Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής
Πρόγραμμα Μεταπτυχιακών Σπουδών
«Προηγμένα Συστήματα Πληροφορικής»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Ανάλυση και σχεδιασμός αναλυτή διαδρομών (Route Analyzer) σε πλατφόρμα Android** **Analysis and design of Route Analyzer application on Android platform.**
Ονοματεπώνυμο Φοιτητή	**Κωσταντίνος Πολάκης**
Πατρώνυμο	**Μιχαήλ**
Αριθμός Μητρώου	**ΜΠΣΠ/ 13092**
Επιβλέπων	**Dr. Αλέπης Ευθύμιος, Λέκτορας**

Ημερομηνία Παράδοσης **Μάιος 2015**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή) (υπογραφή) (υπογραφή)

Όνομα Επώνυμο Βαθμίδα	Όνομα Επώνυμο Βαθμίδα	Όνομα Επώνυμο Βαθμίδα

Περιεχόμενα

Περίληψη

Στην παρούσα πτυχιακή εργασία παρουσιάζεται η ιδέα και η δημιουργία μιας καινοτόμας εφαρμογής που απευθύνεται στους οδηγούς οχημάτων.

Πιο συγκεκριμένα δημιουργήσαμε μια εφαρμογή η οποία εκτελείται σε έξυπνα κινητά τηλέφωνα και αποθηκεύει τα δεδομένα των διαδρομών που πραγματοποιεί ο οδηγός ενός οχήματος. Με τον τρόπο αυτό, ο χρήστης της εφαρμογής μπορεί να γνωρίζει την ακριβή διαδρομή που πραγματοποίησε ο οδηγός του οχήματος καθώς και τα σημεία της διαδρομής όπου πραγματοποίησε κάποια στάση. Η μετέπειτα επεξεργασία των δεδομένων αυτών οδηγούν σε χρήσιμα συμπεράσματα για την οδική συμπεριφορά των οδηγών. Επιπλέον, συγκρίνοντας τις διάφορες διαδρομές μεταξύ τους μπορούμε να καταλάβουμε ότι μια διαδρομή είναι πιο σύντομη ή πιο οικονομική.

Θεωρούμε, πως η εφαρμογή μας κρίνεται ιδιαίτερα σημαντική, καθώς στις ημέρες μας το αυτοκίνητο αποτελεί το βασικό μέσο μετακίνησής του μεγαλύτερου ποσοστού των πολιτών. Μέσω της εφαρμογής μας ο χρήστης έχει πλέον τα δεδομένα που θα τον οδηγήσουν στην αποδοτικότερη χρήση του μέσου αυτού. Οι έντονοι ρυθμοί της καθημερινότητας των πολιτών απαιτούν τη χρήση των συντομότερων διαδρομών κατά τη μετακίνηση τους. Επίσης, η κλιματική αλλαγή, συνέπεια της μόλυνσης του περιβάλλοντος και από τους ρύπους που εκπέμπουν τα οχήματα, σε συνάρτηση με τη συνεχώς αυξανόμενη τιμή των καυσίμων καθιστούν απαραίτητη την αποδοτικότερη διαχείριση του καυσίμου.

Τέλος, σημαντική είναι και η συνδρομή της εφαρμογής στη μείωση των τροχαίων ατυχημάτων που οφείλονται στη μη τήρηση των κανόνων οδικής κυκλοφορίας μέσω καταγραφή της οδικής συμπεριφοράς των οδηγών. Πιο συγκεκριμένα, αποθηκεύοντας τη ταχύτητα του οχήματος σε τακτά χρονικά διαστήματα κατά μήκος μια διαδρομής, μας δίδεται η δυνατότητα να ελέγχουμε κατά πόσο τηρούνται τα όρια ταχύτητας τα οποία αποτελούν μια από τις υψηλότερες αιτίας πρόκλησης τροχαίων ατυχημάτων.

Στις σελίδες που ακολουθούν παρουσιάζεται η ιδέα της εφαρμογής, ο σκοπός της και ένα από τα βασικότερα λειτουργικά συστήματα των έξυπνων κινητών τηλεφώνων σήμερα. Στο δεύτερο μέρος, το οποίο υλοποιήθηκε από τον ΠΑΠΟΥΤΣΗ Εμμανουήλ, παρουσιάζεται η τεχνική κατασκευής της εφαρμογής.

Λέξεις – κλειδιά

Android, Bluetooth, OBD-II, GPS, έξυπνο κινητό τηλέφωνο, τηλεμετρία, αυτοκίνητο, εντοπισμός θέσης μέσω GPS, καταγραφή διαδρομής

Abstract

In this thesis, we present an idea and an innovative application which is designed for vehicles users.

More specifically, we designed an application that runs on smartphones and records the data of the routes that the drive follows. The user of the application knows the exact points of the route he followed and the stops he made along that route. Through the afterwards study of this data, we can make conclusions about the driving characteristics of each driver. In advance comparing one route with another we can configure the faster or more efficient route.

We believe that our application is very important especially now days, where the car is the main mean that most people use for their movement. Through this application the user has access to the data that will help him use the vehicle more efficient. Nowadays, people need to move to their destination using the faster route. Also the climate change, consequence of the environment pollution also form vehicles exhaust gas and the continuously increase of the cost of fuel, imposes the more efficient use of fuel.

Our application can support the reduction of car accidents, due to disobeying to traffic laws as it records the driving characteristics of the driver. Recording the speed of the vehicle at several points of the route a supervisor is able to check whether vehicles driver follows the speed limit of certain points on the route. Those speed limits are the most common reason for car accidents.

In the following chapters, the reader can understand better the idea of our application, it's intense and he will browse in one of the most important smartphone operation systems of our age. Finally, at the second part witch was carried out by PAPOUTSIS Emmanouil, the reader will understand the way that this application was built.

Keywords

Android, Bluetooth, OBD-II, GPS, Smartphone, telemetry, vehicle, GPS geopoint calculation, Route Tracking

1. Εισαγωγή

1.1 Στόχος Της Παρούσας Πτυχιακής.

Στόχος της παρούσας πτυχιακής είναι η ανάπτυξη μιας εφαρμογής, για έξυπνα κινητά τηλέφωνα η οποία θα καταγράφει δεδομένα των διαδρομών που πραγματοποιεί ο χρήστης του κινητού τηλεφώνου. Τα δεδομένα αυτά μεταξύ άλλων θα αποτελούνται από τις συντεταγμένες της διαδρομής, τη χρονική της διάρκεια, τη χρονική διάρκεια των στάσεων, την διανυθείσα απόσταση και τη κατανάλωση καυσίμου του οχήματος. Η διαδρομές αυτές θα αποθηκεύονται στο κινητό τηλέφωνο και θα είναι προσβάσιμες από το χρήστη οποιαδήποτε στιγμή.

Με τον τρόπο αυτό ο χρήστης θα μπορεί να συγκρίνει τις διαδρομές που έχει αποθηκεύσει και να επιλέγει είτε την ποιο γρήγορη, είτε την ποιο οικονομική. Επίσης θα μπορεί να συγκρίνει διάφορα οχήματα ώστε να συμπεράνει ποιο όχημα είναι οικονομικότερο η ποιο γρήγορο. Ομοίως, μια εταιρία που πραγματοποιεί διανομές θα μπορούσε να παρακολουθεί τους διανομείς της ώστε γνωρίζει το δρομολόγιο του κάθε υπαλλήλου, τις στάσεις που έκανε καθώς και το καύσιμο που κατανάλωσε.

Η εφαρμογή θα εκτελείται σε συσκευές με λειτουργικό σύστημα Android και για το λόγο αυτό στο σημείο αυτό κρίνεται σκόπιμη μια αναφορά στο λειτουργικό σύστημα αυτό ώστε ο αναγνώστης να εξοικειωθεί με τη λειτουργία του. Επιπλέον η εφαρμογή χρησιμοποιεί την υπηρεσία Google Maps της εταιρίας Google για να εμφανίζει τη θέση του χρήστη αλλά και των διαδρομών που έχει εκτελέσει. Τέλος τα δεδομένα που αφορούν την ταχύτητα με την οποία κινείται καθώς και τη κατανάλωση καυσίμου του οχήματος προκύπτουν από το ενσωματωμένο σύστημα διάγνωσης του οχήματος. Έτσι μετά την αναφορά στο λειτουργικό σύστημα ακολουθεί μια περιγραφή της υπηρεσίας Google Maps και ακολούθως μια αναφορά στο ενσωματωμένο σύστημα διάγνωσης οχημάτων.

2. Το Λειτουργικό Σύστημα Android.

2.1 Εισαγωγή

"This is the droid you're looking for". Ξεκινώντας με τη φράση αυτή, στις 12 Νοεμβρίου του 2007, το επίσημο blog προγραμματιστών για Android πραγματοποίησε την πρώτη του ανάρτηση, με την οποία ενημέρωνε το κοινό για τη διάθεση μια πρώιμης δοκιμαστικής έκδοσης ενός νέου λειτουργικού συστήματος για κινητά τηλέφωνα με το όνομα Android. Επίσης, έκανε γνωστά τα διαθέσιμα εργαλεία ανάπτυξης καθώς και την εικονική μηχανή όπου το νέο λειτουργικό θα μπορούσε να τρέξει για δοκιμές. Τέλος, ανακοίνωνε ένα διαγωνισμό, με έπαθλα συνολικής αξίας 10 εκατομμυρίων δολαρίων, για την ανάπτυξη νέων και πρωτοποριακών Android εφαρμογών. Πίσω από τη χρηματοδότηση του διαγωνισμού αυτού, βρισκόταν η Google, ένας από τους βασικούς παράγοντες που οδήγησαν στη δημιουργία της Open Handset Alliance (μια εβδομάδα πριν) και στην ανάπτυξη του Android.

Η Open Handset Alliance είναι μια πρωτοβουλία συνεργασίας μεγάλων εταιρειών τεχνολογίας, με στόχο την εξέλιξη των κινητών τεχνολογιών. Πιο συγκεκριμένα, διαβάζουμε στο επίσημο site του οργανισμού:

Τι είναι η Open Handset Alliance;

"Είναι μια ομάδα 84 εταιρειών τεχνολογίας και κινητής τηλεφωνίας που ενώθηκαν για να επιταχύνουν την καινοτομία στις κινητές τεχνολογίες και να προσφέρουν στους καταναλωτές μια πιο πλούσια, λιγότερο ακριβή και πιο καλή εμπειρία κινητών τεχνολογιών. Όλοι μαζί αναπτύξαμε το Android, την πρώτη ολοκληρωμένη, ανοιχτού κώδικα και δωρεάν πλατφόρμα για κινητά. Δεσμευόμαστε να αναπτύξουμε εμπορικά συσκευές και υπηρεσίες χρησιμοποιώντας την πλατφόρμα Android."

Η επιχειρηματική αυτή ένωση περιελάμβανε στο ξεκίνημά της 34 εταιρείες τεχνολογίας, με βασικό μέλος και οργανωτή του εγχειρήματος την Google. Εκτός της Google, πολλές άλλες εταιρείες κολοσσοί στον τομέα της τεχνολογίας έγιναν μέλη, όπως εταιρείες κινητής τηλεφωνίας (T-mobile, China mobile, Telecom Italia), εταιρείες κατασκευής υλικού (HTC, LG, Sony, Motorola, Samsung, Nvidia) κ.ά. Η συνεργασία αυτή έφερε στο φως το πρώτο αποτέλεσμα περίπου ένα χρόνο αργότερα. Στις 22 Οκτωβρίου του 2008, το HTC Dream ή αλλιώς T-Mobile G1, ήταν το πρώτο κινητό σε κυκλοφορία που έτρεχε Android.

Η ιστορία, όμως του Android ξεκινά κάποια χρόνια νωρίτερα. Πιο συγκεκριμένα, τον Οκτώβριο του 2003, οι Andy Rubin, Rich Miner, Nick Sears και Chris White, όλοι τους στελέχη από εταιρείες τεχνολογίας και κινητής τηλεφωνίας, ίδρυσαν την Android inc. Για τα επόμενα χρόνια δούλευαν σε απόλυτη μυστικότητα, ώσπου στις 17 Αυγούστου του 2005 η Android εξαγοράστηκε από την Google. Παρά την εξαγορά, οι βασικοί συντελεστές της μικρής εταιρείας παρέμειναν στις θέσεις τους και δημιούργησαν μια ομάδα ανάπτυξης μέσα στην κολοσσό εταιρεία. Η κίνηση αυτή της Google έδωσε τροφή σε πολλά δημοσιεύματα του τύπου, τα οποία ανέφεραν πως η εταιρεία προσπαθούσε να μπει στην αγορά των κινητών τεχνολογιών κι όχι άστοχα. Η φημολογία συνεχίστηκε μέχρι και το Νοέμβριο του 2007, όπου κι έγιναν τα αποκαλυπτήρια του Android, ως λειτουργικού συστήματος για κινητές συσκευές και της **Open Handset Alliance.** Από το σημείο εκείνο και μετά οι εξελίξεις ήταν καταιγιστικές. Νέες εκδόσεις του λειτουργικού, ανά μερικούς μήνες, με σημαντικές καινοτομίες κι αλλαγές στον τρόπο χρήσης των κινητών τηλεφώνων. Από ένα πειραματικό λειτουργικό σύστημα, το Android έχει φτάσει σήμερα να κατέχει τη πρώτη θέση, στην κατάταξη για τα πιο χρησιμοποιούμενα λειτουργικά συστήματα smart phones.

Ενδεικτικό της ώθησης που δέχτηκε αλλά και των εντυπώσεων που προκάλεσε το Android, είναι ο αριθμός και η συχνότητα των νέων εκδόσεων, από την ημερομηνία πρώτης του κυκλοφορίας ως εμπορικό προϊόν και μετά, καθώς και οι καινούριες ιδέες και δυνατότητες που εμφανίζονταν σε κάθε νέα έκδοση. Το πιο σημαντικό όμως γεγονός στάθηκε η δυνατότητα του Android να τροποποιείται. Έτσι, πολλοί κατασκευαστές μπορούν να εκδίδουν παραμετροποιημένα λειτουργικά μαζί με κάθε συσκευή τους. Η πρακτική αυτή υιοθετήθηκε από πολλούς κατασκευαστές για το θέμα του οπτικού πληκτρολογίου. Η υλοποίηση πληκτρολογίου, που είχε ενσωματωθεί στο Android, ήταν πολύ βασική και δεν ικανοποιούσε τις απαιτήσεις. Έτσι πολλές κατασκευάστριες εταιρείες αναγκάστηκαν να αναπτύξουν τις δικές τους εφαρμογές- πληκτρολόγια και να τις ενσωματώνουν στις ξεχωριστές εκδόσεις που δημιουργούσαν για τις διάφορες σειρές προϊόντων τους.

2.2 Οι Εκδόσεις Android

Η πρώτη νέα έκδοση του Android (1.1) κατέφτασε λίγους μόνο μήνες μετά την κυκλοφορία του *HTC Dream.* Η έκδοση αυτή δεν προσέφερε πολλές καινούριες δυνατότητες, αλλά διόρθωνε μια λίστα από σφάλματα (bugs) της πρώτης έκδοσης. Το ενδιαφέρον ήταν ο τρόπος και η ευκολία με την οποία μπορούσε κάποιος να αναβαθμίσει το λειτουργικό του σύστημα, χωρίς υπερβολικά περίπλοκες διαδικασίες, κάτι πρωτοποριακό για την εποχή του.

Ούτε δυο μήνες αργότερα, στις 30 Απριλίου 2009 κυκλοφόρησε η δεύτερη διορθωτική έκδοση του λειτουργικού (1.5) με κωδικό όνομα *Cupcake*. Παρόμοια με την πρώτη, οι αλλαγές περιορίζονταν σε κάποιες μικρές βελτιώσεις του γραφικού περιβάλλοντος καθώς και σε προσαρμογές στις διάφορες λειτουργίες, που όμως δύσκολα γίνονταν αντιληπτές από τους χρήστες. Μεγάλη εξαίρεση αποτέλεσε η προσθήκη ενός εικονικού πληκτρολογίου στην οθόνη αφής, αφού το HTC Dream διέθετε μόνο φυσικό πληκτρολόγιο το οποίο αποκαλυπτόταν σύροντας το δεύτερο κομμάτι της συσκευής πίσω από την οθόνη. Είναι ενδιαφέρον το γεγονός πως μέχρι και την τρίτη έκδοσή του, το Android δεν είχε το χαρακτηριστικό πλέον πληκτρολόγιο της οθόνης αφής, όμως αυτή η τακτική, καθώς και η μη ικανοποιητική πρώτη υλοποίησή του, ώθησε τις εταιρείες που το χρησιμοποίησαν, να αναπτύξουν τις δικές τους λύσεις. Έτσι έγινε η αρχή για να φτάσουμε στα διάφορα *οπτικά θέματα* (*themes*) που πολλές εταιρείες έχουν δημιουργήσει για τις δικές τους εκδόσεις του Android. Κάτι ακόμη που ξεχωρίζει την έκδοση 1.5 είναι ότι ονομάστηκε “Cupcake”, και αποτέλεσε την απαρχή μιας παράδοσης που θέλει κάθε καινούρια έκδοση Android να παίρνει το όνομά της από κάποιο γλύκισμα, με αλφαβητική σειρά.

Στις 15 Σεπτεμβρίου του ίδιου έτους, μετά την έκδοση Cupcake, οι χρήστες και προγραμματιστές υποδέχτηκαν την έκδοση 1.6 (*Donut)*. Αυτή τη φορά εγκαινιάστηκε η υποστήριξη πολλών διαφορετικών αναλύσεων οθονών, σε αντίθεση με τις μέχρι τότε εκδόσεις που υποστήριζαν μόνο αναλύσεις 320 x 480. Ακόμη, προστέθηκε υποστήριξη για το CDMA πρωτόκολλο, κάτι που μικρή σημασία είχε για τους Ευρωπαίους καταναλωτές και επαγγελματίες της πληροφορικής, αφού το συγκεκριμένο πρωτόκολλο χρησιμοποιείται κυρίως στις Ηνωμένες Πολιτείες. Αυτές, μαζί με άλλες βελτιώσεις του γραφικού περιβάλλοντος αλλά και της διαδικτυακής πλατφόρμας αγοράς εφαρμογών της Google (Android Market) αποτέλεσαν την έκδοση Donut.

Πριν ακόμη προλάβουν οι χρήστες να εμπεδώσουν, καλά- καλά, τις αλλαγές της έκδοσης 1.6, μερικές εβδομάδες αργότερα, στις αρχές Νοεμβρίου βγήκε σε κυκλοφορία η έκδοση 2.0 (*Eclair*). Μαζί με μερικές σημαντικές αλλαγές στη διαχείριση λογαριασμών χρηστών, τηλεφωνικών επαφών, γραφικών βελτιώσεων και άλλων, εγκαινιάστηκε η εφαρμογή *Google Maps*. Η συγκεκριμένη εφαρμογή προσέφερε υπηρεσίες εύρεσης συντεταγμένων θέσης. Οι υπηρεσίες αυτές συνήθως υποστηρίζονταν από εφαρμογές που μέχρι τότε ο χρήστης έπρεπε να πληρώσει είτε σε μορφή αγοράς, είτε σε μορφή μηνιαίου αντιτίμου. Καταλαβαίνει κανείς το πόσο επαναστατική ήταν μια τέτοια δωρεάν υπηρεσία. Τη συγκεκριμένη υπηρεσία αξιοποιήσαμε κι εμείς για την εκπόνηση της συγκεκριμένης πτυχιακής εργασίας. Άλλες καινοτομίες της Eclai*r* ήταν η ευκολότερη αλληλεπίδραση με τα social media, κάτι που δημιουργούσε και δημιουργεί διάφορα θέματα ηθικής και προσωπικών δεδομένων, δεν είναι όμως θέμα της παρούσης εργασίας να αναπτυχθούν, καλύτερα υποστηριζόμενο multi-touch στην οθόνη αφής, που όμως παρέμενε σε μη ικανοποιητικά

επίπεδα. Σημαντική ακόμη, η προσθήκη λογισμικού speech-to-text (ομιλίας- πληκτρολόγησης) και τέλος πολλές και συνεχόμενες καινοτομίες στον τομέα του γραφικού περιβάλλοντος και της διεπαφής του λειτουργικού με τους χρήστες. Με την έκδοση του Android 2.1 -μετά την έκδοση 2.0 βγήκαν δύο ακόμη εκδόσεις οι 2.01 και 2.1 με το ίδιο όμως κωδικό όνομα, οπότε τις θεωρούμε πανομοιότυπες και δεν κάνουμε ξεχωριστή αναφορά- η Google θέλησε να επιδείξει τις δυνατότητες του λειτουργικού της χωρίς παρεμβάσεις από συνεργαζόμενες εταιρείες και άλλους φορείς του εμπορίου. Έτσι, αποφάσισε, σε συνεργασία με την HTC να κυκλοφορήσει στην αγορά την πρώτη κινητή συσκευή, όπου το Android θα παρουσιαζόταν στο κοινό όπως ακριβώς είχε σχεδιαστεί από την ίδια τη Google. Στις 16 Μαρτίου 2010, το *Nexus One* πρωτοκυκλοφόρησε στον Καναδά και σιγά- σιγά σε ολόκληρο τον κόσμο.

Στις 20 Μαΐου 2010 η έκδοση 2.2 ***(Froyo***) κυκλοφόρησε με ακόμη περισσότερες αλλαγές στο γραφικό περιβάλλον του συστήματος και τον τρόπο χρήσης του. Εγκαινιάστηκε επίσης η δυνατότητα κλειδώματος της συσκευής, χωρίς τη χρήση της καθιερωμένης οθόνης, αλλά με τη χρήση κωδικών PIN. Με την κίνηση αυτή, όπως και άλλες συγκεκριμένες βελτιώσεις, η Google φάνηκε να κάνει την πρώτη της απόπειρα να πιάσει την αγορά συσκευών εταιρικών χρηστών, όπου η ασφάλεια είναι πρωταρχικό μέλημα. Η μετάβαση αυτή είναι αρκετά δύσκολη, λόγω της μεγάλης απήχησης των συσκευών της BlackBerry στον τομέα αυτό. Θα ήταν παράλειψη να μην αναφέρουμε ένα ακόμη μεγάλο θέμα που έγινε εμφανές με την έκδοση του Froyo. Η συσκευή που μόλις πριν λίγο είχε κυκλοφορήσει στο εμπόριο από την Google και την HTC μπορούσε να δεχτεί την καινούρια έκδοση, άμεσα μετά την κυκλοφορία της, ενώ οι υπόλοιποι κατασκευαστές έπρεπε να μεριμνήσουν και να βγάλουν δικές τους, ειδικά παραμετροποιημένες, αναβαθμίσεις για τα συστήματά τους.

Ύστερα από αλλεπάλληλες εκδόσεις, που διαδέχονταν η μία την άλλη σε διάστημα μερικών εβδομάδων, κυκλοφόρησε με αρκετή καθυστέρηση, η έκδοση που αποτέλεσε μια από τις πιο σταθερές και δημοφιλείς πρώτες εκδόσεις του λειτουργικού. Ο λόγος για την έκδοση 2.3 (*Gingerbread*). Η έκδοση αυτή συνδυάστηκε με την κυκλοφορία του *Nexus S*, της δεύτερης συσκευής που η Google παρουσίασε στην αγορά ως τη δική της πρόταση για μια κινητή συσκευή Android. Στην έκδοση GingerBread υπήρξαν περισσότερες γραφικές βελτιώσεις, καλύτερη υλοποίηση των λειτουργιών αντιγραφής κι επικόλλησης, καθώς και βελτιωμένο πληκτρολόγιο οθόνης. Σημαντική ήταν επίσης η προσθήκη εφαρμογής για την παρακολούθηση της χρήσης της μπαταρίας, από τις διάφορες εφαρμογές που εκτελούνταν, κάτι που κρίθηκε απαραίτητο λόγω της μεγάλης αυτονομίας που επέτρεπε το σύστημα σε αυτές, με αποτέλεσμα τη γρήγορη εξάντληση της μπαταρίας.

Η συνεχιζόμενη επέλαση της Google προς τις κινητές συσκευές έκανε ένα μικρό διάλειμμα με την κυκλοφορία της έκδοσης 3.0 του Android. Για πρώτη φορά από την κυκλοφορία του λειτουργικού, μια έκδοση δεν προοριζόταν για κινητές συσκευές τηλεφωνίας. Η έκδοση 3.0 *(Honeycomb*) βγήκε στις 22 Φεβρουαρίου του 2011 και υποστήριζε μόνο συσκευές *tablet*. Η πρώτη συσκευή που περιελάβανε τη συγκεκριμένη έκδοση ήταν το Xoom, που δημιουργήθηκε από τη Motorola για λογαριασμό του λειτουργικού της Google. Οι διαφορές της έκδοσης 3.0 από τις προηγούμενες ήταν εξαιρετικά μεγάλες και οι αλλαγές σε επίπεδο γραφικού περιβάλλοντος ήταν δραστικές. Από τη συσκευή αυτή έλειπαν τελείως τα πλήκτρα και όλες οι λειτουργίες διενεργούνταν, για πρώτη φορά, αποκλειστικά μέσω της οθόνης αφής. Επίσης, η κλασική πράσινη απόχρωση του Android αντικαταστάθηκε με ένα έντονο μπλε περιβάλλον.

Αποτέλεσμα του πειραματισμού της ομάδας ανάπτυξης του Android με τις εκδόσεις 3.0, 3.1 και 3.2 αποκλειστικά για tablets, ήταν η νεότερη έκδοση, ξανά για συσκευές *smart-phone*, 4.0 (*Ice Cream Sandwich*). Αν και η νέα έκδοση έμοιαζε περισσότερο με τις παλαιότερες αντίστοιχες για κινητές συσκευές, υιοθέτησε πολλές αλλαγές στο γραφικό περιβάλλον καθώς και τις μπλε αποχρώσεις που επικρατούσαν πια, σε αντίθεση με την πράσινη αίσθηση που είχαν οι προηγούμενες. Μια τεχνολογική καινοτομία άξια λόγου, είναι το *Android Beam*, όπου διαφορετικές συσκευές μπορούν να ανταλλάσσουν δεδομένα απλά ακουμπώντας η μία την άλλη. Το σημαντικό είναι η παραμετροποίηση που μπορούν να επιτύχουν οι σχεδιαστές λογισμικού, καθώς η τεχνολογία αυτή μπορεί να αναπτυχθεί σύμφωνα με τη δημιουργικότητα του κάθε προγραμματιστή.

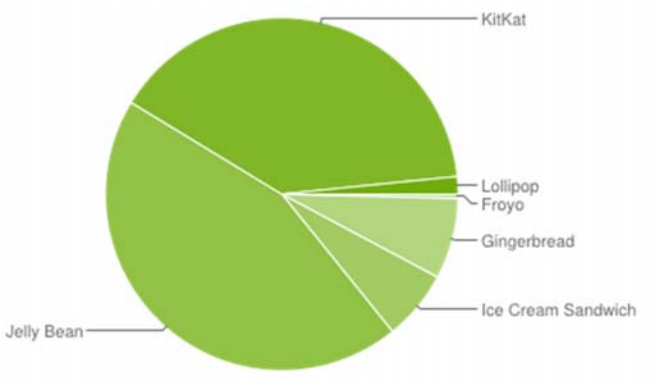
Μια άλλη ενδιαφέρουσα καινοτομία είναι αυτή του ξεκλειδώματος της συσκευής μέσω αναγνώρισης προσώπου. Όχι ιδιαίτερα ασφαλής, αλλά εξαιρετικά ενδιαφέρουσα σε επίπεδο τεχνολογίας.

Ακολούθησαν οι εκδόσεις 4.1 και 4.2 (*Jelly Bean*) που κυκλοφόρησαν στις 27 Ιουνίου και 29 Οκτωβρίου 2012, καθώς και η 4.3 τον Ιούλιο του 2013 οι οποίες αναβάθμισαν το Android ακόμη περισσότερο και πιο συγκεκριμένα, κατάφεραν να κλείσουν την ψαλίδα του λειτουργικού, με το μεγάλο αντίπαλο της αγοράς, το IOS της Apple, όσον αφορά την ανταπόκριση του γραφικού περιβάλλοντος και την ομαλότητα κίνησης των διαφόρων στοιχείων στις οθόνες των κινητών συσκευών. Επιπλέον βελτιώνονται οι τεχνολογίες και υπηρεσίες δικτύου, προσφέροντας πρωτότυπους χειρισμούς για την επίτευξη διαφόρων λειτουργιών, όπως για παράδειγμα, προβολή εικόνων από την κινητή συσκευή στην οθόνη της τηλεόρασης μέσω wifi. Αξιοσημείωτη είναι επίσης και η δυνατότητα ύπαρξης πολλαπλών χρηστών σε συσκευές tablets που πρωτοεμφάνισε η έκδοση 4.2.

Δύο μήνες μετά την 4.3 και συγκεκριμένα τον Σεπτέμβριο του 2013 η Google έθεσε στη διάθεση των χρηστών την 4.4 (*Key Lime Pie*), η οποία αργότερα μετονομάστηκε σε (*KitKat*), βασικό χαρακτηριστικό της οποίας ήταν η δυνατότητα να εκτελείται σε μεγαλύτερη γκάμα συσκευών αφού οι ελάχιστες απαιτήσεις της σε μνήμη RAM ήταν στα 340MB. Ακολούθησαν άλλες τέσσερεις εκδόσεις που διόρθωσαν προβλήματα ασφάλειας κυρίως αλλά και άλλες βελτιώσεις μέχρι τον Ιούνιο του 2014. Τότε κυκλοφόρησε η 4.4 Wearable η οποία είχε εξοπλιστεί με πρόσθετα για φορητές συσκευές όπως τα έξυπνα ρολόγια χειρός.

Η σύντομη ιστορική μας αναδρομή ολοκληρώνεται στην έκδοση 5.0 (*Lollipop*) η οποία τέθηκε σε κυκλοφορία στις 12 Νοεμβρίου του 2014 αρχικά για συγκεκριμένες συσκευές όπως η Nexus και αργότερα, το Νοέμβριο του ίδιου χρόνου για το υπόλοιπο κοινό. Βασικά χαρακτηριστικά της νέας έκδοσης η αντικατάσταση της Dalvik μηχανής με την Android Runtime με συνέπεια τη βελτίωση της συνολικής απόδοσης του συστήματος καθώς επίσης και άλλες αλλαγές που στοχεύουν στην αποδοτικότερη χρήση της μπαταρίας.

Είναι περισσότερο από προφανές πως η επέλαση της Google και της Open Handset Alliance στον τομέα της ανάπτυξης υλικού και λογισμικού για κινητές συσκευές έχει φέρει μια πρώτου μεγέθους επανάσταση στο χώρο. Οι προσπάθειες τους έχουν στεφθεί μέχρι στιγμής με επιτυχία και είναι ενδιαφέρον να δούμε το πλαίσιο μέσα στο οποίο θα διαμορφωθούν οι νέες εκδόσεις του λειτουργικού στο μέλλον. Ειδικά αν αναλογιστούμε πως έρχονται στο προσκήνιο καινούριες προσπάθειες από εταιρείες, όπως η Microsoft και η Nokia, ενώ δεν πρέπει να ξεχνάμε και την Apple με το *iOS*. Οι προοπτικές και οι δυνατότητες που προσφέρει η τεχνολογία επικοινωνιών, δημιουργούν πρωτόγνωρες συνθήκες και μένει να προσδιοριστεί πως αυτές θα επηρεάσουν τις ζωές εκατομμυρίων ανθρώπων στον πλανήτη.



Εικόνα 1: Ποσοστά χρήσης των διαφόρων εκδόσεων του Android το Φεβρουάριο του 2015.

πηγή: http://developer.android.com/about/dashboards/index.html

2.3 Δυνατότητες ανάπτυξης με τη χρήση Android

2.3.1 Τι μπορεί να κάνει

Μέσα στα λίγα χρόνια που πέρασαν από τη δημοσιοποίηση της πρώτης έκδοσης, το Android μεγάλωσε, αναπτύχθηκε κι εμπλουτίστηκε με νέες δυνατότητες, εφαρμογές, υπηρεσίες και υποστηριζόμενες τεχνολογίες, κάποιες προ υπάρχουσες και κάποιες ειδικά υλοποιημένες από τη Google για το λειτουργικό της. Αναφορικά, κάποιες από τις τεχνολογίες αυτές περιλαμβάνουν ασύρματη σύνδεση δικτύου (Wi-Fi), μεταφορά δεδομένων μέσω υπηρεσιών κινητής τηλεφωνίας (3G, edge), σύνδεση με συσκευές σε κοντινές αποστάσεις (blue-tooth), αλλά και την πρωτοεμφανιζόμενη τεχνολογία NFC (Near Field Communications), που χρησιμοποιείται για τη σύνδεση με συσκευές σε κοντινές αποστάσεις, με διαφορετικές απ' ότι μέχρι τώρα δυνατότητες. Να σημειωθεί ότι η συγκεκριμένη τεχνολογία απαιτεί ειδικό υλικό εντός της συσκευής, επομένως δεν υποστηρίζεται από όλα τα συστήματα που τρέχουν Android. Ακόμη, υποστηρίζεται μια ευρεία γκάμα κωδικοποιημένων βίντεο και ήχων για αναπαραγωγή, πράγμα όχι αμελητέο για μια κινητή συσκευή. Η αναγνώριση ομιλίας είναι επίσης μια τεχνολογία που υποστηρίχθηκε από τις πρώτες εκδόσεις του Android και συνεχίζει να βελτιώνεται. Στις σημερινές συσκευές, ο χρήστης έχει τη δυνατότητα όχι μόνο να κάνει αναζήτηση επαφών μέσω ομιλίας, αλλά επίσης να γράφει κείμενα, καθώς και να παίρνει δεδομένα και πληροφορίες μέσω ειδικών αλγορίθμων που παρέχουν τη δυνατότητα αναγνώρισης φωνητικών εντολών. Τέλος, δεν θα μπορούσαμε να μην αναφερθούμε στις τεχνολογίες εντοπισμού θέσης που παρέχουν οι κινητές συσκευές Android. Τέτοιες υπηρεσίες θεωρούνται πλέον τεχνολογικό στάνταρ και περιλαμβάνονται σχεδόν σε όλες τις νέες συσκευές. Η δυνατότητα ανάλυσης σήματος GPS, συνδυασμένη με την βοηθητική αναγνώριση θέσης μέσω των δικτυακών εγκαταστάσεων των διαφόρων εταιρειών, κι όχι μόνο, προσφέρουν μεγάλη ακρίβεια προσδιορισμού της θέσης του χρήστη, κάτι που μπορεί να χρησιμοποιηθεί για εφαρμογές λήψης οδικών οδηγιών, συμπληρωματικές υπηρεσίες καθώς και σε περιπτώσεις έκτακτης ανάγκης, με πολύ ικανοποιητικά αποτελέσματα. Ο τρόπος χρησιμοποίησης των κινητών συσκευών που προαναφέρθηκε ήταν άλλωστε και το όραμα του συνιδρυτή και θεωρούμενου ως πατέρα του Android, *Andy Rubin*.

2.4 Ανταγωνιστικές Λύσεις

2.4.1 Τι άλλο υπάρχει

Όταν η Open Handset Alliance ανακοίνωσε την κυκλοφορία του Android, το νέο λειτουργικό ήρθε να βρει μια πλειάδα ήδη διαθέσιμων επιλογών για τους επαγγελματίες αλλά και τους χρήστες. Ο όρος smart-phone δεν ήταν ακόμη διαδεδομένος και όλες οι εταιρείες λογισμικού και τεχνολογίας κινητών συσκευών κονταροχτυπιόντουσαν για την κυριαρχία στην αγορά αυτή. Υπήρχαν παραδοσιακές δυνάμεις του χώρου που έκαναν τις προτάσεις τους στην αγορά σχετικά με μελλοντικά βήματα στις κινητές συσκευές, υπήρχαν νέες προσπάθειες από εταιρείες με μικρή ή μηδενική σχέση με το συγκεκριμένο αντικείμενο και υπήρχαν οι ανεξάρτητες προσπάθειες οργανισμών. Κάποιες έπιασαν, κάποιες απέτυχαν και άλλες έμειναν στην αφάνεια καθόλη την πορεία τους. Είναι πολύ ενδαφέρον το γεγονός όμως, πως παρά το μεγάλο πλήθος επιλογών, το μερίδιο αγοράς στη κατηγορία αυτή της τεχνολογίας είναι αρκετά πολωμένο. Έτσι βλέπουμε πως, σύμφωνα με πολλές και διαφορετικές στατιστικές πηγές (βλ. πηγές εργασίας), το iOS και το Android κυριαρχούν σαν μερίδια αγοράς με ένα εντυπωσιακό ποσοστό που φτάνει το 80%. Οι τάσεις της τεχνολογίας είναι γνωστό πως

μπορούν να αλλάξουν ραγδαία, ειδικά τώρα που ένας ακόμη μεγάλος παίχτης, η Microsoft, εισέρχεται στο χώρο αυτό πιο δυναμικά με την κυκλοφορία λύσεων λογισμικού, αλλά και προϊόντα φτιαγμένα από την ίδια με προορισμό τους τελικούς καταναλωτές. Παρά το γεγονός της αστάθειας του πεδίου αυτού όμως, είναι σημαντικό για έναν επαγγελματία να επιλέγει λύσεις οι οποίες θα είναι βιώσιμες και ανταποδοτικές. Έτσι, η ανάπτυξη εφαρμογών για μια πλατφόρμα που παρέχει πολλά και καλά εργαλεία λογισμικού, αλλά δεν έχει την απήχηση που χρειάζεται από την αγορά, είναι ένα δίκοπο μαχαίρι και μπορεί να οδηγήσει σε αξεπέραστα εμπόδια, ακόμη και αξιόλογες εφαρμογές, τα οποία δε μπορούν να λυθούν απλά και μόνο με τον Debugger. Ας δούμε όμως πιο αναλυτικά πως έχει διαμορφωθεί το πεδίο των τεχνολογιών κινητών επικοινωνιών σήμερα.

2.4.2 Bada (Samsung)

Η Samsung παρά το γεγονός ότι έχει βγάλει μια πολύ μεγάλη γκάμα συσκευών που λειτουργούν με Android, σχεδιάζει το επόμενο ανεξάρτητο βήμα της. Το συγκεκριμένο λειτουργικό έχει κυκλοφορήσει από το 2010 και διατίθεται σε κάποιες συσκευές της εταιρείας. Το Φεβρουάριο του 2013 η εταιρία ανακοίνωσε την αντικατάσταση του από το Tizen το οποίο απευθύνεται σε μεγαλύτερο εύρος συσκευών και όχι μόνο έξυπνα τηλέφωνα και tablet. Ακόμη, η Samsung δημιούργησε ένα δικτυακό κατάστημα για την εγκατάσταση εφαρμογών (Apps) για το λειτουργικό της.

2.4.3 BlackBerry OS (RIM)

Η εταιρεία δραστηριοποιείται στην κατασκευή συσκευών για πολύ συγκεκριμένο κοινό. Οι συσκευές της είναι απλές στη λειτουργία τους, με περιορισμένες επιλογές στο χρήστη και προορισμένες για εταιρικούς χρήστες, αφού παρέχει μεγάλη αξιοπιστία και ασφάλεια στις υπηρεσίες επικοινωνίας που προσφέρει. Τελευταία, η πολιτική αυτή έχει αλλάξει και η BlackBerry τείνει να προωθήσει τα προϊόντα της και σε απλούς χρήστες. Αξίζει να σημειωθεί πως η μεγάλη κυκλοφορία των συσκευών της στον κόσμο των εταιρειών επλήγη σημαντικά με την κυκλοφορία του IPhone της Apple.

2.4.4 iOS (Apple)

Το λειτουργικό αυτό της Apple, είναι μια έκδοση του λειτουργικού που η εταιρεία χρησιμοποιεί για τους μεγάλους υπολογιστές της (MAC OSX), προσαρμοσμένο στις δυνατότητες του IPhone. Τελευταία, και μετά την κυκλοφορία συσκευών αφής μεγαλύτερου μεγέθους (iPad), το λειτουργικό αυτό πήρε απλά το όνομα iOS, ενώ πριν αναφερόταν απλά ως iPhone OS. Αρχικά είχε σχεδιαστεί ώστε να δέχεται επίσημες εφαρμογές, από την Apple, αλλά σύντομα άνοιξε τη λειτουργικότητά του και σε εφαρμογές τρίτων κατασκευαστών. Το iOS και το iPhone είναι ο μεγάλος αντίπαλος του Android όσον αφορά τη διανομή της πίτας αγοράς smartphone. Είναι διαφορετικής φιλοσοφίας από το Android και διαφοροποιούνται σε αρκετά σημεία σε επίπεδο χρήσης αλλά και ανάπτυξης.

2.4.5 Windows Phone (Microsoft)

Η πρόταση της Microsoft για την αγορά των “έξυπνων” κινητών συσκευών, ήρθε κάπως καθυστερημένα, ενώ μέχρι τώρα δε φαίνεται να διεκδικεί μεγάλο μερίδιο της αγοράς. Μένει να δούμε εάν τα δεδομένα αυτά θα αλλάξουν στο μέλλον. Η Microsoft κυκλοφόρησε το νέο της λειτουργικό για πρώτη φορά το Φεβρουάριο του 2010. Αυτό διατηρεί την ίδια φιλοσοφία των δύο μεγάλων πρωταγωνιστών της κατηγορίας, όπου οι όποιες λειτουργίες γίνονται απλά κι εύκολα με τη χρήση των δακτύλων του χρήστη. Έτσι έχουμε μεγάλα τετράγωνα εικονίδια, που μπορούν να χειριστούν τις λειτουργίες που προσφέρει η συσκευή. Αυτό έρχεται να διορθώσει τη λογική των προηγούμενων λειτουργικών της Microsoft για κινητές συσκευές, όπου η χρήση βοηθητικού στυλό, ήταν απαραίτητη για τη λειτουργία τους.

2.5 Αδειοδότηση ανάπτυξης, χρήσης και διανομής εφαρμογών μέσω Android (Apache License)

2.5.1 Android, ανοιχτού κώδικα;

Το project Android είναι όντως ένα project ανοιχτού κώδικά. Αυτό πρακτικά σημαίνει πως οποιοσδήποτε παράγοντας μπορεί να δει, αλλάξει, διαγράψει κώδικα και να αλλάξει τη λειτουργικότητά του όπως εξυπηρετεί τον ίδιο. Ο τρόπος και οι όροι με τους οποίους μπορεί κάποιος να εκμεταλλευτεί το συγκεκριμένο project υπόκεινται και περιγράφονται αναλυτικά από την άδεια χρήσης του ιδρύματος Apache[1] . Περιληπτικά, η έκθεση αδειοδότησης που έχει δημιουργήσει το ίδρυμα Apache, αναφέρει πως τα έργα που δημιουργούνται υπό την ισχύ της, είναι έργα που οφείλουν να δημιουργούνται και να διανέμονται δωρεάν, χωρίς κόστος διαμοιρασμού και με σαφή αναφορά στην αδειοδότηση στην οποία υπόκεινται. Υπάρχει η δυνατότητα προσφοράς υπηρεσιών υποστήριξης και χρέωσης αυτών, όμως οποιαδήποτε διανομή κώδικα ή ακόμη και εκτελέσιμου λογισμικού, οφείλει (εφόσον βρίσκεται υπό την αδειοδότηση) να αναφέρει αναλυτικά τις αλλαγές που έχουν επιτελεστεί επί του έργου. Σε διαφορετική περίπτωση η άδεια του έργου ανακαλείται αμέσως. Ακόμη, αξιοσημείωτο είναι πως καμία πλευρά που έχει προσφέρει σε κάποιο έργο, δε θεωρείται υπεύθυνη για σφάλματα κώδικα που μπορεί να αποδειχθούν βλαβερά για τα συστήματα στα οποία λειτουργούν ή ακόμη και για απώλεια δεδομένων λόγω βλαβών. Εξαιρούνται φυσικά φθορές που έγιναν εσκεμμένα και με δόλο. Παρά ταύτα, δίνεται η δυνατότητα σε φορείς να προσφέρουν εγγύηση λειτουργίας κάποιου έργου, αφού όμως πρώτα έχουν αποδεχτεί την απαλοιφή οποιασδήποτε ευθύνης τρίτων φορέων που έχουν αναμειχθεί στο έργο.

Αναλύοντας και αντιλαμβανόμενοι το περιεχόμενο και το πνεύμα του κειμένου της αδειοδότησης του ιδρύματος Apache, συμπεραίνουμε πως σκοπός της είναι να προωθήσει έργα λογισμικού βασισμένα στην κοινή εμπειρία και επεξεργασία διαφορετικών φορέων. Δίνει τη δυνατότητα σε εταιρείες και ιδιώτες να δουλέψουν πάνω σε κάποιο project, χωρίς το φόβο των νομικών επιπλοκών που μπορεί να εμπεριέχει μια εμπορική εφαρμογή, ενώ παράλληλα δεν αφαιρεί το δικαίωμα των φορέων που εμπλέκονται στο εν λόγω έργο, να αποκομίσουν κέρδη, οικονομικά και άλλα, από το αυτό. Αν αυτό το μεταφέρουμε στην περίπτωση του Android, μπορούμε να δούμε τη φιλοσοφία με την οποία η Google παρέχει τον κώδικα και τα εργαλεία ώστε τρίτοι φορείς να δημιουργήσουν εφαρμογές σε αυτόν.

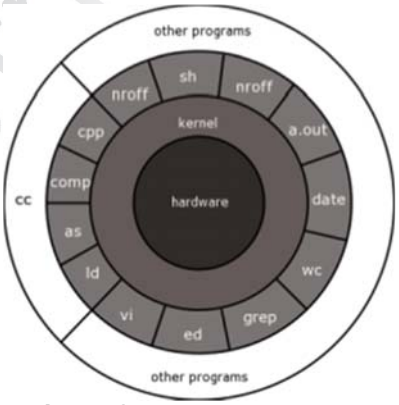
Εδώ όμως πρέπει να ξεκαθαρίσουμε πως η αδειοδότηση αυτή αφορά το έργο Android καθ' εαυτό. Δεν αναφέρεται και δεν περιορίζει τις εφαρμογές τρίτων που προορίζονται για δωρεάν ή επί πληρωμή διαμοιρασμό μέσω της διαδικτυακής πλατφόρμας εφαρμογών του Android (*Google Play*).

Για τη συγκεκριμένη περίπτωση, η εταιρεία έχει δημιουργήσει τρόπους διασφάλισης της αποτροπής, χωρίς κατάλληλη εξουσιοδότηση, εγκατάστασης εφαρμογών τρίτων δημιουργών. Επομένως, ενώ η πλατφόρμα του Android είναι ανοιχτού κώδικά και διαμοιράζεται ελεύθερα σε οποιοδήποτε μέρος έχει ενδιαφέρον, τρίτοι φορείς δεν υποχρεούνται να αναπτύξουν έργα με το ίδιο καθεστώς διαμοιρασμού. Αυτό είναι πολύ σημαντικό, αν αναλογιστεί κανείς την απήχηση που έχει τον τελευταίο καιρό το Android και το πλήθος των εφαρμογών που χρησιμοποιούνται καθημερινά από τις εν λόγω κινητές συσκευές.

2.6 Γενικά Τεχνικά χαρακτηριστικά

2.6.1 Που βασίζεται το Android.

Η ανάπτυξη του Android συντελείται από την Google και ανακοινώνεται συνήθως μετά την κυκλοφορία κάθε έκδοσης. Βασικό του συστατικό είναι ένας πυρήνας *Linux kernel 2.6* για τις εκδόσεις Android πριν την 4.0 και του *Linux kernel 3.X* για τις εκδόσεις Android από την 4.0 και μετά. Οι βιβλιοθήκες, οι *διεπαφές* (*interfaces*) και το *ενδιάμεσο λογισμικό* (*middleware*) είναι γραμμένα σε C, ενώ η ανάπτυξη των εφαρμογών γίνεται με τη χρήση βιβλιοθηκών και της γλώσσας Java. Βασική αρχιτεκτονική για την οποία έχει σχεδιαστεί το Android είναι η *ARM* (επεξεργαστές τύπου *RISC*) αλλά υπάρχουν ομάδες που έχουν δημιουργήσει έργα του Android βασισμένα σε αρχιτεκτονική *x86*. Αξίζει να σημειωθεί πως η Google έχει κάνει επεμβάσεις πάνω στον πυρήνα που χρησιμοποιεί, χωρίς τυπική έγκριση, κάτι που έχει δημιουργήσει τριβές και αντιδράσεις από την κοινότητα ανάπτυξης του Linux. Γίνονται προσπάθειες για συμψηφισμό των αλλαγών αυτών σε μελλοντικές εκδόσεις, όμως ο χρόνος της επίτευξης των στόχων αναμένεται να είναι αρκετά μεγάλος. Λόγω των αλλαγών αυτών, έχει καταστεί δυσχερής η αυτούσια μεταφορά λογισμικού από άλλες πλατφόρμες Linux στο Android. Η τακτική αυτή της Google, να προχωρεί σε αλλαγές στα εργαλεία τα οποία χρησιμοποιεί, έχει προκαλέσει αντιδράσεις από διαφορετικές πλευρές και οργανισμούς. Θα αναφερθούμε πιο αναλυτικά στο κεφάλαιο *Eclipse.*



Εικόνα 2: Ο Linux Kernel.

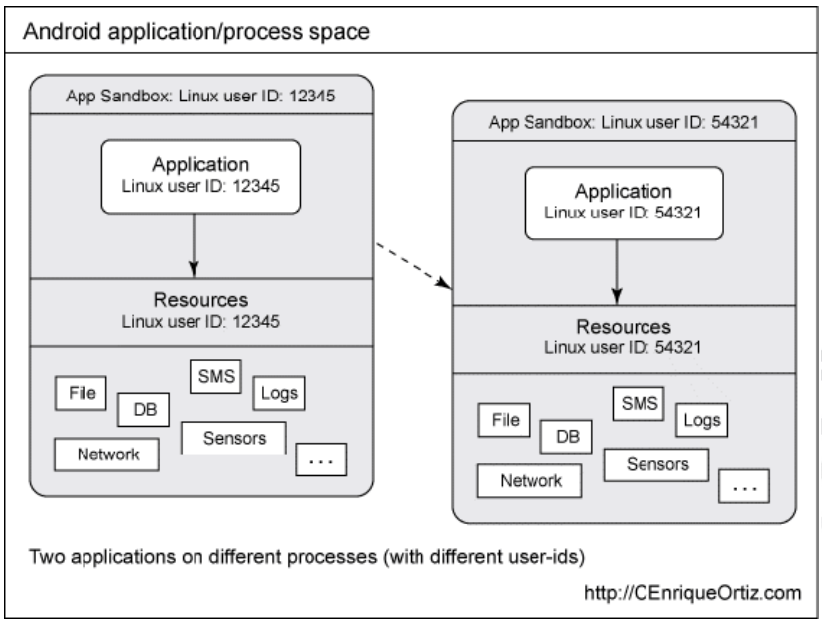
2.6.2 Διαχείριση μνήμης

Εξαιρετικό ενδιαφέρον παρουσιάζει ο τρόπος με τον οποίο γίνεται η διαχείριση των ενεργών εφαρμογών στο σύστημα. Λόγω της φύσης των συσκευών που τρέχουν Android, ο τρόπος με τον οποίο το σύστημα διαχειρίζεται τη μνήμη είναι πρωτευούσης σημασίας. Σε αντιδιαστολή με

τα σταθερά συστήματα που έχουν μια σταθερή ηλεκτρική παροχή ή ακόμη και τους φορητούς υπολογιστές που διαθέτουν μπαταρίες μεγάλης χωρητικότητας, οι κινητές συσκευές, διαθέτουν εξαιρετικά περιορισμένες ηλεκτρικές πηγές. Πολλοί λόγοι συνεισφέρουν σε αυτό, μέγεθος, κόστος υλικών κ.α. Για να αποφευχθούν οι περιορισμοί αυτοί, το σύστημα κατά την ολοκλήρωση της διεργασίας μια εφαρμογής, ή της θέσης αυτής στο περιθώριο την εναποθέτει στη μνήμη, σε μια κατάσταση αδράνειας. Έτσι η εφαρμογή ενώ θεωρητικά “τρέχει”, δεν καταναλώνει πόρους όπως ρεύμα ή υπολογιστική ισχύ. Εάν ο χρήστης θελήσει να χρησιμοποιήσει την εφαρμογή ξανά, το Android την επαναφέρει από την λανθάνουσα κατάστασή της και την ενεργοποιεί. Έτσι επιτυγχάνονται τρία θετικά. Πρώτα υπάρχει άμεση ανταπόκριση της συσκευής στους χειρισμούς του χρήστη, ειδικά σε εφαρμογές που χρησιμοποιούνται συχνά. Δεύτερο, ο χρήστης δε χρειάζεται να απασχολείται με την εκκίνηση και παύση εφαρμογών και τρίτο η κατανάλωση της μπαταρίας της συσκευής κυμαίνεται σε σχετικά χαμηλά επίπεδα. Σε περίπτωση που υπάρχει ανάγκη αποδέσμευσης μνήμης, το σύστημα κλείνει τις εφαρμογές που έχουν μείνει αδρανείς τον περισσότερο χρόνο.

2.6.3 Ασφάλεια

Εύκολα μπορεί κανείς να φανταστεί τους κινδύνους που απειλούν ένα σύστημα το οποίο χειρίζεται μεγάλους όγκους προσωπικών δεδομένων. Ειδικά σε μια εποχή όπου η κινητή συσκευή είναι κάτι περισσότερο από ένα τηλέφωνο, η ασφαλής και χωρίς παρεμβάσεις τρίτων λειτουργία του λειτουργικού μιας τέτοιας συσκευής είναι επίσης πρωτευούσης σημασίας. Ο τρόπος με τον οποίο το Android επιχειρεί να αντιμετωπίσει τις απειλές ασφάλειας είναι με την τεχνική του sandbox. Η τεχνική αυτή παρομοιάζεται με ένα κουτί άμμου για κάθε εφαρμογή που εκτελείται. Περιορισμένη από το υπόλοιπο σύστημα, κάθε εφαρμογή διαθέτει πιστοποιητικά για πρόσβαση σε επιλεγμένους πόρους του συστήματος, κάτι που ο χρήστης ήδη έχει αποδεχτεί κατά την εγκατάσταση. Εάν κάποιος χρήστης επιχειρήσει να εγκαταστήσει μία εφαρμογή στη συσκευή του, θα ενημερωθεί από το διαδικτυακό κατάστημα για τη φύση των αδειών που η απαιτούνται. Εάν είναι σύμφωνος με αυτές, μπορεί να προχωρήσει στην εγκατάσταση, αλλιώς αρνείται και η εγκατάσταση αποτυγχάνει. Τέτοιες άδειες μπορούν να αφορούν πρόσβαση στο διαδίκτυο, πρόσβαση στις επαφές της συσκευής, αποστολή ή ανάγνωση των μηνυμάτων SMS της συσκευής κ.α. Αν και η τεχνική αυτή αποδεικνύεται αρκετά αποτελεσματική, ωστόσο κρούσματα εγκατάστασης βλαβερού κώδικα παρατηρούνται και σε συσκευές Android, κυρίως λόγω της ελλιπούς γνώσης από μέρους των χρηστών της σημασίας των αδειών αυτών στις διάφορες εφαρμογές, με αποτέλεσμα την υποκλοπή ευαίσθητων προσωπικών δεδομένων, την αποστολή μηνυμάτων σε ειδικούς αριθμούς που επιφέρουν υπέρογκες χρεώσεις κ.α, αποδεικνύοντας ξανά και ξανά πως η καλύτερη προστασία ενός συστήματος προέρχεται από την ενημέρωση των χρηστών του.



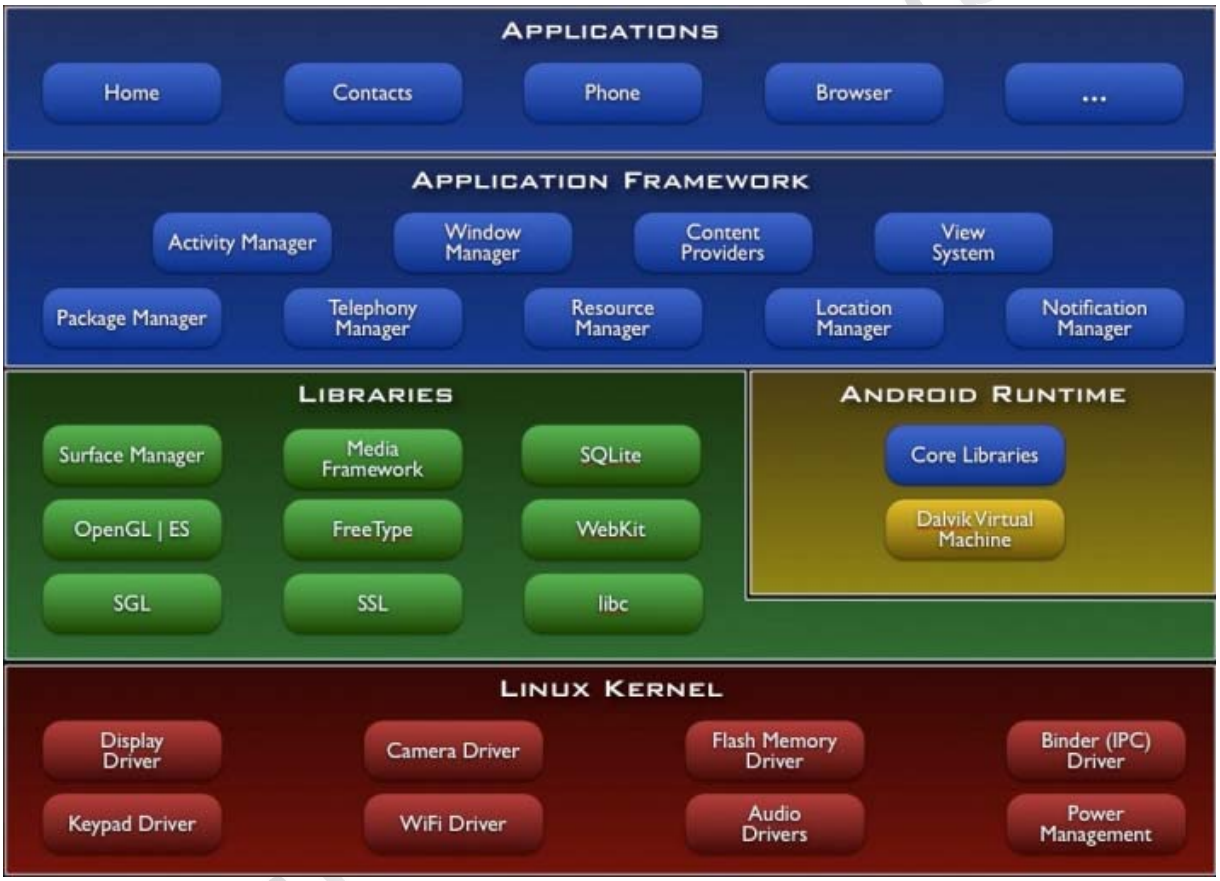
Εικόνα 3: Η τεχνική sandbox.

2.7 Αρχιτεκτονική συστήματος

Το λειτουργικό σύστημα Android αναλύεται σε πέντε βασικά επίπεδα. Ξεκινώντας από το πιο βασικό, το επίπεδο του πυρήνα (Linux Kernel), το επίπεδο των βιβλιοθηκών εγγενή κώδικα (C/C++), το επίπεδο εκτέλεσης των εφαρμογών, το επίπεδο βιβλιοθηκών Java και το επίπεδο των εκτελέσιμων εφαρμογών.

2.7.1 Επίπεδο πυρήνα Linux Kernel

Το επίπεδο αυτό αποτελεί τον πυρήνα του συστήματος. Ο πυρήνας που λειτουργεί σε όλες τις συσκευές Android είναι ο Linux v2.6 ή v3.X. Ο πυρήνας αποτελείται από μια σειρά βιβλιοθηκών που είναι ειδικά χτισμένες ώστε να υποστηρίζουν όλες τις βασικές λειτουργίες της συσκευής, όπως η διαχείριση της οθόνης, η χρήση των δικτυακών συσκευών K.O.K. Η διαχείριση αυτή γίνεται με τη χρήση των *οδηγών συστήματος* (*drivers*). Αυτοί αποτελούν βιβλιοθήκες γραμμένες σε ειδικά



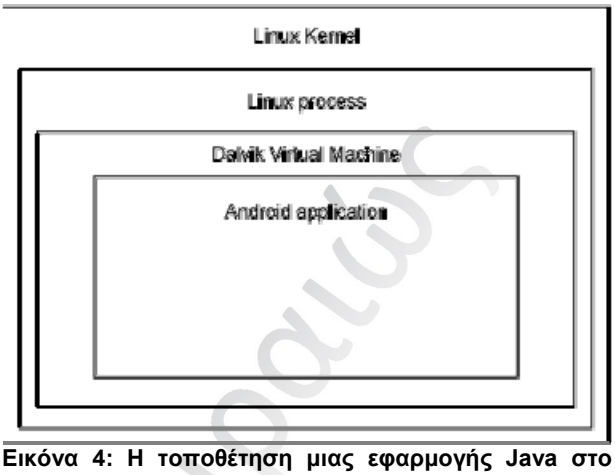
Εικόνα 4: Τα τέσσερα επίπεδα της αρχιτεκτονικής του Android

Πηγή: http://www.android-app-market.com/wp-content/uploads/2012/02/Android-architecture.jpg

διαμορφωμένο κώδικα C, ο οποίος είναι σχεδιασμένος να προσφέρει το δυνατόν χαμηλή κατανάλωση ενέργειας και περιέχουν τις "οδηγίες χρήσης" των πόρων της συσκευής, όπως για παράδειγμα της εσωτερικής μνήμης flash (*Flash Memory Driver*), του Wi-Fi (*Wi-Fi Driver*) κλπ. Ο συνήθης τρόπος διαχείρισης των διαφόρων αιτημάτων του συστήματος γίνεται μέσω του *Surface Manager (Διαχειριστής Επιφάνειας)*. Αυτός λαμβάνει όλα τα αιτήματα από το επίπεδο βιβλιοθηκών και ενεργοποιεί τις ανάλογες διαδικασίες. Εξαίρεση σε αυτόν τον κανόνα αποτελεί το *Media Framework (Πλαίσιο Πολυμέσων)*. Η συγκεκριμένη βιβλιοθήκη αναλαμβάνει την αναπαραγωγή ήχου και βίντεο και βρίσκεται στο επίπεδο πυρήνα λόγω της μεγάλης βελτιστοποίησης που χρειάζονται οι οδηγοί για χρήση σε κάθε συσκευή που λειτουργεί με το σύστημα.

2.7.2 Επίπεδο εκτέλεσης Εφαρμογών

Το επίπεδο αυτό αποτελείται από την εικονική μηχανή *Dalvik* και τις βασικές βιβλιοθήκες της Java. Η εικονική μηχανή με τη συνδρομή των βιβλιοθηκών αυτών, αναλαμβάνει την εκτέλεση των εφαρμογών που είναι γραμμένες σε Java. Καταλαβαίνουμε λοιπόν πως η Dalvik δεν είναι τίποτα άλλο από μια ειδικά τροποποιημένη *Java Virtual Machine* η οποία όμως έχει ως σκοπό την εκτέλεση σε συσκευές με περιορισμένη υπολογιστική ισχύ. Η λειτουργία της *Dalvik* είναι να εκτελεί τον κώδικα εφαρμογών γραμμένων σε Java συνδέοντας τις διάφορες κλήσεις υλικού που γίνονται από την εφαρμογή με το επίπεδο βιβλιοθηκών εγγενή κώδικα και το επίπεδο του πυρήνα. Η μηχανή αυτή στην έκδοση 5.0 αντικαθίσταται από την Android Runtime βασικό χαρακτηριστικό την βελτιωμένη απόδοση του συστήματος. Για να το πετύχει αυτό η νέα μηχανή μεταφράζει το κώδικα μιας εφαρμογής κατά την εγκατάσταση αυτής και όχι κάθε φορά που αυτή εκτελείται, όπως έκανε η Dalvik. Με τον τρόπο αυτό η λειτουργία του επεξεργαστή αποσυμφορείται και βελτιώνεται η απόδοση της μπαταρίας. Επίσης η νέα μηχανή παρέχει νέες λειτουργίες όπως λειτουργίες αποσφαλμάτωσης, βελτιωμένο 'συλλέκτη σκουπιδιών' (garbage collector) και άλλα.



Εικόνα 4: Η τοποθέτηση μιας εφαρμογής Java στο Android.

2.7.3 Επίπεδο βιβλιοθηκών εγγενή κώδικα (C/C++)/ Επίπεδο βιβλιοθηκών Java

Υπάρχουν δύο ειδών βιβλιοθήκες στο σύστημα του Android, αυτές του συστήματος, οι οποίες είναι γραμμένες σε C ή C++, και αυτές των εφαρμογών που είναι γραμμένες σε Java java και εκτελούνται στην εικονική μηχανή.

Επίπεδο Εφαρμογών

Το επίπεδο εφαρμογών, το οποίο είναι γραμμένο σε Java αποτελείται από τις εφαρμογές του συστήματος καθώς και τις εφαρμογές που εγκαθιστά ο χρήστης σε αυτό. Αποτελεί το ανώτερο επίπεδο στην ιεραρχία. Στο επίπεδο αυτό όλες οι εφαρμογές έχουν την ίδια βαρύτητα και εκτελούνται παράλληλα. Παραδείγματα τέτοιων εφαρμογών είναι το τηλέφωνο, οι τηλεφωνικές επαφές, τα μηνύματα κ.α. Το λειτουργικό έχει έναν ιδιαίτερο τρόπο να διαχειρίζεται τις εφαρμογές του. Κάθε εφαρμογή αποτελείται από τις λεγόμενες δραστηριότητες (activities) οι οποίες είναι στην ουσία οι διαφορετικές "οθόνες" που εμφανίζει η εφαρμογή. Έτσι ο χρήστης έχει τη δυνατότητα να εμφανίζει μόνο μία "οθόνη" στο monitor της συσκευής του κάθε φορά, είτε αυτή είναι της ίδιας ή κάποιας διαφορετικής εφαρμογής. Η διαδικασία εκκίνησης ενός activity είναι δαπανηρή σε πόρους συστήματος. Αυτό συμβαίνει γιατί κάθε φορά που εμφανίζεται ένα activity στην οθόνη της συσκευής, υπάρχει μια ολόκληρη διαδικασία που ακολουθείται με λειτουργίες αρχικοποίησης μνήμης, αποστολής αιτημάτων για χρήση συσκευών υλικού κ.α. Στο Android η εναλλαγή διαφορετικών activities είναι ιδιαίτερα συχνή και γίνεται με τη χρήση των intents. Καταλαβαίνει κανείς πως η διαχείριση

Εικόνα 5: Εφαρμογές Android

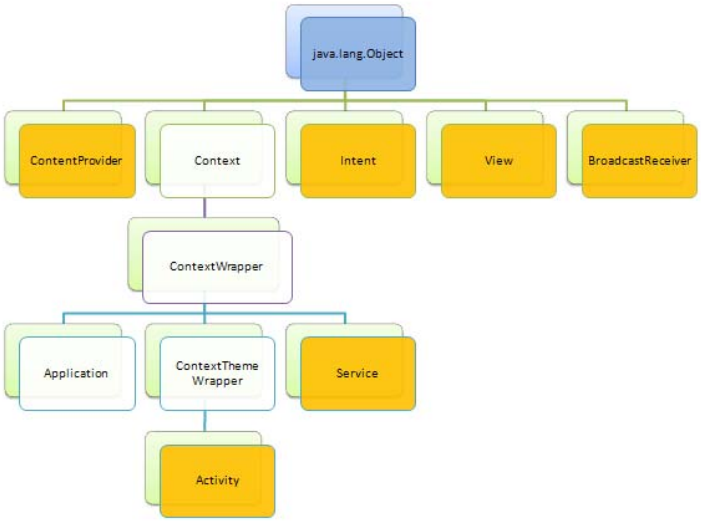
των διαφόρων εφαρμογών θα απόβαινε προβληματική, εάν αυτές συμπεριφέρονταν όπως στα σταθερά υπολογιστικά συστήματα που γνωρίζουμε μέχρι σήμερα. Για το λόγο αυτό όταν ο χρήστης εγκαταλείπει ένα activity αυτό δεν κλείνει άμεσα. Παραμένει σε μια κατάσταση αδράνειας ώστε αν ο χρήστης το χρειαστεί σε μικρό χρονικό διάστημα το λειτουργικό να μπορεί να το επαναφέρει άμεσα χωρίς να σπαταλά πόρους για να το επανεκκινήσει.

Την όλη αυτή διαδικασία εκκίνησης, παύσης, επανεκκίνησης και καταστροφής των activities την διαχειρίζεται ο *Activity Manager (Διαχειριστής Δραστηριοτήτων)*. Έτσι, όταν ο χρήστης ζητήσει ένα activity, ο Activity Manager αναλαμβάνει τις διαδικασίες εκκίνησής του. Σε περίπτωση που γίνει εμφάνιση ενός διαφορετικού activity, ο Activity Manager τοποθετεί το πρώτο σε περιοχή αποθήκευσης έως ότου είτε ο χρήστης να το ξαναζητήσει, είτε με χρήση εσωτερικών κριτηρίων να γίνει τερματισμός ώστε να απελευθερωθεί η μνήμη και οι λοιποί πόροι που καταλαμβάνει. Με τον τρόπο αυτό το λειτουργικό είναι ιδιαίτερα γρήγορο και αποδοτικό, ακόμη και σε συσκευές με μικρές δυνατότητες και πληθώρα ταυτόχρονα εκτελούμενων εφαρμογών.

Εκτός του Activity Manager αξιοσημείωτοι είναι ο *Content Provider* (*Πάροχος Περιεχομένου*), ο οποίος διαχειρίζεται τα δεδομένα στα οποία έχουν πρόσβαση από κοινού περισσότερες από μία εφαρμογές, ο *Resource Manager* (*Διαχειριστής Πόρων*) που διαχειρίζεται τους πόρους των εφαρμογών, ο *Telephony Manager* (*Διαχειριστής Τηλεφώνου*) που διαχειρίζεται τις κλήσεις, καθώς και ο *Location Manager (Διαχειριστής Τοποθεσίας*) που διαχειρίζεται τη γεωγραφική μας θέση με δεδομένα που παίρνει είτε από το GPS είτε από τον Provider.

2.8 Εσωτερική λειτουργία Εφαρμογών (Application)

Η δομή μιας εφαρμογής στο Android βασίζεται σε τέσσερα διαφορετικά, διακριτά συστατικά που είναι τα *Activity*, *Service*, *BroadcastReciever* και *ContentProvider*. Δεν είναι απαραίτητο κάθε εφαρμογή να αποτελείται και από τα τέσσερα αυτά κομμάτια, αλλά για να υπάρχει δυνατότητα εμφάνισης γραφικής διεπαφής απαιτείται τουλάχιστον ένα activity.



Εικόνα 6: Η ιεραρχία των Managers στο Android.

Οι εφαρμογές έχουν τη δυνατότητα να εκκινούν άλλες εφαρμογές ή τμήματα άλλων εφαρμογών στέλνοντας ένα αντικείμενο τύπου *Intent (Πρόθεση)*. Αυτά τα αντικείμενα περιέχουν μεταξύ άλλων, το όνομα της επιθυμητής προς εκτέλεση ενέργειας. Ο *Intent Manager* (*Διαχειριστής Προθέσεων*) αναλαμβάνει τη διαχείριση των εισερχόμενων intents και την εκκίνηση των αντίστοιχων εφαρμογών ή των τμημάτων τους. Η λήψη κάποιου intent μπορεί να απορριφθεί ανάλογα με συγκεκριμένα κριτήρια.

Τα αντικείμενα τύπου Service και *BroadcastReciever* (*Δέκτης Μετάδοσης*) επιτρέπουν την εκτέλεση εργασιών στο παρασκήνιο και παρέχουν επιπλέον λειτουργικότητα σε άλλα τμήματα

κώδικα. Ένα παράδειγμα τέτοιας υπηρεσίας είναι η χρήση των Google Maps από την εφαρμογή μας. Οι broadcast recievers ενεργοποιούνται από *συμβάντα* (*events***)** που πυροδοτούνται και διαρκούν μόνο για μικρά χρονικά διαστήματα, ενώ ένα service μπορεί να εκτελείται για μεγάλο χρονικό διάστημα. Οι broadcast receivers δεν παρέχουν user interface, αλλά μπορούν να ενημερώσουν το χρήστη για κάποιο γεγονός μέσω των status bar notifications. Είναι υπεύθυνοι για τη λήψη των *intents,* που με τη σειρά τους είναι υπεύθυνα για την εναλλαγή μεταξύ των activities.

Ο κώδικας μιας εφαρμογής κι επιπλέον δεδομένα όπως βιβλιοθήκες, εικόνες και άλλα απαραίτητα για την εκτέλεση της εφαρμογής συμπεριλαμβάνονται σε ένα *.apk* αρχείο το οποίο είναι εκτελέσιμο στο περιβάλλον του Android.

2.8.1 AndroidManifest.xml

Όλες οι εφαρμογές που εκτελούνται στην Dalvik εικονική μηχανή χρειάζονται ένα XML αρχείο, το *AndroidManifest.xml* τοποθετημένο στο ριζικό (root) φάκελο της εφαρμογής. Το αρχείο αυτό χρησιμοποιείται από το σύστημα για τη λήψη πληροφοριών που αφορούν την εφαρμογή και έχουν να κάνουν με την οργάνωση και τη διοίκησή της. Μέσα στο αρχείο αυτό περιλαμβάνονται μεταξύ άλλων το όνομα της εφαρμογής, τα συστατικά της, οι άδειες που χρειάζεται για να λειτουργήσει (π.χ. άδεια για μετάδοση πληροφοριών μέσω Wi-Fi), οι απαραίτητες βιβλιοθήκες κ.α.

2.8.2 Δραστηριότητες (Activities)

Δραστηριότητα (activity) είναι το σύνολο των γραφικών στοιχείων στην οθόνη μια δεδομένη στιγμή, όπως μια σελίδα ενός web browser ή μια σελίδα ρυθμίσεων. Περιέχει τα οπτικά στοιχεία που παρουσιάζουν τα δεδομένα κι επιτρέπουν τη διάδραση της εφαρμογής με το χρήση. Κάθε εφαρμογή μπορεί να διαθέτει πολλαπλά activities τα οποία εναλλάσσονται με τη χρήση των intents. Όλα τα activities αποτελούν υποκλάσεις της κλάσης *android.app.Activity* και ο κύκλος ζωής τους ελέγχεται από τις μεθόδους, με πιο βασικές τις:

- **onCreate**() Η αρχική συνάρτηση που εκτελείται όταν δημιουργείται ένα activity.
- **OnDestroy**() Εκτελείται αντίστοιχα όταν καταστρέφεται ένα activity.
- **OnResume**() Εκτελείται όταν ένα activity είναι ορατό στο προσκήνιο και έτοιμο να δεχτεί ή να επεξεργαστεί δεδομένα από το χρήστη.
- **OnPause**() Η συνάρτηση αυτή είναι υπεύθυνη για τις διεργασίες που πρέπει να γίνουν ώστε να ετοιμάσουν το activity για αποθήκευση και μετάβαση στο παρασκήνιο.
- **OnRestart**() Η συνάρτηση αυτή αναλαμβάνει να αποκαταστήσει ένα προηγούμενα αποθηκευμένο activity το οποίο καλείται να εμφανιστεί ξανά.

Προθέσεις (Intents), Φίλτρα προθέσεων και Δέκτες μετάδοσης (Broadcast Receivers)

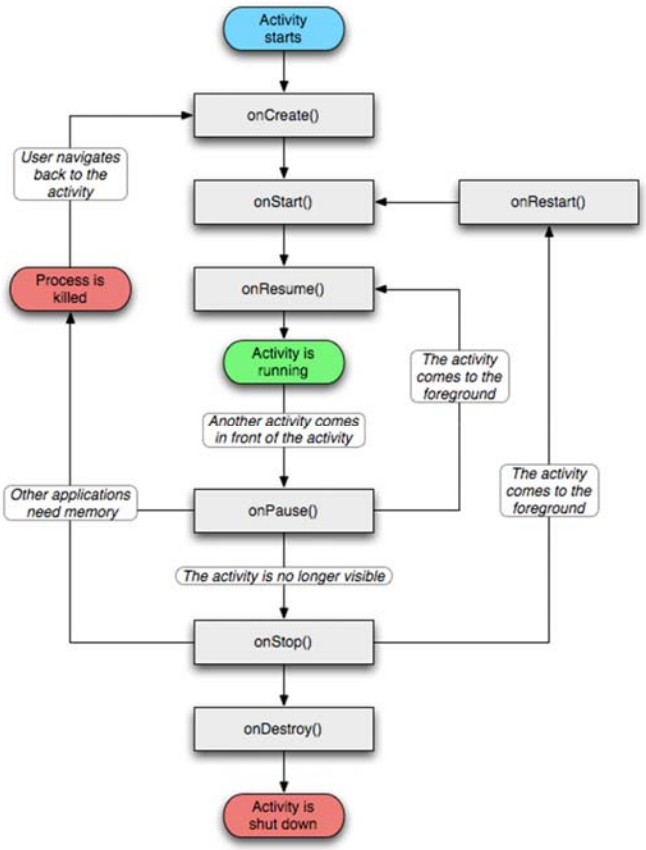
Με εξαίρεση τις λειτουργίες του *Content Provider*, τα υπόλοιπα τρία τμήματα μιας εφαρμογής (activities, broadcast receivers και services) ενεργοποιούνται μέσω *intents*. Το intent είναι ένα αντικείμενο που δημιουργείται κι αποστέλλεται ασύγχρονα (χωρίς την επίτευξη σύγχρονης σύνδεσης) και περιλαμβάνει ένα μήνυμα το οποίο περιγράφει την ενέργεια που χρειάζεται να γίνει. Το μήνυμα αυτό περιέχει είτε το όνομα της προς εκτέλεση *ενέργειας* (*action*), είτε το όνομα της ενέργειας που δημιούργησε το intent. Η πρώτη περίπτωση αφορά τα activities και services με χαρακτηριστικές ενέργειες *ACTION_EDIT* και *ACTION_VIEW*, ενώ η δεύτερη αφορά τα broadcast receivers με χαρακτηριστική ενέργεια την *ACTION_TIME_TICK* . Χώρια από την ενέργεια, το

μήνυμα περιέχει επίσης ένα *URI* (*Uniform Resource Identifier*) που προσδιορίζει τα δεδομένα που θα χρησιμοποιηθούν για την ενέργεια. Προαιρετικά, το intent αντικείμενο μπορεί να διαθέτει κατηγορία, τύπο δεδομένων, όνομα συστατικού (component name) κ.α.

Το Android χρησιμοποιεί διαφορετικά συμβάντα για να παραδώσει τα intents μέσα σε μια εφαρμογή. Στην περίπτωση ενός activity αυτό γίνεται μέσω της μεθόδου *OnNewIntent()*, στην περίπτωση ενός service μέσω της *onBind()* και σε αυτή ενός broadcast μέσω της *Context.sendBroadcast()* ή κάποιας παρόμοιας. Το Android δημιουργεί και στέλνει το intent στην *onReceive()* μέθοδο όλων των κατάλληλα καταχωρημένων παραληπτών. Τα intents μπορούν να φιλτραριστούν από μια εφαρμογή ώστε να καθοριστεί ποια από αυτά μπορούν να επεξεργαστούν από τα διάφορα τμήματα της εφαρμογής. Η λίστα των αδειών της εφαρμογής ορίζεται στο *manifest* αρχείο, και έτσι το σύστημα μπορεί να καθορίσει τα επιτρεπόμενα intents πριν ξεκινήσει μια εφαρμογή.

2.8.3 Πάροχος Περιεχομένου (Content Provider)

Η αποθήκευση και ανάκτηση δεδομένων στις εφαρμογές του Android γίνεται μέσω των *Content Providers (πάροχοι περιεχομένου)*. Αυτοί οι διαχειριστές μπορούν να χρησιμοποιηθούν και για το διαμοιρασμό δεδομένων μεταξύ πολλαπλών εφαρμογών, με την προϋπόθεση ότι οι εμπλεκόμενες εφαρμογές διαθέτουν τις κατάλληλες άδειες για πρόσβαση σε δεδομένα της συσκευής. Το Android έχει προκαθορισμένους διαχειριστές για διάφορους τύπους δεδομένων, όπως εικόνες, βίντεο, επαφές κ.α, τους οποίους μπορεί κάποιος να βρει για χρήση στο πακέτο *android.provider*. Για τη χρήση ενός πάροχου, η εφαρμογή κάνει μια αίτηση στο *Content Resolver* που επιστρέφει το κατάλληλο *ContentProvider* αντικείμενο. Όλοι οι διαχειριστές προσπελαύνονται παρόμοια με μια βάση δεδομένων, μέσω ενός *URI* (Uniform Resource Identifier) καθορίζεται ο απαραίτητος *content provider*, ένα όνομα πεδίου και ο τύπος δεδομένων του. Οι εφαρμογές μπορούν να έχουν πρόσβαση σε έναν Content Provider μόνο μέσω ενός *Content Resolver* και ποτέ ευθέως. Σε περίπτωση που μια εφαρμογή θέλει να αποθηκεύσει δεδομένα που δεν προορίζονται να διαμοιραστούν, μπορεί να το επιτύχει μέσω της χρήσης μια τοπικής *SQLiteDataBase*.



Εικόνα 7: Οι καταστάσεις από τις οποίες περνά ο χρόνος ζωής ενός Activity.

2.8.4 Δραστηριότητες στο Παρασκήνιο(Background Activities)

Συχνά οι εφαρμογές χρειάζεται να εκτελέσουν υποστηρικτικές λειτουργίες στο παρασκήνιο ή δε απαιτούν καθόλου γραφικό περιβάλλον. Το Android για τέτοιες περιπτώσεις διαθέτει τις κλάσεις *BroadcastReceiver* και *Service*. Εάν πρόκειται για μια σύντομης διάρκειας λειτουργία προτιμάται η BroadcastReceiver, ενώ για εργασίες που έχουν μεγάλο χρόνο εκτέλεσης, η Service. Αν και δεν προτείνεται ρητά η χρησιμοποίηση ξεχωριστών threads, συνήθως services ή broadcast receivers εκτελούνται στο δικό τους thread. Διαφορετικά, το Android εκλαμβάνει την καθυστέρηση εκτέλεσης ως μη απόκριση της εφαρμογής και την τερματίζει.

Ο broadcast receiver ενεργοποιείται μέσω της onReceive() μεθόδου και ακυρώνεται μετά το πέρας της. Αυτή η συμπεριφορά καθιστά απαραίτητη την χρήση μόνο σύγχρονα εκτελέσιμων (synchronous) συναρτήσεων από ένα broadcast receiver, αφού με την ολοκλήρωση της εκτέλεσής του, όλοι οι πόροι του θα αποδεσμευτούν προκαλώντας αυτόματο τερματισμό σε τυχών λειτουργίες που συνεχίζουν να εκτελούνται με ασύγχρονο τρόπο (asynchronous). Ο διαχωρισμός synchronous και asynchronous αφορά τον τρόπο εκτέλεσης μιας συνάρτησης όταν αυτή καλείται. Σύγχρονα εκτελεσμένες είναι οι συναρτήσεις αυτές όπου το βασικό νήμα της εφαρμογής περιμένει το πέρας της εκτέλεσής τους για να συνεχίσει. Ασύγχρονα εκτελεσμένες είναι οι συναρτήσεις οι οποίες καλούνται ώστε να ξεκινήσει η εκτέλεσή τους αλλά σε διαφορετικό νήμα. Έτσι, με το που κληθούν, το κεντρικό thread της εφαρμογής και η συνάρτηση συνεχίζουν παράλληλα την εκτέλεσή τους.

Ένα service από την άλλη πλευρά, επιτρέπει σε μια εφαρμογή να εκτελείται για μεγάλα χρονικά διαστήματα στο παρασκήνιο και χρησιμοποιείται από άλλες εφαρμογές του συστήματος. Δύο τρόποι υπάρχουν για τη χρήση ενός service, η εκκίνησή του μέσω μιας εντολής ή απομακρυσμένα μέσω του *RPC* (*Remote Procedure Calls*).

Και τα δύο εργαλεία εκτέλεσης στο παρασκήνιο χρειάζεται να δηλωθούν στο manifest της εφαρμογής για να επιτρέψουν στο Android να καθορίσει την κλάση που θα χρησιμοποιηθεί.

2.8.5 Χρόνος ζωής και κατάσταση μια Εφαρμογής

Η κατάσταση μιας εφαρμογής καθορίζεται από την κατάσταση στην οποία βρίσκονται τα διάφορα τμήματά της και κυρίως τα activities που διαθέτει. Κατά τη διαφοροποίηση της κατάστασης των επιμέρους τμημάτων της, ο τύπος μια εφαρμογής αλλάζει επίσης. Με την εκκίνηση της εφαρμογής τα ξεχωριστά τμήματά της αρχικοποιούνται και εάν πρόκειται για κάποιο activity καλούνται στη σειρά οι *functions* (μέθοδοι***)*** *onCreate(), onStart(), onResume().* Η πρώτη καλείται μόνο μια φορά στο χρόνο ζωής της εφαρμογής, οι υπόλοιπες όμως πιο συχνά. Εάν ένα activity χάσει το focus, τότε καλείται η *onPause()* και όταν το activity δεν είναι πια ορατό η *onStop().* Πριν την καταστροφή ενός activity καλείται η *onDestroy()* που τερματίζει και το χρόνο ζωής του .

Κάθε μέθοδος καλείται σε ειδικό *event* (*συμβάν)* για να επιτρέψει στο activity να διατηρήσει την *κατάστασή* του (*state*) ή να εκκινήσει σωστά.

Ο χρόνος ζωής ενός service έχει λιγότερες πιθανές καταστάσεις αφού οι *onResume(), onPause* και *onStop()* δεν υπάρχουν. Για s`ervices που χρειάζονται αλληλεπίδραση με το χρήστη υπάρχουν οι *μέθοδοι onBind(), onUnbind* και *onRebind()* που καλούνται για εκκίνηση, σταμάτημα κι επανεκκίνησή τους. Ο τύπος εναλλάσσεται μεταξύ του foreground κατά τη δημιουργία και καταστροφή και του service κατά την εκτέλεση. Αντίστοιχα οι broadCastReceivers έχουν μόνο την *onReceive()* μέθοδο.

Εάν ένα service χρειάζεται να αποθηκεύσει την κατάστασή του πριν σταματήσει, μπορεί να χρησιμοποιήσει την *onSaveInstanceState()*, που καλείται πριν την *onPause()* κι επιτρέπει την αποθήκευση των δεδομένων του service. Αργότερα, η αποθηκευμένη κατάσταση του αντικειμένου μπορεί να περάσει στην *onCreate()* ή στην *onRestoreInstance()* και να επαναφέρει την κατάσταση του activity.

2.9 Πως εκτελούνται οι εφαρμογές στο Android

Ο τρόπος με τον οποίο είναι οργανωμένη η αρχιτεκτονική του Android, παρέχει πολλές δυνατότητες για τη σύνθεση, την εκτέλεση και τη διοίκηση μιας εφαρμογής. Για το λόγο αυτό γίνεται σαφής διαφοροποίηση των εννοιών *application* (*εφαρμογή*), *process* (*διεργασία*), *task* (*εργασία*) και *thread* (*νήμα*). Ας δούμε όμως πως χρησιμοποιείται η κάθε μία και πως αλληλοσυνδυάζονται.

2.9.1 Διεργασίες (Processes) και Νήματα (threads)

Πέντε τύποι *process* (*διεργασιών*) διακρίνονται στο Android με στόχο να ελέγχουν τη συμπεριφορά του συστήματος και την εκτέλεση των προγραμμάτων. Οι τύποι αυτοί έχουν διαφορετικά επίπεδα σημαντικότητας αυστηρά ιεραρχημένα ως εξής:

Foreground: Ένα process που εκτελεί ένα activity ή ένα service που παρέχει το activity, ένα service που ξεκινά ή σταματά ή ένας *broadcast reciever* που είναι σε κατάσταση λήψης.

Visible: Εάν ένα process κρατά ένα activity παγωμένο αλλά *ορατό* (*visible*) ή ένα service δεμένο σε ένα ορατό activity χωρίς foreground μέρη, τότε αυτή η διεργασία κατηγοριοποιείται ως visible.

Service: Ένα process που εκτελεί ένα ήδη εκκινηθέν service.

Background: Ένα activity που δεν είναι πλέον ορατό στην οθόνη του χρήστη δεσμεύεται από ένα background process.

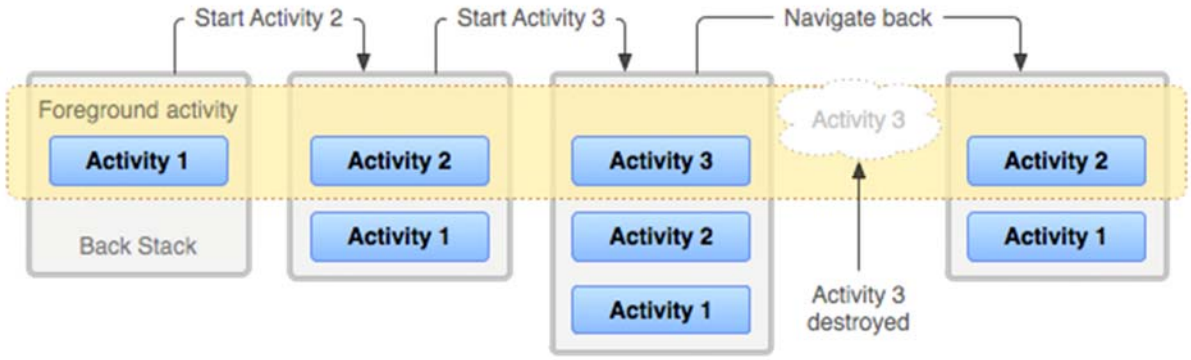
Empty: Αυτά τα processes δεν περιέχουν ενεργά τμήματα εφαρμογών και υπάρχουν μόνο για λόγους προσωρινής αποθήκευσης δεδομένων.

Σε περίπτωση χαμηλής διαθέσιμης μνήμης του συστήματος, η σημαντικότητα ενός process αποτελεί για το σύστημα κρίσιμο παράγοντα προκειμένου να επιλέξει εάν θα οδηγηθεί σε τερματισμό. Έτσι, πρώτα θα τερματιστεί ένα *empty process*, μετά ένα *background* κ.ο.κ. Συνήθως μόνο empty και background processes χρειάζεται να τερματιστούν για την απρόσκοπτη λειτουργία του συστήματος. Το σύστημα είναι σχεδιασμένο να εξαντλεί τα περιθώριά του όσον αφορά τη διαχείριση τμημάτων τα οποία σχετίζονται με οντότητες που επηρεάζουν άμεσα τους χρήστες, όπως ένα activity.

Οι διεργασίες μπορούν να περιέχουν πολλαπλά *threads* (*νήματα*) όπως είναι σύνηθες στα συστήματα τα βασισμένα στο Linux. Οι περισσότερες εφαρμογές αποτελούνται από πολλαπλά threads ώστε να γίνεται διαχωρισμός γραφικού περιβάλλοντος, διαχείρισης δεδομένων και λειτουργιών εισόδου/εξόδου ή χρονοβόρων υπολογισμών. Τα threads αυτά στο επίπεδο εφαρμογών, αποτελούν τα καθιερωμένα *Java threads* που εκτελούνται μέσα στην εικονική μηχανή Dalvik.

2.9.2 Εφαρμογές (Applications) και Εργασίες (Tasks)

Οι δύο όροι αυτοί είναι άρρηκτα συνδεδεμένοι στο περιβάλλον του Android. και πολλές φορές ο χρήστης αντιλαμβάνεται ένα task (εργασία) ως application (εφαρμογή). Στην πραγματικότητα τα tasks είναι μια σειρά από activities, που πιθανά ανήκουν σε πολλά διαφορετικά applications και που εκτελεί ο χρήστης κάποια χρονική στιγμή. Τα activities τοποθετούνται στη stack ("back stack"), με τη σειρά που ανοίγουν. Για παράδειγμα, έστω ένας χρήστης που ανοίγει μια mail εφαρμογή μέσω της οποίας ανοίγει ένα συγκεκριμένο mail, το οποίο περιέχει ένα σύνδεσμο (link) που ανοίγεται μέσω ενός web browser. Σε αυτό το σενάριο το task περιλαμβάνει δύο applications (browser και mail), ενώ υπάρχουν δυο activities από την mail εφαρμογή και ένα από τον browser Η έννοια του task έχει το πλεονέκτημα ότι δίνει στο χρήστη τη δυνατότητα να γυρίσει πίσω βήμα-βήμα.

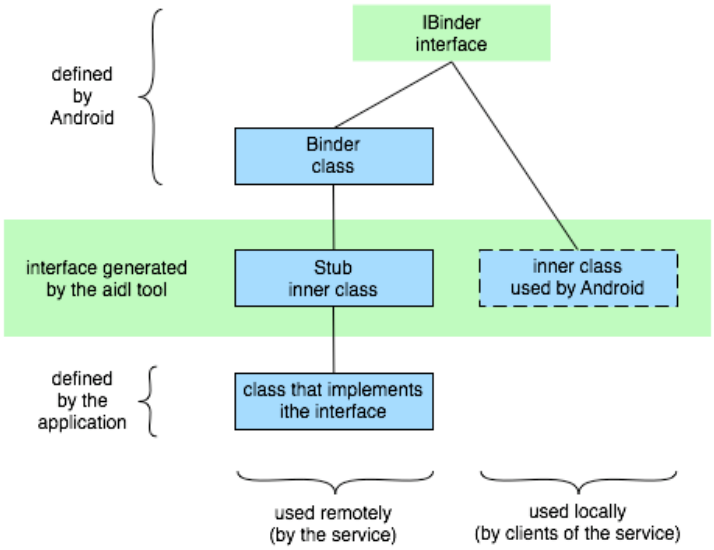


Εικόνα 8 : Ο τρόπος με τον οποίο κάθε νέο activity τοποθετεί ένα αντικείμενο στη στοίβα. Όταν ο χρήστης πατήσει το πλήκτρο πίσω το τρέχον activity καταστρέφεται και το προηγούμενο επιστρέφει στην οθόνη.

2.10 RPC (Remote Procedure Calls)

Το Android διαθέτει μια αρχιτεκτονική τύπου *CORBA* (*Common Object Request Broker Architecture*) και έναν μηχανισμό *COM* (*Component Object Model*). Με τη χρήση αυτών των τεχνολογιών επιτυγχάνει την κλήση απομακρυσμένων συναρτήσεων μέσω δικτυακών διαύλων. Διαθέτει τη δική του γλώσσα για τον ορισμό των *διεπαφών* (*interfaces*), την *AIDL (Android Interface Definition Language*). Το AIDL χρησιμοποιεί κλάσεις μεσολαβητές (proxy classes) για να στέλνει μηνύματα μεταξύ του server και του client. Το AIDL εργαλείο δημιουργεί Java interfaces που πρέπει να είναι διαθέσιμα τόσο στην πλευρά του server όσο και του client.

Το δημιουργημένο interface περιλαμβάνει μια abstract εσωτερική κλάση, που ονομάζεται *Stub*, και που πρέπει να κληρονομηθεί και να υλοποιηθεί από τον client και τον server. Εάν ο server παρέχει κάποιο service, τότε αυτό θα χρειαστεί να επιστρέψει ένα αντικείμενο της κλάσης που υλοποιεί το interface *στην onBind()* μέθοδο του. Η AIDL υποστηρίζει μόνο *συναρτήσεις* (*functions*) και όλες τους είναι σύγχρονου χρονισμού.



Εικόνα 9: Η χρήση του RPC από το Android.

2.10.1 Ασφάλεια Εφαρμογών

Το μοντέλο ασφαλείας του Android βασίζεται πολύ στις δυνατότητες υποστήριξης πολλών χρηστών του Linux. Κάθε εφαρμογή τρέχει με ένα μοναδικό id μέσα στο δικό της *process* (*διεργασία*). Όλες οι

Dalvik εφαρμογές τρέχουν σε έναν δικό τους απομονωμένο χώρο της συσκευής, ο οποίος ονομάζεται *SandBox*. Αρχικά καμία εφαρμογή δεν επιτρέπεται να έχει πρόσβαση σε άλλες διεργασίες ή σε υλικό όπως το GPS ή δικτυακή πρόσβαση.

Αντίθετα απ' ότι σε συστήματα βασισμένα σε εκτελέσιμα αρχεία γραμμένα σε εγγενή κώδικα μηχανής, το Android κάνει εύκολη την επιβολή συγκεκριμένης συμπεριφοράς σε μία εφαρμογή. Ο λόγος γι' αυτό είναι η εικονική μηχανή Dalvik η οποία ελέγχει άμεσα την εκτέλεση του κώδικα και την πρόσβαση στους διάφορους πόρους του συστήματος.

Η βασική τεχνική SandBox απορρίπτει όλα τα αιτήματα μιας εφαρμογής εκτός κι αν υπάρχει ειδική άδεια που να τα επιτρέπει. Οι άδειες αυτές καθορίζονται μέσα στο manifest αρχείο της εφαρμογής. Αυτός ο τρόπος επιτρέπει τη σαφή αναφορά αδειοδοτήσεων που χρειάζεται η εφαρμογή κατά την εκτέλεσή της.

Κατά την εκτέλεσή της μια εφαρμογή παίρνει ένα id το οποίο και χρησιμοποιείται αργότερα για την επιβολή των αδειών σε επίπεδο process και του συστήματος αρχείων. Το id αυτό χρησιμοποιείται για να καθοριστούν επίσης περιπτώσεις διαμοιρασμού δεδομένων μεταξύ των εφαρμογών, και πρέπει να δηλωθεί στο ίδιο manifest αρχείο.

Επίσης, τα επιμέρους τμήματα μιας εφαρμογής είναι δυνατόν να περιοριστούν ώστε να γίνει σίγουρο πως μόνο συγκεκριμένες πηγές μπορούν να ξεκινήσουν κάποιο activity ή να στείλουν κάποια δεδομένα μέσω δικτύου. Ένα service έχει τη δυνατότητα να λάβει συγκατάθεση για χρήση κάποιου πόρου μέσω της *Context.checkCallingPermission()* συνάρτησης. Οι ContentProviders μπορούν επίσης να αποκλείσουν γενική πρόσβαση σε κάποιο αρχείο ή και να επιτρέψουν τη μερική πρόσβαση με βάση κάποιο URI.

2.10.2 Εφαρμογές εγγενούς κώδικα

Είναι πιθανόν κάποιες εφαρμογές να απαιτούν καλύτερους χρόνους εκτέλεσης απ' ότι μπορεί να προσφέρει η Dalvik. Αυτές οι εφαρμογές μπορούν να καταμεριστούν. Ένα κομμάτι τους μένει κι εκτελείται από τη Dalvik για να προσφέρει στην εφαρμογή ένα γραφικό περιβάλλον και το υπόλοιπο κομμάτι εκτελείται ως εγγενής κώδικα (C/C++). Με αυτόν τον τρόπο οι εφαρμογές μπορούν να εκμεταλλευτούν τις δυνατότητες της συσκευής ακόμη κι αν αυτές δεν προσφέρονται μέσω της Dalvik ή του Android.

Τα κομμάτια αυτά κώδικα είναι διαμοιρασμένες βιβλιοθήκες που μπορούν να κληθούν μέσω του *JNI* (*Java Native Interface*). Η διαμοιρασμένη βιβλιοθήκη χρειάζεται να περιληφθεί μέσα στο .apk αρχείο ώστε να φορτωθεί. Ο κώδικας της βιβλιοθήκης αυτής φορτώνεται στο χώρο μνήμης της εφαρμογής στην εικονική μηχανή. Αυτό όμως αφήνει μια τρύπα ασφαλείας σε αντιδιαστολή με την ασφάλεια που αναφέρθηκε σε προηγούμενο υποκεφάλαιο.

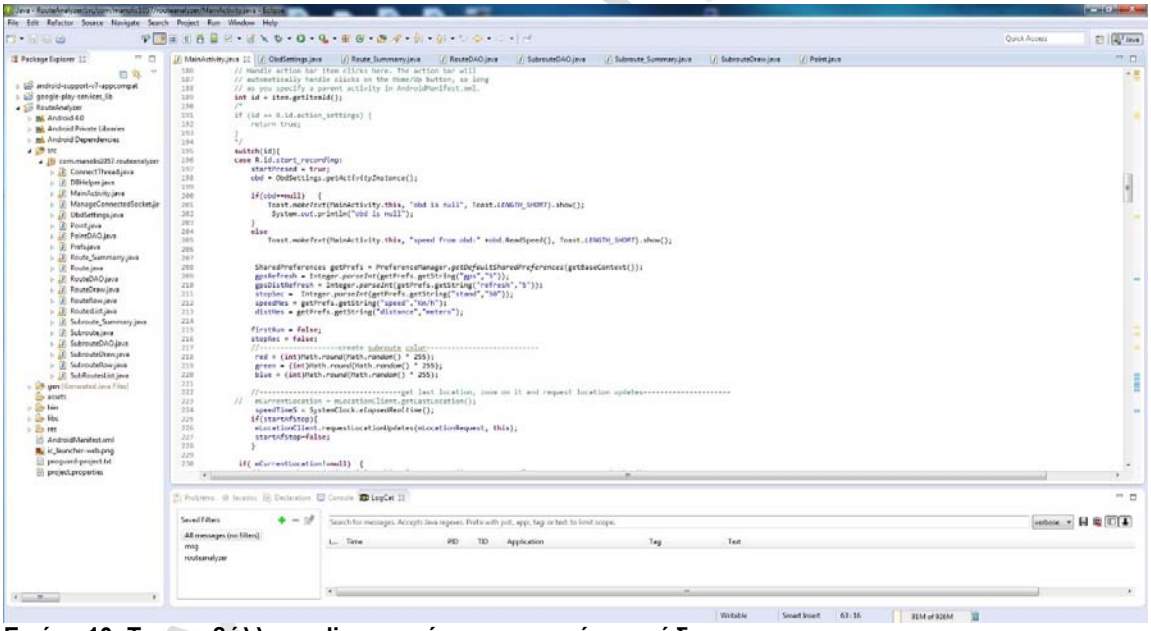
Παρά τα διάφορα θέματα ασφαλείας που παρατηρούνται με τη χρήση εγγενών βιβλιοθηκών στο σύστημα του Android, η μεγάλη διαφορά της απόδοσης των δύο μεθόδων έχει αναγκάσει τη Google να διατηρήσει τη συγκεκριμένη μέθοδο δημιουργίας εφαρμογών.

2.11 Εργαλεία ανάπτυξης- υποστήριξη από Google, κοινότητα

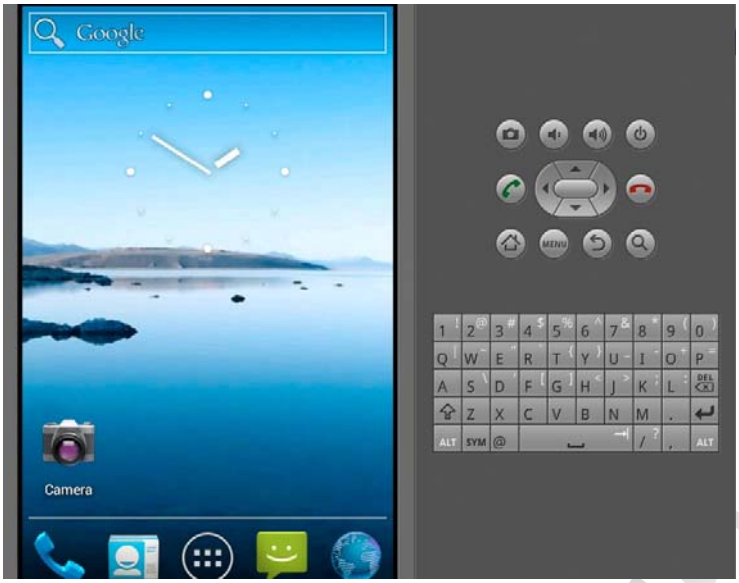
Όπως αναφέραμε και σε προηγούμενα κεφάλαια, η υποστήριξη που μπορεί να βρει ένας μηχανικός λογισμικού είναι από τους βασικότερους παράγοντες για την έγκαιρη και όσο το δυνατόν πιο επιτυχή ολοκλήρωση ενός έργου. Σε αυτόν τον τομέα το Android μπορούμε να πούμε πως διαπρέπει. Εκτός από τους επίσημους δικτυακούς χώρους καθιερωμένους από την ίδια τη Google, υπάρχει πληθώρα πληροφοριών, παραδειγμάτων κώδικα ακόμη και ιδεών που προσφέρονται σε όποιον επιθυμεί να κάνει ανάπτυξη κώδικα για το σύστημα. Ταυτόχρονα διατίθεται και ένα σύνολο πλατφόρμων ανάπτυξης τόσο από την ίδια τη Google όσο και από άλλες εταιρίες ικανές να καλύψουν τις απαιτήσεις του συνόλου των προγραμματιστών. Θα επικεντρωθούμε σε αυτά που ξεχωρίσαμε.

2.11.1 Eclipse IDE

Η πλατφόρμα ανάπτυξης Eclipse, η οποία μέχρι πρότινος υποστηριζόταν επίσημα από τη Google και προωθείτο για ανάπτυξη εφαρμογών. Ήταν διαθέσιμη από το το επίσημο site των προγραμματιστών για Android (http://developer.android.com) έως ότου αντικαταστάθηκε από το Android Studio. Εκεί μπορούσε κανείς να βρει ένα πακέτο με το SDK, το Eclipse και μια μηχανή εικονικής εκτέλεσης του Android για διάφορα σταθερά συστήματα, συμπεριλαμβανομένου των Windows. Η εγκατάσταση του συστήματος και της επέκτασης για Android γίνεται σχετικά εύκολα, ενώ υπήρχαν αναλυτικές οδηγίες αναρτημένες στη σελίδα που προαναφέραμε. Μετά την εγκατάσταση, υπάρχει η δυνατότητα επιλογής μεταξύ πληθώρας εικονικών μηχανών για τις περισσότερες δημοφιλείς συσκευές Android (μεταξύ αυτών και πολλών κατασκευαστών μελών της Handset Alliance). Η ανάπτυξη μπορεί να καθοριστεί για οποιαδήποτε έκδοση του συστήματος από τη 2.0 και μετά. Το εργαλείο αυτό χρησιμοποιήσαμε και εμείς κατά την εκπόνηση της παρούσας εργασίας μιας και ήμασταν εξοικειωμένοι με τη χρήση του. Μόνη διαφοροποίηση η χρήση του εξομοιωτή Genymotion για λόγους που θα αναφέρουμε παρακάτω.



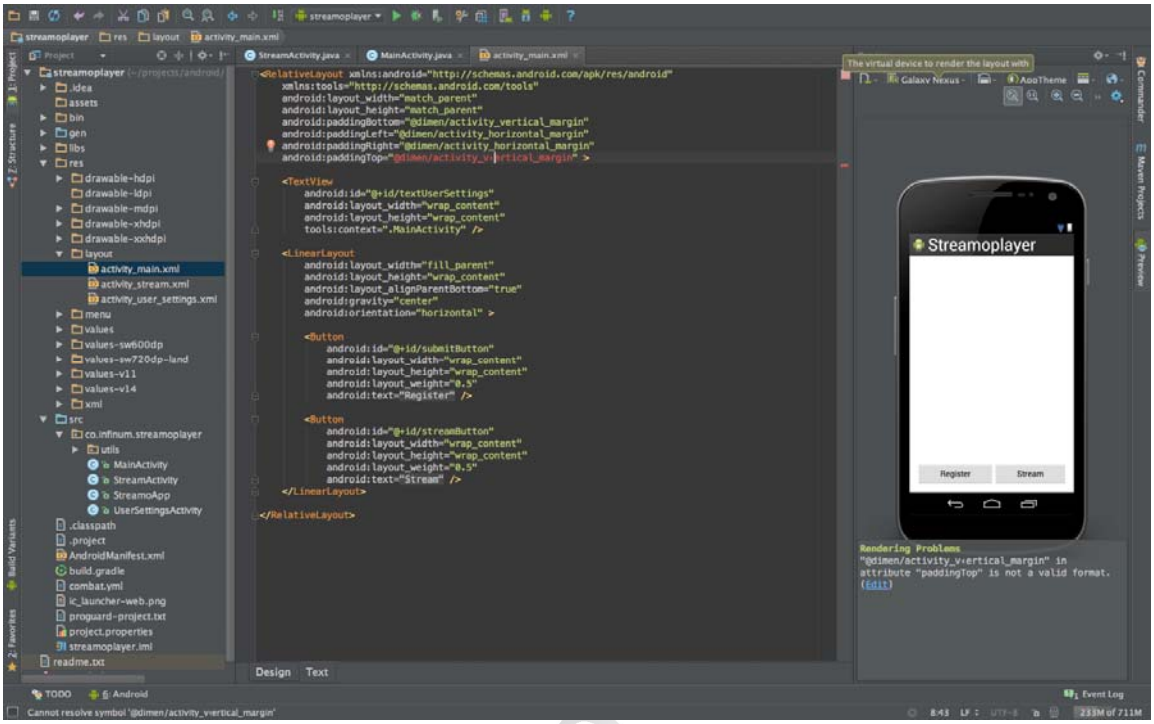
Εικόνα 10: Το περιβάλλον eclipse κατά τη συγγραφή του κώδικα μας.



Εικόνα 11: Ο εξομειωτής Android του Eclipse.

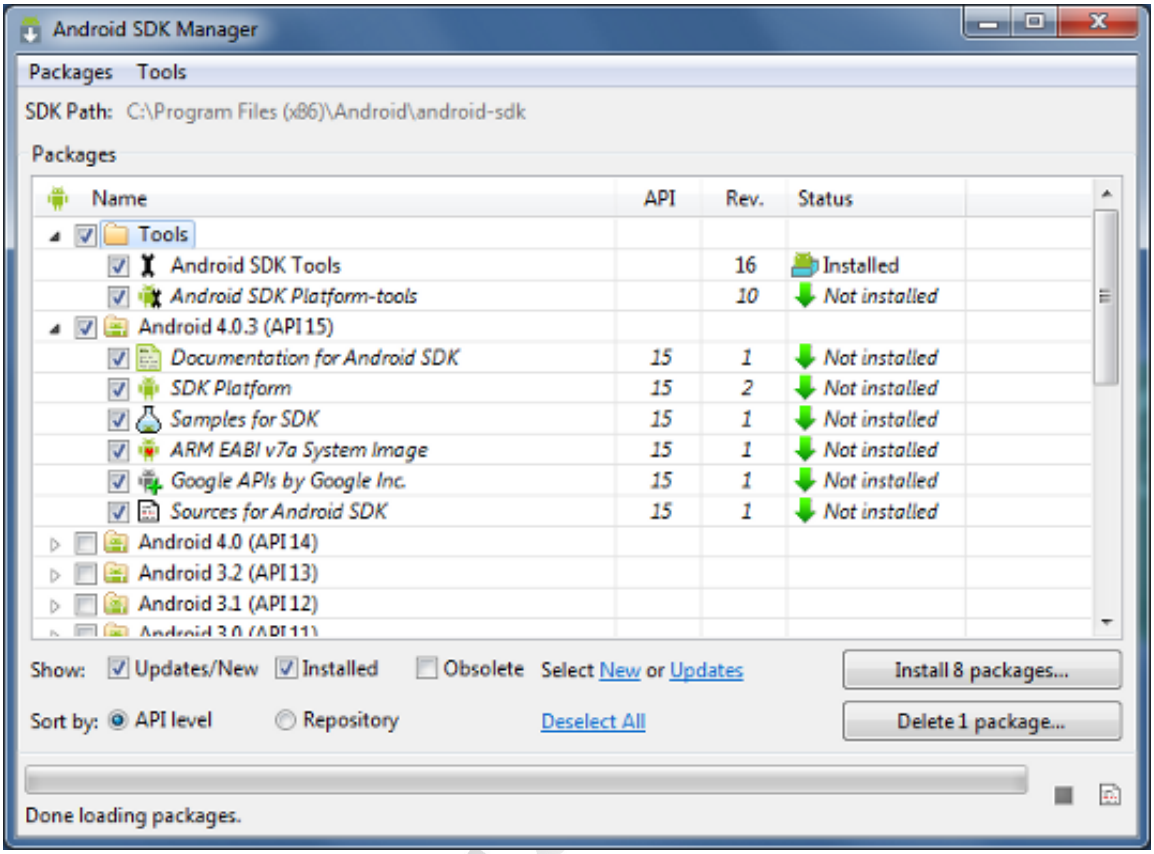
2.11.2 Android Studio

Το πλέον επίσημο περιβάλλον ανάπτυξης εφαρμογών της Google. Είναι διαθέσιμο από τον επίσημο ιστότοπο της εταιρίας http://developer.android.com/sdk/index.html και διατίθεται δωρεάν για windows, Mac OS X αλλά και Linux. Βασισμένο στο IntelliJ IDEA της JetBrains προσφέρει καινοτόμες δυνατότητες όπως υποστήριξή πολλαπλών οθονών, εισαγωγή τμήματα κώδικα από το δημοφιλές ιστότοπο GitHub καθώς και υποστήριξη υπηρεσιών της Google Services.



Εικόνα 12: Περιβάλλον Android Studio. Πηγή : https://www.infinum.co/the-capsized-eight/articles/android-studio-vs-eclipse-1-0

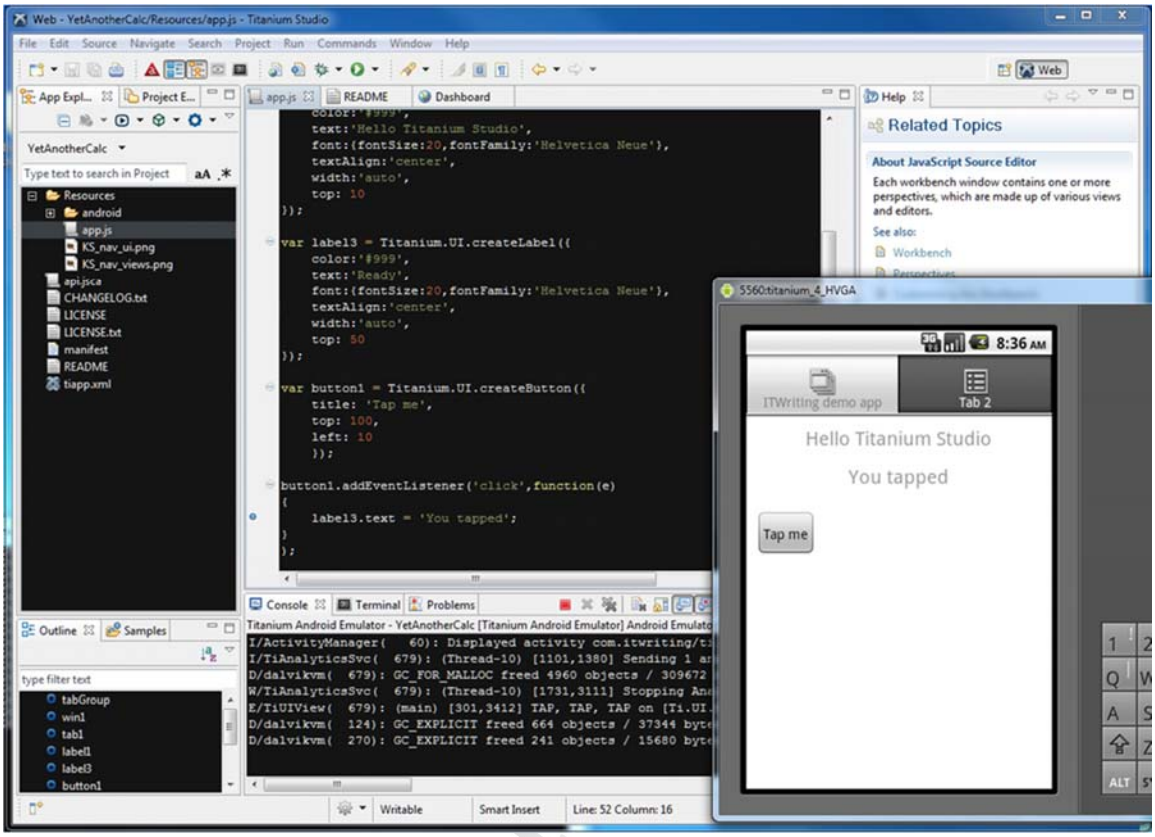
Η εγκατάσταση του είναι αρκετά απλή, προϋποθέτει την ύπαρξη JDK 6 ή ανώτερης. Μετά την εγκατάσταση ο χρήστης θα πρέπει να ενημερώσει το SDK πακέτο του με τα εργαλεία και τις βιβλιοθήκες της έκδοσης Android που θα στοχεύει η εφαρμογή που θα αναπτύξει.



Εικόνα 13. Διαδικασία ενημέρωσης του SDK στο Android Studio. Πηγή: http://developer.android.com/sdk/installing/adding-packages.html

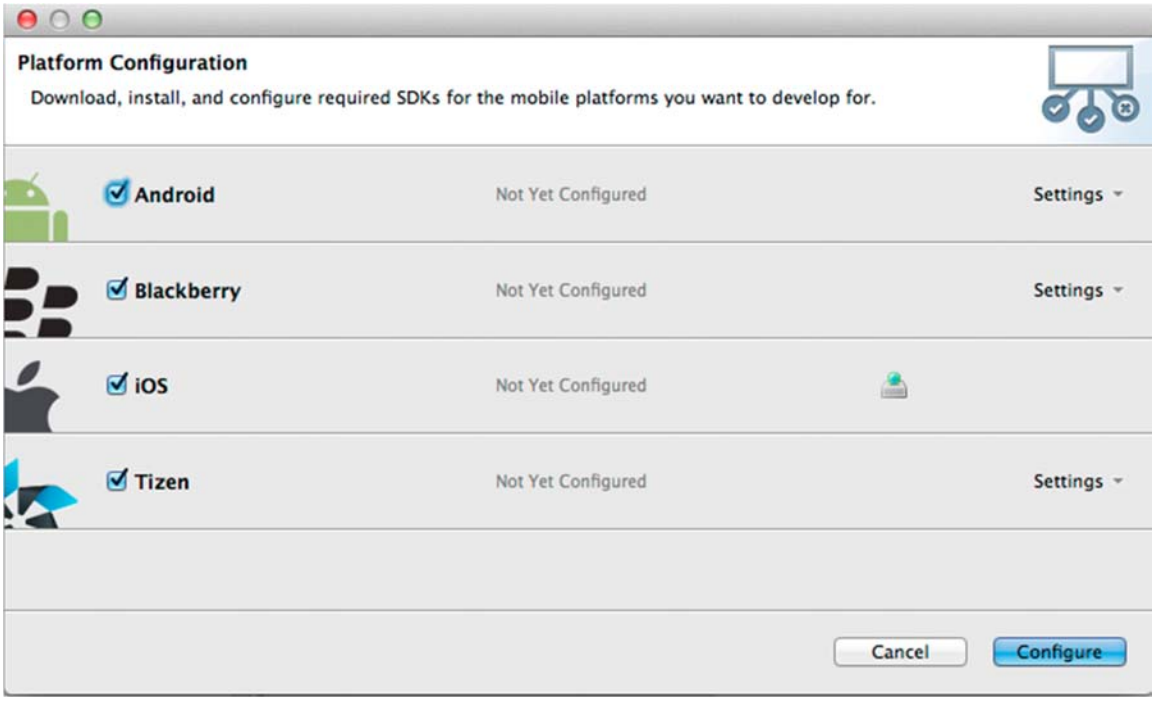
2.11.3 Titanium Studio

Το συγκεκριμένο IDE ανήκει στη κατηγορία των Cross Platform IDE's. Με τα εργαλεία αυτά ο προγραμματιστής μπορεί να αναπτύξει μια εφαρμογή που να εκτελείται σε όλα τα έξυπνα τηλέφωνα ανεξαρτήτου λειτουργικού συστήματος. Η μέθοδος αυτή έρχεται να καλύψει το μεγάλο πρόβλημα των προγραμματιστών. Εφαρμογές για android γράφονται κυρίως σε java ενώ για ios και windows σε C++. Πέρα όμως από τη γλώσσα προγραμματισμού οι πλατφόρμες διαφέρουν και στη λογική λειτουργίας. Με τον τρόπο αυτό ένας προγραμματιστής που είναι εξοικειωμένος με τη java και τη λογική του Android θα πρέπει να δαπανήσει πολύτιμο χρόνο προκειμένου να αναπτύξει εφαρμογές για τις άλλες πλατφόρμες. Με τη χρήση των Cross Platform IDE ο προγραμματιστής αναπτύσσει την εφαρμογή του σε javascript και μπορεί να την εκτελέσει σε οποιαδήποτε πλατφόρμα. Η ευελιξία αυτή όμως έχει κόστος σε λειτουργικότητα μιας και ο προγραμματιστής δεν έχει στη διάθεσή του το σύνολο των δυνατοτήτων που θα είχε αν η εφαρμογή του στόχευε σε συγκεκριμένο λειτουργικό σύστημα.



Εικόνα 14: Το Titanium IDE. Πηγή: http://www.itwriting.com/blog/4511-appcelerator-has-released-titanium-studio-ide-for-cross-platform-mobile-development.html

Το συγκεκριμένο IDE μπορεί κανείς να το προμηθευτεί δωρεάν από τον επίσημο ιστότοπο της εταιρίας Appcelerator http://www.appcelerator.com/titanium/ αφού πρώτα εγγραφτεί στον ιστότοπο. Αφού ολοκληρώσει την εγκατάσταση θα πρέπει να εγκαταστήσει τις βιβλιοθήκες των πλατφόρμων για τις οποίες σκοπεύει να αναπτύξει εφαρμογές.

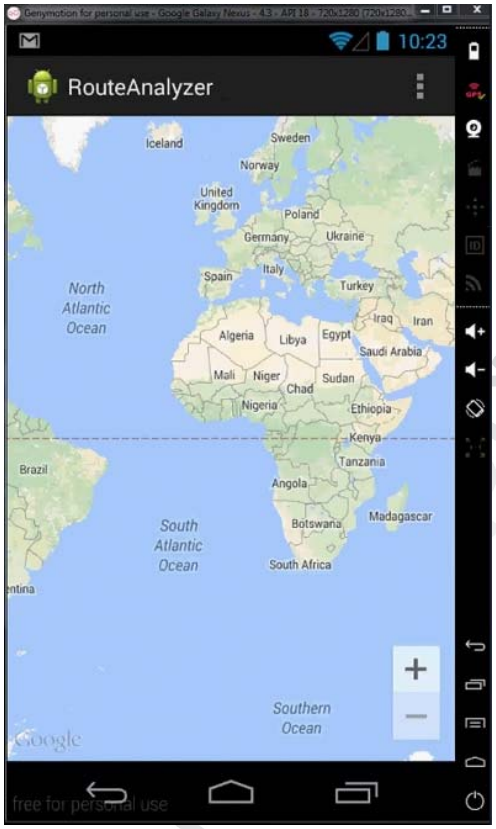


Εικόνα 15: Διαδικασία εγκατάστασης βιβλιοθηκών στο Titanium IDE. Πηγή: http://docs.appcelerator.com/titanium/latest/#!/guide/Quick_Start

2.11.4 Ο Εξομοιωτής Του Android.

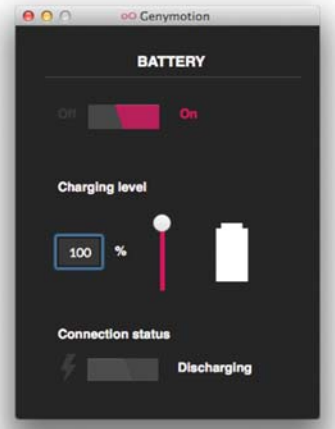
Κατά την ανάπτυξη των εφαρμογών ο προγραμματιστής συχνά θα πρέπει να εκτελέσει το κώδικα που έχει γράψει για να δει κατά πόσο κάνει αυτό που προορίζεται καθώς και αν έχει σφάλματα. Για να το πράξει αυτό θα χρησιμοποιήσει είτε τον Android εξομοιωτή που παρέχουν τα άνω IDE είτε θα τον εκτελέσει απευθείας σε συσκευή που εκτελεί την έκδοση του Android στην οποία απευθύνεται η εφαρμογή που αναπτύσσει. Η χρήση του εξομοιωτή είναι προτιμότερη μιας και δεν απαιτεί την διάθεση συμβατής με την εφαρμογή συσκευής. Επιπλέον στη δική μας περίπτωση η χρήση της συσκευής δεν εξυπηρετούσε ειδικά για το πρώτο στάδιο ανάπτυξης μιας και η εφαρμογή μας χρησιμοποιεί δεδομένα από το GPS αισθητήρα. Έτσι κάθε φορά που θα θέλαμε να δοκιμάσουμε το κώδικά μας θα έπρεπε να περνάμε τη συσκευή και να πραγματοποιούσαμε διαδρομές πράγμα που θα σήμαινε μεγάλη σπατάλη σε πολύτιμο χρόνο. Αντίθετα με τη χρήση του εξομοιωτή μπορούμε να εισάγουμε συντεταγμένες δίνοντας στον εξομοιωτή την αίσθηση ότι βρίσκεται σε κίνηση.

Ο εξομοιωτής που παρέχουν τα άνω IDE είναι αρκετά καλός μέχρι κανείς να ανακαλύψει τον Genymotion[2]. Αποτέλεσμα μιας Γαλλικής StartUp εταιρίας είναι διαθέσιμος από τον ιστότοπο της σε δωρεάν και επι πληρωμής έκδοση. Εμείς χρησιμοποιήσαμε τη δωρεάν μιας και ήταν αρκετή για τις ανάγκες μας.



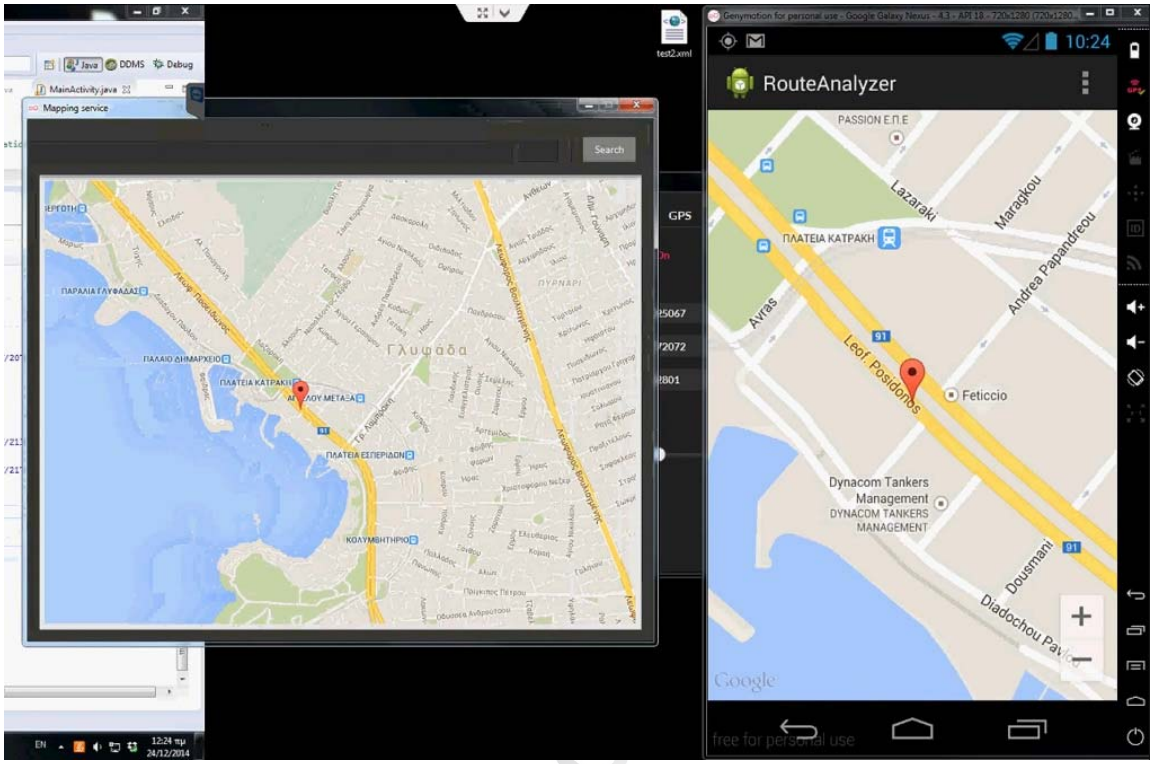
Εικόνα 16: Ο Genymotion εξομοιωτής κατά την εκτέλεση της εφαρμογής μας.

Η εγκατάσταση της είναι απλή καθώς και η χρήση της. Ο χρόνος εκκίνησης είναι σημαντικά βελτιωμένος ενώ χρήστης μπορεί με ένα απλό ‘drag and drop’ να εγκαταστήσει πακέτα σε αντίθεση με τον εξομοιωτή των IDE που για να γίνει αυτό θα πρέπει να εκτελεί εντολές μέσω γραμμής εντολών. Επίσης η ρύθμισή του ποσοστού της μπαταρίας, πράγμα που ήταν απαραίτητο για την εργασία μας, γίνεται απλά με τη μετακίνηση μιας μπάρας.



Εικόνα 17: Ρύθμιση της στάθμης της μπαταρίας στον Genymotion.

Το σημείο όμως που ο Genymotion πραγματικά υπερτερεί σε σχέση με αυτόν των IDE είναι ο τρόπος που εξομοιώνει τον GPS αισθητήρα. Με το που πατήσει ο χρήστης το κουμπί του GPS ο εξομοιωτής ανοίγει το παράθυρο διαχείρισης του GPS. Από το παράθυρο αυτό ο χρήστης ενεργοποιεί το GPS και μπορεί να δώσει συντεταγμένες. Πατώντας όμως πάνω στο κουμπί Google Maps ανοίγει ένα νέο παράθυρο το οποίο εμφανίζει το χάρτη Google Maps και όπου πατήσει ο χρήστης με το ποντίκι οι συντεταγμένες αυτού του σημείου περνάνε στον αισθητήρα. Με τον τρόπο αυτό ο χρήστης έχει τον απόλυτο έλεγχο των τιμών που εισάγει και μπορεί να ελέγξει τα δεδομένα, όπως απόσταση, που υπολογίζει η εφαρμογή του.



Εικόνα 18: Εισαγωγή συντεταγμένων στον Genymotion κατά την εκτέλεση της εφαρμογής μας. Ο χρήστης πατάει στο αριστερό παράθυρο με το ποντίκι, στο σημείο εμφανίζεται μια πινέζα και οι συντεταγμένες αυτές μεταφέρονται στον εξομοιωτή.

Android SDK

2.11.5 Πλατφόρμες Ανάπτυξης

Όπως αναφέραμε και σε προηγούμενα κεφάλαια, η βιβλιογραφία που μπορεί να βρει ένας μηχανικός λογισμικού εάν πάρει την απόφαση να αναπτύξει εφαρμογές για το Android είναι εκτενής. Επίσης τα εργαλεία ανάπτυξης είναι αρκετά και μερικά από αυτά πολύ καλά υποστηριζόμενα. Ας δούμε όμως μερικά από τα πιο σημαντικά εργαλεία που υπάρχουν διαθέσιμα.

2.11.6 Android SDK

Το πακέτο ανάπτυξης της Google είναι ένα περιεκτικότατο σύνολο εργαλείων για ανάπτυξη στο σύστημα του Android. Στο πακέτο περιλαμβάνονται το Android Studio, πλατφόρμα ανάπτυξης Java, το σύνολο των βιβλιοθηκών που διαχειρίζονται τη λειτουργία του συστήματος γραμμένες σε Java, ένας εξομοιωτής συστημάτων Android, τεκμηρίωση του κώδικα των βιβλιοθηκών, ενδεικτικοί κώδικες για τη βιβλιοθήκη και οδηγοί ανάπτυξης. Το πακέτο προσφέρεται για ανάπτυξη σε περιβάλλοντα Linux, Mac OS X από την έκδοση 10.5.8 και ύστερα, καθώς και συστήματα Windows από την έκδοση XP και μετά. Το Android Studio είναι η υποστηριζόμενη πλατφόρμα

ανάπτυξης Java και διατίθεται plug-in για την ανάπτυξη σε Java, αλλά όσο περνά ο καιρός, όλο και περισσότερες πλατφόρμες αποκτούν τα δικά τους πρόσθετα που τους επιτρέπουν πιο εύκολα να πραγματοποιήσουν κώδικα για Android. Πολύ ενδιαφέρον είναι το γεγονός πως υποστηρίζεται η ανάπτυξη σε παλαιότερες εκδόσεις του Android και πως ο προγραμματιστής μπορεί να επιλέξει σε ποιες εκδόσεις θα μπορεί να λειτουργήσει η εφαρμογή του, κάτι που περιορίζεται φυσικά από τις βιβλιοθήκες που χρησιμοποιεί. Η εξαγωγή του πακέτου εγκατάστασης της εφαρμογής γίνεται αυτόματα από το Android Studio και ο χρήστης της εφαρμογής το μόνο που έχει να κάνει είναι να εκτελέσει το .apk αρχείο που χρησιμοποιείται μέσα από τη συσκευή του. Οι εγκατεστημένες εφαρμογές αποθηκεύονται στον φάκελο /data/app όπου κανονικά οι χρήστες δεν έχουν πρόσβαση για λόγους ασφαλείας.

2.11.7 Native Development Kit

Βιβλιοθήκες γραμμένες σε γλώσσα C μπορούν εύκολα να μεταγλωττιστούν για χρήση με την τεχνολογία ARM ή και την πιο διαδεδομένη για x86. Κατόπιν μπορούν να εγκατασταθούν χρησιμοποιώντας το Native Development Kit. Ύστερα μπορούν να κληθούν μέσω το συστήματος κάνοντας χρήση της System.loadLibrary μεθόδου. Ο σχεδιασμός του Android και τα ελλιπή εργαλεία ανάπτυξης που προσφέρονται για ανάπτυξη σε γλώσσα C, καθιστούν το συγκεκριμένο τρόπο ανάπτυξης πολύπλοκο και λίγοι προγραμματιστές τον επιλέγουν.

2.11.8 App Inventor for Android

Τον Ιούλιο του 2010 η Google ανακοίνωσε τη διάθεση του πακέτου αυτού. Πρόκειται για μια πλατφόρμα ανάπτυξης για το Android, η οποία απευθύνεται κυρίως σε ερασιτέχνες προγραμματιστές. Βασίζεται στο Open Blocks Java library project που αναπτύχτηκε στο αμερικάνικο πανεπιστήμιο τεχνολογίας MIT ύπο την εποπτεία του καθηγητή David Wolber. Η πλατφόρμα αυτή προσφέρει πρόσβαση σε πολλές υπηρεσίες κι εργαλεία των κινητών συσκευών, όπως ο δέκτης GPS, υπηρεσίες αποστολής δεδομένων κ.α. Η φιλοσοφία του συγκεκριμένου πακέτου θέτει πολλούς περιορισμούς στη χρήση του από επαγγελματίες της πληροφορικής**.**

2.11.9 HyperNext Android Creator

Παρόμοια με την πλατφόρμα App Inventor, το HyperNext Android Creator επιτρέπει την ανάπτυξη εφαρμογών για το Android από προγραμματιστές που δεν έχουν μεγάλη εξοικείωση με τη Java και το Android SDK. Η φιλοσοφία της πλατφόρμας είναι πως ο χρήστης προγραμματίζει μια σειρά από κάρτες, εκ των οποίων μόνο μία μπορεί να είναι ενεργή κάθε φορά. Φιλοσοφία που ταιριάζει στον τρόπο λειτουργίας των σύγχρονων κινητών συσκευών. Η γλώσσα που χρησιμοποιείται ονομάζεται HyperNext και πρόκειται για μια αναπτυγμένη γλώσσα με μεγάλη ομοιότητα με τα Αγγλικά. Πλατφόρμες σαν αυτή και το App Inventor αναμφίβολα φέρνουν πολλούς περιορισμούς στους χρήστες τους, αλλά είναι γεγονός πως συνεχίζουν να αναπτύσσονται και να βελτιώνονται δίνοντας περισσότερα εργαλεία και τρόπους χρήσης των διαφόρων δυνατοτήτων που προσφέρουν οι κινητές συσκευές και το Android.

2.11.10 The simple project

Το συγκεκριμένο εγχείρημα αφορούσε την ανάπτυξη μιας απλής γλώσσας προγραμματισμού για επαγγελματίες και ερασιτέχνες προγραμματιστές. Παρόμοια με τον τρόπο λειτουργίας της Visual Basic 6, ο προγραμματισμός θα γινόταν μέσω της χρήσης οπτικοποιημένων αντικειμένων (κουμπιά,

labels κ.λ.π) και τον ορισμό *ακροατών συμβάντων* (*event listeners*) για ενέργειες που θα πραγματοποιούνταν με τα συγκεκριμένα αντικείμενα. Το εγχείρημα έχει αφεθεί από τους δημιουργούς του, αφού δεν υπάρχει επίσημη ανανέωση κώδικα από το 2009.

2.11.11 SDL

Η βιβλιοθήκη SDL επίσης προσφέρει δυνατότητες εγκατάστασης εφαρμογών γραμμένες σε C μέσα στο Android. Με τη χρήση ενός middleware κώδικα σε Java, επιτυγχάνεται η εκτέλεση κώδικα C μέσω της μηχανής Java του Android.

Άλλες πλατφόρμες που διατίθενται είναι Ακόμη οι RFO Basic, Basic4Android (Basic4PPC) και Android APIMiner.

2.11.12 Επιλογή πλατφόρμας

Πολλές από τις προαναφερθείσες λύσεις ακούγονται εξαιρετικά ενδιαφέρουσες και χρηστικές και θα μπορούσαν να χρησιμοποιηθούν για την ανάπτυξη κάποιας εφαρμογής για συστήματα Android. Το βασικό όμως πακέτο ανάπτυξης που προσφέρει η Google, είναι εξαιρετικά διαδεδομένο και ολοκληρωμένο. Προσφέρει πλατφόρμα ανάπτυξης, το σύνολο των βιβλιοθηκών για τη διαχείριση του συστήματος και της συσκευής, καθώς και μια εικονική μηχανή για έλεγχο της λειτουργίας της εφαρμογής. Πρόκειται για μια ολοκληρωμένη λύση που εύκολα κάποιος προγραμματιστής που ξεκινάει μια καινούρια δουλειά μπορεί να χρησιμοποιήσει. Φυσικά ο όγκος πληροφορίας που προϋπάρχει για τη συγκεκριμένη λύση ενθαρρύνει ακόμη περισσότερο τη χρήση της. Οι λύσεις που προαναφέρθηκαν, αν και πολύ ενδιαφέρουσες, δεν προσφέρουν στο μέσο προγραμματιστή τα εργαλεία αυτά. Κάποιοι επαγγελματίες, βεβαίως μπορεί να εξυπηρετούνται περισσότερο από κάποια από αυτές. Επαγγελματίες που θέλουν για παράδειγμα να εισάγουν στο Android εφαρμογές που έχουν αναπτύξει σε γλώσσα C, θα προτιμήσουν να χρησιμοποιήσουν τις πιο περίπλοκες τεχνικές για εισαγωγή βιβλιοθηκών C στο Android. Άτομα που δεν έχουν μεγάλη εξοικείωση με τον προγραμματισμό γενικότερα, ίσως προτιμήσουν κάποια από τις λύσεις που προσφέρουν ανάπτυξη με τη χρήση εύκολων εργαλείων όπως το App Inventor. Η επιλογή της πλατφόρμας ανάπτυξης έγκειται στον καθένα ξεχωριστά και πρέπει να γίνεται με κριτήριο το πόσο πιο εύκολη θα κάνει την ανάπτυξη λογισμικού για το σύστημα στο οποίο απευθύνεται. Έτσι κι εμείς, επιλέξαμε τη χρήση της πλατφόρμας που προσφέρει η Google, ως την πιο ολοκληρωμένη για την εφαρμογή που θέλαμε να αναπτύξουμε.

2.12 Περιγραφή Android SDK

2.12.1 Οι βιβλιοθήκες του Android

Ο τρόπος που είναι δομημένες και προσφέρονται στον προγραμματιστή οι βιβλιοθήκες του Android, ακολουθεί τη φιλοσοφία των βασικών βιβλιοθηκών της Java. Κλάσεις που βρίσκονται μέσα σε φακέλους, κατηγοριοποιημένες σύμφωνα με κάποια γενικότερη χρηστικότητα ή κάποια νοηματική σχέση. Έτσι για παράδειγμα οι κλάσεις που αφορούν τη διαχείριση της επικοινωνίας μέσω bluetooth μπορούν να βρεθούν μέσα στο φάκελο bluetooth ο οποίος με τη σειρά του είναι τοποθετημένος στον ευρύτερο φάκελο android. Έτσι έχουμε φακέλους που περιέχουν κλάσεις για χρησιμοποίηση των widgets, φακέλους που περιέχουν κλάσεις για τον καθορισμό της θέσης μέσω GPS κ.α.

2.12.2 Διαθεσιμότητα Documentation

Αξίζει να σημειωθεί στην ενότητα αυτή, πως η συγκεκριμένη πτυχιακή εργασία χρησιμοποιεί πολλές από τις διαθέσιμες τεχνολογίες του Android για την υλοποίηση του πρακτικού μέρους. Είναι

σημαντικό να αναφέρουμε το μέγεθος του όγκου της πληροφορίας που υπάρχει διαθέσιμη σχετικά με τη χρήση των τεχνολογιών Android και τη βοήθεια που μπορεί να λάβει ένας προγραμματιστής που είναι καινούριος στην ανάπτυξη για κινητές συσκευές. Η *τεκμηρίωση* (*documentation*) που παρέχεται από τη Google είναι εξαιρετικά εύχρηστη και πλούσια. Αν αυτό δεν είναι αρκετό, υπάρχουν αμέτρητες αναρτήσεις στο διαδίκτυο από παλαιότερους χρήστες και προγραμματιστές του συστήματος με πολλά παραδείγματα, ενδεικτικές λύσεις και συμβουλές για σωστό προγραμματισμό στην πλατφόρμα. Το μόνο σίγουρο είναι πως η πλατφόρμα του Android προσφέρεται για πειραματισμούς και όποιος αποτολμήσει να προχωρήσει σε ανάπτυξη χρησιμοποιώντας τα εργαλεία της, δε θα βρεθεί μόνος του. Στην προσπάθεια αυτή θα έχει μαζί του πάμπολλους προγραμματιστές που επιχείρησαν το ίδιο ταξίδι πριν ή και ταυτόχρονα με τον αυτόν.

Έχοντας τα παραπάνω στο μυαλό μας ξεκινήσαμε την ανάπτυξη της εφαρμογής μας με κάποιες γενικές κατευθύνσεις και προσαρμόζαμε τις απαιτήσεις, το γραφικό περιβάλλον, τις δυνατότητες και τη λειτουργικότητα από τα νέα πράγματα που εμφανίζονταν μπροστά μας. Η εφαρμογή της πτυχιακής εργασίας μας λοιπόν, χρησιμοποιεί τις διάφορες τεχνολογικές δυνατότητες του Android, έχοντας πάντα τις κεραίες μας ανοιχτές για κάποια καινοτομία που θα ανέβαζε το επίπεδο της εφαρμογής ένα σκαλί παραπάνω.

Αναλυτικό Documentation για τον τρόπο χρήσης των βιβλιοθηκών μπορεί κανείς να βρει στο επίσημο site για προγραμματιστές Android:

http://developer.android.com/reference/packages.html

2.13 GooglePlayServices

2.13.1 Εισαγωγή

Το Google play services είναι ένα API το οποίο προσφέρει η Google για εφαρμογές που τρέχουν σε Android και παρουσιάστηκε πρώτη φορά το 2012. Στην αρχική του έκδοση προσέφερε μόνο πρόσβαση στο Google+ και στο oAuth αλλά εξελίχθηκε και προσφέρει πρόσβαση σε μια πληθώρα εφαρμογών η οποίες συνοπτικά είναι:

- Googleplaygameservices: Δίνει την δυνατότητα σε παιχνίδια να αποκτούν μία κοινωνική (social) διάσταση προσφέροντας leaderboards, achievementsκαι δυνατότητες multiplayer.
- LocationAPI: Προσφέρει πληροφόρηση για την γεωγραφική θέση των χρηστών και την δραστηριότητά τους έτσι ώστε η εφαρμογές να προσαρμόζονται στην δραστηριότητα αυτή.
- Google+: Δίνει την δυνατότητα στους χρήστες να αυθεντικοποιούν την ταυτότητά τους στις εφαρμογές προκειμένου να μοιράζονται προτιμήσεις.
- Maps: Επιτρέπει στις εφαρμογές να χρησιμοποιούν τα google maps και το street view χωρίς να είναι αναγκασμένες να ανοίγουν μια ξεχωριστή εφαρμογή.
- Drive: Είναι μια υπηρεσία cloud storage και μέσω του κινητού επιτρέπει την εύκολη και γρήγορη πρόσβαση και συγχρονισμό σε αρχεία του χρήστη.
- Cast: Επιτρέπει στις συσκευές να προβάλουν τηλεοπτικό περιεχόμενο καθώς και videoκαι ήχο.
- Ads: Δίνει την δυνατότητα να προβάλλονται διαφημίσεις μέσω των εφαρμογών έχοντας σαν στόχο συγκεκριμένα ακροατήρια.
- Wallet: Δίνει την δυνατότητα στους χρήστες να αγοράζουν προϊόντα με έναν απλό τρόπο.
- Other: Άλλα API που προσφέρονται είναι το GoogleFit, Google account authentication καιGoogle Analytics.

2.13.2 Λειτουργία

Οι βιβλιοθήκες του clientπεριέχουν τα interfacesγια τις διάφορες υπηρεσίες και επιτρέπουν να πάρεις έγκριση από τον χρήστη για την πρόσβαση σε αυτές με τα δικά του credentials.Η εφαρμογή έχει μικρό footprintεπομένως ελαχιστοποιεί την επιβάρυνση στο μέγεθος της εφαρμογής. Εάν χρειαστεί η εφαρμογή να χρησιμοποιήσει κάποια καινούργια δυνατότητα μπορεί να αναβαθμιστεί μόλις εκδοθεί η καινούργια έκδοση του play services, χωρίς αυτό όμως να είναι απαραίτητο.

2.13.3 Googleplayservicesapk

Το apk περιέχει τις διάφορες ανεξάρτητες υπηρεσίες και τρέχει στο backgroundτου AndroidOS σαν service. Η επικοινωνία με το serviceγίνεται μέσω του client library και το serviceαναλαμβάνει να κάνει ότι χρειάζεται. Προσφέρει ακόμη μια διαδικασία authorizationγια τον προγραμματιστή και τους χρήστες. Το APKδιανέμεται μέσω του PlayStore επομένως οι αναβαθμίσεις δεν εξαρτώνται από τους τηλεπικοινωνιακούς πάροχους ή τους κατασκευαστές των κινητών. Όλες οι συσκευές με OSμεγαλύτερο από το 2,3 (Gingerbread) μπορούν να αναβαθμιστούν κατευθείαν.Τα πλεονεκτήματα για τις εφαρμογές είναι πως μπορείς να χρησιμοποιείς τις τελευταίες εκδόσεις για του Google Services χωρίς να υπάρχει πρόβλημα με την συσκευή.

Εικόνα 19: Update process

2.13.4 Set Up

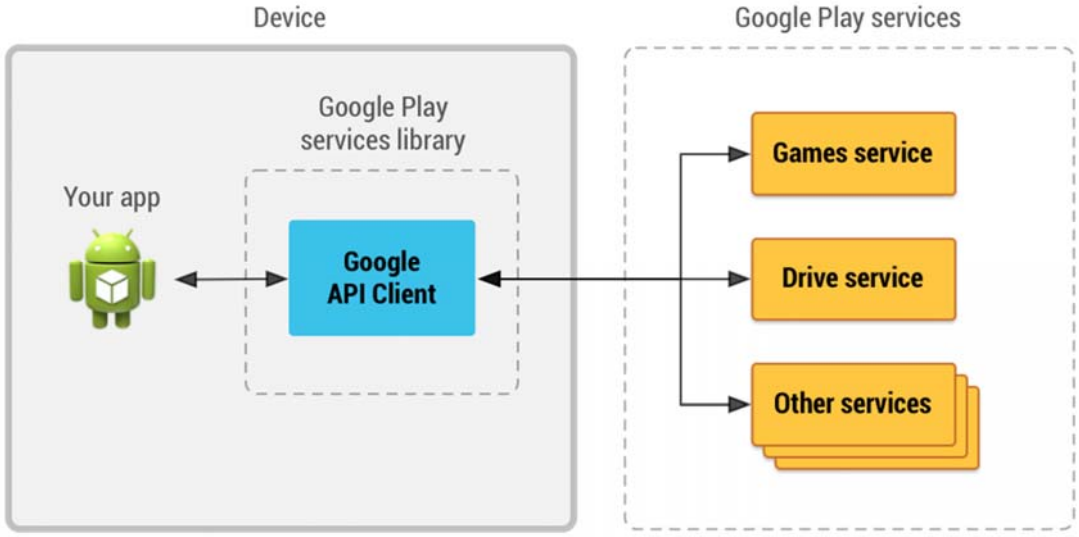
Για την χρήση του Google play services σε μια εφαρμογή πρέπει να χρησιμοποιήσουμε το SDKκαι να έχουμε σαν targetκάποια έκδοση του Android μεγαλύτερη από την 2.3 . Επίσης πρέπει να έχουμε έναν emulator AVD ο οποίος τρέχει σε έκδοση μεγαλύτερη από την 4.4.2. Για την χρήση με το Eclipse AD T χρειάζεται να ακολουθήσουμε τα εξής βήματα:

- Βάζουμε την βιβλιοθήκη του προγράμματος `<android-sdk>/extras/google/google_play_services/libproject/google-play-services_lib/`στην τοποθεσία που έχουμε τις androidεφαρμογές.
- Κάνουμε importτην βιβλιοθήκη στο Eclipse workspace.
- Στην εφαρμογή μας κάνουμε referencετο συγκεκριμένο library.
- Εφόσον το έχουμε σαν dependency πλέον στο project μας δηλώνουμε το εξής στο manifestfile μας μέσα στο <application>tag:<meta-dataandroid:name=”com.google.android.gms.version” android:value=”@integer/google_lpay_services_version” />

Το goggle play services ενημερώνεται αυτόματα, όμως δεν γίνεται ταυτόχρονα για όλους τους χρήστες, ανάλογα με την έκδοση του Androidπου διαθέτουν. Επομένως η εφαρμογή πρέπει να ελέγχει την έκδοση στην οποία βρίσκεται πριν κάνει API transactions.Η βιβλιοθήκη περιέχει μεθόδους που ελέγχουν την έκδοση και εάν βρουν κάποια ασυμβατότητα οδηγούν τον χρήστη στο playstore προκειμένου να κατεβάσει κάποια νεότερη έκδοση. Η μέθοδος που υλοποιεί αυτό τον μηχανισμό είναι η isGooglePlayServicesAvailable() η οποία εάν επιστρέψει SUCCESSσημαίνει πως η έκδοση που έχουμε είναι up to date.

2.13.5 Πρόσβαση στο Google API

Όταν θέλουμε να συνδεθούμε σε ένα από τα Google APIs που προσφέρονται από το play services πρέπει να δημιουργήσουμε ένα instance του Google Api Client, το οποίο προσφέρει ένα κοινό τύπο πρόσβασης (entry point) για όλα τα servicesκαι διαχειρίζεται την σύνδεση μεταξύ συσκευής του χρήστη και του Google service.



Εικόνα 20: Πως το Google API client παρέχει ένα interface για την επικοινωνία με τα services.

2.13.6 Αρχικοποίηση σύνδεσης

Μόλις η εφαρμογή είναι πλέον συνδεδεμένη με την βιβλιοθήκη του Google play, δημιουργούμε ένα instance του Google Api Client χρησιμοποιώντας τον GoogleApiClient.Builderμέσα στην onCreate() μέθοδο. Ο GoogleApiClient.Builder παρέχει μεθόδους που επιτρέπουν να δηλώσεις ποιες μεθόδους θέλεις να χρησιμοποιήσεις καθώς και τα επιθυμητά scopesγια το oAuth 2.

```
GoogleApiClient mGoogleApiClient =newGoogleApiClient.Builder(this)
 .addApi(Drive.API)
 .addScope(Drive.SCOPE_FILE)
 .build();
```

Μπορούμε να προσθέσουμε πολλαπλά API και πολλαπλά scopes στον ίδιο APIclient με την χρήση του addAPI() και του addScope(). Πριν ξεκινήσουμε την σύνδεση με την χρήση της connect() πρέπει να προσδιορίσουμε ένα implementation για τα callbackinterfaces. Τα interfaces αυτά λαμβάνουν callbacks από την ασύγχρονη connect() σε περίπτωση που έχουμε success,fail ή το service έχει γίνει suspended. Εφόσον έχουμε υλοποιήσει αυτά τα callbacks καλούμε την connect() στην onStart() και την disconnect() στην onStop().

2.13.7 Διαχείριση Failures

Όταν λάβουμε κάποιο cal lστην onConnectionFailed() πρέπει να καλέσουμε την hasResolution() στο ConnectionResult object. Εάν επιστρέψει true, μπορούμε να ζητήσουμε από τον user να λύσει το πρόβλημα καλώντας την startResolutionForResult() στο ConnectionResultobject. Η startResolutionForResult() συμπεριφέρεται σαν την startActivityForResult() και ξεκινά την κατάλληλη activity για την επίλυση του error. Εάν η hasResolution() επιστρέψει false καλούμε την GooglePlayServicesUtil.gerErrorDialog() και περνούμε τον κώδικα που επέστρεψε το λάθος. Αυτή επιστρέφει ένα Dialog κατάλληλο για το λάθος που εμφανίστηκε. Αυτό το Dialog μπορεί να επιστρέψει απλά ένα μήνυμα ή ένα prompt για το launch ενός activity που το επιλύει. Όταν επιλυθεί το πρόβλημα η activity που βρισκόμαστε επιστρέφει την onActivityResult() και τότε εμείς μπορούμε να καλέσουμε την connect() πάλι.

Για την αποφυγή της onConnectionFailed() ενώ βρισκόμαστε εν μέσω μίας άλλης προσπάθειας επίλυσης του error, πρέπει να διατηρούμε μια Boolean μεταβλητή που ελέγχει εάν μία τέτοια προσπάθεια τρέχει ήδη. Η μεταβλητή αυτή γίνεται true κάθε φορά που καλούμε την startResolutionForResult() ή το GooglePlayServicesUtil.getErrorDialog(). Γίνεται πάλι false όταν λαμβάνουμε το RESULT_OK στην onActivityResult(). Για να έχουμε εικόνα της μεταβλητής μεταξύ των επανεκκινήσεων της activity την αποθηκεύουμε στο savedinstancedata με την συνάρτηση onSaveInstanceState() και την λαμβάνουμε πάλι στο onCreate().

2.13.8 Επικοινωνία

Μόλις γίνει η σύνδεση ο client μπορεί να κάνει read και writecalls μέσω του συγκεκριμένου API που η εφαρμογή είναι πιστοποιημένη. Όταν γίνει το request το αποτέλεσμα γυρνάει σαν ένα PendingResultobject. Αυτό είναι ένα object που αναπαριστά το request το οποίο όμως δεν έχει ακόμα παραδοθεί στο GoogleService. Για παράδειγμα ένα requestγια διάβασμα ενός αρχείο στο GoogleDrive:

```
uery query =newQuery.Builder()
 .addFilter(Filters.eq(SearchableField.TITLE, filename));
PendingResult result =Drive.DriveApi.query(mGoogleApiClient, query);
```

Όταν έχουμε πλέον το pending request μπορούμε να κάνουμε το call μας είτε σύγχρονο, είτε ασύγχρονο.

2.13.9 Asynchronous calls

Για να κάνουμε το call ασύγχρονο καλούμε την setResultCallback() στο PendingResult και δίνουμε μία υλοποίηση της ResultCallback. Όταν η εφαρμογή λάβει ένα Resultobject στην onResult(), παραδίδεται σαν ένα instance της κατάλληλης υποκλάσης που προσδιορίζεται από το APIπου χρησιμοποιούμαι (πχ DriveApi.MetadataBufferResult).

2.13.10 Synchronous calls

Εάν θέλουμε ο κώδικάς μας να εκτελεστεί με μία αυστηρά καθορισμένη σειρά, επειδή ίσως το αποτέλεσμα του ενός call να είναι παράμετρος για την κλήση κάποιου άλλου, μπορούμε να κάνουμε το call μας σύγχρονο καλώντας την await() στην PendingResult. Αυτή κάνει block το thread και επιστρέφει το Resultobject όταν η κλήση ολοκληρωθεί, πάλι με την μορφή ενός instance της κλάσης που προσδιορίζει το API.

Επειδή ακριβώς η await() κάνει block() το thread μέχρι να επιστρέψει αποτέλεσμα δεν πρέπει ποτέ να κληθεί στο UIthread. Επομένως εάν θέλουμε να κάνουμε σύγχρονα requests στο GooglePlayservice, πρέπει να δημιουργούμε καινούργιο thread, όπως με το AsyncTask μέσα από το οποίο θα κάνουμε το call.

2.13.11 Authorizing

Όταν χρειάζεται να έχουμε πρόσβαση στα GoogleAPIs μέσω HTTPτο GoogleAuthUtil προσφέρουν στους χρήστες έναν ασφαλή και εύκολο τρόπο να επιλέγουν τον λογαριασμό που θα κάνουν login χρησιμοποιώντας ένα oAuthtoken. Αυτό το token μπορούμε να το χρησιμοποιήσουμε και για άλλες HTTP-based επικοινωνίες οι οποίες δεν περιλαμβάνονται στο Google play services library όπως το Blogger και το Translate.

2.13.12 Εγγραφή της εφαρμογής

Πριν εκδώσουμε μια εφαρμογή που λαμβάνει oAuthtokens πρέπει να δηλώσουμε την εφαρμογή στο Google Cloud Console δίνοντας το όνομα της εφαρμογής καθώς και το SHA1 fingerprint του keystore με το οποίο κάνεις sign το release APK. Για να κάνουμε register την εφαρμογή μας στο Google Cloud Console κάνουμε τα εξής βήματα:

- Εάν έχουμε κάποιο υπάρχον project στο οποίο προσθέτουμε την εφαρμογή επιλέγουμε το συγκεκριμένο, διαφορετικά επιλέγουμε CreateProject και προσθέτουμε όνομα και ID.
- Στην πλοήγηση στην αριστερή μεριά επιλέγουμε APIs&auth.
- Επιλέγουμε τα API που θα χρησιμοποιήσουμε θέτοντας το status στο ON.
- Στην αριστερή πλοήγηση πάλι επιλέγουμε credentials.
- Επιλέγουμε create new client ID ή create new key.
- Συμπληρώνουμε την φόρμα που εμφανίζεται.
- Για να πάρουμε το SHA1 fingerprint της εφαρμογής μας στο terminal εκτελούμε: keytool -exportcert -alias <keystore_alias> -keystore <keystore_path> -list –v

Μετά από αυτή τη διαδικασία μας δίνεται το oAuth 2 clientID και το androidkey, αλλά δεν χρειαζόμαστε αυτά για να αυθεντικοποιήσουμε τους χρήστες. Απλώς κάνοντας register την εφαρμογή μας κάνει τα Googleservices προσβάσιμα από αυτή. Για να κάνουμε acquire το oAuthtoken που θα δώσει πρόσβαση στο GoogleAPI πρέπει να αναγνωρίσουμε το account του user. Για τον λόγο αυτό τα Googleplayservices προσφέρουν ένα account picker dialog από τον οποίο καλούμε τον AccountPicker. Το αποτέλεσμα παραδίδεται στην activity μας είναι το όνομα του account που θα χρησιμοποιήσει το oAuthtoken.

Για το άνοιγμα του accountpickerdialog καλούμε την startActivityForResult() χρησιμοποιώντας ένα intent που επιστρέφεται από την AccountPicker.newChooseAccountIntent(). Όταν εκτελείται αυτός ο κώδικας ένα παράθυρο διαλόγου ανοίγει και όταν ο χρήστης επιλέξει τον λογαριασμό που θέλει η activity παίρνει σαν αποτέλεσμα την onActivityResult(). Οι περισσότερες εφαρμογές πρέπει να χρησιμοποιούν την newChooseAccountIntent() η οποία δείχνει πώς:

-Δεν έχει επιλεγεί κάποιος λογαριασμός.

-Δεν υπάρχει λίστα με λογαριασμούς.

-Το παράθυρο πρέπει να δείχνει μόνο λογαριασμούς που σχετίζονται με το domain com.google.
-Εάν υπάρχει μόνο ένας λογαριασμός να μην υποχρεώνει τον χρήστη να επιλέγει.
-Δεν απαιτείται κάποιο auth token.

3. Λήψη Δεδομένων Ταχύτητας Και Κατανάλωσης

3.1 Εισαγωγή

Η ταχύτητα του οχήματος δυνατέ να προκύψει από το σύστημα GPS της συσκευής. Το σύστημα αυτό, ανά χρονικό διάστημα που έχει ορίσει ο χρήστης, μας δίνει την τρέχουσα τοποθεσία. Αποθηκεύοντας την προηγούμενη τοποθεσία μπορούμε να γνωρίζουμε την απόσταση της από την τρέχουσα. Ταυτόχρονα με την τοποθεσία αποθηκεύουμε και τη χρονική στιγμή που τη λάβαμε. Με τον τρόπο αυτό γνωρίζουμε τη διανυθείσα απόσταση καθώς και το χρόνο που απαιτήθηκε για να τη διανύσουμε οπότε έχουμε την ταχύτητα με την οποία κινούμαστε.

Η μέθοδος αυτή δεν είναι αρκετά ακριβής αφού εξαρτάται από τον ρυθμό λήψης δεδομένων τοποθεσίας. Ένας χαμηλός ρυθμός λήψης δεδομένων συνεπάγεται με μειωμένη ακρίβεια. Ομοίως αυξάνει και την πολυπλοκότητα της εφαρμογής αφού θα πρέπει να λαμβάνετε υπόψιν η κατάσταση στην οποία ο χρήστης βρίσκετε σε στάση. Επιπλέον η ανάγκη για αξιόπιστα δεδομένα κατανάλωσης καυσίμου μας οδήγησε στο να αναζητήσουμε διαφορετική πηγή για τις πληροφορίες αυτές. Κατόπιν έρευνας καταλήξαμε στο ενσωματωμένο σύστημα διάγνωσης του οχήματος.

Το σύστημα αυτό, τα τελευταία χρόνια, συναντάται σε όλα τα οχήματα. Επικοινωνεί με την κεντρική μονάδα ελέγχου του οχήματος καθώς και με το σύνολο των αισθητήρων που βρίσκονται στο όχημα. Μέσω αυτού έχουμε πρόσβαση σε ένα μεγάλο σύνολο δεδομένων υψηλής ακρίβειας που αφορούν την λειτουργία του οχήματος και γενικά τη κατάστασή του. Τα δεδομένα αυτά τα επεξεργάζεται η κεντρική μονάδα ελέγχου του οχήματος και καθορίζει την λειτουργία του οχήματος. Κατά συνέπεια μέσω του συστήματος αυτού έχουμε πρόσβαση σε υψηλής ακρίβειας πληροφορία για την ταχύτητα του οχήματος καθώς και για την κατανάλωση την κάθε στιγμή.

3.2 Η Γέννηση Του Ενσωματωμένου Συστήματος Διάγνωσης

Ο κινητήρας ενός οχήματος αποτελείται από ένα σύνολο ευαίσθητων εξαρτημάτων τα οποία συχνά εμφανίζουν δυσλειτουργίες με αποτέλεσμα να επηρεάζεται συνολικά η λειτουργία του κινητήρα. Η μη κανονική λειτουργία του κινητήρα έχει σαν συνέπεια, μεταξύ άλλων, να αυξάνεται η κατανάλωση καυσίμου καθώς και η ποσότητα των ρύπων που εκπέμπονται στο περιβάλλον. Για να περιορίσει το πρόβλημα η πολιτεία της Καλιφόρνια το 1966 καθιέρωσε συστήματα ελέγχου εκπομπής ρύπων στα οχήματα. Η κίνηση αυτή αποτέλεσε την ανάγκη δημιουργίας ενός συστήματος το οποίο θα είναι σε θέση να εντοπίζει τα προβλήματα του κινητήρα και να ενημερώνει τους μηχανικούς ώστε να προβαίνουν στην επισκευή τους. Το σύστημα αυτό ονομάστηκε Ενσωματωμένο Σύστημα Διάγνωσης (On Board Diagnostics) και εφαρμόστηκε αρχικά από την General Motors το 1981. Ακολούθως εξελίχθηκε σε OBD II[3] το οποίο είναι τυποποιημένο και πλέον σήμερα υπάρχει σε όλα τα οχήματα.

3.3 Επικοινωνία Με Το Ενσωματωμένο Σύστημα Διάγνωσης

Οι τεχνικοί προκειμένου να διαβάσουν τα μηνύματα του OBD είναι εξοπλισμένοι με συσκευές οι οποίες συνδέονται στη θύρα του συστήματος και επικοινωνούν με αυτό. Η θύρα αυτή βρίσκεται εντός του οχήματος σε ακτίνα ενός μέτρου από τη θέση του οδηγού.

Εικόνα 21: Θύρα OBD II σε TOYOTA YARIS μοντέλο 1999.

Εικόνα 22: Ενσύρματη συσκευή ανάγνωσης μνημάτων του ODBII.

Εικόνα 23: Ασύρματη συσκευή ανάγνωσης μνημάτων του ODBII μέσω Bluetooth.

Οι συσκευές αυτές αποτελούνται ουσιαστικά από ένα μικροϋπολογιστή, με επικρατέστερο αυτό της Elm Electronics[4] τον ELM327[5] ο οποίος επικοινωνεί με το OBD II και έχει πρόσβαση στα δεδομένα του. Ο μικροϋπολογιστής αυτός γνωρίζει τα πρωτόκολλά που χρησιμοποιούν τα

οχήματα. Έτσι όταν δέχεται μια εντολή η οποία απευθύνεται στο OBDII γνωρίζει πώς να τη μεταφέρει στο όχημα αλλά και πώς να μεταφέρει την απάντηση στο χρήστη.

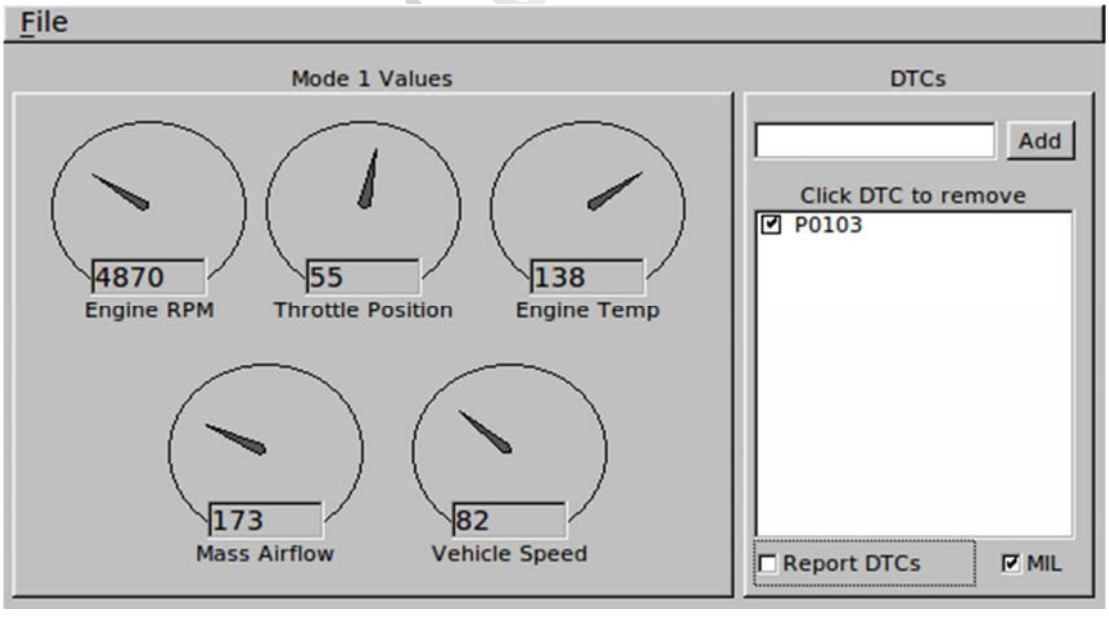
Η χρήση ειδικά του τελευταίου τύπου συσκευών στις μέρες μας είναι ιδιαίτερα διαδεδομένη πέρα από τους τεχνικούς και στους απλούς χρήστες αφού υπάρχει πληθώρα λογισμικού τόσο για υπολογιστές όσο και για έξυπνα κινητά τηλέφωνα μέσω του οποίου έχουν πρόσβαση στα δεδομένα του οχήματος. Με τον τρόπο αυτό ο οποιοσδήποτε μπορεί να συνδέσει τη συσκευή στη OBD II θύρα του οχήματος του, να εγκαταστήσει στη συσκευή του το απαραίτητο λογισμικό και να έχει εικόνα για τις στροφές του κινητήρα, τη ταχύτητα του οχήματος, τη κατανάλωση καθώς και πολλά άλλα.

Εμείς για τις ανάγκες εκπόνησης της παρούσας διατριβής μας χρειαζόμασταν μια διαφορετική προσέγγιση. Η χρήση των άνω συσκευών δεν ήταν η κατάλληλη μιας και η συγγραφή του κώδικα θα απαιτούσε σπατάλη πολύτιμου χρόνου πραγματικών δοκιμών πάνω στο όχημα. Επίσης οι συσκευές αυτές δεν είναι ιδιαίτερα διαδεδομένες στη χώρα μας και η προμήθεια τους θα απαιτούσε επιπλέον χρόνο αφού θα έπρεπε να παραγγελθούν από το εξωτερικό. Για τους λόγους αυτούς καταλήξαμε στη χρήση ενός εξομοιωτή του OBD II.

3.4 Ο Εξομοιωτής OBDSim6.

Πρόκειται για ένα δωρεάν πρόγραμμα ανοιχτού κώδικα γραμμένο σε C++ συμβατό με windows αλλά και linux. Δέχεται εντολές (ερωτήσεις) και επιστρέφει τιμές (απαντήσεις). Τις τιμές μπορεί να τις εισάγει ο χρήστης μέσω του γραφικού περιβάλλοντος είτε να προκύπτουν τυχαία. Διαθέτει διάφορες λειτουργίες αλλά εμείς επικεντρωθήκαμε σε αυτές που μας αφορούν.

Η χρήση του είναι αρκετά απλή. Εμείς κατεβάσαμε από τον ισότοπο http://icculus.org/obdgpslogger/obdsim.html την έκδοση για windows σε έναν υπολογιστή που διαθέτει windows XP και Bluetooth συσκευή. Εκτελώντας την εντολή obdsim.exe -g gui_fltk -w COM7 σε περιβάλλον εντολών ξεκινήσαμε την εκτέλεση του. Η παράμετρος -g gui_fltk εκκινεί το γραφικό περιβάλλον ενώ η -w COM7 είναι η θύρα στην οποία έχει συνδεθεί το έξυπνο κινητό τηλέφωνο μέσω Bluetooth.



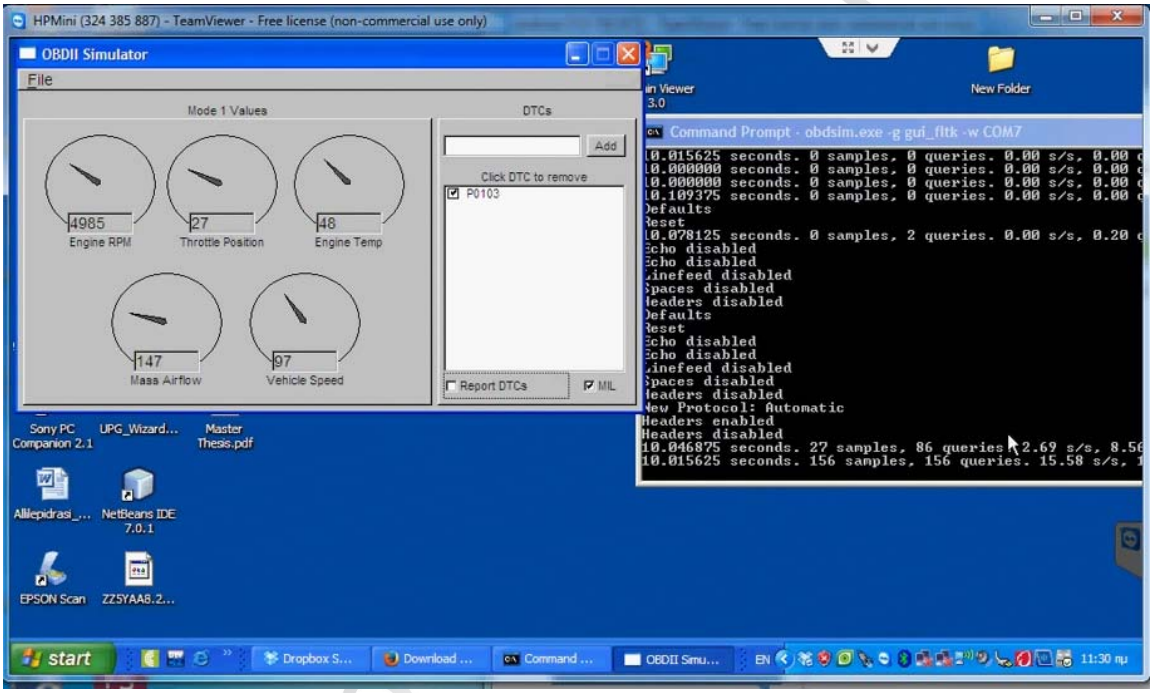
Εικόνα 24: Ο εξομοιωτής OBDSim με γραφικό περιβάλλον.

Μέσω του παραθύρου αυτού ο χρήστης εισάγει τις τιμές που ο εξομοιωτής θα απαντά μετακινώντας με το ποντίκι τους δείκτες των οργάνων. Για παράδειγμα εδώ ο χρήστης έχει δώσει

σαν ταχύτητα την τιμή 82. Η τιμή αυτή είναι σε χιλιόμετρα ανά ώρα οπότε στην ερώτηση ποια η ταχύτητα του οχήματος ο εξομοιωτής θα απαντήσει 82 Km/h.

3.4.1 Δοκιμάζοντας Τη Λειτουργία Του Εξομοιωτή.

Προκειμένου να βεβαιωθούμε για την καλή λειτουργία του εξομοιωτή εγκαταστήσαμε στο κινητό μας τηλέφωνο που διαθέτει λειτουργικό Android την δωρεάν έκδοση της εφαρμογής Torque. Η εφαρμογή αυτή είναι διαθέσιμη στο Google Play και η λειτουργία της είναι να απεικονίζει στο χρήστη τις τιμές των πληροφοριών που επιθυμεί στέλνοντας στο OBDII τα κατάλληλα ερωτήματα. Κατόπιν συνδέσαμε το κινητό με τον υπολογιστή μέσω της Bluetooth θύρας και εκτελέσαμε την εφαρμογή Torque.



Εικόνα 25: Screenshot του υπολογιστή που εκτελεί το OBDSim.

Στο παράθυρο εντολών του υπολογιστή βλέπουμε τα μηνύματα που ανταλλάσσονται μεταξύ των δύο συσκευών. Για παράδειγμα με το που ξεκινά η εφαρμογή Torque στέλνει ένα Reset στον εξομοιωτή ώστε να το επαναφέρει στις στάνταρ ρυθμίσεις. Κατόπιν του ζητά να απενεργοποιήσει τον αντίλαλο (Echo) ώστε να οι απαντήσεις του να μην ξεκινούν με την ερώτηση που δεχτικέ (Echo disabled), να μην χρησιμοποιεί κενά στις απαντήσεις του (Spaces disabled) και να αναζητήσει το πρωτόκολλο επικοινωνίας με το όχημα (New Protocol: Automatic). Αφού ολοκληρωθεί η διαδικασία του ορισμού των κανόνων επικοινωνίας μεταξύ των δυο συσκευών το λογισμικό Torque εμφανίζει στο χρήστη τις τρέχουσες τιμές που αφορούν τη ταχύτητα του οχήματος, τις στροφές του κινητήρα , τη θέση του γκαζιού, τη θερμοκρασία του ψυκτικού υγρού και τέλος τη κατανάλωση καυσίμου. Η κατανάλωση προκύπτει από την ποσότητα αέρα που εισέρχεται στο θάλαμο καύσης του κινητήρα, και ορίζεται από τη τιμή Mass Airflow. Ο χρόνος απόκρισης είναι άμεσος ενώ παραιτείται διαφορά μιας μονάδας μεταξύ των τιμών λόγο της στρογγυλοποίησης που εκτελεί η εφαρμογή Torque στις δεκαδικές τιμές που δίνει ο εξομοιωτής.

Εικόνα 26: Screenshot του κινητού που εκτελεί το Torque.

Μετά από αρκετές δοκιμές καταλήξαμε ότι ο εξομοιωτής λειτουργεί αρκετά καλά και ότι θα μπορούσαμε να βασιστούμε σε αυτόν για να αναπτύξουμε την εφαρμογή μας. Ακολούθησε έρευνα για να εντοπίσουμε τις εντολές που πρέπει να χρησιμοποιήσουμε ώστε να αντλήσουμε τις τιμές που μας ενδιαφέρουν από τον εξομοιωτή καθώς και γενικότερες πληροφορίες που αφορούν τη χρήση του εξομοιωτή και κατ' επέκταση τη χρήση του OBD II.

3.5 Δουλεύοντας με το OBD II.

Η επικοινωνία με τις Bluetooth συσκευές στο λειτουργικό σύστημα Android γίνεται μέσω Sockets. Οι εντολές στέλνονται μέσω ρευμάτων εξόδου (OutputStreams) ενώ οι απαντήσεις διαβάζονται μέσω του ρεύματος εισόδου (InputStream). Προκυμμένου να εξοικειωθούμε με το OBDII εγκαταστήσαμε στο κινητό μια εφαρμογή τερματικού μέσω Bluetooth. Εφαρμογές τέτοιου τύπου υπάρχουν αρκετές δωρεάν και είναι διαθέσιμες μέσω του Google Play. Κατόπιν αρχίσαμε να στέλνουμε εντολές στο OBD II να επεξεργαζόμαστε τις απαντήσεις.

Οδηγίες για τη χρήση του ELM327 και για τις εντολές που δέχεται υπάρχουν στο εγχειρίδιο οδηγιών του αλλά και στο internet γενικότερα. Οι εντολές για το OBDII ομοίως υπάρχουν στο internet και συγκεκριμένα στον ισότοπο της Wikipedia : http://en.wikipedia.org/wiki/OBD-II_PIDs. Μελετώντας τις άνω πηγές εντοπίσαμε ότι το ELM327 δέχεται δύο κατηγορίες εντολών. Οι εντολές που ξεκινάν με τα γράμματα AT απευθύνονται στο ELM327 και προορίζονται για δική του χρήση ενώ οι εντολές που είναι γραμμένες στο δεκαεξαδικό σύστημα απευθύνονται στο ODBII. Στέλνοντας για παράδειγμα την εντολή ATZ το ELM327 μας απαντάει Reset και ακολούθως το χαρακτήρα '>'. Αποτέλεσμα της εντολής είναι ο ELM327 να κάνει επανεκκίνηση στις στάνταρ ρυθμίσεις ενώ ο χαρακτήρας '>' αποτελεί το τέλος του μηνύματος του. Άρα εμείς θα πρέπει να εισάγουμε στο ρεύμα εξόδου την εντολή μας και ακολούθως να διαβάζουμε το ρεύμα εισόδου μέχρι να εντοπίσουμε το χαρακτήρα '>'.

Οι στάνταρ εντολές του OBDII , όπως ορίζεται από το πρωτόκολλο SAE J1979 χωρίζονται σε δέκα λειτουργίες (Modes). Οι εντολές που εμείς χρειαζόμαστε ανήκουν στη πρώτη κατηγορία (Mode 01) η οποία περιέχει τις εντολές που χρειάζονται για την απεικόνιση των τιμών της συγκεκριμένης χρονικής στιγμής. Για να εντοπίσουμε ποιες εντολές στη κατηγορία αυτή υποστηρίζει το δικό μας OBDII στέλνουμε την εντολή 01 00 στον εξομοιωτή καθώς αυτός εκτελείτε σε λειτουργία γραφικού περιβάλλοντος. Ο εξομοιωτής απαντά 41 00 FB 7D EC 66. Η ίδια εντολή σε λειτουργία τυχαίων τιμών επιστρέφει 41 00 FF FF FF FF.

Τα πρώτα δύο ψηφία (41) σημαίνουν ότι το ELM απαντά σε ερώτηση λειτουργίας Mode 01 και τα δύο επόμενα είναι η εντολή που έλαβε, δηλαδή το 00. Τα υπόλοιπα είναι οι εντολές της λειτουργίας αυτής που μπορεί να απαντήσει. Για να τις εντοπίσουμε ξεκινάμε από τα αριστερά ψηφιά, τα μετατρέπουμε στο δυαδικό σύστημα και όπου 1 η εντολή υποστηρίζεται ενώ όπου 0 η εντολή δεν υποστηρίζεται. Για παράδειγμα:

Δεκαεξαδικό:	F				B				7				D			
Δυαδικό:	1	1	1	1	1	0	1	1	0	1	1	1	1	1	0	1
PID:	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10

Πίνακας 1: Τμήμα των διαθέσιμων εντολών (PID's) που δέχεται ο εξομοιωτής.

Από τον πίνακα προκύπτει ότι για παράδειγμα η εντολή OD η οποία μας δίνει την ταχύτητα του οχήματος είναι διαθέσιμη καθώς και η εντολή 10 η οποία μας δίνει την ποσότητα αέρα σε γραμμάρια ανά δευτερόλεπτο που εισέρχεται στο χώρο καύσης. Η θερμοκρασία στο χώρο εισαγωγής του κινητήρα , εντολή 09 , δεν είναι διαθέσιμη. Αντίθετα όταν ο εξομοιωτής εκτελείται σε λειτουργία τυχαίων τιμών είναι διαθέσιμο το σύνολο των εντολών του Mode 01.

3.5.1 Επεξεργασία των απαντήσεων του ODBII

Προκειμένου να αντλήσουμε από το OBD II την ταχύτητα του οχήματος γράφουμε στο ρεύμα εξόδου την εντολή 01 0D. Το σύστημα απάντα για παράδειγμα 41 0D 2F και τερματίζει με το χαρακτήρα '>'. Από τον πίνακα 2 γνωρίζουμε ότι η απάντηση είναι ένα byte (είδαμε πριν ότι τα πρώτα δύο αποτελούν το Mode και την ερώτηση αντίστοιχα), οπότε η ταχύτητα είναι το 2F. Μετατρέποντας στο δεκαδικό σύστημα παίρνουμε την τιμή 47 η οποία αποτελεί την ταχύτητα μας σε χιλιόμετρα ανά ώρα.

Ομοίως από τον πίνακα υπολογίζουμε και την ποσότητα του αέρα που εισέρχεται στο χώρο καύσης. Με την εντολή 01 10 το σύστημα απαντά για παράδειγμα: 41 10 A3 12. Το byte A είναι το A3 ενώ το B είναι το 12. Κάνοντας τις απαραίτητες μετατροπές και εφαρμόζοντας τον τύπο του πίνακα καταλήγουμε στο αποτέλεσμα 417.46 το οποίο αντιστοιχεί στην ποσότητα αέρα, σε γραμμάρια ανά δευτερόλεπτο, που εισέρχονται στο χώρο καύσης.

PID (hex)	Data bytes returned	Description	Min value	Max value	Units	Formula[a]
0C	2	Engine RPM	0	16,383.75	rpm	((A*256)+B)/4
0D	1	Vehicle speed	0	255	km/h	A
0E	1	Timing advance	-64	63.5	° relative to #1 cylinder	(A-128)/2
0F	1	Intake air temperature	-40	215	°C	A-40
10	2	MAF air flow rate	0	655.35	grams/sec	((A*256)+B) / 100

Πίνακας 2: Τμήμα των PID's του Mode 01. Πηγή Wikipedia.

Στο σημείο αυτό θα πρέπει να αναφερθούμε στο τρόπο που ο ELM327 χειρίζεται τις απαντήσεις του OBD II. Ο ELM327 όταν λαμβάνει μια απάντηση δεν την προωθεί αμέσως αλλά περιμένει 200ms για να πάρει και άλλες τιμές από άλλους αισθητήρες. Αυτό συμβαίνει για το λόγο ότι μια τιμή δύναται να την παρέχουν περισσότεροι από ένας αισθητήρες του οχήματος και έτσι ο ELM327 αναμένει το προαναφερθέν χρονικό διάστημα προκειμένου να λάβει και τις υπόλοιπες απαντήσεις προτού προωθήσει τις απαντήσεις. Αποτέλεσμα αυτού είναι να παρατηρείται μια καθυστέρηση στην επικοινωνία με το όχημα. Για να αντιμετωπιστεί το πρόβλημα αυτό ο κατασκευαστής του ELM327 δίνει τη δυνατότητα στο χειριστή να προσθέτει ένα χαρακτήρα μετά το τέλος της κάθε εντολής που προορίζεται στο OBDII. Ο χαρακτήρας αυτός καθορίζει πόσες απαντήσεις θα περιμένει ο ELM327 προτού προωθήσει την ζητούμενη τιμή. Έτσι τοποθετώντας το χαρακτήρα 1 μετά την εντολή ο ELM327 θα προωθήσει την πρώτη απάντηση που θα έχει διαθέσιμη μειώνοντας με τον τρόπο αυτό δραστικά το χρόνο επικοινωνίας με το όχημα.

3.5.2 Υπολογισμός Της Κατανάλωσης Καυσίμου.

Από τη ποσότητα του αέρα που εισέρχεται στο χώρο καύσης μπορούμε να υπολογίσουμε την κατανάλωση καυσίμου σε μίλια ανά γαλόνι ακολουθώντας τη μέθοδο του Bruce D. Lighter[7]. ως ακολούθως:

$$MPG = 7{,}107 \times \frac{VSS}{MAF}$$

όπου VSS η ταχύτητα του οχήματος και MAF η ποσότητα του αέρα. Η τιμή 7,107 αναφέρετε σε καύσιμο τύπου βενζίνη.

Η μέθοδος βασίζεται στο γεγονός ότι η ποσότητα καυσίμου είναι σταθερή (ανά είδος καυσίμου) ανά ποσότητα αέρα και συγκεκριμένα 14.7 γραμμάρια αέρα ανά 1 γραμμάριο βενζίνης. Μεγαλύτερη ποσότητα καυσίμου θα μειώσει τη ζωή του κινητήρα ενώ μικρότερη θα μειώσει την απόδοση του. Για το πετρέλαιο η σχέση είναι 14.6 γραμμάρια αέρα ανά 1 γραμμάριο πετρελαίου. Γνωρίζοντας λοιπόν το είδος του καυσίμου μπορούμε να έχουμε μια αρκετά ακριβή εκτίμηση της κατανάλωσης τη κάθε χρονική στιγμή. Για παράδειγμα αν λάβουμε τις προηγούμενες τιμές (ταχύτητα VSS = 47, ποσότητα αέρα MAF = 417.46) παίρνουμε την τιμή 0,800 μίλια ανά γαλόνι η σε λίτρα ανά 100 χιλιόμετρα : 235.214 / mpg = 294.01 l/100km.

4. Ανασκόπηση Πεδίου

4.1 Εισαγωγή

Στις μέρες μας στο εμπόριο κυκλοφορεί πληθώρα εφαρμογών που κινούνται στο συγκεκριμένο πεδίο. Από εφαρμογές υψηλού κόστους διαχείρισης στόλου οχημάτων μέχρι και δωρεάν εφαρμογές καταγραφής θέσης ή εμφάνισης στην οθόνη του κινητού δεδομένων του κινητήρα. Παραθέτουμε κάποιες που ξεχωρίσαμε ξεκινώντας από αυτή που εμείς χρησιμοποιήσαμε στα πλαίσια εκπόνησης της παρούσας διατριβής.

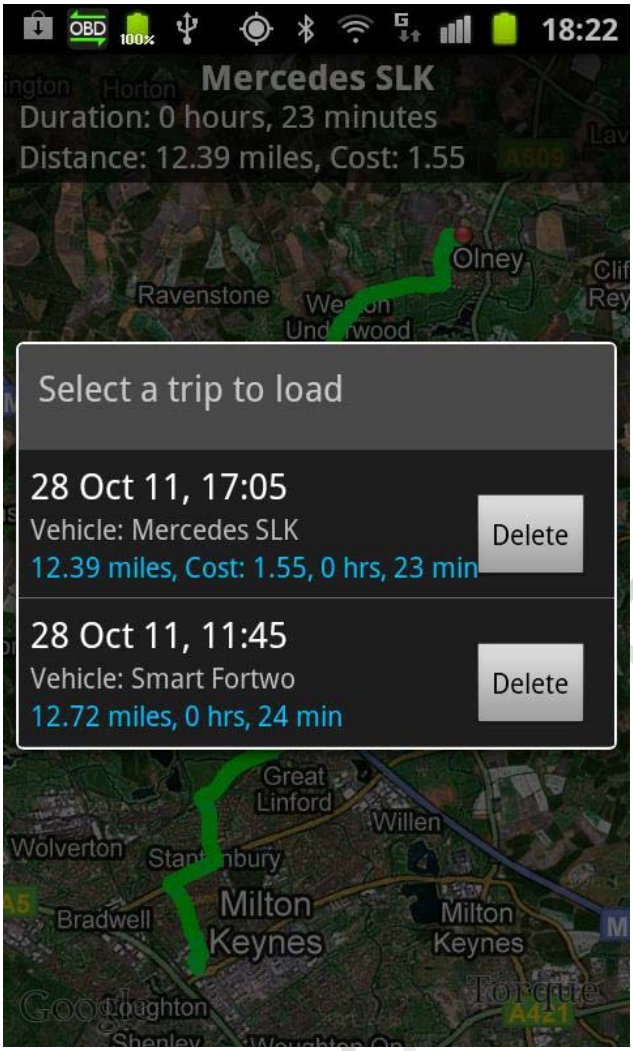
4.2 Η Εφαρμογή Torque.

Η αναφορά μας ξεκινά με το λογισμικό Torque λόγω του ότι το συγκεκριμένο λογισμικό προσφέρει λειτουργίες αντίστοιχες με αυτές τις δικής μας εφαρμογής. Η εφαρμογή αυτή προορίζεται για συσκευές που εκτελούν λειτουργικό σύστημα Android, είναι διαθέσιμη στο Google Play, και προσφέρετε σε δύο εκδόσεις, την δωρεάν και την επί πληρωμή.

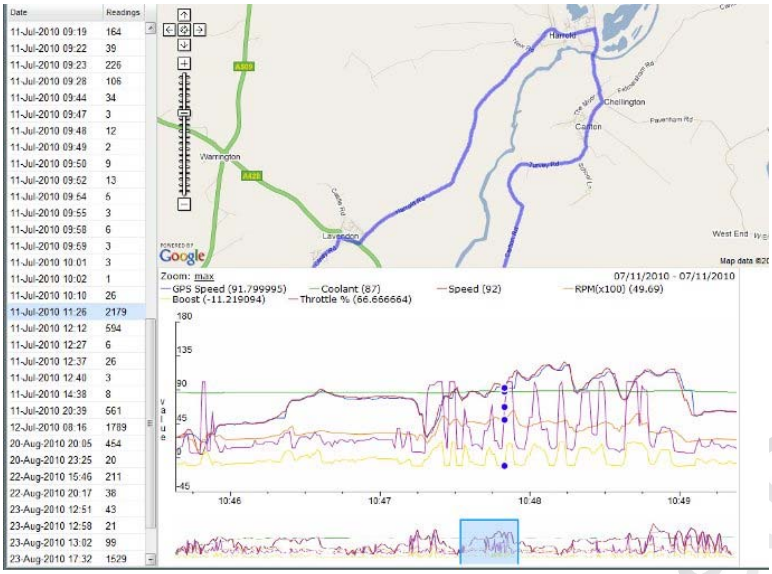
Στα πλαίσια εκπόνησης της συγκεκριμένης πτυχιακής χρησιμοποιήσαμε τη δωρεάν έκδοση της εφαρμογής ώστε να εκτιμήσουμε τα δεδομένα που μας έστελνε το δικό μας ενσωματωμένο σύστημα διάγνωσης. Η εφαρμογή συνδέετε με το σύστημα μέσω Bluetooth και αντλεί από αυτό τα δεδομένα που επιλέγει ο χρήστης και κατόπιν τα εμφανίζει στην οθόνη της έξυπνης συσκευής. Στην Pro έκδοση, με κόστος 3,55 € παρέχει στο χρήστη, μεταξύ άλλων, τη δυνατότητα να εμφανίζει στο χάρτη της συσκευής τις διαδρομές του με διάφορους τρόπους, όπως για παράδειγμα διαφορετικό χρώμα στο τμήμα της διαδρομής με διαφορετική ταχύτητα, καθώς και στατιστικά δεδομένα των διαδρομών. Άξια αναφοράς είναι και η δυνατότητα που δίδετε στο χρήστη να ανεβάζει τα δεδομένα του σε εξυπηρετητή ιστοσελίδων ώστε αυτά να είναι διαθέσιμα μέσω Internet από οπουδήποτε.

Εικόνα 27: Το βασικό μενού της εφαρμογής Torque Pro. Πηγή : Google Play.

Εικόνα 28: Τρέχοντα δεδομένα της εφαρμογής Torque Pro. Πηγή : Google Play.



Εικόνα 29: Εμφάνιση διαδρομών αλλά και στατιστικών του χρήστη. Αξίζει να αναφερθεί η υποστήριξη πολλαπλών οχημάτων. Πηγή : Google Play.



Εικόνα 30: Εμφάνιση δεδομένων μέσω Web. Πηγή http://torque-bhp.com/

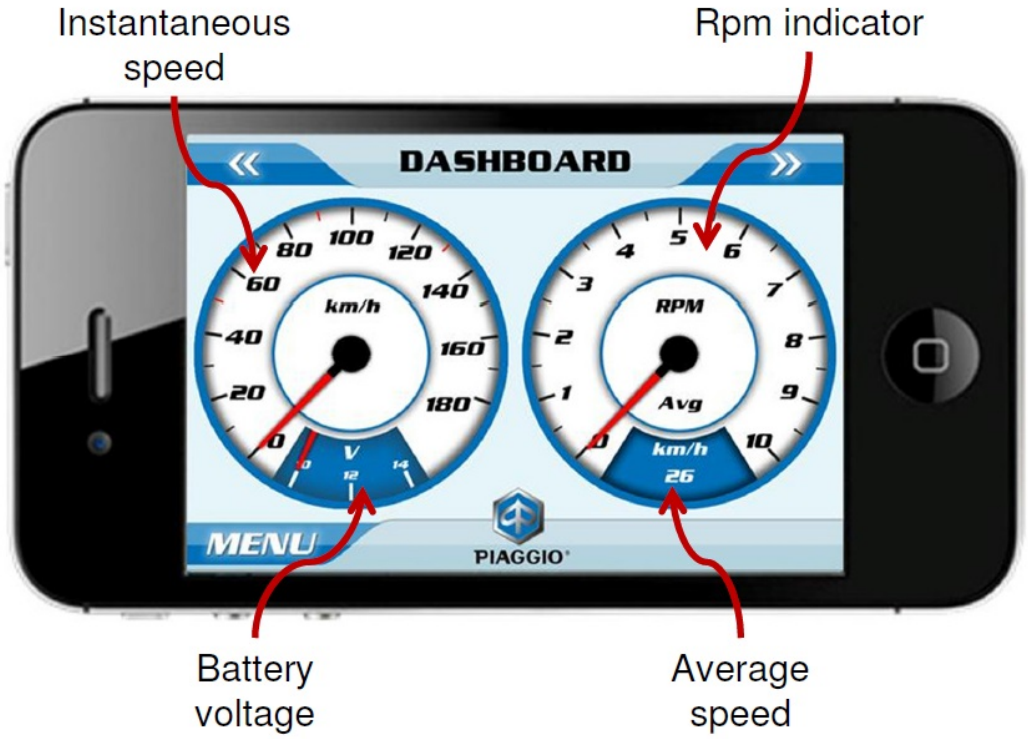
Η συγκεκριμένη εφαρμογή για να έχει πρόσβαση στα δεδομένα του οχήματος θα πρέπει ο χρήστης να εγκαταστήσει στο όχημα του τη συσκευή που επικοινωνεί με το ενσωματωμένο σύστημα διάγνωσης και επιτρέπει στο έξυπνο κινητό τηλέφωνο να αντλεί δεδομένα από το όχημα.

4.3 Λογισμικό Από Εταιρίες Κατασκευής Οχημάτων

Το διάστημα που συγγράφονταν η παρούσα πτυχιακή δύο εταιρίες κατασκευής οχημάτων προσέφεραν λογισμικό που αντλεί δεδομένα από το όχημα, η Opel και η Piaggio. Θα αναφερθούμε σε αυτό της Piaggio λόγω του ότι αυτό προσφέρει κάποιες λειτουργίες μέσω χάρτη, ενώ η Opel περιορίζεται στο να εμφανίζει δεδομένα του οχήματος στο χρήστη. Το λογισμικό της Piaggio παρέχετε δωρεάν στους χρήστες συσκευών IPhone και είναι συμβατό με τα μοντέλα X10 125cc έως X10 500cc. Τα μοντέλα αυτά είναι εξοπλισμένα με μια συσκευή η οποία επικοινωνεί με το ενσωματωμένο σύστημα διάγνωσης της μοτοσυκλέτας και διαθέτει Bluetooth διασύνδεση. Ο χρήστης που διαθέτει συμβατό IPhone μπορεί να συνδεθεί μέσω της Bluetooth θύρας με τη μοτοσυκλέτα και να αντλεί τα δεδομένα που τον ενδιαφέρουν.

Τα δεδομένα της μοτοσυκλέτας συνδυάζονται με δεδομένα αισθητήρων από το κινητό τηλέφωνο και εμφανίζονται στην οθόνη μέσω των διαφόρων λειτουργιών της εφαρμογής. Με τον τρόπο αυτό ο χρήστης μπορεί να διαβάζει τα τρέχοντα δεδομένα, (ταχύτητα, κατανάλωση, κλήση της μοτοσυκλέτας και άλλα) η να έχει και στατιστικά διαδρομής.

Εικόνα 31: Το βασικό μενού της εφαρμογής PIAGGIO MULTIMEDIA PLATFORM. Πηγή User Manual.



Εικόνα 32: Τρέχοντα δεδομένα της μοτοσυκλέτας. Πηγή User Manual.

Εικόνα 33: Στατιστικά διαδρομής. Πηγή User Manual.

Εικόνα 34: Εμφάνιση της τελευταίας θέσης της μοτοσυκλέτας στο χάρτη. Πηγή User Manual.

Το λογισμικό αυτό δεν έχει τη δυνατότητα να εμφανίζει στο χάρτη τη τρέχουσα θέση της μοτοσυκλέτας όταν αυτή βρίσκεται σε κίνηση. Η δυνατότητα αυτή παρέχετε μόνο όταν η μοτοσυκλέτα είναι ακινητοποιημένη ώστε να μπορεί να την εντοπίσει ο ιδιοκτήτης της. Επίσης υπάρχει η δυνατότητα καταγραφής θέσης σε αρχείο ώστε να είναι διαχειρίσιμο από υπολογιστή. Επιπλέον δεν υπάρχει η δυνατότητα εμφάνισης σε χάρτη των διαδρομών που έχει κάνει ο χρήστης.

4.4 Πακέτα Παρακολούθησης Στόλου Οχημάτων.

Κλείνοντας την αναφορά μας θα αναφερθούμε στα διάφορα πακέτα παρακολούθησης στόλου οχημάτων που διατίθενται στο εμπόριο. Πάροχοι υπηρεσιών κινητής τηλεφωνίας όπως η Cosmote[8] και η Wind[9] καθώς και εταιρίες που ειδικεύονται στη παρακολούθηση οχημάτων προσφέρουν την υπηρεσία εντοπισμού και καταγραφής θέσης οχημάτων είτε συνδρομητικά είτε με τη μορφή πακέτου λογισμικού. Οι εταιρίες αυτές εγκαθιστούν στο όχημα εξοπλισμό ο οποίος επικοινωνεί με το ενσωματωμένο σύστημα διάγνωσης, διαθέτει GPS αισθητήρα και αποστέλλει μέσω τηλεφώνου η ιντερνέτ πληροφορίες σε ένα κεντρικό εξυπηρετητή. Ο χρήστης τη υπηρεσίας μπορεί να συνδεθεί μέσω ίντερνετ στο κεντρικό εξυπηρετητή και να παρακολουθεί τη θέση του οχήματος που τον ενδιαφέρει μέσω ιστοσελίδας.

Οι εταιρίες αυτές παρέχουν την υπηρεσία για όλους του τύπους οχημάτων αλλά και σκαφών. Άλλες μπορούν να παρακολουθούν τη θέση ανθρώπων η και ζώων όπως η GPS Locate[10]. Με τον τρόπο αυτό ο χρήστης μπορεί να γνωρίζει τη θέση του παιδιού η του ηλικιωμένου ατόμου η κάποιου ζώου και μπορεί να το εντοπίσει όταν αυτό έχει χαθεί.

Τα χαρακτηριστικά των πακέτων ποικίλουν όπως επίσης ποικίλουν και οι εταιρίες που προσφέρουν αυτού του είδους τις υπηρεσίες. Κάποιες εφοδιάζουν τους χρήστες με έξυπνες συσκευές οι οποίες προσφέρουν αυξημένες δυνατότητες ενώ άλλες περιορίζονται στη χρήση τερματικών που απλά μεταφέρουν πληροφορίες στο κεντρικό εξυπηρετητή καθιστώντας έτσι την πληροφορία αξιοποιήσιμη μόνο μέσω αυτού του κεντρικού εξυπηρετητή.

5. ΑΝΑΦΟΡΕΣ – LINKS

1. Apache, URL : http://www.apache.org/licenses/LICENSE-2.0
2. Genymotion Android Emulator, URL: https://www.genymotion.com/#!/
3. OBD II, URL : http://www.obdii.com/
4. Elm Electronics, URL : http://elmelectronics.com/index.html
5. ELM327, URL : http://elmelectronics.com/obdic.html
6. OBD Simulator, URL : http://icculus.org/obdgpslogger/obdsim.html
7. Bruce D. Lighter, AVR-Based Fuel Consumption Gauge. URL : http://www.lightner.net/lightner/bruce/Lightner-183.pdf
8. COSMOTE Fleet Management, URL : http://www.cosmote.com/cosmoportal/page/T21/xml/Business__service__service_Cosmote_fleet_management_bus/section/Machine_to_Machine#
9. Wind Fleet Management, URL : http://www.wind.gr/gr/gia-tin-epiheirisi/exupnoi-sunduasmoi/wind-business-cloud-services/upiresia-diaheirisis-stolou-ohimaton-wind-fleet-/
10. GPS Locate Fleet Management, URL : http://www.gpslocate.gr/index.aspx