Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Τομέας Επικοινωνιών, Ηλεκτρονικής και Συστημάτων Πληροφορικής

Ολοκληρωμένο Σύστημα Συλλογής και Διαχείρισης Δεδομένων Ποιότητας Δικτύων Κινητών Επικοινωνιών

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Χαρούλα Γ. Ρεμούνδου

Γεώργιος Δ. Ριζοθανάσης

Επιβλέπων: Μιχαήλ Ε. Θεολόγου
Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2011

Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Τομέας Επικοινωνιών, Ηλεκτρονικής και Συστημάτων Πληροφορικής

Ολοκληρωμένο Σύστημα Συλλογής και Διαχείρισης Δεδομένων Ποιότητας Δικτύων Κινητών Επικοινωνιών

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Χαρούλα Γ. Ρεμούνδου

Γεώργιος Δ. Ριζοθανάσης

Επιβλέπων: Μιχαήλ Ε. Θεολόγου
Καθηγητής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την

……	……	……
Μιχαήλ Θεολόγου	Ευστάθιος Συκάς	Γεώργιος Στασινόπουλος
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2011

…….
Χαρούλα Γ. Ρεμούνδου
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Γεώργιος Δ. Ριζοθανάσης
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περιεχόμενα

Περίληψη

Τα κινητά τηλέφωνα χρησιμοποιούνται στη σύγχρονη εποχή σε παγκόσμιο επίπεδο και από όλες τις ηλικιακές ομάδες. Οι υπηρεσίες που παρέχουν έχουν ξεφύγει από την απλή μετάδοση φωνής και συμπεριλαμβάνουν την πλοήγηση στον παγκόσμιο ιστό εν κινήσει, την ανάγνωση εφημερίδων, την παροχή υπηρεσιών βασισμένων στη θέση των χρηστών και πολλές ακόμη λειτουργίες. Προς αυτή την κατεύθυνση έχει βοηθήσει η εξάπλωση των δικτύων 3ης γενιάς και η εμφάνιση των δικτύων 4ης γενιάς, που προσφέρουν υψηλές ταχύτητες μετάδοσης δεδομένων, καθώς και η ευρύτατη διάδοση των “έξυπνων” κινητών συσκευών (smartphones), που παρέχουν πολλές εφαρμογές και δυνατότητες για τους χρήστες.

Η ύπαρξη, λοιπόν, ενός δικτύου κινητών επικοινωνιών που θα μπορεί να αντεπεξέλθει στις ανάγκες των σύγχρονων χρηστών για καλή ποιότητα επικοινωνίας κάθε χρονική στιγμή σε οποιοδήποτε σημείο αποτελεί πρωταρχικό μέλημα κάθε παρόχου κινητών επικοινωνιών. Στην ύφανση αυτής της συλλογιστικής, ο έλεγχος, η αντιμετώπιση σφαλμάτων και η βελτιστοποίηση του τρόπου λειτουργίας των δικτύων συνιστούν καθημερινές διαδικασίες μίας εταιρείας παροχής κινητών επικοινωνιών.

Σκοπός της παρούσας διπλωματικής είναι η μελέτη των παραγόντων που επηρεάζουν την ποιότητα λειτουργίας των δικτύων κινητών επικοινωνιών και η ανάπτυξη ενός ολοκληρωμένου συστήματος συλλογής και διαχείρισης δεδομένων ποιότητας. Πιο συγκεκριμένα, δημιουργήθηκε μία εφαρμογή για κινητά τερματικά με λειτουργικό σύστημα Android, η οποία καταγράφει την ισχύ του λαμβανόμενου σήματος του κινητού και τη θέση του. Για τη διαχείριση των καταγραφών αναπτύχθηκε ένα σύστημα επεξεργασίας και αποθήκευσής τους σε βάση δεδομένων.

Επιπλέον, σχεδιάστηκε και μία ιστοσελίδα η οποία απεικονίζει την κάλυψη σύμφωνα με το σύνολο όλων των καταγραφών που υφίστανται στη βάση δεδομένων. Αυτό γίνεται με την αποτύπωση της μετρημένης λαμβανόμενης ισχύς σε γεωγραφικό χάρτη (Google Maps). Έτσι, ο τελικός χρήστης/ καταναλωτής έχει μία αντικειμενική εποπτική αντίληψη για την ποιότητα του δικτύου ενός παρόχου κινητών επικοινωνιών.

Λέξεις Κλειδιά:

κινητά τερματικά, δίκτυα κινητών επικοινωνιών, ποιότητα δικτύου, μετρήσεις ποιότητας δικτύου, λαμβανόμενη ισχύς σήματος, θέση κινητού, βάση δεδομένων, αποτύπωση μετρήσεων, gps, android, google maps

Summary

Mobile phones are being used in the modern era at a global level and by all age groups. Provided services include far more than voice transmission, like world wide web surfing with high mobility, reading newspapers, location based services and much more. One reason for that are the growth of 3G networks and the appearance of 4G networks, which offer high data speed connections. Another reason is the widely used smart phones, which offer numerous applications and potentials to users.

So, the existence of a mobile communications network that is able to fulfill the needs of today users for fine quality communication every single moment at every location is a major concern of all mobile communications providers. According to that, controlling, troubleshooting and optimizing constitute everyday procedures for corporations in the field of mobile communications.

The purpose of this diploma thesis is the study of factors affecting the quality of wireless networks performance, as well as the development of an integrated system of data collection and management. The system comprises of an Android based mobile application, which records the received signal strength of the cell phone and its position and of a processing and storing into database algorithm for managing the records.

Furthermore, a web page was designed that shows the network coverage according to all the measurements stored into the database. Specifically, the received signal strength is displayed on a Google map. Thus the end-user/ consumer has an objective view for the quality of a mobile communications provider.

Key Words:
mobile phones, mobile communications networks, network quality, quality data for network, received signal strength, cell phone location, database, measurements display, gps, android, google maps.

ΚΕΦΑΛΑΙΟ 1: Εισαγωγή

1.1 Συστήματα Κινητών Επικοινωνιών - Είδη και Εξέλιξη

Συστήματα 1ης Γενιάς (1G)

Βασικό χαρακτηριστικό των συστημάτων 1ης γενιάς είναι ότι ήταν πλήρως αναλογικά, χρησιμοποιούσαν την αναλογική διαμόρφωση FM (Frequency Modulation) και την τεχνική πολλαπλής πρόσβασης FDMA (Frequency Division Multiple Access). Τα περισσότερα από τα δίκτυα 1ης γενιάς είχαν μια απόσταση 45MHz μεταξύ των συχνοτήτων εκπομπής και λήψης.

- Το πρώτο κυψελωτό δίκτυο που λειτούργησε ήταν στην Ιαπωνία το 1979 από την εταιρεία NTT. Το δίκτυο αυτό χρησιμοποιούσε 600 FM duplex διαύλους με εύρος 25kHz στα 925-940/870-885MHz.
- Το πρώτο ευρωπαϊκό κυψελωτό σύστημα ήταν το NMT450 και αναπτύχθηκε το 1981 στις σκανδιναβικές χώρες στη ζώνη 450-470MHz. Το 1986 εξελίχθηκε στο NMT900 στη ζώνη 890-915/917-950MHz.
- Στις ΗΠΑ αναπτύχθηκε το AMPS το 1983 στη ζώνη 824-849/869-894MHz με εύρος διαύλου 30kHz.
- Παρόμοια με το AMPS είναι το TACS που λειτούργησε στο Ηνωμένο Βασίλειο, το C-450 στη Γερμανία και στην Πορτογαλία, το Radiocom2000 στη Γαλλία και το RTMS στην Ιταλία.

Συστήματα 2ης Γενιάς (2G)

Τα συστήματα 2ης γενιάς είναι πλήρως ψηφιακά και τα κύρια πλεονεκτήματά τους είναι η ψηφιακή κρυπτογράφηση των συνομιλιών, η αποδοτικότερη χρησιμοποίηση του φάσματος με αποτέλεσμα την εξυπηρέτηση περισσότερων συνδρομητών και η δυνατότητα υπηρεσιών δεδομένων, όπως το SMS (Short Message Service).

- Το πρώτο ψηφιακό κυψελωτό σύστημα αναπτύχθηκε στην Ευρώπη στις αρχές της δεκαετίας του 1990 και είναι το GSM (Global System for Mobile Communications). Βασίζεται στην τεχνική TDMA (Time Division Multiple Access) με 8 χρονοσχισμές ανά δίαυλο και 200kHz εύρος ζώνης ραδιοδιαύλων. Αρχικά το GSM είχε στόχο να λειτουργεί μόνο σε περιοχές συχνοτήτων γύρω από τα 900MHz, αλλά αργότερα προστέθηκαν κι οι προδιαγραφές για τη λειτουργία του γύρω από τα 1800MHz. Στην Αμερική χρησιμοποιούνται επίσης ζώνες γύρω από τα 850MHz και τα 1900MHz. Ας σημειωθεί ότι αποτελεί το δημοφιλέστερο σύστημα κινητών επικοινωνιών παγκοσμίως, με πάνω από το 80% όλων των συνδρομητών.
- Στις ΗΠΑ αναπτύχθηκε το IS-54 ή D-AMPS (με εξέλιξη το IS-136). Χρησιμοποιεί την τεχνική TDMA, με απόσταση φερόντων τα 30kHz και συχνότητες λειτουργίας ίδιες με εκείνες του AMPS (824-894MHz).
- Επιπλέον στις ΗΠΑ υιοθετήθηκε και το IS-95 που βασίζεται στην τεχνική CDMA (Code Division Multiple Access) και προτάθηκε από την εταιρεία Qualcomm. Με την τεχνική CDMA, που χρησιμοποιεί διασπορά φάσματος απευθείας ακολουθίας (spread spectrum direct sequence), μια συχνότητα χρησιμοποιείται ταυτόχρονα από πολλά κινητά τερματικά σε δοθείσα κυψέλη και τα σήματα διαχωρίζονται εξαπλώνοντάς τα με διαφορετικούς κώδικες. Το εύρος ζώνης του διεσπαρμένου σήματος είναι 1,25MHz. Το IS-95 αποτέλεσε οδηγό και βάση εκκίνησης για τα συστήματα 3ης γενιάς.
- Στην Ιαπωνία αναπτύχθηκε το 1989 το σύστημα PDC (Personal Digital Cellular), στηριζόμενο στις αρχές του IS-54, με TDMA τεχνική πολλαπλής πρόσβασης και απόσταση φερόντων 25kHz. Οι συχνότητες λειτουργίας του είναι τα 810-826/940-956MHz και 1429-1453/1477-1501MHz.

Η ανάγκη για παροχή υπηρεσιών με υψηλούς ρυθμούς μετάδοσης δεδομένων, ώστε να μεταδίδονται εικόνες υψηλής ποιότητας και video πραγματικού χρόνου, ή να παρέχεται πρόσβαση στο internet με υψηλές ταχύτητες, οδήγησε στη σχεδίαση των συστημάτων 2,5 γενιάς (2.5G)

- Η αναβάθμιση του GSM με την υποστήριξη υπηρεσιών δεδομένων με τεχνολογία μεταγωγής πακέτου οδήγησε στο GPRS (General Packet Radio Service). Η βασική ιδέα στο GPRS είναι η χρησιμοποίηση των χρονοσχισμών του GSM που δεν χρησιμοποιούνται για φωνή, για τη μεταφορά ασύγχρονων δεδομένων με μεταγωγή πακέτου, ενώ οι υπηρεσίες φωνής εξακολουθούν να εξυπηρετούνται από την τεχνολογία μεταγωγής κυκλώματος του GSM. Οι ρυθμοί μετάδοσης δεδομένων κυμαίνονται από τα 56kbps μέχρι τα θεωρητικά 115kbps.
- Μια επιπλέον εξέλιξη του GPRS είναι το σύστημα EDGE (Enhanced Data rates for GSM Evolution) (το οποίο αποκαλείται και Enhanced-GPRS, EGPRS). Η βασική διαφορά του είναι ότι χρησιμοποιεί διαφορετικό τρόπο διαμόρφωσης (8-PSK) και διαφορετικούς τύπους κωδικοποίησης διαύλου. Οι ρυθμοί μετάδοσης δεδομένων που μπορεί να επιτύχει φθάνουν μέχρι και τα 236,8kbps.

Συστήματα 3ης Γενιάς (3G)

Βασικό χαρακτηριστικό των συστημάτων 3ης γενιάς είναι οι υψηλοί ρυθμοί μετάδοσης δεδομένων. Αυτό δίνει τη δυνατότητα για εφαρμογές όπως η video - κλήση, mobile TV, υπηρεσίες βασισμένες στη θέση (location based services, LBS) και άλλες πολυμεσικές εφαρμογές.

- Το UMTS (Universal Mobile Telecommunications System) αναπτύχθηκε από τον Ευρωπαϊκό Οργανισμό Προτυποποίησης (ETSI) και χρησιμοποιεί W-CDMA (Wideband CDMA) ραδιοεπαφές με εύρος διαύλου 5MHz και ασύρματη πρόσβαση δύο τύπων FDD (Frequency Division Duplexing) και TDD (Time Division Duplexing). Μπορεί να φτάσει ρυθμούς μετάδοσης μέχρι και 2Mbps. Το UMTS αποτελεί εξέλιξη του GSM, αφού χρησιμοποιεί το δίκτυο κορμού του GSM και τα περισσότερα κινητά τερματικά υποστηρίζουν λειτουργία και σε GSM και σε UMTS με δυνατότητα μεταπομπής μεταξύ των συστημάτων για βελτίωση της κάλυψης και εξισορρόπηση του τηλεπικοινωνιακού φορτίου. Οι ζώνες συχνοτήτων του UMTS είναι συνήθως γύρω από τα 2100MHz και σπανιότερα γύρω από τα 900MHz ή τα 1850MHz. Το σύστημα αυτό χρησιμοποιείται παγκοσμίως, εκτός από την Κίνα όπου χρησιμοποιείται με ραδιοεπαφές TD-SCDMA (Time Division Synchronous CDMA), με εύρος διαύλου 1,6MHz.
- Εξέλιξη του IS-95 (2G) είναι το σύστημα CDMA2000, το οποίο λειτουργεί κυρίως στη Βόρειο Αμερική και στη Νότια Κορέα. Με την τελευταία τεχνολογία EVDO Rev B (Evolution Data Optimized) το δίκτυο κορμού βασίζεται εξολοκλήρου στη μεταγωγή πακέτου, οπότε δεν δεσμεύεται από τους περιορισμούς ενός δικτύου μεταγωγής κυκλώματος και μπορεί να φτάσει ρυθμούς μετάδοσης μέχρι και 14,7Mbps. Το εύρος ζώνης κάθε καναλιού είναι 1,25MHz.

Όπως συνέβη και με τα δίκτυα 2ης γενιάς, έτσι και με τα δίκτυα 3ης γενιάς η ανάγκη για ακόμη υψηλότερους ρυθμούς μετάδοσης δεδομένων οδήγησε στα δίκτυα 3,5 γενιάς. Προς την κατεύθυνση αυτή συνέβαλε και η εκτεταμένη χρήση των smartphones (κινητών τηλεφώνων με ανεπτυγμένες δυνατότητες σύνδεσης στο διαδίκτυο) και των mobile modems στα laptop και netbook.

- Το HSPA (High Speed Packet Access) αποτελεί μια συγχώνευση δύο πρωτοκόλλων του HSDPA για την κάτω ζεύξη και του HSUPA για την άνω ζεύξη, το οποίο επεκτείνει και βελτιώνει την απόδοση των UMTS W-CDMA συστημάτων. Εκτός από αυξημένους ρυθμούς μετάδοσης, 7,2Mbps για το

HSDPA και 5,8Mbps για το HSUPA, το HSPA μειώνει την καθυστέρηση και αυξάνει τη χωρητικότητα του συστήματος.

- Εξέλιξη του HSPA είναι το HSPA+ ή Evolved HSPA που ξεκίνησε να χρησιμοποιείται από το 2009 και μπορεί να προσφέρει θεωρητικά ταχύτητες μέχρι και 84Mbps για την κάτω ζεύξη και 22Mbps για την άνω ζεύξη. Το HSPA+ χρησιμοποιεί την τεχνική των πολλαπλών κεραιών (MIMO) και διαμόρφωση 64QAM.

Συστήματα 4ης Γενιάς (4G)

Το 2009 η ITU-R (International Telecommunication Union – Radiocommunication sector) έθεσε τις απαιτήσεις για τα συστήματα 4ης γενιάς στα 100Mbps για υψηλής κινητικότητας επικοινωνία, όπως τρένα ή αυτοκίνητα και στο 1Gbps για χαμηλής κινητικότητας επικοινωνία, όπως πεζούς και χρήστες σε σταθερή θέση. Άλλες βασικές απαιτήσεις που πρέπει να ικανοποιεί ένα σύστημα 4ης γενιάς είναι:

- Χρήση του πρωτοκόλλου IP τόσο για τη μετάδοση φωνής όσο και για τη μεταφορά δεδομένων.
- Δυναμικός διαμοιρασμός και χρησιμοποίηση των πόρων του δικτύου ώστε να υποστηρίζονται ταυτόχρονα περισσότεροι χρήστες ανά κυψέλη.
- Κλιμακωτό εύρος ζώνης καναλιού, ανάμεσα σε 5 MHz και 20 MHz, προαιρετικά μέχρι και 40 MHz.
- Φασματική αποδοτικότητα της ζεύξης και του συστήματος.
- Ομαλές μεταπομπές ανάμεσα σε ετερογενή δίκτυα.

Για να καλυφθούν οι παραπάνω απαιτήσεις χρησιμοποιούνται μια σειρά προηγμένων τεχνικών, όπως:

- Κεραίες τύπου MIMO, οι οποίες βελτιώνουν την ποιότητα της ζεύξης αυξάνοντας τον ρυθμό μετάδοσης και την εμβέλεια του συστήματος χωρίς να αυξάνονται οι απαιτήσεις σε ισχύ εκπομπής και εύρος ζώνης.
- Χρήση τεχνικών διαμόρφωσης πολλαπλού φέροντος, οι οποίες διαιρούν την κύρια ροή δεδομένων σε μικρότερες ροές, κάθε μία από τις οποίες διαμορφώνει ένα ξεχωριστό φέρον. Έτσι, γίνεται εκμετάλλευση των χαρακτηριστικών του ασύρματου διαύλου και αντιμετωπίζονται επιλεκτικά και στοχευμένα οι δυσμενείς συνθήκες διάδοσης που επικρατούν στις διάφορες ζώνες συχνοτήτων.
- Χρήση τεχνικών δυναμικής απόδοσης διαύλων, οι οποίες επιτρέπουν τη βέλτιστη κατανομή και επαναχρησιμοποίηση των διαθέσιμων διαύλων.
- Χρήση τεχνικών δρομολόγησης που λαμβάνουν υπόψη τόσο τις απαιτήσεις του κάθε χρήστη σε εύρος ζώνης όσο και τις συνθήκες διάδοσης που επικρατούν στον κάθε δίαυλο.

Το πρώτο σύστημα που παρουσιάστηκε ως 4ης γενιάς, αν και δεν πληροί τις παραπάνω απαιτήσεις είναι το LTE.

- Το LTE (Long Term Evolution) στηρίζεται στην τεχνολογία των GSM/EDGE και UMTS/HSPA και η αρχιτεκτονική του βασίζεται πλήρως στο πρωτόκολλο IP, προσφέροντας ταχύτητες μέχρι και 300Mbps και 75Mbps για την κάτω και άνω ζεύξη αντίστοιχα. Το πρώτο δίκτυο LTE λειτούργησε στα τέλη του 2009 στη Σουηδία και τη Νορβηγία. Ας σημειωθεί ότι η ασύρματη διεπαφή του LTE είναι ασύμβατη με τα δίκτυα 2ης και 3ης γενιάς, γι' αυτό πρέπει να λειτουργήσει σε ξεχωριστό ραδιοφάσμα.
- Το LTE Advanced αποτελεί εξέλιξη του LTE και στοχεύει όχι μόνο να ικανοποιήσει τις απαιτήσεις της ITU-R για τα δίκτυα 4ης γενιάς, αλλά και να τις ξεπεράσει. Το LTE Advanced αναμένεται να είναι εμπορικά διαθέσιμο στο 2012.

1.2 Στατιστικά

Σύμφωνα με την ITU, το 2002 σε 2 μόνο χώρες η διείσδυση των κινητών κυψελωτών συστημάτων ξεπερνούσε το 100% του πληθυσμού τους. Στο τέλος του 2010 οι χώρες αυτές έφτασαν σχεδόν τις 100. Μάλιστα σε 17 από αυτές η διείσδυση ξεπερνά το 150% του πληθυσμού τους.

Τα δίκτυα 2ης γενιάς καλύπτουν το 90% του παγκόσμιου πληθυσμού, ενώ τα δίκτυα 3ης γενιάς το 45%.

Παρακάτω φαίνεται ένας πίνακας με τις 25 χώρες με τους περισσότερους συνδρομητές ευρυζωνικών συνδέσεων (mobile-broadband) ανά 100 κατοίκους.

Χώρα	**Ενεργοί συνδρομητές με ευρυζωνικές συνδέσεις ανά 100 κατοίκους**
Νότιος Κορέα	91,0
Ιαπωνία	87,8
Σουηδία	84,0
Αυστραλία	82,7
Φινλανδία	78,1
Χονγκ Κονγκ, Κίνα	74,5
Πορτογαλία	72,5
Λουξεμβούργο	72,1
Σιγκαπούρη	69,7
Αυστρία	67,4
Νέα Ζηλανδία	66,2
Κουβέιτ	63,5
Ισραήλ	62,2
Σουλτανάτο του Μπρουνέι	61,4
Κύπρος	61,3
Ιταλία	59,4
Ενωμένα Αραβικά Εμιράτα	58,4
Ελλάδα	58,3
Σαουδική Αραβία	57,8

Μακάο, Κίνα	56,1
Ηνωμένο Βασίλειο	56,0
Ισπανία	55,7
Δανία	54,7
Ηνωμένες Πολιτείες της Αμερικής	54,0
Ιρλανδία	47,3

ΚΕΦΑΛΑΙΟ 2: Δείκτες Ποιότητας Υπηρεσιών Συστημάτων Κινητών Επικοινωνιών.

Οι Δείκτες Ποιότητας (Δ.Π.) δημοσιεύονται στις ιστοσελίδες των παρόχων και της ΕΕΤΤ και διευκολύνουν τους καταναλωτές στη σύγκριση της ποιότητας των υπηρεσιών που προσφέρονται από τους διάφορους παρόχους, καθώς επίσης και στην πιστοποίηση της ποιότητας υπηρεσίας που ήδη λαμβάνουν. Οι συγκεκριμένοι δείκτες ποιότητας που μετρούνται και παρουσιάζονται για τις υπηρεσίες συστημάτων κινητών επικοινωνιών είναι οι ακόλουθοι :

α. Δείκτες ανεξάρτητες της υπηρεσίας:
Μ01: Διαθεσιμότητα δικτύου-ραδιοκάλυψη

β. Δείκτες υπηρεσίας τηλεφωνίας:
Μ02: Πιθανότητα εμπλοκής κλήσης φωνής
Μ03: Πιθανότητα διακοπής κλήσης φωνής
Μ04: Ποιότητα φωνής
Μ05: Χρόνος αποκατάστασης κλήσης φωνής

γ. Δείκτες υπηρεσίας εικονοτηλεφωνίας:
Μ06: Πιθανότητα εμπλοκής κλήσεων εικονοτηλεφωνίας
Μ07: Πιθανότητα διακοπής κλήσης εικονοτηλεφωνίας
Μ08: Ποιότητα φωνής εικονοτηλεφωνίας
Μ09: Ποιότητα βίντεο
Μ10: Χρόνος εγκατάστασης κλήσης εικονοτηλεφωνίας

δ. Δείκτες ποιότητας web browsing:
Μ11: Πιθανότητα αποτυχίας μεταφοράς δεδομένων http
Μ12: Μέσος ρυθμός δεδομένων http

Πιο αναλυτικά:
Μ01 : Διαθεσιμότητα δικτύου-ραδιοκάλυψη

Ο Δείκτης Μ01 (Διαθεσιμότητα δικτύου-ραδιοκάλυψη) δηλώνει τη γεωγραφική κάλυψη για την παροχή οποιασδήποτε υπηρεσίας μέσω ενός δικτύου κινητών υπηρεσιών σε πανελλαδική κλίμακα.

Ο Δείκτης Ποιότητας Μ01 εκφράζεται με ποσοστό και ορίζεται ως το πηλίκο του πλήθους των σημείων μέτρησης στα οποία υπάρχει ραδιοκάλυψη προς το συνολικό αριθμό των σημείων μέτρησης. Επίσης, ο Δείκτης Ποιότητας Μ01 εκφράζεται με έναν από τους ακόλουθους όρους κατηγοριών ποιότητας ραδιοκάλυψης:

- καλή
- αποδεκτή
- κακή και
- δεν υπάρχει

Υπόχρεοι παροχής του Δείκτη Ποιότητας Μ01 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίες μέσω συστημάτων κινητών επικοινωνιών και ειδικότερα μέσω συστημάτων GSM/DCS, UMTS και TETRA.

Ο Δείκτης Ποιότητας Μ01 μετράται και παρουσιάζεται ανά διετία.

M02: Πιθανότητα εμπλοκής κλήσης φωνής

Ο Δείκτης M02 (Πιθανότητα εμπλοκής κλήσης φωνής) αφορά στην εμπλοκή κλήσεων φωνής σε ένα δίκτυο κινητών επικοινωνιών και χαρακτηρίζει την προσβασιμότητα στην υπηρεσία φωνής του δικτύου.

Ο Δείκτης Ποιότητας M02 εκφράζεται με ποσοστό και ορίζεται ως το πηλίκο του πλήθους των κλήσεων φωνής στις οποίες παρουσιάστηκε εμπλοκή προς το συνολικό αριθμό των κλήσεων φωνής για τις οποίες υπάρχει διαθεσιμότητα του δικτύου.

Τα αποτελέσματα των μετρήσεων παρουσιάζουν την πιθανότητα πρόσβασης, η οποία προκύπτει εάν από τη μονάδα αφαιρεθεί ο Δείκτης Ποιότητας M02.

Υπόχρεοι παροχής του Δείκτη Ποιότητας M02 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίες μέσω συστημάτων κινητών επικοινωνιών και ειδικότερα μέσω συστημάτων GSM/DCS, UMTS και TETRA.

Ο Δείκτης Ποιότητας M02 μετράται και παρουσιάζεται ανά διετία.

M03: Πιθανότητα διακοπής κλήσης φωνής

Ο Δείκτης M03 (Πιθανότητα διακοπής κλήσης φωνής) αφορά στην πιθανότητα τερματισμού (διακοπής) μιας επιτυχημένης προσπάθειας κλήσης φωνής, για την οποία υπάρχει διαθεσιμότητα του δικτύου (ραδιοκάλυψη), για οποιοδήποτε λόγο εκτός από σκόπιμο τερματισμό του καλούντος ή του καλούμενου.

Ο Δείκτης Ποιότητας M03 εκφράζεται με ποσοστό και ορίζεται ως το πηλίκο του πλήθους των επιτυχημένων κλήσεων φωνής που τερματίστηκαν για οποιοδήποτε λόγο εκτός από σκόπιμο τερματισμό του καλούντος ή του καλούμενου, προς το συνολικό πλήθος κλήσεων για τις οποίες υπάρχει διαθεσιμότητα του δικτύου.

Τα αποτελέσματα των μετρήσεων παρουσιάζουν την πιθανότητα διατήρησης της κλήσης, η οποία προκύπτει εάν από τη μονάδα αφαιρεθεί ο Δείκτης Ποιότητας M03.

Υπόχρεοι παροχής του Δείκτη Ποιότητας M03 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίες μέσω συστημάτων κινητών επικοινωνιών και ειδικότερα μέσω συστημάτων GSM/DCS, UMTS και TETRA.

Ο Δείκτης Ποιότητας M03 μετράται και παρουσιάζεται ανά διετία.

M04: Ποιότητα φωνής

Ο Δείκτης M04 (Ποιότητα φωνής) αποτελεί το δείκτη ποιότητας μετάδοσης από άκρο σε άκρο της φωνής της υπηρεσίας κινητής τηλεφωνίας. Εκφράζεται με έναν από τους ακόλουθους όρους κατηγοριών ποιότητας:

- πολύ καλή ποιότητα
- καλή ποιότητα
- μέση ποιότητα

- χαμηλή ποιότητα και
- πολύ χαμηλή ποιότητα.

Υπόχρεοι παροχής του Δείκτη Ποιότητας Μ04 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίεςμέσω συστημάτων κινητών επικοινωνιών και ειδικότερα μέσω συστημάτων GSM/DCS, UMTS και TETRA.

Ο Δείκτης Ποιότητας Μ04 μετράται και παρουσιάζεται ανά διετία.

Μ05: Χρόνος αποκατάστασης κλήσης φωνής

Ο Δείκτης Μ05 (Χρόνος αποκατάστασης κλήσης φωνής) εκφράζει το χρόνο στον οποίο αποκαθίσταται η κλήση φωνής από τη στιγμή που ο τελικός χρήστης συμπληρώνει τον αριθμό του καλούμενου συνδρομητή.

Ο Δείκτης Ποιότητας Μ05 εκφράζεται σε δευτερόλεπτα και ορίζεται ως ο χρόνος από τη συμπλήρωση από τον καλούντα της πληροφορίας διεύθυνσης, δηλαδή του αριθμού τηλεφώνου του καλούμενου, μέχρι τη λήψη ειδοποίησης εγκατάστασης κλήσης φωνής.
Υπόχρεοι παροχής του Δείκτη Ποιότητας Μ05 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίεςμέσω συστημάτων κινητών επικοινωνιών και ειδικότερα μέσω συστημάτων GSM/DCS, UMTS και TETRA.

Ο Δείκτης Ποιότητας Μ05 μετράται και παρουσιάζεται ανά διετία.

Μ06: Πιθανότητα εμπλοκής κλήσεων εικονοτηλεφωνίας

Ο Δείκτης Μ06 (Πιθανότητα εμπλοκής κλήσεων εικονοτηλεφωνίας) αφορά στην εμπλοκή κλήσεων εικονοτηλεφωνίας σε ένα δίκτυο κινητών επικοινωνιών και χαρακτηρίζει την προσβασιμότητα στην υπηρεσία εικονοτηλεφωνίας του δικτύου.

Ο Δείκτης Ποιότητας Μ06 εκφράζεται με ποσοστό και ορίζεται ως το πηλίκο του πλήθους των κλήσεων εικονοτηλεφωνίας στις οποίες παρουσιάστηκε εμπλοκή προς το συνολικό αριθμό των κλήσεων εικονοτηλεφωνίας για τις οποίες υπάρχει διαθεσιμότητα του δικτύου.

Τα αποτελέσματα των μετρήσεων παρουσιάζουν την πιθανότητα πρόσβασης, η οποία προκύπτει εάν από τη μονάδα αφαιρεθεί ο Δείκτης Ποιότητας Μ06.

Υπόχρεοι παροχής του Δείκτη Ποιότητας Μ06 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίεςμέσω συστημάτων κινητών επικοινωνιών UMTS.

Ο Δείκτης Ποιότητας Μ06 μετράται και παρουσιάζεται ανά διετία.

Μ07: Πιθανότητα διακοπής κλήσης εικονοτηλεφωνίας

Ο Δείκτης Μ07 (Πιθανότητα διακοπής κλήσης εικονοτηλεφωνίας) αφορά στην πιθανότητα τερματισμού (διακοπής) μιας επιτυχημένης προσπάθειας κλήσης εικονοτηλεφωνίας, για την οποία υπάρχει διαθεσιμότητα του δικτύου (ραδιοκάλυψη), για οποιοδήποτε λόγο εκτός από σκόπιμο τερματισμό του καλούντος ή του καλούμενου.

Ο Δείκτης Ποιότητας Μ07 εκφράζεται με ποσοστό και ορίζεται ως το πηλίκο του πλήθους των επιτυχημένων κλήσεων εικονοτηλεφωνίας που τερματίστηκαν για οποιοδήποτε λόγο εκτός από σκόπιμο τερματισμό του καλούντος ή του καλούμενου προς το συνολικό πλήθος κλήσεων εικονοτηλεφωνίας για τις οποίες υπάρχει διαθεσιμότητα του δικτύου.

Τα αποτελέσματα των μετρήσεων παρουσιάζουν την πιθανότητα διατήρησης της κλήσης, η οποία προκύπτει εάν από τη μονάδα αφαιρεθεί ο Δείκτης Ποιότητας Μ07.

Υπόχρεοι παροχής του Δείκτη Ποιότητας Μ07 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίεςμέσω συστημάτων κινητών επικοινωνιών UMTS.

Ο Δείκτης Ποιότητας Μ07 μετράται και παρουσιάζεται ανά διετία.

Μ08: Ποιότητα φωνής εικονοτηλεφωνίας

Ο Δείκτης Μ08 (Ποιότητα φωνής εικονοτηλεφωνίας) αποτελεί το δείκτη ποιότητας μετάδοσης από άκρο σε άκρο της φωνής της υπηρεσίας εικονοτηλεφωνίας. Εκφράζεται με έναν από τους ακόλουθους όρους κατηγοριών ποιότητας:

- πολύ καλή ποιότητα
- καλή ποιότητα
- μέση ποιότητα
- χαμηλή ποιότητα
- πολύ χαμηλή ποιότητα.

Υπόχρεοι παροχής του Δείκτη Ποιότητας Μ08 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίεςμέσω συστημάτων κινητών επικοινωνιών UMTS.

Ο Δείκτης Ποιότητας Μ08 μετράται και παρουσιάζεται ανά διετία.

Μ09: Ποιότητα βίντεο

Ο Δείκτης Μ09 (Ποιότητα βίντεο) αποτελεί το δείκτη ποιότητας μετάδοσης από άκρο σε άκρο του βίντεο της υπηρεσίας εικονοτηλεφωνίας. Εκφράζεται με έναν από τους ακόλουθους όρους κατηγοριών ποιότητας:

- πολύ καλή ποιότητα
- καλή ποιότητα
- μέση ποιότητα
- χαμηλή ποιότητα και
- πολύ χαμηλή ποιότητα.

Υπόχρεοι παροχής του Δείκτη Ποιότητας Μ09 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίεςμέσω συστημάτων κινητών επικοινωνιών UMTS.

Ο Δείκτης Ποιότητας Μ09 μετράται και παρουσιάζεται ανά διετία

Μ10: Χρόνος εγκατάστασης κλήσης εικονοτηλεφωνίας

Ο Δείκτης Μ10 (Χρόνος εγκατάστασης κλήσης εικονοτηλεφωνίας) εκφράζει το χρόνο στον οποίο αποκαθίσταται η κλήση εικονοτηλεφωνίας από τη στιγμή που ο τελικός χρήστης συμπληρώνει τον αριθμό του καλούμενου συνδρομητή.

Ο Δείκτης Ποιότητας Μ10 εκφράζεται σε δευτερόλεπτα και ορίζεται ως ο χρόνος από τη συμπλήρωση από τον καλούντα της πληροφορίας διεύθυνσης, δηλαδή του αριθμού τηλεφώνου του καλούμενου, μέχρι τη λήψη ειδοποίησης εγκατάστασης κλήσης εικονοτηλεφωνίας.

Υπόχρεοι παροχής του Δείκτη Ποιότητας Μ10 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίεςμέσω συστημάτων κινητών επικοινωνιών UMTS.

Ο Δείκτης Ποιότητας Μ10 μετράται και παρουσιάζεται ανά διετία

Μ11: Πιθανότητα αποτυχίας μεταφοράς δεδομένων http

Ο Δείκτης Μ11 (Πιθανότητα αποτυχίας μεταφοράς δεδομένων http) εκφράζει την πιθανότητα μη ολοκλήρωσης προσπάθειας μεταφοράς δεδομένων με βάση το πρωτόκολλο http.

Ο Δείκτης Ποιότητας Μ11 εκφράζεται με ποσοστό και ορίζεται ως το πηλίκο του πλήθους των μη ολοκληρωμένων προσπαθειών μεταφοράς δεδομένων με βάση το πρωτόκολλο http ως προς το συνολικό αριθμό επιτυχώς αρχικοποιημένων προσπαθειών.

Υπόχρεοι παροχής του Δείκτη Ποιότητας Μ11 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίες μέσω συστημάτων κινητών επικοινωνιών UMTS.

Ο Δείκτης Ποιότητας Μ11 μετράται και παρουσιάζεται ανά διετία.

Μ12: Μέσος ρυθμός δεδομένων http

Ο Δείκτης Μ12 (Μέσος ρυθμός δεδομένων http) εκφράζει το ρυθμό μεταφοράς δεδομένων με βάση το πρωτόκολλο http κατά τη διάρκεια μιας κλήσης, μετά την επιτυχημένη εγκατάσταση σύνδεσης δεδομένων.

Ο Δείκτης Ποιότητας Μ12 ορίζεται ως το πηλίκο του όγκου των δεδομένων που μεταφέρθηκαν προς το χρόνο που παρήλθε από τη στιγμή που ξεκίνησε η σύνδεση μέχρι τη στιγμή που παρελήφθη το ζητηθέν περιεχόμενο και μετράται σε Kbps.

Υπόχρεοι παροχής του Δείκτη Ποιότητας Μ12 είναι οι πάροχοι ηλεκτρονικών επικοινωνιών που παρέχουν υπηρεσίεςμέσω συστημάτων κινητών επικοινωνιών UMTS.

Ο Δείκτης Ποιότητας Μ12 μετράται και παρουσιάζεται ανά διετία.

ΚΕΦΑΛΑΙΟ 3: Εμπορικά Συστήματα Παρακολούθησης και Καταγραφής Ποιότητας Δικτύων Κινητών Επικοινωνιών

3.1 TEMS Portfolio

Η εταιρεία Ascom προσφέρει το TEMS Portfolio, ένα ολοκληρωμένο σύνολο λύσεων για έλεγχο, συγκριτική αξιολόγηση, αποτύπωση και ανάλυση της απόδοσης του δικτύου, διευκολύνοντας την ανάπτυξη, τη βελτιστοποίηση και τη συντήρηση δικτύων κινητών επικοινωνιών. Αν και επικεντρώνεται στην LTE τεχνολογία, είναι κατάλληλο για τα περισσότερα σύγχρονα είδη δικτύων επικοινωνιών. Συγκεκριμένα το TEMS Portfolio παρέχει:

- TEMS Discovery: εργαλείο για επεξεργασία των μετρούμενων δεδομένων από τη ραδιοεπαφή. Υποστηρίζει cdma2000, 1xEV, GSM/GPRS/EDGE, W-CDMA, HSDPA/HSUPA/HSPA+, TD-SCDMA, WiMAX, και LTE τεχνολογίες, χρησιμοποιεί ενσωματωμένους χάρτες Google και Microsoft Bing για απεικόνιση και ανάλυση των δεδομένων και εισάγει τα δεδομένα κυψέλης σε TEMS xml αρχεία ή αρχεία μορφής csv (comma separated values)/tab.
- TEMS Investigation: εργαλείο για συλλογή δεδομένων με αμάξι. Παρέχει πληροφορίες για την κάτω και άνω ζεύξη LTE δικτύων σε διάφορες ζώνες συχνοτήτων, τη ρυθμαπόδοση, τους χρόνους καθυστέρησης και τις γειτονικές κυψέλες. Χρησιμοποιείται για την επαλήθευση της συμπεριφοράς του τερματικού και της κάλυψης, χωρητικότητας, προσβασιμότητας και ακεραιότητας των κυψελών.
- TEMS Symphony: εργαλείο για αξιολόγηση επιδόσεων του LTE και έλεγχο δικτύου με απαιτήσεις ποιότητας υπηρεσίας (QoS).
- TEMS Pocket: εργαλείο χειρός για την επαλήθευση, τη διατήρηση και την αντιμετώπιση σφαλμάτων ασύρματων δικτύων ακόμη και σε εσωτερικούς χώρους. Υποστηρίζει μετρήσεις για τις τεχνολογίες GSM/GPRS, EDGE, W-CDMA, HSPA, CDMA και EV-DO και έχει τη δυνατότητα αποθήκευσής τους για περαιτέρω επεξεργασία.
- TEMS Automatic: εργαλείο για την επισκόπηση της ποιότητας του δικτύου όπως την αντιλαμβάνονται οι συνδρομητές κάθε χρονική στιγμή. Συλλέγει και αναλύει δεδομένα για το δίκτυο με αυτόματο τρόπο, ελέγχει την ποιότητα φωνής και τη μετάδοση δεδομένων και εντοπίζει σφάλματα και περιοχές συμφόρησης. Ακόμη συλλέγει δεδομένα από ανταγωνιστικά δίκτυα για συγκριτική αξιολόγηση.
- TEMS Monitor Master: εφαρμογή για τον έλεγχο, αναφορά και την αποτύπωση της απόδοσης των δραστηριοτήτων που μπορεί να επιτελέσει ένας συνδρομητής με το κινητό του τηλέφωνο. Οι δραστηριότητες αυτές περιλαμβάνουν αποστολή μηνυμάτων (SMS), VoIP κλήσεις, video συνεχούς ροής, υπηρεσίες HTML, Wap, i-mode και υλικό που μπορεί να κατεβάσει ο χρήστης, όπως εικόνες και ήχους κλήσης.
- TEMS Visualization: εργαλείο για ανάλυση δεδομένων που έχουν αποσπαστεί απευθείας από την υποδομή του δικτύου με βάση συντελεσμένα γεγονότα. Επιτρέπει τον άμεσο προσδιορισμό προβλημάτων που σχετίζονται με την ποιότητα υπηρεσίας και αφορούν συγκεκριμένους συνδρομητές ή συγκεκριμένα μοντέλα κινητών τερματικών. Κατάλληλο για επίλυση προβλημάτων στη ραδιοσυχνότητα και στη χωρητικότητα του δικτύου. Υποστηρίζει τεχνολογίες GSM/GPRS, W-CDMA, HSPA/HSPA+ και LTE.
- TEMS Support: πλήθος πακέτων ώστε ότι οι παγκόσμιοι πελάτες του TEMS Portfolio να επιλέγουν τα εργαλεία και τα επίπεδα υπηρεσιών που ταιριάζουν καλύτερα στις τεχνικές τους απαιτήσεις.

3.2 Nemo Network Testing Solutions

Η εταιρεία Anite προσφέρει ένα πακέτο προϊόντων για μέτρηση, έλεγχο, βελτιστοποίηση και ανάλυση της ποιότητας δικτύων κινητών επικοινωνιών. Τα εργαλεία που παρέχει είναι τα εξής:

- Nemo Outdoor: εργαλείο βασισμένο σε laptop που υποστηρίζει πολλαπλές ταυτόχρονες μετρήσεις και λειτουργεί για όλες τις τεχνολογίες και τα πρωτόκολλα. Οι μετρήσεις αποθηκεύονται σε μορφή ASCII Nemo για έλεγχο σφαλμάτων, ανάλυση και περαιτέρω επεξεργασία.
- Nemo Handy: εργαλείο βασισμένο σε smartphone με λειτουργικό Symbian, Windows Mobile ή Android. Χρησιμοποιείται για μετρήσεις ποιότητας υπηρεσίας (QoS) και ποιότητας εμπειρίας (QoE) της ασύρματης ραδιοεπαφής και των εφαρμογών του κινητού τερματικού.
- Nemo Invex: εργαλείο για καταγραφή μετρήσεων με αμάξι. Έχει τη δυνατότητα για μετρήσεις ποιότητας υπηρεσίας και ποιότητας εμπειρίας, συμπεριλαμβάνοντας αλγορίθμους για μετρήσεις ποιότητας φωνής και video.
- Nemo Autonomous: εργαλείο που εκτελεί αυτόματες, μεγάλης κλίμακας μετρήσεις ραδιοεπαφής. Από ένα σημείο πρόσβασης ο χρήστης μπορεί να έχει εικόνα και να ελέγξει πολλαπλά σταθερά ή κινητά σημεία εξέτασης, επιταχύνοντας έτσι τη διαδικασία απόκρισης σε σφάλματα του δικτύου, καθώς και τη βελτίωση των παρεχόμενων υπηρεσιών.
- Nemo FSR1: ψηφιακό δέκτης σάρωσης για μετρήσεις σε δίκτυα LTE, WCDMA, HSDPA, GSM, CDMA και EVDO. Δίνει τη δυνατότητα αλλαγής της ραδιοεπαφής για αναβάθμιση στο κατάλληλο δίκτυο.
- Nemo Analyze: εργαλείο για ανάλυση, συγκριτική αξιολόγηση, αυτόματη επίλυση σφαλμάτων και στατιστική επεξεργασία των μετρήσεων που προέρχονται από τα προηγούμενα εργαλεία. Είναι δυνατόν να συγκρίνει δεδομένα από άλλους παρόχους, άλλες τεχνολογίες, άλλα χρονικά πλαίσια και να οπτικοποιήσει τα αποτελέσματα σε μία μοναδική αναφορά.

3.3 SwissQual Network Benchmarking, Optimization and Service Monitoring

Η εταιρεία SwissQual έχει αναπτύξει τρία συστήματα για αξιολόγηση, βελτιστοποίηση και απεικόνιση της ποιότητας δικτύων κινητών επικοινωνιών. Τα συστήματα αυτά με τα εργαλεία τους είναι:
Diversity

- Diversity Benchmarker: εργαλείο για μετρήσεις ποιότητας δικτύου με αμάξι σε διάφορα είδη περιβάλλοντος, σχεδιασμένο να αντέχει σε κραδασμούς και μεγάλο εύρος θερμοκρασιών.
- Diversity Explorer: εργαλείο για αυτόματη, αυτόνομη και από απόσταση καταγραφή θέσης και δεδομένων ποιότητας υπηρεσιών, με δυνατότητα απεικόνισης σε πραγματικό χρόνο.

- Diversity Ranger: φορητό εργαλείο για συγκριτική ανάλυση επιδόσεων. Κατάλληλο για δίκτυα 4ης γενιάς.
- Diversity Optimizer Pro: εργαλείο για επίλυση σφαλμάτων και βελτιστοποίηση υπηρεσιών και εφαρμογών κινητών τερματικών που λειτουργούν κυρίως σε LTE δίκτυα, αλλά και σε δίκτυα προηγούμενων γενεών.

QualiPoc

Η οικογένεια προϊόντων QualiPoc αποτελεί μια ελαφρύτερη εκδοχή της οικογένειας Diversity.

- QualiPoc Handheld: φορητό εργαλείο χειρός για απεικόνιση επίδοσης και βελτιστοποίηση υπηρεσιών φωνής, βίντεο και δεδομένων.
- QualiPoc Freerider: εκτελεί συγκριτική αξιολόγηση επιδόσεων σε μικρές περιοχές. Κατάλληλο για απεικόνιση της κάλυψης ανταγωνιστικών δικτύων.
- QualiPoc Static: χρησιμοποιείται σε συγκεκριμένα σημεία ενδιαφέροντος για αποτύπωση επιδόσεων κάθε χρονική στιγμή.
- QualiPoc Mobile: αυτόνομο εργαλείο για συνεχή αποτύπωση επιδόσεων δικτύων σε συνθήκες κινητικότητας που εγκαθίσταται σε διάφορους τύπους οχημάτων, όπως ταξί, λεωφορεία και μοτοσυκλέτες.

NetQual

- NQDI: σύστημα για επεξεργασία των μετρήσεων που έχουν προέρθει από τα εργαλεία των δύο προηγούμενων οικογενειών προϊόντων.
- NQView: χρησιμοποιείται για εμφάνιση και διερεύνηση των μετρήσεων σε πραγματικό χρόνο.
- NQWeb: εργαλείο προσπελάσιμο από το διαδίκτυο για έλεγχο, οργάνωση, απεικόνιση και δημιουργία αναφορών των δεδομένων που έχουν συλλεχθεί.

ΚΕΦΑΛΑΙΟ 4: Το Προτεινόμενο Σύστημα

4.1 Αρχιτεκτονική

4.1.1 Γενική Αρχιτεκτονική

Το προτεινόμενο σύστημα αποτελείται από δύο κύρια μέρη: την εφαρμογή του κινητού τερματικού και τη δομή του εξυπηρετητή (server). Πιο συγκεκριμένα, με την εφαρμογή του κινητού γίνεται καταγραφή της θέσης του και της λαμβανόμενης ισχύς σήματος και μεταφορά των δεδομένων αυτών στον server. Από τη μεριά του ο server αποθηκεύει σε μια βάση όλα τα δεδομένα και παρέχει τη δυνατότητα απεικόνισής τους σε χάρτη. Το όνομα της εφαρμογής, αλλά και της σελίδας για τη διαχείριση και την παρουσίαση των μετρήσεων είναι SignalLocation.

4.1.2. Αρχιτεκτονική στην πλευρά του εξυπηρετητή

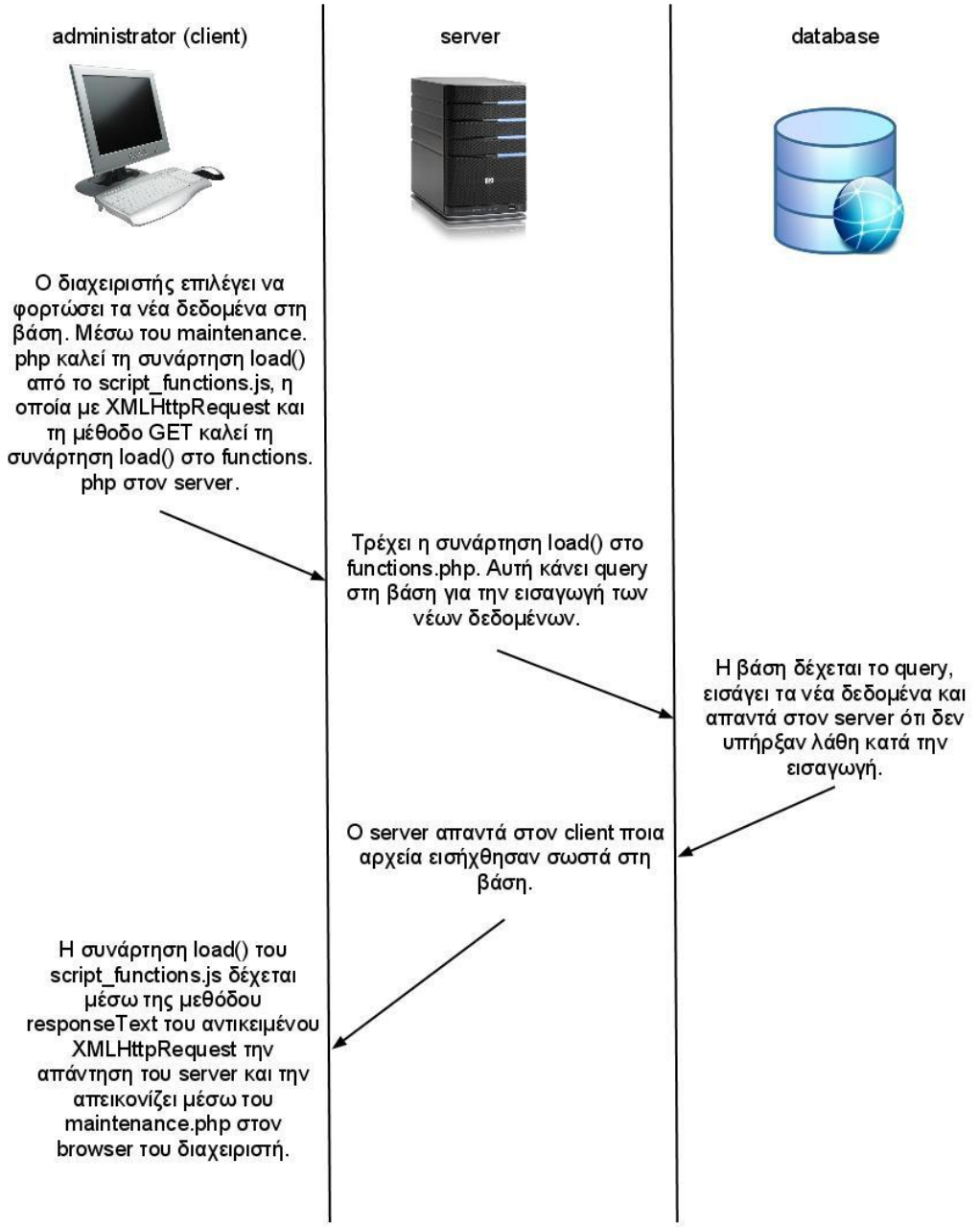
Σημαντικό κομμάτι της δομής του εξυπηρετητή είναι η βάση δεδομένων. Εκεί αποθηκεύονται όλες οι καταγραφές των κινητών τερματικών κι από εκεί αντλούνται οι απαραίτητες πληροφορίες για την παρουσίαση των μετρήσεων.
Ο εξυπηρετητής δέχεται τα αρχεία δεδομένων που του αποστέλλουν τα κινητά τερματικά και τα αποθηκεύει σε έναν προσωρινό φάκελο. Αποδεκτά αρχεία δεδομένων που θα εισαχθούν στη βάση έχουν επέκταση txt ή xml και μέγεθος μικρότερο από 2MB.
Στη συνέχεια ο διαχειριστής του συστήματος έχει τη δυνατότητα να φορτώσει στη βάση δεδομένων τις νέες καταγραφές, καθώς και να διαχειριστεί όλα τα αρχεία που έχουν ανέβει στον server (αποδεκτά ή μη). Πριν την εισαγωγή τους στη βάση, οι νέες καταγραφές υπόκεινται σε επεξεργασία ώστε όλα τα γεωγραφικά σημεία που θα υπάρχουν στη βάση να απέχουν τουλάχιστον 10 μέτρα μεταξύ τους.
Επιπλέον λειτουργία του εξυπηρετητή είναι η απάντηση σε αιτήματα τελικών χρηστών (clients) για απεικόνιση των δεδομένων της βάσης σε χάρτη.

4.1.3 Αρχιτεκτονική στην πλευρά του τερματικού

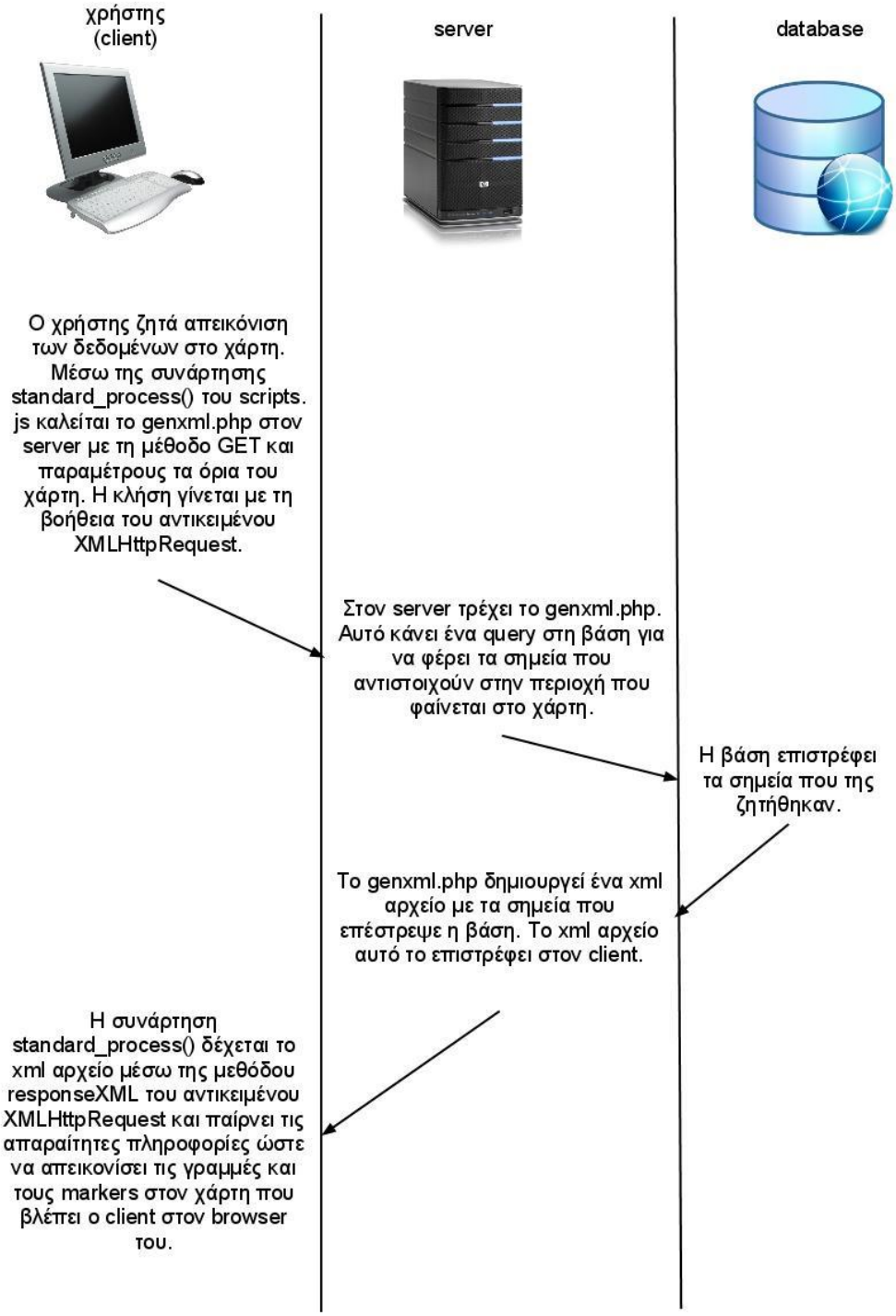
Τα τερματικά που είναι συμβατά με την εφαρμογή για την καταγραφή των δεδομένων είναι τα τερματικά με λειτουργικό σύστημα android μιας και η εφαρμογή έχει γραφτεί για την πλατφόρμα android με τη γλώσσα προγραμματισμού java. Τα τερματικά αυτά είναι απαραίτητο να έχουν δέκτη κινητής τηλεφωνίας , κάρτα μνήμης για αποθήκευση των δεδομένων καθώς και δέκτη gps ή/και δυνατότητα σύνδεσης σε δίκτυο wifi. Το τερματικό μέσω της εφαρμογής κάνει καταγραφή της θέσης, του σήματος καθώς και κάποιων άλλων παραμέτρων και τα αποθηκεύει σε μορφή αρχείου .txt σε έναν φάκελο στην κάρτα μνήμης. Η εφαρμογή αναλαμβάνει να κάνει το upload των αρχείων στον εξυπηρετητή αμέσως μόλις βρει σύνδεση δικτύου wifi χωρίς να χρειαστεί να επέμβει ο χρήστης σε αυτό.

4.1.4 Information Flow Diagrams

Για την πλευρά του διαχειριστή έχουμε:



Ενώ για την πλευρά του τελικού χρήστη:



Τέλος, για το κινητό τερματικό:

4.1.5 Δομές δεδομένων

Τα στοιχεία που καταγράφονται από την εφαρμογή και τελικά αποθηκεύονται στη βάση δεδομένων είναι τα εξής:

- sim operator: αποτελείται από τον κωδικό της χώρας (mcc: mobile country code) και τον κωδικό του παρόχου κινητής τηλεφωνίας (mnc: mobile network code). Για την Ελλάδα mcc = 202 και για τις τρεις εταιρείες κινητής τηλεφωνίας (η Q-telecoms έχει συγχωνευτεί με τη Wind)

Name	Cosmote	Vodafone	Wind	Q-telecoms
mnc	01	05	10	09

- location area code: μοναδικός αριθμός που αντιστοιχεί σε ένα σύνολο σταθμών βάσης (BTS: base transceiver station)
- cell identity: μοναδικός αριθμός για αναγνώριση της κυψέλης στην οποία έγινε η καταγραφή. Κάθε κυψέλη αντιστοιχεί σε έναν σταθμό βάσης ή σε έναν τομέα ενός σταθμού βάσης
- network type: το είδος του δικτύου στο οποίο έγινε η μέτρηση

Αντιστοιχία τιμών με είδος δικτύου:

GPRS	EDGE	UMTS	CDMA	HSDPA	HSUPA	HSPA	LTE	HSPAP
1	2	3	4	8	9	10	13	15

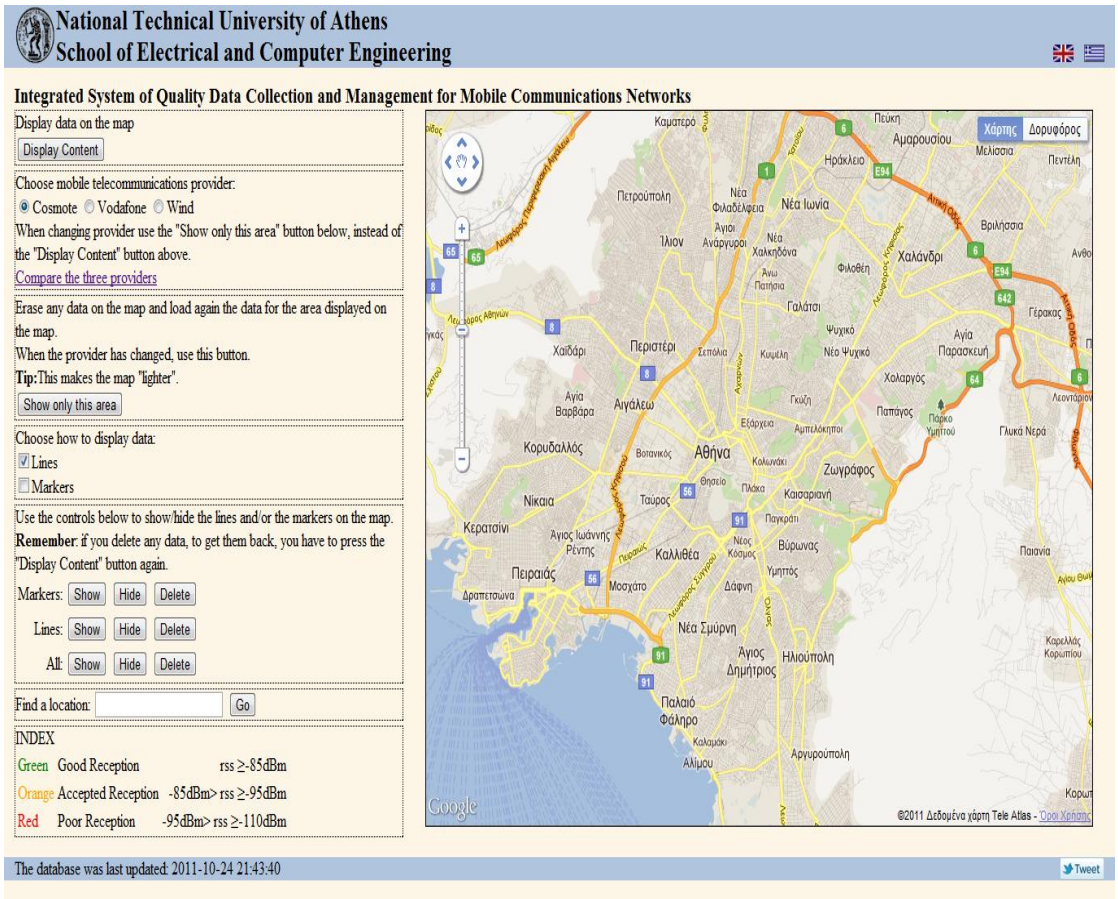
- location provider: τρόπος με τον οποίο βρέθηκε η θέση του κινητού τερματικού. Είτε μέσω GPS, είτε μέσω του δικτύου.
- latitude: το γεωγραφικό πλάτος της θέσης της καταγραφής, μετρημένο σε μοίρες με ακρίβεια 8 δεκαδικών ψηφίων
- longitude: το γεωγραφικό μήκος της θέσης της καταγραφής, μετρημένο σε μοίρες με ακρίβεια 8 δεκαδικών ψηφίων
- altitude: το υψόμετρο της θέσης της καταγραφής, μετρημένο σε μέτρα με ακρίβεια ενός δεκαδικού ψηφίου
- speed: η ταχύτητα του κινητού τερματικού τη στιγμή της καταγραφής, μετρημένη σε μέτρα ανά δευτερόλεπτο, με ακρίβεια δύο δεκαδικών ψηφίων
- accurancy: η ακρίβεια της θέσης της καταγραφής, μετρημένη σε μέτρα
- received signal strength: η λαμβανόμενη ισχύς σήματος, μετρημένη σε dBm. Όπου χρειάζεται επεξεργασία της ισχύος γίνεται πρώτα μετατροπή σε mW.
- date: η ακριβής ημερομηνία της στιγμής της καταγραφής, δηλαδή έτος/μήνας/μέρα ώρα/λεπτά/δευτερόλεπτα
- international mobile equipment identity (imei): το imei είναι ένας αριθμός, μοναδικός για κάθε κινητό τερματικό, από τον οποίο μπορούμε να αντλήσουμε πληροφορίες για τον κατασκευαστή του τερματικού

Ας σημειωθεί ότι επιπλέον στη βάση δεδομένων υπάρχει και ένα πεδίο id με έναν μοναδικό αύξων αριθμό για κάθε εγγραφή, ο οποίος εισάγεται αυτόματα από τη βάση. Επίσης, εγγραφές που είτε τους λείπει κάποιο από τα latitude ή longitude, είτε έχουν μηδενικό rss (received signal strength), δηλαδή δεν έχει γίνει μέτρηση της ισχύος, δεν εισάγονται στη βάση, αφού αυτά τα τρία στοιχεία αποτελούν το λόγο ύπαρξης του όλου εγχειρήματος. Τέλος, δεν εισάγονται εγγραφές που στο πεδίο accurancy (acc) έχουν τιμή μεγαλύτερη από 100 μέτρα.

4.2 Επίδειξη - Σενάρια Χρήσης

4.2.1 Για τον τελικό χρήστη

Όταν ο χρήστης πληκτρολογήσει στον browser του “url του εκάστοτε server που φιλοξενεί την εφαρμογή”/SignalLocation, θα δει την επόμενη ιστοσελίδα:



Εδώ υπάρχουν τέσσερα δομικά στοιχεία

- Η επικεφαλίδα στο γαλάζιο πλαίσιο πάνω. Στο δεξιό άκρο υπάρχει η δυνατότητα επιλογής της γλώσσας.
- Η αριστερή στήλη των στοιχείων ελέγχου. Εδώ βρίσκονται διάφορα κουμπιά για εμφάνιση και επιλογές εμφάνισης των δεδομένων στο χάρτη, καθώς και σύντομες οδηγίες χρήσης των κουμπιών αυτών.
- Δεξιά βρίσκεται η περιοχή του χάρτη. Πρόκειται για έναν τυπικό Google χάρτη, με όλα τα βασικά στοιχεία ελέγχου, όπως η δυνατότητα για εστίαση (zoom-in και zoom-out), η μετακίνηση του χάρτη και η εναλλαγή μεταξύ προβολής χάρτη και δορυφόρου.
- Στο γαλάζιο πλαίσιο στο κάτω μέρος, φαίνεται η ακριβής ημερομηνία της τελευταίας τροποποίησης της βάσης δεδομένων. Η μορφή εμφάνισης της ημερομηνίας είναι έτος-μήνας-μέρα ώρα:λεπτά:δευτερόλεπτα.

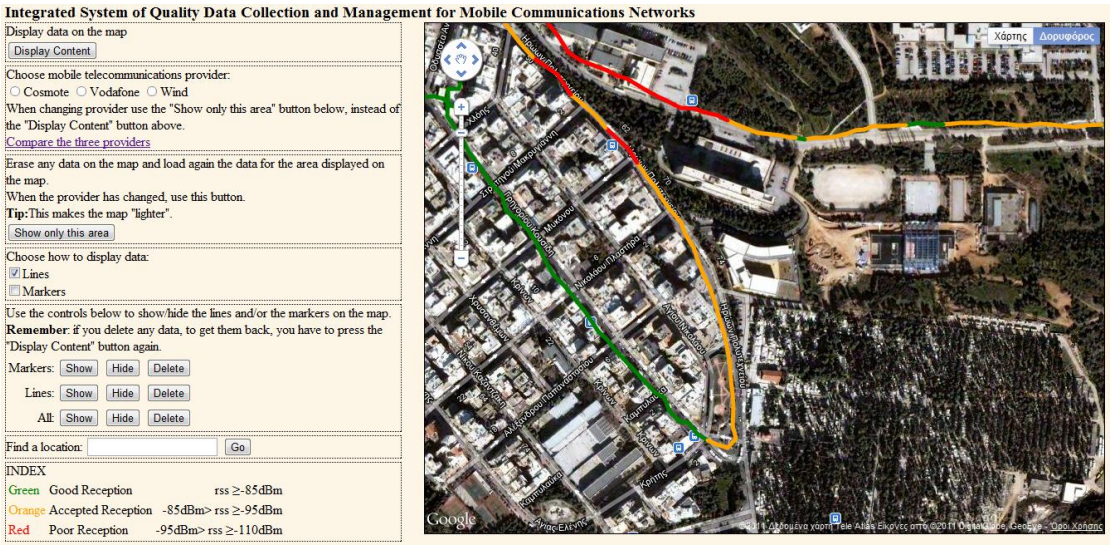
Οι δυνατότητες των κουμπιών στην αριστερή στήλη είναι:

- Πατώντας το “Display Content” ο χρήστης εμφανίζει όλες τις μετρήσεις που αντιστοιχούν στην περιοχή που φαίνεται εκείνη τη στιγμή στο χάρτη.
- Η επιλογή του παρόχου προς εμφάνιση γίνεται ακριβώς από κάτω. Για μία συγκριτική εμφάνιση και των τριών παρόχων ο χρήστης μπορεί να πατήσει

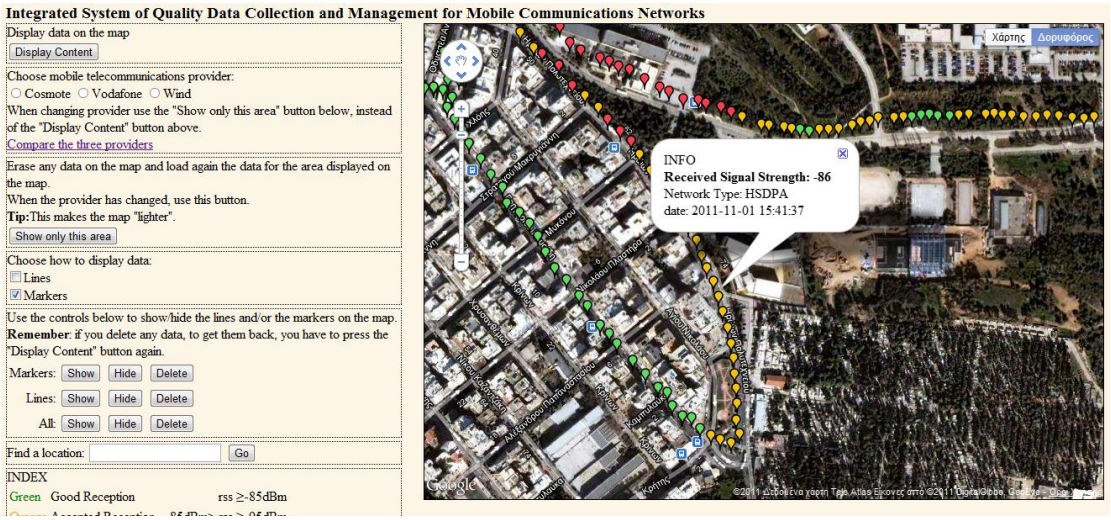
στο “Compare the three providers”, με αποτέλεσμα να ανακατευθυνθεί σε μια νέα ιστοσελίδα που θα εξεταστεί παρακάτω.

- Υπάρχουν δύο τρόποι εμφάνισης των δεδομένων: με γραμμές και με σημαίες. Ο χρήστης μπορεί να επιλέξει έναν από τους δύο τρόπους (ή και τους δύο) κάνοντας κλικ στο “Lines” ή “Markers” (ή και στα δύο).

Οι γραμμές προσφέρουν καλύτερη εποπτική εικόνα της ποιότητας του δικτύου και είναι κατάλληλες για μικρή εστίαση στον χάρτη.



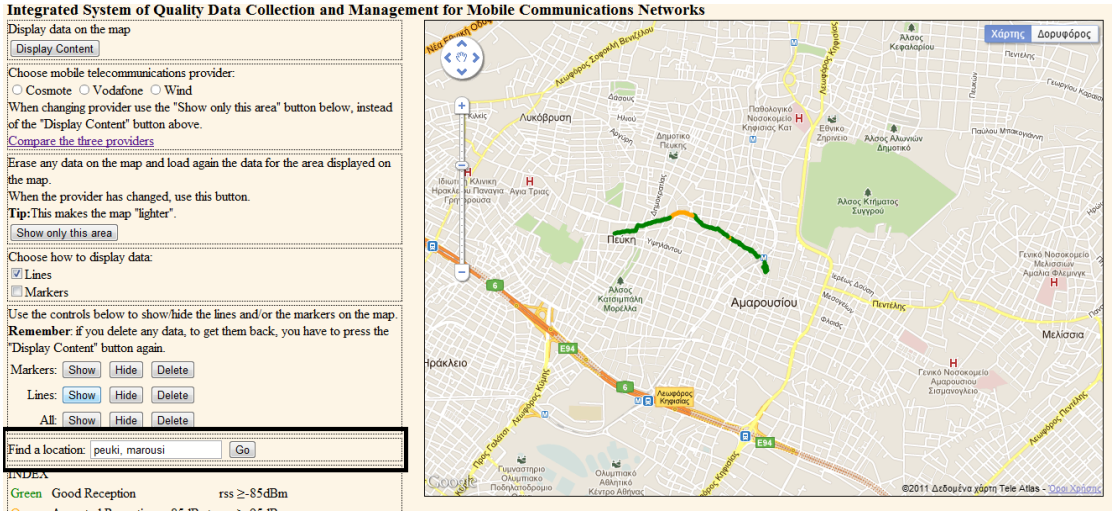
Από την άλλη μεριά, με τις σημαίες μπορεί ο χρήστης να κάνει κλικ πάνω τους για να δει κάποιες βασικές πληροφορίες, όπως η ακριβής τιμή της λαμβανόμενης ισχύς του σήματος, ο τύπος του δικτύου στον οποίο έγινε η μέτρηση και η ακριβής ημερομηνία της μέτρησης. Οι σημαίες είναι καταλληλότερες όταν ο χρήστης επιθυμεί να ελέγξει την ποιότητα του δικτύου για μία μεμονωμένη περιοχή με μεγάλη εστίαση στον χάρτη.



- Πατώντας το κουμπί “Show only this area” ο χρήστης διαγράφει όλα τα δεδομένα που έχει μέχρι εκείνη τη στιγμή φέρει στο χάρτη, ακόμη και για περιοχές που δεν είναι ορατές στην τρέχουσα απεικόνιση και εμφανίζει τις μετρήσεις που αντιστοιχούν στην τρέχουσα περιοχή του χάρτη. Όταν ο χρήστης αποφασίσει να αλλάξει πάροχο προς απεικόνιση, είναι απαραίτητο

να χρησιμοποιήσει αυτό το κουμπί, γιατί αλλιώς τα δεδομένα του νέου παρόχου θα εμφανίζονται μαζί με τα δεδομένα που υπάρχουν από τον προηγούμενο, χωρίς κάποια διάκριση μεταξύ τους. Επιπροσθέτως, επειδή η ύπαρξη πολλών μετρήσεων κάνει την απόκριση του χάρτη πιο αργή, π.χ. στη μετακίνησή του ή στη φόρτωση νέων περιοχών, αποτελεί καλή τακτική ο χρήστης να διαγράφει τα δεδομένα που δεν τον ενδιαφέρουν πια και να εμφανίζει μόνο αυτά για την περιοχή που τον αφορά, με τη βοήθεια του “Show only this area”.

- Ακόμη και μετά τη φόρτωση των δεδομένων στο χάρτη, ο χρήστης έχει τη δυνατότητα επιλογής εμφάνισης είτε με γραμμές είτε με σημαίες είτε και με τους δύο τρόπους, καθώς και διαγραφή αυτών. Αυτό γίνεται με τα στοιχεία ελέγχου “Show”, “Hide”, “Delete”. Ας σημειωθεί ότι εάν ο χρήστης διαγράψει τις γραμμές ή τις σημαίες ή και τα δύο με το κουμπί “Delete”, τότε δεν έχει τη δυνατότητα επαναφοράς τους με το κουμπί “Show”. Θα πρέπει να ξαναφέρει τις μετρήσεις πατώντας είτε το κουμπί “Display Content”, είτε το κουμπί “Show only this area”.
- Εκτός από χειροκίνητη αναζήτηση, σύροντας το χάρτη, ο χρήστης μπορεί επιπλέον να εμφανίσει διάφορες γεωγραφικές περιοχές στο χάρτη πληκτρολογώντας την ονομασίας τους στο πλαίσιο αναζήτησης που υπάρχει στην ιστοσελίδα και πατώντας enter ή το κουμπί “Go”.

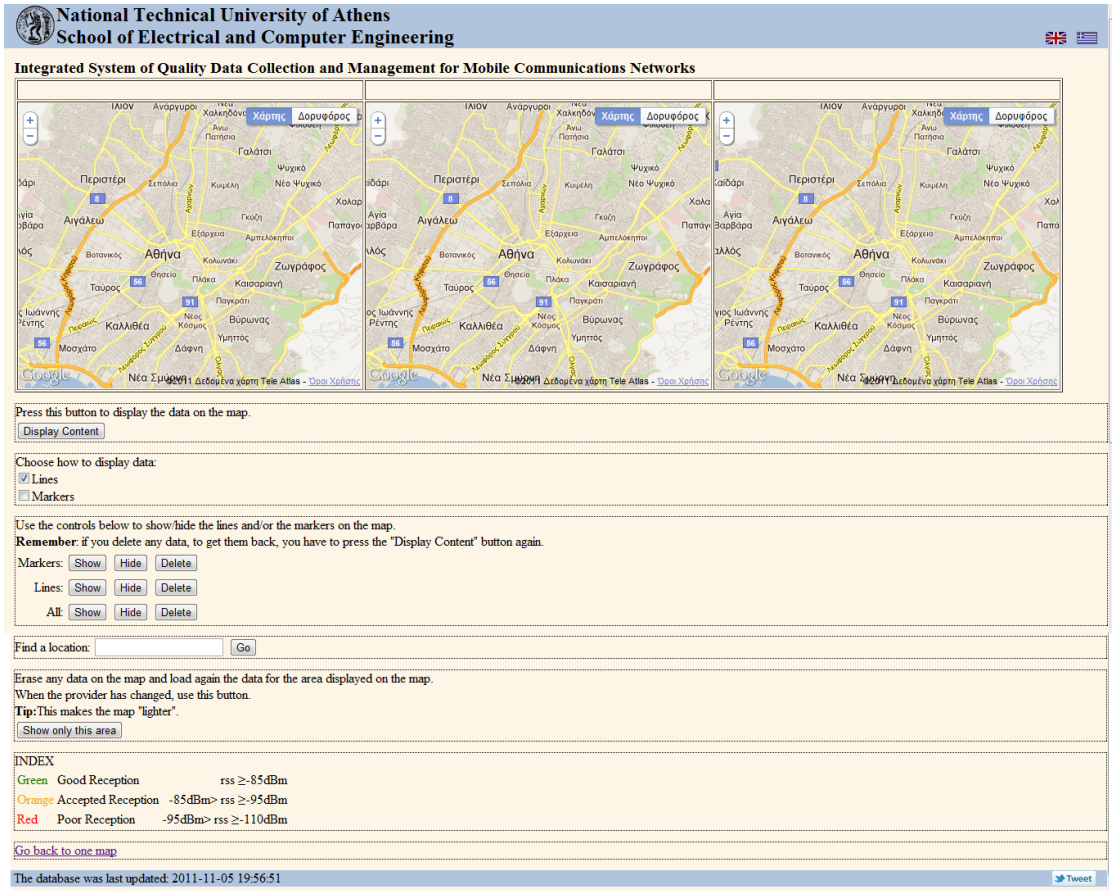


- Στο τέλος της ιστοσελίδας υπάρχει ένα υπόμνημα με την αντιστοίχηση των χρωμάτων, με τα οποία εμφανίζονται οι μετρήσεις στο χάρτη, με το χαρακτηρισμό της λήψης και τα εύρη τιμών σε dBm της λαμβανόμενης ισχύος σήματος (rss) σύμφωνα με την ΕΕΤΤ (Εθνική Επιτροπή Τηλεπικοινωνιών και Ταχυδρομείων).

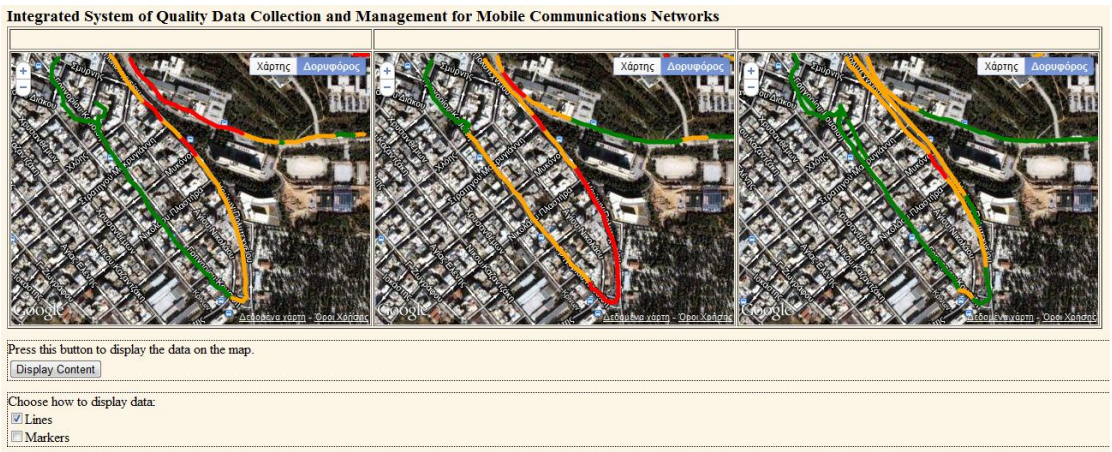
INDEX

Green	Good Reception	rss ≥-85dBm
Orange	Accepted Reception	-85dBm> rss ≥-95dBm
Red	Poor Reception	-95dBm> rss ≥-110dBm

Πατώντας στο “Compare the three providers” ο χρήστης κατευθύνεται σε μία νέα σελίδα:



Η σελίδα αυτή έχει την ίδια περίπου δομή με την προηγούμενη, εκτός του ότι εμφανίζονται τρεις χάρτες, ένας για κάθε πάροχο που δραστηριοποιείται στην Ελλάδα και του ότι τα στοιχεία ελέγχου βρίσκονται κάτω από τους χάρτες.
Εδώ ο χρήστης έχει τη δυνατότητα ταυτόχρονης απεικόνισης μετρήσεων για την ίδια περιοχή και από τους τρεις παρόχους, πατώντας το “Display Content”.



Ο χρήστης έχει τις ίδιες δυνατότητες με πριν κι επιπλέον μπορεί να γυρίσει στην ιστοσελίδα με τον ένα χάρτη πατώντας στο κουμπί “Go back to one map”.

INDEX		
Green	Good Reception	rss ≥-85dBm
Orange	Accepted Reception	-85dBm> rss ≥-95dBm
Red	Poor Reception	-95dBm> rss ≥-110dBm

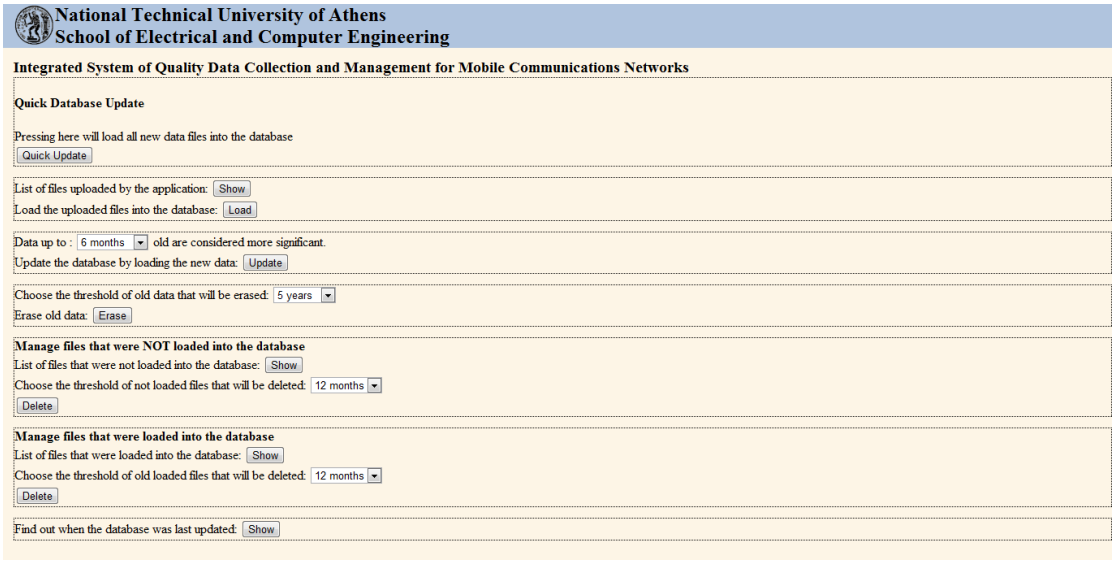
Go back to one map

The database was last updated: 2011-11-05 19:56:51

4.2.2 Για τον διαχειριστή

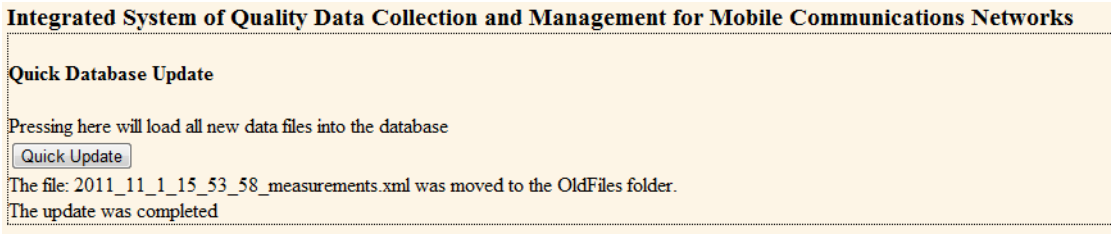
Ο διαχειριστής πληκτρολογεί στον browser: “url του εκάστοτε server που φιλοξενεί την εφαρμογή”/SignalLocation/maintenance.php και λαμβάνει την εξής σελίδα:

Ο λόγος ύπαρξης αυτής της σελίδας είναι να μην έχει ο κάθε χρήστης πρόσβαση σε στοιχεία ελέγχου που επηρεάζουν τη λειτουργία της βάσης δεδομένων, καθώς και των αρχεία που υπάρχουν στον εξυπηρετητή. Αφού εισάγει το σωστό όνομα χρήστη και κωδικό πρόσβασης και πατήσει “Login” κατευθύνεται στην επόμενη σελίδα:

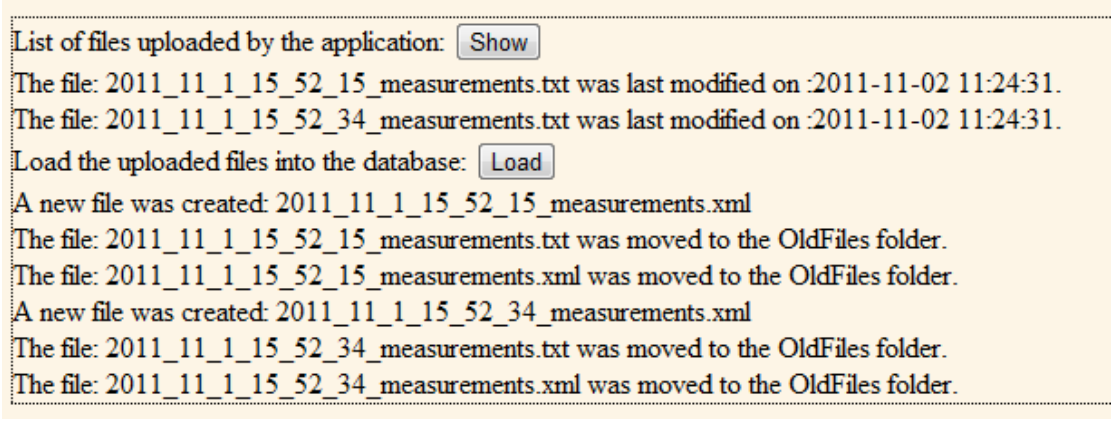


Εδώ βρίσκονται διάφορα κουμπιά για φόρτωση δεδομένων στη βάση και διαχείριση αρχείων που υπάρχουν στον εξυπηρετητή, καθώς και σύντομες οδηγίες χρήσης των κουμπιών αυτών. Πιο συγκεκριμένα:

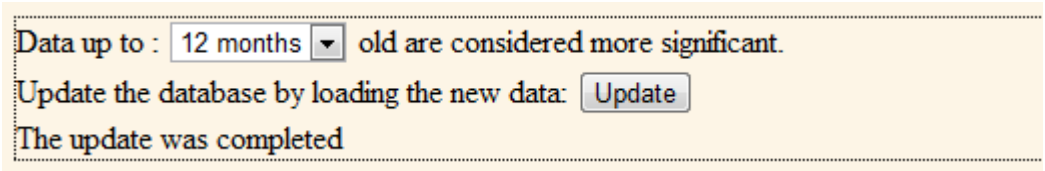
- Πατώντας το “Quick Update” ο διαχειριστής μπορεί να φορτώσει τα τυχόν αρχεία καταγραφής μετρήσεων που έχουν ανέβει στον εξυπηρετητή από τα κινητά τερματικά στη βάση δεδομένων, χωρίς να ασχοληθεί με περαιτέρω επιλογές. Όταν η διαδικασία τελειώσει θα ενημερωθεί με το ποια αρχεία επηρεάστηκαν και με το μήνυμα “The update was completed”.



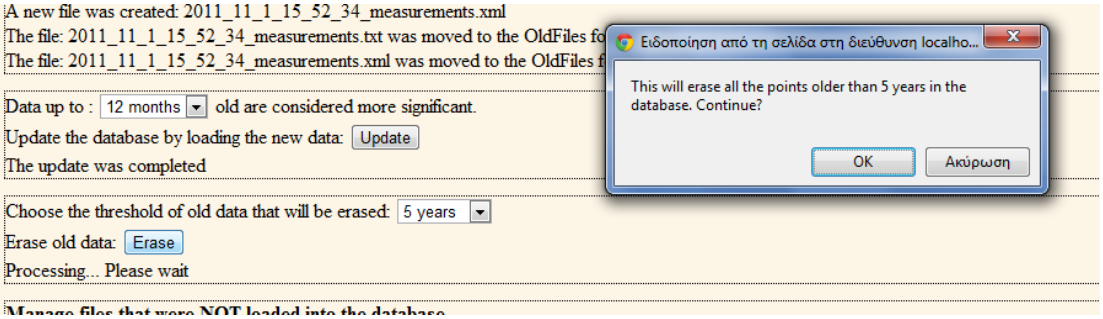
- Παρακάτω μπορεί να πατήσει το κουμπί “Show” για να εμφανίσει ποια αρχεία έχουν ανέβει από τα κινητά τερματικά στον εξυπηρετητή. Με το κουμπί “Load” φορτώνει τα αρχεία αυτά στη βάση. Ωστόσο οι νέες αυτές μετρήσεις δεν είναι ακόμη έτοιμες για απεικόνιση. Πρέπει να γίνει και ανανέωση της βάσης δεδομένων.



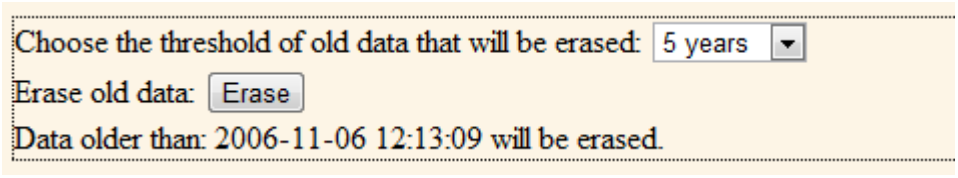
- Η ανανέωση της βάσης γίνεται με το κουμπί “Update”. Ο διαχειριστής έχει τη δυνατότητα επιπλέον να καθορίσει το χρονικό κατώφλι για το οποίο οι μετρήσεις θεωρούνται πρόσφατες ή όχι. Αυτό έχει σημασία για τον αλγόριθμο που ανανεώνει τη βάση δεδομένων, καθώς δίνει περισσότερο βάρος στις πιο πρόσφατες μετρήσεις.



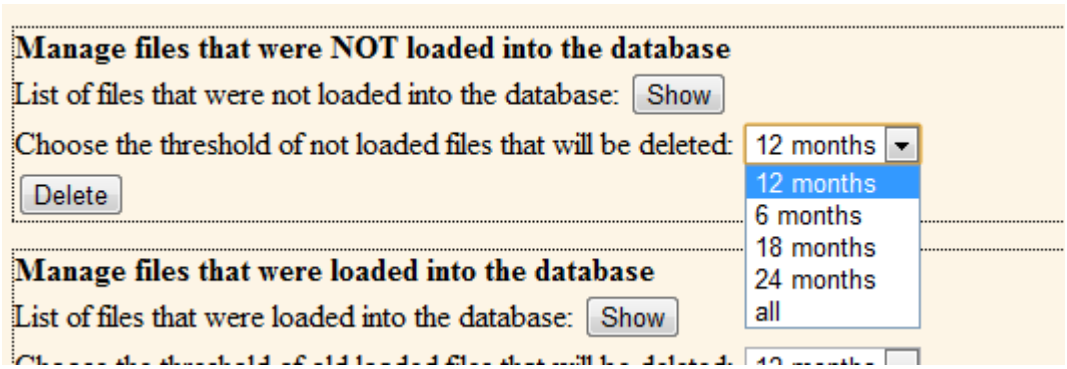
- Στη συνέχεια ο διαχειριστής μπορεί να σβήσει παλιά δεδομένα με το κουμπί “Erase”, καθορίζοντας πάλι το χρονικό κατώφλι. Πριν γίνει η διαγραφή, ο διαχειριστής πρέπει να επιβεβαιώσει την πρόθεσή του.



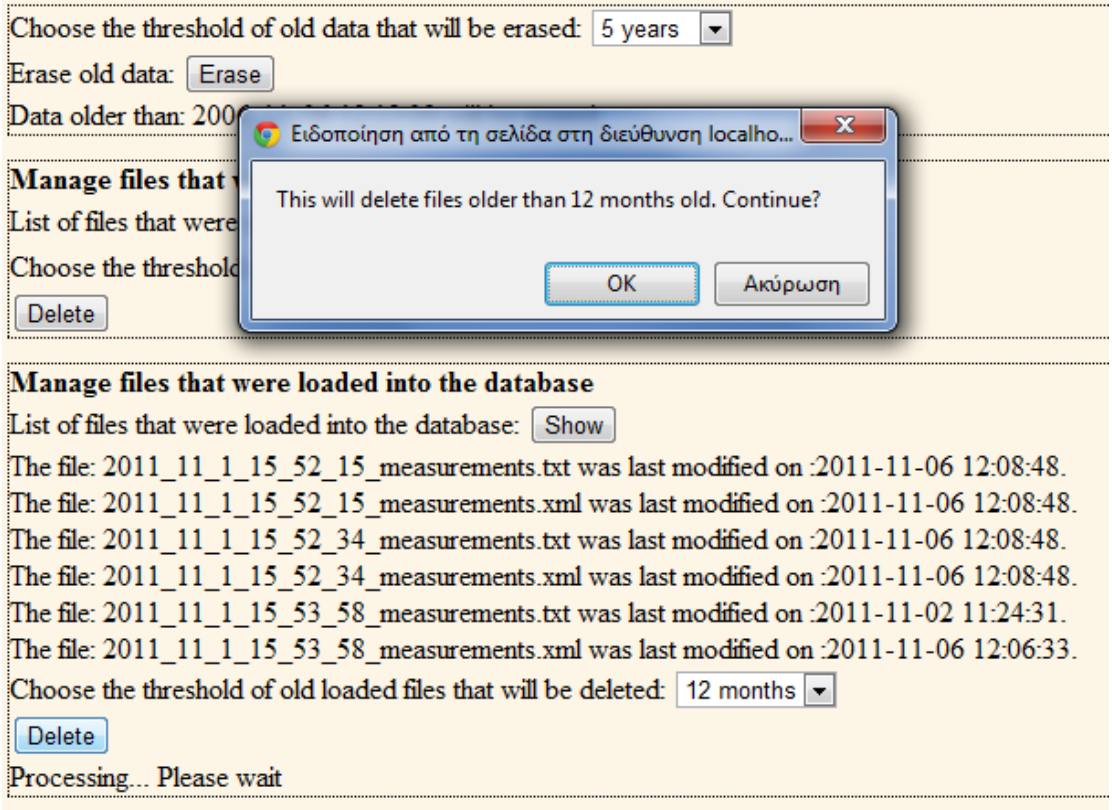
Ο διαχειριστής ενημερώνεται για τη διαγραφή των δεδομένων.



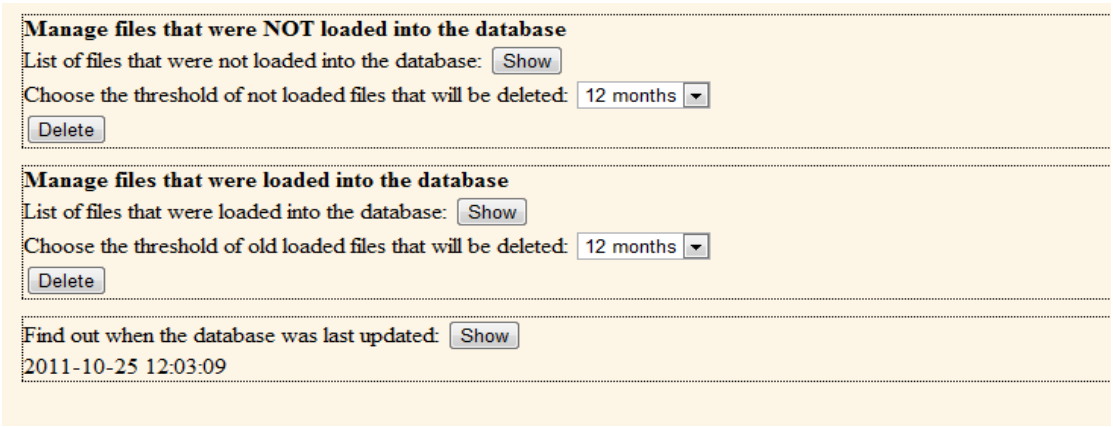
- Η διαχείριση των αρχείων που απέτυχαν τα κριτήρια για φόρτωση στη βάση γίνεται με το παρακάτω πλαίσιο ελέγχου. Οι δυνατότητες είναι εμφάνιση των εν λόγω αρχείων, επιλογή του χρονικού κατωφλίου για τη διαγραφή τους και τέλος διαγραφή των αρχείων σύμφωνα με το κατώφλι.



- Ομοίως με πριν ο διαχειριστής μπορεί να δει και να διαγράψει τα αρχεία που έχουν τελικά φορτωθεί στη βάση. Ας σημειωθεί ότι πριν γίνει διαγραφή των αρχείων, ο διαχειριστής πρέπει να επιβεβαιώσει αυτή του την πρόθεση.



- Στο τέλος της σελίδας πατώντας το “Show” υπάρχει η δυνατότητα εμφάνισης της ακριβής ημερομηνίας της τελευταίας τροποποίησης της βάσης δεδομένων.



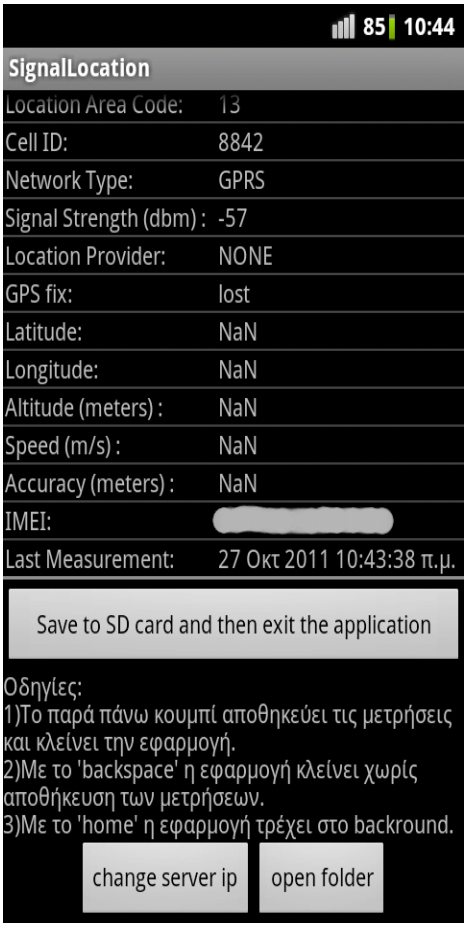
4.2.3 Για το κινητό τερματικό

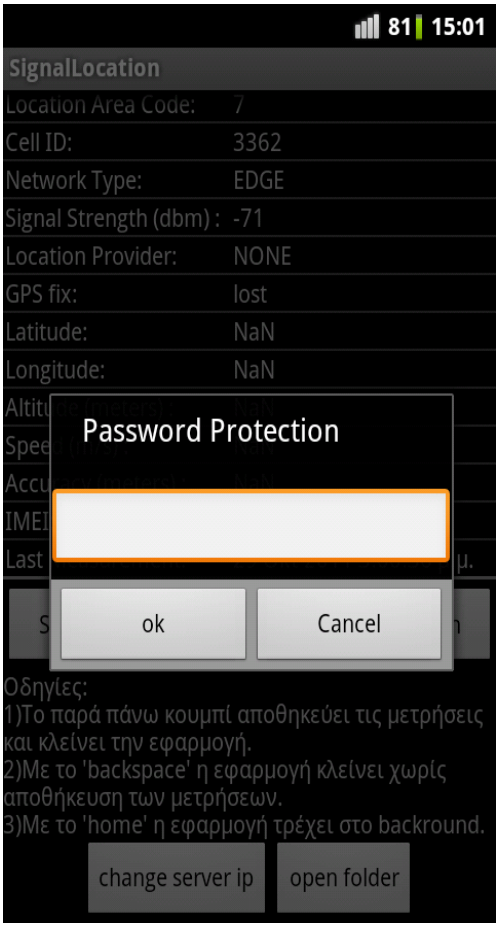
Η εφαρμογή είναι πολύ απλή στη λειτουργία της. Αρκεί ο χρήστης να την ανοίξει και αμέσως ξεκινάει η καταγραφή των μετρήσεων. Ανοίγοντας την εφαρμογή βλέπει κανείς το ui (user interface) το οποίο εμφανίζει κάποιες χρήσιμες πληροφορίες καθώς και τις τωρινές τιμές των μετρήσεων. Φυσικά για τον εντοπισμό της θέσης απαιτείται σύνδεση σε δορυφόρο με gps ή σύνδεση στο διαδίκτυο μέσω wifi. Γι'αυτό το λόγο υπάρχουν δύο κουμπιά στο πάνω μέρος της τα οποία αλλάζουν την κατάσταση του gps και του wifi μέσα από την εφαρμογή ώστε να μην χρειαστεί ο χρήστης να βγει από αυτή αν δεν το επιθυμεί.

Ο χρήστης έχει τη δυνατότητα να αφήσει την εφαρμογή να τρέχει στο παρασκήνιο καταγράφοντας μετρήσεις και να κάνει οποιαδήποτε άλλη λειτουργία στο κινητό του. Αυτό μπορεί να επιτευχθεί πατώντας το κουμπί "home" που βρίσκεται σε όλα τα android κινητά. Για την αποθήκευση των μετρήσεων είναι αναγκαίο να ξανανοίξει την εφαρμογή, εφόσον αυτή βρίσκεται στο παρασκήνιο και να πατήσει το κουμπί "save to SD card and then exit the application". Με αυτό τον τρόπο θα αποθηκευτούν οι μετρήσεις και θα κλείσει η εφαρμογή. Όλες οι μετρήσεις σώζονται ως αρχεία .txt στον φάκελο "mymeasurements" που βρίσκεται στην κάρτα SD. Αν δεν επιθυμεί να κάνει καταγραφή των μετρήσεων που μόλις πήρε αρκεί να πατήσει το κουμπί "back" που βρίσκεται σε όλα τα android κινητά. Το upload των αποθηκευμένων αρχείων γίνεται αυτόματα οποιαδήποτε στιγμή η εφαρμογή βρει ενεργή σύνδεση σε δίκτυο wifi.
Για να γίνει το upload απαραίτητη προϋπόθεση είναι να υπάρχει έγκυρη διεύθυνση ip, του εξυπηρετητή στον οποίο θα ανέβουν οι μετρήσεις. Με την πρώτη εγκατάσταση της εφαρμογής δεν υπάρχει κανένα αποθηκευμένο ip. Γι' αυτό το λόγο εμφανίζεται ένα μήνυμα τη στιγμή που η εφαρμογή θα χρειαστεί να κάνει upload που προτρέπει το χρήστη να δώσει μία έγκυρη ip διεύθυνση.

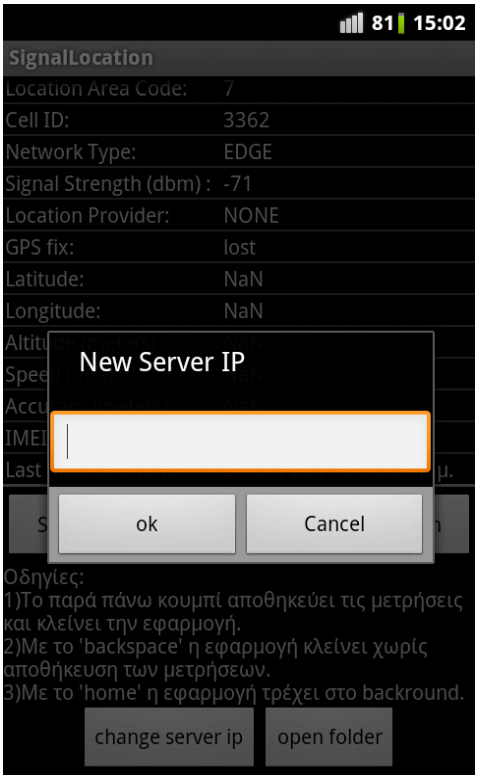
Η διεύθυνση IP μπορεί να αλλάξει μέσα από την εφαρμογή πατώντας το κουμπί "change server ip". Αυτό βρίσκεται στο κάτω μέρος της εφαρμογής και η μετάβαση σε αυτό γίνεται κυλώντας προς τα κάτω. Σε αυτό το σημείο εμφανίζονται και κάποιες

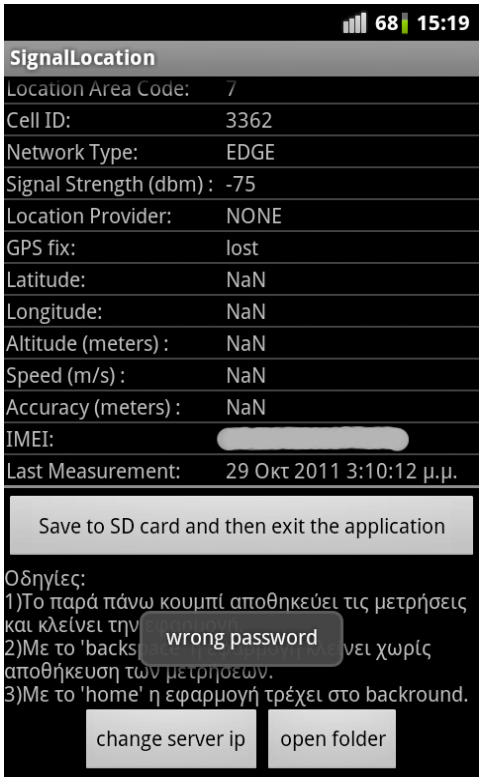
χρήσιμες βασικές οδηγίες καθώς και το κουμπί “open folder” το οποίο χρησιμεύει για άνοιγμα του φακέλου που περιέχει τις μετρήσεις.



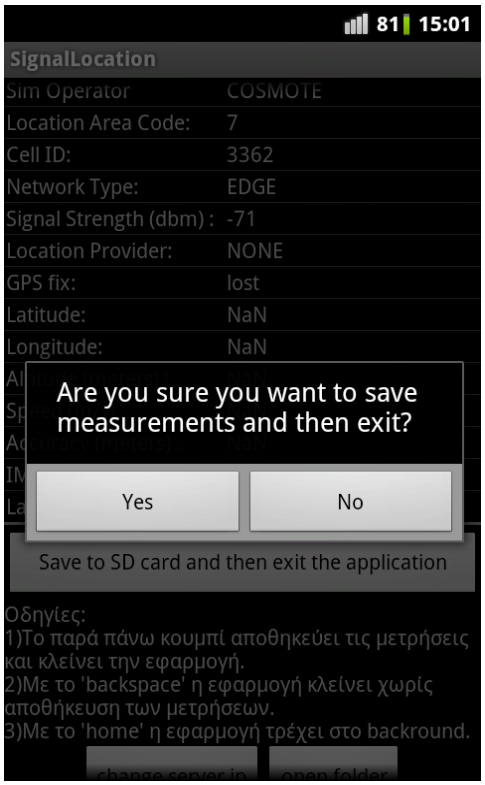


Για να εισάγει κάποιος νέο ip είναι αναγκαίο να εισάγει πρώτα τον κατάλληλο κωδικό που του ζητάει η εφαρμογή. Αν ο κωδικός αυτός εισαχθεί σωστά ένα νέο popup-dialog εμφανίζεται όπου το νέο ip μπορεί να εισαχθεί. Σε αντίθετη περίπτωση αν ο κωδικός είναι εσφαλμένος η εφαρμογή δείχνει το μήνυμα “wrong password”.





Για την αποτελεσματικότερη λειτουργία της εφαρμογής εμφανίζονται dialog-alerts που ενημερώνουν το χρήστη σε περίπτωση που πατηθεί το κουμπί back ή το κουμπί save στην εφαρμογή. Έτσι προλαμβάνονται ατυχήματα όπως να κλείσει η εφαρμογή και να χαθούν μετρήσεις σε περίπτωση που κάποιο από αυτά πατηθεί κατά λάθος.



Η εφαρμογή ενημερώνει με ειδοποιήσεις το χρήστη για διάφορα συμβάντα που συμβαίνουν στο παρασκήνιο όπως το upload των αρχείων με τις μετρήσεις ή

διάφορα σφάλματα όπως να μην μπορούν να αποθηκευτούν οι μετρήσεις στην κάρτα sd ή να βρεθεί έγκυρη ip διεύθυνση server.

4.3 Υλοποίηση

4.3.1 Βάση Δεδομένων

Στον εξυπηρετητή λειτουργεί μία MySQL βάση δεδομένων. Χρησιμοποιώντας το εργαλείο phpmyadmin είναι εφικτή η διαχείριση των διαφόρων βάσεων δεδομένων που υπάρχουν εκεί. Έτσι, δημιουργείται η βάση του συστήματος με όνομα mydb3. Η βάση αυτή θα περιέχει δύο πίνακες, τον temp που θα περιέχει προσωρινά τα δεδομένα των εκάστοτε νέων καταγραφών και τον markers που είναι ο κύριος πίνακας του συστήματος και περιέχει όλα τα δεδομένα που έχουν ανέβει από τα κινητά τερματικά. Η SQL εντολή που χρησιμοποιείται για την δημιουργία του temp πίνακα είναι:

```
CREATE TABLE `temp` (
  `id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,
  `so` INT (6),
  `lac` INT ,
  `cellID` INT ,
  `net` INT(2),
  `lp` INT(1),
  `lat` DOUBLE( 12, 8 ) NOT NULL ,
  `lng` DOUBLE( 12, 8 ) NOT NULL ,
  `alt` DOUBLE( 5,1 ),
  `speed` DOUBLE( 4,2 ),
  `acc` INT(3),
  `rss` INT(3) NOT NULL ,
  `date` DATETIME,
  `imei` BIGINT
) ENGINE = MYISAM
```

παρόμοια είναι και η εντολή για τον πίνακα markers.

4.3.2. Εξυπηρετητής

Τα αρχεία και οι φάκελοι που χρησιμοποιούνται είναι τα εξής:

Φάκελοι:

Images

Περιέχει τις εικόνες που χρησιμοποιούνται στις διάφορες ιστοσελίδες

NoUpload

Εδώ μεταφέρονται τα αρχεία που δεν μπορούν να φορτωθούν στη βάση δεδομένων, δηλαδή αρχεία τύπου διαφορετικό από txt ή xml και με μέγεθος μεγαλύτερο από 2MB.

OldFiles

Εδώ μεταφέρονται τα αρχεία δεδομένων αφού έχουν φορτωθεί στη βάση.

TempFiles

Εδώ αποθηκεύονται όλα τα αρχεία που έχουν ανέβει από τα κινητά τερματικά, πριν φορτωθούν στη βάση δεδομένων.

Αρχεία:

dbinfo.php

Σ' αυτό το αρχείο υπάρχουν μόνο οι εξής τέσσερις μεταβλητές: username, password, database, folder. Οι τρεις πρώτες χρησιμοποιούνται για την πρόσβαση στη βάση δεδομένων και η τελευταία περιέχει το όνομα του φακέλου στο οποίο βρίσκεται η εφαρμογή web. Οι μεταβλητές αυτές αποτελούν ουσιαστικά κάποιες σταθερές που χρησιμοποιούνται από τα περισσότερα υπόλοιπα αρχεία κώδικα.

uploadFile.php

Το php αρχείο στο οποίο στέλνει η κινητή συσκευή τα αρχεία δεδομένων. Είναι υπεύθυνο για την αποθήκευσή τους στον TempFiles φάκελο και την αποστολή κατάλληλου μηνύματος επιτυχίας/αποτυχίας στην κινητή συσκευή.
Που θα αποθηκευτούν τα ανεβασμένα αρχεία:

```
$target_path  = "TempFiles/";
```

Uploadedfile είναι το όνομα της φόρμας που δημιουργεί η εφαρμογή SignalLocation για να ανεβάσει τα αρχεία με τη μέθοδο POST

```
$target_path = $target_path . basename( $_FILES["uploadedfile"]["name"]);
if(move_uploaded_file($_FILES["uploadedfile"]["tmp_name"], $target_path)){
        echo "The file " . basename( $_FILES["uploadedfile"]["name"]). " has
been uploaded";}
else{
        echo "There was an error uploading the file, please try again.";}
```

index.html, index_greek.html

Περιέχει τη δομή της ιστοσελίδας, δηλαδή τη θέση των διάφορων αντικειμένων. Πρόκειται για την αρχική σελίδα που βλέπει ο client, δηλαδή αυτή με τον έναν χάρτη και την επιλογή παρόχου. Η λειτουργία του κάθε αντικειμένου προσδιορίζεται από το Javascript κώδικα που βρίσκεται στο scripts.js και γι' αυτό πρέπει να εισαχθεί στην html σελίδα. Επιπλέον, πρέπει να εισαχθεί Javascript κώδικας και από τη Google για τη φόρτωση του χάρτη. Τέλος, η μορφοποίηση της σελίδας, δηλαδή, χρώμα φόντου, αποστάσεις μεταξύ αντικειμένων κτλ, γίνεται από το stylesheet.css, το οποίο επίσης πρέπει να εισαχθεί. Έτσι στο κομμάτι head του html έχουμε τις εξής δηλώσεις:

```
<link rel="stylesheet" type="text/css" href="stylesheet.css" />
<script type="text/javascript"
src="http://maps.google.com/maps/api/js?sensor=false&libraries=geometry"><
/script>
<script type="text/javascript" src="scripts.js"></script>
```

Τα διάφορα στοιχεία ελέγχου, όπως για παράδειγμα τα checkboxes για την επιλογή απεικόνισης με γραμμές ή με σημαίες, βρίσκονται μέσα σε τμήματα (divisions, div) της κλάσης "box", για να είναι εφικτή η από κοινού επεξεργασία τους από το stylesheet.css.

```
<div class="box">
        Choose how to display data:<br/>
        <input type="checkbox" id="checkLines" checked="checked"/>Lines
        <br/><input type="checkbox" id="checkMarkers"/>Markers
```

```
</div>
```

Το index_greek.html, όπως είναι προφανές, περιέχει τον ίδιο ακριβώς κώδικα απλά τα στοιχεία που εμφανίζονται στον client είναι στα ελληνικά.

comparison.html, comparison_greek.html

Ομοίως με προηγουμένως, πρόκειται για τη σελίδα με τους τρεις χάρτες, έναν για κάθε πάροχο. Εδώ χρειάζεται το scripts2.js επιπλέον του Javascript κώδικα από τη Google και του stylesheet.css για τη μορφοποίηση της σελίδας.

genxml.php

Το αρχείο αυτό που τρέχει στον server καλείται από τα scripts.js και scripts2.js με τη μέθοδο GET και παραμέτρους δύο οριακά σημεία (τα latitude και longitude αυτών)

```
$lat1 = $_GET["lat1"];//south west
$lng1 = $_GET["lng1"];//south west
$lat2 = $_GET["lat2"];//north east
$lng2 = $_GET["lng2"];//north east
```

και εκτελεί την εξής διαδικασία: βρίσκει με ερώτηση από τη βάση δεδομένων τα σημεία που περικλείονται στο τετράγωνο που σχηματίζουν αυτά τα οριακά σημεία

```
$query = "SELECT * FROM markers WHERE lat>$lat1 AND lat<$lat2 AND
lng>$lng1 AND lng<$lng2";
$result = mysql_query($query);
```

και δημιουργεί ένα xml αρχείο με στοιχεία τα σημεία αυτά και attributes τα διάφορα πεδία που υπάρχουν στη βάση δεδομένων

```
$dom = new DOMDocument("1.0");
$node = $dom->createElement("markers");
$parnode = $dom->appendChild($node);
header("Content-type: text/xml");
while ($row = mysql_fetch_assoc($result)) {
        //add to xml document node
        $node = $dom->createElement("marker");
        $newnode = $parnode->appendChild($node);
        $newnode->setAttribute("so", $row['so']);
        $newnode->setAttribute("lac", $row['lac']);
        $newnode->setAttribute("cellID", $row['cellID']);
        $newnode->setAttribute("net", $row['net']);
        $newnode->setAttribute("lp", $row['lp']);
        $newnode->setAttribute("lat", $row['lat']);
        $newnode->setAttribute("lng", $row['lng']);
        $newnode->setAttribute("alt", $row['alt']);
        $newnode->setAttribute("speed", $row['speed']);
        $newnode->setAttribute("acc", $row['acc']);
        $newnode->setAttribute("rss", $row['rss']);
        $newnode->setAttribute("date", $row['date']);
        $newnode->setAttribute("imei", $row['imei']);
}
echo $dom->saveXML();
```

Το xml αρχείο αυτό επιστρέφεται στον client για περαιτέρω επεξεργασία.
Ας σημειωθεί ότι αν το genxml.php έχει κληθεί από το scripts.js η ερώτηση στη βάση διαφοροποιείται ώστε να συμπεριλάβει και τον επιλεγμένο πάροχο. Δηλαδή:

```
$mnc = $_GET["mnc"];//mobile network code
$query = "SELECT * FROM markers WHERE so=$mnc+20200 AND lat>$lat1
AND lat<$lat2 AND lng>$lng1 AND lng<$lng2";
```

scripts.js, scripts2.js

Περιέχουν τον client-side Javascript κώδικα που χρειάζεται για την απεικόνιση των δεδομένων στο χάρτη, καθώς και για τις λειτουργίες που επιτελούν τα διάφορα στοιχεία ελέγχου στην ιστοσελίδα. Πιο συγκεκριμένα, μόλις φορτωθεί το σώμα (body) του index.html καλείται η συνάρτηση load(). Αυτή δημιουργεί και τον χάρτη που θα φαίνεται στην ιστοσελίδα

map = new google.maps.Map(document.getElementById("map_canvas"), {

με διάφορες επιλογές, όπως το κέντρο του

center: new google.maps.LatLng(37.983987,23.728344),

το επίπεδο εστίασης

zoom: initial_zoom,

(το initial_zoom είναι καθολική σταθερά ορισμένη εκτός της συνάρτησης load()

var initial_zoom=12;)

τον τύπο του χάρτη, δηλαδή, χάρτης με εικόνες των δρόμων ή χάρτης με απεικόνιση από δορυφόρο

mapTypeId: 'roadmap',

και τέλος η μη ύπαρξη στοιχείων ελέγχου StreetView, αφού κάτι τέτοιο δεν υποστηρίζεται ακόμη στην Ελλάδα

streetViewControl: false

});

Επιπλέον, στη συνάρτηση load() γίνεται χρήση AJAX (Asynchronous JAvascript Xml) προγραμματισμού μέσω της συνάρτησης downloadUrl() στο scripts.js και του lastmodified.php στον server ώστε να εμφανιστεί στην ιστοσελίδα η ημερομηνία της τελευταίας τροποποίησης της βάσης δεδομένων. Αυτό αναλύεται και παρακάτω, αλλά η γενική ιδέα του AJAX προγραμματισμού είναι η επικοινωνία client και server μέσω διαδικτύου, με τη μορφή αίτησης από τον client μέσω Javascript και απόκρισης του server μέσω Xml.

Όταν ο χρήστης επιλέξει να εμφανίσει τα δεδομένα στο χάρτη ενεργοποιείται η συνάρτηση standard_process(). Εκεί αρχικά προσδιορίζεται ποιος πάροχος έχει επιλεγεί.

```
var cosmote = document.getElementById("cosmote").checked;
var vodafone = document.getElementById("vodafone").checked;
var wind = document.getElementById("wind").checked;
var provider_selected = 0;
if(cosmote) {
        provider_selected = 1;
}
else if(vodafone) {
        provider_selected = 5;
}
```

```
else if(wind) {
        provider_selected = 10;
};
```
Κατόπιν, βρίσκονται τα όρια της τρέχουσας απεικόνισης του χάρτη.
```
var lat1 = map.getBounds().getSouthWest().lat();
var lng1 = map.getBounds().getSouthWest().lng();
var lat2 = map.getBounds().getNorthEast().lat();
var lng2 = map.getBounds().getNorthEast().lng();
```

Για να φορτωθούν τα δεδομένα που αντιστοιχούν στην τρέχουσα απεικόνιση του χάρτη πρέπει να γίνει μία αίτηση στον server, ο οποίος με τη σειρά του θα απευθυνθεί στη βάση δεδομένων και θα δώσει πίσω την απάντηση στον client. Η αίτηση στον server γίνεται με τη βοήθεια της συνάρτησης downloadUrl().
```
function downloadUrl(url, callback) {
        var request = window.ActiveXObject ?
                new ActiveXObject('Microsoft.XMLHTTP') :
                new XMLHttpRequest;
        request.onreadystatechange = function() {
                if (request.readyState == 4 && request.status == 200) {
                        request.onreadystatechange = doNothing;
                        callback(request, request.status); }
        };
        request.open('GET', url, true);
        request.send(null);
}
```
Εδώ γίνεται χρήση του AJAX (Asynchronous JavaScript and XML) προγραμματισμού μέσω του αντικειμένου XMLHttpRequest. Στον server καλείται το url με τη μέθοδο GET και μόλις επιστραφεί η απάντηση εκτελείται η callback συνάρτηση στον client. Έτσι, είναι δυνατή η επικοινωνία server και client χωρίς να έχουμε φόρτωση ολόκληρης της ιστοσελίδας ξανά.
Οπότε έχουμε:
```
downloadUrl("genxml.php?lat1="+lat1+"&lng1="+lng1+"&lat2="+lat2+"&lng2="
+lng2+"&mnc="+provider_selected, function(data) {
      var xml = data.responseXML;
      var markers = xml.documentElement.getElementsByTagName("marker");
      for (var i = 0; i < markers.length-1; i++) {
```
και ακολουθεί η επεξεργασία για κάθε στοιχείο που έχει επιστρέψει το xml που δημιούργησε το genxml.php. Η επεξεργασία αυτή περιλαμβάνει για κάθε σημείο τη δημιουργία marker χρώματος ανάλογου του rss (λαμβανόμενη ισχύς)
```
if (rss>=good_rss) {
        var marker = new google.maps.Marker({
                map: null,
                position: point,
                icon:
'http://labs.google.com/ridefinder/images/mm_20_green.png'
        });
```
καθώς και τη δημιουργία γραμμών μεταξύ διαδοχικών σημείων
```
if (next_rss>=good_rss) {
```

```
        createOneLine(point,next_point,1);
}
```

Σύμφωνα με την ΕΕΤΤ (Εθνική Επιτροπή Τηλεπικοινωνιών και Ταχυδρομείων) ο χαρακτηρισμός του σήματος είναι:
Μέχρι και var good_rss=-85; καλή λήψη (πράσινο χρώμα marker)
Μέχρι και var mid_rss=-95; αποδεκτή λήψη (πορτοκαλί χρώμα marker)
Κάτω από mid_rss κακή λήψη (κόκκινο χρώμα marker)

Ο κώδικας της συνάρτησης createOneLine() είναι:

```
var dist =
google.maps.geometry.spherical.computeDistanceBetween(pointX,pointY);
if (dist<max_dist){
        var bounds = [pointX, pointY];
        var sC;
        switch (color){
        case 1:
                sC = "green";
                break;
        case 2:
                sC = "orange";
                break;
        case 3:
                sC = "red";
                break;
        }
        var simplePoly = new google.maps.Polyline({
                path: bounds,
                strokeColor: sC,
                strokeWeight: line_weight,
                map: null
        });
```

Όπου γίνεται χρήση της έτοιμης συνάρτησης computeDistanceBetween() από την Google, για τον υπολογισμό της απόστασης δύο σημείων πάνω σε επιφάνεια σφαίρας.

Αν δύο διαδοχικά σημεία ανήκουν σε διαφορετική κατηγορία ποιότητας λήψης, τότε χρησιμοποιείται ο μέσος όρος του rss. Όμως, επειδή το rss είναι εκφρασμένο σε dBm, για να βρεθεί ο μέσος όρος πρέπει πρώτα να μετατραπεί το rss των δύο διαδοχικών σημείων σε mW, να υπολογιστεί ο μέσος όρος και το αποτέλεσμα να μετατραπεί πάλι σε dBm.

```
var av =
10*Math.log((Math.pow(10,rss/10)+Math.pow(10,next_rss/10))/2)/Math.log(10);
if (av>=good_rss) {
        createOneLine(point,next_point,1);
}
```

Τέλος, για κάθε σημείο δημιουργείται και μια “φυσαλίδα” με πληροφορίες για το είδος δικτύου και τη χρονική στιγμή της μέτρησης, η οποία προσκολλάται στον κάθε marker και ενεργοποιείται όταν ο χρήστης κάνει κλικ πάνω σε marker.

```
var html = "INFO<br/><b>Received Signal Strength: "+rss+"</b><br/>Network Type: "+net_type+"<br/>date: "+date;
bindInfoWindow(marker, map, infoWindow, html);

function bindInfoWindow(marker, map, infoWindow, html) {
   google.maps.event.addListener(marker, 'click', function() {
                infoWindow.setContent(html);
                infoWindow.open(map, marker);
   });
}
```

Όπως μπορεί να παρατηρηθεί και οι markers και η γραμμές δεν τοποθετούνται απ’ ευθείας στον χάρτη (map: null). Αρχικά εισάγονται σε πίνακες:

```
markersArray.push(marker);
linesArray.push(simplePoly);
```

Και η εμφάνισή τους στον χάρτη γίνεται με τις συναρτήσεις showMarkers() και showLines() αντίστοιχα.

```
function showMarkers() {
   if (markersArray) {
                for (i in markersArray) {
                        markersArray[i].setMap(map);
                }
   }
}

function showLines() {
   if (linesArray) {
                for (i in linesArray) {
                        linesArray[i].setMap(map);
                }
   }
}
```

Στο scripts.js υπάρχουν επιπροσθέτως και συναρτήσεις που αντιστοιχούν στα διάφορα στοιχεία ελέγχου της ιστοσελίδας. Για παράδειγμα στο index.html υπάρχει:

```
Find a location:
        <input id="address" type="tex" onkeydown="find_location()">
        <input type="button" id="find_location" value="Go"
onclick="codeAddress()">
```

και στο scripts.js υπάρχουν:

```
function codeAddress() {
var address = document.getElementById("address").value;
geocoder.geocode( { 'address': address}, function(results, status) {
        if (status == google.maps.GeocoderStatus.OK) {
                map.setCenter(results[0].geometry.location);
```

```
                map.setZoom(geocoder_zoom);
        } else {
                alert("Geocode was not successful for the following reason: " +
status);
        }
});
}

function find_location() {
        if (event.keyCode == 13)
        {
                document.getElementById("find_location").click();
        }
}
```

Τα οποία λειτουργούν ως εξής:
Στο textbox ο χρήστης μπορεί να πληκτρολογήσει μία περιοχή, την οποία θέλει να δει στο χάρτη. Όταν ο χρήστης πατήσει enter (keyCode=13) στο textbox ή πατήσει το κουμπί Go ενεργοποιείται η συνάρτηση codeAddress() η οποία με τη βοήθεια του αντικειμένου geocoder που υπάρχει στον Javascript κώδικα της Google, βρίσκει την περιοχή αυτή και κεντράρει τον χάρτη εκεί.

Ας σημειωθεί ότι όλα τα προηγούμενα αναφέρονται στο scripts.js, αλλά τα ίδια υπάρχουν και στο scripts2.js που χρησιμοποιείται από το comparison.html, με τη διαφορά ότι δεν υπάρχουν οι επιλογές για τον πάροχο κινητής τηλεφωνίας, αφού και οι τρεις πάροχοι απεικονίζονται στο comparison.html. Μια ενδιαφέρουσα δυνατότητα στο scripts2.js που δεν υπάρχει στο scripts.js είναι η ταυτόχρονη μετακίνηση και των τριών χαρτών. Πιο συγκεκριμένα αν ο χρήστης μετακινήσει έναν από τους τρεις χάρτες που αντιστοιχούν στους παρόχους στο comparison.html, τότε και οι άλλοι δύο χάρτες ακολουθούν την κίνησή του. Αυτό γίνεται βάζοντας τρεις Listener έναν σε κάθε χάρτη στη συνάρτηση load() που ενεργοποιείται με τη φόρτωση της ιστοσελίδας

```
google.maps.event.addListener(m1, 'center_changed',
function(){m1_moved();});
google.maps.event.addListener(m2, 'center_changed',
function(){m2_moved();});
google.maps.event.addListener(m3, 'center_changed',
function(){m3_moved();});
```

Όταν αλλάξει το κέντρο για παράδειγμα του πρώτου χάρτη, τότε ενεργοποιείται η συνάρτηση m1_moved()

```
function m1_moved() {
        m2_moving=1;
        if (m1_moving==0){
                m2.setCenter(m1.getCenter());
                m2.setZoom(m1.getZoom());
        }
        m2_moving=0;
        m3_moving=1;
        if (m1_moving==0){
                m3.setCenter(m1.getCenter());
```

```
            m3.setZoom(m1.getZoom());
        }
        m3_moving=0;
}
```

Αρχικά θέτονται τα m2_moving, m3_moving ίσα με 1 ώστε να είναι γνωστό ότι οι χάρτες 2 και 3 κινούνται λόγω κίνησης του χάρτη 1. Έπειτα, αν το m1_moving είναι 0, δηλαδή ο χάρτης 1 κινείται από το χρήστη κι όχι από τις συναρτήσεις m2_moved ή m3_moved, το κέντρο και η εστίαση των χαρτών 2 και 3 γίνονται ίδια με το κέντρο και την εστίαση του χάρτη 1. Τέλος, οι μεταβλητές m2_moving και m3_moving θέτονται και πάλι ίσες με 0 που σημαίνει ότι η συνάρτηση m1_moved έχει σταματήσει τη διαδικασία μετακίνησης των χαρτών 2 και 3. Χωρίς τις μεταβλητές m1_moving, m2_moving και m3_moving μετακίνηση για παράδειγμα του χάρτη 1 θα σήμαινε αρχικά μετακίνηση των χαρτών 2 και 3, αλλά η αλλαγή θέσης αυτών θα προκαλούσε εκ νέου μετακίνηση του χάρτη 1 με αποτέλεσμα ο κώδικας να πέσει σε ατέρμων βρόχο και να κολλήσει.

stylesheet.css

Όπως έχει αναφερθεί αυτό το αρχείο εισάγεται σε όλα τα html αρχεία και περιέχει τον τρόπο εμφάνισής τους. Αυτό που αξίζει να προσεχθεί εδώ είναι ότι όλα τα στοιχεία που περιέχουν χάρτη πρέπει να έχουν σαφώς καθορισμένο το μέγεθός τους. Το ίδιο ισχύει και για τα στοιχεία που περιέχουν έστω και ένα στοιχείο το οποίο περιέχει χάρτη, κοκ. Η λογική αυτή φτάνει μέχρι το html στοιχείο, που είναι και το ανώτερο στην ιεραρχία.

Όπως αναφέρθηκε πριν στο index.html όλα τα στοιχεία ελέγχου βρίσκονται σε div της κλάσης “box”. Ο κώδικας για τη μορφοποίηση που αντιστοιχεί σε αυτά είναι:

```
.box {
        border-style:dotted;
        border-width:thin;
        margin-bottom:1%;
}
```

Που δείχνει ότι: όλα τα αντικείμενα της κλάσης “box” θα έχουν περίγραμμα με λεπτές διακεκομμένες γραμμές (dotted, thin) και περιθώριο προς τα κάτω ίσο με το 1% του μεγέθους του αντικειμένου που τα περιέχει.

lastmodified.php

Το αρχείο αυτό κάνει μία ερώτηση στη βάση δεδομένων για την ημερομηνία που τροποποιήθηκε τελευταία φορά ο πίνακας markers

```
$result = mysql_query("SELECT UPDATE_TIME FROM
information_schema.tables WHERE  TABLE_SCHEMA = '$database' AND
TABLE_NAME = 'markers'");
$row = mysql_fetch_assoc($result);
```

Στη MySQL, η πληροφορία αυτή βρίσκεται στη βάση information_schema στον πίνακα tables. Στη συνέχεια, δημιουργείται ένα xml αρχείο που περιέχει την πληροφορία αυτή

```
$dom = new DOMDocument("1.0");
$node = $dom->createElement("parent");
```

```
$parnode = $dom->appendChild($node);
header("Content-type: text/xml");
$node = $dom->createElement("node");
$newnode = $parnode->appendChild($node);
$newnode->setAttribute("update_time", $row['UPDATE_TIME']);
echo $dom->saveXML();
```

Το lastmodified.php και πιο συγκεκριμένα το xml αρχείο που παράγει χρησιμοποιείται από τον Javascript κώδικα μέσω του XMLHttpRequest αντικειμένου για να εμφανίζει στις διάφορες ιστοσελίδες την ημερομηνία της τελευταίας τροποποίησης του κύριου πίνακα της βάσης δεδομένων, του πίνακα markers δηλαδή.

maintenance.php

Το αρχείο αυτό είναι ουσιαστικά ένα html αρχείο με διάφορα στοιχεία ελέγχου για τη διαχείριση της βάσης δεδομένων και των αρχείων που ανεβαίνουν από τα κινητά τερματικά. Προορίζεται για χρήση μόνο από το διαχειριστή του συστήματος, γι' αυτό και απαιτείται η εισαγωγή username και password για να μπορέσει κάποιος να δει την ιστοσελίδα. Για ευκολία, το username και το password είναι τα ίδια με αυτά που χρειάζονται για την πρόσβαση στη βάση δεδομένων, γι' αυτό και εισάγεται το αρχείο dbinfo.php, που περιέχει αυτές τις πληροφορίες.

```
<body>
<?php
require("dbinfo.php");
if ($_POST['txtUsername'] != $username || $_POST['txtPassword'] !=
$password) {
?>
        <div id="subheader">
        <h1>Login</h1>
        <form name="form" method="post" action="<?php echo
$_SERVER['SCRIPT_NAME']; ?>">
                Username:<br/>
                <input type="text" title="Enter your Username"
name="txtUsername" /><br/>
                Password:<br/>
                <input type="password" title="Enter your password"
name="txtPassword" /><br/>
                <input type="submit" name="Submit" value="Login" />
        </form>
        </div>
<?php
}
else {
?>
```

κι ακολουθεί ο html κώδικας για την ιστοσελίδα. Ο κώδικας που χρειάζεται για να λειτουργήσουν τα διάφορα στοιχεία ελέγχου βρίσκεται στο script_functions.js, οπότε εισάγεται στο κομμάτι head του html.

```
<script type="text/javascript" src="script_functions.js"></script>
```

Αξίζει να σημειωθεί εδώ ότι κάθε στοιχείο ελέγχου ακολουθείται από ένα τμήμα (div) το οποίο είναι αρχικά κενό:

```
<div>
        List of files uploaded by the application:
        <input type=button onclick="check_temp()" value="Show"/>
</div>
<div id="check_temp"></div>
```

Το div αυτό θα χρησιμοποιηθεί αργότερα από το script_functions.js για να εμφανίσει την απάντηση του server.

script_functions.js

Οι Javascript συναρτήσεις που βρίσκονται εδώ έχουν βασικά όλες την ίδια λειτουργία: κάνουν αίτηση στον server και αλλάζουν ένα στοιχείο του maintenance.php σύμφωνα με την απόκριση του server (AJAX programming). Το στοιχείο του server που καλούν είναι το functions.php το οποίο περιέχει τις αντίστοιχες συναρτήσεις. Στα scripts.js και scripts2.js η συνάρτηση που επικοινωνεί με τον εξυπηρετητή είναι η downloadUrl() που αναλύθηκε προηγουμένως. Στο script_functions.js χρησιμοποιείται η loadXMLDoc() η οποία ακολουθεί την ίδια λογική δημιουργίας του αντικειμένου XMLHttpRequest.

```
function loadXMLDoc(url,callback) {
        if (window.XMLHttpRequest) {
```

κώδικας για Firefox, Chrome, Opera, Safari και εκδόσεις Internet Explorer από 7 και πάνω

```
                xmlhttp=new XMLHttpRequest();
        }
        else {
```

κώδικας για Internet Explorer έκδοση 5 και 6

```
                xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
        }
        xmlhttp.onreadystatechange=callback;
        xmlhttp.open("GET",url,true);
        xmlhttp.send();
}
```

Ένα παράδειγμα συνάρτησης στο script_functions.js είναι:

```
function check_temp(){
```

αρχικά γράφει στο maintenance.php στο div με id “check_temp” τη φράση “Processing... Please wait”

```
        document.getElementById("check_temp").innerHTML="Processing...
Please wait";
```

κατόπιν, καλεί τη συνάρτηση check_temp() από το functions.php

```
        loadXMLDoc("functions.php?func=check_temp", function(){
```

αν ο server αποκριθεί θετικά

```
                if (xmlhttp.readyState==4 && xmlhttp.status==200) {
```

τότε γράφει το αποτέλεσμα που θα έχει επιστραφεί στη θέση του div με id “check_temp”, δηλαδή στη θέση του “Processing... Please wait”

```
                document.getElementById("check_temp").innerHTML=
xmlhttp.responseText;
```

```
            }
        });
}
```

functions.php

Το functions.php καλείται με τη μέθοδο GET από το script_functions.js. Η μεταβλητή $func χρησιμοποιείται για να ενεργοποιηθεί η σωστή συνάρτηση από το functions.php ανάλογα με τη συνάρτηση του script_functions.js που το κάλεσε.

```
$func = $_GET["func"];
switch ($func) {
case "quick_update":
        quick_update();
        break;
case "erase_points":
        erase_points();
        break;
case "check_temp":
        check("TempFiles");
        break;
case "load":
        load();
        break;
case "update":
        update();
        break;
case "check_noupload":
        check("NoUpload");
        break;
case "delete_noupload":
        delete_files("NoUpload");
        break;
case "check_oldfiles":
        check("OldFiles");
        break;
case "delete_oldfiles":
        delete_files("OldFiles");
        break;
default:
        echo "There was a problem executing this command.";
}
```

Οι συναρτήσεις του functions.php και η λειτουργία τους είναι οι εξής:

load()

Εδώ γίνεται φόρτωση των δεδομένων από τα αρχεία που έχουν ανέβει από τα κινητά τερματικά στη βάση δεδομένων. Πιο συγκεκριμένα ανοίγει πρώτα ο φάκελος στον οποίο έχουν αποθηκευτεί τα αρχεία δεδομένων

```
$dirname = $folder . "TempFiles";
```

$dir = opendir($dirname);
Για κάθε αρχείο στον φάκελο TempFiles
while(($file = readdir($dir)) != false)
ανακαλύπτεται το είδος του αρχείου κοιτώντας την επέκτασή του
$ext = substr($file, strrpos($file,'.')+1);
εάν η επέκταση δεν είναι txt ή xml και το μέγεθος του αρχείου είναι μεγαλύτερο από 2MB, τότε το αρχείο μετακινείται στον φάκελο NoUpload, αλλιώς τα xml αρχεία φορτώνονται στη βάση, ενώ τα txt χρειάζονται πρώτα μετατροπή σε xml. Το xml περιέχει τον parent κόμβο με όνομα markers, στοιχεία (elements) με όνομα marker και παιδιά των elements τις διάφορες πληροφορίες για το κάθε σημείο. Τα πεδία των πληροφοριών πρέπει να είναι ακριβώς 13 και αποτελούν τα πεδία που αργότερα θα εισαχθούν στη βάση δεδομένων.

```
if ($ext=="txt") {
        $fp = fopen("TempFiles/$file", 'r');
        $xml = new SimpleXMLElement('<markers></markers>');
        while ($line = fgetcsv($fp)) {
        if (count($line) == 13) {
                $node = $xml->addChild('marker');
                $node->addChild('so', $line[0]);
                $node->addChild('lac', $line[1]);
                $node->addChild('cellID', $line[2]);
                $node->addChild('net', $line[3]);
                $node->addChild('lp', $line[4]);
                $node->addChild('lat', $line[5]);
                $node->addChild('lng', $line[6]);
                $node->addChild('alt', $line[7]);
                $node->addChild('speed', $line[8]);
                $node->addChild('acc', $line[9]);
                $node->addChild('rss', $line[10]);
                $node->addChild('date', $line[11]);
                $node->addChild('imei', $line[12]);
        }
}
$newfile = basename($file, ".txt").".xml";
$xml->asXML("TempFiles/$newfile");
```

Για να φορτωθεί ένα xml αρχείο στη βάση καλείται η συνάρτηση myloadXML() με παράμετρο το αρχείο xml. Ας σημειωθεί ότι η myloadXML() αποθηκεύει τα νέα δεδομένα στον πίνακα temp της βάσης κι όχι στον πίνακα markers, που είναι και ο κύριος πίνακας του συστήματος.

Μία επιπλέον λειτουργία της συνάρτησης load() είναι η διασφάλιση ότι τα σημεία που υπάρχουν στον πίνακα temp της βάσης θα απέχουν μεταξύ τους τουλάχιστον 10 μέτρα (σταθερά RAY).
define("RAY", "0.0001"); //0.0001 in lat, lng is about 10m
Αυτό γίνεται ως εξής: για κάθε στοιχείο του πίνακα temp
$result = mysql_query("SELECT * FROM temp");
while($row = mysql_fetch_assoc($result))

αναζητούνται σημεία του temp που να βρίσκονται το πολύ 10 μέτρα μακριά του

```
$lat1 = $row['lat']+RAY;
$lat2 = $row['lat']-RAY;
$lng1 = $row['lng']+RAY;
$lng2 = $row['lng']-RAY;
$result2 = mysql_query("SELECT * FROM temp WHERE so={$row['so']} AND lat<$lat1 AND lat>$lat2 AND lng<$lng1 AND lng>$lng2 AND id!={$row['id']}");
```

στη συνέχεια υπολογίζεται ο μέσος όρος του rss του αρχικού σημείου μαζί με τα σημεία που βρίσκονται κοντά του (“γειτονικά σημεία”).
Ο μέσος όρος φυσικά θα πρέπει να βρεθεί μετατρέποντας τα dBm του rss σε mW

```
$sum = pow(10,$row['rss']/10);
$i = 1;
while($row2 = mysql_fetch_assoc($result2)) {
        $sum = $sum + pow(10,$row2['rss']/10);
        $keys[$k_i] = $row2['id'];
        $i++;
        $k_i++;
}
$newrss = 10*log10($sum/$i);
```

τέλος γίνεται ενημέρωση του αρχικού σημείου με το νέο rss και διαγραφή των υπόλοιπων γειτονικών σημείων

```
$result3 = mysql_query("UPDATE temp SET rss=$newrss WHERE id={$row['id']}");
for ($j=$k_i-$i+1;$j<$k_i;$j++) {
        $result4 = mysql_query("DELETE FROM temp WHERE id=$keys[$j]");
}
```

myloadXML()

Συνάρτηση για φόρτωση xml αρχείων δεδομένων στον πίνακα temp της βάσης. Καλείται από τη συνάρτηση load() με παράμετρο το αρχείο xml. Η εισαγωγή νέων εγγραφών στη βάση γίνεται μόνο αν τα πεδία lat (latitude) και lng (longitude) είναι διάφορα του NaN, το πεδίο rss είναι διάφορο του μηδενός και το πεδίο acc έχει τιμή μικρότερη από 100, δηλαδή μόνο όταν έχει γίνει καταγραφή των συγκεκριμένων πεδίων από την εφαρμογή του κινητού τερματικού με ακρίβεια 100 μέτρων.

```
if ($x[6]!="NaN" && $x[7]!="NaN" && $x[11]!=0 && $x[10]<100) {
        $result = mysql_query("INSERT INTO temp (so, lac, cellID, net, lp, lat, lng, alt, speed, acc, rss, date, imei) VALUES ($x[1], $x[2], $x[3], $x[4], $x[5], $x[6], $x[7], $x[8], $x[9], $x[10], $x[11], '$x[12]', $x[13])");
}
```

update()

Η συνάρτηση update() είναι υπεύθυνη για τη μεταφορά των δεδομένων από τον πίνακα temp στον πίνακα markers. Πριν γίνει εισαγωγή των στοιχείων του temp ελέγχεται αν υπάρχουν στοιχεία του markers σε απόσταση 10 μέτρα από αυτό (σταθερά RAY).

```
define("RAY", "0.0001"); //0.0001 in lat, lng is about 10m
```

Αν υπάρχουν τότε βρίσκεται σταθμισμένος μέσος όρος του rss ανάλογα με την παλαιότητα των δεδομένων. Αναλυτικότερα, ο διαχειριστής μέσω του maintenance.php αποφασίζει το χρονικό κατώφλι για το οποίο ένα σημείο θα θεωρείται παλιό. Αυτό περνάει στη μεταβλητή $period της συνάρτησης update() μέσω της μεθόδου GET

```
$period = $_GET["period"];
```

Για κάθε σημείο του πίνακα temp

```
$result = mysql_query("SELECT * FROM temp");
while($row = mysql_fetch_assoc($result)) {
```

αναζητούνται σημεία του markers που να βρίσκονται το πολύ 10 μέτρα μακριά του

```
$lat1 = $row['lat']+RAY;
$lat2 = $row['lat']-RAY;
$lng1 = $row['lng']+RAY;
$lng2 = $row['lng']-RAY;
$result2 = mysql_query("SELECT * FROM markers WHERE so={$row['so']}
AND id<$LastID AND lat<$lat1 AND lat>$lat2 AND lng<$lng1 AND lng>$lng2");
```

η συνθήκη id<$LastID χρησιμοποιείται ώστε η αναζήτηση στα σημεία του markers να μην συνεχίζει και στα νέα σημεία που μόλις έχουν εισαχθεί από τον temp, αφού για αυτά είναι γνωστό ότι απέχουν μεταξύ τους τουλάχιστον 10 μέτρα, χάρη στη συνάρτηση load(). Γι' αυτό το $lastID υπολογίζεται πριν αρχίσει η επεξεργασία του πίνακα markers.

```
$result2 = mysql_query("SELECT max(id) AS LastID FROM markers");
$x = mysql_fetch_assoc($result2);
if($x['LastID']!=NULL) {
        $LastID = $x['LastID'];
} else {
        $LastID = 0;
}
```

Εάν δεν βρεθούν γειτονικά σημεία, τότε το σημείο του temp εισάγεται αυτούσιο στον πίνακα markers

```
if (mysql_num_rows($result2)==NULL) {
        $result3 = mysql_query("INSERT INTO markers (so, lac, cellID, net, lp,
        lat, lng, alt, speed, acc, rss, date, imei) VALUES ({$row['so']},
        {$row['lac']}, {$row['cellID']}, {$row['net']}, {$row['lp']}, {$row['lat']},
        {$row['lng']}, {$row['alt']}, {$row['speed']}, {$row['acc']}, {$row['rss']},
        '{$row['date']}', {$row['imei']})");
}
```

Διαφορετικά πρέπει να υπολογιστεί ο σταθμισμένος μέσος όρος του rss, πάλι μετατρέποντας τα dBm σε mW:

```
$sum = pow(10,$row['rss']/10);
$i = 1;
$sum_factor = 1;
```

Επίσης, επειδή τελικά θα διατηρηθεί το πιο πρόσφατο σημείο, θα πρέπει να βρεθεί και η πιο πρόσφατη ημερομηνία. Αρχικά θεωρείται ότι πιο πρόσφατο είναι το σημείο του πίνακα temp

```
$max_date = $row['date'];
```

$temp_flag = 1;

Έτσι, έχουμε για κάθε γειτονικό σημείο:

while($row2 = mysql_fetch_assoc($result2)) {

αν νεώτερο είναι το σημείο του πίνακα temp, το γειτονικό σημείο του πίνακα markers θα έχει βάρος 0,5

if(date("Y-m-d H:i:s",strtotime($period,strtotime($row['date']))>$row2['date']) {
$factor = 0.5;
}

αν νεώτερο είναι το σημείο του πίνακα markers, τότε θα έχει βάρος 1,5

elseif(date("Y-m-d H:i:s",strtotime($period,strtotime($row2['date']))>$row['date'])
{
$factor = 1.5;
}

αν τα σημεία του temp και του markers δεν απέχουν χρονικά περισσότερο από $period, τότε το βάρος του γειτονικού στοιχείου του markers θα είναι 1

else {
$factor = 1;
}

υπολογίζεται το άθροισμα των rss και των βαρών, προσέχοντας να μετατραπούν τα dBm σε mW

$sum = $sum + pow(10,$row2['rss']/10)*$factor;
$sum_factor = $sum_factor + $factor;

ελέγχεται αν το γειτονικό σημείο που βρίσκεται στον πίνακα markers είναι πιο πρόσφατο

if($row2['date']>$max_date) {
$max_date = $row2['date'];
$temp_flag = 0;
}

στον πίνακα $keys αποθηκεύονται τα id των γειτονικών σημείων, ώστε να είναι δυνατόν αργότερα να επεξεργαστούν

$keys[$i]=$row2['id'];
$i++;
}

υπολογίζεται ο σταθμισμένος μέσος όρος, μετατρέποντας τα mW σε dBm

$newrss = 10*log10($sum/$sum_factor);

στη συνέχεια υπάρχουν δύο περιπτώσεις:

1. το πιο πρόσφατο σημείο βρίσκεται στον πίνακα markers, οπότε όλα τα γειτονικά σημεία του πίνακα markers πρέπει να αποκτήσουν το νέο rss και το σημείο στον πίνακα temp δεν θα εισαχθεί

if ($temp_flag==0) {
for($j=1;$j<$i;$j++) {
$result3=mysql_query("UPDATE markers SET rss=$newrss
WHERE id=$keys[$j]");
}
}

2. το πιο πρόσφατο σημείο είναι αυτό στον πίνακα temp, οπότε εισάγεται στον πίνακα markers αλλά με το νέο rss, ενώ όλα τα γειτονικά σημεία που βρίσκονται στον πίνακα markers διαγράφονται

```
else {
        $result3 = mysql_query("INSERT INTO markers (so, lac, cellID, net, lp,
        lat, lng, alt, speed, acc, rss, date, imei) VALUES ({$row['so']},
        {$row['lac']}, {$row['cellID']}, {$row['net']}, {$row['lp']}, {$row['lat']},
        {$row['lng']}, {$row['alt']}, {$row['speed']}, {$row['acc']}, $newrss,
        '{$row['date']}', {$row['imei']})");
        for ($j=1;$j<$i;$j++) {
                $result4 = mysql_query("DELETE FROM markers WHERE
id=$keys[$j]");
        }
}
```

Τελική λειτουργία της συνάρτησης update() είναι το άδειασμα του πίνακα temp, αφού οτιδήποτε χρειαζόταν έχει μεταφερθεί στον πίνακα markers.

```
$result = mysql_query("TRUNCATE TABLE temp");
```

Έτσι, ο πίνακας temp είναι έτοιμος να υποδεχτεί νέα δεδομένα από νέα xml αρχεία δεδομένων.

check()

Με τη συνάρτηση αυτή ελέγχεται αν ένας φάκελος περιέχει αρχεία και τα εμφανίζει αν υπάρχουν, καθώς και την ημερομηνία της τελευταίας τροποποίησής τους. Ο φάκελος περνάει ως παράμετρος $f με την κλήση της συνάρτησης.

Το dbinfo.php περιέχει τη διαδρομή (path) μέχρι τον φάκελο που θα ελεγχθεί, όπως έχει αναφερθεί και προηγουμένως.

```
require("dbinfo.php");
$dirname = $folder . $f;
$dir = opendir($dirname);
$flag = true;
while(($file = readdir($dir)) != false) {
        if($file!="." && $file!="..") {
                echo "The file: ".$file." was last modified on :".date("Y-m-d
H:i:s",filemtime("$f/$file")).".<br/>";
                $flag = false;
        }
}
```

Εάν ο φάκελος δεν περιέχει αρχεία τότε η μεταβλητή $flag είναι ακόμη αληθής (true) και ο η συνάρτηση check() ενημερώνει αναλόγως:

```
if ($flag) {
        echo "No files in the ".$f." folder.<br/>";
}
closedir($dir);
```

delete_files()

Με τη συνάρτηση αυτή διαγράφονται τα αρχεία ενός φακέλου ανάλογα αν είναι παλιότερα από ένα κατώφλι που προσδιορίζεται από τον διαχειριστή. Το κατώφλι αυτό περνάει στη συνάρτηση μέσω της μεταβλητής $threshold:

```
$threshold = $_GET["threshold"];
```

Το dbinfo.php περιέχει τη διαδρομή (path) μέχρι τον φάκελο που θα ελεγχθεί:

```
require("dbinfo.php");
$dirname = $folder . $f;
$dir = opendir($dirname);
```

Εάν το κατώφλι είναι διάφορο από all, δηλαδή διαγραφή όλων

```
if ($threshold!="all") {
```

τότε η μεταβλητή $threshold μετατρέπεται στη μορφή ημερομηνίας που αποθηκεύονται και οι υπόλοιπες ημερομηνίες στο σύστημα:

```
        $threshold = "-".$threshold." months";
        $min_date = date("Y-m-d H:i:s",strtotime($threshold));
        echo "Files older than: ".$min_date." will be deleted.<br/>";
```

Στη συνέχεια εξετάζονται όλα τα αρχεία που περιέχονται στον επιλεγμένο φάκελο:

```
        while(($file = readdir($dir)) != false) {
                if($file!="." && $file!="..") {
```

και πάλι γίνεται μετατροπή της ημερομηνίας τελευταίας τροποποίησης του έκαστου αρχείου στη συνηθισμένη μορφή του συστήματος:

```
                $file_date = date("Y-m-d H:i:s",filemtime("$f/$file"))."<br/>";
```

Αν το αρχείο είναι παλιότερο από το χρονικό κατώφλι, τότε διαγράφεται:

```
                        if($file_date<=$min_date) {
                                unlink( $folder."$f/$file" );
                                echo "The file: ".$file." was deleted.<br/>";
                        }
                }
        }
}
```

Εάν το κατώφλι είναι all, τότε διαγράφονται όλα τα αρχεία που υπάρχουν στον συγκεκριμένο φάκελο, ανεξάρτητα από την ημερομηνία τελευταίας τροποποίησής τους:

```
else {
        while(($file = readdir($dir)) != false) {
                if($file!="." && $file!="..") {
                        unlink( $folder."$f/$file" );
                        echo "The file: ".$file." was deleted.<br/>";
                }
        }
}
closedir($dir);
```

erase_points()

Με τη συνάρτηση αυτή διαγράφονται σημεία του πίνακα markers ανάλογα αν είναι παλιότερα από ένα κατώφλι που προσδιορίζεται από τον διαχειριστή. Το κατώφλι αυτό περνάει από το maintenance.php στο script_functions.js και μετά με τη μέθοδο GET στο functions.php και στη συνάρτηση erase_points() μέσω της μεταβλητής

$range και μετατρέπεται σε μορφή όμοια με αυτή που αποθηκεύονται οι ημερομηνίες στη βάση:

$range = "-".$range." years";

$min_date = date("Y-m-d H:i:s",strtotime($range));

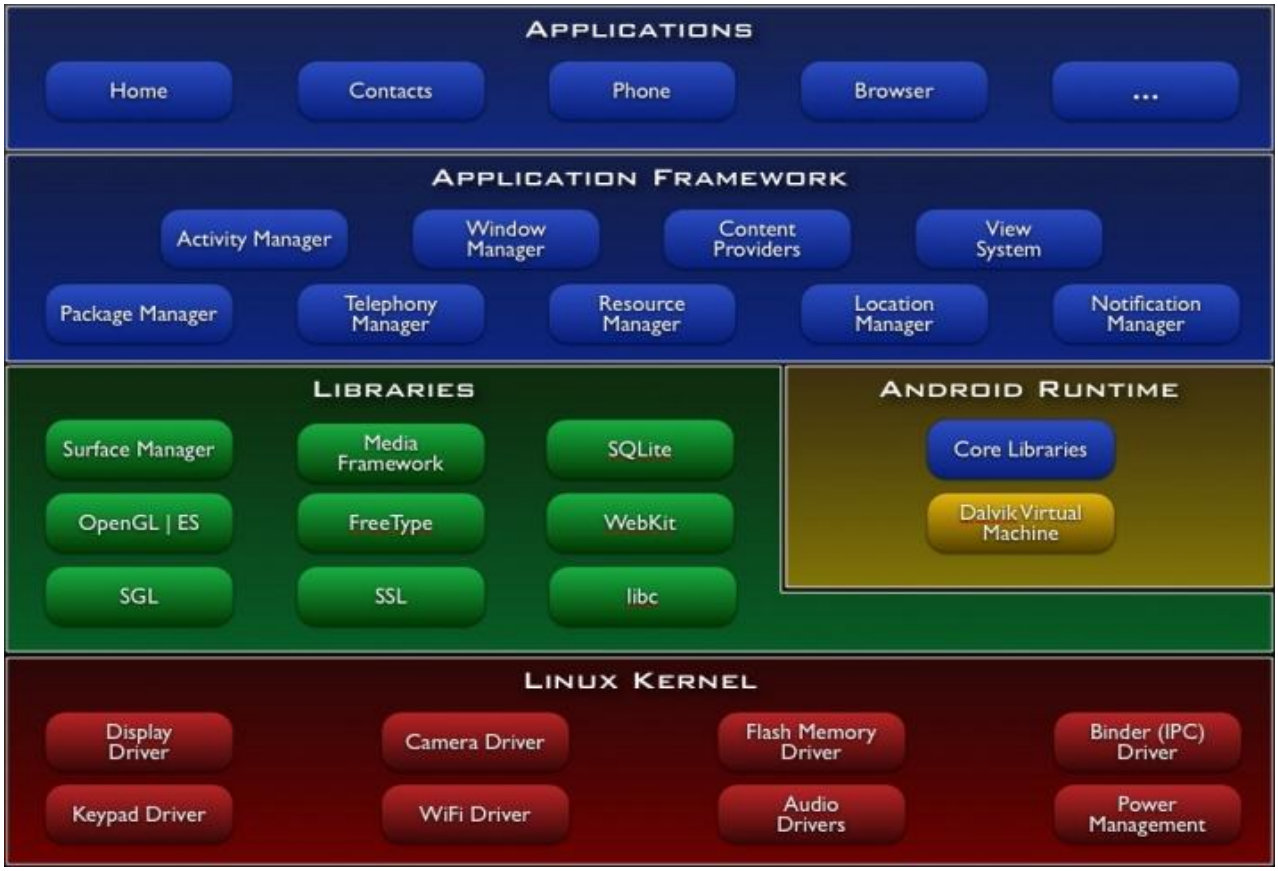
Κατόπιν, γίνεται κλήση στη βάση για τη διαγραφή των παλιών δεδομένων

$result = mysql_query("DELETE FROM markers WHERE date<'$min_date'");

4.3.3 Εισαγωγή στο Android

Το Android είναι ένα πακέτο λογισμικού για κινητές συσκευές που περιλαμβάνει ένα λειτουργικό σύστημα, ενδιάμεσο λογισμικό και βασικές εφαρμογές. Το Android SDK παρέχει τα εργαλεία και APIs (Application Programming Interface) για την ανάπτυξη εφαρμογών για την πλατφόρμα Android χρησιμοποιώντας τη γλώσσα προγραμματισμού Java.

Το παρακάτω διάγραμμα δείχνει τα βασικά συστατικά του λειτουργικού συστήματος Android. Κάθε ενότητα περιγράφεται αναλυτικότερα στη συνέχεια.



Application

Το Android περιλαμβάνει ένα σύνολο βασικών εφαρμογών, συμπεριλαμβανομένων e-mail client, το πρόγραμμα SMS, ημερολόγιο, χάρτες, πρόγραμμα περιήγησης, επαφές, και άλλα. Όλες οι εφαρμογές είναι γραμμένες χρησιμοποιώντας τη γλώσσα προγραμματισμού Java.

Application-framework

Παρέχοντας μια ανοιχτή πλατφόρμα ανάπτυξης, το Android προσφέρει στους προγραμματιστές τη δυνατότητα να χτίσουν εξαιρετικά πλούσιες και καινοτόμες εφαρμογές. Οι προγραμματιστές έχουν τη δυνατότητα να επωφεληθούν από το hardware της συσκευής, να έχουν πρόσβαση σε πληροφορίες για τον εντοπισμό της θέσης, να εκτελούν υπηρεσίες στο παρασκήνιο, να προσθέτουν ειδοποιήσεις στη γραμμή κατάστασης και πολλά άλλα.
Οι προγραμματιστές έχουν πλήρη πρόσβαση στο ίδιο πλαίσιο APIs που χρησιμοποιείται από τις βασικές εφαρμογές. Η αρχιτεκτονική των εφαρμογών έχει σχεδιαστεί για να απλοποιεί την επαναχρησιμοποίηση των κατασκευαστικών στοιχείων. Οποιαδήποτε εφαρμογή μπορεί να δημοσιεύσει τις δυνατότητές της και οποιαδήποτε άλλη εφαρμογή μπορεί να κάνει τότε χρήση αυτών των δυνατοτήτων. Αυτός ο ίδιος μηχανισμός επιτρέπει σε συστατικά να αντικατασταθούν από το χρήστη.
Πίσω από όλες τις εφαρμογές υπάρχει ένα σύνολο υπηρεσιών και συστημάτων, συμπεριλαμβανομένων:

- Ένα πλούσιο και επεκτάσιμο σύνολο **Views** που μπορούν να χρησιμοποιηθούν για τη δημιουργία μιας εφαρμογής και συμπεριλαμβάνει τους πίνακες, τα δίκτυα, πλαίσια κειμένου, κουμπιά, ακόμη και μια δυνατότητα ενσωμάτωσης σε πρόγραμμα περιήγησης στο Web
- Πάροχοι περιεχομένων (**Content Providers**) που επιτρέπουν στις εφαρμογές να έχουν πρόσβαση στα δεδομένα από άλλες εφαρμογές (όπως Επαφές), ή να μοιραστούν τα δικά τους δεδομένα
- Ένας Διαχειριστής Πόρων (**Resource Manager**) που διασφαλίζει την παροχή πρόσβασης σε πόρους χωρίς κωδικό, όπως σε τοπικά strings, γραφικά και σε αρχεία υπεύθυνα για τη διάταξη του ui
- Ένας Διαχειριστής ειδοποιήσεων (**Notification Manager**) που δίνει τη δυνατότητα σε όλες οι εφαρμογές να εμφανίζουν προσαρμοσμένες ειδοποιήσεις στη γραμμή κατάστασης
- Ένας Διαχειριστής Δραστηριοτήτων **(Activity Manager**) που διαχειρίζεται τον κύκλο ζωής των εφαρμογών και παρέχει μια κοινή πλοήγηση σε μορφή λίστας

Libraries
Το Android περιλαμβάνει ένα σύνολο από C / C + + βιβλιοθήκες που χρησιμοποιούνται από διάφορες συνιστώσες του συστήματος Android. Αυτές οι δυνατότητες εκτίθενται σε προγραμματιστές μέσω του application framework.

Android Runtime
Το Android περιλαμβάνει μια σειρά από βασικές βιβλιοθήκες που παρέχουν τις περισσότερες από τις λειτουργίες που διατίθεται σε βασικές βιβλιοθήκες της γλώσσας προγραμματισμού Java.

Κάθε εφαρμογή Android τρέχει στη δική της διαδικασία, με δική της δημιουργία εικονικής μηχανής Dalvik. Η Dalvik έχει γραφτεί έτσι ώστε μια συσκευή να μπορεί να εκτελέσει πολλές εικονικές μηχανές αποτελεσματικά. Η Dalvik Virtual Machine εκτελεί αρχεία σε Dalvik (. DEX) μορφή εκτελέσιμων αρχείων, πράγμα το οποίο έχει βελτιστοποιηθεί για ελάχιστη χρησιμοποίηση μνήμης. Η VM είναι βασισμένη σε καταχωρητές και τρέχει classes που έχουν συνταχθεί από έναν compiler για γλώσσα Java και έχουν μετατραπεί σε μορφή DEX από το συμπεριλαμβανόμενο "DX" εργαλείο. Τέλος η Dalvik VM βασίζεται στον πυρήνα του Linux για λειτουργικότητα, όπως το threading και τη διαχείριση χαμηλού επιπέδου μνήμης.

Linux Kernel

Το Android στηρίζεται στην έκδοση Linux 2.6 για υπηρεσίες στον πυρήνα, όπως η ασφάλεια, η διαχείριση της μνήμης, η διαχείριση των διαδικασιών και το μοντέλο οδηγών. Ο πυρήνας λειτουργεί επίσης ως ένα συνδετικό επίπεδο μεταξύ του υλικού και του υπόλοιπου της στοίβας λογισμικού.

Βασικές αρχές Εφαρμογών

Οι Android εφαρμογές είναι γραμμένες στη γλώσσα προγραμματισμού Java. Τα εργαλεία του Android SDK μεταφράζουν τον κώδικα, μαζί με τα δεδομένα και τα αρχεία των πόρων, σε ένα *πακέτο Android* (ένα συμπιεσμένο αρχείο με .apk κατάληξη). Όλος ο κώδικας σε ένα ενιαίο .apk αρχείο θεωρείται ότι είναι μία εφαρμογή και είναι το αρχείο που οι συσκευές με λειτουργικό Android χρησιμοποιούν για να εγκαταστήσουν την εφαρμογή.

Μόλις εγκατασταθεί σε μια συσκευή, κάθε εφαρμογή Android ζει στη δική της ασφάλεια του sandbox:

- Το λειτουργικό σύστημα Android είναι ένα multi-user σύστημα Linux στο οποίο κάθε εφαρμογή είναι διαφορετικός χρήστη.
- Από προεπιλογή, το σύστημα εκχωρεί σε κάθε εφαρμογή ένα μοναδικό αναγνωριστικό (ID) χρήστη Linux (το αναγνωριστικό χρησιμοποιείται μόνο από το σύστημα και είναι άγνωστο για την εφαρμογή). Το σύστημα ορίζει δικαιώματα για όλα τα αρχεία σε μια εφαρμογή, έτσι ώστε μόνο ο χρήστης (εφαρμογή) που διατεθεί τον εν λόγω κωδικό να έχει πρόσβαση σε αυτά.
- Κάθε διαδικασία έχει τη δική της εικονική μηχανή (VM), έτσι κώδικας μιας εφαρμογής τρέχει σε απομόνωση από άλλες εφαρμογές.
- Από προεπιλογή, κάθε εφαρμογή τρέχει σε Linux δική της διαδικασία. Το Android ξεκινά την διαδικασία όταν οποιοδήποτε από τα στοιχεία της εφαρμογής πρέπει να εκτελεστεί και μετά τερματίζει τη διαδικασία όταν δεν χρειάζεται πλέον ή όταν το σύστημα πρέπει να ανακτήσει τη μνήμη για άλλες εφαρμογές.

Με τον τρόπο αυτό, το σύστημα Android εφαρμόζει την αρχή least privilege. Δηλαδή, κάθε εφαρμογή, από προεπιλογή, έχει πρόσβαση μόνο στα στοιχεία που απαιτεί για να κάνει τη δουλειά της και όχι περισσότερα. Αυτό δημιουργεί ένα πολύ ασφαλές περιβάλλον στο οποίο οι εφαρμογές δεν μπορούν να έχουν πρόσβαση σε μέρη του συστήματος για τα οποία δεν έχει δοθεί η άδεια.

Ωστόσο, υπάρχουν τρόποι για μια εφαρμογή να μοιράζεται δεδομένα με άλλες εφαρμογές και να έχει πρόσβαση στις υπηρεσίες του συστήματος, όπως δύο εφαρμογές να μοιράζονται το ίδιο ID και έτσι να έχουν πρόσβαση η μία στα αρχεία της άλλης ή να ζητήσει άδεια η εφαρμογή για να έχει πρόσβαση στα δεδομένα της συσκευής μέσω του AndroidManifest.xml .

Δομικά Συστατικά Εφαρμογής

Τα δομικά συστατικά μιας εφαρμογής είναι τα στοιχεία από τα οποία αποτελείται μια εφαρμογή Android. Κάθε στοιχείο είναι ένα διαφορετικό σημείο μέσω του οποίου το σύστημα μπορεί να εισέλθει στην εφαρμογή. Δεν είναι όλα τα στοιχεία σημεία εισόδου για το χρήστη και κάποια εξαρτώνται από άλλα στοιχεία, αλλά το καθένα υπάρχει σαν δική του οντότητα και παίζει ένα συγκεκριμένο ρόλο που βοηθά να καθορίσει τη συνολική συμπεριφορά μιας εφαρμογής.

Υπάρχουν τέσσερις διαφορετικοί τύποι στοιχείων της εφαρμογής. Κάθε τύπος εξυπηρετεί ένα συγκεκριμένο σκοπό και έχει ένα ξεχωριστό κύκλο ζωής που ορίζει τον τρόπο με τον οποίο το στοιχείο δημιουργείται και καταστρέφεται.

Δραστηριότητες (Activities)

Μια *δραστηριότητα* αντιπροσωπεύει μία μόνο οθόνη με μια διεπαφή χρήστη. Για παράδειγμα, μία εφαρμογή ηλεκτρονικού ταχυδρομείου μπορεί να έχει μία δραστηριότητα που δείχνει μια λίστα με τα νέα μηνύματα ηλεκτρονικού ταχυδρομείου, μια άλλη δραστηριότητα για τη σύνθεση ενός μηνύματος ηλεκτρονικού ταχυδρομείου, καθώς και άλλη δραστηριότητα για την ανάγνωση μηνυμάτων ηλεκτρονικού ταχυδρομείου. Αν και οι δραστηριότητες συνεργάζονται για να σχηματίσουν μια συνεκτική εμπειρία για το χρήστη από την εφαρμογή e-mail, κάθε μία είναι ανεξάρτητη από τις άλλες. Ως εκ τούτου, μια διαφορετική εφαρμογή μπορεί να ξεκινήσει οποιαδήποτε από αυτές τις δραστηριότητες (εάν η εφαρμογή e-mail το επιτρέπει). Για παράδειγμα, μια εφαρμογή της κάμερας μπορεί να αρχίσει τη δραστηριότητα στην εφαρμογή e-mail που συνθέτει νέο mail, έτσι ώστε ο χρήστης να μοιραστεί μια εικόνα.

Υπηρεσίες (Services)

Μια *υπηρεσία* είναι ένα στοιχείο που εκτελείται στο παρασκήνιο για να εκτελέσει μακροχρόνιες λειτουργίες ή να εκτελέσει εργασίες για απομακρυσμένες διαδικασίες. Μια υπηρεσία δεν παρέχει περιβάλλον εργασίας. Για παράδειγμα, μια υπηρεσία θα μπορούσε να παίξει μουσική στο παρασκήνιο, ενώ ο χρήστης βρίσκεται σε διαφορετική εφαρμογή, ή μπορεί να κατεβάσει δεδομένα μέσω του δικτύου χωρίς να εμποδίζει την αλληλεπίδραση του χρήστη με μια δραστηριότητα. Ένα άλλο στοιχείο, όπως μια δραστηριότητα, μπορεί να ξεκινήσει μια υπηρεσία για να την αφήσει να τρέξει ή να συνδεθεί με αυτή, ούτως ώστε να αλληλεπιδράσει με αυτή.

Οι πάροχοι περιεχομένου (Content providers)

Ένας Content provider διαχειρίζεται ένα σύνολο δεδομένων εφαρμογής. Τα δεδομένα μπορούν να αποθηκευτούν στο σύστημα αρχείων, σε μια βάση δεδομένων SQLite, στο διαδίκτυο, ή σε οποιαδήποτε άλλη θέση αποθήκευσης η εφαρμογή μπορεί να έχει πρόσβαση. Μέσα από τον Content provider, άλλες εφαρμογές μπορούν να ζητήσουν ή ακόμα και να τροποποιήσουν τα δεδομένα (αν ο Content provider επιτρέπει). Για παράδειγμα, το σύστημα Android παρέχει μια υπηρεσία παροχής περιεχομένου που διαχειρίζεται τα στοιχεία των επαφών. Ως εκ τούτου, κάθε εφαρμογή με τα κατάλληλα δικαιώματα μπορεί να υποβάλει ερώτημα σε κάποιο τμήμα του Content provider (όπως ContactsContract.Data) για να διαβάσει και να γράψει πληροφορίες σχετικά με μια συγκεκριμένη επαφή.

Οι Content providers είναι επίσης χρήσιμοι για την ανάγνωση και εγγραφή δεδομένων που είναι ιδιωτικά σε μία εφαρμογή και δεν μοιράζονται. Για παράδειγμα, η εφαρμογή του σημειωματάριου χρησιμοποιεί μια υπηρεσία παροχής περιεχομένου για να αποθηκεύσετε τις σημειώσεις.

Broadcast receivers

Ένας Broadcast receiver είναι ένα στοιχείο που ανταποκρίνεται σε broadcast announcements του συστήματος. Για παράδειγμα, μια broadcast announcement από το σύστημα ανακοινώνει ότι η οθόνη είναι απενεργοποιημένη, η μπαταρία είναι

χαμηλή, ή μια εικόνα συλλήφθηκε. Οι εφαρμογές μπορούν επίσης να εκκινήσουν broadcast announcements. Για παράδειγμα, να αφήσουν άλλες εφαρμογές να γνωρίζουν ότι κάποια δεδομένα έχουν κατέβει στη συσκευή και είναι διαθέσιμα για να τα χρησιμοποιήσουν. Αν και οι Broadcast receivers δεν εμφανίζουν περιβάλλον εργασίας, μπορούν να δημιουργήσουν μια ειδοποίηση στη γραμμή κατάστασης η οποία προειδοποιεί το χρήστη όταν ένα broadcast event συμβαίνει. Πιο συχνά, όμως, ένας broadcast receiver είναι απλά μια "πύλη" προς άλλα στοιχεία και έχει ως στόχο να κάνει μια πολύ ελάχιστη ποσότητα εργασίας. Για παράδειγμα, θα μπορούσε να ξεκινήσει μια υπηρεσία για να εκτελέσει κάποια εργασία που βασίζεται σε ένα event.

4.3.4 Τερματικό

Στην εφαρμογή για τη συλλογή και καταγραφή των δεδομένων έχουν χρησιμοποιηθεί τα παραπάνω δομικά στοιχεία. Συγκεκριμένα, η εφαρμογή εκτελείται σε μία service και μία activity οι οποίες επικοινωνούν με broadcast receiver για τη συχνή ανανέωση του γραφικού περιβάλλοντος της activity αλλά είναι και συνδεδεμένες με bind έτσι ώστε να είναι δυνατή η μεταφορά δεδομένων. Επιπλέον η service χρησιμοποιεί αρκετούς broadcast receivers για να αντιλαμβάνεται τις braodcast announcements του συστήματος, όπως πότε το wifi συνδέεται στο διαδίκτυο.

Η εφαρμογή αποτελείται ουσιαστικά από 4 αρχεία, καθώς και τις αντίστοιχες βιβλιοθήκες και πηγές λογισμικού για την πλατφόρμα android.

- Το SignalLocationActivity.java περιέχει τις διεργασίες που εκτελούνται ενώ η εφαρμογή βρίσκεται στο προσκήνιο όπως η ανανέωση του ui (user interface).
- Το SignalLocationService.java περιέχει τις διεργασίες που εκτελούνται όσο η εφαρμογή βρίσκεται στο παρασκήνιο.
- Το main.xml είναι υπεύθυνο για τη σχεδίαση του ui.
- Το AndroidManifest.xml αναφέρει χρήσιμες πληροφορίες για την εφαρμογή οι οποίες είναι απαραίτητες πριν αυτή εκτελεστεί όπως είναι η αναφορά και η περιγραφή των δομικών στοιχείων της εφαρμογής (αρχεία java) ή η δήλωση των αδειών που πρέπει να αποκτήσει από το λειτουργικό η εφαρμογή για να έχει πρόσβαση σε διάφορες λειτουργίες του τερματικού.

Συγκεκριμένα χρησιμοποιούνται συνολικά:
4 broadcast receivers
5 event listeners

Ο πρώτος broadcast receiver βρίσκεται στην SignalLocationActivity και αντιλαμβάνεται πότε η SignalLocationService ανανεώνει τις μεταβλητές που χρειάζονται για το ui. Αυτό συμβαίνει κάθε 1 δευτερόλεπτο και συνεπώς κάθε δευτερόλεπτο ο broadcast receiver καλεί τη συνάρτηση που ανανεώνει το ui με τις καινούριες τιμές (intent από την SignalLocationService).

```
private BroadcastReceiver broadcastReceiver = new BroadcastReceiver() {
@Override
public void onReceive(Context context, Intent intent) {
updateUI(intent);
}
```

```
};
```

Αυτός ο broadcast receiver ενεργοποιείται κάθε φορά που καλείται η activity, δηλαδή η εφαρμογή μπαίνει στο προσκήνιο, και απενεργοποιείται όποτε η εφαρμογή μπαίνει στο παρασκήνιο.

```
public void onResume() {
super.onResume();
startService(intent);
registerReceiver(broadcastReceiver, new
IntentFilter(SignalLocationService.BROADCAST_ACTION));
}
public void onPause() {
super.onPause();
unregisterReceiver(broadcastReceiver);
stopService(intent);
}
```

Όλοι οι υπόλοιποι broadcast receivers βρίσκονται στη SignalLocationService. Συγκεκριμένα ο **wificonnection** καλείται όποτε υπάρχει κάποια αλλαγή (connected-disconnected) στη σύνδεση στο internet μέσω wifi. Αν δημιουργηθεί σύνδεση, ανανεώνει την κλήση του **onLocationChangedNETWORK** listener ώστε να είναι άμεση η ενημέρωση της θέσης μέσω του wifi location provider. Επίσης καλεί τη συνάρτηση upload η οποία ξεκινάει τις διαδικασίες για uploading των μετρήσεων στον server. Αντίθετα αν διακοπεί η σύνδεση, ελέγχει αν υπάρχει σύνδεση του GPS με το δορυφόρο και αν δεν υπάρχει μηδενίζει τις τιμές των μεταβλητών εφόσον δεν υπάρχει τρόπος να βρεθεί η θέση του κινητού τερματικού.

```
BroadcastReceiver wificonnection = new BroadcastReceiver(){
@Override
public void onReceive(Context context, Intent intent) {
// TODO Auto-generated method stub
if (cm.getNetworkInfo(1).getState() == NetworkInfo.State.CONNECTED){
// Toast.makeText(context, "internet connection has been established",
50).show();
if (isGPSFix == false){
lmgr.removeUpdates(onLocationChangeNETWORK);
lmgr.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0 , 5f ,
onLocationChangeNETWORK);
}
new Thread(new Runnable() {
public void run() {
upload(contextd);
}
}).start();
}
else if (cm.getNetworkInfo(1).getState() ==
NetworkInfo.State.DISCONNECTED){
```

```
// Toast.makeText(context, "internet connection has been lost", 50).show();
if (isGPSFix==false){
zeromeasurements();  }
}
}};
```

Η συνάρτηση **upload** και όλες οι συναρτήσεις οι οποίες αυτή καλεί εκτελούνται σε καινούριο thread έτσι ώστε να μην επιβαρύνουν το thread του ui. Κάθε εφαρμογή android τρέχει σε ένα μοναδικό thread (αυτό του ui) εκτός και αν οριστεί κάτι διαφορετικό από τον προγραμματιστή. Αυτό όμως έχει το αρνητικό ότι εάν κάποια διεργασία είναι χρονοβόρα κολλάει και η επιφάνεια εργασίας. Έτσι για διεργασίες που έχουν μεγάλη διάρκεια ή που η διάρκεια τους δεν μπορεί να προβλεφτεί εξαιτίας αστάθμητων παραγόντων (όπως οι εργασίες στο διαδίκτυο) χρησιμοποιείται νέο thread ώστε να μην επηρεάζεται το ui.

Ο **wifistate** broadcast receiver ομοίως αντιλαμβάνεται πότε ανοίγει ή κλείνει ο δέκτης του wifi (**WIFI_STATE_DISABLED, WIFI_STATE_ENAVLED, WIFI_STATE_DISABLING, WIFI_STATE_ENABLING**) και ανάλογα εκτελεί τις κατάλληλες λειτουργίες όπως το να βγάζει ειδοποιήσεις.

Οι 2 αυτοί broadcast receivers ενεργοποιούνται κατά τη δημιουργία της service στην void onCreate με τις εντολές:

```
IntentFilter filter1 = new IntentFilter();
IntentFilter filter2 = new IntentFilter();
filter1.addAction(ConnectivityManager.CONNECTIVITY_ACTION);
filter2.addAction(WifiManager.WIFI_STATE_CHANGED_ACTION);
registerReceiver(wificonnection,filter1);
registerReceiver(wifistate,filter2);
```

και απενεργοποιούνται κατά τον τερματισμό ολόκληρης της εφαρμογής και συνεπώς και της service με τις εντολές:

```
unregisterReceiver(wificonnection);
unregisterReceiver(wifistate);
```

Ο broadcastReceiver ScreenReceiver αναλαμβάνει να ακούει πότε σβήνει η οθόνη και αν συμβεί αυτό την ξανανοίγει σε 200 millisecond.

```
BroadcastReceiver ScreenReceiver = new BroadcastReceiver(){
@Override
public void onReceive(Context context, Intent intent) {
if (intent.getAction().equals(Intent.ACTION_SCREEN_OFF)) {
final Handler handler = new Handler();
Timer timer = new Timer();
TimerTask task = new TimerTask() {
public void run() {
handler.post(new Runnable() {
```

```
public void run() {
PowerManager pm2 = (PowerManager)
contextd.getSystemService(Context.POWER_SERVICE);
PowerManager.WakeLock wl2 ;
wl2 = pm2.newWakeLock(PowerManager.SCREEN_DIM_WAKE_LOCK |
PowerManager.ACQUIRE_CAUSES_WAKEUP |
PowerManager.ON_AFTER_RELEASE, "open screen");
wl2.acquire(100);
}
});
}
};
timer.schedule(task, 200);
}
}};
```

Αυτός ο broadcastReceiver είναι πολύ σημαντικός για την σωστή λειτουργία της εφαρμογής μιας και τα android κινητά έχουν πολύ αυστηρή πολιτική στο θέμα εξοικονόμησης ενέργειας. Κάθε φορά που κλείνει η οθόνη, το κινητό πέφτει σε κατάσταση εξοικονόμησης ενέργειας (sleep mode) κατά την οποία ο επεξεργαστής λειτουργεί με την μικρότερη δυνατή ισχύ καθιστώντας κάποιες διεργασίες αδύνατες ή πολύ αργές. Επίσης με το κλείσιμο της οθόνης το android κινητό σταματάει τα broadcast announcements του Signal Strength με αποτέλεσμα να σταματάνε και οι καταγραφές του σήματος από την εφαρμογή. Είναι αναγκαίο λοιπόν η οθόνη του τερματικού να μην σβήνει ποτέ. Σε αυτό βοηθάνε και οι παρακάτω συναρτήσεις με τις οποίες η οθόνη δεν σβήνει ποτέ και ο επεξεργαστής δεν πέφτει ποτέ σε sleep mode:

```
//Keep cpu running all the time
  pm = (PowerManager) getSystemService(Context.POWER_SERVICE);
   wlcpu = pm.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK , "My
Tag");
   wlcpu.acquire();
//Keep screen open all the time
   wlscreen = pm.newWakeLock(PowerManager.SCREEN_DIM_WAKE_LOCK ,
"My Tag");
   wlscreen.acquire();
```

Ο ScreenReceiver λοιπόν αναλαμβάνει να ξανανοίξει την οθόνη εάν αυτή κλείσει με κάποιο κουμπί και η συνάρτηση wlscreen.acquire() να την κρατήσει ανοιχτή. Αυτό δεν θα μπορούσε να συμβεί αποδοτικά αν δεν υπήρχε η wlcpu.acquire αφού με το κλείσιμο της οθόνης ο επεξεργαστής θα σταμάταγε και θα ήταν αμφίβολη η ενεργοποίηση της οθόνης ξανά.

Υπάρχουν επίσης και 5 listeners οι οποίοι κάνουν παρόμοια δουλειά με τους broadcast receivers, αλλά περιέχουν και κάποιες συγκεκριμένες void ορισμένες από το σύστημα android για επιπλέον λειτουργίες (όπως είναι η void onProviderEnebled για τον LocationListener).

Ο πρώτος listener είναι ο LocationListener **onLocationChangeGPS** ο οποίος ενημερώνεται από το GPS location provider όταν υπάρχει κάποια αλλαγή στη γεωγραφική θέση του τερματικού και τότε ενημερώνει με τις καινούριες συντεταγμένες τις μεταβλητές και τις αποθηκεύει με την void saveSignalStrength. Επίσης οι 2 void που περιέχει (onProviderEnabled και onProviderDisabled) καλούνται όταν ανοίγει ή κλείνει το gps.

```
LocationListener onLocationChangeGPS = new LocationListener()
{
public void onLocationChanged(Location location1)
{
lat = location1.getLatitude();
lng = location1.getLongitude();
alt = location1.getAltitude();
sp = location1.getSpeed();
acc = location1.getAccuracy();
Calendar now = Calendar.getInstance();
int month = now.get(Calendar.MONTH)+1 ;
dt = now.get(Calendar.YEAR) + "_" + month + "_" +
now.get(Calendar.DAY_OF_MONTH) + "_" +
now.get(Calendar.HOUR_OF_DAY) + "_" +
now.get(Calendar.MINUTE) + "_" + now.get(Calendar.SECOND) ;
ntpro = 1;
time3 = new Date().toLocaleString();
saveSignalStrength();
}

public void onProviderDisabled(String provider1)
{ if (gpsstart){
Context context = getApplicationContext();
Toast.makeText(context,"GPS is off", 50).show();
}
gpsstart = true;
}
public void onProviderEnabled(String provider1)
{
Context context = getApplicationContext();
Toast.makeText(context,"GPS is on", 50).show();

}
public void onStatusChanged(String provider, int status, Bundle extras)
{}
};
```

Ο επόμενος listener είναι ο LocationListener **onLocationChangeNETWORK** και κάνει ακριβώς την ίδια δουλειά με τον onLocationChangeGPS με τη διαφορά ότι

ενημερώνεται από το Network location provider για την αλλαγή στη θέση του τερματικού.

Επίσης στην εφαρμογή υπάρχει και ένας PhonestateListener, ο **onSignalStrengthChange,** ο οποίος ενημερώνεται όταν αλλάζει η ένταση του σήματος του δικτύου κινητών επικοινωνιών στο τερματικό. Όταν συμβαίνει αυτό ενημερώνονται με τις καινούριες τιμές οι μεταβλητές που αφορούν το δίκτυο κινητών επικοινωνιών και τις αποθηκεύει με την void saveSignalStrength εάν ήδη έχει γίνει αποδεκτή μέτρηση συντεταγμένων γεωγραφικής θέσης. Επίσης σε αυτό το σημείο γίνεται και η μετατροπή σε dBm.

```
PhoneStateListener onSignalStrengthChange = new PhoneStateListener()
 {
 public void onSignalStrengthsChanged (SignalStrength sigstrength)
 {
 lac = gsm_cl.getLac();
 cid = gsm_cl.getCid();
 nt = tm.getNetworkType();
 Calendar now = Calendar.getInstance();
 int month = now.get(Calendar.MONTH)+1 ;
 dt = now.get(Calendar.YEAR) + "_" + month + "_" +
 now.get(Calendar.DAY_OF_MONTH) + "_" +
now.get(Calendar.HOUR_OF_DAY) + "_" +
 now.get(Calendar.MINUTE) + "_" + now.get(Calendar.SECOND) ;
 if (sigstrength.isGsm())
 {
 ss = sigstrength.getGsmSignalStrength();
 if (ss==99)
 { ss = 0; }
 else
 { ss = (2*ss)-113; }
 }
 //time3 = new Date().toLocaleString();
 if (!lat.isNaN()){
 saveSignalStrength();
 }
 }
 };
```

Τέλος έχουμε και τον listener ο οποίος ακούει τις αλλαγές στο status του gps. Αυτός είναι ο onGpsStatusChange. Στην ουσία αυτός ο listener καλείται όποτε αλλάζει η κατάσταση σύνδεσης του gps με τον δορυφόρο. Αν και δεν μπορεί να αντιληφθεί πότε έχει κλειδώσει το gps με κάποιο δορυφόρο για να αρχίσει να λαμβάνει στίγμα της θέσης, αντιλαμβάνεται τις αλλαγές στο σήμα που δέχεται από τους δορυφόρους. Έτσι σε κάθε αλλαγή ελέγχει αν η τελευταία μέτρηση που έχει ληφθεί από το onLocationChange2GPS είναι νεότερη των 10 δευτερολέπτων.

```
else if (event == GpsStatus.GPS_EVENT_SATELLITE_STATUS) {
```

if (mLastLocation != null) {
isGPSFix = (SystemClock.elapsedRealtime() - mLastLocationMillis) < 10000; }

Αν συμβαίνει αυτό θεωρεί ότι το gps έχει κλειδώσει και συνεπώς μπορεί να πάρει στίγμα της θέσης. Με κάθε αλλαγή της κατάστασης του gps ενώ υπάρχει σήμα αυξάνεται το **gpsfixchange** κατά 1. Αυτό λειτουργεί ως flag για το πότε βρίσκεται στίγμα πρώτη φορά μετά από πολλή ώρα.

if (isGPSFix) { // A fix has been acquired.
gpsfixchange++;
if (gpsfixchange==1){
Toast.makeText(context, "Gps connected with satellite", 50).show(); } }

Αν ο έλεγχος **isGPSFix = (SystemClock.elapsedRealtime() - mLastLocationMillis) < 10000** έχει αποτέλεσμα false μηδενίζονται οι μεταβλητές (ώστε να μην λαμβάνονται μετρήσεις με παλιότερη θέση) αν δεν υπάρχει σύνδεση στο Internet μέσω wifi και καλείται ξανά ο onLocationChangeNetwork για άμεση ενημέρωση της θέσης μέσω δικτύου αν υπάρχει πρόσφατη μέτρηση.

else { // The fix has been lost.
if (gpsfixchange != 0) {
Toast.makeText(context, "Gps FIX LOST after: " + gpsfixchange + "attempts", 50).show();
if (cm.getNetworkInfo(ConnectivityManager.TYPE_WIFI).isConnected()==false){ zeromeasurements(); }
lmgr.removeUpdates(onLocationChangeNETWORK);
lmgr.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0 , 5f , onLocationChangeNETWORK); }
gpsfixchange = 0; }

Για τη σωστή λειτουργία του onGpsStatusChange έχει δημιουργηθεί και ένας επιπλέον listener, ο onLocationChange2Gps ο οποίος είναι ο ίδιος ποιοτικά listener με τον onLocationChangeGps με τη διαφορά ότι αυτός καλείται ανά 0 μέτρα ενώ ο onLocationChangeGps ανά 5 μέτρα για την ανανέωση της θέσης. Ο onLocationChange2Gps λοιπόν έχει δημιουργηθεί για να ανανεώνει συνεχώς (και όχι ανά 5 μέτρα) τις μεταβλητές:
mLastLocationMillis = SystemClock.elapsedRealtime();
mLastLocation = location ;
οι οποίες χρησιμοποιούνται από το onGpsStatusChange για να ανιχνευτεί αν υπάρχει σύνδεση με δορυφόρο.

Οι listeners ενεργοποιούνται κατά την δημιουργία της service στην void onCreate με τις εντολές:
lmgr.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0 , 5f , onLocationChangeGPS);
lmgr.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0 , 0f , onLocationChange2GPS);
lmgr.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0 , 5f ,

onLocationChangeNETWORK);
lmgr.addGpsStatusListener(onGpsStatusChange);
tm.listen(onSignalStrengthChange,
PhoneStateListener.LISTEN_SIGNAL_STRENGTHS);
και απενεργοποιούνται κατά τον τερματισμό ολόκληρης της εφαρμογής και συνεπώς και της service με τις εντολές:
lmgr.removeUpdates(onLocationChangeGPS);
lmgr.removeUpdates(onLocationChange2GPS);
lmgr.removeUpdates(onLocationChangeNETWORK);
lmgr.removeGpsStatusListener(onGpsStatusChange);

Η service αναλαμβάνει να κάνει το upload των αρχείων με τις μετρήσεις στον server. Αυτό πραγματοποιείται μέσω 3 συναρτήσεων. Η πρώτη είναι η **upload** η οποία αναλαμβάνει να βρει το αποθηκευμένο σε ένα text αρχείο ip του server.

```
String ip = "";
StringBuilder text = new StringBuilder();
File file = new File("/data/data/com.SignalLocation/myserversip/ip.txt");
try {
BufferedReader br = new BufferedReader(new FileReader(file));
String line;
String line2 = "";
while ((line = br.readLine()) != null) {
text.append(line);
text.append('\n');
line2 +=line ;
}
ip = line2;
uploadMeasurements(context , ip);
```

Αφού το βρει καλεί την **uploadMeasurements** η οποία αναλαμβάνει να βρει ένα-ένα τα αρχεία τα οποία περιέχουν τις μετρήσεις.

```
File dir = new File("/sdcard/mymeasurements/");
dir.mkdirs();
if (dir.isDirectory()) {
String[] children = dir.list();
for (int i = 0; i < children.length; i++) {
String filepath = dir + "/" + children[i] ;
uploadSinglefile (filepath , children[i], context, ip);
}
```

Για κάθε ένα από αυτά καλεί την **uploadSinglefile** η οποία κάνει τελικά το upload με POST στον server για κάθε αρχείο περιμένοντας την απάντηση του server.
Αν δεν προκύψει κάποιο σφάλμα και η απάντηση από τον server είναι θετική σβήνει το εν λόγω αρχείο από τον φάκελο όπου είναι αποθηκευμένο.

```
String responsename = "The file " + filename + " has been uploaded" ;
```

```
if (response.equals(responsename)){
File myfile = new File(filepath);
myfile.delete();
filecounter++;
}
```

Όπως ήδη αναφέρθηκε το ip του server αποθηκεύεται σε ένα αρχείο. Αυτό συμβαίνει για να μπορεί να αλλαχτεί μέσα από το ui της εφαρμογής. Ωστόσο, δεν μπορεί ο καθένας να το αλλάξει, αφού είναι προστατευμένο με password. Πατώντας το κουμπί **change server ip** καλείται η συνάρτηση **onChangePressed** η οποία δημιουργεί ένα alert-dialog το οποίο ζητά να εισαχθεί ο κατάλληλος κωδικός.

```
final EditText input = new EditText(this);
new AlertDialog.Builder(SignalLocationActivity.this)
.setMessage("Password Protection")
.setView(input)
.setPositiveButton("ok", new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog, int whichButton) {
String value = input.getText().toString();
```

Αν εισαχθεί σωστά καλείται η **onChangeIpPressed** συνάρτηση, αλλιώς βγαίνει το μήνυμα **"wrong password"**.

```
if (value.equals(password)){
onChangeIpPressed();
}
else {
Toast.makeText(contextd, "wrong password", 1000).show();
}
```

Εφόσον κληθεί η **onChangeIpPressed** με τη σειρά της δημιουργεί ένα alert-dialog για αποθήκευση του νέου ip. Μόλις ο χρήστης πληκτρολογήσει το νέο ip και πατήσει “ok” καλείται η συνάρτηση **savefile** η οποία χρησιμοποιείται για αποθήκευση των μετρήσεων αλλά και του αρχείου του ip.

```
String value = input.getText().toString();
try {
out = new OutputStreamWriter(openFileOutput("ip.txt",
MODE_WORLD_WRITEABLE));
out.write(value);
out.close();
savefile("ip.txt", "/data/data/com.SignalLocation/myserversip/" , "ip.txt");
if (worked) {
Toast.makeText(contextd, "Ip is saved", 2000).show();
new Thread(new Runnable() {
public void run() {
SignalLocationService.upload(contextd);  }
```

Σε αυτό το σημείο πρέπει να γίνει σαφές ότι με κάθε νέα απεγκατάσταση και επανεγκατάσταση της εφαρμογής πρέπει να εισάγεται το ip του server για να δημιουργηθεί το συγκεκριμένο αρχείο. Σε αντίθετη περίπτωση όταν η συνάρτηση upload στη service προσπαθήσει να βρει το ip για να συνδεθεί στο server θα βγάλει μήνυμα λάθους: **"can't read ip from file, please insert ip".**

Στην activity δημιουργούνται και άλλα alert-dialogs για να προειδοποιήσουν το χρήστη. Παραδείγματος χάριν εάν ο χρήστης πατήσει το κουμπί back για έξοδο από την εφαρμογή ένα alert-dialog θα εμφανιστεί όπου θα ζητάει επιβεβαίωση για αυτή την ενέργεια. Αυτό είναι πολύ χρήσιμο διότι με την έξοδο από την εφαρμογή χωρίς save χάνονται όλες οι μετρήσεις.

Πολύ σημαντικές συναρτήσεις στην εφαρμογή, είναι αυτές που αφορούν την αποθήκευση στην κάρτα SD.
Αρχικά οι μετρήσεις καταγράφονται σε μια προσωρινή OutputStreamWriter μεταβλητή με το όνομα out που βρίσκεται στην service.

```
private void saveSignalStrength()
{
try
{
String lineOut = so + ","  +lac  + ","  +cid + "," + nt + "," + locpro + "," + lat  +  ","
+  lng +  "," + alt + "," + sp + "," + acc + "," + ss +  ","  + dt  + "," + im  ;
out.write(lineOut);
out.write("\r\n");
}
catch (Throwable t)
{
//Toast.makeText(this, "Exception4: "+t.toString(), 2000).show();
}
}
```

Στη συνέχεια η μεταβλητή αυτή περνάει στην activity με το πάτημα του κουμπιού της αποθήκευσης και αμέσως καλείται η συνάρτηση savefile().

```
public void onClick(DialogInterface dialog, int id) {
out = s.out ;
try {
out.close();
}
catch (IOException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
savefile(targetFileName, "/sdcard/mymeasurements/", outputFileName);
```

Το περιεχόμενο λοιπόν της μεταβλητής out καταγράφεται σε ένα αρχείο .txt και αποθηκεύεται στον φάκελο mymeasurements της κάρτας sd.

```
private void savefile(String targetFileName, String targetfolderpath, String
myoutputFileName)
{
try
{
File mypersonalfolder = new File(targetfolderpath);
mypersonalfolder.mkdirs();
if (mypersonalfolder.canWrite())
{
InputStream is = openFileInput(myoutputFileName);
InputStreamReader isr = new InputStreamReader(is);
BufferedReader ibr = new BufferedReader(isr);
String str;
File of = new File(mypersonalfolder, targetFileName);
FileWriter ofw = new FileWriter(of);
BufferedWriter obw = new BufferedWriter(ofw);
while ((str = ibr.readLine()) != null)
{
obw.write(str);
obw.newLine();
}
is.close();
obw.close();
worked = true ;
}
else
{
worked = false ;
}
}
catch (Throwable t)
{
Toast.makeText(this, "Exception3: "+t.toString(), 5000).show();
}
}
```

Η συνάρτηση αυτή καλείται επίσης και για την αποθήκευση του ip στην κάρτα μνήμης.

Όλα τα πλήκτρα υλοποιούνται με τον ίδιο περίπου τρόπο στην activity.
Παραδείγματος χάριν το πλήκτρο για άνοιγμα του φακέλου των μετρήσεων:

```
cbtn = (Button) findViewById(R.id.change);
cbtn.setOnClickListener(onChangePressed);
Button.OnClickListener onSavePressed = new Button.OnClickListener()
```

```
 {
 public void onClick(View v)
 {
  try
  {
   onSavePressed();
  }
   catch (Exception e)
   {
    e.printStackTrace();
   }
  }
};
```

Όταν πατηθεί λοιπόν αυτό το πλήκτρο καλείται η συνάρτηση που ανοίγει τον φάκελο mymeasurements στην κάρτα sd:

```
private void openMeasurements()
{
Intent intent = new Intent();
intent.setAction(android.content.Intent.ACTION_VIEW);
File file = new File("/sdcard/mymeasurements/");
file.mkdirs();
intent.setDataAndType(Uri.fromFile(file), "*/*");
startActivity(intent);
}
```

Όλα τα πλήκτρα και γενικά όλα τα αντικείμενα στο ui σχηματίζονται μέσω του αρχείου main.xml.
Παραδείγματος χάριν το προηγούμενο button ορίζεται :

```
<Button android:id="@+id/change"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="change server ip"
/>
```

Με παρόμοιο τρόπο ορίζεται στο main.xml και το κείμενο που εμφανίζει τις τιμές των μετρήσεων.
Παραδείγματος χάριν η τιμή του γεωγραφικού πλάτους παρουσιάζεται με τον εξής τρόπο:

```
<TableRow
  android:layout_margin="1dip">
<TextView android:layout_gravity="left"
  android:text="Latitude:"
   android:layout_marginRight="2px"
/>
```

```
<TextView android:id="@+id/latText"
  android:gravity="left"
  android:layout_span="2" />
 </TableRow>
```

Στο AndroidManifest.xml είναι αναγκαίο να οριστούν όλες οι services και οι activities που χρησιμοποιούνται στην εφαρμογή. Έτσι στην συγκεκριμένη περίπτωση οι ***SignalLocationActivity*** και ***SignalLocationService*** ορίζονται ως εξής :

```
<application android:icon="@drawable/icon"
android:label="@string/app_name">
<activity android:name="com.SignalLocation.SignalLocationActivity"
        android:label="@string/app_name">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
<service android:name="com.SignalLocation.SignalLocationService" />
</application>
```

Τέλος στο AndroidManifest.xml ορίζονται όπως ήδη έχει αναφερθεί και οι άδειες που χρειάζονται από το λειτουργικό για να έχει πρόσβαση η εφαρμογή σε κάποιες λειτουργίες του τερματικού. Στην συγκεκριμένη εφαρμογή οι άδειες αυτές είναι:

```
<uses-permission android:name=
"android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name=
"android.permission.ACCESS_COARSE_UPDATES"/>
<uses-permission android:name=
"android.permission.ACCESS_GPS"/>
<uses-permission android:name=
"android.permission.ACCESS_LOCATION"/>
<uses-permission android:name=
"android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name=
"android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name=
"android.permission.READ_PHONE_STATE"/>
<uses-permission android:name=
"android.permission.INTERNET"/>
<uses-permission android:name=
"android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name=
"android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name=
"android.permission.CHANGE_WIFI_STATE"/>
<uses-permission android:name=
"android.permission.WAKE_LOCK"/>
```

ΚΕΦΑΛΑΙΟ 5: Δοκιμές και Αποτελέσματα

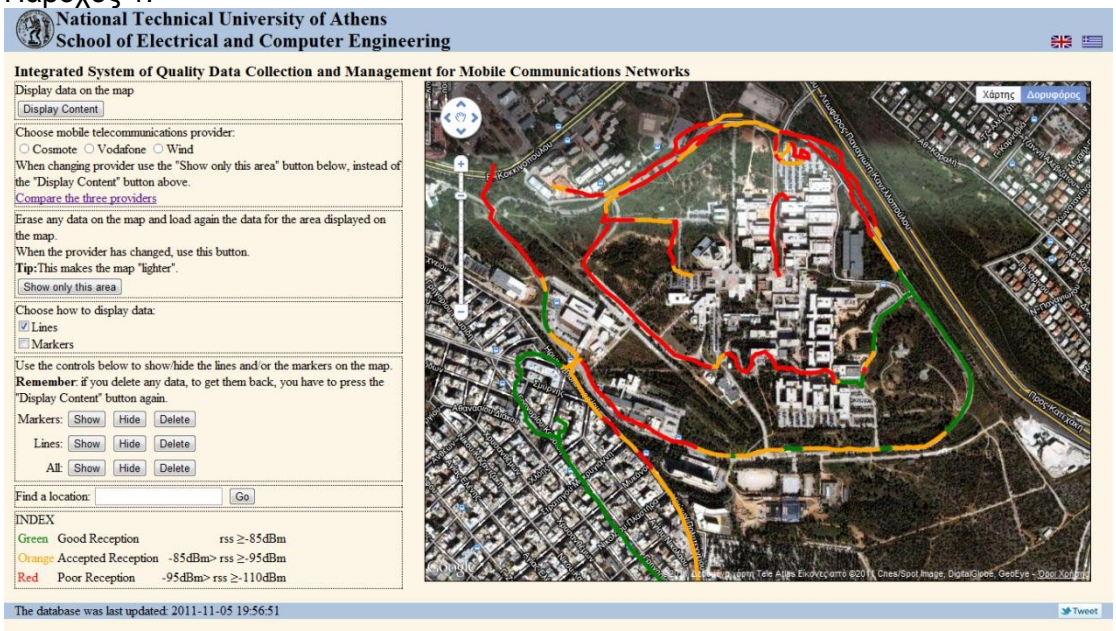
Στο πλαίσιο της παρούσας διπλωματικής και για καλύτερη παρουσίαση του προτεινόμενου συστήματος πραγματοποιήθηκε μία σειρά μετρήσεων στην περιοχή της Πολυτεχνειούπολης, στην Αθήνα. Οι μετρήσεις έγιναν ταυτόχρονα με τρία ίδια HTC Desire κινητά και τρεις διαφορετικές κάρτες sim, μία για κάθε ελληνικό πάροχο. Τα κινητά βρίσκονταν μέσα σε κινούμενο αυτοκίνητο, κοντά σε παράθυρο ώστε να λαμβάνουν σήμα GPS.

Αφού έγιναν οι μετρήσεις στα κινητά τερματικά, ανέβηκαν τα αρχεία καταγραφής στον εξυπηρετητή με IP 147.102.7.162 και στη συνέχεια φορτώθηκαν οι εγγραφές στη βάση δεδομένων με τη βοήθεια του SignalLocation/maintenance.php, όπως έχει περιγραφεί στο κεφάλαιο 4.2.

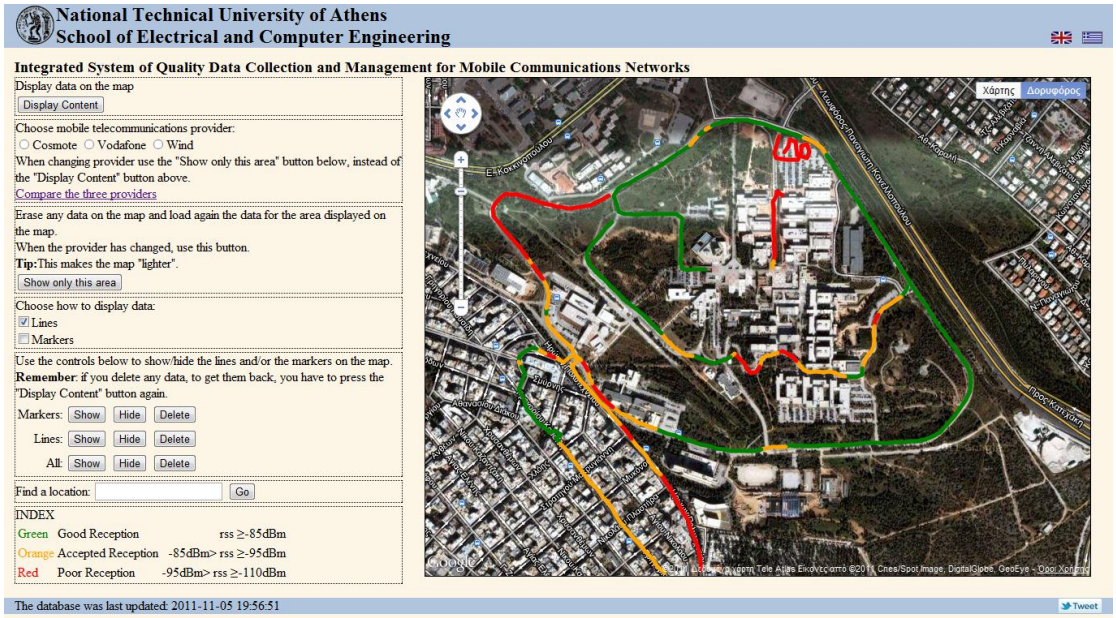
Για τα αποτελέσματα των μετρήσεων αποτυπωμένα σε χάρτη, ο χρήστης μπορεί να πληκτρολογήσει στον browser του 147.102.7.162/SignalLocation και αφού επιλέξει τον πάροχο που τον ενδιαφέρει να πατήσει το κουμπί “Display Content”.

Εστιάζοντας στην περιοχή της Πολυτεχνειούπολης και αλλάζοντας την προβολή του Google map από χάρτη σε δορυφόρο, ώστε να φαίνονται οι δρόμοι της Πολυτεχνειούπολης, προκύπτουν τα εξής:

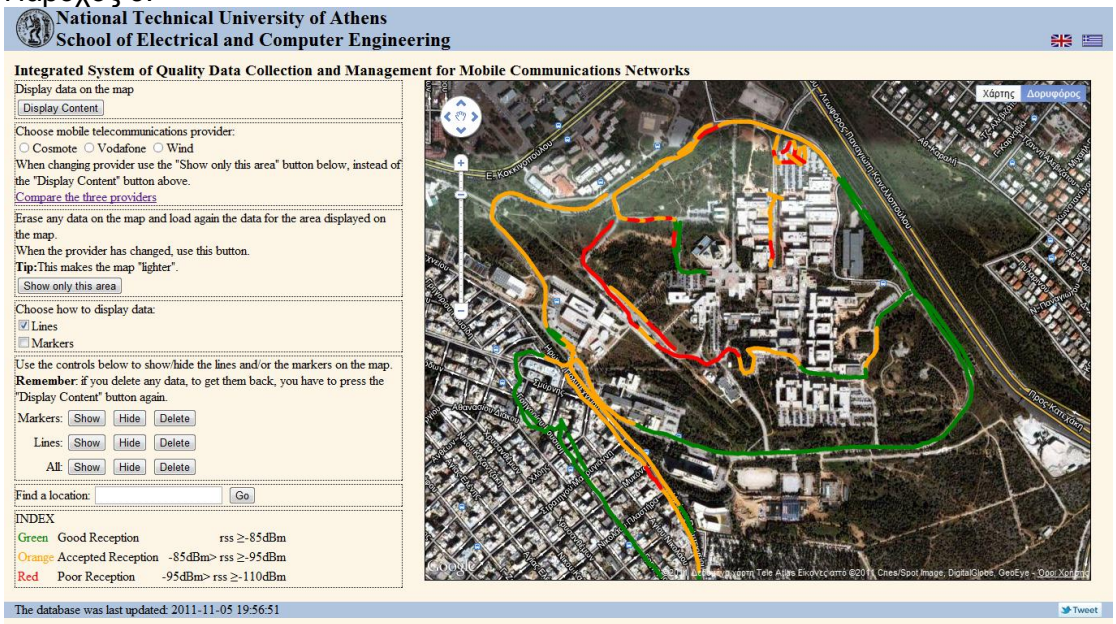
Πάροχος 1:



Πάροχος 2:



Πάροχος 3:



Ας σημειωθεί ότι η εναλλαγή μεταξύ εταιρειών γίνεται επιλέγοντας το όνομα της εταιρείας και στη συνέχεια πατώντας το κουμπί “Show only this area” και όχι το “Display Content”, γιατί τότε δεν θα καθαριζόταν ο χάρτης από τα δεδομένα της προηγούμενης εταιρείας.

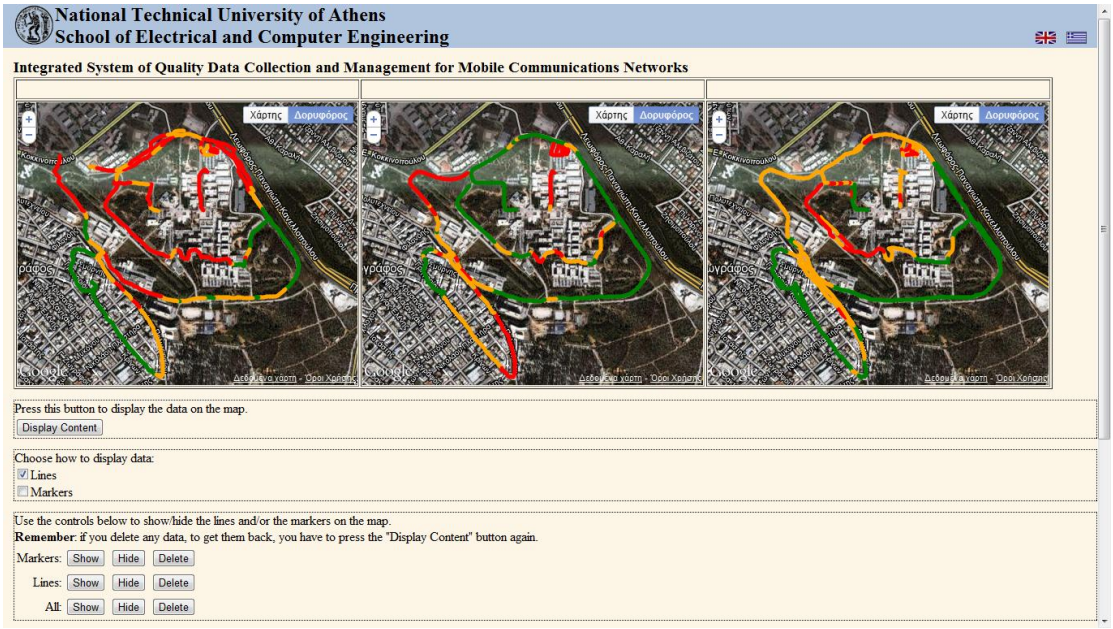
Το σήμα έχει εναλλαγές ακόμη και σε αυτή τη μικρή σχετικά γεωγραφική περιοχή. Αυτό οφείλεται στο φαινόμενο των πολλαπλών οδεύσεων. Τα κτίρια, τα αυτοκίνητα, τα δέντρα που υπάρχουν στην Πολυτεχνειούπολη προκαλούν διαδοχικές ανακλάσεις και διάχυση στα ηλεκτρομαγνητικά κύματα που έρχονται από τις κεραίες κινητών επικοινωνιών. Το κινητό τερματικό είναι πιθανό να λαμβάνει περισσότερες από μία εκδοχές του ίδιου σήματος, με διαφορετική ισχύ και φάση η κάθε μία. Έτσι, ακόμη και σε δύο διαδοχικές μετρήσεις είναι σπάνιο η τιμή της ισχύος του σήματος να παραμένει σταθερή.

Εμφανίζοντας τους Markers με το κουμπί “Show” και εξαφανίζοντας τις γραμμές με το κουμπί “Hide” παρατηρείται ότι στην κάτω δεξιά γωνία του χάρτη ο πάροχος 3 παρουσιάζει αυξημένη ισχύ σήματος με μέγιστη τιμή -63dBm και σε δίκτυο 3ης γενιάς.

Αυτό δικαιολογείται από το γεγονός ότι στην συγκεκριμένη περιοχή υπάρχουν κοντά κεραίες του παρόχου αυτού, οπότε οι απώλειες στην ισχύ του σήματος λόγω διάδοσής του στο χώρο είναι λίγες.

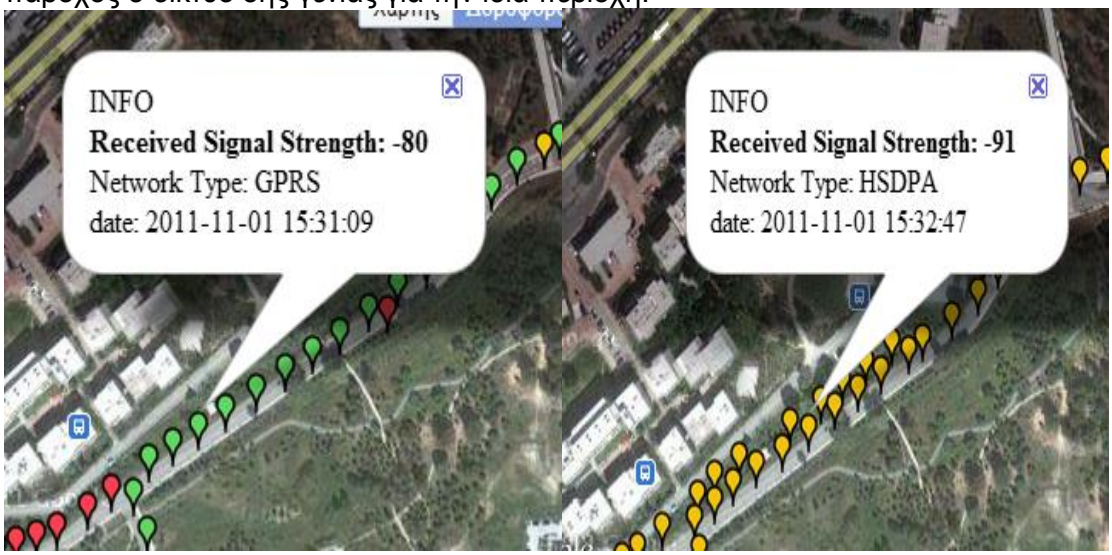
Για καλύτερη συγκριτική αξιολόγηση των τριών παρόχων, ο χρήστης μπορεί να χρησιμοποιήσει την ιστοσελίδα 147.102.7.162/SignalLocation/comparison.html ή από την προηγούμενη σελίδα να πατήσει τον σύνδεσμο “Compare the three providers”.

Και πάλι κάνοντας κλικ στο κουμπί “Display Content” ο χρήστης λαμβάνει το εξής:



Η κάλυψη είναι διαφορετική στους τρεις παρόχους. Σημεία που στον πάροχο 1 για παράδειγμα έχουν χαμηλή ποιότητα λήψης, στον πάροχο 2 έχουν υψηλή ποιότητα λήψης. Αυτό οφείλεται στη διαφορετική θέση στο χώρο των κεραιών των τριών παρόχων, καθώς και στη διαφορετική ισχύ εκπομπής των κεραιών.

Με μια γρήγορη ματιά φαίνεται ότι ο δεύτερος πάροχος προσφέρει καλύτερης ποιότητας σήμα σε μεγαλύτερη γεωγραφική περιοχή. Ωστόσο, χρησιμοποιώντας τους markers βλέπουμε ότι ο πάροχος 2 προσφέρει δίκτυο 2ης γενιάς, ενώ ο πάροχος 3 δίκτυο 3ης γενιάς για την ίδια περιοχή.



ΚΕΦΑΛΑΙΟ 6: Επίλογος – Συμπεράσματα

6.1 Συμπεράσματα

Η συλλογή και διαχείριση δεδομένων ποιότητας δικτύων κινητών επικοινωνιών αφορά πρωτίστως τους παρόχους κινητών επικοινωνιών. Βοηθά στον έλεγχο και τη συντήρηση των δικτύων, καθώς και στην αντιμετώπιση προβλημάτων στις υπάρχουσες εγκαταστάσεις. Ακόμη, βασιζόμενοι στα αποτελέσματα των μετρήσεων, οι πάροχοι μπορούν να αποφασίσουν ποια σημεία του δικτύου χρήζουν επέκτασης ή αναβάθμισης. Τέλος, η ύπαρξη δεδομένων ποιότητας δικτύου από ανταγωνιστικούς παρόχους δίνει τη δυνατότητα σε μια εταιρεία κινητών επικοινωνιών να γνωρίζει ποιους τομείς πρέπει να αναπτύξει ώστε να έχει ελκυστικό προφίλ στην αγορά.

Επιπροσθέτως, ένα αντικειμενικό σύστημα συλλογής και απεικόνισης δεδομένων ποιότητας δικτύων κινητών επικοινωνιών αποτελεί σημαντικό εργαλείο και για τους τελικούς χρήστες των δικτύων. Συγκρίνοντας την κάλυψη που παρέχουν οι διάφοροι πάροχοι, οι καταναλωτές μπορούν να επιλέξουν τα πακέτα υπηρεσιών που πραγματικά καλύπτουν τις ανάγκες τους στις περιοχές που τους ενδιαφέρουν.

6.2 Μελλοντικές Επεκτάσεις

Η εφαρμογή που αναπτύχθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας σχεδιάστηκε και οργανώθηκε έτσι ώστε να είναι πλήρως επεκτάσιμη. Έχει χωριστεί σε διαφορετικά τμήματα ανάλογα με τις προσφερόμενες λειτουργίες, παρέχοντας έτσι τη δυνατότητα στους προγραμματιστές να επεκτείνουν συγκεκριμένες λειτουργίες χωρίς να επηρεάζονται τα υπόλοιπα τμήματα της εφαρμογής. Μελλοντικές επεκτάσεις της εφαρμογής αφορούν:

- Υλοποίηση ασφαλούς μεταφοράς. Ένα θέμα που δεν υλοποιήθηκε πλήρως είναι αυτό της ασφαλούς μεταφοράς των δεδομένων μεταξύ της κινητής συσκευής και του server. Παρόλο που χρησιμοποιήθηκαν όλες εκείνες οι προγραμματιστικές μέθοδοι για την ασφάλεια της εφαρμογής, δεν γίνεται κρυπτογράφηση των http μηνυμάτων. Η απόφαση αυτή λήφθηκε έτσι ώστε να απλοποιηθεί η υλοποίηση της εφαρμογής. Έτσι, σε μελλοντικές εκδόσεις της εφαρμογής μπορεί να συμπεριληφθεί η δυνατότητα κρυπτογράφησης με κάποιο πρωτόκολλο όπως τα SSL, TSL.
- Διαφοροποίηση ανάλογα με το είδος δικτύου. Ενώ στην εφαρμογή γίνεται καταγραφή του είδους του δικτύου, δηλαδή αν πρόκειται για GPRS, EDGE, UMTS, HSDPA κτλ, τα δεδομένα απεικονίζονται χωρίς να λαμβάνεται αυτό υπόψη. Έτσι, μπορεί να φαίνεται ότι ένας πάροχος έχει καλύτερη κάλυψη σε μία περιοχή από έναν άλλο, αλλά αυτό να οφείλεται στο γεγονός ότι οι μετρήσεις έγιναν σε δίκτυο 2ης γενιάς, ενώ του άλλου παρόχου σε δίκτυο 3ης γενιάς. Σε μελλοντική επέκταση θα μπορούσε να δοθεί η δυνατότητα επιλογής από τον χρήστη για το είδος του δικτύου που τον ενδιαφέρει να απεικονίσει.
- Ευέλικτη διαχείριση των αρχείων καταγραφής. Ο διαχειριστής έχει τη δυνατότητα να δει ποια αρχεία έχουν ανέβει στον εξυπηρετητή και στη βάση και να διαγράψει κάποια απ' αυτά σύμφωνα με κάποιο χρονικό κατώφλι. Ωστόσο, θα ήταν επιθυμητό ο ίδιος ο χρήστης της εφαρμογής του κινητού να μπορεί να επιλέξει ποια αρχεία ανεβαίνουν στον εξυπηρετητή, γιατί για παράδειγμα γνωρίζει ότι σε μία καταγραφή η κεραία του τερματικού ήταν

καλυμμένη με αποτέλεσμα οι μετρήσεις της λαμβανόμενης ισχύος να έχουν μεγάλο σφάλμα. Επιπλέον, και από την πλευρά του διαχειριστή θα ήταν βολικό να μπορεί να επιλέξει χειροκίνητα ποια αρχεία φορτώνονται στη βάση και ποια διαγράφονται, ανεξάρτητα από την ημερομηνία δημιουργία τους.

- Αυξημένη ασφάλεια της βάσης δεδομένων, της σελίδας διαχείρισης και της αλλαγής IP του server. Οι κωδικοί για την πρόσβαση στη βάση, καθώς και για την πρόσβαση στην ιστοσελίδα του διαχειριστή αφενός είναι οι ίδιοι, αφετέρου βρίσκονται στον ίδιο φάκελο με τα html αρχεία, εκτεθειμένοι στο internet. Αυτό έγινε για ευκολία της υλοποίησης, αλλά κάνει την εφαρμογή ευάλωτη σε κακόβουλες επιθέσεις από το διαδίκτυο. Σε μελλοντικές επεκτάσεις θα μπορούσε να υπάρχει ένας αλγόριθμος κρυπτογράφησης των κωδικών πρόσβασης και ασφαλέστερη αποθήκευσή τους. Επιπροσθέτως, ο κωδικός που απαιτείται για την αλλαγή IP του εξυπηρετητή στον οποίο θα ανέβουν οι μετρήσεις είναι απλοϊκός και ίδιος για όλα τα κινητά τερματικά. Θα μπορούσε ο κωδικός αυτός να έχει μια πιο προηγμένη μορφή, όπως για παράδειγμα να αποτελεί κωδικοποίηση του imei του κινητού τερματικού.
- Περισσότερες πληροφορίες για την ποιότητα του δικτύου. Στην παρούσα διπλωματική έγινε καταγραφή μόνο της λαμβανόμενης ισχύς, αλλά η ποιότητα υπηρεσίας περιλαμβάνει και άλλες παραμέτρους όπως η καθυστέρηση και η πιθανότητα αποκλεισμού μιας κλήσης. Επίσης, η συνεχώς αυξανόμενη ζήτηση για πρόσβαση στο διαδίκτυο μέσω των δικτύων κινητών επικοινωνιών, θέτει το προσφερόμενο bitrate (ρυθμό μετάδοσης δεδομένων) ως έναν από τους σημαντικότερους δείκτες αξιολόγησης ενός δικτύου. Οπότε, πιθανή μελλοντική επέκταση θα ήταν και η καταγραφή επιπλέον δεικτών και η απαραίτητη επεξεργασία τους για απεικόνιση.
- Κρυπτογράφηση του imei στη βάση δεδομένων. Το imei του κινητού τερματικού που χρησιμοποιήθηκε για τις εκάστοτε μετρήσεις καταγράφεται και αποθηκεύεται αυτούσιο. Αυτό είναι αντίθετο με την προστασία των ιδιωτικών δεδομένων, εφόσον η εφαρμογή του κινητού μπορεί να χρησιμοποιηθεί και από απλούς χρήστες. Μία επέκταση θα ήταν λοιπόν η κρυπτογράφηση του imei (hash imei) πριν την αποθήκευσή του στη βάση δεδομένων.

Βιβλιογραφία

[1] Μ. Ε. Θεολόγου, “Δίκτυα Κινητών και Προσωπικών Επικοινωνιών”, Εκδόσεις Τζιόλα, 2008

[2] Α. Κανάτας, Φ. Κωνσταντίνου, Γ. Πάντος, “Συστήματα Κινητών Επικοινωνιών”, Εκδόσεις Παπασωτηρίου, Αθήνα, 2008

[3] International Telecommunication Union, ITU Statshot, Issue 7, August 2011, http://www.itu.int/net/pressoffice/stats/2011/03/index.aspx

[4] International Telecommunication Union, “The World in 2011: ICT Facts and Figures”, http://www.itu.int/ITU-D/ict/facts/2011/material/ICTFactsFigures2011.pdf

[5] Google, “Google Maps Javascript API V3 Reference”, http://code.google.com/intl/el-GR/apis/maps/documentation/javascript/reference.html

[6] Α. Ανδρονικάκης, “Σύστημα Προσδιορισμού Θέσης Σταθμών Βάσης σε Δίκτυα Κινητών Επικοινωνιών”, 2010

[7] Α. Ντάκας, “Σύστημα Καταγραφής της Εμπειρίας Κινητών Χρηστών σε Δίκτυα Ασύρματων Επικοινωνιών”, 2011

[8] Γ. Ι. Στασινόπουλος, “Διαδίκτυο και Εφαρμογές”, Αθήνα, 2010

[9] PHP Manual, 2011, http://www.php.net/manual/en/manual.php

[10] http://www.w3schools.com

[11] MySQL 5.5 Reference Manuals, http://dev.mysql.com/doc/refman/5.5/en/index.html

[12] Android Developers, http://developer.android.com/index.html

[13] Εθνική Επιτροπή Τηλεπικοινωνιών και Ταχυδρομείων (ΕΕΤΤ), http://www.eett.gr/opencms/opencms/EETT

[14] Ascom, TEMS for Wireless Network Testing and Measurement, http://www.ascom.com/en/index/group/divisions/network-testing-home.htm

[15] Anite, Nemo Network Testing Solutions, http://www.anite.com/anite/en/solutions/nemotesting

[16] SwissQual, Network Benchmarking, Optimization and Service Monitoring, http://www.swissqual.com/index.php/systems/