους. Λεπτομέρειες για την υλοποίηση και παράπλευρες υλοποιήσεις για την ανάκτηση συλλογών δεδομένων, τόσο από τον φυσικό κόσμο, το περιβάλλον τοπικά εδώ στην Κύπρο, όσο και από το Διαδίκτυο και συγκεκριμένα το Twitter, καθώς και διάφορα στιγμιότυπα οθόνης που πήραμε από την συσκευή κατά τη διάρκεια της επίδειξης, θα επεξηγηθούν και παρουσιαστούν αργότερα, σε επόμενα κεφάλαια.

2.3 Ανάλυση αλγορίθμων

2.3.1 Περιγραφή άπληστου (BNA) αλγορίθμου

Στην ενότητα αυτή θα επεξηγηθεί ο άπληστος αλγόριθμος που χρησιμοποιήθηκε σαν μέτρο σύγκρισης, αλλά και σαν επιλογή στα πειράματα που ακολουθούν σε επόμενο κεφάλαιο.

Ο αλγόριθμος αυτός, παίρνει σαν δεδομένο τις εγγραφές από τις επερωτήσεις των χρηστών. Αυτές οι εγγραφές, ανάμεσα σε άλλα, περιέχουν το γεωγραφικό μήκος και γεωγραφικό πλάτος της κινητής συσκευής. Αυτή την πληροφορία και μόνο χρειάζεται ο αλγόριθμος αυτός για να μπορεί να κάνει τους υπολογισμούς του. Θα μπορούσαμε να τον χαρακτηρίσουμε ως ολιγαρκή, αφού σε σχέση με τον άλλο αλγόριθμο, χρειάζεται μόνο τις εγγραφές με τους χρήστες και μόνο με αυτές μπορεί να υπολογίσει με άπληστο τρόπο τις αποστάσεις μεταξύ των και έπειτα να τους επιστρέψει τα αποτελέσματα που περιέχουν τους Κ κοντινότερους χρήστες προς αυτούς. Στο σχήμα 2.3 που ακολουθεί θα προσπαθήσουμε να δείξουμε την λογική αυτού του αλγορίθμου.



Σχήμα 2.3 : Συγκρίσεις BNA

Στο σχήμα φαίνεται η λογική του αλγορίθμου μας στις συγκρίσεις που κάνει μεταξύ των κόμβων, για να βρει την μεταξύ τους απόσταση. Συγκεκριμένα, ο αλγόριθμος είναι της τάξεως του $O(m^2)$, και ο κάθε κόμβος, ο οποίος αποτελεί μία καταγεγραμμένη επερώτηση από μία κινητή συσκευή, συγκρίνεται με τον κάθε επόμενο κόμβο, για να βρούμε την μεταξύ τους απόσταση. Στην συνέχεια, ο κόμβος που έχει συγκριθεί ήδη με τους υπόλοιπους, δεν συγκρίνεται ξανά μαζί τους. Τα αποτελέσματα από τις συγκρίσεις, μπαίνουν σε μία δομή, ξεχωριστή για τον κάθε κόμβο. Αυτή η δομή, είναι μία συνδεδεμένη λίστα της οποίας ο κάθε κόμβος αντιστοιχεί στην κάθε κινητή συσκευή. Σε κάθε κόμβο αυτής της λίστας, υπάρχει μία άλλη συνδεδεμένη λίστα, όπου εκεί κρατάμε την πληροφορία της σύγκρισης της διαφοράς απόστασης από τις υπόλοιπες κινητές συσκευές. Επίσης, κατά την δημιουργία αυτής της λίστας, γίνεται εισαγωγή μαζί με ταξινόμηση, επομένως, μόλις ο αλγόριθμος μας τελειώσει με τις συγκρίσεις των αποστάσεων των κινητών συσκευών, έχει έτοιμα τα αποτελέσματα για την κάθε κινητή συσκευή, ταξινομημένα, για να μπορεί με ευκολία να διαλέξει τους K κοντινότερους προς αυτόν και να του στείλει πίσω το αποτέλεσμα. Έτσι με αυτό τον τρόπο, δεν χρειάζεται να κάνουμε εκ νέου ταξινόμηση για να μπορέσουμε να βγάλουμε τους K κοντινότερους για να μπορούμε να απαντήσουμε την επερώτηση που μας έχει θέσει η κάθε κινητή συσκευή.

2.3.2 Περιγραφή PNA αλγορίθμου

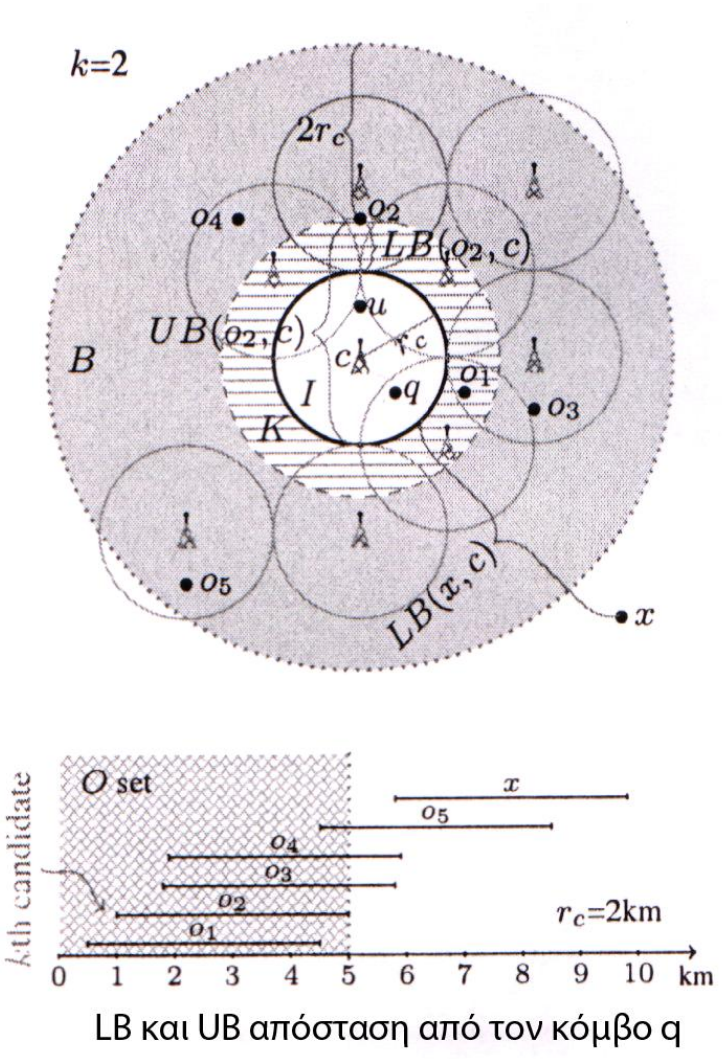
Στην ενότητα αυτή θα επεξηγηθεί ο αλγόριθμος [40] γειτνίασης που χρησιμοποιήθηκε σαν το κύριο συστατικό γύρω από την υλοποίηση του εξυπηρετητή μας, αλλά και σαν επιλογή στα πειράματα που ακολουθούν σε επόμενο κεφάλαιο.

Ο αλγόριθμος αυτός, είναι μπορούμε να πούμε πιο απαιτητικός, ως προς τα δεδομένα που χρειάζεται για να μπορεί να τρέξει και να παράγει αποδοτικά τα αποτελέσματα για τις επερωτήσεις των κινητών συσκευών. Συγκεκριμένα, χρειάζεται να έχει δύο εισόδους από συλλογές δεδομένων για να μπορεί να δουλέψει. Η μία συλλογή που χρειάζεται, είναι ένα σύνολο από τις κυψέλες της κινητής τηλεφωνίας, το οποίο θα περιλαμβάνει το Cell ID τους, καθώς επίσης και την θέση του κέντρου τους, δηλαδή το γεωγραφικό μήκος και γεωγραφικό πλάτος και τέλος την ακτίνα που έχει η ραδιοκάλυψη της κυψέλης. Όταν έχει αυτή την πληροφορία, για μια περιοχή, τότε μπορεί να αρχίσει να δέχεται επερωτήσεις από τους χρήστες των κινητών συσκευών.

Η άλλη είσοδος που χρειάζεται ο αλγόριθμος αυτός, είναι οι ροές δεδομένων από τους χρήστες των κινητών συσκευών, οι οποίοι αποστέλλουν τις πληροφορίες αυτές με μορφή επερωτήσεων. Δηλαδή, η επερώτηση περιλαμβάνει μέσα τα στοιχεία που χρειάζεται ο αλγόριθμος για να υπολογίσει τους K κοντινότερους χρήστες κινητών τηλεφώνων. Σε σύγκριση με τον προηγούμενο αλγόριθμο, αυτός ο αλγόριθμος θα ήταν πιο αποδοτικός εάν στα δεδομένα αυτά που αναφέρουμε, υπάρχει και η πληροφορία για το Cell ID στο οποίο είναι εγγεγραμμένος ο χρήστης τη δεδομένη στιγμή. Επίσης θα ήταν πολύ καλή η περίπτωση, εάν αποστέλλονται και τα δεδομένα που αφορούν τις γειτονικές κυψέλες και αυτό για την κυψέλη που είναι συνδεδεμένος ο χρήστης, τότε πολύ εύκολα θα μπορεί να τρέξει ο αλγόριθμος του υπολογισμού των κοντινότερων κινητών συσκευών, ο οποίος κατ' ακρίβεια κάνει μεγάλη χρήση της ύπαρξης των κυψελών, δηλαδή εκμεταλλεύεται την ύπαρξη των ραδιοκαλύψεων αυτών για να περιορίζει τον αριθμό των συγκρίσεων που πρέπει να κάνει για να έχουμε το τελικό αποτέλεσμα στις επερωτήσεις από τους χρήστες κινητών συσκευών. Στην υλοποίηση που έχει γίνει, για την αποτίμηση του αλγορίθμου, έχει χρησιμοποιηθεί η πληροφορία

της τοποθεσίας της κινητής συσκευής, σε συνδυασμό με την συλλογή από τις κυψέλες, και τις πληροφορίες θέσης του κέντρου και μήκους της ακτίνας της κάθε μίας, για να αποτιμηθεί κατά πόσο μια κινητή συσκευή ανήκει σε κάποια κυψέλη, είναι γειτονική ή δεν ανήκει καθόλου. Αφού η πληροφορία αυτή μας είναι διαθέσιμη λόγω της χρήσης των πραγματικών κινητών συσκευών, έτσι μπορούμε να γλιτώσουμε κάποιο υπολογισμό που θα έπρεπε να κάνουμε, σε σχέση με τον αρχικά υλοποιημένο αλγόριθμο.

Στη συνέχεια έχουμε ένα σχεδιάγραμμα για να δούμε τον τρόπο που δουλεύει ο συγκεκριμένος αλγόριθμος, σε υψηλό επίπεδο, καθώς επίσης και για να δούμε την σύγκριση και διαφορά μεταξύ των δυο αλγορίθμων.



Σχήμα 2.4 : Αλγόριθμος PNA [40]

Όπως βλέπουμε και στο σχεδιάγραμμα, έχουμε το σύνολο από τις κυψέλες της κινητής τηλεφωνίας, οι οποίες χαρακτηρίζονται από την ακτίνα r. Έχουμε επίσης τον χρήστη κινητής συσκευής που εκτελεί μια επερώτηση προς εμάς, τον q, στον οποίο πρέπει να απαντήσουμε με τους K κοντινότερους χρήστες προς αυτόν. Η επερώτηση του q περιλαμβάνει εκτός άλλων, την τοποθεσία του, δηλαδή το γεωγραφικό μήκος και γεωγραφικό πλάτος, την κυψέλη στην οποία είναι συνδεδεμένος και μία λίστα από κυψέλες τις οποίες μπορεί να δει, επομένως βρίσκεται και στην ραδιοκάλυψη τους. Επίσης, βλέπουμε τους χρήστες O_i, οι οποίοι αποτελούν το σύνολο O, που στην ουσία είναι οι υποψήφιοι κόμβοι που δεν καλύπτονται από την ραδιοκάλυψη της συγκεκριμένης κυψέλης, για του κοντινότερους K, για κάθε χρήστη που ανήκει στη συγκεκριμένη κυψέλη. Επομένως έχουμε την δομή O, η οποία είναι διαφορετική για κάθε κυψέλη. Έχουμε επίσης την δομή I, στην οποία μπαίνουν οι υποψήφιοι κόμβοι, οι οποίοι όμως εμπίπτουν στην περιοχή κάλυψης της κυψέλης που εξετάζουμε. Τέλος, έχουμε μία δομή ανά κυψέλη, που μας λέει ποίοι κόμβοι είναι συνδεδεμένοι μέσω της συγκεκριμένης κυψέλης.

Η χρήση αυτών των τριών δομών, μας επιτρέπει να περιορίσουμε τις συγκρίσεις που πρέπει να κάνουμε για να βρούμε τις γειτονικές κινητές συσκευές, για κάθε χρήστη μας. Πιο ειδικά, για κάθε αίτηση από χρήστη που λαμβάνουμε, τον εισάγουμε στην λίστα με τους συνδεδεμένους χρήστες, της κυψέλης με την οποία είναι συνδεδεμένος. Στη συνέχεια, εισάγουμε στην κάθε δομή I, τους κόμβους τους οποίους καλύπτει η ραδιοκάλυψη, για την κάθε κυψέλη, παίρνοντας αυτή την πληροφορία από την λίστα με τις γειτονικές κυψέλες που μας αποστέλλει ο κάθε χρήστης. Η σημαντικότερη δομή από τις τρείς, είναι η επόμενη, η δομή O. Η δομή O είναι το κλειδί για την βελτιστοποίηση που υπόσχεται ο αλγόριθμος PNA. Ο τρόπος που εισάγονται στην δομή αυτή οι εγγραφές από τους κόμβους, είναι αυτό που κάνει το μεγαλύτερο μέρος της δουλειάς για τον αλγόριθμο και είναι το πιο έξυπνο κομμάτι του.

Ο τρόπος εισαγωγής των κόμβων στην δομή είναι να γίνει πρώτα ο υπολογισμός του UB και του LB, από την κυψέλη, προς το κέντρο του τρέχον κόμβου.

$$UB(c,u) = d(c_{center},u) + r_c$$

Το UB υπολογίζεται βρίσκοντας την απόσταση που έχουν τα δύο σημεία, το κέντρο της κυψέλης μας και το σημείο που έχει αναφέρει σαν γεωγραφική θέση ο χρήστης. Έπειτα, προστίθεται σε αυτό το μήκος της ακτίνας.

$$LB(c,u) = d(c_{center},u) - r_c$$

Στην αντίθετη περίπτωση, το LB υπολογίζεται πάλι με τον ίδιο τρόπο, όμως εκτός από την διαφορά ότι από την απόσταση μεταξύ των δύο σημείων, αφαιρούμε την ακτίνα. Έπειτα, γεμίζουμε την δομή O με κόμβους, μέχρι να φτάσουμε τον K^{o} κόμβο. Όταν φτάσουμε σε αυτό το σημείο, έχοντας ταξινομημένους τους κόμβους με βάση το LB της απόστασης τους από το κέντρο της κυψέλης c, επισημάνουμε τον κόμβο αυτό και από τούδε και στο εξής κατά την εισαγωγή των επόμενων κόμβων, κάνουμε τον εξής έλεγχο:

$$LB(u,c) \leq UB(u_k,c)$$

Αυτό σημαίνει ότι μόνο εάν το LB του κόμβου που εξετάζουμε είναι μικρότερο ή ίσο με το UB του $K^{ού}$ κόμβου θα μπει στην δομή O, ο κόμβος ο οποίος εξετάζεται την συγκεκριμένη στιγμή. Στο σχεδιάγραμμα 2.10, αυτό φαίνεται από τον πίνακα που βρίσκεται στο κάτω μέρος του σχήματος. Ο $K^{ος}$ κόμβος στην περίπτωση μας είναι ο κόμβος O_2, έτσι οι υπόλοιποι κόμβοι που το LB τους εμπίπτει στο διάστημα που είναι ίσο ή μικρότερο από το UB του O_2, μπορούν να μπούνε στη δομή O.

Επίσης προκύπτει από τα εξής ότι:

$$\left.\begin{aligned} UB_k &= d(c_{center},u_k) + r_c \\ LB_k &= d(c_{center},u_k) - r_c \end{aligned}\right\}$$

$$\Longrightarrow \quad UB_k = LB_k + 2 * r_c$$

Επομένως, όποιος κόμβος βρίσκεται σε απόσταση μεγαλύτερη του UB_k, όπως φαίνεται και στο σχεδιάγραμμα, δεν μπαίνει στην δομή Ο.

Στη συνέχεια έχουμε την φάση της επιλογής των κοντινότερων Κ κόμβων, για κάθε χρήστη, την φάση δηλαδή που γίνονται οι συγκρίσεις μεταξύ των γεωγραφικών σημείων των κόμβων, για να βρεθεί πρώτα η απόσταση με τον καθένα ξεχωριστά και έπειτα να γίνει η σωστή ταξινόμηση για να μπορούν να αποσταλούν τα αποτελέσματα σαν απάντηση του επερωτήματος που είχε θέσει στον εξυπηρετητή η κινητή συσκευή.

Ο τρόπος που γίνεται η επιλογή των κόμβων που θα συγκριθούν μεταξύ τους μέσα από τις δομές που έχουμε περιγράψει, είναι να διαβάζονται γραμμικά όλες οι εγγραφές που αφορούν τις κυψέλες. Στη συνέχεια, μέσα από κάθε εγγραφή που αφορά μία κυψέλη c, διαπερνάμε τους κόμβους που απαρτίζουν την δομή με τους χρήστες οι οποίοι είναι συνδεδεμένοι με την συγκεκριμένη κυψέλη. Έπειτα συγκρίνουμε αυτό τον κόμβο με τους κόμβους οι οποίοι βρίσκονται στην δομή Ι η οποία αφορά τους κόμβους που καλύπτονται από την ραδιοκάλυψη της κυψέλης c. Στο επόμενο βήμα, οι έλεγχοι που πρέπει να γίνουν είναι μεταξύ του κόμβου που εξετάζουμε και των κόμβων που βρίσκονται στην δομή Ο της συγκεκριμένης κυψέλης. Όταν φτάσουμε στο σημείο που ελέγξαμε μέχρι και τον τελευταίο κόμβο από τη λίστα Ο, τότε και μόνο τότε είμαστε σίγουροι ότι έχουμε διαλέξει τους σωστούς κοντινότερους Κ για τον κόμβο τον οποίο εξετάζαμε. Εδώ αξίζει να αναφέρουμε ότι και πάλι τα αποτελέσματα που παίρνουμε από τις συγκρίσεις που γίνονται μεταξύ των κόμβων, φυλάγονται σε μία δομή που όπως και στον άπληστο αλγόριθμο, όπου χρησιμοποιείται και ταξινόμηση κατά την εισαγωγή των κόμβων αυτών στην λίστα με τους κοντινότερους Κ του κάθε κόμβου.

2.3.3 Τρόπος εύρεσης της απόστασης μεταξύ των γεωγραφικών σημείων των κόμβων

Ο τρόπος που χρησιμοποιείται για να βρούμε τις αποστάσεις, τόσο μεταξύ των γεωγραφικών σημείων των κόμβων, όσο και τις αποστάσεις που απέχουν οι κόμβοι που εξετάζουμε από τα άκρα του κύκλου που απεικονίζει την κυψέλη μας, δεν είναι η απλή απόσταση που έχουν δύο σημεία σε δισδιάστατο χώρο, αλλά είναι η συνάρτηση που

μπορεί να λάβει υπόψη της, βάσει των γεωγραφικών σημείων των κόμβων που συγκρίνουμε, την καμπύλη που σχηματίζεται λόγω του ελλειπτικού σχήματος που έχει η Γή. Η συνάρτηση αυτή είναι η εξής:

$$haversin\left(\frac{d}{R}\right) = haversin(\varphi_2 - \varphi_1) + \cos(\varphi_1)\cos(\varphi_2)\, haversin(\Delta\lambda)$$

2.4 Πίνακας Συμβολισμών

Ακολουθεί πίνακας με τους συμβολισμούς τους οποίους θα χρησιμοποιήσουμε στην συνέχεια της διπλωματικής εργασίας για να αναφερόμαστε σε συγκεκριμένα αντικείμενα και ποσότητες.

Συμβολισμός	Περιγραφή
M ή m	Αριθμός κινητών συσκευών οι οποίες είναι συνδεδεμένες με τον εξυπηρετητή μας σε συγκεκριμένο στιγμιότυπο.
u	Κινητή συσκευή
N ή n	Ο αριθμός των γειτονικών κυψελών κινητής τηλεφωνίας που βλέπει η κάθε κινητή συσκευή κατά την συλλογή των στοιχείων για αποστολή προς τον εξυπηρετητή.
K	Ο αριθμός των κορυφαίων αποτελεσμάτων, τα οποία θα επιστραφούν στον κάθε χρήστη σαν αποτέλεσμα του αλγορίθμου εγγύτητας. (Αριθμός κοντινότερων κινητών συσκευών προς τον κάθε χρήστη)
Q	Η επερώτηση η οποία θέτει η κάθε κινητή συσκευή προς τον εξυπηρετητή, η οποία πρέπει στο τέλος να απαντηθεί από αυτόν.
C	Αριθμός των κυψελών κινητής τηλεφωνίας που θα χρησιμοποιούμε σε κάθε σενάριο, οι οποίες καλύπτουν για μία συγκεκριμένη γεωγραφική περιοχή.
c	Κυψέλη (Cell)

r	Ακτίνα της κυψέλης
BNA	Brute-force Neighbors Algorithm, είναι ο άπληστος αλγόριθμος ο οποίος χρησιμοποιήσαμε.
PNA	Proximity Neighbors Algorithm, είναι ο αλγόριθμος εγγύτητας που χρησιμοποιήσαμε.
D_{ij}	Είναι η απόσταση σε μέτρα της κινητής συσκευής i από την κινητή συσκευή j.
UB	Upper Bound για κόμβους στο σύνολο O του PNA
LB	Lower Bound για κόμβους στο σύνολο O του PNA
Κινητή Συσκευή	Έτσι θα ονομάζουμε τις «έξυπνες» κινητές συσκευές που χρησιμοποιούμε. Μπορούν επίσης να εμφανίζονται και σαν χρήστες, πελάτες, κόμβοι, συσκευές Android άλλα σημαίνουν το ίδιο και θα χρησιμοποιούνται ποικιλοτρόπως.

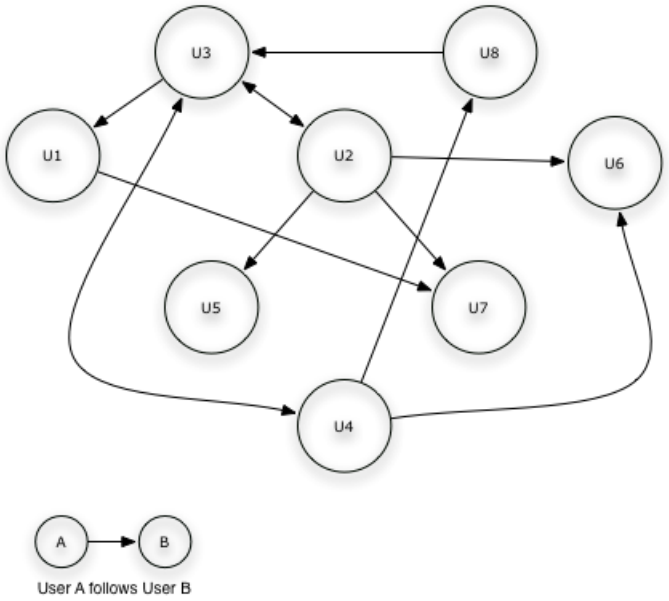
Πίνακας 2.1 : Πίνακας Συμβολισμών και εννοιών

Στα εναπομείναντα κεφάλαια, θα χρησιμοποιούμε τους πιο πάνω συμβολισμούς, καθώς επίσης και διάφορες έννοιες που επεξηγήθηκαν προηγουμένως στο κεφάλαιο αυτό, χωρίς κάποια επιπλέον αναφορά. Ο αναγνώστης προτρέπεται να επιστρέφει στο κεφάλαιο και στον πίνακα αυτό για σκοπούς υπενθύμισης της σημασίας του συμβολισμού, όταν κάτι δεν μπορεί να το καταλάβει.

2.5 Σχετική Δουλειά

Όπως έχουμε αναφέρει, η διπλωματική αυτή εργασία, ο τρόπος που σχετίζεται με την υπηρεσία κοινωνικής δικτύωσης Twitter είναι με την χρήση της υπηρεσίας για ανταλλαγή μηνυμάτων, βασιζόμενοι στον παράγοντα της γεωγραφικής θέσης των συσκευών που συμμετέχουν. Επίσης, έχουμε αναπτύξει μία εφαρμογή με γλώσσα προγραμματισμού Java, η οποία λειτουργεί σαν Twitter Crawler, δηλαδή προσπαθεί να απομυζήσει πληροφορίες από τον κοινωνικό γράφο του Twitter, με σκοπό να τις κατεβάσει τοπικά για να τις φυλάξει. Αυτό το πρόγραμμα, καθώς και άλλες μικρές εφαρμογές που έχουμε αναπτύξει στα πλαίσια της διπλωματικής, θα αναλυθούν στο επόμενο κεφάλαιο.

Σχετική δουλειά, η οποία αφορά το Twitter, έγινε από διάφορους ερευνητές [22,23,24,25,26,27,28], με σκοπό να καταλάβει η επιστημονική κοινότητα, πρώτο περί τίνος πρόκειται, τον σκοπό και την χρήση του Twitter, αλλά και την επίδραση που είχε στους ανθρώπους. Επίσης αυτό που έχουν κάνει οι ερευνητές, είναι να συλλέξουν πολλά δεδομένα, από την κινητικότητα και τις ενέργειες γενικά των χρηστών, στο κοινωνικό δίκτυο του Twitter. Ενδεικτικά, κάποια δεδομένα που χρησιμοποιούν, είναι ποιος χρήστης ακολουθεί ποιόν, το οποίο μας αφήνει στο τέλος με ένα γράφο ο οποίος απεικονίζει όλου ς τους χρήστες του κοινωνικού δικτύου σαν κόμβους και οι ακμές να συμβολίζουν τις μονόπλευρες σχέσεις μεταξύ τους. Υπενθυμίζουμε ότι στο κοινωνικό δίκτυο Twitter, εάν κάποιος είναι ακόλουθος σου, δεν σημαίνει ότι και εσύ πρέπει να είσαι δικός του ακόλουθος. Επομένως, ο γράφος που έχουμε δεν είναι αμφίδρομος.



Σχήμα 2.5 : Γράφος ακολούθων Twitter

Όπως φαίνεται στο πιο πάνω σχήμα, αυτός είναι ένας γράφος που μπορεί να δημιουργηθεί παίρνοντας πληροφορίες από τους χρήστες του Twitter.

Οι ενδείξεις δείχνουν ότι αυτή η ιδέα της χρήσης μικρού μεγέθους μηνύματα για την κοινοποίηση των σκέψεων και γενικά ότι θέλουν οι χρήστες του Twitter, είναι κάτι που οι χρήστες λάτρεψαν και το χρησιμοποιούν. Επίσης αναφέρεται [22] ότι το Twitter έφτασε σε σημείο να θεωρείται ένα μέσο πληροφόρησης, και όχι μόνο ένα κοινωνικό δίκτυο, λόγω του όγκου των πληροφοριών που αναρτώνται από τους χρήστ4ες του

καθημερινώς και αφορούν οποιοδήποτε θέμα μπορεί να φανταστεί ο ανθρώπινος νους, περιορισμένο σε 140 χαρακτήρες κειμένου. Επίσης, πολλοί ερευνητές, χρησιμοποιώντας το API για προγραμματισμό που προσφέρει το Twitter, μαζεύουν ολόκληρες συλλογές από δεδομένα που αφορούν τους χρήστες, με σκοπό να τα χρησιμοποιήσουν στα δικά τους πειράματα [25,26,27,28], για να είναι πιο πειστικά, λόγω του ότι είναι πραγματικά δεδομένα [29,31,33]. Αυτό το API χρησιμοποιήσαμε και εμείς για την υλοποίηση του Twitter crawler που θα αναλύσουμε σε μετέπειτα κεφάλαιο.

Άλλο παράδειγμα σχετικής δουλειάς που υπάρχει, είναι το SurroundSence [34], το οποίο προσπαθεί να κάνει χρήση της γεωγραφικής θέσης του χρήστη της κινητής συσκευής, με σκοπό να είναι σε θέση η υπηρεσία να αντιλαμβάνεται την παρουσία του χρήστη σε συγκεκριμένες περιοχές ενδιαφέροντος. Συγκεκριμένα, αναφέρει την χρήση της υπηρεσίας αυτής για λόγους διαφήμισης. Για παράδειγμα, αν ο χρήστης εισέλθει σε μία καφετέρια, να μπορεί να του σταλεί ένα κουπόνι που να του λέει για κάποια προσφορά ή να του δίνει κάποια επιπλέον έκπτωση. Κάτι δηλαδή σαν διαφήμιση με επίγνωση θέσης του χρήστη.

Μία άλλη σημαντική δουλειά που έγινε, η εφαρμογή CenceMe [35], η οποία αφορά την ανάπτυξη εφαρμογής για κινητά τηλέφωνα Nokia N95, για προσφορά υπηρεσιών οι οποίες επηρεάζονται από το περιβάλλον. Συγκεκριμένα, αναφέρεται σε υπηρεσίες οι οποίες καταγράφουν δεδομένα χρήσης της συσκευής, δεδομένα σχετικά με τον περιβάλλον χώρο, με σκοπό να τα χρησιμοποιήσουν για να προσαρμόσουν διάφορες ρυθμίσεις της συσκευής. Ακόμα, άλλη χρήση της υπηρεσίας CenceMe, ήταν η ενημέρωση των υπολοίπων συμμετασχόντων στο πείραμα, παραδείγματος χάρη με μία ανάρτηση στο κοινωνικό δίκτυο Facebook, καθορίζοντας όμως και το επίπεδο ιδιωτικότητας τους, σε αυτό που θέλουν εκείνοι, χωρίς να τους επιβάλλεται κάποιο. Παραπέρα, το CenceMe, θα μπορούσε να λειτουργήσει και για κάποιο σαν μέσο με το οποίο θα μάθει τον εαυτό το, δηλαδή θα δει με ένα τρίτο μάτι πως συμπεριφέρεται, τις συνήθειες του και τον τρόπο που κινείται γενικά στην ζωή του και στις συναναστροφές του με τους συνανθρώπους του. Σχετική δουλειά με το CenceMe, παρουσιάζει επίσης το Micro-Blog [36].

Μία δουλειά η οποία μοιάζει πολύ με την δική μας εφαρμογή του αλγορίθμου για εύρεση κοντινών συσκευών, για την παραπέρα χρήση τους με οποιοδήποτε τρόπο επιθυμεί ο προγραμματιστής της εφαρμογής, είναι το E-SmalTalker [37]. Το E-SmalTalker, αφουγκραζόμενο το περιβάλλον που περιβάλλει την κινητή συσκευή, προσπαθεί να περάσει μέσω του Bluetooth των συσκευών πληροφορίες που αφορούν το άτομο το οποίο είναι ο κάτοχος του κινητού, με τέτοιο τρόπο ώστε να μπορεί ο δεύτερος χρήστης κινητής συσκευής να μάθει πληροφορίες που θα τον βοηθήσουν να μην νιώθει αμηχανία και να είναι σε πλήρη άγνοια, εάν επιθυμήσει να μιλήσει πρόσωπο με πρόσωπο με τον πρώτο χρήστη. Δηλαδή μπορούμε να την σκεπτόμαστε σαν μια εφαρμογή η οποία αναλαμβάνει να κάνει τις συστάσεις για εμάς, αντί να περιμένουμε πότε θα κάνει την κίνηση ο άλλος.

Το είδος σχετικής δουλειάς αυτό, αφορά την ασφάλεια [38,39] που προσφέρεται μέσω των δικτύων των κινητών τηλεφώνων. Κάποιες δουλειές αναφέρονται στην δυνατότητα καταπολέμησης των απειλών, με διάφορους τρόπους, όπου ακόμα αναφέρουν ότι πρέπει το πρόγραμμα για καταστολή του κακόβουλου λογισμικού, να αντιγράφεται και να εξαπλώνεται με μεγαλύτερο ρυθμό από εκείνο που εξαπλώνεται το κακόβουλο λογισμικό, σε σκοπό να καταφέρει να το σταματήσει σε κάποια στιγμή.

Κεφάλαιο 3

Αρχιτεκτονική Συστήματος

3.1 Υπόβαθρο

3.1.1 Γλώσσα Προγραμματισμού

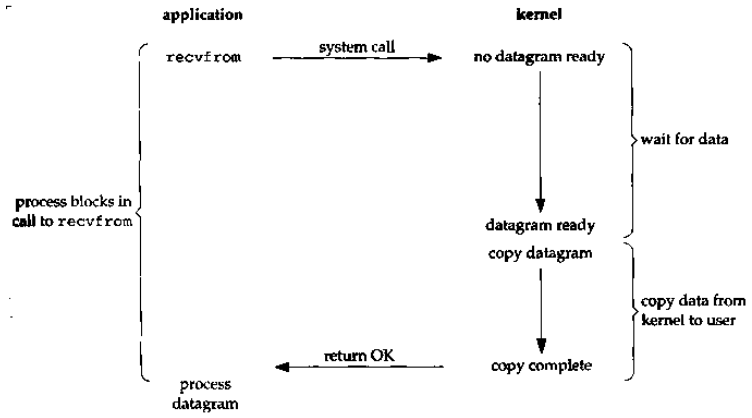
Ο εξυπηρετητής που υλοποιήθηκε, λόγω του ότι θα δέχεται χιλιάδες αιτήσεις, και πρέπει να είναι έτοιμος να αντεπεξέλθει σε τυχών μαζεμένες αιτήσεις για εξυπηρέτηση, έπρεπε να γίνει όσο πιο αποδοτικός μπορούσε να γίνει. Συγκεκριμένα, για τον λόγο αυτό, αποφασίστηκε η ανάπτυξη του σε γλώσσα C, η οποία είναι πιο χαμηλού επιπέδου από άλλες γλώσσες, έτσι μπορεί να εκτελέσει κάποιες ενέργειες με λιγότερες επιβαρύνσεις, απ' ότι θα είχαμε με μία γλώσσα προγραμματισμού πιο υψηλού επιπέδου, όπως για παράδειγμα η Java. Με αυτό τον τρόπο, ο εξυπηρετητής μας μπορεί να αντέξει σε μεγαλύτερο φόρτο εργασίας από αιτήσεις που θα έχει να εξυπηρετήσει, αλλά επίσης, λόγω της χρήσης της γλώσσας προγραμματισμού C, έχουμε και το

πλεονέκτημα της άμεσης δέσμευσης και αποδέσμευσης μνήμης, σε αντίθεση με άλλες γλώσσες που υλοποιούν την λογική του συλλέκτη σκουπιδιών (garbage collector), ο οποίος δεν σε αφήνει να διαχειριστείς άμεσα την αποδέσμευση της μνήμης, αλλά εν αντιθέσει, διαχειρίζεται μόνος του την αποδέσμευση αυτή, έχοντας ως αποτέλεσμα να μην ξέρουμε πότε στην πραγματικότητα θα απελευθερωθεί η μνήμη που τώρα μας είναι αχρείαστη.

3.1.2 Τεχνικές Εισόδου/Εξόδου

Επίσης έγινε χρήση σύγχρονης εισόδου/εξόδου (Synchronous I/O) [8], το οποίο είναι καλύτερο από την απλή περίπτωση με την διαδικασία της αποδοχής αιτήσεων από τον εξυπηρετητή. Η ιδέα πίσω από αυτή την υλοποίηση με την σύγχρονη είσοδο/έξοδο, δίνει στον εξυπηρετητή την ικανότητα να μην κολλάει κατά τη διάρκεια των αποδοχών των αιτήσεων από τους πελάτες. Ο εξυπηρετητής κατά την διάρκεια που περιμένει να έρθει νέα αίτηση, μπορεί να μπει σε κατάσταση αναμονής, δηλαδή να μπορεί να μην χρησιμοποιεί πόρους του επεξεργαστή, αλλά όταν έρθει αίτηση, να μπορεί να το καταλαβαίνει και έτσι να μπορεί να μπαίνει σε ενεργή κατάσταση και να εξυπηρετεί την εισερχόμενη αίτηση του πελάτη, χωρίς καθυστερήσεις. Ο τρόπος που γίνεται αυτό, είναι με την κλήση της συνάρτησης συστήματος (system call), select(). Η συνάρτηση αυτή, μαζί και με τις υπόλοιπες συναρτήσεις που προσφέρονται για υλοποίηση εξυπηρετητή με σύγχρονη είσοδο/έξοδο, θα αναλυθούν στο επόμενο υποκεφάλαιο. Οι πέντε τρόποι που θα συγκριθούν, είναι οι εξής: (α) Blocking I/O, (β) Non-blocking I/O, (γ) I/O Multiplexing (select and poll), (δ) Signal driven I/O (SIGIO), (ε) Asynchronous I/O (the POSIX aio_ functions).

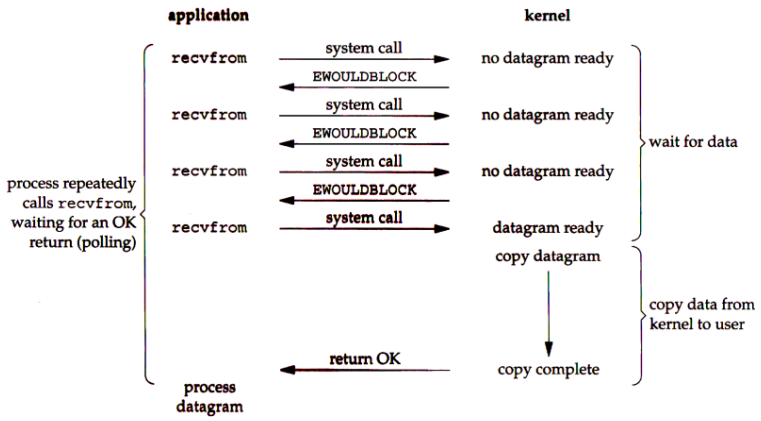
3.1.3 Σύγκριση τρόπων Εισόδου/Εξόδου



Σχήμα 3.1 : Blocking I/O

3.1.3.1 Blocking I/O

Η πρώτη μας επιλογή θα ήταν η απλή κλήση της συνάρτησης συστήματος accept(), η οποία θα μπορούσε να δέχεται τις αιτήσεις των πελατών, όταν αυτές έρχονται στον εξυπηρετητή. Το πρόβλημα με αυτή την προσέγγιση, θα ήταν ότι η κλήση στην συνάρτηση accept() θα κολλούσε την εκτέλεση του προγράμματος μας, μέχρι να παραληφθεί το πρώτο πακέτο δεδομένων, το οποίο προήρθε από κάποιο πελάτη που έκανε αίτηση σύνδεσης με τον εξυπηρετητή μας.

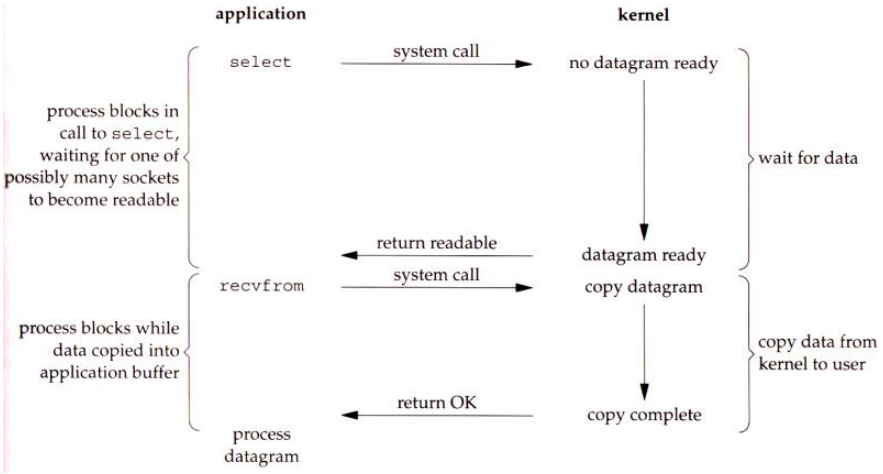


Σχήμα 3.2 : Non-blocking I/O

3.1.3.2 Non-blocking I/O

Έπειτα, η αμέσως επόμενη επιλογή που έχουμε, είναι η χρήση και πάλι της συνάρτησης συστήματος accept(), όμως σε συνδυασμό με την χρήση παραμέτρων για την υποδοχή

(socket), οι οποίες θα δηλώνουν στον πυρήνα του λειτουργικού ότι δεν πρέπει να αφήνει τις κλήσεις συναρτήσεων που αφορούν τη συγκεκριμένη υποδοχή να κολλάνε. Επομένως, όταν μία κλήση σε συνάρτηση E/E (Εισόδου/Εξόδου), πρόκειται να φέρει την εκτέλεση μας σε σημείο όπου θα κολλήσει, τότε η συνάρτηση επιστρέφει με ένα κωδικό λάθους, συγκεκριμένα τον κωδικό «EWOULDBLOCK». Με αυτό τον τρόπο, μπορεί η εκτέλεση του εξυπηρετητή μας να μην κολλάει, αλλά να επιστρέφει και έτσι να μπορούμε εμείς να βάλουμε την εκτέλεση μας να σε αναμονή, έτσι ώστε να μην επιβαρύνουμε τον επεξεργαστή μας. Όμως, το μειονέκτημα της προσέγγισης αυτής, είναι το γεγονός ότι με το να κάνεις συνεχώς κλήσεις συστήματος έχεις αρκετή επιβάρυνση του επεξεργαστή με μη ωφέλιμες εντολές, αλλά επίσης και την ώρα που επιστρέφει η κλήση της συνάρτησης με τον κωδικό λάθους, τότε έχουμε καθυστέρηση μέχρι να ξαναζητήσουμε από τον πυρήνα να μας πει εάν έχει φτάσει άλλη αίτηση. Αυτό θα έχει ως αποτέλεσμα να αργήσουμε να εξυπηρετήσουμε κάποιες αιτήσεις, που αυτό το πρόβλημα γίνεται μεγαλύτερο εάν βάζουμε την εφαρμογή μας σε κατάσταση αναμονής, με αποτέλεσμα να αργεί ακόμα περισσότερο να εξυπηρετήσει τυχών αιτήσεις.

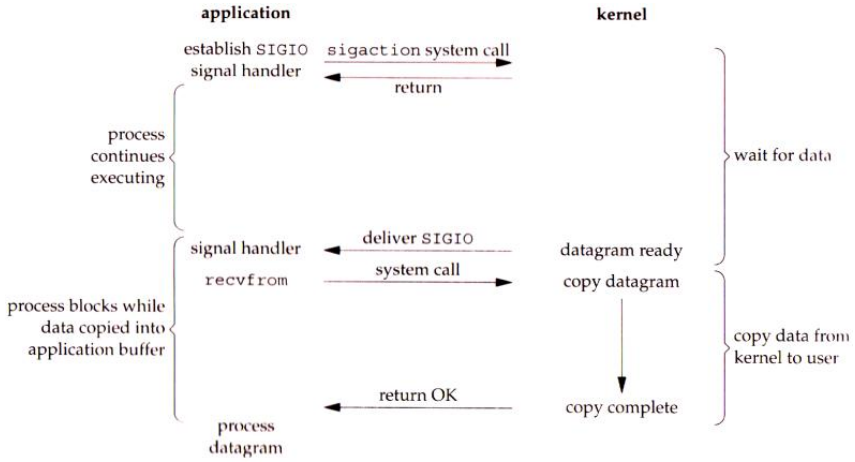


Σχήμα 3.3 : I/O Multiplexing (select)

3.1.3.3 I/O Multiplexing (select and poll)

Σε σχέση με την προηγούμενη προσέγγιση, υπάρχει και η κλήση συστήματος poll(), η οποία κάνει αυτό ακριβώς που αναφέραμε. Δηλαδή, προσπαθεί να δει εάν υπάρχει κάποια εισερχόμενη αίτηση από κάποιο πελάτη στον εξυπηρετητή μας, και εάν υπάρχει,

τότε επιστρέφει ένα θετικό κωδικό, για να ξέρουμε ότι πρέπει να κάνουμε χρήση της accept(), μιας και τώρα είμαστε σίγουροι ότι η accept() θα επιστρέψει αμέσως με τα τις κατάλληλες δομές για την εξυπηρέτηση κάποιου πελάτη. Πολύ παρόμοια με την poll(), είναι και η select(), με την διαφορά όμως ότι αντί να ελέγχει συνέχεια για νέες εισερχόμενες αιτήσεις, ελέγχει μία φορά και όταν υπάρχει κάποια, τότε και μόνο τότε επιστρέφει η συνάρτηση (εκτός σε περιπτώσεις σφάλματος) και ακολούθως κάνουμε κλήση της accept(). Η διαφορά αυτών των δύο συναρτήσεων, είναι ότι η εκτέλεση μας δεν κολλάει στην κλήση συστήματος για Ε/Ε αυτή καθ' αυτή, αλλά κολλάει στην κλήση μίας εκ των δύο προαναφερθέντων συναρτήσεων συστήματος. Το καλό με την χρήση της select(), είναι ότι μπορούμε να περιπλέξουμε (multiplexing) περισσότερες από μία υποδοχές και να τις διαχειριζόμαστε όλες με μία κλήση προς την select().

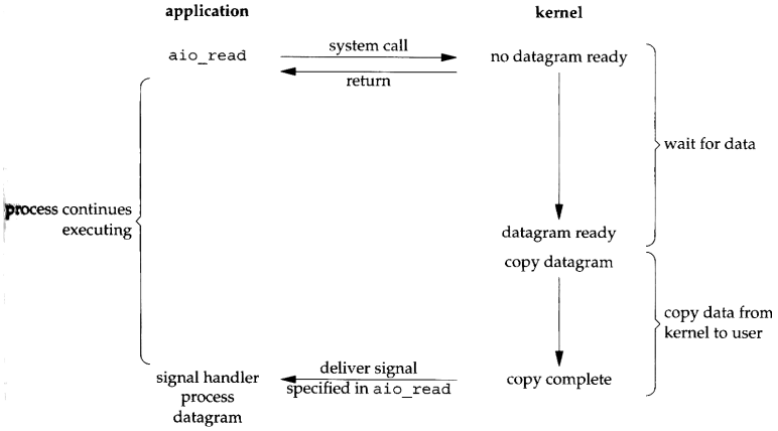


Σχήμα 3.4 : Signal driven I/O (SIGIO)

3.1.3.4 Signal driven I/O (SIGIO)

Μπορούμε επίσης να χρησιμοποιήσουμε signals για να πούμε στον πυρήνα να μας ειδοποιήσει όταν είναι έτοιμος ο περιγραφέας αρχείου για να διαβάσουμε από μέσα. Αυτό σημαίνει ότι κάποια δεδομένα έχουν σταλεί από τον πελάτη στην υποδοχή, αυτά τα δεδομένα έχουν αντιγραφεί στο χώρο του χρήστη στην κύρια μνήμη από τον χώρο του πυρήνα, άρα είναι έτοιμα να διαβαστούν από τον περιγραφέα αρχείου που έχουμε για τη συγκεκριμένη υποδοχή. Σε αυτή την προσέγγιση, δηλώνουμε μία συνάρτηση η οποία θα καλείται όταν πάρουμε ειδοποίηση (signal) από τον πυρήνα του λειτουργικού ότι έχει έτοιμα τα δεδομένα από τον πελάτη στον χώρο μνήμης του χρήστη, και έπειτα θα καταχωρήσουμε την αίτηση για τον διαχειριστή των ειδοποίηση (register the signal

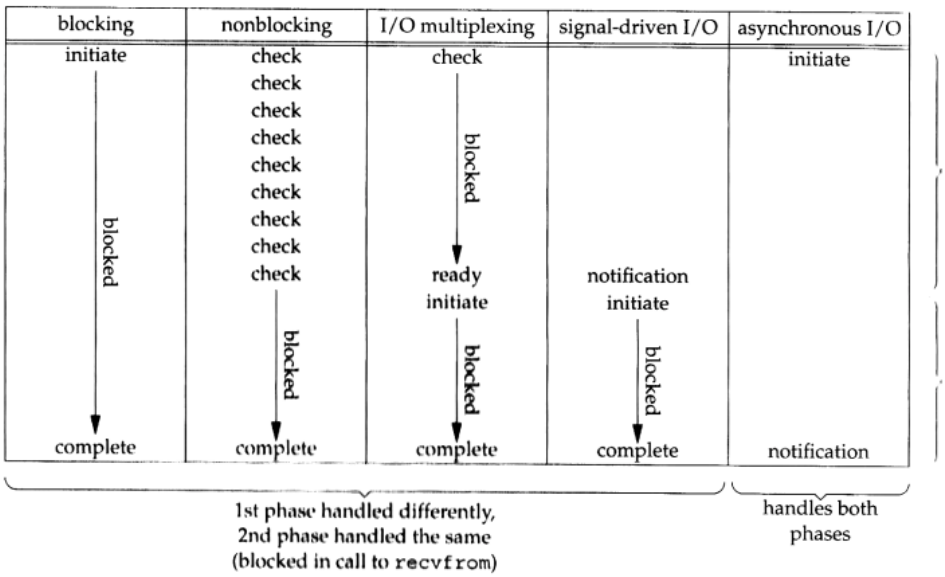
handler). Το πρόβλημα αυτής της προσέγγισης είναι ότι δεν είναι εύκολο να χρησιμοποιηθεί σε αρχιτεκτονική η οποία περιλαμβάνει πολλαπλά νήματα ελέγχου, αφού οι ειδοποιήσεις προορίζονται για τις διεργασίες και όχι για τα νήματα.



Σχήμα 3.5 : Asynchronous I/O Model

3.1.3.5 Asynchronous I/O (the POSIX aio_ functions)

Η διαφορά με την προηγούμενη προσέγγιση, είναι ότι ο πυρήνας εδώ μας λέει όταν τελειώσει η μεταφορά δεδομένων από το χώρο του πυρήνα, στον χώρο του χρήση, ενώ στην προηγούμενη περίπτωση μας ειδοποιούσε όταν άρχιζε να κάνει την μεταφορά των δεδομένων.



Σχήμα 3.6 : Asynchronous I/O Model

3.1.3.6 Σύγκριση των πέντε προσεγγίσεων

Όπως βλέπουμε, η τελευταία προσέγγιση με την χρήση των POSIX aio_ συναρτήσεων θα ήταν η ιδανικότερη. Το πρόβλημα είναι ότι δεν μπορούμε να καλέσουμε την accept() με αυτό τον τρόπο, αλλά μόνο για διάβασμα και γράψιμο σε ένα υποδοχέα. Το επόμενο που θα πληρούσε επάξια τις ανάγκες μας, είναι η χρήση του SIGIO, με την δήλωση κάποιου διαχειριστή ειδοποιήσεων ο οποίος θα αναλαμβάνει να εκτελέσει τις πράξεις που έπονται μετά την παραλαβή κάποιας αίτησης από κάποιο πελάτη. Αυτή όμως η προσέγγιση είναι λίγο δύσκολα υλοποιήσιμη, αφού δεν θα μπορούσαμε με εύκολο τρόπο να διαχειριστούμε τις αιτήσεις σε πολλαπλά νήματα ελέγχου, αφού οι ειδοποιήσεις αφορούν το επίπεδο των διεργασιών του συστήματος και η διαδικασία για να φτιάξεις μία υλοποίηση με πολλαπλά νήματα ελέγχου και ειδοποιήσεις ξεχωριστά για κάθε νήμα, είναι αρκετά περίπλοκη και επίπονη.

Έχοντας αναφέρει αυτά, αυτό που επιλέξαμε να χρησιμοποιήσουμε, ήταν η κλήση της συνάρτησης select(), η οποία κολλάει και επιστρέφει μόνο σε περιπτώσεις σφάλματος ή όταν έχει έρθει μία νέα αίτηση από κάποιο πελάτη. Επίσης, η χρήση της select(), επιτρέπει την πολυπλεξία, η οποία θα μπορούσε να μας προσφέρει την ταυτόχρονη ακρόαση σε πολλαπλούς υποδοχείς, εάν αυτό ήταν αναγκαίο από την υλοποίηση του εξυπηρετητή μας. Εκτός από αυτό, η select() προσφέρει ακόμα μία δυνατότητα, σου λέει πόσες αιτήσεις έχουν έρθει συνολικά και βρίσκονται στον χώρο μνήμης του πυρήνα, στην ουρά που έχει οριστεί από την συνάρτηση συστήματος listen(), η οποία και καθορίζει το μέγιστο μέγεθος αυτής της ουράς. Έτσι, μπορούμε να ξέρουμε τον ακριβή αριθμό κλήσεων προς την accept(), κάθε φορά που μας επιστέφει χωρίς σφάλμα η select(). Οι άλλες δύο περιπτώσεις, του blocking I/O, αλλά και του non-blocking I/O, δεν επιλέγονται για τον λόγο ότι στην πρώτη περίπτωση κολλάει η εκτέλεση του εξυπηρετητή μας μέσα στο σώμα της συνάρτησης συστήματος που διαχειρίζεται το I/O, ενώ στην δεύτερη περίπτωση έχουμε μη αποδοτική χρήση του χρόνου του επεξεργαστή, αφού γίνονται αλλεπάλληλες κλήσεις προς τον πυρήνα, χωρίς να υπάρχει κάποιος λόγος, αφού δεν υπάρχει κάποια εισερχόμενη αίτηση.

3.1.4 Αρχιτεκτονική Εξυπηρετητή

3.1.4.1 Μέθοδοι διαχείρισης συνδέσεων σε εξυπηρετητές

Ο εξυπηρετητής μας, αποτελείται από ένα κομμάτι, το οποίο είναι πολύ σημαντικό, όσο αφορά το θέμα της επίδοσης του. Θα μπορούσαμε να έχουμε με δύο τρόπους τη δυνατότητα για να διαχειριζόμαστε πολλές συνδέσεις προς τον εξυπηρετητή μας. Ο ένας τρόπος είναι με πολλές διεργασίες, οι οποίες θα αναλαμβάνουν την επικοινωνία του εξυπηρετητή με τους πελάτες, δηλαδή θα αναλαμβάνουν να απαντήσουν στην επερώτηση που θέτει ο πελάτης. Ο δεύτερος τρόπος και σχετικά ποιο φτηνός, είναι η χρήση νημάτων για την εξυπηρέτηση των πελατών. Συγκεκριμένα, θα αναλύσουμε δύο τεχνικές οι οποίες χρησιμοποιούν τα νήματα για την παροχή ταυτοχρονίας στον εξυπηρετητή μας.

i. Τεχνική Α: Δημιουργία ενός καινούριου νήματος για την επεξεργασία μιας αίτησης, ενώ το αρχικό νήμα περιμένει για νέες αιτήσεις.
ii. Τεχνική Β: Δημιουργία ενός thread-pool για την διαχείριση των νέων αιτήσεων. Για την περίπτωση αυτή θα δούμε δύο διαφορετικές προσεγγίσεις και θα πούμε περισσότερα γι' αυτή που επιλέξαμε να υλοποιήσουμε.

Τεχνική Α

Η προσέγγιση αυτή αποτελεί τον απλούστερο τρόπο πολυνηματικής εφαρμογής. Υπάρχει το αρχικό νήμα το οποίο περιμένει αιτήσεις από πελάτες (εκτός από το νήμα αυτό μπορεί να υπάρχουν και άλλα νήματα τα οποία διαχειρίζονται άλλου είδους εργασίες πέραν της εξυπηρέτησης πελατών, όπως π.χ., της διαχείρισης γραμμής εντολών). Για κάθε εισερχόμενη αίτηση το αρχικό νήμα δημιουργεί ένα καινούργιο νήμα το οποίο εξυπηρετεί τον αιτητή, και με το πέρας της αποστολής του το νήμα αυτό τερματίζει τη λειτουργία του.

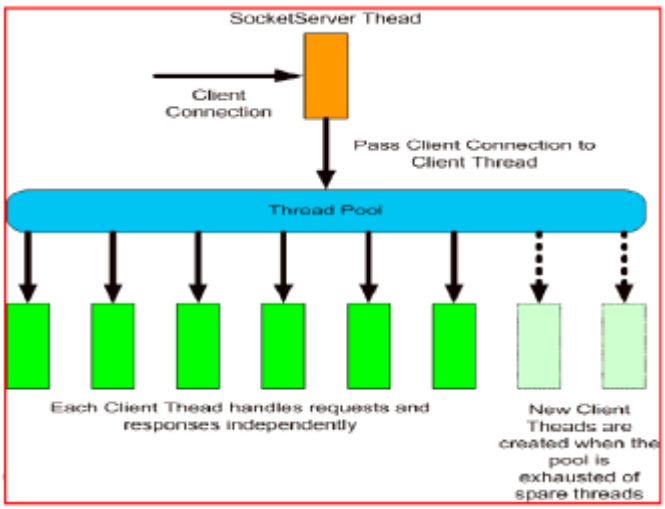
Η προσέγγιση αυτή δεν είναι πολύ καλή για τους ακόλουθους λόγους:

1. Η δημιουργία και καταστροφή νημάτων δεν είναι ιδιαίτερα ελεγχόμενη, γεγονός που μπορεί να αποβεί εξαιρετικά προβληματικό σε κάποιες περιπτώσεις.

2. Η δημιουργία ενός καινούργιου νήματος είναι ακριβή (παρόλο που είναι πιο φθηνή από τη δημιουργία μιας νέας διεργασίας). Στην συνεχεία πρέπει να αποδεσμεύσουμε τα δεδομένα που σχετίζονται με τον νήμα και τέλος να το απελευθερώσουμε, γεγονός που επιφέρει μεγαλύτερο κόστος
3. Ανά πάσα στιγμή δεν μπορούμε να ξέρουμε τον μέγιστο αριθμό νημάτων, επομένως μπορεί η διεργασία του εξυπηρετητή να ξεπεράσει το επιτρεπτό όριο μνήμης με αποτέλεσμα να χάσουμε την υπηρεσία.

Τεχνική Β

Η δεύτερη προσέγγιση αφορά τη δημιουργία ενός thread-pool από το αρχικό νήμα. Με τον όρο thread-pool, αναφερόμαστε στην δημιουργία ενός σταθερού αριθμού νημάτων-εργατών (ο αριθμός επιλέγεται ανάλογα με την εφαρμογή). Τα νήματα που βρίσκονται στο thread-pool συναγωνίζονται για την εξυπηρέτηση κάθε καινούργιας αίτησης. Δηλαδή κάθε καινούργια αίτηση ανατίθεται σε κάποιο από τα νήματα που δεν έχει δουλειά. Τα νήματα του thread-pool, αφού εξυπηρετήσουν ένα πελάτη, δεν τερματίζουν, αλλά μεταβαίνουν σε κατάσταση αναμονής για να εξυπηρετήσουν άλλο πελάτη. Αν δεν υπάρχει διαθέσιμο νήμα, το αρχικό θα πρέπει να περιμένει μέχρι να υπάρξει, χωρίς να δέχεται νέες αιτήσεις. Πιο συγκεκριμένα εάν υπάρξουν περισσότερες αιτήσεις από το μέγιστο αριθμό νημάτων στο pool τότε το σύστημα απορρίπτει την αίτηση κλείνοντας το socket (χωρίς να επιστρέφει οποιανδήποτε απάντηση στον πελάτη).



Σχήμα 3.7 : Παράδειγμα Thread Pool Server

Για τη διαχείριση των νημάτων του pool και την εξυπηρέτηση των αιτήσεων μπορούμε να χρησιμοποιήσουμε μεταξύ άλλων τις πιο κάτω μεθόδους:

Τεχνική Β1: Μοντέλο παραγωγού-καταναλωτή:

1. Τα νήματα εργάτες δημιουργούνται από την αρχή στον εξυπηρετητή και τα thread IDs τους αποθηκεύονται σε ένα πίνακα.
2. Ο έλεγχος ταυτοχρονίας υπακούει στο μοντέλο παραγωγού καταναλωτή (producer - consumer) χρησιμοποιώντας ένα αριθμό δυαδικών σηματοφόρων (τα οποία προσφέρονται από την βιβλιοθήκη νημάτων). Τα νήματα τα οποία δεν εργάζονται σε κάποια δεδομένη στιγμή είναι μπλοκαρισμένα πάνω σε ένα από αυτούς τους σηματοφόρους. Ο σηματοφόρος που είναι υπεύθυνος για αυτή τη δουλειά κλειδώνεται πριν τη δημιουργία των νημάτων εργατών, με αποτέλεσμα τα νήματα να είναι μπλοκαρισμένα πριν την αποδοχή μιας αίτησης. Μόλις έρθει μια νέα σύνδεση, το κυρίως νήμα ξεκλειδώνει αυτό το σηματοφόρο και το νήμα εργάτης που αναλαμβάνει την αίτηση τον ξανακλειδώνει.
3. Οι υπόλοιποι σηματοφόροι χρησιμοποιούνται για έλεγχο των κρίσιμων σημείων στο κώδικα.
4. Για να μπορέσουμε να περάσουμε το new_socket που δημιουργείται από την κλήση συστήματος accept(), στο κυρίως νήμα, σε ένα από τα νήματα εργάτες ορίζουμε το new_socket ως καθολική μεταβλητή.

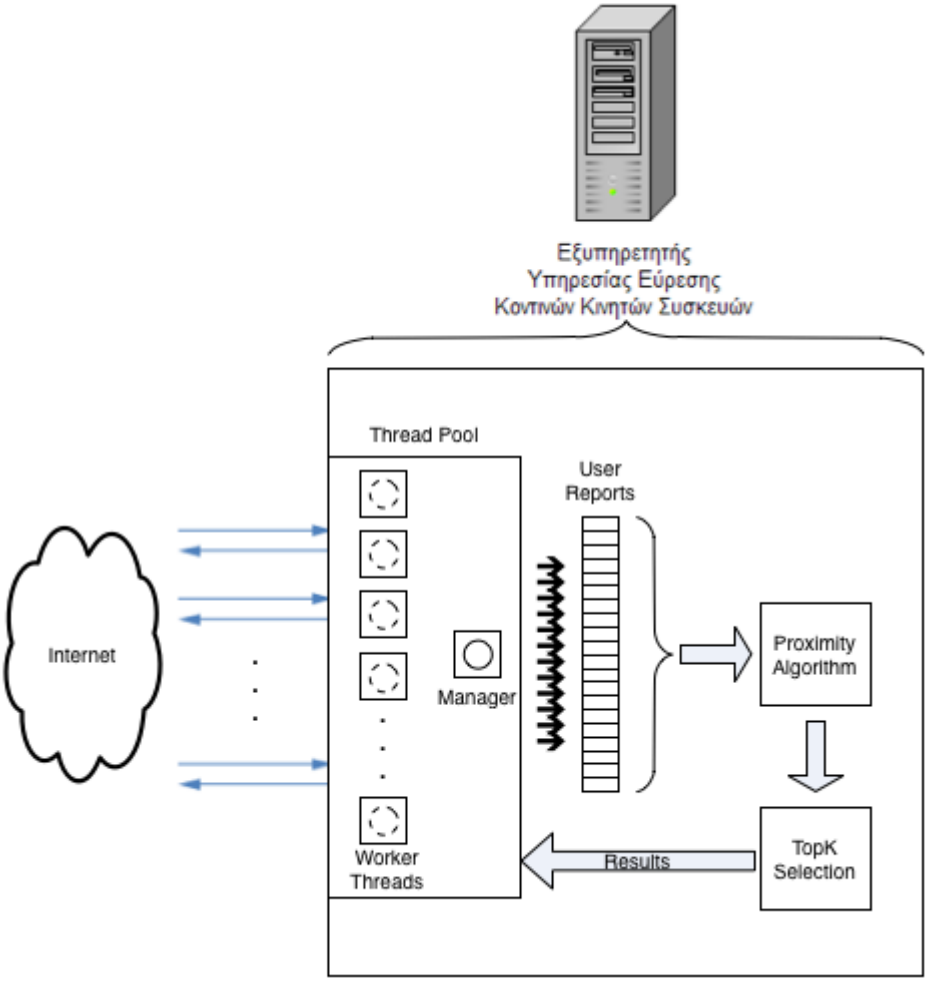
Τεχνική Β2: Χρήση Ουράς από Εργασίες:
Στη μέθοδο αυτή, το thread-pool υλοποιείται ως μια ουρά από εργασίες που εκκρεμούν. Είναι μια δομή στην οποία πέραν από τον πίνακα με τα νήματα εργάτες και τους σηματοφόρους ελέγχου έχουμε και μια ουρά εργασιών. Κάθε εργασία αναπαριστάται από μια δομή δεδομένων (struct) η οποία περιέχει μεταξύ άλλων τη συνάρτηση την οποία θα καλεστεί από το νήμα εργάτη, τις παραμέτρους της συνάρτησης αυτής και δείκτη στην επόμενη εργασία.

Στη μέθοδο αυτή, όταν η ουρά δεν έχει εργασίες για να τύχουν επεξεργασίας, τα νήματα εργάτες είναι αδρανή. Με την είσοδο μιας εργασίας στην ουρά ο σηματοφόρος ξεκλειδώνει και ένα νήμα εργάτης αναλαμβάνει την εξυπηρέτηση του.

Για να πετύχουμε τα πιο πάνω χρειαζόμαστε μια συνάρτηση η οποία θα είναι υπεύθυνη για την εισαγωγή διεργασιών και ξεκλειδώματος των σηματοφόρων. Επίσης χρειαζόμαστε μια άλλη συνάρτηση η οποία θα τρέχει τον κώδικα των νημάτων.

Η Τεχνική η οποία επιλέξαμε ήταν η B2, η οποία είναι η καλύτερη σε σχέση με τις υπόλοιπες, για τον λόγο που ότι τα νήματα δεν καταστρέφονται, αλλά επαναχρησιμοποιούνται, αλλά επίσης και για τον λόγο ότι οι εργασίες που εκκρεμούν μπαίνουν σε μία ουρά, και όποιο νήμα προλάβει να πάρει το κλείδωμα για την ουρά από τις εργασίες, μπορεί να πάρει στο τέλος την διεργασία και να την εξυπηρετήσει.

3.1.4.2 Επεξήγηση Αλγορίθμου Εξυπηρετητή

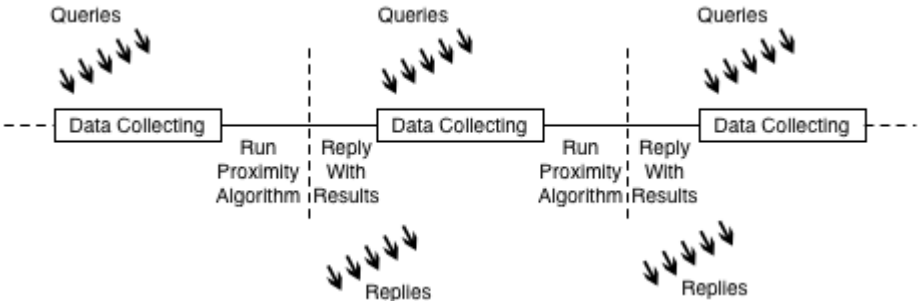


Σχήμα 3.8 : Διάγραμμα Εξυπηρετητή Proximity

Στο πιο πάνω σχεδιάγραμμα, βλέπουμε τον αλγόριθμο που ακολουθεί ο εξυπηρετητής μας. Έχουμε στα αριστερά το νέφος το οποίο αντικατοπτρίζει το Διαδίκτυο. Έπειτα βλέπουμε τις αιτήσεις που έρχονται από τους διάφορους πελάτες και ζητούν εξυπηρέτηση από τον εξυπηρετητή μας. Υπάρχει ένα κεντρικό νήμα ελέγχου το οποίο αναλαμβάνει να δεχτεί τις αιτήσεις αυτές. Αυτό συγκεκριμένα είναι το νήμα που χρησιμοποιεί την συνάρτηση select(), η οποία επεξηγήθηκε σε προηγούμενο υποκεφάλαιο. Οι αιτήσεις, όταν γίνουν δεκτές από το κεντρικό νήμα ελέγχου, μπαίνουν σε μία ουρά, όπως αναφέρεται και στο προηγούμενο εδάφιο, με σκοπό να δοθούν για εξυπηρέτηση σε κάποιο νήμα από αυτά που υπάρχουν στο Thread Pool. Όταν έρθει κάποια αίτηση, υπάρχει ένα νήμα, το οποίο φαίνεται στο σχήμα (manager), το οποίο αναλαμβάνει να ειδοποιήσει με σήμα (notify) τα νήματα στο Thread Pool ότι υπάρχει δουλειά που περιμένει να εξυπηρετηθεί. Μόλις γίνει αυτό, τότε τα νήματα που είναι ελεύθερα αρχίζουν να διαγωνίζονται για να πάρουν το κλείδωμα το οποίο διαφυλάσσει την ασφαλή προσπέλαση της λίστας με τις εισερχόμενες αιτήσεις. Όταν κάποιο νήμα καταφέρει να πάρει το κλείδωμα για την λίστα, τότε μπορεί να διαβάσει τα στοιχεία για την εργασία που έχει να επιτελέσει, δηλαδή τον πελάτη που έχει αναλάβει να εξυπηρετήσει.

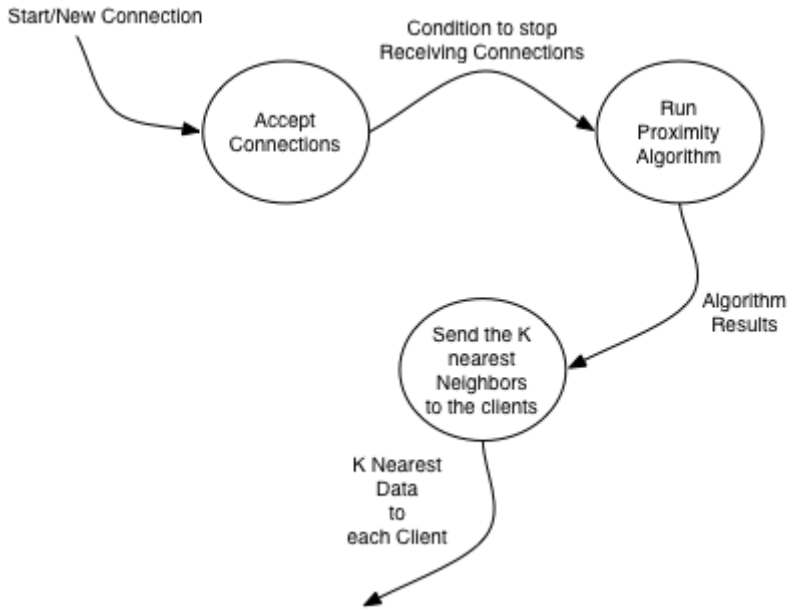
Στη συνέχεια, αρχίζει μία επικοινωνία μεταξύ του εξυπηρετητή και του πελάτη. Κατά την διάρκεια αυτής της επικοινωνίας ανταλλάζονται τα απαραίτητα δεδομένα τα οποία είναι αναγκαία από τον εξυπηρετητή, και συγκεκριμένα από τους αλγορίθμους οι οποίοι θα εκτελεστούν για να βρουν τους κοντινότερους γειτονικούς κόμβους. Αυτά τα δεδομένα, όπως αναφέρθηκε και στην ανάλυση των δύο αλγορίθμων που χρησιμοποιούμε, είναι στοιχεία που αφορούν την γεωγραφική θέση του πελάτη, πληροφορία η οποία αποτελεί μέρος του επερωτήματος που θέτει ο χρήστης προς τον εξυπηρετητή. Συγκεκριμένα, περιλαμβάνει το γεωγραφικό μήκος, γεωγραφικό πλάτος, ακρίβεια μέτρησης γεωγραφικής θέσης, Επίσης, περιέχει την πληροφορία για την κυψέλη κινητής τηλεφωνίας με την οποία είναι συνδεδεμένη η κινητή συσκευή του χρήστη, καθώς και τις γειτονικές κυψέλες που μπορεί να βλέπει από την θέση που βρίσκεται η κινητή συσκευή. Αυτά τα στοιχεία χρησιμοποιούνται μόνο από τον ένα αλγόριθμο τον οποίο αναπτύξαμε στα πλαίσια αυτής της διπλωματικής εργασίας.

Όταν οι απαραίτητες πληροφορίες για την λειτουργία του αλγορίθμου συλλεχτούν από τον πελάτη, τότε το νήμα το οποίο τον εξυπηρετεί καταχωρεί ένα νέο κόμβο σε μία λίστα, η οποία περιλαμβάνει όλες τις αναφορές χρηστών, με τα στοιχεία που σύλλεξε και αφορούν τον πελάτη. Έπειτα το νήμα το οποίο έχει πλέον εισάγει τα στοιχεία που σύλλεξε, πρέπει να περιμένει να εκτελεστεί ο αλγόριθμος. Ένα σχεδιάγραμμα ακολουθεί, το οποίο δείχνει αυτή τη συμπεριφορά της αρχιτεκτονικής μας. Είναι γνωστή ως αλγόριθμος του βαγονιού.



Σχήμα 3.9 : Αλγόριθμος Βαγονιού

Συγκριμένα, στο σχεδιάγραμμα βλέπουμε να έχουμε αιτήσεις οι οποίες έρχονται κατά τη διάρκεια της συλλογής στοιχείων/δεδομένων από τους χρήστες. Εκεί βλέπουμε ότι σε κάποια στιγμή (η οποία θα μπορούσε να σημαίνει την λήξη κάποιου χρονομετρητή, ή την λήψη κάποιου ελάχιστου αριθμού από αιτήσεις κ.α.) το παράθυρο αποδοχής των αιτήσεων σταματά να λειτουργεί και εκεί αρχίζει να εκτελεί ένα άλλο νήμα τον αλγόριθμο εύρεσης των κοντινότερων γειτονικών συσκευών. Μόλις φέρει εις πέρας την επεξεργασία των δεδομένων και έχει τα αποτελέσματα από την εκτέλεση του αλγορίθμου, επιστρέφει τις απαντήσεις αυτές, αφού πρώτα τις ταξινομήσει, στα νήματα τα οποία διαχειρίζονται τις συνδέσεις με τους πελάτες. Συγκεκριμένα, είναι τα ίδια νήματα τα οποία είχαν λάβει αρχικά τα δεδομένα που χρειάζονται οι αλγόριθμοι για να τρέξουν. Έπειτα, τα νήματα αυτά αναλαμβάνουν να στείλουν τα αποτελέσματα του αλγορίθμου πίσω στους πελάτες, επικοινωνώντας με τον ίδιο υποδοχέα, τον οποίο είχαν αρχίσει να επικοινωνούν από την αρχή της σύνδεσης και έπειτα τερματίζουν αυτή την σύνδεση που είχαν με τον πελάτη.



Σχήμα 3.10 : Λογική σύνδεσης από πελάτη

Τέλος, εδώ βλέπουμε τις καταστάσεις από τις οποίες περνάει μία σύνδεση η οποία είναι εισερχόμενη προς τον εξυπηρετητή μας. Αρχικά εγκαθιδρύεται η σύνδεση, έχουμε το σήμα για παύση της αποδοχής νέων αιτήσεων, έπειτα τον υπολογισμό των TopK για κάθε πελάτη και τέλος την αποστολή των αποτελεσμάτων από κάθε υπεύθυνο νήμα προς τον πελάτη τον οποίο εξυπηρετεί.

3.2 Πελάτης

3.2.1 Γλώσσα προγραμματισμού

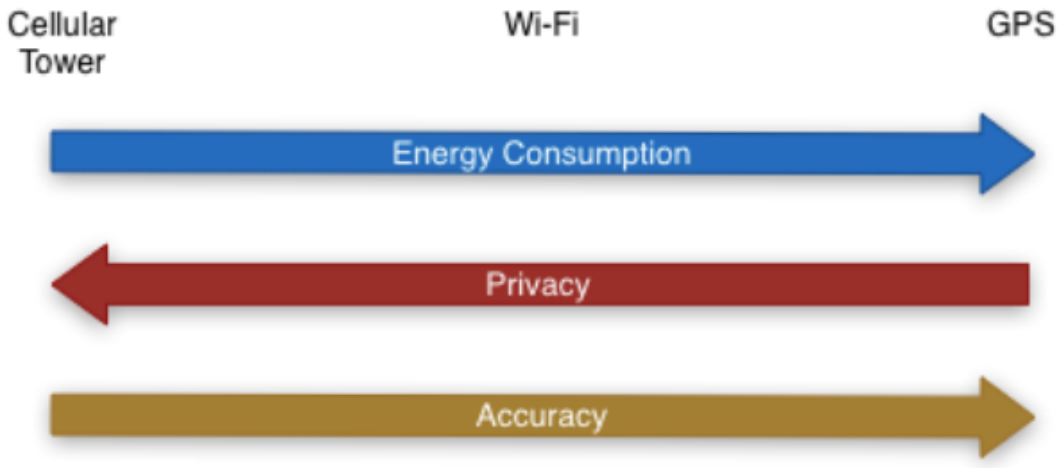
Η γλώσσα προγραμματισμού που χρησιμοποιήσαμε για την ανάπτυξη της εφαρμογής μας, την οποία είχαμε φτιάξει για την πλατφόρμα κινητών συσκευών, με λειτουργικό σύστημα Android, είναι η γλώσσα Java, η οποία είναι η επίσημη γλώσσα για προγραμματισμό στην πλατφόρμα αυτή. Θα ήταν καλό να αναφέρουμε ότι υπάρχει η δυνατότητα χρήσης γλώσσας προγραμματισμού C, οι οποίες όμως σε αυτή την περίπτωση, δεν προσφέρεται η δυνατότητα για χρήση γραφικού περιβάλλοντος, πράγμα το οποίο ήταν κάτι που η εφαρμογή μας έπρεπε να υλοποιεί, δηλαδή την διαφάνεια αλληλεπίδρασης των χρηστών με την οθόνη του κινητού. Έτσι και αλλιώς η Java και οι βιβλιοθήκες που παρέχονται από την εταιρία η οποία είναι ο κατασκευαστής του λειτουργικού συστήματος Android, η οποία είναι η Google, είναι αρκετά πλήρης και

δεν χρειάζεται οποιαδήποτε διαδικασία για να μπορούμε να τρέξουμε το πρόγραμμα μας στην συσκευή, παρά μόνο να γίνει η εγκατάσταση του λογισμικού μας πάνω στην συσκευή.

3.2.2 Περιγραφή Αλγορίθμου

Στο υποκεφάλαιο αυτό, θα αναφέρουμε γενικά τον αλγόριθμο που θα πρέπει να ακολουθεί κάποια υλοποίηση πελάτη, η οποία θέλει να κάνει χρήση του εξυπηρετητή μας. Όπως είχε αναφερθεί και στην εισαγωγή, ο αλγόριθμος στον εξυπηρετητή, χρειάζεται να έχει δεχτεί κάποιο αριθμό αιτήσεων από τους πελάτες, μέσω δικτύου, οι οποίες να περιέχουν κάποια συγκεκριμένα χαρακτηριστικά/δεδομένα, τα οποία αφορούν τον κάθε πελάτη ο οποίος κάνει την αίτηση.

Συγκεκριμένα, τα δεδομένα που πρέπει να αποστέλλει ο πελάτης στον εξυπηρετητή, είναι κυρίως η γεωγραφική του θέση, αποτελούμενη από το γεωγραφικό μήκος και γεωγραφικό πλάτος της συσκευής, καθώς επίσης και προαιρετικά την μέτρηση του λάθους/απόκλισης που μπορεί να έχει η μέτρηση της γεωγραφικής θέσης από την πραγματική του θέση, που αυτό είναι ανάλογο με τον τρόπο που έγινε ο υπολογισμός της θέσης του. Εάν δηλαδή έχει γίνει με τον δέκτη GPS, θα είναι μία πολύ μικρή τιμή, εάν έχει γίνει με το Wi-Fi, θα είναι λίγο μεγαλύτερη τιμή και τέλος εάν έχει γίνει με την πληροφορία από τις κυψέλες κινητής τηλεφωνίας θα είναι ακόμα πιο μεγάλη η τιμή αυτή. Επίσης, ο κάθε πελάτης αποστέλλει τον αριθμό της κυψέλης κινητής τηλεφωνίας με την οποία είναι συνδεδεμένος, καθώς επίσης και πιθανές γειτονικές κυψέλες, από τις οποίες μπορεί να λάβει κάποιο σήμα, όμως δεν είναι συνδεδεμένος μαζί τους.

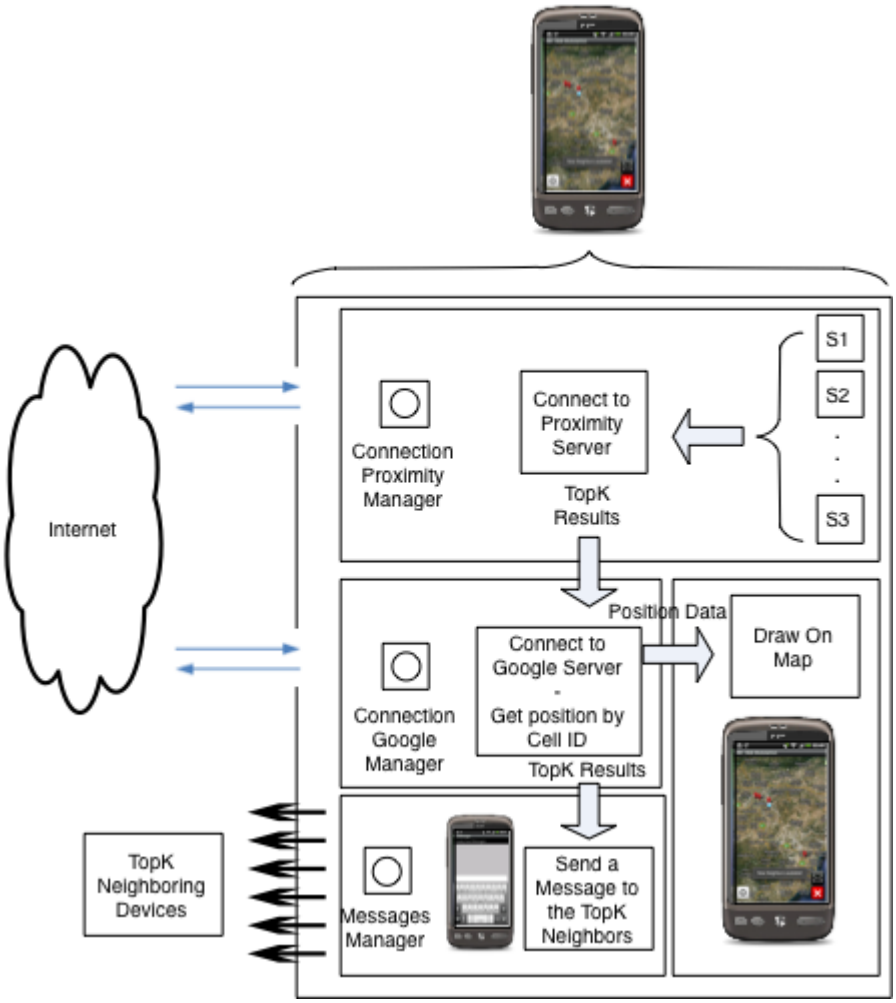


Σχήμα 3.11 : Συμβιβασμός για καταγραφή Γεωγραφικής τοποθεσίας

Στο παραπάνω σχήμα, βλέπουμε τον συμβιβασμό που έχουμε μεταξύ της κατανάλωσης ενέργειας, της μυστικότητας και της ακρίβειας της γεωγραφικής θέσης που λαμβάνουμε όταν χρησιμοποιούμε ένα από τους τρείς τρόπους για καθορισμό της γεωγραφικής θέσης της συσκευής μας. Άρα, ανάλογα με το αποτέλεσμα που θέλουμε να πετύχουμε, χρησιμοποιούμε ανάλογα τον τρόπο που μας καλύπτει.

3.2.3 Αρχιτεκτονική Πελάτη

Εδώ θα αναλύσουμε την αρχιτεκτονική που έχουμε υλοποιήσει συγκεκριμένα στην εφαρμογή η οποία έγινε για σκοπούς επίδειξης, στα πλαίσια των δραστηριοτήτων του Πανεπιστημίου Κύπρου, αλλά και για σκοπούς απόδειξης σημαντικότητας της χρήσης της πληροφορίας που αφορά την γειτνίαση κινητών συσκευών.



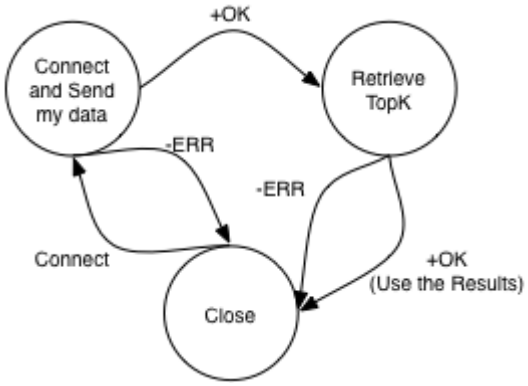
Σχήμα 3.12 : Διάγραμμα WebModulator Εφαρμογής σε Android OS 2.2 (Froyo)

Στο πιο πάνω σχεδιάγραμμα βλέπουμε την αρχιτεκτονική που ακολουθείται από τον πελάτη, την εφαρμογή που φτιάξαμε για την κινητή συσκευή HTC Desire με λειτουργικό σύστημα Android 2.2 (Froyo).

Τα βήματα που ακολουθεί η εφαρμογή, είναι να συλλέξει πρώτα τα στοιχεία που θα χρειαστεί να αποστείλει στον εξυπηρετητή, τα οποία προέρχονται από τους αισθητήρες που υπάρχουν στην συσκευή. Οι συγκεκριμένοι αισθητήρες αφορούν το GPS της συσκευής, την κεραία για λήψη τηλεφωνικού σήματος και τέλος τον αισθητήρα που βασικά είναι η κάρτα ασύρματου δικτύου. Στην συνέχεια, μόλις έχει συλλέξει όλα τα στοιχεία που χρειάζεται, αποστέλλει αυτά, μαζί με μία θύρα, στην οποία θα ακούει για εισερχόμενα μηνύματα από γειτονικές συσκευές. Έπειτα, παίρνει απάντηση από τον εξυπηρετητή μας, ο οποίος έχει υπολογίσει τις κοντινότερες K συσκευές προς την συσκευή η οποία είχε αποστείλει την επερώτηση και τις αποστέλλει στην συσκευή.
Μόλις η συσκευή παραλάβει αυτές τις κοντινότερες K συσκευές προς αυτή, δημιουργεί μία σειρά από JSON αιτήσεις προς κάποια υπηρεσία του Google, η οποία δίνοντας της στοιχεία όπως το IP της γειτονικής συσκευής, διάφορα στοιχεία για την κυψέλη στην οποία είναι συνδεδεμένη, όπως τον κωδικό της, το MCC, MNC, τα οποία είναι το Mobile Country Code και το Mobile Network Code, και είναι βασικά αναγνωριστικοί κωδικοί για την χώρα και τον παροχέα υπηρεσιών εντός της συγκεκριμένης χώρας, αντίστοιχα. Αυτό που επιστρέφει ο εξυπηρετητής του Google, είναι στοιχεία που αφορούν την γεωγραφική θέση της γειτονικής συσκευής, εμφωλευμένα μέσα σε μία απάντηση JSON.

Αυτά τα στοιχεία, η συσκευή μπορεί να τα χρησιμοποιήσει με σκοπό να σχεδιάσει προσεγγιστικά την θέση της κάθε κοντινής συσκευής από το σύνολο TopK το οποίο της είχε επιστέψει η εκτέλεση του αλγορίθμου γειτνίασης στον εξυπηρετητή μας. Λόγω του ότι δεν επιστρέφονται από τον εξυπηρετητή μας στοιχεία που αφορούν την ακριβή τοποθεσία του γειτονικού χρήστη, όπως αυτός την έχει αποστείλει στον εξυπηρετητή. Με αυτό τον τρόπο επιτυγχάνουμε και κάποιας μορφής απόκρυψη της θέσης του γειτονικού χρήστη, αφού το η μόνη τοποθεσία που μπορεί να βρει ο χρήστης που έκανε την επερώτηση, είναι η θέση της κεραίας της κυψέλης κινητής τηλεφωνίας στην οποία είναι συνδεδεμένος ο γειτονικός χρήστης.

Παραπέρα από την παρουσίαση της θέσης του γειτονικού κόμβου, ο χρήστης έχει την επιλογή να αποστείλει μήνυμα κειμένου στους γειτονικούς κόμβους οι οποίοι του έχουν επιστραφεί από την εκτέλεση του αλγορίθμου της γειτνίασης. Ο τρόπος που το κάνει είναι να επιλέξει από τα εικονίδια της εφαρμογής την επιλογή για αποστολή μηνύματος. Έπειτα, μπορεί να δει τα μηνύματα τα οποία έχει αποστείλει και τέλος να αποστείλει ένα νέο μήνυμα σε όλους τους γειτονικούς κόμβους που βρίσκονται στο σύνολο των TopK που του έχει επιστρέψει ο εξυπηρετητής μας.



Σχήμα 3.13 : Λογική σύνδεσης Proximity

Αυτό το σχεδιάγραμμα είναι οι καταστάσεις από τις οποίες περνά η σύνδεση η οποία συνδέεται με τον εξυπηρετητή μας, για να αποστείλει τα στοιχεία που απαρτίζουν τα μέλη του επερωτήματος. Έπειτα παραλαμβάνει τα αποτελέσματα και τα χρησιμοποιεί με τον τρόπο που έχει σκοπό η κάθε εφαρμογή. Από εκεί και πέρα, εάν χρειάζεται να επαναληφθεί η διαδικασία, γίνεται εκ νέου η σύνδεση.

3.3 Twitter Crawler

Στο υποκεφάλαιο αυτό θα περιγράψουμε την δουλειά που κάναμε σχετικά με την συλλογή δεδομένων από το κοινωνικό δίκτυο του Twitter. Όπως έχουμε αναφέρει, το Twitter προσφέρει ένα ευρύ φάσμα από κλήσεις προς το σύστημα του. Με τον τρόπο αυτό μπορούμε να έχουμε πρόσβαση στην πληθώρα πληροφοριών τα οποία διακινούνται κάθε μέρα από το κοινωνικό δίκτυο του Twitter.

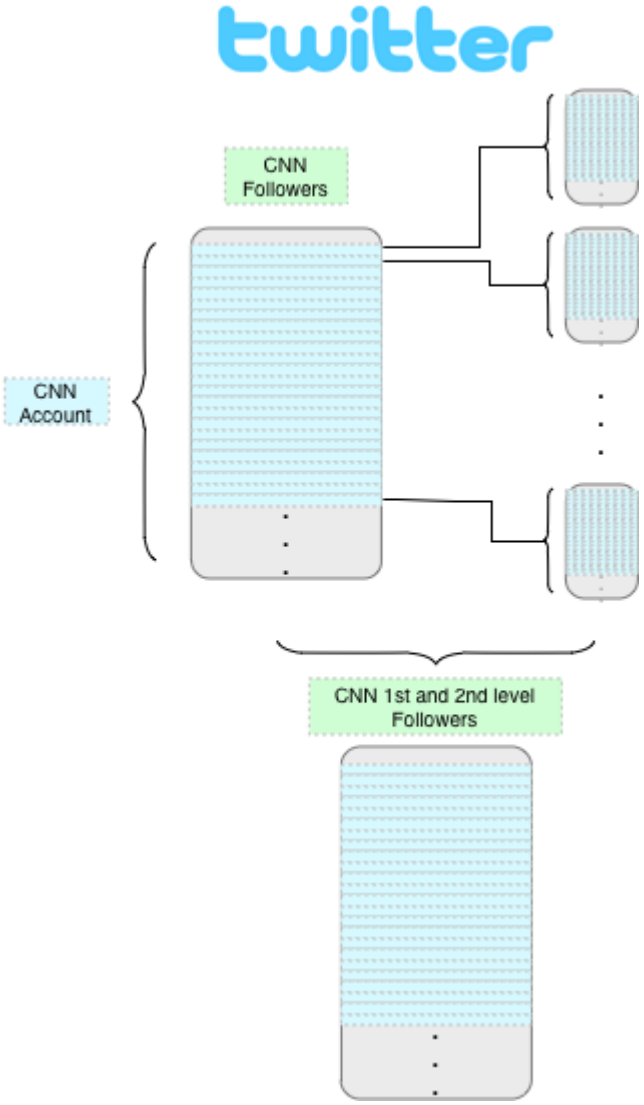
3.3.1 Γλώσσα προγραμματισμού

Η γλώσσα προγραμματισμού που χρησιμοποιήσαμε, ήταν η Java. Το API που προσφέρει επίσημα το Twitter, είναι βασικά με εντολές HTTP, οι οποίες αποστέλλονται υπό τη μορφή κλήσεων προς τους εξυπηρετητές ιστού του Twitter. Εμείς χρησιμοποιήσαμε την βιβλιοθήκη για Java, Twitter4J, η οποία προσφέρει κλάσεις και μεθόδους της γλώσσας Java, οι οποίες αναλαμβάνουν να εκτελέσουν τις κατάλληλες κλήσεις προς τους εξυπηρετητές ιστού του Twitter, να πάρουνε πίσω τα αποτελέσματα, να τα επεξεργαστούν από την μορφή που τα στέλνει δια μέσου του δικτύου το Twitter και τέλος να μας δώσουν την απάντηση στην πράξη που είχαμε ζητήσει με δεδομένα εμφωλευμένα σε τύπους της Java.

3.3.2 Υλοποίηση

Συγκεκριμένα, η υλοποίηση μας αυτό που προσπαθεί να κάνει, είναι χωρισμένο σε δύο μέρη. Αρχικά την εξασφάλιση κάποιας συλλογής από στοιχεία για ένα μέρος του κοινωνικού δικτύου του Twitter, δηλαδή μέρος του γράφου που δημιουργείται από τους χρήστες του. Έπειτα είναι η εξασφάλιση κάποιων από τα μηνύματα που αυτοί οι χρήστες ανταλλάζουν μεταξύ τους, μέσω των αναρτήσεων που κάνει ο καθένας. Έπειτα βάση της πληροφορίας από την προηγούμενη συλλογή δεδομένων με τον γράφο των ακολούθων των χρηστών, μπορούμε να γνωρίζουμε σε ποιους ήταν και σε ποιους όχι ορατή μία ανάρτηση ενός συγκεκριμένου μηνύματος από ένα συγκεκριμένο χρήστη.

3.3.2.1 Εξασφάλιση Συλλογής Ακόλουθων

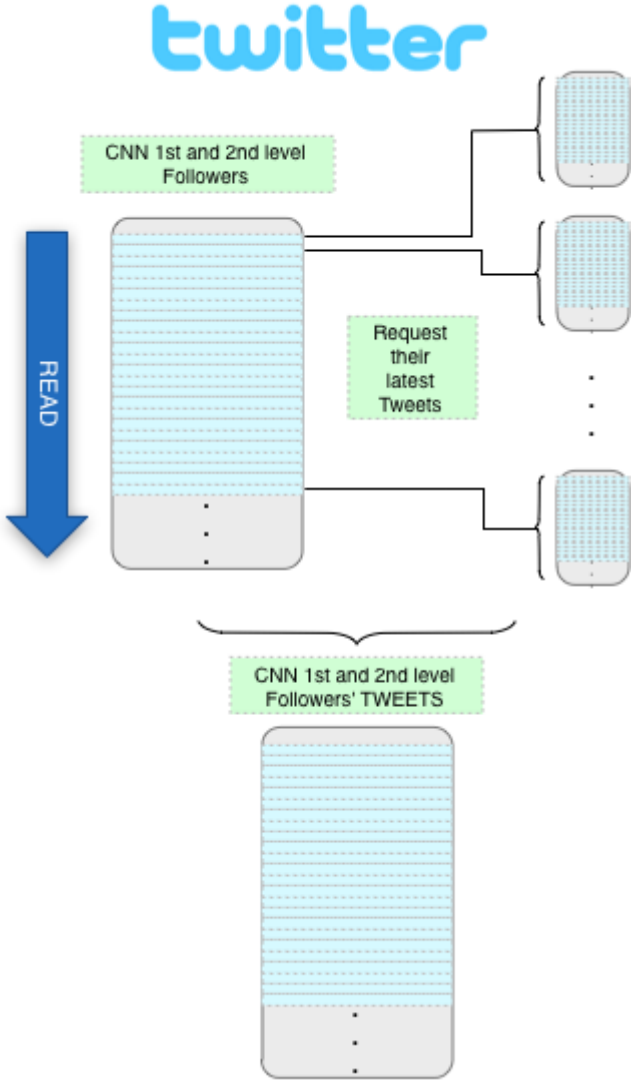


Σχήμα 3.14 : Αλγόριθμος Ανάκτησης Ακολούθων Twitter

Στο πιο πάνω σχεδιάγραμμα, βλέπουμε τον τρόπο με τον οποίο εξασφαλίσαμε την ανάκτηση συλλογής των δεδομένων των χρηστών του Twitter και των ακολούθων τους. Συγκεκριμένα, αρχίσαμε την αναζήτηση των χρηστών από κάποιο χρήστη ο οποίος είχε μερικά εκατομμύρια από ακόλουθους. Αυτός ο χρήστης ήταν ο επίσημος λογαριασμός Twitter που διατηρεί το γνωστό κανάλι ειδήσεων, CNN [41].

Όταν συλλέξαμε το πρώτο επίπεδο από ακόλουθους του CNN, πήραμε τον κάθε χρήστη από αυτούς και πήραμε τους δικούς τους ακόλουθους, έτσι έχουμε τα πρώτα δύο επίπεδα από ακολούθους του CNN. Στη συνέχεια πήραμε τους μοναδικούς από όλους αυτούς, έτσι έχουμε την λίστα με τους χρήστες μας και τις σχέσεις μεταξύ τους.

3.3.2.2 Εξασφάλιση συλλογής μηνυμάτων από χρήστες



Σχήμα 3.14 : Αλγόριθμος Ανάκτησης Μηνυμάτων Twitter

Στη συνέχεια πήραμε τους μοναδικούς από όλους τους ακόλουθους του CNN σε βάθος δύο και αφού διαπεράσαμε την λίστα με τους χρήστες, ζητήσαμε από το Twitter να μας επιστρέψει τα τελευταία μηνύματα τα οποία έχουν αναρτήσει οι χρήστες αυτοί στον λογαριασμό τους στο Twitter. Αυτό είχε σαν αποτέλεσμα μία μεγάλη συλλογή από στοιχεία μηνυμάτων τα οποία αναρτώνται από τους χρήστες τους οποίους συλλέξαμε στο προηγούμενο βήμα.

Κεφάλαιο 4

Πειραματική Μελέτη

4.1 Πειραματική Μεθοδολογία

Στο υποκεφάλαιο αυτό, θα αναλύσουμε την πειραματική μεθοδολογία που ακολουθήσαμε με σκοπό να αποτιμήσουμε την λειτουργία του εξυπηρετητή μας. Σκοπός της διπλωματικής αυτής εργασίας ήταν η δημιουργία ενός εξυπηρετητή που να προσφέρει την υπηρεσία της γειτνίασης σε κινητές συσκευές Android. Ήταν χωρισμένη δηλαδή σε τρείς φάσεις:

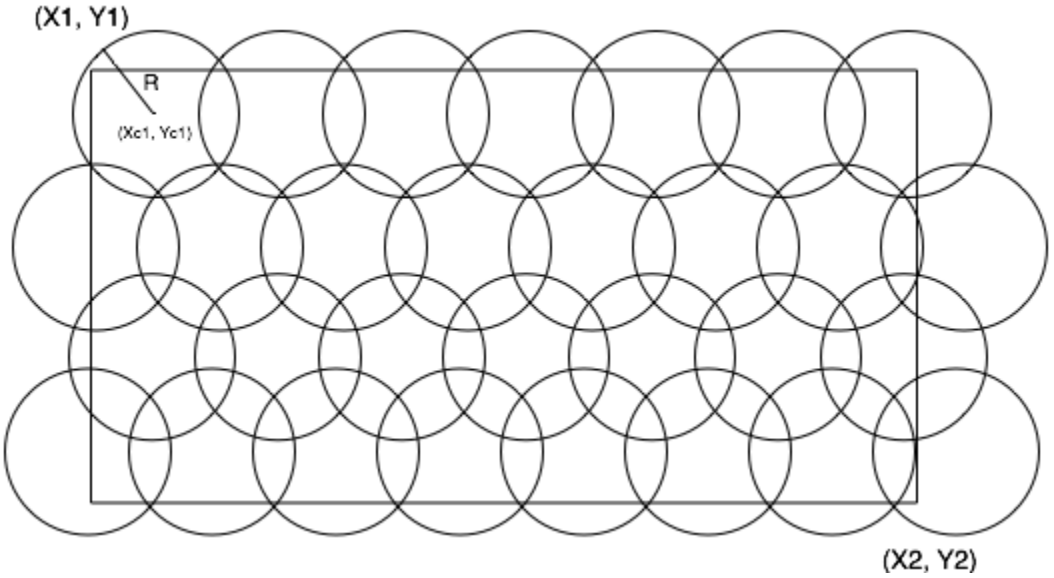
(α) Υλοποίηση εξυπηρετητή
(β) Υλοποίηση αλγορίθμων γειτνίασης
(γ) Υλοποίηση εφαρμογής σε κινητή συσκευή Android

4.1.1 Συλλογές δεδομένων

Η βασική συλλογή δεδομένων, στην οποία στηριχτήκαμε για τα πειράματα μας όσο αφορά τον εξυπηρετητή, αλλά και τα πειράματα που έχουν σχέση με τους αλγορίθμους γειτνίασης, ήταν τα δεδομένα από το Oldenburg [42], το οποίο αποτελεί ένα εργαλείο για παραγωγή συνθετικών δεδομένων για κινητά αντικείμενα, κατά την παραγωγή των οποίων λαμβάνονται υπόψη τα κινητά δίκτυα στα οποία συμμετάσχουν.

Χρησιμοποιώντας αυτά τα δεδομένα, έγινε εφικτή η παραγωγή των αναφορών που χρειάζεται να αποστέλλουν οι πελάτες προς τον εξυπηρετητή, αλλά επίσης κατασκευάστηκαν και τα δεδομένα που χρειάζεται ο αλγόριθμος PNA για να μπορεί να κάνει τον υπολογισμό με πιο αποδοτικό τρόπο από τον BNA.

Ο τρόπος που χρησιμοποιήθηκαν τα δεδομένα από τη συλλογή δεδομένων του Oldenburg, ήταν να υπολογιστεί αρχικά η έκταση που καταλαμβάνουν όλες οι αναφορές χρηστών από τις τροχιές οι οποίες απαρτίζουν τη συλλογή δεδομένων και στη συνέχεια να καλυφθεί όλη η περιοχή αυτή με κυψέλες κινητής τηλεφωνίας.



Σχήμα 4.1 : Αλγόριθμος υπολογισμού των συλλογών δεδομένων

Στο πιο πάνω σχήμα, φαίνεται παραστατικά η χρήση κυψελών κινητής τηλεφωνίας, με κέντρο (Xci, Yci) και ακτίνα R, οι οποίες καλύπτουν την γεωγραφική περιοχή από το (X1, Y1) μέχρι το (X2, Y2). Με αυτό τον τρόπο στη συνέχεια, αφού γνωρίζουμε την τοποθέτηση των κυψελών, λαμβάνοντας υπόψη και την γεωγραφική θέση της κάθε αναφοράς χρήστη, μπορέσαμε να προσθέσουμε την πληροφορία του Cell ID στα δεδομένα των αναφορών των χρηστών, τόσο για την κυψέλη στην οποία είναι εγγεγραμμένοι, όσο και για τις γειτονικές κυψέλες, τις από τις οποίες μπορεί και λαμβάνει σήμα μία συσκευή, άρα εμπίπτει στην ραδιοκάλυψη των.

Επίσης, η πληροφορία που μόλις αναφέραμε, δηλαδή το κέντρο (Xci, Yci) και η ακτίνα R για την κάθε κυψέλη που έχει κατασκευαστεί στο προηγούμενο βήμα, μας χρησιμεύει για την προμήθεια του εξυπηρετητή μας των δεδομένων αυτών, για την χρήση τους από τον PNA αλγόριθμο, ο οποίος βασίζεται στην γνώση των κυψελών κινητής τηλεφωνίας για να μπορεί να λειτουργήσει.

Τέλος, μια άλλη συλλογή δεδομένων, η οποία έγινε για τους σκοπούς της εφαρμογής επίδειξης σε κινητή συσκευή Android, η οποία έγινε στα πλαίσια των δραστηριοτήτων του τμήματος πληροφορικής, έχει συλλεχτεί. Συγκεκριμένα, για την συλλογή των πληροφοριών αυτών, υλοποιήθηκε μια εφαρμογή στην ίδια την κινητή συσκευή Android, η οποία σκοπός της ήταν η καταγραφή κάποιων κυψελών κινητής τηλεφωνίας και τοποθεσιών, κατά την διάρκεια κίνησης με όχημα, εντός και εκτός των πόλεων της Λευκωσίας κυρίως, αλλά και της Λάρνακας. Η συλλογή δεδομένων που φτιάχτηκε τελικά, δεν ήταν τόσο μεγάλης εκτάσεως, αλλά σημαντικό ήταν ότι τα δεδομένα αυτά ήταν πραγματικά.

4.1.2 Υλοποίηση Εξυπηρετητή

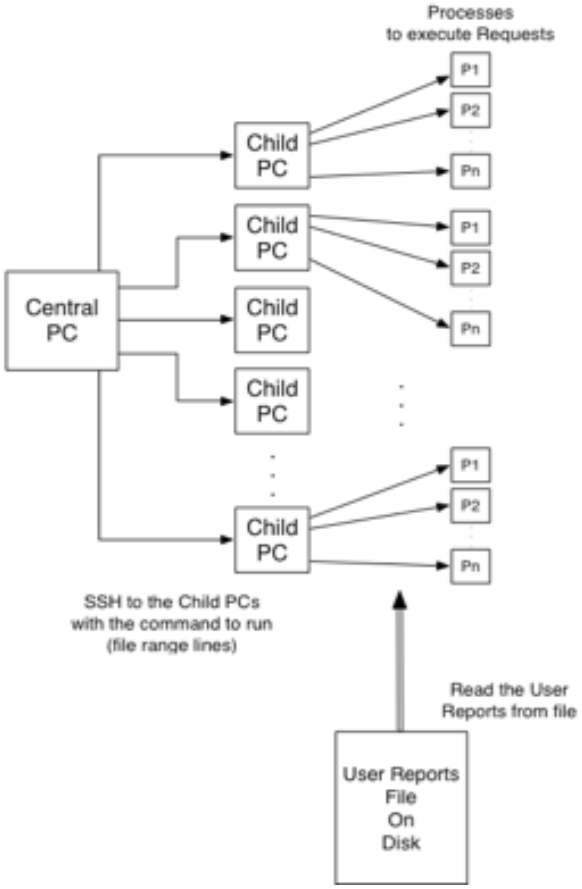
Η περιγραφή που έγινε στο προηγούμενο κεφάλαιο, που αφορά τον εξυπηρετητή, είναι αυτή που υλοποιήθηκε για την λειτουργία του εξυπηρετητή μας, με σκοπό την εξυπηρέτηση μεγάλου όγκου από πελάτες, από ένα και μόνο εξυπηρετητή. Επίσης έγινε και χρήση κάποιων συλλογών δεδομένων, οι οποίες έχουν αναλυθεί στο προηγούμενο υποκεφάλαιο.

Συγκεκριμένα, για την αποτίμηση του εξυπηρετητή μας, την σύγκριση δηλαδή με των δύο τεχνικών αποδοχής των αιτήσεων, δηλαδή τη μία με τη χρήση της συνάρτησης select( ) και έπειτα της συνάρτησης accept( ), αλλά και την άλλη με την χρήση της συνάρτησης accept( ) μόνο, χρησιμοποιήσαμε την ίδια υλοποίηση του εξυπηρετητή, τρέχοντας όμως δύο διαφορετικά εκτελέσιμα, τα οποία υλοποιούσαν αντίστοιχα την κάθε τεχνική που αναφέραμε. Αξίζει να αναφερθεί ότι κατά τη διάρκεια των πειραμάτων αυτών δεν έγινε χρήση του αλγορίθμου γειτνίασης. Δηλαδή, η λειτουργία που περιλάμβανε ο εξυπηρετητής, σε συνδυασμό με τους πελάτες, ήταν η σύνδεση που

γινόταν ανάμεσα στις δύο πλευρές, πελάτη και εξυπηρετητή, έπειτα η αναφορά των στοιχείων που αφορούν τη συσκευή, εκ μέρους του πελάτη και τέλος η αποσύνδεση του πελάτη από τον εξυπηρετητή, χωρίς να γίνεται ο υπολογισμός των K κοντινότερων συσκευών και η αποστολή των δεδομένων αυτών από τον εξυπηρετητή προς τους πελάτες.

Η υλοποίηση προγραμμάτων πελατών που κάναμε για τη συγκεκριμένη σειρά πειραμάτων, ήταν κάποια Bash Scripts τα οποία τρέχουν παράλληλα σε πολλές μηχανές, οι οποίες βρίσκονται στα εργαστήρια του τμήματος. Χρησιμοποιήθηκε η ιδέα του Public-Private key και σε συνδυασμό με την χρήση της εντολής ssh, έγινε εφικτή η παράλληλη εκτέλεση πολλών χιλιάδων πελατών, οι οποίοι θα εκτελούν ταυτόχρονα εκκλήσεις σύνδεσης προς τον εξυπηρετητή μας. Αξίζει να σημειωθεί ότι γι' αυτό το λόγο, χρειάστηκε να αυξήσουμε τα όρια στα ανοικτά αρχεία που έχει ο εξυπηρετητής μας, για να μπορεί να υποστηρίζει τόσες πολλές ταυτόχρονες συνδέσεις.

Όπως αναφέρθηκε, κάναμε χρήση πελατών, οι οποίοι ήταν στην ουσία ένα Bash Script σε λειτουργικό σύστημα Unix. Η λογική του φαίνεται στο επόμενο σχεδιάγραμμα.



Σχήμα 4.2 : Αλγόριθμος εκτέλεσης πελατών για τα πειράματα

Συγκεκριμένα, στο σχήμα φαίνεται ο τρόπος με τον οποίο η διεργασία από τον κεντρικό υπολογιστή, από τον οποίο τρέχουμε τα πειράματα μας, ελέγχει την εκτέλεση των υπολοίπων πελατών. Όπως αναφέραμε και προηγουμένως, ο τρόπος που εκτελεί εντολές, δηλαδή άλλα Bash Script στις απομακρυσμένες μηχανές, είναι με την χρήση της ssh εντολής, η οποία προσφέρει αυτή την δυνατότητα. Με αυτό τον τρόπο, δεδομένου ενός αρχείου με τις αναφορές, οι οποίες κατασκευάστηκαν με τον τρόπο που έχει αναφερθεί στο προηγούμενο υποκεφάλαιο, από τα δεδομένα που παράχθηκαν με την χρήση του Oldenburg, κάθε απομακρυσμένη μηχανή, μπορεί να αποστείλει συγκεκριμένες γραμμές που της υποδεικνύει η διεργασία από τον κεντρικό υπολογιστή. Στη συνέχεια, στον κάθε απομακρυσμένο υπολογιστή, δημιουργείται μια διεργασία ανά γραμμή που πρέπει να αποσταλεί στον εξυπηρετητή, με αποτέλεσμα οι συνδέσεις που εκτελούνται από τους πελάτες να είναι παράλληλες.

```
#!/bin/bash

for i in `cat hosts.txt`
do
    eval "ssh $i time ~/run/execConnections.sh ~/run/userData/UsersReports2000 1 10 | grep real: &"
done
```

```
for myLine in `cat $1`
do
    (( counter= $counter + 1 ))

    if [[ $counter -lt $min  ]]; then
        continue
    fi

    if [[ $counter -gt $max ]]; then
        break
    fi
    eval "~/run/execClient.sh $SERVER_IP $SERVER_PORT $myLine &"
done
```

Σχήμα 4.3 : Κομμάτι κώδικα από εκτέλεσης πελατών για τα πειράματα

Στο πιο πάνω κομμάτι κώδικα, βλέπουμε αυτό που περιγράψαμε στις προηγούμενες παραγράφους. Στο πρώτο μέλος βλέπουμε την εκτέλεση script που έχουμε στον κεντρικό υπολογιστή, ο οποίος τρέχει με την εντολή ssh το script που βλέπουμε στο κάτω μέρος, στους απομακρυσμένους υπολογιστές. Στο κάτω μέρος βλέπουμε το πώς ο απομακρυσμένος υπολογιστής διαβάζει το αρχείο και για κάθε γραμμή του, δημιουργεί μία νέα διεργασία η οποία είναι υπεύθυνη για να την αποστείλει στον εξυπηρετητή και έπειτα να πάρει τις απαντήσεις αντίστοιχα.

4.1.3 Υλοποίηση Αλγορίθμων Γειτνίασης

Σε αυτό το υποκεφάλαιο θα εξηγήσουμε τι δουλειά έγινε όσο αφορά την υλοποίηση των αλγορίθμων γειτνίασης, δηλαδή του αλγορίθμου BNA και του αλγορίθμου PNA.

4.1.3.1 Υλοποίηση Αλγορίθμου BNA

Ο αλγόριθμος BNA, ο οποίος έχει αναπτυχθεί και αναλυθεί προηγουμένως, σε προηγούμενο κεφάλαιο, έχει υλοποιηθεί για τους σκοπούς προσφοράς της υπηρεσίας της γειτνίασης, στον εξυπηρετητή μας.

Συγκεκριμένα, ο αλγόριθμος αυτός έχει υλοποιηθεί και έχει προστεθεί στο κομμάτι του εξυπηρετητή όπου γίνεται ο υπολογισμός των γειτονικών συσκευών, δηλαδή εκεί που βάσει κάποιων παραγόντων και στη συγκεκριμένη περίπτωση, μετά το πέρας μίας χρονικής περιόδου, έχουμε την ενεργοποίηση της λειτουργίας του αλγορίθμου. Στα πειράματα που θα παρουσιάσουμε τα αποτελέσματα σε επόμενο υποκεφάλαιο, αυτή η χρονική περίοδος ήταν 30 δευτερόλεπτα. Δηλαδή ο αλγόριθμος εκτελείτο κάθε μισό λεπτό, κάνοντας χρήση ενός σήματος από τον πυρήνα του λειτουργικού και ταυτόχρονα σταματούσε τον εξυπηρετητή από την λήψη περεταίρω αιτήσεων από πελάτες.

4.1.3.2 Υλοποίηση Αλγορίθμου PNA

Ο αλγόριθμος PNA, τον οποίο έχουμε αναλύσει και ο οποίος έχει δοκιμαστεί έναντι άλλων αντίστοιχων του, δεν έχει υλοποιηθεί στα πλαίσια αυτής της διπλωματικής εργασίας. Ο λόγος που έγινε αυτό, ήταν η φύση της συγκεκριμένης διπλωματικής εργασίας, της οποίας ο σκοπός δεν ήταν η σύγκριση αλγορίθμων εξεύρεσης κοντινών γειτονικών συσκευών, αλλά η χρησιμοποίηση τέτοιων αλγορίθμων για την παροχή της συγκεκριμένης υπηρεσίας σε πραγματικές κινητές συσκευές. Αξίζει δεν να αναφερθεί ότι έχει γίνει προσπάθεια μεταφοράς του αλγορίθμου από τη γλώσσα Java στην οποία είχε αρχικά αναπτυχθεί, στην γλώσσα που έχει αναπτυχθεί ο εξυπηρετητής μας, χωρίς όμως τελικά να γίνει κατορθωτή η εξ ολοκλήρου υλοποίηση του. Επομένως, η υλοποίηση και αξιολόγηση του συγκεκριμένου αλγορίθμου, αλλά και η σύγκριση του με τον BNA αλγόριθμο, επαφίεται ως μελλοντική εργασία.

4.1.4 Υλοποίηση εφαρμογής σε κινητή συσκευή Android

Όπως έχει αναφερθεί και στο υποκεφάλαιο του κεφαλαίου αυτού, το οποίο αφορά τις συλλογές δεδομένων οι οποίες έχουν χρησιμοποιηθεί, είχαμε αναπτύξει μία εφαρμογή για σκοπούς επίδειξης στα πλαίσια δραστηριοτήτων του τμήματος πληροφορικής. Για τους σκοπούς της επίδειξης αυτής, είχε γίνει υλοποίηση μίας εφαρμογής για την απόδειξη της σημαντικότητας της χρήσης της υπηρεσίας της γειτνίασης. Συγκεκριμένα η εφαρμογή αυτή περιλάμβανε την αποστολή των στοιχείων της συσκευής προς τον εξυπηρετητή, δηλαδή στοιχεία που αφορούσαν το γεωγραφικό μήκος και γεωγραφικό

πλάτος της συσκευής, την κυψέλη κινητής τηλεφωνίας με την οποία είναι συνδεδεμένη η συσκευή μας, καθώς επίσης και πληροφορίες για τις γειτονικές κυψέλες, από τις οποίες λαμβάνει σήμα η συσκευή μας.

Στη συνέχεια, στον εξυπηρετητή γίνεται η καταγραφή των στοιχείων των πελατών σε μία δομή. Στην επόμενη φάση, οι πελάτες μπαίνουν σε μια κατάσταση αναμονής. Ο λόγος είναι για την αναμονή του εξυπηρετητή για να εφαρμόσει τον αλγόριθμο της γειτνίασης και να τους επιστρέψει τα αποτελέσματα των κοντινότερων K γειτονικών κόμβων που αντιστοιχούν στον κάθε ένα. Στην πραγματικότητα, στα πλαίσια της υλοποίησης του πελάτη στο κινητό τηλέφωνο Android, οι γειτονικοί κόμβοι που επιστρέφονταν ήταν τυχαίοι από το σύνολο των εγγραφών τις οποίες είχαμε συλλέξει με το ίδιο το κινητό τηλέφωνο, τις οποίες ο εξυπηρετητής απόστελλε στα τηλέφωνα σαν αποτελέσματα του αλγορίθμου γειτνίασης, ο οποίος στην ουσία δεν έτρεχε.

Όταν ο κάθε πελάτης παραλάμβανε τα στοιχεία που αφορούσαν την κάθε γειτονική συσκευή, τότε τα παρουσίαζε στον χάρτη της εφαρμογής, για να μπορεί να δει διαισθητικά ο χρήστης σε ποια τοποθεσία είναι οι γειτονικοί του κόμβοι. Ο εξυπηρετητής σε αυτή την περίπτωση αποστέλλει τα δεδομένα που αφορούν την κυψέλη με την οποία είναι συνδεδεμένος ο γειτονικός κόμβος, με αποτέλεσμα να μην προδίδεται η πραγματική θέση του γειτονικού κόμβου. Τότε, ο πελάτης, εκτελεί κλήσεις προς τον εξυπηρετητή της Google, ο οποίος δεδομένου των στοιχείων της κυψέλης, μπορεί να επιστρέψει τις πληροφορίες που αφορούν την τοποθεσία της. Επομένως, αυτό που βλέπει ο χρήστης να του παρουσιάζεται, είναι στην ουσία μια τοποθεσία η οποία εμπίπτει στην ραδιοκάλυψη της κυψέλης στην οποία είναι συνδεδεμένη η γειτονική κινητή συσκευή. Τέλος, παρουσιάστηκε και η δυνατότητα της αποστολής μηνυμάτων στις K κοντινότερες συσκευές εκ μέρους του πελάτη που ζήτησε την υπηρεσία.

4.2 Πειραματικά Αποτελέσματα

Σε αυτό το υποκεφάλαιο, θα παρουσιάσουμε τα αποτελέσματα που είχαμε από τα πειράματα που έγιναν.

Συγκεκριμένα, θα παρουσιάσουμε τα εξής:

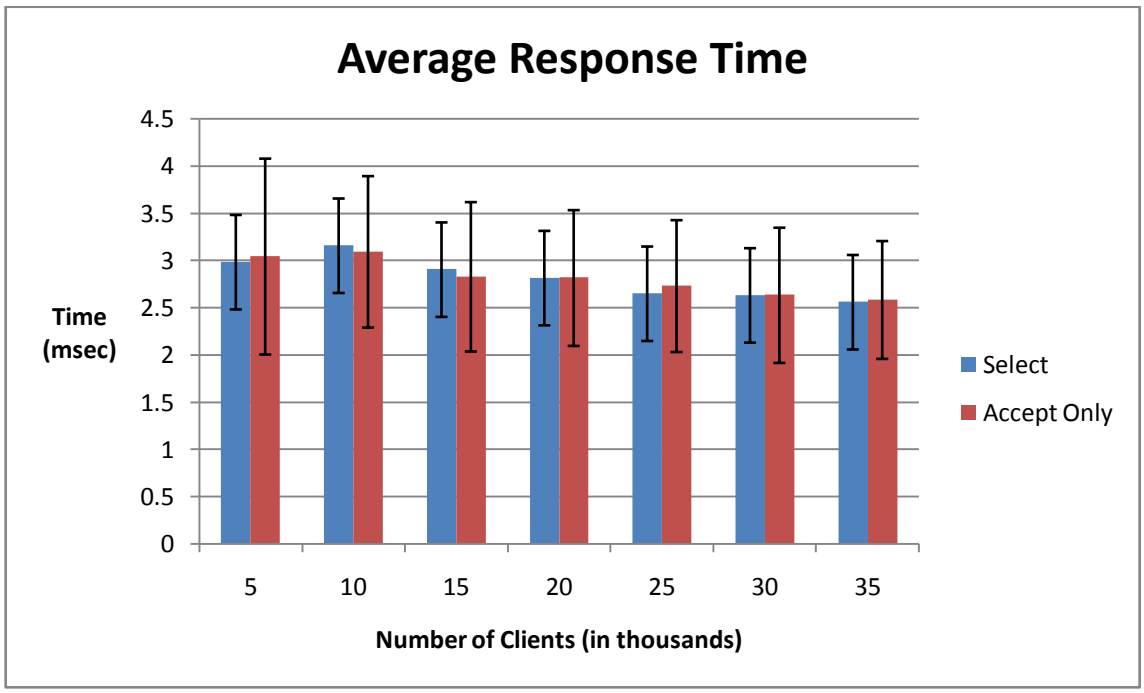
α. Τον χρόνο απόκρισης κατά την αποστολή των στοιχείων των πελατών στον εξυπηρετητή για την καταγραφή τους μόνο.

β. Τον χρόνο που χρειάζεται ο εξυπηρετητής για να τρέξει τον αλγόριθμο BNA, μόλις περάσει ο χρόνο των 30 δευτερολέπτων από την προηγούμενη φορά που είχε τρέξει ο αλγόριθμος.

γ. Παρουσίαση μερικών στιγμιότυπων οθόνης τις οποίες πήραμε από την εφαρμογή επίδειξης, από την κινητή συσκευή Android.

4.2.1 Χρόνος Απόκρισης Εξυπηρετητή

Εδώ θα παραθέσουμε την σχετική γραφική παράσταση η οποία παρουσιάζει τα δεδομένα, δηλαδή τον χρόνο απόκρισης του εξυπηρετητή σε αντιστοιχία με τον αριθμό των χρηστών.



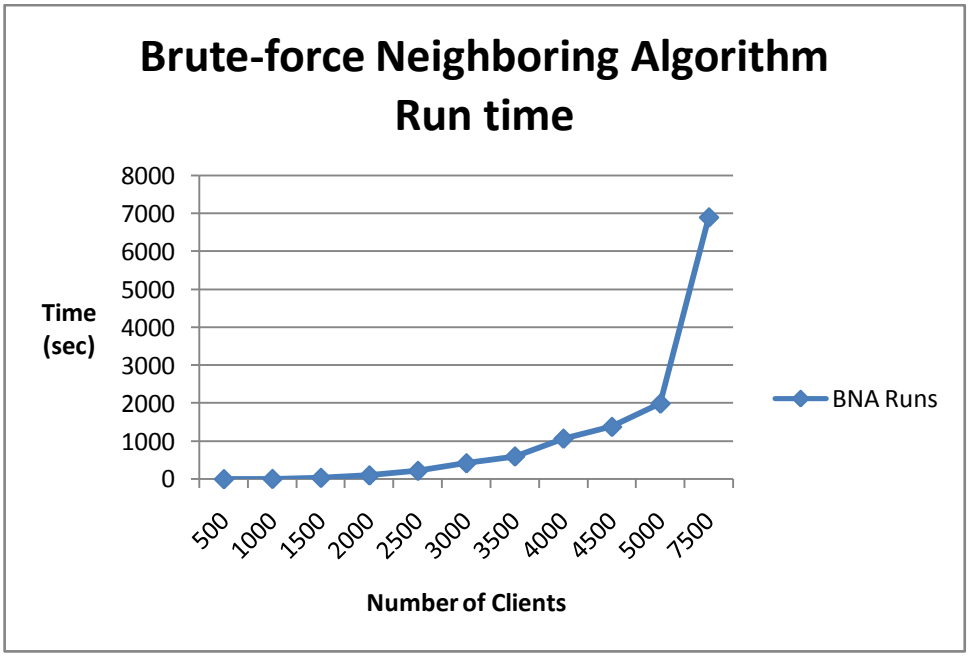
Σχήμα 4.4 : Μέσος όρος χρόνου απόκρισης του χρήστη ανά αριθμό πελατών

Όπως βλέπουμε στην γραφική, τα αποτελέσματα που συγκρίνουν την χρήση της συνάρτησης select( ) στον εξυπηρετητή μας, σε σχέση με την χρήστη της συνάρτησης accept( ) μόνο, είναι αρκετά πανομοιότυπα. Θα μπορούσε κάποιος να πει ότι δεν αξίζει να γίνεται χρήση της συνάρτησης select( ), αλλά μόνο η χρήση της accept( ). Όμως, εκτός από το γεγονός ότι όσο περισσότερους χρήστες έχουμε, τόσο καλύτερα αποτελέσματα έχουμε για την select, παρατηρούμε επίσης την τυπική απόκλιση των δύο μετρήσεων, να είναι πιο μικρή στην περίπτωση του select. Αυτό σημαίνει ότι όσες φορές και να τρέξουμε το πείραμα μας, το αποτέλεσμα που θα έχουμε θα είναι ένα περίπου το ίδιο. Στην αντίθετη περίπτωση, η χρήση μόνο της συνάρτησης accept, μας δίνει τυπική απόκλιση κατ' επανάληψη μεγαλύτερες μετρήσεις σε σχέση με τον συνδυασμό με την συνάρτηση select. Επομένως, μπορούμε να παρατηρήσουμε μία πιο σταθερή και επαναλαμβανόμενη συμπεριφορά κατά την χρήση του συνδυασμού των συναρτήσεων select+accept παρά μόνο με την χρήση της συνάρτησης accept μόνο, κατά την διάρκεια των αποδοχών των αιτήσεων που αποστέλλονται από τους πελάτες προς τον εξυπηρετητή.

Ο τρόπος που έγινε η συλλογή, ήταν με την εκτέλεση της εντολής γραμμής κελύφους time, η οποία επιστρέφει τρείς χρόνους που αφορούν την εκτέλεση μιας διεργασίας. Από αυτούς τους χρόνους, εμείς πήραμε τον πραγματικό για κάθε πελάτη που εκτελέσαμε, ο οποίος απόστειλε αίτημα προς τον εξυπηρετητή και στο τέλος βρήκαμε τον μέσο όρο των μετρήσεων αυτών. Στη συνέχεια, βάλαμε όλους αυτούς τους μέσους όρους στην γραφική παράσταση, που προηγήθηκε, και για τις δύο τεχνικές που αναπτύξαμε με σκοπό να τις συγκρίνουμε.

4.2.2 Χρόνος Εκτέλεσης Αλγορίθμου BNA

Όπως προαναφέραμε, έχει γίνει η υλοποίηση του αλγορίθμου BNA, ο οποίος είναι ο άπληστος αλγόριθμος σε σχέση με τους δυο που έχουμε αναφέρει στα πλαίσια της διπλωματικής αυτής εργασίας. Το όνομα του, καθώς και τα αναμενόμενα αποτελέσματα φαίνεται να επαληθεύονται κατά την πειραματική μελέτη του αλγορίθμου αυτού. Τα αποτελέσματα του παρουσιάζονται στην παρακάτω γραφική παράσταση:



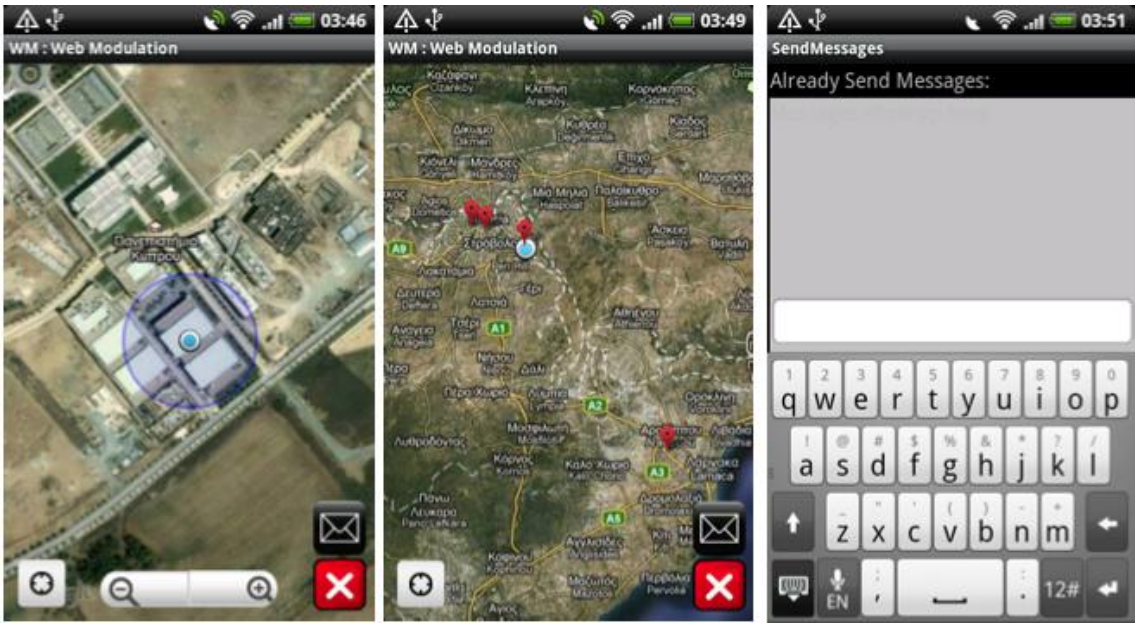
Σχήμα 4.5 : Χρόνος εκτέλεσης του αλγορίθμου BNA σε σχέση με τον αριθμό χρηστών

Όπως αναφέραμε, τα πειραματικά αποτελέσματα φαίνεται να επαληθεύουν τους κακούς/απαγορευτικούς χρόνους εκτέλεσης που περιμέναμε από την πειραματική αποτίμηση του αλγορίθμου BNA. Φαίνεται επίσης και η εκθετική αύξηση που έχουμε στον χρόνο εκτέλεσης του αλγορίθμου, σε σχέση με τον αριθμό των χρηστών, οι οποίοι έχουν αποστείλει την τοποθεσία τους προς τον εξυπηρετητή.

Τα αποτελέσματα που είχαμε από τον αλγόριθμο BNA είναι μία ακόμη ένδειξη ότι για να έχουμε καλύτερη απόκριση από την υπηρεσία μας προς τους χρήστες της, επιβάλλεται να υλοποιήσουμε ένα άλλο αλγόριθμο, ο οποίος θα μπορεί να βρει το σωστό αποτέλεσμα σε πολύ μικρότερο χρόνο από αυτό που χρειάζεται ο BNA για να μπορέσει να βρει τις K κοντινότερες κινητές συσκευές. Θα μπορούσαμε να βελτιώσουμε την χρήση του BNA, κάνοντας πιθανώς κάποιες αλλαγές, οι οποίες θα γλίτωναν κάποιες συγκρίσεις ή κάποιες εισαγωγές στις δομές μας, αλλά επιφέροντας και πάλι το ίδιο αποτέλεσμα με τον αλγόριθμο μας. Όμως, η πιο προφανής λύση θα ήταν η χρήση του PNA αλγορίθμου, ο οποίος έχει ήδη συγκριθεί από τους δημιουργούς του με άλλους αλγορίθμους και φαίνεται να έχει πολύ καλύτερα αποτελέσματα σε χρόνους εκτέλεσης παρά ένας αλγόριθμος του τύπου του BNA.

4.2.3 Παρουσίαση στιγμιότυπων Οθόνης της εφαρμογής σε Android

Σε αυτό το υποκεφάλαιο θα παρουσιάσουμε απλά μερικά στιγμιότυπα οθόνης τα οποία έχουμε συλλέξει από την παρουσίαση της εφαρμογής που έχουμε υλοποιήσει για σκοπούς επίδειξης και εκτίμησης των δυνατοτήτων ενός αλγορίθμου εύρεσης γειτνιαζόντων κινητών συσκευών. Η εφαρμογή αυτή έχει περιγραφεί εκτενώς σε προηγούμενα κεφάλαια, αλλά και υποκεφάλαια, ως εκ τούτου δεν θα περιγραφεί περεταίρω σε αυτό το υποκεφάλαιο, παρά μόνο μερική περιγραφή για τα στιγμιότυπα οθόνης που θα περιγραφούν.



Σχήμα 4.6 : Στιγμιότυπα οθόνης από την εφαρμογή σε κινητή συσκευή Android

Στην πρώτη οθόνη βλέπουμε την ένδειξη η οποία αφορά την τοποθεσία την δική μας, έπειτα βλέπουμε στην δεύτερη οθόνη την παρουσίαση των K κοντινότερων συσκευών προς την δική μας, όπου συγκεκριμένα στο στιγμιότυπο το K=4 και τέλος έχουμε την οθόνη η οποία παρουσιάζει την επιλογή για αποστολή μηνύματος κειμένου στις γειτονικές κινητές συσκευές, οι οποίες έχουν ανευρεθεί από την σύνδεση της κινητής συσκευής με τον εξυπηρετητή.

Κεφάλαιο 5

Συμπεράσματα και μελλοντική δουλειά

5.1 Συμπεράσματα

Σε αυτό το υποκεφάλαιο θα συνοψίσουμε τα αποτελέσματα που είχαμε τόσο κατά την εκτέλεση των πειραμάτων στον εξυπηρετητή, όσο και από την χρήση της κινητής συσκευής Android.

Όπως έχουμε αναφέρει στην εισαγωγή, η ανάπτυξη των κινητών συσκευών τα τελευταία τρία με τέσσερα χρόνια, υπήρξε ραγδαία. Αυτό ώθησε πολλούς προγραμματιστές, εταιρίες παραγωγής λογισμικού, αλλά και υλικού, καθώς επίσης και ερευνητές σε ακαδημαϊκό επίπεδο, με την σοβαρή ενασχόληση με τέτοιου είδους κινητές συσκευές. Επίσης, οι τάσεις των καιρών, δείχνουν ότι η εξέλιξη σε τέτοιου είδους συσκευές, δεν πρόκειται να εκλείψει, αν όχι καθόλου, δεν πρόκειται να εκλείψει σύντομα.

Εκτός από αυτή την τάση που επικρατεί, έχουμε τις συσκευές αυτές να περιλαμβάνουν δεκάδες αισθητήρες οι οποίοι μπορούν να αντιλαμβάνονται το περιβάλλον στο οποίο βρίσκεται η κινητή συσκευή. Ίσως ο σημαντικότερος και ακριβότερος σε θέματα ενέργειας αισθητήρας, είναι αυτός του GPS. Ο αισθητήρας αυτός, με ακρίβεια δεκάδων μέτρων, μπορεί να δώσει την θέση της συσκευής μας. Η πληροφορία αυτή της θέσης της συσκευής, μπορεί να χρησιμοποιηθεί για αμέτρητους σκοπούς. Οι πιο κοινοί, είναι η χρήση του για την προβολή της θέσης μας πάνω στον χάρτη της συσκευής ή η χρήση της για σκοπούς πλοήγησης, από τέτοιου είδους εφαρμογές.

Επίσης, από τον καιρό που είχαμε την ανάπτυξη των κινητών αυτών συσκευών, είχαμε και την παράλληλη έξαρση στην χρήση της προσφοράς υπηρεσιών βάσει γεωγραφικής τοποθεσίας της κινητής συσκευής. Ένα πολύ διαδεδομένο παράδειγμα, είναι η χρήση της τοποθεσίας των χρηστών για διαφήμιση, όπως πολύ επιτυχημένα κάνει η Google με το Google Ad. Αυτή η τάση ώθησε και εμάς στα πλαίσια αυτής της διπλωματικής

εργασίας να αναπτύξουμε μία υπηρεσία η οποία να προσφέρει αυτού του είδους υπηρεσία, η οποία να αφορά τις κοντινότερες κινητές συσκευές προς την θέση μας.

Υλοποιήσαμε ένα εξυπηρετητή, ο οποίος φτιάχτηκε με σκοπό να μπορεί να εξυπηρετεί μεγάλο όγκο πελατών. Συγκεκριμένα ο εξυπηρετητής αυτός έκανε χρήση της συνάρτησης select για σκοπούς καλύτερης επίδοσης. Κατά την υλοποίηση, είχε γίνει σύγκριση των δύο προσεγγίσεων, με τα αποτελέσματα να δικαιώνουν εν μέρει την επιλογή μας αυτή να κάνουμε χρήση της συνάρτησης αυτής. Το αποτέλεσμα ήταν ο εξυπηρετητής αυτός να μπορεί να εξυπηρετεί πάρα πολύ μεγάλο αριθμό χρηστών, χρησιμοποιώντας αποδοτικά τους πόρους του συστήματος.

Το επόμενο πειραματικό μας αποτέλεσμα, ήταν η καταμέτρηση του χρόνου ο οποίος χρειαζόταν για να τρέξει ο απλός αλγόριθμος BNA ο οποίος και είχε υλοποιηθεί για σκοπό του υπολογισμού των κοντινότερων K κινητών συσκευών για κάθε χρήστη. Τα αποτελέσματα που πήραμε ήταν απαγορευτικά και μας καταδεικνύουν την ανάγκη για την υλοποίηση ενός αλγορίθμου, ο οποίος να είναι πιο αποδοτικός σε σχέση με τον BNA. Αυτός ο αλγόριθμος είναι ο PNA, ο οποίος έχει περιγραφεί αρκετά στην διπλωματική αυτή εργασία.

5.2 Μελλοντική δουλειά

Στην διπλωματική αυτή εργασία, σκοπός ήταν η υλοποίηση ενός εξυπηρετητή ο οποίος να προσφέρει την υπηρεσία της γειτνίασης σε κινητές συσκευές Android, με σκοπό την αποστολή μηνυμάτων μεταξύ των κινητών αυτών συσκευών, κάνοντας χρήση των δύο αλγορίθμων γειτνίασης και παρουσίαση των γειτονικών συσκευών σε χάρτη στην κινητή συσκευή.

Από τις προαναφερθέν λειτουργίες, αυτές οι οποίες δεν υλοποιήθηκαν, ήταν η ανταλλαγή μηνυμάτων, επομένως και η υλοποίηση δικτύου από ομότιμους (peer to peer network) επάνω στις κινητές συσκευές Android, αλλά επίσης δεν έγινε υλοποίηση του αλγορίθμου PNA, ο οποίος είναι πολύ αποδοτικός στην εύρεση των K κοντινότερων

γειτονικών συσκευών προς την συσκευή που εκτελεί την επερώτηση. Όταν γίνει η υλοποίηση του αλγορίθμου αυτού, τότε θα γίνει η σύγκριση με τον αλγόριθμο BNA ο οποίος έχει ήδη υλοποιηθεί.

Εκτός από αυτό, μελλοντική δουλειά μπορεί να γίνει στην εφαρμογή του πελάτη στην κινητή συσκευή είναι η αληθινή αποστολή στοιχείων από τους κόμβους οι οποίοι είναι συνδεδεμένοι με τον εξυπηρετητή και όχι τα πλασματικά τυχαία δεδομένα τα οποία αποστέλλονται τώρα. Επίσης υπάρχει η ιδέα για την παροχή της υπηρεσίας της γειτνίασης, σε μορφή βιβλιοθήκης, για να μπορούν οι υπόλοιποι προγραμματιστές να την χρησιμοποιήσουν με σκοπό να τους παρέχεται η υπηρεσία μας, κάνοντας μόνο κλήσεις προς τις συναρτήσεις της βιβλιοθήκης μας, οι οποίες θα είναι υπεύθυνες για την αλληλεπίδραση με τον εξυπηρετητή μας.

Πέρα από αυτά, υπάρχουν σκέψεις για την μεταφορά της εφαρμογής κινητής συσκευής σε έξυπνη κινητή συσκευή iPhone, λόγω της μεγάλης εξάπλωσης της κινητής συσκευής αυτής, πιο γενικά του iOS, του λειτουργικού δηλαδή που τρέχουν οι κινητές συσκευές της εταιρίας Apple.

Στην υλοποίηση του εξυπηρετητή για μελλοντική δουλειά, μπορεί να γίνει διαχωρισμός του εξυπηρετητή της υπηρεσίας της γειτνίασης, σε δύο εξυπηρετητές. Έτσι, με αυτό τον τρόπο θα έχουμε τον διαχωρισμό των εφαρμογών που εξυπηρετούνται, με τρόπο ώστε να μην μπλέκονται οι χρήστες της μίας εφαρμογής με αυτούς της άλλης. Ακόμα, θα μπορούσε να γίνει αλλαγή έτσι ώστε ο εξυπηρετητής να μην προϋποθέτει την συνεχή σύνδεση του πελάτη με τον εξυπηρετητή, αλλά όταν ανευρεθούν τα αποτελέσματα, να αποστέλλονται στον κάθε πελάτη ανοίγοντας νέα σύνδεση.

Τέλος, τα στοιχεία που αποστέλλει ένας χρήστης για συμπερίληψη του στους υπολογισμούς, να διατηρούνται και για μερικές επόμενες εκτελέσεις του αλγορίθμου, έτσι να μην χάνονται συνεχώς αυτά τα στοιχεία και να έχουμε πολύ μικρό αριθμό από πελάτες, αλλά επίσης και για να τα εκμεταλλευτούμε με σκοπό να μειώσουμε τις συγκρίσεις.

Βιβλιογραφία

[1]. Wikipedia, Ορισμός του Twitter, http://en.wikipedia.org/wiki/Twitter

[2]. W3C, Πως λειτουργεί το Geolocation, http://whatismyipaddress.com/w3c-geolocation

[3]. The Economist, γραφικές παραστάσεις με δεδομένα για κατασκευάστριες εταιρίες έξυπνων κινητών συσκευών, http://www.economist.com/blogs/dailychart/2011/02/daily_chart_mobile-phone_market

[4]. Χρήσης την γεωγραφικής θέσης στην καθημερινότητα, http://nowsourcing.com/2010/07/27/10-ways-geolocation-is-changing-the-world/

[5]. Proximity 360 εφαρμογή, http://www.threespaces.com/home.html

[6]. Pubnub εφαρμογή, http://www.pubnub.com/

[7]. Ζεϊναλιπούρ, Δ (2010) *Σημειώσεις Μαθήματος ΕΠΛ371 – Προγραμματισμός Συστημάτων*, Τμήμα Πληροφορικής, Πανεπιστήμιο Κύπρου, http://www.cs.ucy.ac.cy/courses/EPL371/.

[8]. Stevens W. Richard, Fenner Bill, Rudoff M. Andrew. UNIX Network Programming: The sockets Networking API. Massachusetts: Boston, 2004.

[9]. PowerTutor εφαρμογή, http://powertutor.org

[10]. Android, Λειτουργικό σύστημα «έξυπνων» κινητών συσκευών, http://www.android.com/

[11]. Tathagata Das, Prashanth Mohan, Venkata N. Padmanabhan, Ramachandran Ramjee, Asankhaya Sharma, "*PRISM: Platform for Remote Sensing using Smartphones*" In MobilSys, 2010

[12]. Martin Azizyan, Ionut Constandache, Romit Roy Choudhury, *"SurroundSense: Mobile Phone Localization via Ambience Fingerprinting"* In MibiCom'09

[13]. Andrew T. Campbell, Shane B. Eisenman, Nicholas D. Lane, Emiliano Miluzzo, Ronald A. Peterson, "*PeopleCentric Urban Sensing"* In WICON, 2006

[14]. Ting Liu, Christopher M. Sadler, Pei Zhang, "*Margaret Martonosi,Implementing Software on Resource-Constrained Mobile Sensors: Experiences with Impala and ZebraNet*" In MobilSys, 2004

[15]. Emre Sarigol, Oriana Riva, Gustavo Alonso, *"A Tuple Space for Social Networking on Mobile Phones"*

[16]. Arvind Thiagarajan, Lenin Ravindranath, Katrina LaCurts, Sivan Toledo Jakob Eriksson ,"*VTrack: Accurate, Energy-aware Road Traffic Delay Estimation Using Mobile Phones*", In SenSys, 2009

[17]. Yu Zheng, Like Liu, Longhao Wang, Xing Xie, "*Learning Transportation Mode from Raw GPS Data for Geographic Applications on the Web*", In WWW'08

[18]. Yu Zheng, Lizhu Zhang, Xing Xie, Wei-Ying Ma*, "Mining Interesting Locations and Travel Sequences* from. *GPS Trajectories*.", In WWW'09

[19]. Zeinalipour-Yazti D., Lin S., Gunopulos D., *"Distributed Spatio-Temporal Similarity Search"* In CIKM, 2006.

[20]. Chow C-Y., Mokbel M.F., Aref W.G., *"Casper*: Query Processing for Location Services without Compromising Privacy"* In ACM TODS 34(4), pp. 1-48, 2009.

[21]. Wiley Publishing, Inc.10475, Crosspoint Boulevard, Indianapolis, IN 46256 ,2009, "Professional Android Application Development" .978-0-470-34471-2

[22]. Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon, *"What is Twitter, a Social Network or a News Media?"*

[23]. B. A. Huberman, D. M. Romero, and F. Wu. Social networks that matter: *Twitter under the microscope*. arXiv:0812.1045v1, Dec 2008.

[24]. A.Java,X.Song,T.Finin,andB.Tseng. *Why we twitter: understanding microblogging usage and communities*. In Proc. of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis. ACM, 2007.

[25]. Anlei Dong, Ruiqiang Zhang, Pranam Kolari, Jing Bai, Fernando Diaz, Yi, Chang, Zhaohui Zheng: *Time is of the Essence: Improving Recency Ranking Using Twitter Data*

[26]. Tianyin Xu, Yang Chen, Xiaoming Fu, Pan Hui, *Twittering by Cuckoo – Decentralized and Socio-Aware Online Microblogging Services*

[27]. Michael Mathioudakis, Nick Koudas, *TwitterMonitor: Trend Detection over the Twitter Stream*

[28]. Barıs Guc, Maria Grineva, Donald Kossmann, Master's Thesis: *Information Filtering on Micro-blogging Services*

[29]. Twitter Data Project, 140kit : http://140kit.com/

[30]. Cellular Mobile Network datasets, Crawdad : http://crawdad.cs.dartmouth.edu/

[31]. KAIST, Dataset from Twitter, http://an.kaist.ac.kr/pub_date.html

[32]. Reality Mining , http://reality.media.mit.edu/

[33]. Twitter Followers, ICWSM 2010 Dataset, http://twitter.mpi-sws.org/data-icwsm2010.html

[34]. Martin Azizyan, Ionut Constandache, Romit Roy Choudhury, *SurroundSense: Mobile Phone Localization via Ambience Fingerprinting*

[35]. Emiliano Miluzzo, Nicholas D. Lane, Kristóf Fodor, Ronald Peterson, Hong Lu, Mirco Musolesi, Shane B. Eisenman, Xiao Zheng, Andrew T. Campbell, *Sensing Meets Mobile Social Networks: The Design, Implementation and Evaluation of the CenceMe Application*

[36]. Shravan Gaonkar, Romit Roy Choudhury, *Micro-Blog: Map-casting from Mobile Phones to Virtual Sensor Maps*

[37]. Zhimin Yang, Boying Zhang, Jiangpeng Dai, Adam C. Champion, Dong Xuan Du Li, *E-SmallTalker: A Distributed Mobile System for Social Networking in Physical Proximity*

[38]. Balachander Krishnamurthy, Craig E. Wills, *Privacy Leakage in Mobile Online Social Networks*

[39]. Feng Li, Yinying Yang, Jie Wu, *CPMC: An Efficient Proximity Malware Coping Scheme in Smartphone-based Mobile Networks*

[40]. G. Chatzimilioudis, D. Zeinalipour-Yazti, M. D. Dikaiakos, Technical Report, University of Cyprus 2011

[41]. CNN, www.cnn.com

[42]. Thomas Brinkhoff, *A Framework for Generating Network-Based Moving Objects*, GeoInformatica, Vol. 6, No. 2, 2002, pp. 153-180.

Παράρτημα Α

Πρόγραμμα Εξυπηρετητή σε γλώσσα προγραμματισμού C.

```
#include <string.h>
#include "arraylist.h"

/*
  constants
*/
#define ARRAYLIST_INITIAL_CAPACITY 10
#define ARRAYLIST_CAPACITY_DELTA 10

static const size_t object_size = sizeof(Object);
/*
  structures
*/
struct Arraylist_Struct {
  int _current_capacity;
  Object *_data;
  int _size;
  const Boolean (*_equals)();
};


/*
  private function declarations
*/
static void *checked_malloc(const size_t size);


void arraylist_free(const Arraylist list)
{
  free(list->_data);
  free(list);
}

Arraylist arraylist_create(const Boolean (*equals)(const Object
object_1, const Object object_2))
{
  Arraylist list;

#ifdef GOK_DEBUG
  list = malloc(sizeof(struct Arraylist_Struct));
#else
  list = checked_malloc(sizeof(struct Arraylist_Struct));
#endif
  list->_current_capacity = ARRAYLIST_INITIAL_CAPACITY;
#ifdef GOK_DEBUG
  list->_data = malloc(object_size * list->_current_capacity);
#else
  list->_data = checked_malloc(object_size * list-
>_current_capacity);
#endif
  list->_size = 0;
  list->_equals = equals;

  return list;
}

Boolean arraylist_add(const Arraylist list, Object object)
{
  int old_size = arraylist_size(list);
  int new_capacity;
  Object *new_data;

  (list->_size)++;
  if (old_size == list->_current_capacity)
    {
      new_capacity = list->_current_capacity +
ARRAYLIST_CAPACITY_DELTA;
#ifdef GOK_DEBUG
      new_data = malloc(object_size * new_capacity);
#else
      new_data = checked_malloc(object_size * new_capacity);
#endif
      memcpy(new_data, list->_data, object_size * old_size);
      free(list->_data);
      (list->_data) = new_data;
      list->_current_capacity = new_capacity;
    }
  (list->_data)[old_size] = object;
  return TRUE;
}

Boolean arraylist_remove(const Arraylist list, const Object
object)
{
  int length = arraylist_size(list);
  int last_index = length - 1;
  int new_size, new_capacity;
  int index;

  for (index = 0; index < length; index++)
    {
      if ((*list->_equals)(arraylist_get(list, index), object))
                {
                  (list->_size)--;
                  if (index < last_index)
                    {
                      memmove(list->_data + index, list->_data + index
+ 1, object_size * (last_index - index));
                      new_size = list->_size;
                      new_capacity = list->_current_capacity -
ARRAYLIST_CAPACITY_DELTA;
                      if (new_capacity > new_size)
                                {
                                  list->_data = realloc(list->_data,
object_size * new_capacity);
                                  list->_current_capacity = new_capacity;
                                }
                    }
                  return TRUE;
                }
    }
  return FALSE;
}

Boolean arraylist_contains(const Arraylist list, const Object
object)
{
  return (arraylist_index_of(list, object) > -1);
}
```

```
int arraylist_index_of(const Arraylist list, const Object object)
{
 int length = arraylist_size(list);
 int index;

 for (index = 0; index < length; index++)
  {
   if ((*list->_equals)(arraylist_get(list, index), object))
            {
             return index;
            }
  }
 return -1;
}

Boolean arraylist_is_empty(const Arraylist list)
{
 return (0 == arraylist_size(list));
}

int arraylist_size(const Arraylist list)
{
 return list->_size;
}

Object arraylist_get(const Arraylist list, const int index)
{
 return list->_data[index];
}

void arraylist_clear(const Arraylist list)
{
 list->_data = realloc(list->_data, object_size *
ARRAYLIST_INITIAL_CAPACITY);
 list->_current_capacity = ARRAYLIST_INITIAL_CAPACITY;
 list->_size = 0;
}

void arraylist_sort(const Arraylist list, const int
(*compare)(const Object object_1, const Object object_2))
{
 qsort(list->_data,
            arraylist_size(list),
            sizeof(Object),
            (int (*))compare);
}

static void *checked_malloc(const size_t size)
{
 void *data;

 data = malloc(size);
 if (data == NULL)
  {
   fprintf(stderr, "\nOut of memory.\n");
   fflush(stderr);
   exit(EXIT_FAILURE);
  }

 return data;
}

#ifndef __defined_arraylist_h
#define __defined_arraylist_h

#include <stdlib.h>
#include <stdio.h>
/*
 constants
*/
#undef TRUE
#define TRUE 1

#undef FALSE
#define FALSE 0

/*
 type definitions
*/
#undef Boolean
#define Boolean short unsigned int

#undef Object
#define Object void*

typedef struct Arraylist_Struct *Arraylist;

/*
 function declarations
*/
void arraylist_free(const Arraylist list);
Arraylist arraylist_create(const Boolean (*equals)(const Object
object_1, const Object object_2));
Boolean arraylist_add(const Arraylist list, Object object);
Boolean arraylist_remove(const Arraylist list, const Object
object);
Boolean arraylist_contains(const Arraylist list, const Object
object);
int arraylist_index_of(const Arraylist list, const Object object);
Boolean arraylist_is_empty(const Arraylist list);
int arraylist_size(const Arraylist list);
Object arraylist_get(const Arraylist list, const int index);
void arraylist_clear(const Arraylist list);
void arraylist_sort(const Arraylist list, const int
(*compare)(const Object object_1, const Object object_2));
#endif /* __defined_arraylist_h */

/*
 * haslink.c
 *
 * Authors:                    Theodoros Ioannou
 * Description:                Hash table for thread-pool
 */

#include "hashlink.h"
//global variables
int flag = 0;
//addnode function
Node *AddNode(pthread_t id, Node *head,int sock) {
        Node *new_node = NULL, *cur = NULL, *pre =
NULL;
        new_node = (Node *) malloc(sizeof(Node));
        time_t start;
        time(&start);
        new_node->start=start;
        new_node->sock=sock;
        if (new_node == NULL) {
                perror("Error allocation memory\n");
                exit(-1);
        }
        new_node->id = id;
        if (head == NULL) {//if the list is empty
                head = new_node;
                new_node->next = NULL;
        } else {
                pre = head;
                cur = head;
                while (cur) {
                        pre = cur;
                        cur = cur->next;
                }
                if (cur == NULL) {
                        new_node->next = NULL;
                        pre->next = new_node;
                }
        }
        return head;
}
```

```
//to delete a node from link list and free the space
Node *DeleteNode(Node *head ,pthread_t id) {
        Node *temp, *cur, *pre;
        if (!head)
                return head;
        pre = head;
        cur = head;
        while (cur) {
                if (cur->id == id)
                        break;
                pre = cur;
                cur = cur->next;
        }
        if (cur == pre) {
                temp = head;
                head = head->next;
        }
        if (cur) {
                temp = cur;
                pre->next = cur->next;
        }
        free(temp);
        return head;
}
//this a hashfunction
int hashfunction(pthread_t thr) {
        pthread_t sum = thr;
        if (sum < 0)
                sum = (-1) * (int)thr;
        return ((int)sum % HASHSIZE);
}

void freeList(struct h* list) {

   struct node* cur = list->head;
   struct node* temp = NULL;
   list->head = NULL;

   while (cur){
      temp = cur;
      cur = cur->next;
      free(temp);
   }
}

void insertNode(struct node* n, struct h* list) {

   struct node* cur = NULL;
   struct node* prev = NULL;

   if (list->head == NULL) {
      list->head = n;
   } else if (list->head != NULL) {

      cur = list->head;
      prev = NULL;
      while (cur != NULL) {
         if (n->d <= cur->d) {

            n->next = cur;
            if (prev == NULL) {
               list->head = n;
            } else {
               prev->next = n;
            }

            break;
         }

         prev = cur;
         cur = cur->next;
      }

      if (cur == NULL) {
         prev->next = n;
      }
   }
}
```

```
/*
 * hashlink.c
 *
 * Authors:                    Theodoros Ioannou
 * Description:                The header file for hashlink.c
 */

#ifndef HASHLINK_h_
#define HASHLINK_h_
#include "queue.h"
#define SUCC 1
#define ERR 0
#define HASHSIZE 50
//struct for link list
typedef struct Node_t Node;
struct Node_t{
        int sock;
        time_t start;
        pthread_t id;
        Node *next;
};
//addnode function
Node *AddNode(pthread_t id, Node *head,int sock);
//delete the  node with id
Node *DeleteNode(Node *head ,pthread_t id);
//hash function for the hash table
int hashfunction(pthread_t id);

struct h {
   struct node* head;
   int count;
};

struct node {
   struct client_t* compare2;
   double d;
   struct node* next;
};

void insertNode(struct node* n, struct h* list);

void freeList(struct h* list);

#endif /* HASHLINK_h_ */


#include "proximityAlgorithm.h"
#include <time.h>
clock_t startm, stopm;
#define START if ( (startm = clock()) == -1) {printf("Error
calling clock");exit(1);}
#define STOP if ( (stopm = clock()) == -1) {printf("Error calling
clock");exit(1);}
#define PRINTTIME printf( "\t%6.3f\n", ((double)stopm-
startm)/CLOCKS_PER_SEC);


extern lcond;
extern lmutex;

extern *clientsList;

extern proxRunning;
extern calculated;
extern resSendCount;

/// @brief The usual PI/180 constant
#define DEG_TO_RAD
0.017453292519943295769236907684886
```

```
/// @brief Earth's quatratic mean radius for WGS-84
#define EARTH_RADIUS_IN_METERS 6372797.560856

/** @brief Computes the arc, in radian, between two WGS-84 positions.
 *
 * The result is equal to <code>Distance(from,to)/EARTH_RADIUS_IN_METERS</code>
 *    <code>= 2*asin(sqrt(h(d/EARTH_RADIUS_IN_METERS )))</code>
 *
 * where:<ul>
 *    <li>d is the distance in meters between 'from' and 'to' positions.</li>
 *    <li>h is the haversine function: <code>h(x)=sin²(x/2)</code></li>
 * </ul>
 *
 * The haversine formula gives:
 *    <code>h(d/R) = h(from.lat-to.lat)+h(from.lon-to.lon)+cos(from.lat)*cos(to.lat)</code>
 *
 * @sa http://en.wikipedia.org/wiki/Law_of_haversines
 */


//haversin is the haversine function, haversin(θ) = sin2(θ/2) = (1−cos(θ))/2
//d is the distance between the two points (along a great circle of the sphere; see spherical distance),
//R is the radius of the sphere,
//φ1 is the latitude of point 1,
//φ2 is the latitude of point 2, and
//Δλ is the longitude separation,


double ArcInRadians(double lat1, double long1, double lat2, double long2) {
        double latitudeArc  = (lat1 - lat2) * DEG_TO_RAD;
        double longitudeArc = (long1 - long2) * DEG_TO_RAD;
        double latitudeH = sin(latitudeArc * 0.5);
        latitudeH *= latitudeH;
        double lontitudeH = sin(longitudeArc * 0.5);
        lontitudeH *= lontitudeH;
        double tmp = cos(lat1*DEG_TO_RAD) * cos(lat2*DEG_TO_RAD);
        return 2.0 * asin(sqrt(latitudeH + tmp*lontitudeH));
}

/** @brief Computes the distance, in meters, between two WGS-84 positions.
 *
 * The result is equal to <code>EARTH_RADIUS_IN_METERS*ArcInRadians(from,to)</code>
 *
 * @sa ArcInRadians
 */
double DistanceInMeters(double lat1, double long1, double lat2, double long2) {
        return EARTH_RADIUS_IN_METERS*ArcInRadians(lat1, long1, lat2, long2);
}


int runGreedyAlgorithm()
{
        int length = arraylist_size(clientsList);
        int index, index2;

        for (index = 0; index < length; index++)
        {
                for (index2=index+1; index2 < length; index2++){

                        Client *temp1, *temp2;

                        temp1=((Client *)arraylist_get(clientsList, index));
                        temp2=((Client *)arraylist_get(clientsList, index2));
                        double distance1_2=DistanceInMeters(temp1->latitude, temp1->longitude, temp2->latitude, temp2->longitude);

                        struct node* n= (struct node*) malloc(sizeof(struct node));
                        struct node* n2= (struct node*) malloc(sizeof(struct node));
                        if (!n || !n2) {
                                perror("AKNN:");
                                return;
                        }
                        n->compare2=temp2;
                        n->d=distance1_2;
                        n->next=NULL;

                        n2->compare2=temp1;
                        n2->d=distance1_2;
                        n2->next=NULL;

                        insertNode(n, temp1->resultSet);
                        insertNode(n2, temp2->resultSet);
                }
        }
}

int runCellProximityAlgorithm()
{

}


int runAlgorithm()
{
        proxRunning=TRUE;
        sleep(1);
        int length1 = arraylist_size(clientsList);
        printf("Number of clients: %d\n", length1);
        if(length1==0)
        {
                proxRunning=FALSE;
                return;
        }

        START
        runGreedyAlgorithm();
        STOP
        printf("Algorithm Greedy Run => DONE\n");
        PRINTTIME

        calculated=TRUE;
        while(resSendCount!=0)
        {
                sleep(1);
        }


        //FREE MEM of clients
        int length = arraylist_size(clientsList);
        int index;

        for (index = 0; index < length; index++)
```

```
		{
			freeList(((Client *)arraylist_get(clientsList, index))->resultSet);
			((Client *)arraylist_get(clientsList, index))->resultSet=NULL;
			arraylist_free(((Client *)arraylist_get(clientsList, index))->ncells);
		}

		arraylist_free(clientsList);
		//arraylist_clear(clientsList);
		clientsList = arraylist_create(clientEquals);

		calculated=FALSE;
		proxRunning=FALSE;
}



#ifndef PROX_ALGORITHM_h_
#define PROX_ALGORITHM_h_

#include <math.h>
#include "serverFunctions.h"


// todo: structures to be used with the algorithms
// todo: function prototypes


double ArcInRadians(double lat1, double long1, double lat2, double long2);
double DistanceInMeters(double lat1, double long1, double lat2, double long2);

int runGreedyAlgorithm();
int runCellProximityAlgorithm();
int runAlgorithm();


#endif /* PROX_ALGORITHM_h_ */

/*
 * proximityServer.c
 *
 * Authors:		Theodoros Ioannou
 * Description:		The main program's source code
 */

#include "proximityServer.h"

/* Server with Internet stream sockets */
pthread_cond_t cond = PTHREAD_COND_INITIALIZER;
pthread_mutex_t Qmutex =
PTHREAD_MUTEX_INITIALIZER;

//Mutex for the list of connected clients
//pthread_cond_t lcond = PTHREAD_COND_INITIALIZER;
pthread_mutex_t lmutex = PTHREAD_MUTEX_INITIALIZER;
pthread_mutex_t cotsiosMutex =
PTHREAD_MUTEX_INITIALIZER;
//Mutex for the sent Results
pthread_mutex_t Rmutex =
PTHREAD_MUTEX_INITIALIZER;

Node *HashTable[HASHSIZE];

Arraylist clientsList;


int proxRunning=FALSE;
int calculated=FALSE;
int resSendCount=0;

int main(int argc, char *argv[]) {
		/* Initialize mutex and condition variable objects */

		//Replies
		pthread_mutex_init(&Rmutex, NULL);

		//List of user reports
		pthread_mutex_init(&lmutex, NULL);
//		pthread_cond_init(&lcond, NULL);

		clientsList = arraylist_create(clientEquals);

		//default variables
		config.threads = 50;
		config.port = 65002;
		config.timeout = 10;//minutes
		/*initialize configurations*/
		readConfigFile();

		/*variables to use*/
		int sock, newsock, serverlen, clientlen, i, tempsock;
		struct sockaddr_in server, *clientptr;
		struct sockaddr *serverptr;


		/* Create socket */
		if ((sock = socket(AF_INET, SOCK_STREAM, 0)) <
0) {
			perror("socket");
			exit(1);
		}

		server.sin_family = AF_INET; /* Internet domain */
		server.sin_addr.s_addr = htonl(INADDR_ANY); /*
My Internet address */
		server.sin_port = htons(config.port); /* The given port
*/
		serverptr = (struct sockaddr *) &server;
		serverlen = sizeof(server);


		int optval;
		int optlen;
		char *optval2;

		// set SO_REUSEADDR on a socket to true (1):
		optval = 1;
		setsockopt(sock, SOL_SOCKET, SO_REUSEADDR,
&optval, sizeof optval);

		/* Bind socket to address */
		while (bind(sock, serverptr, serverlen) < 0) {
			perror("bind");
			sleep(5);
		}
		/* Listen for connections */
		if (listen(sock, 1000) < 0) { /* 5 max. requests in
queue */
			perror("listen");
			exit(1);
		}
		printf("Listening for connections to port %d\n",
config.port);

		int maxfd, maxi;
		int nready, client[FD_SETSIZE];
		fd_set rset, allset;

		maxfd=sock;
		maxi=-1;

		for (i = 0; i < FD_SETSIZE; ++i) {
			client[i]=-1;
		}
```

```
		FD_ZERO(&allset);
		FD_SET(sock, &allset);

		pthread_t tid;

	/////
	pthread_create(&tid, NULL, pthreadAlarm, NULL);
	/////

		while (TRUE) {/*start of while*/

			rset=allset;
			nready=select(maxfd+1, &rset, NULL,
NULL, NULL);

			if (FD_ISSET(sock, &rset))
			{

				while (nready>0)
				{
					//allocate memory to
be fill
					clientptr=(struct
sockaddr_in *)malloc(sizeof(struct sockaddr_in));

				/* Accept connection */
				if ((newsock =
accept(sock,(struct sockaddr *) clientptr, &clientlen)) < 0) {
					perror("accept");
					exit(1);
				}

				//setsockopt(newsock,
SOL_SOCKET, SO_RCVTIMEO,(struct timeval
*)&tv,sizeof(struct timeval));
				args *tempArgs=(args *)
malloc(sizeof(args));
				tempArgs->newsock=newsock;
				tempArgs->client= clientptr;

				pthread_create(&tid, NULL,
pthreadFunction, (void *)tempArgs);

				nready--;

				}
			}

		}/*end of while*/
		return 0;
} /* end of main() */

/*
 * proximityServer.h
 *
 * Authors:		Theodoros Ioannou
 * Description:		The main program's header file
 */

#ifndef PROXIMITYSERVER_H_
#define PROXIMITYSERVER_H_
#include "serverFunctions.h"
#include <sys/select.h>
#include <sys/time.h>


#endif /* PROXIMITYSERVER_H_ */

/*
 * queue.c
 *
 * Authors:		Theodoros Ioannou
 * Description:		Thread-pool queue
 */

#include "queue.h"

extern cond;
extern mutex;
extern Qmutex;

void enqueue(int i, struct sockaddr_in *client) {
		node *new_node;
		new_node = (node *) malloc(sizeof(node) * 2);
		new_node->next = NULL;
		new_node->sock = i;
		new_node->client = *client;

		if (queueofthread.size == 0) {
			queueofthread.tail = new_node;
			queueofthread.head = new_node;
		} else {
			(queueofthread.tail)->next = new_node;
			queueofthread.tail = new_node;
		}
		queueofthread.size++;

}
node* dequeue() {
		node *temp;
		if (queueofthread.size == 0)
			return NULL;
		temp = queueofthread.head;
		queueofthread.head = queueofthread.head->next;
		queueofthread.size--;
		return temp;
}

int getQueueSize(){

		return queueofthread.size;

}

/*
 * queue.h
 *
 * Authors:		Theodoros Ioannou
 * Description:		Queue header
 */

#ifndef QUEUE_h_
#define QUEUE_h_

#include <stdio.h> // NULL
#include <stdlib.h> // exit
#include <sys/types.h>//system types
#include <ctype.h>//Character handling functions
#include <dirent.h> // opendir, readdir, closedir
#include <sys/stat.h> // lstat
#include <string.h>//string functions
#include <pwd.h>//for pwd
#include <grp.h>//group id
#include <getopt.h>//for the getopt function
#include <fcntl.h>//file functions
#include <sys/types.h> /* For sockets */
#include <sys/socket.h> /* For sockets */
#include <netinet/in.h> /* For Internet sockets */
#include <netdb.h> /* For gethostbyaddr() */
#include <fcntl.h>// file control options:
#include <unistd.h>// standard symbolic constants and types for
unix
#include <sys/stat.h>
#include <dirent.h>
#include <pthread.h> //For pthreads
#include <time.h>
#include <signal.h> /* For SIGTERM, SIGSTOP, SIGCONT */
```

```
#include <arpa/inet.h>//checking ip address
#include <errno.h> /* For errno variable */
typedef struct Nodes node;
struct Nodes {
	int sock;
	struct sockaddr_in client;
	node *next;
};
typedef struct queue_t queue;
struct queue_t {
	node *head;
	node *tail;
	int size;
};

queue queueofthread;
node* dequeue();
void enqueue(int i, struct sockaddr_in *client);
int getQueueSize();


#endif /* QUEUE_h_ */

/*
 * serverFunctions.c
 *
 * Authors:		Theodoros Ioannou
 * Description:		Server's utilities functions
 */

#include "serverFunctions.h"

//extern lcond;
extern lmutex;

extern Rmutex;

extern cotsiosMutex;


extern *clientsList;

extern proxRunning;
extern calculated;
extern resSendCount;
//read configuration  file
void readConfigFile() {

	int i, k = 0, fd, n; //file descriptor
	char buf[BYTES * 2], *variable; //original buffer 1024
	char line[BYTES]; //new buffer

	variable = (char *) malloc(sizeof(char) * 1024);

	//variable must not be null
	if (!variable) {
		perror("malloc");
		exit(1);
	}

	if ((fd = open("./serverconf.cfg", O_RDONLY)) == -1) {
		perror("WARNING: Using defaults values\nopen(serverconf.cfg)");
		return;
	}

	bzero(buf, sizeof buf); /* Initialize buffer */
	while ((n = read(fd, buf, sizeof(buf))) > 0) {

		// move inside the buffer
		for (i = 0; i < n; i++) {
			// when a new line is found
			if (buf[i] == '\n') {

				line[k] = '\0';
				k = 0;

				if (i != n - 1) // if its not the end of buffer
					i++; // move i to the next char
				if (line[k] != '#' && line[k] != '\n' && line[k] != '\r') {
					variable = strtok(line, "="); //Variable contains the name of the variable that will be initialized
					variable = convertToUpperCase(variable);
					if (strcmp(variable, "THREADS") == 0) {

						variable = strtok(NULL, "="); //variable will now contain the initialized number

						if (variable != NULL)

							config.threads = atoi(variable);

						else

							fprintf(

								stderr,

								"Warning! You didn't give a value for the threads!\n"

								"As default the value for threads will be %d",

								config.threads);
					} else if (strcmp(variable, "PORT") == 0) {

						variable = strtok(NULL, "="); //variable will now contain the initialized number

						if (variable != NULL)

							config.port = atoi(variable);

						else

							fprintf(

								stderr,

								"Warning! You didn't give a value for the port!\n"

								"As default the value for port will be %d",

								config.port);
					} else if (strcmp(variable, "TIMEOUT") == 0) {

						variable = strtok(NULL, "="); //variable will now contain the initialized number

						if (variable != NULL)

							config.timeout = atoi(variable);
```

```
		else
			fprintf(
					stderr,
					"Warning! You
didn't give a value for timeout!\n"
					"As default the value
for timeout will be %d",
					config.timeout);
				}
			}
			// Re-Initialize
buffers
			bzero(line, sizeof
line);
		}
		if (buf[i] == '\0' || buf[i] == '\n')
			continue;
		// store the characters of buf in a
line
		line[k] = buf[i];
		k++;
	}
	bzero(buf, sizeof buf); /* Initialize buffer */
    }
}

int countWords(char *line) {

    int i = 0, count = 0;

    line[(strlen(line)) - 2] = '\0';

    if (isspace(line[0]))
	return 999;

    while (line[0]) {

	if (isspace(line[i])) {
	    if (isspace(line[i + 1]) || line[i +
1] == '\0')
		return 999;
	    else
		++count;
	}

	if (line[i + 1] == '\0')
	    return ++count;

	i++;
    }

    return 0;
}

void serverMLANS(int sock, char *msg) {
    int len = strlen(msg);
    char buf[len];
    bzero(buf, sizeof buf); /* Initialize buffer */
    sprintf(buf, "%s", msg); /* Ready message */
    if (write(sock, buf, sizeof(buf)) < 0) {/* Send message
*/
	perror("serverANSwrite");
	return;
    }
}

void serverANS(int sock, char *msg) {
    int len = strlen(msg) + 4;
    char buf[len];
    bzero(buf, sizeof buf); /* Initialize buffer */
    sprintf(buf, "%s", msg); /* Ready message */
    buf[strlen(buf)] = '\r';
    buf[strlen(buf)] = '\n';
    if (write(sock, buf, sizeof(buf)) < 0) {/* Send message
*/
	perror("serverANSwrite");
	return;
    }
}

void serverOK(int sock, char *msg) {
    int len = strlen(msg) + 6;
    char buf[len];
    bzero(buf, sizeof buf); /* Initialize buffer */
    sprintf(buf, "+OK %s", msg); /* Ready message */
    buf[strlen(buf)] = '\r';
    buf[strlen(buf)] = '\n';
    if (write(sock, buf, sizeof(buf)) < 0) {/* Send message
*/
	perror("serverOKwrite");
	return;
    }
}

void serverERR(int sock, char *msg) {
    int len = strlen(msg) + 7;
    char buf[len];
    bzero(buf, sizeof buf); /* Initialize buffer */
    sprintf(buf, "-ERR %s\r\n", msg); /* Ready message */
    buf[strlen(buf)] = '\r';
    buf[strlen(buf)] = '\n';
    if (write(sock, buf, sizeof(buf)) < 0) {/* Send message
*/
	perror("serverERRORwrite");
	return;
    }
}

void *pthreadFunction(void* pthreadArg) {

    pthread_detach(pthread_self());


    while (proxRunning==TRUE)
	sleep(1);

    args *tempArgs=(args *) pthreadArg;

    /*initialization  state */
    initialization(tempArgs->newsock, tempArgs->client);

    pthread_mutex_lock(&Rmutex);
    resSendCount++;
    pthread_mutex_unlock(&Rmutex);

//  while (calculated==FALSE)
//  {
//	sleep(1);
//  }

    //SEND KNN

    //sendKNearest(pthreadArgTemp->newsock,
pthreadArgTemp->client);
    //			if(arraylist_size(clientsList) >=
3){
    //
    printClients(clientsList);
    //sendKNearest(pthreadArgTemp->newsock,
pthreadArgTemp->client, 4);
    //			}
```

```
		pthread_mutex_lock(&Rmutex);
		resSendCount--;
		pthread_mutex_unlock(&Rmutex);

		close(tempArgs->newsock);

}

void *pthreadAlarm(void* pthreadArg){

		pthread_detach(pthread_self());
		signal(SIGALRM,ALARMhandler);
		alarm(PROX_ALG_RUN_SEC);
		while (1) sleep(1);
}

void  ALARMhandler(int sig)
{
		runAlgorithm();
		alarm(PROX_ALG_RUN_SEC);		/* set
alarm for next run  */
		signal(SIGALRM, ALARMhandler);	/* reinstall the
handler  */
}


void printClients(Arraylist al){

		Client* tempClient;
		int i=0;
		while(tempClient=(Client*)arraylist_get(al,i)){
			printf("===>Client %d:\n", i+1);
			printClient(tempClient);
			i++;
		}
}

const Boolean clientEquals(const void *c1, const void *c2)  {

		if (strcmp(((Client*)c1)->ip, ((Client*)c2)->ip)!=0){
			if (((Client*)c1)->port!=((Client*)c2)-
>port){
				return 0;
			}
		}
		return 1;
}

const Boolean cellEquals(const void *c1, const void *c2){

		if (((Cell*)c1)->cid != ((Cell*)c2)->cid){
			return 0;
		}
		return 1;
}

const Boolean resSetEquals(const void* c1, const void* c2)
{
		if (((ResSetNode*)c1)->distance ==
((ResSetNode*)c2)->distance)
			return 1;
}

const int clientCompare(const void *c1, const void *c2){

		//eq:0  first:-1 second:1
		if (strcmp(((Client*)c1)->ip, ((Client*)c2)->ip)==0)
			if(((Client*)c1)->port==((Client*)c2)-
>port)
				return 0;
			else if (((Client*)c1)->port>((Client*)c2)-
>port)
				return -1;
			else return 1;
		else
			return strcmp(((Client*)c1)->ip,
((Client*)c2)->ip);
}

const int cellCompare(const void *c1, const void *c2){
		//eq:0  first:-1 second:1
		if (((Cell*)c1)->cid == ((Cell*)c2)->cid)
			return 0;
		else if (((Cell*)c1)->cid > ((Cell*)c2)->cid)
			return -1;
		else return 1;
}

const int resSetCompare(const void *c1, const void *c2)
{
		//eq:0  first:-1 second:1
		if (((ResSetNode*)c1)->distance ==
((ResSetNode*)c2)->distance)
			return 0;
		else if (((ResSetNode*)c1)->distance >
((ResSetNode*)c2)->distance)
			return -1;
		else return 1;
}

/*
 * serverFunctions.h
 *
 * Authors:			Theodoros Ioannou
 * Description:			The main program's header file
 */

#ifndef SERVERFUNCTIONS_H_
#define SERVERFUNCTIONS_H_
#include "serverStates.h"

#define PROX_ALG_RUN_SEC 30


// struct for storing details about the configuration file
typedef struct config_file_t config_file;
struct config_file_t {
		int timeout;
		int threads;
		int port;
};

// struct for storing details about a cellular tower
typedef struct cell_t Cell;
struct cell_t {
		int cid;
};

// struct for the arraylist of the result set
typedef struct resultSetNode_t ResSetNode;
struct resultSetNode_t {
		Client *comparedTo;
		double distance;
};

//struct for arguments for pthread
typedef struct arg_t args;
struct arg_t {
		int newsock;
		struct sockaddr_in *client;
		char ipAddress[512];
};

//Global struct for configuration file
config_file config;

//compilation replacement values
```

```
//#define ERROR 0
//#define FALSE 0
//#define SUCCESS 1
//#define TRUE 1
#define BYTES 512
#define SECONDS 20


// functions prototype's
void serverOK(int sock, char *msg);
void serverERR(int sock, char *msg);
void serverANS(int sock, char *msg);
void serverMLANS(int sock, char *msg);
int countWords(char *line);
void readConfigFile();
void *pthreadFunction(void *pthreadArg);
void timeoutChecker();
void sighand(void *args);
void resetThread(int sock,pthread_t self);
void resetTimer(pthread_t self);
void *threadPoolManager();

void printClients(Arraylist al);

const Boolean clientEquals(const void* c1, const void* c2);
const Boolean cellEquals(const void* c1, const void* c2);
const Boolean resSetEquals(const void* c1, const void* c2);


const int clientCompare(const void *c1, const void *c2);
const int cellCompare(const void *c1, const void *c2);
const int resSetCompare(const void *c1, const void *c2);

void  ALARMhandler(int sig);
void *pthreadAlarm();

#endif /* SERVERFUNCTIONS_H_ */

/*
 * authorization.c
 *
 * Authors:                    Theodoros Ioannou
 * Description:                The authorization state source
code
 */

#include "serverFunctions.h"

//extern cond;
//extern mutex;
//extern Qmutex;

//extern lcond;
extern lmutex;
extern *clientsList;

extern proxRunning;
extern calculated;


// representing the authorization state
void initialization(int sock, struct sockaddr_in *client) {
        char buf[1024];
        //      struct hostent *rem;

        serverOK(sock, "Proximity server ready"); /* Ready
message */

        /* Find client's address */
        //      if ((rem = gethostbyaddr((char *)
&client.sin_addr.s_addr,
        //
        sizeof(client.sin_addr.s_addr), client.sin_family)) ==
NULL) {
        //              herror("gethostbyaddr"); //
herror(): Similar to perror but uses the h_errno variable (set by
name resolution functions to return error values).
        //              exit(1);
        //      }


//      printf("Accepted connection\n"); //from %s\n",
inet_ntoa(client.sin_addr));

        bzero(buf, sizeof(buf)); /* RE-Initialize buffer */
        if (read(sock, buf, sizeof(buf)) < 0) { /* Get message
*/
                perror("read");
                return;
        }

        //if someone closes the terminal
        if (!strcmp(buf, "")) {
                printf("The terminal has closed\n");
                return;
        }

        //      printf("CLIENT??! %s\n",inet_ntoa(client-
>sin_addr));

        //Create a new node of client
        Client *ctemp= (Client*) malloc(sizeof(Client));
        if (!ctemp) {
                perror("ClientListNode:");
                return;
        }


        ctemp->resultSet=(struct h *) malloc(sizeof (struct
h));

        if (!(ctemp->resultSet)) {
                perror("ClientResultSet:");
                return;
        }

        ctemp->resultSet->head=NULL;
        ctemp->resultSet->count=0;



        //Thread ID
        ctemp->threadId=pthread_self();

        //IP
        ctemp->ip=(char*) malloc
(sizeof(char)*(strlen(inet_ntoa(client->sin_addr))+1));
        if (!ctemp->ip) {
                perror("ClientIPstring:");
                return;
        }
        bzero(ctemp->ip,sizeof(char)*(strlen(inet_ntoa(client-
>sin_addr))+1));
        strcpy(ctemp->ip,inet_ntoa(client->sin_addr));

        //buf consist of :
port|latitude|Longitude|Accuracy|CID|LAC|MCC|MNC|nCells*

        //      if (strchr(buf,'\r')!=NULL)
        //              buf[strchr(buf,'\r')-buf]='\0';
        //      else
        //              buf[strlen(buf)-1]='\0';
        //      printf ("Splitting string \"%s\" into
tokens:\n",buf);



        char * character_pointer;
```

```
		//Port
		character_pointer = strtok (buf,",");
		if(character_pointer)
			ctemp->port=atoi(character_pointer);

		//Latitude
		character_pointer = strtok (NULL, ",");
		if(character_pointer)
			ctemp->latitude=atof(character_pointer);

		//Longitude
		character_pointer = strtok (NULL, ",");
		if(character_pointer)
			ctemp->longitude=atof(character_pointer);

		//		//Accuracy
		//		character_pointer = strtok (NULL, "|");
		//		ctemp->accuracy=atof(character_pointer);

		//CellId
		character_pointer = strtok (NULL, ",");
		if(character_pointer){
			ctemp->mycell= (MCell*) malloc(sizeof(MCell));
			if (!ctemp->mycell) {
				perror("MyCellM:");
				return;
			}

			ctemp->mycell->cid=atoi(character_pointer);
		}
		//		//Cell LAC
		//		character_pointer = strtok (NULL, "|");
		//		ctemp->mycell->lac=atoi(character_pointer);
		//
		//		//Cell MCC
		//		character_pointer = strtok (NULL, "|");
		//		ctemp->mycell->mcc=atoi(character_pointer);
		//
		//		//Cell MNC
		//		character_pointer = strtok (NULL, "|");
		//		ctemp->mycell->mnc=atoi(character_pointer);

		//nCells of the Neighboring cells
		character_pointer = strtok (NULL, ",");
		ctemp->ncells=arraylist_create(cellEquals);

		while (character_pointer != NULL)
		{
			Cell *tempCell=(Cell*) malloc(sizeof(Cell));
			if (!tempCell) {
				perror("NeigCellM:");
				return;
			}
			tempCell->cid=atoi(character_pointer);

			arraylist_add(ctemp->ncells, tempCell);
			character_pointer = strtok (NULL, ",");
		}


		ctemp->threadId=pthread_self();

		//		printClient(ctemp);

		//Insert the node of newclient into the clientsList list
		pthread_mutex_lock(&lmutex);
		arraylist_add(clientsList,ctemp);
		pthread_mutex_unlock(&lmutex);

		ctemp=NULL;

		serverOK(sock, "Data Received");

		//		printf("DATA RECEIVED!!!\n");
}

void sendKNearest(int sock, struct sockaddr_in client, int topK)
{

		//		while (calculated==FALSE)
		//			sleep(1);
		//
		//		//here send the results back to the users - arraylist_get loop to find my pthread_self()
		//
		//
		//		char buf[512];
		//		Arraylist topKList;
		//		//TODO: RUN PROXIMITY ALGORITHM TO DETERMINE NEIGHBORING DEVICES (not from here - cond-notify when ready )
		//
		//
		//		//TODO: SEND THE K CLOSER TO THE CLIENT (when notified)
		//
		//		//		pthread_mutex_lock(&lmutex);
		//		//		int pos = rand() % arraylist_size(clientsList);
		//		//		Client * tempClient= (Client *)arraylist_get(clientsList,pos);
		//		//
		//		pthread_mutex_unlock(&lmutex);
		//		//		printf("Nearest K sending\n%s\n",tempClient->ip);
		//
		//		//		Call playback
		//		//		serverANS(sock, tempClient->ip);
		//
		//		if(readPlayBackFile(&topKList))
		//		{
		//			//sent the topK neighbors
		//
		//			printf("SIZE of TOPK LIST : %d",arraylist_size(topKList));
		//
		//			Client* tempClient;
		//			int i=0;
		//			while(i<topK){
		//
		//				int r = rand() % arraylist_size(topKList);
		//				printf("===>Client %d:\n", i+1);
		//
		//				tempClient=(Client*)arraylist_get(topKList,r);
		//
		//
		//				char bufout[512]="";
		//				bzero(bufout, sizeof bufout);
		//				sprintf(bufout, "|%s|%d|%d|%d|%d|%d|",
		//
		//				tempClient->ip,
		//
		//				tempClient->port,
		//
		//				tempClient->mycell->cid,
```

```
//
tempClient->mycell->lac,
//
tempClient->mycell->mcc,
//
tempClient->mycell->mnc
//                                      );
//
printf("%s\n",bufout);
//                                      serverANS( sock,
bufout);
//
//                                      i++;
//                              }
//
//
//
//                              printf("Nearest K sent\n");
//
//
//              }
//              else
//              {
//                              perror("Nearest K not sent");
//                              return;
//              }
//
//              //Tell client that the list of topK neighbors
was sent
//              serverOK(sock,"Nearest K sent");
//
//              //GET the +OK from the client
//              bzero(buf, sizeof(buf)); /* RE-Initialize
buffer */
//              if (read(sock, buf, sizeof(buf)) < 0) { /* Get
message */
//                              perror("read");
//                              resetThread(sock,
pthread_self());
//              }
//
//              //if someone closes the terminal
//              if (!strcmp(buf, "")) {
//                              printf("The terminal has
closed\n");
//                              resetThread(sock,
pthread_self());
//                              return;
//              }
//              printf("Answer : %s\n",buf);
//
//              if (!strstr(buf,"+OK"))
//              {
//                              printf("An error occurred during
the communication with the client...\n");
//                              resetThread(sock,
pthread_self());
//              }




}

void printClient(Client *c){

        printf("IP : %s\nPORT : %d\nLatitude : %f\nLongitude
: %f\nAccuracy : %f\nMyCell : %d\nLAC : %d\nMCC :
%d\nMNC : %d\n",
                                c->ip, c->port, c->latitude,c-
>longitude, c->accuracy, c->mycell->cid, c->mycell->lac, c-
>mycell->mcc, c->mycell->mnc);

        Cell* tempCell;
        int i=0;
        while(tempCell=(Cell*)arraylist_get(c->ncells,i)){
                printf("NCell%d : %d\n",i+1, tempCell-
>cid);
                i++;
        }



}
//read PlayBack  file
int readPlayBackFile( Arraylist *topKList ) {

        *topKList = arraylist_create(clientEquals);

        int i, k = 0, fd, n; //file descriptor
        char buf[BYTES * 2], *variable; //original buffer
1024
        char line[BYTES]; //new buffer

        variable = (char *) malloc(sizeof(char) * 1024);

        //variable must not be null
        if (!variable) {
                perror("malloc");
                exit(1);
        }

        if ((fd = open("./cellIDs.txt", O_RDONLY)) == -1) {
                perror("WARNING: Using playback
file.\nopen(cellIDs.txt)");
                return FALSE;
        }

        bzero(buf, sizeof buf); /* Initialize buffer */
        while ((n = read(fd, buf, sizeof(buf))) > 0) {


                // move inside the buffer
                for (i = 0; i < n; i++) {

                        // when a new line is found
                        if (buf[i] == '\n') {

                                line[k] = '\0';
                                k = 0;

                                // the new client to
go into the topKlist
                                Client *ctemp=
(Client*) malloc(sizeof(Client));

                                if (i != n - 1) // if its
not the end of buffer
                                        i++; //
move i to the next char
                                if (line[k] != '#' &&
line[k] != '\n' && line[k] != '\r') {

                                        // IP
                                        variable =
strtok(line, "\t");
                                        ctemp-
>ip=(char*) malloc (sizeof(char)*(strlen(variable)+1));
                                        if
(!ctemp->ip) {

        perror("malloc");

        exit(1);
                                        }

        bzero(ctemp->ip,sizeof(char)*(strlen(variable)+1));
```

```
            strcpy(ctemp->ip,variable);

                                                    // PORT
variable =
strtok(NULL, "\t");
                                                    ctemp-
>port= atoi(variable);

                                                    // CellID
variable =
strtok(NULL, "\t");
                                                    ctemp-
>mycell= (MCell*) malloc(sizeof(MCell));
                                                    if
(!ctemp->mycell)
                                                    {

            perror("malloc");

            exit(1);
                                                    }
                                                    ctemp-
>mycell->cid=atoi(variable);

                                                    // LAC
                                                    variable =
strtok(NULL, "\t");
                                                    ctemp-
>mycell->lac= atoi(variable);
                                                    // MCC
                                                    variable =
strtok(NULL, "\t");
                                                    ctemp-
>mycell->mcc= atoi(variable);
                                                    // MNC
                                                    variable =
strtok(NULL, "\t");
                                                    ctemp-
>mycell->mnc= atoi(variable);

                                                    // NCells
                                                    ctemp-
>ncells=NULL;

            arraylist_add(*topKList,ctemp);

                                                    //

            printf("%s\n",ctemp->ip);

                                                    //
                                                    char
bufout[512]="";
                                                    //

            bzero(bufout, sizeof bufout);
                                                    //

            sprintf(bufout, "|%s|%d|%d|%d|%d|%d|", address, port,
cellId, Lac, Mcc, Mnc);

                                                    //

            sprintf(bufout, "|%s|%d|%d|%d|%d|%d|",
                                                    //

                        ctemp->ip,
                                                    //

                        ctemp->port,
                                                                        //
                                                                ctemp->mycell->cid,
                                                                        //
                                                                ctemp->mycell->lac,
                                                                        //
                                                                ctemp->mycell->mcc,
                                                                        //
                                                                ctemp->mycell->mnc
                                                                        //
                                                    );
                                                                        //
                                                    printf("%s\n",bufout);
                                                                        //
                                                    serverANS( sock, bufout);

                                                                }
                                                                // Re-Initialize
                                                    buffers
                                                                bzero(line, sizeof
                                                    line);
                                                            }

                                                            if (buf[i] == '\0' || buf[i] == '\n')
                                                                continue;
                                                            // store the characters of buf in a
                                                    line
                                                            line[k] = buf[i];
                                                            k++;
                                                    }
                                                    bzero(buf, sizeof buf); /* Initialize buffer */

                                                    //              Client* tempClient;
                                                    //              int i=0;
                                                    //
                                                    while(tempClient=(Client*)arraylist_get(topKList,i)){
                                                    //
                                                    printf("===>Client %d:\n", i+1);
                                                    //
                                                    //
                                                    //
                                                    //                                      char
                                                    bufout[512]="";
                                                    //
                                                    bzero(bufout, sizeof bufout);
                                                    //
                                                    sprintf(bufout, "|%s|%d|%d|%d|%d|%d|",
                                                    //
                                                    tempClient->ip,
                                                    //
                                                    tempClient->port,
                                                    //
                                                    tempClient->mycell->cid,
                                                    //
                                                    tempClient->mycell->lac,
                                                    //
                                                    tempClient->mycell->mcc,
                                                    //
                                                    tempClient->mycell->mnc
                                                    //                                      );
                                                    //
                                                    printf("%s\n",bufout);
                                                    ////
                                                    serverANS( sock, bufout);
```

```
                        //
                        //                              i++;
                        //                  }



                }
                return TRUE;
}

//convert a string To UpperCase
char* convertToUpperCase(char *str) {
                int i;
                char *temp = (char*) malloc(sizeof(char) *
(strlen(str) + 1));

                //Temp must not be null
                if (!temp) {
                                perror("malloc");
                                pthread_cancel(pthread_self());
                }

                for (i = 0; i < strlen(str); i++) {
                                temp[i] = toupper(str[i]);
                }
                temp[i] = '\0';

                return temp;
}


/*
 * proximityServer.h
 *
 * Authors:                 Theodoros Ioannou
 * Description:             The main program's header file
 */

#ifndef SERVERSTATES_H_
#define SERVERSTATES_H_
#include "hashlink.h"
#include "arraylist.h"


typedef struct mcell_t MCell;
struct mcell_t {
                int cid;
                int lac;
                int mnc;
                int mcc;
};




typedef struct client_t Client;
struct client_t {
                //pthread_self()
                pthread_t threadId;
                char * ip;
                int port;

                float latitude;
                float longitude;
                float accuracy;
                MCell *mycell;
                Arraylist ncells;

                struct h *resultSet;

};



void initialization(int sock, struct sockaddr_in *client);
void sendKNearest(int sock, struct sockaddr_in client, int
topK);
void printClient(Client *c);
int readPlayBackFile( Arraylist *topKList);
char* convertToUpperCase(char *str);

#endif /* SERVERSTATES_H_ */
```

MAKEFILE

```
all: proximityServer

proximityServer: serverStates.o hashlink.o proximityServer.o
queue.o serverFunctions.o arraylist.o fheap.o
proximityAlgorithm.o
                gcc serverStates.o hashlink.o proximityServer.o
queue.o serverFunctions.o arraylist.o fheap.o
proximityAlgorithm.o -o proximityServer -pthread -lm

serverStates.o: serverStates.c serverStates.h serverFunctions.h
hashlink.h queue.h arraylist.h fheap.h proximityAlgorithm.h
                gcc -c serverStates.c -pthread

hashlink.o: hashlink.c hashlink.h
                gcc -c hashlink.c

proximityServer.o: proximityServer.c proximityServer.h
serverStates.h serverFunctions.h hashlink.h queue.h arraylist.h
                gcc -c proximityServer.c -pthread

queue.o: queue.c queue.h
                gcc -c queue.c

serverFunctions.o: serverFunctions.c hashlink.h queue.h
serverStates.h serverFunctions.h
                gcc -c serverFunctions.c -pthread

arraylist.o: arraylist.c arraylist.h proximityServer.h
                gcc -c arraylist.c -pthread

fheap.o: fheap.c fheap.h
                gcc -c fheap.c -pthread -lm

proximityAlgorithm.o: proximityAlgorithm.c
proximityAlgorithm.h
                gcc -c proximityAlgorithm.c -pthread -lm

clean:
                rm -rf *o proximityServer
```

Παράρτημα Β

Προγράμματα Java

- Proximity – Android
  - Manifest files

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="cs.ucy.WMClient"
    android:versionCode="1"
    android:versionName="1.0">
  <application                    android:icon="@drawable/icon"
android:label="@string/app_name">
  <uses-library android:name="com.google.android.maps" />
    <activity class=".WMClient"
                    android:name="WMClient"
                    android:label="@string/app_name"
          android:screenOrientation="portrait">
       <intent-filter>
          <action android:name="android.intent.action.MAIN" />
          <category
android:name="android.intent.category.LAUNCHER" />
       </intent-filter>
    </activity>
    <activity
          android:name="Preferences"
          android:label="Preferences">
                  </activity>
                  <activity
                          class=".SendMessages"
                          android:name="SendMessages"
                          android:label="SendMessages"

          android:screenOrientation="portrait">
                  </activity>

  </application>

<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION"
></uses-permission>
<uses-permission
android:name="android.permission.INTERNET" />
<uses-permission
android:name="android.permission.CONTROL_LOCATION_UP
DATES"></uses-permission>
<uses-permission
android:name="android.permission.ACCESS_LOCATION_EXTR
A_COMMANDS"></uses-permission>

<uses-permission
android:name="android.permission.ACCESS_WIFI_STATE"></u
ses-permission>
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STAT
E"></uses-permission>
<uses-permission
android:name="android.permission.CHANGE_NETWORK_STAT
E"></uses-permission>
<uses-permission
android:name="android.permission.CHANGE_WIFI_STATE"></
uses-permission>



</manifest>
```

  - Source code files

```
package cs.ucy.WMClient;

import java.io.BufferedReader;
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.Socket;
import java.util.ArrayList;
import java.util.List;

import com.google.android.maps.GeoPoint;


import android.location.Location;
import android.os.Message;
```

```
import android.telephony.NeighboringCellInfo;
import android.telephony.gsm.GsmCellLocation;

class Neighbor{

	public String Address;
	public int Port;

	public int CellId;
	public int Lac;
	public int Mcc;
	public int Mnc;

	public GeoPoint pos;

	Neighbor(String Addr, int p, int c, int l, int mc, int mn){
		Address=new String(Addr);
		Port=p;
		CellId=c;
		Lac=l;
		Mcc=mc;
		Mnc=mn;

		pos = new GeoPoint(0, 0);
	}

}

public class Connect {
	private String serverAddress = new String();
	private int serverPort;
	private Socket connection = null;

	private DataOutputStream out;
	private BufferedReader in;
	private InputStream inStream = null;
	private OutputStream outStream = null;

	private GsmCellLocation location;

	Connect(){
		serverAddress = WMClient.preferences.getString("serverAdd", "n/a");
		if (serverAddress.equals("n/a") || serverAddress.equals("")) {
			Message msg = new Message();
			msg.arg1 = WMClient.ADDRINV;

			WMClient.messageHandler.sendMessage(msg);
			return;
		}

		serverPort = Integer.parseInt(WMClient.preferences.getString("serverPort", "0"));
		if (serverPort==0) {
			Message msg = new Message();
			msg.arg1 = WMClient.ADDRINV;

			WMClient.messageHandler.sendMessage(msg);
			return;
		}

	}

	static public void Send(String msg, OutputStream out) throws IOException {
		out.write((msg + WMClient.CRLF).getBytes());
		out.flush();
	}

	boolean connectToServer(){

		String buf = new String();

		try {
			// CONNECT TO SERVER
			connection = new Socket(serverAddress, serverPort);
			// INITIALIZE THE IN AND OUT STREAMS
			inStream = connection.getInputStream();
			in = new BufferedReader(new InputStreamReader(inStream));
			outStream = connection.getOutputStream();
```

```
                    out = new DataOutputStream(outStream);
                } catch (IOException e) {
                    return false;
                }

                try {
                    Message msg = new Message();


                    buf = in.readLine();

                    if (!buf.startsWith("+OK")) {
                        msg = new Message();
                        msg.arg1 = WMClient.TOASTIT;
                        msg.obj="ERROR ON READY";

                WMClient.messageHandler.sendMessage(msg);
                        return false;
                    }

                    location = (GsmCellLocation) WMClient.telephonymanager.getCellLocation();
                    //
                if (location==null) return false;

                    List<NeighboringCellInfo> NCells=WMClient.telephonymanager.getNeighboringCellInfo();
                    //
                if (NCells==null) return false;


                    //buf consist of : port|latitude|Longitude|Accuracy|CID|LAC|MCC|MNC|nCells*
                    buf = "";

                    Location myLoc = WMClient.positionOverlay.getLocation();
                    if (myLoc==null) return false;

                    //TODO: Put an actual port that the phone will be listening
                    buf+="50000";
                    buf += "|"+String.valueOf(myLoc.getLatitude());
                    buf += "|"+String.valueOf(myLoc.getLongitude());
                    //TODO: Change with "intended" error
                    buf += "|"+String.valueOf(myLoc.getAccuracy());


                    if (WMClient.playBack==true)
                    {
                        buf+="|"+WMClient.positionOverlay.getCid();
                        buf+="|"+WMClient.positionOverlay.getLac();
                    }
                    else if (location!=null)
                    {
                        buf+="|"+location.getCid();

                        buf+="|"+location.getLac();
                    }
                    else
                    {
                            buf+="|99999";

                            buf+="|999";
                    }

                    String networkOperator = WMClient.telephonymanager.getNetworkOperator();

                    if (WMClient.playBack==true)
                    {
                        buf+="|"+WMClient.positionOverlay.getMcc();
                        buf+="|"+WMClient.positionOverlay.getMnc();
                    }
                    else if (networkOperator != null && networkOperator.length() > 0) {
                            try {
```

```
                                        int mcc =
Integer.parseInt(networkOperator.substring(0, 3));
                                        int mnc =
Integer.parseInt(networkOperator.substring(3));


            buf+="|"+mcc;

            buf+="|"+mnc;

                                    }           catch
(NumberFormatException e) {
                                    }
                                }
                                else
                                {
                                    buf+="|999";
                                    buf+="|99";
                                }

                                if (WMClient.playBack==true)
                                {
                                    ArrayList<String>
temp1=WMClient.positionOverlay.getNcids();

                                    for (String temp2:
temp1)
                                    {
            buf+="|"+temp2;
                                    }
                                }
                                else if (NCells!=null)
                                {
                                    for
(NeighboringCellInfo temp2: NCells)
                                    {
            buf+="|"+temp2.getCid();
                                    }
                                }
                                Send(buf, out);



                                buf = in.readLine();


                                if (!buf.startsWith("+OK")) {
                                    msg = new
Message();
                                    msg.arg1 =
WMClient.TOASTIT;

                                    msg.obj="ERROR
ON READY";

                WMClient.messageHandler.sendMessage(msg);
                                    return false;
                                }



                    } catch (IOException e) {
                        return false;
                    }
                    return true;

                }

                public ArrayList<Neighbor> getNeighbors() {
                    String buf = new String();

                    ArrayList<Neighbor> neighbors= new
ArrayList<Neighbor>();
                    neighbors.clear();
                    //          Message msg = new
Message();
                    //          msg = new
Message();
                    //          msg.arg1 =
WMClient.TOASTIT;
                    //
                    //          msg.obj="BEFORE
TOP K";
                    //
                WMClient.messageHandler.sendMessage(msg);
                    //          Thread.sleep(2000);

                    try {


                        buf = in.readLine();
                        if (buf.equals(""))
                        {
                            closeConnection();
                            return null;
```

```
                }

                while(!buf.contains("+OK")){
                        String[] temp =
buf.split("\\|");
                        if (temp==null)
                        {
                                Message
msg;
                                msg =
new Message();
                                msg.arg1
= WMClient.TOASTIT;

        msg.obj="SPLIT ERROR";

        WMClient.messageHandler.sendMessage(msg);
                                return
null;
                        }
                        //
                buf.trim();
                        //
                Message msg = new Message();
                        //
                msg = new Message();
                        //
                msg.arg1 =
WMClient.DISPRSVMSG;
                        //
                        //
                msg.obj=" IP:"+buf.substring(0,
11)+"\n"+temp[1];
                        //

        WMClient.messageHandler.sendMessage(msg);

                        //address, port,
cellId, Lac, Mcc, Mnc
                        Neighbor temp2 =
new Neighbor(
                                temp[1],

                                Integer.parseInt(temp[2]),

                                Integer.parseInt(temp[3]),

                                Integer.parseInt(temp[4]),

                                Integer.parseInt(temp[5]),

                                Integer.parseInt(temp[6])
                                        );
                        neighbors.add(temp2);
                                        buf="";

                                        buf = in.readLine();
                                        if (buf.equals(""))
                                        {
                        closeConnection();
                                                return
null;
                                        }
                                }

                                        buf="+OK Nearest Neighbors
Received";
                                        Send(buf, out);

                                        return neighbors;

                        } catch (IOException e) {
                                // TODO Auto-generated catch
block
                                e.printStackTrace();
                        }
                        return null;
                }
                public void closeConnection() {
                        // TODO Auto-generated method stub
                        try {
                                connection.close();
                                Message msg = new Message();
                                msg.arg1 =
WMClient.REQTOPKDIS;
```

```
            WMClient.messageHandler.sendMessage(msg);
                    } catch (IOException e) {
                            Message msg = new Message();
                            msg.arg1 =
WMClient.CONEXEPTION;

            WMClient.messageHandler.sendMessage(msg);
                    }
            }
}

package cs.ucy.WMClient;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.lang.Thread.UncaughtExceptionHandler;
import java.util.ArrayList;

import android.os.Message;

public class GoogleServerInteract extends Thread {

            public GoogleServerInteract() {
                    // TODO Auto-generated constructor stub

                    this.setUncaughtExceptionHandler(new
UncaughtExceptionHandler() {

                            public void
uncaughtException(Thread thread, Throwable ex) {
                                    // TODO Auto-
generated method stub

                                    File outFile = null;
                                    outFile = new
File("sdcard/WMErrorsUncaught.txt");
                                    BufferedWriter
writer;
                                    try {
                                            writer =
new BufferedWriter(new FileWriter(outFile,true));



            writer.append(ex.getMessage()+"\n");
            writer.close();
                    } catch (IOException
e1) {
                            // TODO
Auto-generated catch block

            e1.printStackTrace();
                    }

                    return;
            }
        });

    }



    public void run() {

            WMClient.requestNeighborLocations();

            Message msg;
            if (WMClient.reqTopK==true)
            {
                    msg = new Message();
                    msg.arg1 =
WMClient.TOASTIT;
                    msg.obj=new String("New
Neighbors available!");
            WMClient.messageHandler.sendMessage(msg);
                    }

            if (WMClient.reqTopK==true)
            {
                    msg = new Message();
                    msg.arg1 =
WMClient.DRAWTOPK;

            WMClient.messageHandler.sendMessage(msg);
                    }

                    //
            WMClient.drawNeighPoints();
            }


}
```

```
package cs.ucy.WMClient;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.util.Random;

import android.content.Context;
import android.os.Message;

public class MsgRetrieval extends Thread {

//	private String serverAddress = new String();


	BufferedReader reader = null;
	private File inFile = null;
	private String custFilePath = new String();

	MsgRetrieval(Context c) {
		this.setUncaughtExceptionHandler(new
UncaughtExceptionHandler() {

			public void
uncaughtException(Thread thread, Throwable ex) {
				Message msg = new
Message();
				msg.arg1 =
WMClient.ERROR;

	WMClient.messageHandler.sendMessage(msg);
			}
		});
	}

	public void run() {


		// open the messages file
		try {
			custFilePath="";
			custFilePath =
WMClient.preferences.getString("inpFile", "n/a");
			if (custFilePath.equals("n/a") ||
custFilePath.equals("")) {
				Message msg = new
Message();
				msg.arg1 =
WMClient.DEFFILE;
	WMClient.messageHandler.sendMessage(msg);

				inFile = new
File("sdcard/teoMsgs.txt");
			} else {
				inFile = new
File(custFilePath);
			}

			reader = new
BufferedReader(new FileReader(inFile));

		} catch (Exception e) {
			Message msg = new Message();
			msg.arg1 =
WMClient.FILENOTFOUND;

	WMClient.messageHandler.sendMessage(msg);
		}

		String line = new String();
		try {
			Random rand = new Random();

			while ((line = reader.readLine())
!= null) {
				if
(WMClient.IN_PROGRESS==true){
//					Message
msg = new Message();
//					msg.arg1
= WMClient.DISPRSVMSG;
//
	msg.obj=(Object) line;
//
	WMClient.messageHandler.sendMessage(msg);

	Thread.sleep((rand.nextInt(20 - 2 + 1) + 2)*1000);
				}
				else
				{
					break;

				}
			}

		} catch (Exception e) {
			Message msg = new Message();
```

```
                msg.arg1 =
WMClient.FILEEXEPTION;

        WMClient.messageHandler.sendMessage(msg);
                }

                return;
        }

}

package cs.ucy.WMClient;

import java.util.ArrayList;

import android.graphics.drawable.Drawable;

import com.google.android.maps.ItemizedOverlay;
import com.google.android.maps.OverlayItem;

public class MyItemizedOverlay extends ItemizedOverlay {


        private ArrayList<OverlayItem> mOverlays = new
ArrayList<OverlayItem>();

        public MyItemizedOverlay(Drawable defaultMarker)
{

        super(boundCenterBottom(defaultMarker));
                // TODO Auto-generated constructor stub
        }

        @Override
        protected OverlayItem createItem(int i) {
          return mOverlays.get(i);
        }

        @Override
        public int size() {
                // TODO Auto-generated method stub
                return mOverlays.size();
        }

        public void addOverlay(OverlayItem overlay) {
           mOverlays.add(overlay);
           populate();
        }

        public void removeAllItems() {
//                      for ()
                        mOverlays.clear();
//                      mOverlays.add(overlay);
                   populate();
                }

}

package cs.ucy.WMClient;

import java.util.ArrayList;

import android.graphics.Canvas;
import android.graphics.Paint;
import android.graphics.Paint.Style;
import android.graphics.Point;
import android.graphics.drawable.Drawable;
import android.location.Location;
import android.os.Message;

import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import com.google.android.maps.Projection;

public class MyPositionOverlay extends Overlay { //extends
ItemizedOverlay<OverlayItem> {


        private Paint accuracyPaint;
        private Drawable drawable;
        private Drawable centerDrawable;
        private int width;
        private int height;
        private Point center;
        private Point left;
        private GeoPoint lastKnownPoint;
        private Location lastKnownLocation;

        private String Cid;
        private String Lac;
        private String Mcc;
        private String Mnc;
        private ArrayList<String> Ncids;


        public MyPositionOverlay(Drawable defaultMarker){
```

```
                super();
                this.centerDrawable = defaultMarker;
        }

        public void setCellData(String cid, String lac, String mcc, String mnc, ArrayList<String> ncids ) {

                this.Cid=cid;
                this.Lac=lac;
                this.Mcc=mcc;
                this.Mnc=mnc;
                this.Ncids=ncids;

        }

        public Location getLocation() {
                return lastKnownLocation;
        }
        public void setLocation(Location location) {
                this.lastKnownLocation = location;
                lastKnownPoint= new GeoPoint((int) (location.getLatitude() * 1E6), (int) (location.getLongitude() * 1E6));
        }

//      @Override
//      public boolean onTap(GeoPoint point, MapView mapView) {
//              Message msg =new Message();
//              msg.arg1=WMClient.TOASTIT;
//              if (lastKnownLocation!=null)
//              msg.obj="POS: "+lastKnownLocation.toString();
//              else
//                      msg.obj="NO POS";
//
//
        WMClient.messageHandler.sendMessage(msg);
//
//
//              return false;
//      }

        @Override
        public void draw(Canvas canvas, MapView mapView, boolean shadow) {
                if (shadow == false) {
                        if (lastKnownLocation!=null){
//                              // Get the current location
//                              Double latitude = location.getLatitude()*1E6;
//                              Double longitude = location.getLongitude()*1E6;
//                              GeoPoint geoPoint;
//                              geoPoint = new GeoPoint(latitude.intValue(),longitude.intValue());
//
//                              // Convert the location to screen pixels
//                              Point point = new Point();
//                              projection.toPixels(geoPoint, point);
//
//                              RectF oval = new RectF(point.x - mRadius, point.y - mRadius,
//                                                      point.x + mRadius, point.y + mRadius);
//
//                              // Setup the paint
//                              Paint paint = new Paint();
//                              paint.setARGB(250, 255, 255, 255);
//                              paint.setAntiAlias(true);
//                              paint.setFakeBoldText(true);
//
//                              Paint backPaint = new Paint();
//                              backPaint.setARGB(175, 50, 50, 50);
//                              backPaint.setAntiAlias(true);
//
//                              RectF backRect = new RectF(point.x + 2 + mRadius,
//                                                      point.y - 3*mRadius,
//                                                      point.x + 65, point.y + mRadius);
//
//                              // Draw the marker
//                              canvas.drawOval(oval, paint);
//
                                canvas.drawRoundRect(backRect, 5, 5, backPaint);
//                              canvas.drawText("Here I Am",
//                                                      point.x + 2*mRadius, point.y,
//                                                      paint);

                                drawMyLocation(canvas, mapView, lastKnownLocation, lastKnownPoint, 0);
```

```
                }

            }
            super.draw(canvas, mapView, shadow);
        }


        protected void drawMyLocation(Canvas canvas,
MapView mapView, Location lastFix, GeoPoint myLoc,
                long when) {

            accuracyPaint = new Paint();
            accuracyPaint.setAntiAlias(true);
            accuracyPaint.setStrokeWidth(2.0f);

//          Resources res = WMClient.getResources();
            drawable = centerDrawable;
            width = drawable.getIntrinsicWidth();
            height = drawable.getIntrinsicHeight();
            center = new Point();
            left = new Point();

            Projection projection = mapView.getProjection();

            double latitude = lastFix.getLatitude();
            double longitude = lastFix.getLongitude();
            float accuracy = lastFix.getAccuracy();

            float[] result = new float[1];

            Location.distanceBetween(latitude, longitude,
latitude, longitude + 1, result);
            float longitudeLineDistance = result[0];

            GeoPoint leftGeo = new GeoPoint((int) (latitude *
1e6), (int) ((longitude - accuracy
                    / longitudeLineDistance) * 1e6));
            projection.toPixels(leftGeo, left);
            projection.toPixels(myLoc, center);
            int radius = center.x - left.x;

            accuracyPaint.setColor(0xff6666ff);
            accuracyPaint.setStyle(Style.STROKE);
            canvas.drawCircle(center.x, center.y, radius,
accuracyPaint);

//          accuracyPaint.setColor(0x186666ff);
            accuracyPaint.setARGB(69, 16, 78, 139);
            accuracyPaint.setStyle(Style.FILL);
            canvas.drawCircle(center.x, center.y, radius,
accuracyPaint);

            drawable.setBounds(center.x - width / 2, center.y -
height / 2, center.x + width / 2, center.y + height / 2);
            drawable.draw(canvas);
        }


        public GeoPoint getLocationGeo() {
            if (lastKnownPoint!=null)
                return lastKnownPoint;
            return null;
        }

        public String getCid() {
            return Cid;
        }

        public String getLac() {
            return Lac;
        }

        public String getMcc() {
            return Mcc;
        }

        public String getMnc() {
            return Mnc;
        }

        public ArrayList<String> getNcids() {
            return Ncids;
        }
}
package cs.ucy.WMClient;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.util.ArrayList;
import java.util.Random;

import com.google.android.maps.GeoPoint;

import android.content.Context;
import android.os.Bundle;
import android.os.Message;
```

```
public class PositionPlaybackRetrieval extends Thread {

//		private String serverAddress = new String();



		BufferedReader reader = null;
		private File inFile = null;
		private String custFilePath = new String();

		PositionPlaybackRetrieval(Context c) {
			this.setUncaughtExceptionHandler(new
UncaughtExceptionHandler() {

				public void
uncaughtException(Thread thread, Throwable ex) {
					Message msg = new
Message();
					msg.arg1 =
WMClient.ERROR;

		WMClient.messageHandler.sendMessage(msg);
				}
			});
		}

		public void run() {



			// open the messages file
			try {
				custFilePath="";
				custFilePath =
WMClient.preferences.getString("posPlayBackFile", "n/a");
				if (custFilePath.equals("n/a") ||
custFilePath.equals("")) {
					Message msg = new
Message();
					msg.arg1 =
WMClient.DEFFILE;

		WMClient.messageHandler.sendMessage(msg);
					inFile = new
File("sdcard/positionsPlaybackFile.txt");
				} else {
					inFile = new
File(custFilePath);
				}
```

```
					reader = new
BufferedReader(new FileReader(inFile));

				} catch (Exception e) {
					Message msg = new Message();
					msg.arg1 =
WMClient.FILENOTFOUND;

		WMClient.messageHandler.sendMessage(msg);
				}

				String line = new String();
				try {
					Random rand = new Random();

					while ((line = reader.readLine())
!= null) {
						if
(line.startsWith("#")) continue;
						if
(line.equalsIgnoreCase("")) continue;
						if
(WMClient.playBack==true){
							Message
msg = new Message();
							msg.arg1
= WMClient.NEWINCOMINGPOS;

							//LINE =
LATITUDE|LONGITUDE|ACCURACY|LAC|MCC|MNC|CID|
NCIDS
							String[]
temp = line.split("\\|");

							Bundle
data = new Bundle();

		data.putDouble("latitude",
Double.parseDouble(temp[0]));

		data.putDouble("longitude",
Double.parseDouble(temp[1]));

		data.putDouble("accuracy",
Double.parseDouble(temp[2]));

		data.putString("lac", temp[3]);

		data.putString("mcc", temp[4]);
```

```
                data.putString("mnc", temp[5]);
                data.putString("cid", temp[6]);


                ArrayList<String> nCids= new ArrayList<String>();
                for (int i=7;i<temp.length;i++)
                {
                nCids.add(temp[i]);
                }
                data.putStringArrayList("ncids", nCids);

                msg.setData(data);

                WMClient.messageHandler.sendMessage(msg);

                Thread.sleep((rand.nextInt(20 - 5 + 1) + 5)*1000);
                        }
                        else
                        {
                                break;

                        }
                }
                reader.close();

        } catch (Exception e) {
                Message msg = new Message();
                msg.arg1 = WMClient.FILEEXEPTION;

                WMClient.messageHandler.sendMessage(msg);
        }

        return;
```

```
        }

}

package cs.ucy.WMClient;

import android.os.Bundle;
import android.preference.PreferenceActivity;

public class Preferences extends PreferenceActivity {

//      ListPreference test;

        /** Called when the activity is first created. */
        @Override
        public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          addPreferencesFromResource(R.xml.preferences);

//        test.setOnPreferenceClickListener(new OnPreferenceClickListener(){
//
//              public boolean onPreferenceClick(Preference preference) {
//                      startActivity(new Intent(Settings.ACTION_WIFI_SETTINGS));
//
//                      return false;
//              }
//
//
//      });
//
//      test = (Button) findViewById(R.id.appClose);
//
//      btnRetrieveMsgs.setOnClickListener(new ImageButton.OnClickListener() {
//
//              public void onClick(View arg0)
//              {
//
//                      if (IN_PROGRESS == false) {
//
//      IN_PROGRESS = true;
//
//      btnRetrieveMsgs.setEnabled(false);
```

```
//
		btnStopRetr.setEnabled(true);
//
//
		Toast.makeText(WMClient.this,
//
				"Now retrieving Messages",
//
				Toast.LENGTH_LONG).show();
//
//
new MsgRetrieval(getApplicationContext());
//
		msgRtr.start();
//
//
//
//
		});
	}
}

package cs.ucy.WMClient;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;

import android.os.Message;

public class ProximityServerInteract extends Thread {

	public ProximityServerInteract() {
		// TODO Auto-generated constructor stub

		this.setUncaughtExceptionHandler(new
UncaughtExceptionHandler() {

			public void
uncaughtException(Thread thread, Throwable ex) {
//
generated method stub

				File outFile = null;
				outFile = new
File("sdcard/WMErrorsUncaught.txt");
				BufferedWriter
writer;
				try {
```

```
writer =
new BufferedWriter(new FileWriter(outFile,true));

				writer.append(ex.getMessage()+"\n");

				writer.close();
				} catch (IOException
e1) {
//
Auto-generated catch block

				e1.printStackTrace();
				}

				return;
			}
		});

		}

		public void run() {

			Connect con = new Connect();
			Boolean conSuccess=false;
			while(WMClient.reqTopK==true){

		if(conSuccess=con.connectToServer()){

				//Collect the results
from the server

		ArrayList<Neighbor> topK= con.getNeighbors();
				// Check and use the
results
				Message msg = new
Message();
				msg = new
Message();
				msg.arg1 =
WMClient.PLACETOPKONMAP;

				msg.obj=topK;

		WMClient.messageHandler.sendMessage(msg);
			}
```

```
                                else
                                {
                                        Message msg = new Message();
                                        msg = new Message();
                                        msg.arg1 = WMClient.ADDRINV;
                WMClient.messageHandler.sendMessage(msg);
                                        break;
                                }

                                try {
                                        sleep(13000);
//              this.getResources().getInteger(R.string.WAIT_MAX);
//              Integer.getInteger(R.string.WAIT_MAX);
//                                      sleep(1000*(20 + (int)(Math.random() * ((40 - 20) + 1))));
                                } catch (InterruptedException e) {
                                        // TODO Auto-generated catch block
                                        e.printStackTrace();
                                }

                        }
                        if (conSuccess)
                                con.closeConnection();
                        else
                        {
                                Message msg = new Message();
                                msg.arg1 = WMClient.REQTOPKDIS;
                WMClient.messageHandler.sendMessage(msg);

                        }
                }

}
```

```
package cs.ucy.WMClient;

import android.app.Activity;
import android.inputmethodservice.KeyboardView.OnKeyboardActionListener;
import android.os.Bundle;
import android.text.method.ScrollingMovementMethod;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

public class SendMessages extends Activity {

        private TextView tvOldMsgs;
        private Button btnSendMsg;
        private EditText etNewMsg;
        private EditText etNewMsgEdit;

        private Toast toast;

        /** Called when the activity is first created. */
        @Override
        public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.sendmsgs);

          initControls(this);

          toast = Toast.makeText(SendMessages.this, "", Toast.LENGTH_LONG);
                View textView = toast.getView();
                LinearLayout lay = new LinearLayout(SendMessages.this);
        lay.setOrientation(LinearLayout.HORIZONTAL);
                lay.addView(textView);
                toast.setView(lay);
                toast.setText("Here you can compose and send messages...");
                toast.show();


        }
```

```
	private void initControls(SendMessages a) {
//			tvOldMsgs=			(TextView)
findViewById(R.id.tvOldMsgs);
//			tvOldMsgs.setMovementMethod(new
ScrollingMovementMethod());
//			btnSendMsg		=		(Button)
findViewById(R.id.sendMsg);
//			etNewMsg		=		(EditText)
findViewById(R.id.etNewMsg);
			etNewMsgEdit		=		(EditText)
findViewById(R.id.etNewMsgEdit);

			etNewMsgEdit.bringToFront();

//			preferences				=
PreferenceManager.getDefaultSharedPreferences(a);


//			btnSendMsg.setOnClickListener(new
Button.OnClickListener() {
//
//				public void onClick(View arg0)
{
//
//
//
//				}
//			});

	}

}

package cs.ucy.WMClient;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.util.ArrayList;
import java.util.List;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONException;
import org.json.JSONObject;

import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import com.google.android.maps.OverlayItem;


import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.graphics.drawable.Drawable;
import android.location.Criteria;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.net.ConnectivityManager;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.preference.PreferenceManager;
import android.provider.Settings;
import android.telephony.TelephonyManager;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.ImageButton;
import android.widget.LinearLayout;
import android.widget.Toast;

public class WMClient extends MapActivity{


		//Constants for the JSON Request
		public static final String OPERATOR = "operator";
		public static final String CARRIER = "carrier";
		//		private static final String TAG =
"Wapdroid";
		private static final String VERSION = "version";
```

```
	private static final String GMAPS_VERSION =
"1.1.0";
	private static final String HOST = "host";
	private static final String GMAPS =
"maps.google.com";
	//	private static final String
ACCESS_TOKEN = "access_token";
	//	private static final String HOME_MCC =
"home_mobile_country_code";
	//	private static final String HOME_MNC =
"home_mobile_network_code";
	private static final String CELL_ID = "cell_id";
	private static final String LAC =
"location_area_code";
	private static final String MCC =
"mobile_country_code";
	private static final String MNC =
"mobile_network_code";
	private static final String CELLT = "cell_towers";
	//	private static final String LAT = "latitude";
	//	private static final String LONG =
"longitude";
	//	private static final String WIFIT =
"wifi_towers";
	//	private static final String MAC =
"mac_address";
	//	private static final String SS =
"signal_strength";
	private static final String RTYPE = "radio_type";
	private static final String RADD = "request_address";
	private static final String ADDLANG =
"address_language";



	public volatile static boolean IN_PROGRESS = false;
	public volatile static SharedPreferences preferences;
	public volatile static boolean reqTopK=false;
	public volatile static boolean playBack=false;

	//	private TextView tvMsgs;
	private ImageButton btnRetrieveMsgs;
	private ImageButton btnStopRetr;
	private ImageButton btnSendMsg;
	private ImageButton btnClose;
	private ImageButton btnMyGoToLocation;
	private static MapView myMapView;



	static List<Overlay> mapOverlays;
	Drawable drawable;
	static MyItemizedOverlay itemizedOverlay;

	private ProximityServerInteract srvInteraction;
	private GoogleServerInteract gooInteraction;

	private MsgRetrieval msgRtr = null;

	public volatile static LocationManager
locationManager = null;
	public volatile static MyLocationListener
locationListener = null;
	public volatile static Handler messageHandler;
	private Toast toast;


	public volatile static TelephonyManager
telephonymanager = null;

	//private LocationManager locationManager = null;
	//private LocationListener locationListener = null;
	//	private File inFile = null;
	//private Socket connection = null;
	BufferedWriter writer = null;
	BufferedReader reader = null;

	//private String serverAddress = new String();
	//private String distance = new String();
	//	private String custFilePath = new String();
	static MapController mapController;
	public volatile static MyPositionOverlay
positionOverlay = null;


	public volatile static ArrayList<Neighbor> topK;


	//Constants
	public static final int CONN = 110;
	public static final int ADDRINV = 111;
	public static final int REQTOPKDIS = 112;
	public static final int NEWINCOMINGPOS = 113;
	public static final int ERROR = 118;
	public static final int FILENOTFOUND = 119;
	public static final int FILEEXEPTION = 120;
	public static final int DEFFILE = 121;
	public static final int CONEXEPTION = 122;
	public static final int TOASTIT = 123;
	public static final int PLACETOPKONMAP = 124;
	public static final int DRAWTOPK = 125;
```

```
public static final String CRLF = "\r\n";

/** Called when the activity is first created. */
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    myMapView=
(MapView)findViewById(R.id.myMapView);
    mapController =
myMapView.getController();

    myMapView.setSatellite(true);

myMapView.setBuiltInZoomControls(true);

    drawable =
this.getResources().getDrawable(R.drawable.pin2);
    itemizedOverlay = new
MyItemizedOverlay(drawable);

    locationListener = new
MyLocationListener();

    initControls(this);

    telephonymanager = (TelephonyManager)
getSystemService(Context.TELEPHONY_SERVICE);

    topK = new ArrayList<Neighbor>();
    toast = Toast.makeText(WMClient.this, "",
Toast.LENGTH_LONG);
    View textView = toast.getView();
    LinearLayout lay = new
LinearLayout(WMClient.this);

lay.setOrientation(LinearLayout.HORIZONTAL);
    // ImageView view =
new ImageView(WMClient.this);
    //
view.setImageResource(R.drawable.warning);
    // lay.addView(view);
    lay.addView(textView);
    toast.setView(lay);

        messageHandler = new Handler() {

    @SuppressWarnings("unchecked")
        public void
handleMessage(Message msg) {

    super.handleMessage(msg);
            switch (msg.arg1) {
            case CONN:

    toast.setText("Connecting...");
    toast.show();
                break;
            case ADDRINV:

    toast.setText("Invalid Server Address/Port...");

    toast.show();
                break;
            case
FILENOTFOUND:

    toast.setText("Error when opening file...");

    toast.show();
                break;
            case
FILEEXEPTION:

    toast.setText("Exception while reading from file...");

    toast.show();
                break;
            case DEFFILE:

    toast.setText("Using default file...");
                break;
            case REQTOPKDIS:

    reqTopK=false;
                break;
            case DRAWTOPK:

    drawNeighPoints();
                break;
            case TOASTIT:
```

```
                                                    //
            Toast.makeText(WMClient.this,
                                                    //
                    (String)(msg.obj.toString()),
                                                    //
                    Toast.LENGTH_LONG).show();
            toast.setText(""+(String)(msg.obj.toString()));
            toast.show();
                                                    break;
                                            case
PLACETOPKONMAP:

            topK.clear();
                                                    topK =
(ArrayList<Neighbor>) msg.obj;
            if(topK==null)
                                                    {
            toast.setText("NULL!!");
            toast.show();
            break;
                                                    }
                                                    if
(!topK.isEmpty())
                                                    {
//
            toast.setText("NEWWW!! "+topK.size());
//
            toast.show();
            gooInteraction = new GoogleServerInteract();
            gooInteraction.start();
                                                    }
                                                    //
            toast.setText("Top K Neighbors received.
Processing...");
                                                    //
            toast.show();
                                                    //
            requestNeighborLocations();
                                                    //
            toast.setText("Top K Neighbors processed.
Drawing...");
                                                    //
            toast.show();
                                                    //
            drawNeighPoints();
                                                    //
            toast.setText("Top K Neighbors been displayed!");
                                                    //
            toast.show();
                                                    break;
                                            case
CONEXEPTION:
            toast.setText("Error with the server connection...");
            toast.show();
                                                    break;
                                            case
NEWINCOMINGPOS:
                                                    Bundle
data=msg.getData();
            //
                    34.915004,33.601491
                                                    Location
loc=new Location("PLAYBACK");
                    loc.setAccuracy((float) data.getDouble("accuracy"));
                    loc.setLatitude(data.getDouble("latitude"));
                    loc.setLongitude(data.getDouble("longitude"));
                    positionOverlay.setLocation(loc);

                    positionOverlay.setCellData(
```

```
                    data.getString("cid"),
                    data.getString("lac"),
                    data.getString("mcc"),
                    data.getString("mnc"),
                    data.getStringArrayList("ncids"));
//
            mapController.setZoom(12);
            mapController.animateTo(new
GeoPoint((int)(loc.getLatitude()*1E6),
(int)(loc.getLongitude()*1E6)));

                                                    break;
                                            default:
            toast.setText("Unknown error occurred.");
            toast.show();
                                                    break;
                                            }
                                    }
                    };

                    //animate to the current position
                    mapController.setZoom(8);

//                  positionOverlay = new
MyPositionOverlay();
//
            myMapView.getOverlays().add(positionOverlay);
                    //
            positionOverlay.enableMyLocation();
                    //TODO: enable my location on listener

                    //
            positionOverlay.runOnFirstFix(new Runnable(){
                    //
                    //                              public
void run() {
                    //
                    //
            Location location=positionOverlay.getLocation();
                    //
            Double geoLat = location.getLatitude()*1E6;
                    //
            Double geoLng = location.getLongitude()*1E6;
                    //
            GeoPoint point = new GeoPoint(geoLat.intValue(),
                    //
                            geoLng.intValue());
                    //
                    //
            mapController.setZoom(12);
                    //
            mapController.animateTo(point);
                    //


                    ////
            try {
                    ////
                    Thread.sleep(1000);
                    ////
            } catch (InterruptedException e) {
                    ////
                    // TODO Auto-generated catch block
                    ////
                    e.printStackTrace();
                    ////
            }
                    ////
            point=                                  new
GeoPoint((int)(34.892126*1E6),(int)(33.614731*1E6));
                    ////
            positionOverlay.
                    ////
            mapController.animateTo(point);
                    //


                    //                              }});



            btnRetrieveMsgs.setVisibility(View.INVISIBLE);
                    //
            btnClose.setVisibility(View.INVISIBLE);
                    //
            btnSendMsg.setVisibility(View.INVISIBLE);

            btnStopRetr.setVisibility(View.INVISIBLE);
```

```
	}

	public void onResume(Bundle savedInstanceState) {
		super.onResume();
		initLocationManager();
	}

	public void onPause(Bundle savedInstanceState) {
		super.onPause();
		//
	positionOverlay.disableMyLocation();

	locationManager.removeUpdates(locationListener);

		reqTopK=false;
		finish();
		System.exit(0);
	}

	@Override
	protected void onDestroy() {
		// TODO Auto-generated method stub
		super.onDestroy();
		reqTopK=false;
		//
	positionOverlay.disableMyLocation();

	locationManager.removeUpdates(locationListener);

		finish();
		System.exit(0);
	}


	@Override
	public boolean onPrepareOptionsMenu(Menu menu) {

		menu.clear();
		if (reqTopK)
			menu.add(0,
R.id.stopRetrieveNei,                                    0,
"Disconnect").setIcon(R.drawable.disser1);
		else
			menu.add(0, R.id.retrieveNei, 0,
"Connect").setIcon(R.drawable.conser);
			menu.add(0,          R.id.appPref,          1,
"Settings").setIcon(R.drawable.prefer);

			if (!playBack)
				menu.add(0,
R.id.playbackMode,                2,                "Playback
Mode").setIcon(R.drawable.playback);
			else
				menu.add(0,    R.id.activeMode,
2, "Active Mode").setIcon(R.drawable.active_mode);

			return
super.onPrepareOptionsMenu(menu);
		}


		@Override
		public boolean onCreateOptionsMenu(Menu menu) {
			if (reqTopK==false)
				menu.add(0, R.id.retrieveNei, 0,
"Connect").setIcon(R.drawable.conser);
			//                else
			//
			menu.add(0,          R.id.stopRetrieveNei,          0,
"Disconnect").setIcon(R.drawable.disser);
			//                menu.add(0,
R.id.SmrtTrc, 0, "Smart Trace").setIcon(R.drawable.smart);
			//                menu.add(0,
R.id.OnOff, 0, "Online/Offline").setIcon(R.drawable.offgps);
			//                menu.add(0,
R.id.mapMode, 0, "Map Mode").setIcon(R.drawable.mode);
			menu.add(0,          R.id.appPref,          1,
"Settings").setIcon(R.drawable.prefer);
			//                menu.add(0,
R.id.more, 2, "More ...").setIcon(R.drawable.more);
			//                menu.add(0,
R.id.stopRetrieveNei,                                    0,
"Disconnect").setIcon(R.drawable.disser);
			return true;
		}


		// This method is called once the menu is selected
		public    boolean    onOptionsItemSelected(MenuItem
item) {
			Intent i;
			switch (item.getItemId()) {
			// We have only one menu option

			case R.id.appPref:
```

```
                    // Launch Preference activity
                    i = new Intent(WMClient.this,
Preferences.class);
                    startActivity(i);
                    // A toast is a view containing a
quick little message for the user.
                    Toast.makeText(WMClient.this,
                                "Here are
the application's preferences.",

        Toast.LENGTH_LONG).show();
                    break;
                case R.id.retrieveNei:
                    final ConnectivityManager
connMgr = (ConnectivityManager)
this.getSystemService(Context.CONNECTIVITY_SERVICE);
                    final android.net.NetworkInfo
wifi =
connMgr.getNetworkInfo(ConnectivityManager.TYPE_WIFI);

                    if (wifi.isAvailable())
                    {
                        srvInteraction = new
ProximityServerInteract();
                        srvInteraction.start();
                        reqTopK = true;
                    }
                    else
                    {

                        final
AlertDialog.Builder builder = new AlertDialog.Builder(this);

        builder.setMessage("Your WiFi seems to be disabled,
do you want to enable it?")

        .setCancelable(false).setPositiveButton("Yes",

        new DialogInterface.OnClickListener() {
                            public
void onClick(DialogInterface dialog, int which) {

        startActivity(new
Intent(Settings.ACTION_WIFI_SETTINGS));

                            }

        }).setNegativeButton("No",
                        new
DialogInterface.OnClickListener() {
                            public
void onClick(DialogInterface dialog, int which) {
                dialog.cancel();
                            }
        });
        final AlertDialog
alert = builder.create();
        alert.show();

        //
Toast.makeText(WMClient.this,
        //
                "Wifi not
available.",
        //
Toast.LENGTH_LONG).show();
        break;
                    }

                    break;
            case R.id.stopRetrieveNei:
                    reqTopK = false;
                    break;
            case R.id.activeMode:
                    playBack = false;

                    initLocationManager();

                    break;
            case R.id.playbackMode:
                    playBack = true;

                locationManager.removeUpdates(locationListener);

                    PositionPlaybackRetrieval
pPlayBack = new
PositionPlaybackRetrieval(getApplicationContext());
                    pPlayBack.start();

                    break;

            }
            return true;
        }
```

```
public static String requestLocationJSON(Neighbor x)
throws IOException {
        //                int cid, int lac, int
mcc, int mnc
        InputStream is = null;
        try {
                StringBuilder uri = new
StringBuilder("https://www.google.com/loc/json");
                HttpPost post = new
HttpPost(uri.toString());
                post.setEntity(new
StringEntity(makeQueryString( x.CellId, x.Lac, x.Mnc,
x.Mcc)));
                post.setHeader("Accept",
"application/json");
                post.setHeader("Content-type",
"application/json");

                // Send the HttpGet request
                HttpClient httpClient = new
DefaultHttpClient();
                // HttpResponse response =
httpClient.execute(request);
                HttpResponse response =
httpClient.execute(post);

                // Check the response status
                int status =
response.getStatusLine().getStatusCode();
                if (status !=
HttpURLConnection.HTTP_OK) {
                        switch (status) {
                        case
HttpURLConnection.HTTP_NO_CONTENT:
                                return
("The cell " + x.CellId + " could not be found in the database");
                        case
HttpURLConnection.HTTP_NOT_FOUND:
                                return
("The url doesn't exist");
                        case
HttpURLConnection.HTTP_BAD_REQUEST:
                                return
("Check if some parameter is missing or misspelled");
                        case
HttpURLConnection.HTTP_UNAUTHORIZED:
                                return
("Make sure the API key is present and valid");
                        case
HttpURLConnection.HTTP_FORBIDDEN:
                                return
("You have reached the limit for the number of requests per
day.\nThe maximum number of requests per day is "
                                        + "currently 500.");
                        default:
                                return
("HTTP response code: " + status);
                        }
                }
                // The response was ok
(HTTP_OK) so lets read the data
                HttpEntity entity =
response.getEntity();

                // Creates a new InputStream
object of the entity.
                is = entity.getContent();

                StringBuffer b = new
StringBuffer();
                // read all the data from the
input stream and generate JSON object
                int car;
                while ((car = is.read()) != -1) {
                        b.append((char) car);
                }


                if (b != null) {
                        try {
                                // Get the
JSONObject value associated with a key.
                        JSONObject location = new JSONObject(new
String(b.toString()))

                        .getJSONObject("location");

                        x.pos=new
GeoPoint((int)(location.getDouble("latitude")*1E6),
(int)(location.getDouble("longitude")*1E6)) ;


                        } catch
(JSONException e) {
```

```
            e.printStackTrace();
                                    } catch (Exception e)
{
            e.printStackTrace();
                                    }
                            }
                    } catch (MalformedURLException e) {
                            Log.e("ERROR",
e.getMessage());
                            // if there is a problem with the
url then we should close is
                    } catch (IllegalArgumentException e) {
                            throw new IOException("URL
was incorrect. Did you forget to set the API_KEY?");
                    } finally {
                            // make sure we clean up
                            try {
                                    if (is != null)
                                            is.close();
                            } catch (Exception e) {
                            }
                    }
                    return "Result is Updated";
            }


            private static String makeQueryString(int cid, int lac,
int mnc, int mcc) {

                    return "{\""
                    + VERSION + "\": \""
                    + GMAPS_VERSION + "\",\""
                    + HOST + "\": \""
                    + GMAPS + "\", \""
                    + RTYPE + "\": \"gsm\",  \""
                    + CARRIER + "\": \"Vodafone\",  \""
                    + RADD + "\": true,  \""
                    + ADDLANG + "\": \"en_GB\",\""
                    + CELLT + "\": [{\""
                    + CELL_ID + "\": \""+cid+"\",\""
                    + LAC + "\": "+lac+",\""
                    + MCC + "\": "+mcc+",\""
                    + MNC + "\": "+mnc+", \"AGE\": 0 " + "}"
+"]"+"}" ;
            }


            public static void requestNeighborLocations(){

                    // Loop to call JSON with
                    // cid, lac, mnc, mcc
                    File outFile = null;
                    outFile = new
File("sdcard/WMErrors.txt");
                    BufferedWriter writer;
                    try {
                            writer = new
BufferedWriter(new FileWriter(outFile,true));

                            for (Neighbor x: topK){
                                    try {

                    writer.append("\n"+x.CellId+":");

                    writer.append(

                            requestLocationJSON(x)+"\n");

                    writer.append("==> POS("+x.pos.getLatitudeE6()+",
"+x.pos.getLongitudeE6()+");\n");
                                    } catch (IOException
e) {
                                            // TODO
Auto-generated catch block

                    e.printStackTrace();
                                    }
                            }
                            writer.close();
                    } catch (IOException e1) {
                            // TODO Auto-generated catch
block
                            e1.printStackTrace();
                    }

            }


            public static void drawNeighPoints(){

                    mapOverlays =
myMapView.getOverlays();
                    ArrayList<GeoPoint> myList=new
ArrayList<GeoPoint>();
                    itemizedOverlay.removeAllItems();

                    for (Neighbor x: topK){
```

```
                    if (x.pos.getLatitudeE6()!=0 && x.pos.getLongitudeE6()!=0)
                        myList.add(new GeoPoint(x.pos.getLatitudeE6(), x.pos.getLongitudeE6()));

                }
                //Calculate my position as well with the displayed nodes as well
                // GeoPoint myLoc =WMClient.positionOverlay.getMyLocation();

                GeoPoint myLoc =WMClient.positionOverlay.getLocationGeo();

                int maxLatitude= Math.max(Integer.MIN_VALUE, myLoc.getLatitudeE6());
                double minLatitude=Math.min(Integer.MAX_VALUE, myLoc.getLatitudeE6());
                double maxLongitude=Math.max(Integer.MIN_VALUE, myLoc.getLongitudeE6());
                double minLongitude=Math.min(Integer.MAX_VALUE, myLoc.getLongitudeE6());

                for(GeoPoint y:myList)
                {

                    OverlayItem overlayitem = new OverlayItem(y, "", "");

            itemizedOverlay.addOverlay(overlayitem);

                    maxLatitude =Math.max(maxLatitude, y.getLatitudeE6());
                    minLatitude =Math.min(minLatitude, y.getLatitudeE6());
                    maxLongitude =Math.max(maxLongitude, y.getLongitudeE6());
                    minLongitude =Math.min(minLongitude, y.getLongitudeE6());

                }

                mapOverlays.add(itemizedOverlay);

                maxLatitude = maxLatitude + (int)((maxLatitude-minLatitude) * 0.1);
                minLatitude = minLatitude - (int)((maxLatitude-minLatitude) * 0.1);

                maxLongitude = maxLongitude + (int)((maxLongitude-minLongitude) * 0.1);
                minLongitude = minLongitude - (int)((maxLongitude-minLongitude) * 0.1);

                mapController.zoomToSpan((int)Math.abs(maxLatitude - minLatitude), (int)Math.abs(maxLongitude - minLongitude));
                mapController.animateTo(new GeoPoint((int)((maxLatitude + minLatitude) / 2), (int)((maxLongitude + minLongitude) / 2)));
            }

            private void initControls(WMClient a) {
                // tvMsgs = (TextView) findViewById(R.id.tvMsgs);
                //
                tvMsgs.setMovementMethod(new ScrollingMovementMethod());
                btnRetrieveMsgs = (ImageButton) findViewById(R.id.retrieveMsgs);
                btnStopRetr = (ImageButton) findViewById(R.id.stopRetr);
                btnSendMsg = (ImageButton) findViewById(R.id.sendNewMsg);
                btnClose = (ImageButton) findViewById(R.id.appClose);
                btnMyGoToLocation =(ImageButton) findViewById(R.id.myGoToLocation);
                preferences =PreferenceManager.getDefaultSharedPreferences(a);

                positionOverlay = new MyPositionOverlay(this.getResources().getDrawable(R.drawable.bullet_blue));
                List<Overlay> overlays =myMapView.getOverlays();
                overlays.add(positionOverlay);

                initLocationManager();

                btnRetrieveMsgs.setOnClickListener(new ImageButton.OnClickListener() {
```

```
                public void onClick(View arg0)
{
                    if (IN_PROGRESS
== false) {

        IN_PROGRESS = true;

        btnRetrieveMsgs.setEnabled(false);

        btnStopRetr.setEnabled(true);


        Toast.makeText(WMClient.this,

                "Now retrieving Messages",

                Toast.LENGTH_LONG).show();

                    msgRtr=
new MsgRetrieval(getApplicationContext());

        msgRtr.start();

                    }
                }
            });

            btnStopRetr.setOnClickListener(new
ImageButton.OnClickListener() {
                public void onClick(View arg0)
{

                    if (IN_PROGRESS
== true) {

        IN_PROGRESS = false;

        btnRetrieveMsgs.setEnabled(true);

        btnStopRetr.setEnabled(false);

        msgRtr=null;

                    }
                }
            });

            btnSendMsg.setOnClickListener(new
ImageButton.OnClickListener() {
                public void onClick(View v) {
                    Intent i = null;
                    i = new
Intent(WMClient.this, SendMessages.class);
                    startActivity(i);
                }
            });

            btnClose.setOnClickListener(new
ImageButton.OnClickListener() {
                public void onClick(View v) {
                    appClose();
                }
            });


        btnMyGoToLocation.setImageResource(R.drawable.l
ocation);


        btnMyGoToLocation.setOnClickListener(new
ImageButton.OnClickListener() {
                public void onClick(View v) {

                    GeoPoint myLoc
=WMClient.positionOverlay.getLocationGeo();
                    if(myLoc!=null)
                    {
        //
        mapController.setZoom(12);

        mapController.animateTo(myLoc);

                    }
                }
            });


        }

        private void initLocationManager() {

            String context =
Context.LOCATION_SERVICE;
            locationManager =
(LocationManager)getSystemService(context);

            Criteria criteria = new Criteria();

        criteria.setAccuracy(Criteria.ACCURACY_FINE);
```

```
                criteria.setAltitudeRequired(false);
                criteria.setBearingRequired(false);
                criteria.setCostAllowed(true);
        criteria.setPowerRequirement(Criteria.NO_REQUIRE
MENT);
                String provider =
locationManager.getBestProvider(criteria, true);

        positionOverlay.setLocation(locationManager.getLast
KnownLocation(provider));

                mapController.setZoom(12);

        mapController.animateTo(positionOverlay.getLocatio
nGeo());

//
        locationManager.requestLocationUpdates(provider,
this.getResources().getInteger(R.string.GPS_INTERVAL),
this.getResources().getInteger(R.string.GPS_DISTANCE),
locationListener);

        locationManager.requestLocationUpdates(provider,
60000, 15, locationListener);
        }
        private void appClose() {
                finish();
                //
        positionOverlay.disableMyLocation();

        locationManager.removeUpdates(locationListener);
                reqTopK=false;
                System.exit(0);
        }

        @Override
        protected boolean isRouteDisplayed() {
                return false;
        }
        private class MyLocationListener implements
LocationListener {

                public void onLocationChanged(Location
loc) {
                        if (loc != null) {
                                try {
                positionOverlay.setLocation(loc);
                                        Double
geoLat = loc.getLatitude()*1E6;
                                        Double
geoLng = loc.getLongitude()*1E6;
                                        GeoPoint
point = new GeoPoint(geoLat.intValue(),
                                                geoLng.intValue());

                mapController.setZoom(12);

                mapController.animateTo(point);

                                } catch (Exception e)
{

                toast.setText("Exception in onLocationChanged(): " +
e.getMessage());

                toast.show();
                                }
                        }
                }
                public void onProviderDisabled(String
provider) {
                }
                public void onProviderEnabled(String
provider) {
                }
                public void onStatusChanged(String
provider, int status, Bundle extras) {
                }
        }
}
```

- XML Files

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout
xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
            >
 <com.google.android.maps.MapView
  android:id="@+id/myMapView"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:enabled="true"
  android:clickable="true"

android:apiKey="0ShgnRVu8vaCE0hsY1wNJcrDwFVX8yTJrm1tmGg"
 />
            <!--
            Debug Maps Key:
  0ShgnRVu8vaCE0hsY1wNJcrDwFVX8yTJrm1tmGg
  Other:
  0EEkzHi9atze-owoLv4ZSsQMXKUE_GGqLOi_v4g
  0EEkzHi9atzcweHAhrw5Qsenwr405wa1jS1wDuQ
            -->-->

            <!--
  android:orientation="vertical"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  >
  -->

<!--<TextView android:id="@+id/tvMsgs"
            android:layout_marginLeft="30px"
            android:layout_marginRight="30px"
            android:layout_width="316px"
            android:layout_height="150px"
            android:maxLines="20"
            android:textColor="#ff0000"
            android:scrollbars="vertical"
            android:textSize="18sp"
            android:e
/>
--><ImageButton
            android:id="@+id/retrieveMsgs"
            android:layout_width="153px"
            android:layout_height="43px"
            android:layout_x="2px"
            android:layout_y="337px"
>
</ImageButton>
<ImageButton
            android:id="@+id/stopRetr"
            android:layout_width="153px"
            android:layout_height="48px"
            android:layout_x="2px"
            android:layout_y="382px"
            android:enabled="false"
>
</ImageButton>
<ImageButton
            android:id="@+id/sendNewMsg"
            android:layout_width="46px"
            android:layout_height="47px"
            android:background="@drawable/new_msg"
            android:layout_x="268px"
            android:layout_y="380px"
>
</ImageButton>
<ImageButton
            android:id="@+id/appClose"
            android:layout_width="46px"
            android:layout_height="46px"
            android:background="@drawable/close_app"
            android:layout_x="268px"
            android:layout_y="430px"
>
</ImageButton>

<ImageButton
            android:id="@+id/myGoToLocation"
            android:layout_width="48px"
            android:layout_height="48px"
            android:layout_x="8px"
            android:layout_y="430px"
>
</ImageButton>
</AbsoluteLayout>
```

- Twitter Crawler

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

import twitter4j.ResponseList;
import twitter4j.Status;
import twitter4j.Twitter;
import twitter4j.TwitterException;
import twitter4j.TwitterFactory;
```

```
public class msgCrawlerNoAuth {

	/**
	 * @param args
	 */
	public static void main(String[] args) {
		//
	http://api.twitter.com/oauth/authorize?oauth_token=lyzCYXH5RbzNXINS1EohICU9DUv67hinoHapsGSV4c
		//		PIN 5192802
		//		161696130
		//		token : 161696130-TB7vtNTYOw212a4Yei2JsoehrIsUBTN2sWiIESb8
		//		tokenSecret : cSIj018rtgQUVpr8RNSYfmgQCVVrkAUp6nnvxEUDPk
		//		ConsumerKey VAVOgCOBw94TOpwHyrewjA
		//		ConsumerSecret 92y2B2rJfGRWm04KwcbdPtFcUzYdNTU38Jl1WYp0U4

		//		USAGE:

		//		Twitter myTw;
		//		myTw = new TwitterFactory().getInstance();
		//
	myTw.setOAuthConsumer(consumerKey, consumerSecret);
		//		AccessToken accessToken = new AccessToken(twitterToken, twitterTokenSecret);
		//
	myTw.setOAuthAccessToken(accessToken);


		TwitterFactory twFctr = new TwitterFactory();
		Twitter myTw;
//		AccessToken accessToken = new AccessToken("161696130-TB7vtNTYOw212a4Yei2JsoehrIsUBTN2sWiIESb8", "cSIj018rtgQUVpr8RNSYfmgQCVVrkAUp6nnvxEUDPk");

	myTw=twFctr.getOAuthAuthorizedInstance("VAVOgCOBw94TOpwHyrewjA",
	"92y2B2rJfGRWm04KwcbdPtFcUzYdNTU38Jl1WYp0U4");

		/* //used to get the OAuth
		Twitter myTw=twFctr.getOAuthAuthorizedInstance("VAVOgCOBw94TOpwHyrewjA",
	"92y2B2rJfGRWm04KwcbdPtFcUzYdNTU38Jl1WYp0U4");

		RequestToken requestToken = null;
		try {
			requestToken = myTw.getOAuthRequestToken();
		} catch (TwitterException e) {
			e.printStackTrace();
		}

	AccessToken accessToken = null;
	BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
	while (null == accessToken) {
	 System.out.println("Open the following URL and grant access to your account:");

	System.out.println(requestToken.getAuthorizationURL());
	 System.out.print("Enter the PIN(if aviailable) or just hit enter.[PIN]:");
	String pin = null;
			try {
				pin = br.readLine();
			} catch (IOException e) {
				e.printStackTrace();
			}
	try{


	  if(pin.length() > 0){
	   accessToken = myTw.getOAuthAccessToken(requestToken, pin);
	  }else{
	   accessToken = myTw.getOAuthAccessToken();
	  }
	 } catch (TwitterException te) {
	  if(401 == te.getStatusCode()){
	   System.out.println("Unable to get the access token.");
	  }else{
	   te.printStackTrace();
	  }
	 }
	}
```

```
                try {
                        System.out.println(myTw.verifyCredentials().getId());
                } catch (TwitterException e) {
                        e.printStackTrace();
                }
                System.out.println("token : " + accessToken.getToken());
                System.out.println("tokenSecret : " + accessToken.getTokenSecret());
                */
                BufferedWriter out=null;

                try {
                        //This was for the first level of followers
                        //
                int basicID=myTw.showUser("CNN").getId();
                        //

                        //

                        //
                //loop
                        //
                //basicID=read from file num <- "num"
                        //
                IDs temp= myTw.getFollowersIDs(basicID,-1);


                        try {
                                out = new BufferedWriter(new FileWriter("msgsLevel1.txt",true));
                        } catch (IOException e2) {
                                e2.printStackTrace();
                        }

                        BufferedReader in = new BufferedReader(new FileReader("level1.txt"));
                        String str;

                        //Go to a specific user ID to continue from there
                //      while ((str=in.readLine())!=null)
                //      {
                //              if (str.contains("187656106"))
                //                      break;
                //      }
                        int levelCounter=0;
                        while ((str = in.readLine()) != null &&levelCounter<5000) {
                                String [] myLine=str.split(" <- ");

                                int basicID=Integer.parseInt(myLine[1].toString());

                ResponseList<Status> temp = null;
                                try {
                                        temp = myTw.getUserTimeline(basicID);
                                } catch(TwitterException e2) {

                                System.out.println("ERROR");
                                        if (401==e2.getStatusCode()){ //not auth to access this data

                                e2.printStackTrace();

                                continue;
                                        }

                                if(e2.exceededRateLimitation()==true)
                                        {

                                try {

                                        out.flush();

                                        Thread.sleep(3600*1000);

                                        temp = myTw.getUserTimeline(basicID);

                                } catch (TwitterException e3) {

                                        System.out.println("ERROR");

                                        if (401==e3.getStatusCode()){ //not auth to access this data

                                                e3.printStackTrace();

                                                continue;

                                        }
```

```
                                                    }

            } catch (InterruptedException e1) { //during sleep      import java.io.BufferedReader;
                                                                     import java.io.BufferedWriter;
                        e1.printStackTrace();                        import java.io.FileReader;
                                                                     import java.io.FileWriter;
                        continue;                                    import java.io.IOException;

            }                                                        import twitter4j.ResponseList;
                                                    }                import twitter4j.Status;
                                        }                            import twitter4j.Twitter;
                                                                     import twitter4j.TwitterException;
                                        if(temp==null)               import twitter4j.TwitterFactory;
                                                    continue;        import twitter4j.http.AccessToken;

            System.out.println("======>>>BasicID= "+basicID);

                                                                     public class msgCrawler {

                                        for (Status status :                 /**
temp) {                                                                       * @param args
                                                                              */
            out.write(status.getCreatedAt()+"||"+status.getUser().g          public static void main(String[] args) {
etId() + "||" +                                                                       //
                                                                                 http://api.twitter.com/oauth/authorize?oauth_token=ly
status.getText()+"||"+status.getGeoLocation()+"||"+status.getPlace               zCYXH5RbzNXINS1EohICU9DUv67hinoHapsGSV4c
()                                                                               //                  PIN 5192802
                                    +"||"+status.getSource()+"\n");              //                  161696130
                                                                                 //                  token : 161696130-
                                                                                 TB7vtNTYOw212a4Yei2JsoehrIsUBTN2sWiIESb8
                                                                                 //                  tokenSecret :
                        }                                                        cSIj018rtgQUVpr8RNSYfmgQCVVrkAUp6nnvxEUDPk
                                                                                 //                  ConsumerKey
                                                                                 VAVOgCOBw94TOpwHyrewjA
                                        levelCounter++;                          //                  ConsumerSecret
                                                                                 92y2B2rJfGRWm04KwcbdPtFcUzYdNTU38Jl1WYp0U4
            System.out.println("********      TOTAL      USERS:
"+levelCounter);                                                                 //                  USAGE:
                                    }
                                                                                 //                  Twitter myTw;
                                    in.close();                                  //                  myTw      =      new
                                    out.close();                                 TwitterFactory().getInstance();
                                                                                 //
                                                                                       myTw.setOAuthConsumer(consumerKey,
                                                                                 consumerSecret);
                        } catch (IOException e) {                                //                  AccessToken
                                    e.printStackTrace();                         accessToken      =      new      AccessToken(twitterToken,
                                                                                 twitterTokenSecret);
                        }                                                        //
            }                                                                          myTw.setOAuthAccessToken(accessToken);
```

```
                TwitterFactory twFctr = new TwitterFactory();
                Twitter myTw;
                AccessToken accessToken = new AccessToken("161696130-TB7vtNTYOw212a4Yei2JsoehrIsUBTN2sWiIESb8", "cSIj018rtgQUVpr8RNSYfmgQCVVrkAUp6nnvxEUDPk");

                myTw=twFctr.getOAuthAuthorizedInstance("VAVOgCOBw94TOpwHyrewjA", "92y2B2rJfGRWm04KwcbdPtFcUzYdNTU38Jl1WYp0U4", accessToken);



                /* //used to get the OAuth
                Twitter myTw= twFctr.getOAuthAuthorizedInstance("VAVOgCOBw94TOpwHyrewjA", "92y2B2rJfGRWm04KwcbdPtFcUzYdNTU38Jl1WYp0U4");

                RequestToken requestToken = null;
                try {
                        requestToken =myTw.getOAuthRequestToken();
                } catch (TwitterException e) {
                        e.printStackTrace();
                }



        AccessToken accessToken = null;
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        while (null == accessToken) {
          System.out.println("Open the following URL and grant access to your account:");

System.out.println(requestToken.getAuthorizationURL());
          System.out.print("Enter the PIN(if aviailable) or just hit enter.[PIN]:");
          String pin = null;
                try {
                        pin = br.readLine();
                } catch (IOException e) {
                        e.printStackTrace();
                }
        try{


          if(pin.length() > 0){
                accessToken = myTw.getOAuthAccessToken(requestToken, pin);
          }else{
                accessToken = myTw.getOAuthAccessToken();
          }
        } catch (TwitterException te) {
          if(401 == te.getStatusCode()){
            System.out.println("Unable to get the access token.");
          }else{
            te.printStackTrace();
          }
        }
      }

        try {

      System.out.println(myTw.verifyCredentials().getId());
                } catch (TwitterException e) {
                        e.printStackTrace();
                }
            System.out.println("token : " + accessToken.getToken());
            System.out.println("tokenSecret : " + accessToken.getTokenSecret());
                 */
                BufferedWriter out=null;


                try {
                        //                This was for the first level of followers
                        //
            int basicID=myTw.showUser("CNN").getId();
                        //

                        //

                        //
            //loop
                        //
            //basicID=read from file num <- "num"
                        //
            IDs temp= myTw.getFollowersIDs(basicID,-1);


                        try {
                                out = new BufferedWriter(new FileWriter("msgsLevel1.txt",true));
                        } catch (IOException e2) {
```

```
                    e2.printStackTrace();
                }

                BufferedReader in = new
BufferedReader(new FileReader("level1.txt"));
                String str;

                //Go to a specific user ID to
continue from there
//              while ((str=in.readLine())!=null)
//              {
//                  if
(str.contains("187656106"))
//                      break;
//              }



                int levelCounter=0;
                while ((str = in.readLine()) !=
null &&levelCounter<5000) {
                    String [] myLine
=str.split(" <- ");


                    int
basicID=Integer.parseInt(myLine[1].toString());

        ResponseList<Status> temp = null;
                    try {
                        temp =
myTw.getUserTimeline(basicID);
                    } catch
(TwitterException e2) {

        System.out.println("ERROR");
                        if
(401==e2.getStatusCode()){ //not auth to access this data

        e2.printStackTrace();

        continue;
                        }

        if(e2.exceededRateLimitation()==true)
                        {

        try {

                out.flush();
                        Thread.sleep(3600*1000);

                        temp = myTw.getUserTimeline(basicID);

                    } catch (TwitterException e3) {

                        System.out.println("ERROR");

                        if (401==e3.getStatusCode()){ //not auth to
access this data

                            e3.printStackTrace();

                            continue;


                        }

                    } catch (InterruptedException e1) { //during sleep

                        e1.printStackTrace();

                        continue;

                    }
                                                    }
                                                }

                                                if(temp==null)
                                                    continue;
                    System.out.println("======>>>BasicID= "+basicID);

                                                for (Status status :
temp) {

                        out.write(status.getCreatedAt()+"||"+status.getUser().g
etId() + "||" +
                                status.getText()+"\n");

                        System.out.println(status.getCreatedAt()+"||"+status.ge
tUser().getId() + "||" +
                                status.getText());


                    }
```

```
                                        levelCounter++;
            System.out.println("******** TOTAL USERS:
"+levelCounter);
                                }

                                in.close();
                                out.close();



                        } catch (IOException e) {
                                e.printStackTrace();

                        }
                }

}

/*
Copyright (c) 2007-2010, Yusuke Yamamoto
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:
  * Redistributions of source code must retain the above copyright
    notice, this list of conditions and the following disclaimer.
  * Redistributions in binary form must reproduce the above copyright
    notice, this list of conditions and the following disclaimer in the
    documentation and/or other materials provided with the distribution.
  * Neither the name of the Yusuke Yamamoto nor the
    names of its contributors may be used to endorse or promote products
    derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY Yusuke Yamamoto ``AS IS'' AND ANY
EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL Yusuke Yamamoto BE LIABLE FOR ANY
DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*/


import twitter4j.Status;
import twitter4j.Twitter;
import twitter4j.TwitterException;
import twitter4j.TwitterFactory;

import java.util.List;

/**
* Example application that gets public, user and friend timeline using specified account.<br>
* Usage: java twitter4j.examples.GetTimelines ID Password
* @author Yusuke Yamamoto - yusuke at mac.com
*/
public final class GetTimelines {
  /**
   * Usage: java twitter4j.examples.GetTimelines ID Password
   * @param args String[]
   */
  public static void main(String[] args) {

    Twitter unauthenticatedTwitter = new TwitterFactory().getInstance();
    System.out.println("Showing public timeline.");
    try {
      List<Status> statuses = unauthenticatedTwitter.getPublicTimeline();
      for (Status status : statuses) {
        System.out.println(status.getUser().getName() + ":" +
                           status.getText());
      }
      if (args.length < 2) {
        System.out.println(
```

```
            "You need to specify TwitterID/Password
combination to show UserTimelines.");
            System.out.println(
              "Usage: java twitter4j.examples.GetTimelines ID
Password");
            System.exit(0);
        }

        // Other methods require authentication
        Twitter twitter = new
TwitterFactory().getInstance(args[0], args[1]);
        statuses = twitter.getFriendsTimeline();
        System.out.println("-------------------------------");
        System.out.println("Showing " + args[0] + "'s friends
timeline.");
        for (Status status : statuses) {
            System.out.println(status.getUser().getName() + ":" +
                    status.getText());
        }
        statuses = twitter.getUserTimeline();
        System.out.println("-------------------------------");
        System.out.println("Showing " + args[0] + "'s timeline.");
        for (Status status : statuses) {
            System.out.println(status.getUser().getName() + ":" +
                    status.getText());
        }
        Status status = twitter.showStatus(81642112l);
        System.out.println("-------------------------------");
        System.out.println("Showing " +
status.getUser().getName() +
                "'s status updated at " + status.getCreatedAt());
        System.out.println(status.getText());
        System.exit(0);
      } catch (TwitterException te) {
        System.out.println("Failed to get timeline: " +
te.getMessage());
        System.exit( -1);
      }
    }
}

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

import twitter4j.IDs;
import twitter4j.Twitter;
import twitter4j.TwitterException;
import twitter4j.TwitterFactory;
import twitter4j.http.AccessToken;


public class crawlerTest {

        /**
         * @param args
         */
        public static void main(String[] args) {
                TwitterFactory twFctr = new
TwitterFactory();
                Twitter myTw;
                AccessToken accessToken = new
AccessToken("161696130-
TB7vtNTYOw212a4Yei2JsoehrIsUBTN2sWiIESb8",
"cSIj018rtgQUVpr8RNSYfmgQCVVrkAUp6nnvxEUDPk");

                myTw=twFctr.getOAuthAuthorizedInstance("VAVOg
COBw94TOpwHyrewjA",
"92y2B2rJfGRWm04KwcbdPtFcUzYdNTU38Jl1WYp0U4",
accessToken);

                BufferedWriter out=null;

                try {
                        //                Initial
user's followers
                        //
        int basicID=myTw.showUser("CNN").getId();
                        //

                        //

                        //
        //loop
                        //
        //basicID=read from file num <- "num"
                        //
        IDs temp= myTw.getFollowersIDs(basicID,-1);



                        long sum = 0;
                        int o=0;



                        try {
```

```
                                        out = new
BufferedWriter(new FileWriter("level2.txt",true));
                                } catch (IOException e2) {
                                        e2.printStackTrace();
                                }

                                BufferedReader in = new
BufferedReader(new FileReader("level1.txt"));
                                String str;
                                //Go to a specific user ID to
continue from there
//                              while ((str=in.readLine())!=null)
//                              {
//                                      if
(str.contains("187656106"))
//                                              break;
//                              }



                                int levelCounter=0;
                                while ((str = in.readLine()) !=
null &&levelCounter<5000) {
                                        String [] myLine
=str.split(" <- ");


                                        int
basicID=Integer.parseInt(myLine[1].toString());
                                        IDs temp = null;
                                        try {
                                                temp =
myTw.getFollowersIDs(basicID,-1);
                                        }               catch
(TwitterException e2) {

        System.out.println("ERROR");
                                                if
(401==e2.getStatusCode()){

        e2.printStackTrace();

        continue;
                                                }

        if(e2.exceededRateLimitation()==true)
                                                {

        try {
                                                        out.flush();

                                                        Thread.sleep(3600*1000);

                                                        temp= myTw.getFollowersIDs(basicID,-1);

                                        } catch (TwitterException e3) {

                                                        System.out.println("ERROR");

                                                        if (401==e3.getStatusCode()){

                                                                e3.printStackTrace();

                                                                continue;

                                                        }

                                        } catch (InterruptedException e1) {

                                                        e1.printStackTrace();

                                                        continue;

                                        }
                                                                        }
                                                                }

                                                                if(temp==null)
                                                                        continue;
                                        System.out.println("======>>>BasicID= "+basicID);

                                                                int[] x;

                                                                do
                                                                {


                                                        x=temp.getIDs();

                                                        sum+=x.length;

                                                        System.out.println("\nTOTAL                 "+(++o)+"
:"+x.length+"\n");
                                                                for     (int
i=0;i<x.length;i++) {
```

```
                out.write(basicID+" <- "+x[i]+"\n");
                                        }

                                        if
(x.length==5000)
                                        {

                try{

                        temp=
myTw.getFollowersIDs(basicID,temp.getNextCursor());

                }

                catch (TwitterException e) {

                        if(e.exceededRateLimitation()==true)

                        {

                                try {

                                        out.flush();

                Thread.sleep(3600*1000);

                                        temp=
myTw.getFollowersIDs(basicID,temp.getNextCursor());

                                } catch (InterruptedException
e1) {

                                        e1.printStackTrace();

                                        break;

                                }

                        }

                }

                                        }

                                        while
(x.length==5000);
                                        //loop
                                        levelCounter++;
                                }

                                in.close();
                                out.close();


                        } catch (TwitterException e) {
                                e.printStackTrace();

                        } catch (IOException e) {
                                e.printStackTrace();

                        }
                }
}
```

- GSM3G

```
package GSM3G.apk;

import java.io.BufferedWriter;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.util.List;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONException;
import org.json.JSONObject;

import android.app.Activity;
import android.content.Context;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.telephony.NeighboringCellInfo;
import android.telephony.TelephonyManager;
```

```
import android.telephony.gsm.GsmCellLocation;
import android.text.method.ScrollingMovementMethod;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

public class GSM3G extends Activity {

	public static final String OPERATOR = "operator";
	public static final String CARRIER = "carrier";
	private static final String TAG = "Wapdroid";
	private static final String VERSION = "version";
	private static final String GMAPS_VERSION = "1.1.0";
	private static final String HOST = "host";
	private static final String GMAPS = "maps.google.com";
	private static final String ACCESS_TOKEN = "access_token";
	private static final String HOME_MCC = "home_mobile_country_code";
	private static final String HOME_MNC = "home_mobile_network_code";
	private static final String CELL_ID = "cell_id";
	private static final String LAC = "location_area_code";
	private static final String MCC = "mobile_country_code";
	private static final String MNC = "mobile_network_code";
	private static final String CELLT = "cell_towers";
	private static final String LAT = "latitude";
	private static final String LONG = "longitude";
	private static final String WIFIT = "wifi_towers";
	private static final String MAC = "mac_address";
	private static final String SS = "signal_strength";
	private static final String RTYPE = "radio_type";
	private static final String RADD = "request_address";
	private static final String ADDLANG = "address_language";
	private TelephonyManager telephonymanager;

	private WifiManager wifimanager;

	private GsmCellLocation location;
	private int cid, lac, mcc, mnc, cellPadding;
	private Toast toast;
	private TextView TextView01;
	private TextView TextView02;
	private TextView TextView03;
	private TextView TextView04;
	private TextView TextViewLong;
	private TextView TextViewLat;
	private TextView TextViewCoun;
	private TextView TextViewCity;
	private TextView TextViewEntity;
	private Button GetPositionButton;
	private TextView TextViewStreet;
	private TextView TextViewAccuracy;

	private Button UpdateCellButton;

	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
		toast = Toast.makeText(GSM3G.this, "", Toast.LENGTH_LONG);
		View textView = toast.getView();
		LinearLayout lay = new LinearLayout(GSM3G.this);

		lay.setOrientation(LinearLayout.HORIZONTAL);
		ImageView view = new ImageView(GSM3G.this);

		view.setImageResource(R.drawable.warning);
		lay.addView(view);
		lay.addView(textView);
		toast.setView(lay);

		telephonymanager = (TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE);

		// wifimanager = (WifiManager) getSystemService(Context.WIFI_SERVICE);

		TextView01 = (TextView) findViewById(R.id.TextView01);
		TextView02 = (TextView) findViewById(R.id.TextView02);
		TextView03 = (TextView) findViewById(R.id.TextView03);
		TextView04 = (TextView) findViewById(R.id.TextView04);
```

```
                TextViewLong = (TextView)
findViewById(R.id.TextViewLong);
                TextViewLat = (TextView)
findViewById(R.id.TextViewLat);
                TextViewCoun = (TextView)
findViewById(R.id.TextViewCountry);
                TextViewCity = (TextView)
findViewById(R.id.TextViewCity);
                TextViewEntity = (TextView)
findViewById(R.id.Entity);
                TextViewStreet = (TextView)
findViewById(R.id.TextViewStreet);
                TextViewAccuracy = (TextView)
findViewById(R.id.TextViewAcc);

                /*
                 * Setup a listener for the
UpdateCellButton. Pressing this button will
                 * fetch the current cell info from the
phone.
                 */
                UpdateCellButton = (Button)
findViewById(R.id.UpdateCellButton);
                UpdateCellButton.setOnClickListener(new
View.OnClickListener() {
                        public void onClick(View v) {
                                location
(GsmCellLocation) telephonymanager.getCellLocation();

        List<NeighboringCellInfo>
NCells=telephonymanager.getNeighboringCellInfo();

                                Context context =
getApplicationContext();
                                WifiManager wfM =
(WifiManager)
context.getSystemService(Context.WIFI_SERVICE);
                                wfM.startScan();
                                try {

        Thread.sleep(3000);
                                } catch
(InterruptedException e2) {
                                        // TODO
Auto-generated catch block

        e2.printStackTrace();
                                }
                                List<ScanResult> srl
= wfM.getScanResults();
                                File outFile = null;
                                outFile = new
File("sdcard/GSM3Gout.txt");
                                BufferedWriter
writer;
                                try {
                                        writer =
new BufferedWriter(new FileWriter(outFile,true));

                                        writer.append("##############################
################");

                                        writer.append("\nCell
Info:\n+++++++++++++++\n"+"Type \tCid \tLac \tRssi
\tdBm\n");

                                        writer.append(telephonymanager.getNetworkOperator
Name()+"\t==>"+
telephonymanager.getNetworkType()+"\t"+location.getCid()+"\t"
=+location.getLac()+"\n");

                                                for
(NeighboringCellInfo temp2: NCells)
                                                {

                                        int rssi=temp2.getRssi();

                                        int dBm = -113 + 2*rssi;

                                        if (rssi!=99)

                                        writer.append(temp2.getNetworkType()+"\t"+temp2.g
etCid()+"\t"+temp2.getLac()+"\t"+rssi+"\t"+ dBm +"\n");

                                        else

                                        writer.append(temp2.getNetworkType()+"\t"+temp2.g
etCid()+"\t"+temp2.getLac()+"\tn/a\tn/a\n");
                                                }
```

```
            writer.append("\n=>WiFi Info\nSSID \tBSSID \tLEVEL \tCAPABILITIES \tFREQUENCY\n");

                                                for (ScanResult temp : srl)
                                                {

            writer.append(temp.SSID + "\t" + temp.BSSID + "\t" + temp.level + "\t" + temp.capabilities + "\t" + temp.frequency + "\n");
                                                }

            writer.close();
                                        } catch (IOException e1) {
            e1.printStackTrace();
                                        }

                                        cid = location.getCid();
                                        lac = location.getLac();
                                        //cid=7703;
                                        //lac=8721;
                                        // Mcc(Mobile Country Code ) and mnc Mobile (Network Code)is
                                        // concatenated in the networkOperatorString. The
                                        // first 3 chars is the mcc and the last 2 is the mnc.

                                        String networkOperator = telephonymanager.getNetworkOperator();
                                        if (networkOperator != null && networkOperator.length() > 0) {
                                                try {
            mcc = Integer.parseInt(networkOperator.substring(0, 3));

            mnc = Integer.parseInt(networkOperator.substring(3));

            //                                  mcc=01;
            //                                  mnc=262;
                                                } catch (NumberFormatException e) {
                                                }
                                        }

                                        // Check if the current cell is a (Universal Mobile
                                        // Telecommunications System )UMTS (3G) cell. If a 3G cell
                                        // the cell id,mcc,mcn and lac padding will be 8 numbers, if not
                                        // 4 numbers.
                                        // the padding is used to to convert cid and lac in hex
                                        if (telephonymanager.getNetworkType() == TelephonyManager.NETWORK_TYPE_UMTS) {
                                                cellPadding = 8;
                                        } else {
                                                cellPadding = 4;
                                        }

                                        // set the updated text
                                        TextView01.setText("CellID: " + getPaddedInt(cid, cellPadding));

                                        TextView02.setText("Lac: " + getPaddedInt(lac, 4));

                                        TextView03.setText("Mcc: " + getPaddedInt(mcc, 2));

                                        TextView04.setText("Mnc: " + getPaddedInt(mnc, 3));
                                                }
                                        });

                                        // Setup a listener for the GetPositionButton. When pressing this button
                                        // the cell info is sent to the server and hopefully we will get a
                                        // longitude and latitude back.

                                        GetPositionButton = (Button) findViewById(R.id.GetPositionButton);
                                        GetPositionButton.setOnClickListener(new View.OnClickListener() {
```

```
                public void onClick(View v) {
                        String strResult;
                        // Seems that cid and lac must be in hex. Cid should be padded
                        // with zero's to 8 numbers if UMTS (3G) cell, otherwise to 4
                        // numbers. Mcc padded to 3 numbers. Mnc padded to 2 numbers.
                        try {
                                // Update the current location
                                strResult = updateLocation(getPaddedInt(cid,cellPadding),
                                getPaddedInt(lac, 4), getPaddedInt(mnc, 3), getPaddedInt(mcc, 2));

                        } catch (IOException e) {
                                strResult = "Error!\n" + e.getMessage();
                        }
                        toast.setText(strResult);
                        toast.show();
                    }
                });
        }

        // Convert an integer to String And pad with zeros 0's.
        String getPaddedInt(int nr, int zeros) {
                String str = Integer.toString(nr);
                if (str != null) {
                        while (str.length() < zeros) {
                                str = "0" + str;
                        }
                }
                return str;
        }

        private String updateLocation(String cid, String lac, String mnc, String mcc)
        throws IOException {
                InputStream is = null;
                ByteArrayOutputStream bos = null;
                try {
                        StringBuilder uri = new StringBuilder("https://www.google.com/loc/json");
                        HttpPost post = new HttpPost(uri.toString());
                        post.setEntity(new StringEntity(makeQueryString( cid, lac, mnc, mcc)));
                        post.setHeader("Accept", "application/json");
                        post.setHeader("Content-type", "application/json");
                        // Send the HttpGet request
                        HttpClient httpClient = new DefaultHttpClient();
                        // HttpResponse response = httpClient.execute(request);
                        HttpResponse response = httpClient.execute(post);

                        // Check the response status
                        int status = response.getStatusLine().getStatusCode();
                        if (status != HttpURLConnection.HTTP_OK) {
                                switch (status) {
                                case HttpURLConnection.HTTP_NO_CONTENT:
                                        return ("The cell " + cid + " could not be found in the database");
                                case HttpURLConnection.HTTP_NOT_FOUND:
                                        return ("The url doesn't exist");
                                case HttpURLConnection.HTTP_BAD_REQUEST:
                                        return ("Check if some parameter is missing or misspelled");
                                case HttpURLConnection.HTTP_UNAUTHORIZED:
                                        return ("Make sure the API key is present and valid");
                                case HttpURLConnection.HTTP_FORBIDDEN:
                                        return ("You have reached the limit for the number of requests per day.\nThe maximum number of requests per day is "
                                                + "currently 500.");
                                default:
```

```
                return
("HTTP response code: " + status);
            }
        }
        // The response was ok
(HTTP_OK) so lets read the data
        HttpEntity entity =
response.getEntity();

        // Creates a new InputStream
object of the entity.
        is = entity.getContent();
        bos = new
ByteArrayOutputStream();
        byte buf[] = new byte[256];
        StringBuffer b = new
StringBuffer();
        // read all the data from the
inputstream and generate JSON object
        int car;
        while ((car = is.read()) != -1) {
            b.append((char) car);
        }

    TextViewEntity.setText(b.toString());

    TextViewEntity.setMovementMethod(new
ScrollingMovementMethod());

        if (b != null) {
            try {
                // Get the
JSONObject value associated with a key.
    JSONObject location = new JSONObject(new
String(b.toString()))

    .getJSONObject("location");

                // Set new
text to the below textViews

    TextViewLong.setText("Longitude: " +
location.getDouble("longitude"));

    TextViewLat.setText("Latitude: " +
location.getDouble("latitude"));

        TextViewAccuracy.setText("Acc: " +
location.getDouble("accuracy"));
        location =
location.getJSONObject("address");
        TextViewCity.setText("City: " +
location.getString("city"));

        TextViewCoun.setText("Country: " +
location.getString("country"));

        TextViewStreet.setText("Street: "+
location.getString("street"));

        File
outFile = null;
        outFile =
new File("sdcard/GSM3Gout.txt");
        BufferedWriter writer;
        try {

        writer = new BufferedWriter(new
FileWriter(outFile,true));

        location = new JSONObject(new
String(b.toString())).getJSONObject("location");

        writer.append("\n>>Longitude: " +
location.getDouble("longitude"));

        writer.append("\n>>Latitude: " +
location.getDouble("latitude"));

        writer.append("\n>>Accuracy: " +
location.getDouble("accuracy"));

        location = location.getJSONObject("address");

        writer.append("\n>>City: " +
location.getString("city"));

        writer.append("\n>>Country: " +
location.getString("country"));
```

```
                writer.append("\n>>Street:                    "+
location.getString("street"));



                writer.close();

                                                        }     catch
(IOException e1) {

                e1.printStackTrace();

                                                        }


                                                }                  catch
(JSONException e) {

                e.printStackTrace();

                                                } catch (Exception e)
{

                e.printStackTrace();

                                                }
                                        }
                                } catch (MalformedURLException e) {
                                        Log.e("ERROR",
e.getMessage());
                                        // if there is a problem with the
url then we should clos bos and is
                                } catch (IllegalArgumentException e) {
                                        throw new IOException("URL
was incorrect. Did you forget to set the API_KEY?");
                                } finally {
                                        // make sure we clean up
                                        try {
                                                if (is != null)
                                                        is.close();
                                        } catch (Exception e) {
                                        }
                                }
                                return "Result is Updated";
                }


                public String makeQueryString(String cid, String lac,
String mnc, String mcc) {
                                return "{\""
                                + VERSION
                                + "\": \""
                                + GMAPS_VERSION
                                + "\",\""
                                        + HOST
                                        + "\": \""
                                        + GMAPS
                                        + "\", \""
                                        //
                                + HOME_MCC
                                        //
                                + "\": 310,  \""
                                        //
                                + HOME_MNC
                                        //
                                + "\": 410, \""
                                        + RTYPE
                                        + "\": \"gsm\",  \""
                                        + CARRIER
                                        + "\": \"Vodafone\",  \""
                                        + RADD
                                        + "\": true,  \""
                                        + ADDLANG
                                        + "\": \"en_GB\",\""
                                        + CELLT
                                        + "\": [{\""
                                        + CELL_ID
                                        + "\": \""+cid+"\",\""
                                        + LAC
                                        + "\": "+lac+",\""
                                        + MCC
                                        + "\": "+mcc+",\""
                                        + MNC
                                        + "\": "+mnc+", \"AGE\": 0 "
                                        + "}"
                                        +"]"
                                        +"}" ;

                                        //\"wifi_towers\":  [  {  \"mac_address\":
\"01-23-45-67-89-ab\",      \"signal_strength\": 8,  \"age\": 0 }, {
\"mac_address\":          \"01-23-45-67-89-ac\",\"signal_strength\":
4,\"age\": 0}]}";

                                }
}
```

- Dataset Converter

```
/**
 *
 */
package proxData;
```

```
/**
 * @author amazingteo
 *
 */
public class DataTranslator {

	/**
	 * @param args
	 */
	public static void main(String[] args) {
		CellPlacement cp=new CellPlacement();


	cp.run("/Users/amazingteo/Desktop/RUNS/Oldenburg
_icde_data/data/oldenburg_2000.txt",  "cellData/Cells2000.txt",
"userData/UsersReports2000", 2000, 1);
			//
/Users/amazingteo/Desktop/RUNS/Oldenburg_icde_data/data/old
enburg_5000.txt
	}

}

/**
 *
 */
package proxData;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Map;
import java.util.Map.Entry;

import javax.swing.text.html.HTMLDocument.Iterator;

import proxData.helperClasses.*;


/**
 * @author Georgios Chatzimilioudis (gchatzim@gmail.com)
 *
 * Construct cell placement
 *
 */
public class CellPlacement {


	private static final char MANUAL = 'x';
	private static final char SPANAKIS = 's';
	private static final char LAOUDIASOLDENBURG =
'l';
	private static final char OLDENBURG	= 'o';
	private static final char LAOUDIASGEOLIFE = 'g';
	private static final String DELIMITER = "\t";

	private Integer k;
	private Integer minUserThreshold = 0;
	private HashMap<Integer, Integer> nPerEpoch;
	private HashMap<Integer, ArrayList<UserReport>>
reportsPerEpoch = null;

	/*
**********************************************************
**********************
	 * Private member variables
	 *
**************************************************
*/

	/* ******* END of private member variables
********************************** */

	/*
**********************************************************
**********************
	 * Constructors / Destructors
	 *
**************************************************
*/

	/* ******* END of constructors/destructors
********************************** */

	/*
**********************************************************
**********************
	 * Getters / Setters
	 *
**************************************************
*/

	/* ******* END of getters / setters
********************************** */
```

```
        /*
**************************************************
**********
         * Functions
         *
**************************************************
*/

        /**
         * place cells in a regular hatched grid with radius
         * grid has distance between columns and rows =
radius
         */
        public static HashSet<CircularCell> hatchedGrid(
Double radius,        CommutativePair<DoublePoint> space) {

                HashSet<CircularCell> cellSet = new
HashSet<CircularCell>();

                double spaceWidth =
space.getSecond().getX() - space.getFirst().getX();
                double spaceHeight =
space.getSecond().getY() - space.getFirst().getY();

                /* compute distances between base stations
*/
                double distBetweenColumns = radius;
                double distBetweenRows = radius;

                /* for each base station point */
                for (int x=0; x * distBetweenColumns <
spaceWidth ; x++){
                        for (int y=0; (y-1) *
distBetweenRows < spaceHeight; y++){
                                /* here you create
the hatched grid */
                                DoublePoint center =
new DoublePoint(
        ( x * distBetweenColumns ) + distBetweenColumns/4
+ ( (y % 2) * distBetweenColumns/2 ),
        y * distBetweenRows + distBetweenRows/2
                                );
                                CircularCell newCell
= new CircularCell( center, radius );
                                /* add cell to cellset
*/
                                cellSet.add( newCell
);
                        }
                }

                return cellSet;

        }

        /**
         * place m cells in a regular hatched grid
         * compute radius so that cells cover the whole space.
         */
        public static HashSet<CircularCell>
hatchedGrid(Integer m,
        CommutativePair<DoublePoint> space) {

                HashSet<CircularCell> cellSet = new
HashSet<CircularCell>();

                /* compute number of rows and column in
our hatched grid (floor) */
                double spaceWidth =
space.getSecond().getX() - space.getFirst().getX();
                double spaceHeight =
space.getSecond().getY() - space.getFirst().getY();
                double analogy = (spaceWidth /
spaceHeight);
                double factors = m / analogy;
                int numOfCentersX = (int) Math.floor(
analogy * Math.sqrt( factors ));
                int numOfCentersY = (int) Math.sqrt(
factors );

                /* compute distances between base stations
*/
                double distBetweenColumns =
(double)spaceWidth / ( (double)numOfCentersX ); // +1 one
because we do not need basestations right on the border
                double distBetweenRows =
(double)spaceHeight / ( (double)numOfCentersY );

                /* compute approximation to optimal cell
radius to cover whole space
                 * find maximum between
distanceBetweenRows and distanceBetweenColumns/2
                 * and set radius to be this distance */
```

```
                //TODO revisit
                Double radius = (distBetweenRows > distBetweenColumns) ? distBetweenRows : distBetweenColumns;

                /* for each base station point */
                for (int x=0; x < numOfCentersX; x++){
                        for (int y=0; y < numOfCentersY; y++){

                                /* here you create the hatched grid */
                                DoublePoint center = new DoublePoint( ( x * distBetweenColumns ) + distBetweenColumns/4 + ( (y % 2) * distBetweenColumns/2 ), y * distBetweenRows + distBetweenRows/2);
                                CircularCell newCell = new CircularCell( center, radius );

                                /* add cell to cellset */
                                cellSet.add( newCell );

                        }
                }

                return cellSet;

        }


        private HashMap<Integer, ArrayList<UserReport>> constructReports(
                        HashMap<Integer, ArrayList<User>> locationsPerEpoch,
                        HashSet<CircularCell> cellSet) {

                HashMap<Integer, ArrayList<UserReport>> result = new HashMap<Integer, ArrayList<UserReport>>();

                /* for each epoch */
                for ( Integer currEpoch : locationsPerEpoch.keySet() ) {
                        ArrayList<UserReport> reportsList = new ArrayList<UserReport>();

                        /* ignore epochs where users < k !! */
                        if ( locationsPerEpoch.get( currEpoch ).size() - 1 < this.k
                                        || locationsPerEpoch.get( currEpoch ).size() < this.minUserThreshold ) {
                                /* remove epoch from nPerEpoch!! to be fair to Koudas and CMP computing the optimal cellsize */
                                this.nPerEpoch.remove( currEpoch );
                                continue;
                        }

                        /* for each user */
                        for ( User currUser : locationsPerEpoch.get( currEpoch ) ) {

                                // TODO: implement location obfuscation using function

                                /* cell the user should be registered in */
                                CircularCell regCell = null;

                                /* cells that cover a user */

                                HashSet<CircularCell> coveringCells = new HashSet<CircularCell>();

                                /* for finding the minimum distance ratio */
                                Double minRatio = Double.POSITIVE_INFINITY;

                                /* for each cell */
                                for ( CircularCell currCell : cellSet ) {

                                        /* compute distance from user to cell location */
                                        Double distance = currUser.getObfLocation().distance( currCell.getCenter() );
```

```
                                                    result.put( currEpoch,
                                /*  checkreportsList );
whether user is inside cell (distance<radius) */
                                if  (                       } // end for each epoch
distance < currCell.getRadius() ) {
                                                    return result;

        /* add cell to the covering cells of the user */        }

        coveringCells.add( currCell );

                                }
                                                /*  *******  END  of  functions
                                /*  ********************************* */
compute the ratio of distance to radius */
                                Double
distRatio = distance / currCell.getRadius();
                                                public void run(String lfn, String cofn, String ufn, int
                                /*  keepik, int mut ) {
the cell with the smallest ratio */
                                if  (                       String locationsFilename = lfn;
distRatio < minRatio ) {                                    String cellsOutFileName=cofn;
                                                            String usersFilename=ufn;
                                                            k=ik;
        minRatio = distRatio;                               minUserThreshold=mut;

        regCell = currCell;
                                                            HashSet<CircularCell> cellSet = null;
                                }                           ArrayList<Double> cellRadiusSizes = null;
                                                            CommutativePair<DoublePoint>  space  =
                                                    null;
                        } // end for each cell

                                                            char
                        /* create user report  locationsFileFormat=LAOUDIASOLDENBURG;
/* add the cell with the smallest ratio as the registered cell of the          String cellsFilename, outputFilename;
user */
                        UserReport                          /* parse the files to create datastructures */
userReport = new UserReport(currUser, coveringCells, regCell);          HashMap<Integer,  ArrayList<User>>
                                                    locationsPerEpoch = null;
                        /*  add  user  report              /* choose fileparser according to input file
into reportlist */                                  format */
                        reportsList.add(                    LocationsFileParserInterface
userReport );                                       locationsFileParser = null;
                                                            switch ( locationsFileFormat ) {
                    } // end for each user                   //          case MANUAL:
                                                            //
                    /*  put  the  list  of  user  reports       locationsFileParser  =  new
into map */                                         ManualLocationsFileParser();
                                                            //                  break;
```

```
                //                  case OLDENBURG:
                //
        locationsFileParser = new OldenburgFileParser();
                //                  break;
                case LAOUDIASOLDENBURG:
                        locationsFileParser = new
LaoudiasOldenburgFileParser();
                        break;
                        //                  case
SPANAKIS:
                        //
        locationsFileParser          =          new
SpanakisLocationsFileParser();
                        //
        break;
                case LAOUDIASGEOLIFE:
                        locationsFileParser = new
LaoudiasGeolifeFileParser();
                        break;
                default:
                        System.out.println("ERROR:
file format " + locationsFilename + " is unknown");
                        System.exit(-1);
                }

                try {

                        locationsPerEpoch =
locationsFileParser.parseFile( locationsFilename );
                        /* store average N */
                        space =
locationsFileParser.getSpace();
                        nPerEpoch =
locationsFileParser.getNPerEpoch();

                        System.out.println("space " +
space );

                } catch (FileNotFoundException e) {
                        System.out.println("Exception:
file not found " + locationsFilename);
                        e.printStackTrace();
                }
```

```
                        if ( cellRadiusSizes == null ) {
                                cellRadiusSizes = new
ArrayList<Double>(1);
                //                  cellRadiusSizes.add(1000.0);   //
1km
                                cellRadiusSizes.add(4000.0);   //
4km
                                //
                cellRadiusSizes.add(16000.0); // 16km
                                //
                cellRadiusSizes.add(64000.0); // 64km
                                //
                cellRadiusSizes.add(256000.0); // 256km
                        }

                        /* for all radius values */
                        for ( double currRadius : cellRadiusSizes )
{

                                /*      create      cells      and
reportsPerEpoch */
                                cellSet =
CellPlacement.hatchedGrid( currRadius , space );
                                reportsPerEpoch =
constructReports(locationsPerEpoch, cellSet);

                                try {

                                        File outFile = null;
                                        outFile = new
File(cellsOutFileName);
                                        BufferedWriter
writer;

                                        writer = new
BufferedWriter(new FileWriter(outFile,true));



                //
                writer.append(""+currRadius);
```

```
for (CircularCell cell :cellSet){
    // Write text to file
    writer.append(""+cell.getId());
    writer.append(","+cell.getCenter().getX());
    writer.append(","+cell.getCenter().getY());
    writer.append(","+cell.getRadius()+"\n");
}
writer.close();
} catch (IOException e){
    e.printStackTrace();
}

try {
    File outFile = null;
    outFile = new File(usersFilename);
    BufferedWriter writer;
    writer = new BufferedWriter(new FileWriter(outFile,true));

//  writer.append(""+currRadius);

    writer.append("#ID,X,Y,CID,nCells\n");
```

```
for (ArrayList<UserReport> value : reportsPerEpoch.values()) {
    for (UserReport user: value)
    {
        writer.append(""+user.getUser().getId());

        writer.append(","+user.getUser().getObfLocation().getX());

        writer.append(","+user.getUser().getObfLocation().getY());

        writer.append(","+user.getRegCell().getId());

        for (CircularCell cCells: user.getCoveringCells())
        {
            writer.append(","+cCells.getId());
        }
        writer.append("\n");
    }
}
writer.close();
} catch (IOException e){
    e.printStackTrace();
}
}
}
}
```

Παράρτημα Γ

Scripts για τον έλεγχο και πειράματα στον server

```
#!/bin/bash

for i in `cat hosts.txt`
do
        if [[ $i =~ "#" ]]; then
                # echo "SXOLIA"
                continue
        fi
        eval "ssh $i time ~/run/execConnections.sh ~/run/userData/UsersReports2000 1 500 | grep real: &"
        echo "====> $i"
done
```

```
#!/bin/bash

if [[ ! -f $1 ]]; then
        echo "file $1 not found"
        exit
fi

if [[ -z $2 || -z $3 ]]; then
        echo "usage: execConnections filename startingLine endingLine"
        exit
fi

declare -i SOCK=3
SERVER_IP="10.16.20.22"

# My Mac
# SERVER_IP="10.16.5.173"

SERVER_PORT="65002"


declare -i counter=0
declare -i min=$2
declare -i max=$3
# echo "$1  $2  $3"

for myLine in `cat $1`
# cat $1 | while read myLine
do
```

```
        if [[ $myLine =~ "#" ]]; then
                # echo "SXOLIA"
                continue
        fi

        (( counter= $counter + 1 ))
        # echo $counter

        if [[ $counter -lt $min  ]]; then
                continue
        fi
        if [[ $counter -gt $max ]]; then
                break
        fi
        eval "~/run/execClient.sh $SERVER_IP $SERVER_PORT $myLine &"


done

#!/bin/bash
declare -i SOCK=3
SERVER_IP=$1
SERVER_PORT=$2
myLine=$3

eval "exec $SOCK<>/dev/tcp/$SERVER_IP/$SERVER_PORT"

eval "read line <&$SOCK"
echo $line

eval "echo $myLine >&$SOCK"

eval "read line <&$SOCK"

echo $line

eval "exec $SOCK>&-"
eval "exec $SOCK<&-"
```

Παράρτημα Δ