ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΑ
ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Υλοποίηση εφαρμογής κινητού τηλεφώνου με χρήση της πλατφόρμας Google Android

Εμμανουήλ Α. Γιακουμέλης

Επιβλέπων: Χρήστος Ξενάκης, Λέκτορας Πανεπιστημίου Πειραιά

ΑΘΗΝΑ
ΝΟΕΜΒΡΙΟΣ 2009

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Υλοποίηση εφαρμογής κινητού τηλεφώνου με χρήση της πλατφόρμας Google Android

Εμμανουήλ Α. Γιακουμέλης

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ:

Χρήστος Ξενάκης, Λέκτορας Πανεπιστημίου Πειραιά

ΠΕΡΙΛΗΨΗ

Το Google Android είναι η πρώτη πλήρης ανοιχτού λογισμικού πλατφόρμα ανάπτυξης εφαρμογών που προορίζονται για την εγκατάσταση σε κινητά τηλέφωνα. Οι εφαρμογές που αναπτύσσονται με το Google Android επιτρέπουν την πρόσβαση σε βασικές λειτουργίες των κινητών συσκευών μέσω υπαρχόντων APIs. Αποτελεί ένα πλήρες περιβάλλον όπου οι εφαρμογές μπορούν να έχουν πολλαπλές διεργασίες που τρέχουν παράλληλα και μπορούν εύκολα να ενσωματώσουν HTML, Javascript και style sheet. Σκοπός της συγκεκριμένης διπλωματικής εργασίας είναι η παρουσίαση της πλατφόρμας Google Android και η ανάπτυξη μίας εφαρμογής μέσα από την οποία θα δούμε να εφαρμόζονται πολλά από τα βασικά χαρακτηριστικά του Android OS. Η εφαρμογή που υλοποιήθηκε στα πλαίσια της διπλωματικής αυτής εργασίας λειτουργεί ως εφαρμογή κινητού τηλεφώνου που δίνει στους χρήστες δυνατότητες να διαβάσουν και να τροποποιήσουν δεδομένα που βρίσκονται διαθέσιμα σε ένα κεντρικό υπολογιστή. Τέλος, η εργασία αυτή εστίασε και μελέτησε τις βασικές δυνατότητες και τους βασικούς περιορισμούς σε θέματα ασφάλειας του λειτουργικού συστήματος Android.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: εφαρμογές με χρήση της πλατφόρμας Google Android

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Google Android, Android Security

Στους γονείς μου Μαρία και Αναστάσιος

και την αδερφή μου Ευαγγελία

για την αγάπη και

την υποστήριξη τους

ΠΡΟΛΟΓΟΣ

Η παρούσα εργασία γράφτηκε από τον φοιτητή Εμμανουήλ Γιακουμέλη στα πλαίσια του μεταπτυχιακού κύκλου σπουδών του στο τμήμα ψηφιακών συστημάτων του Πανεπιστημίου Πειραιά, υπό την εποπτεία του Λέκτορα, κύριου Χρήστου Ξενάκη. Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κύριο Χρήστο Ξενάκη για τις πολύτιμες συμβουλές και οδηγίες του, για το ενδιαφέρον του, την φιλική συμπεριφορά του και την καθοδήγηση του σε όλη την διάρκεια της συγγραφής της συγκεκριμένης διπλωματικής εργασίας. Επιπλέον, θα ήθελα να ευχαριστήσω την οικογένεια μου για την συμπαράσταση και την στήριξη τους σε αυτή μου την προσπάθεια.

Πίνακας περιεχομένων

Πίνακας σχημάτων

1.Λογισμικό Android

1.1.Γενικά στοιχεία για το Android

Το Android αποτελεί μια στοίβα λογισμικού για κινητές συσκευές η οποία περιλαμβάνει ένα λειτουργικό σύστημα και βασικές εφαρμογές. Εφαρμογές για αυτήν την πλατφόρμα μπορούν να αναπτυχθούν χρησιμοποιώντας το Android SDK και την γλώσσα προγραμματισμού Java. Οι εφαρμογές αυτές τρέχουν στην Dalvik Virtual Machine, μια τροποποιημένη Virtual Machine σχεδιασμένη για χρήση σε ενσωματωμένα συστήματα η οποία τρέχει πάνω σε έναν πυρήνα Linux έκδοση 2.6.

Το Android SDK παρέχει APIs για την χρήση web browser, εμφάνιση δισδιάστατων και τρισδιάστατων γραφικών, δομημένη αποθήκευση δεδομένων σε βάση δεδομένων, εμφάνιση πολυμεσικού υλικού (ήχος, βίντεο, εικόνες), χρήση των τεχνολογιών GSM, Bluetooth, EDGE, 3G και WiFi, χρήση συσκευών όπως φωτογραφική μηχανή, GPS κ.α.

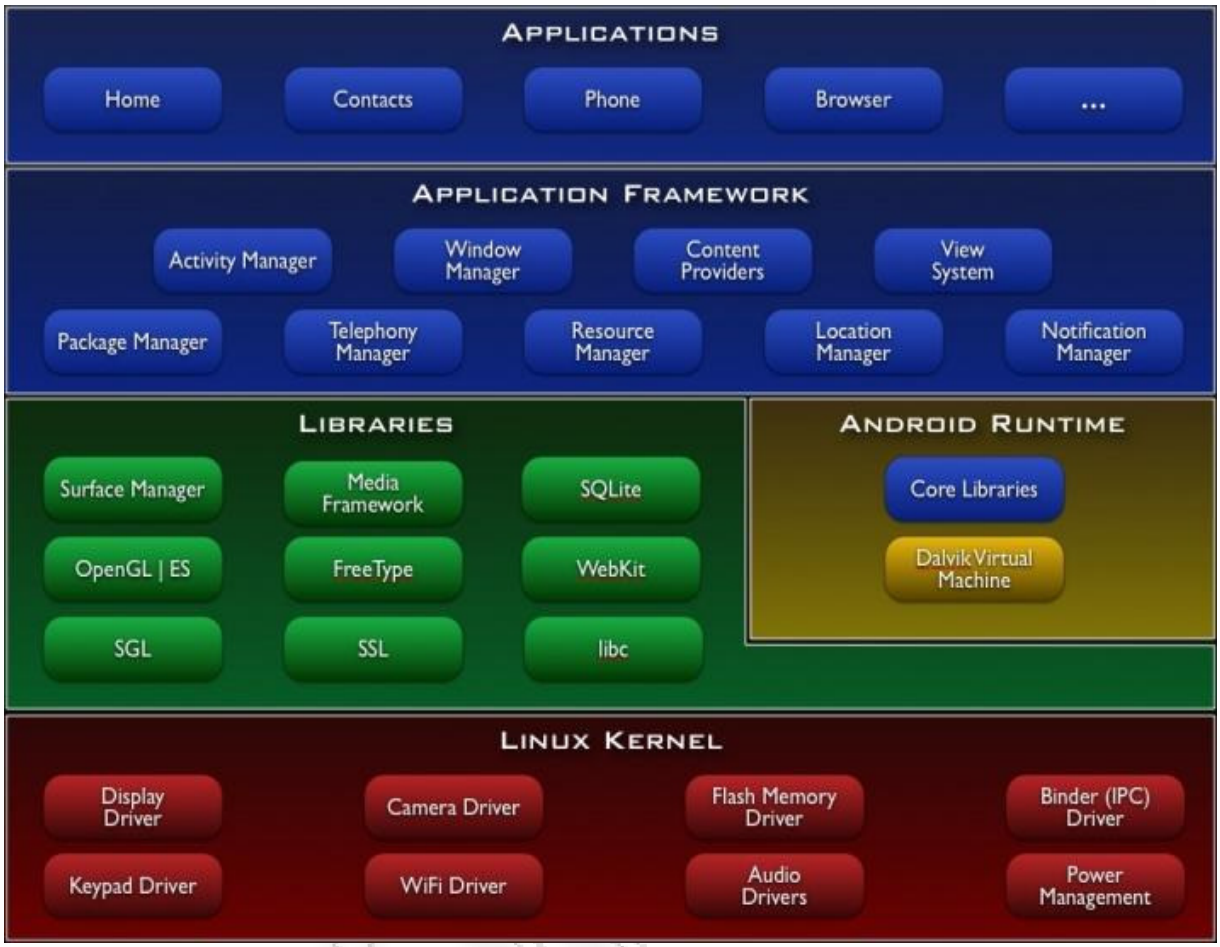
1.2.Αρχιτεκτονική του Android

Η πλατφόρμα του Android αποτελείται από μια στοίβα λογισμικού. Τα επίπεδα της στοίβας του Android από το υψηλότερο προς το χαμηλότερο περιγράφονται παρακάτω:

- Επίπεδο Εφαρμογών (Blue layer - Application). Στο επίπεδο αυτό περιλαμβάνεται ένα σύνολο από βασικές εφαρμογές μερικές από τις οποίες είναι e-mail client, πρόγραμμα SMS, ημερολόγιο, χάρτες, browser, επαφές κ.α. Όλες οι εφαρμογές είναι γραμμένες με χρήση της γλώσσας προγραμματισμού Java.
- Επίπεδο Πλαισίου Εφαρμογών (Blue layer - Application Framework). Βρίσκεται κάτω από το επίπεδο Εφαρμογών και αποτελείται ένα σύνολο συστημάτων και υπηρεσιών:

- Ένα σύνολο από γραφικά στοιχεία (Views) για την δημιουργία γραφικού περιβάλλοντος συμπεριλαμβανομένων λιστών (lists), πλεγμάτων (grids), κουτιών κειμένου (text boxes), κουμπιών (buttons) κ.α.
- Ένα διαχειριστή περιεχομένου (Content Manager) ο οποίος επιτρέπει στις εφαρμογές την πρόσβαση σε δεδομένα άλλων εφαρμογών ή τον διαμοιρασμό των δικών τους δεδομένων με άλλες εφαρμογές.
- Ένα διαχειριστή πόρων (Resource Manager) για την πρόσβαση στους πόρους όπως strings, εικόνες, layout files.
- Έναν διαχειριστή ειδοποιήσεων (Notification Manager) ο οποίος επιτρέπει την προβολή ειδοποιήσεων στην μπάρα κατάστασης (status bar).
- Έναν διαχειριστή δραστηριοτήτων (Activity Manager) ο οποίος διαχειρίζεται τον κύκλο ζωής των εφαρμογών.

- Επίπεδο Βιβλιοθηκών (Green layer - Libraries). Το οποίο περιλαμβάνει ένα σύνολο από βιβλιοθήκες γραμμένες σε C/C++ οι οποίες χρησιμοποιούνται από διάφορα στοιχεία του συστήματος του Android. Οι δυνατότητες που προσφέρουν αυτές οι βιβλιοθήκες είναι προσβάσιμες στους προγραμματιστές δια μέσου του επιπέδου πλαισίου εφαρμογής.
- Επίπεδο Εκτέλεσης (Green layer - Android Runtime). Το οποίο αποτελείται από ένα σύνολο από βασικές βιβλιοθήκες και την Dalvik Virtual Machine.
- Ο Πυρήνας του Linux (Red Layer). Το Android βασίζεται στον πυρήνα Linux έκδοση 2.6 για βασικές υπηρεσίες συστήματος όπως ασφάλεια, διαχείριση μνήμης, διαχείριση διεργασιών, στοίβα δικτύου, και οδηγούς συσκευών. Ο πυρήνας λειτουργεί επίσης ως ένα ενδιάμεσο επίπεδο αφαίρεσης μεταξύ της στοίβας λογισμικού και του υλικού.



Σχημα 1 : Αρχιτεκτονική Android

1.3.Εργαλεία ανάπτυξης εφαρμογών

1.3.1.Περιβάλλοντα ανάπτυξης εφαρμογών

Η ανάπτυξη και αποσφαλμάτωση των εφαρμογών μπορούν να γίνουν με την χρήση του ολοκληρωμένου περιβάλλοντος ανάπτυξης εφαρμογών Eclipse και του Android Development Tools Plug-in το οποίο δίνει την δυνατότητα πρόσβασης στα εργαλεία ανάπτυξης του Android SDK μέσα από το περιβάλλον του Eclipse.

Εναλλακτικά η ανάπτυξη και αποσφαλμάτωση των εφαρμογών μπορούν να γίνουν με χρήση κάποιου άλλου περιβάλλοντος ανάπτυξης εφαρμογών και των εργαλείων που παρέχει το Android SDK.

1.3.2.Ο εξομοιωτής

Προκειμένου να γίνει ευκολότερη η διαδικασία της ανάπτυξης και αποσφαλμάτωσης μιας εφαρμογής το Android SDK περιλαμβάνει έναν εξομοιωτή μιας εικονικής κινητής συσκευής η οποία τρέχει το λειτουργικό του Android. Έτσι δεν είναι η αναγκαία η ύπαρξη πραγματικής κινητής συσκευής για την εκτέλεση και δοκιμή των εφαρμογών. Ο εξομοιωτής προσομοιώνει ένα μεγάλο πλήθος λειτουργιών μιας τυπικής συσκευής η οποία τρέχει το Android:

- Παρέχει μια ποικιλία πλήκτρων πλοήγησης και ελέγχου.
- Παρέχει μια οθόνη για την προβολή των εφαρμογών που τρέχουν στον εξομοιωτή.
- Επιτρέπει στις εφαρμογές την χρήση των υπηρεσιών που προσφέρει η πλατφόρμα του Android δηλαδή την κλήση άλλων εφαρμογών, την πρόσβαση στο δίκτυο, την αναπαραγωγή ήχου και βίντεο, την αποθήκευση και επαναφορά δεδομένων, την ειδοποίηση χρήστη, το γραφικό περιβάλλον του Android.

Επίσης παρέχει ένα πλήθος λειτουργιών για την ευκολότερη αποσφαλμάτωση:

- Κονσόλα για την καταγραφή της εξόδου του πυρήνα.
- Προσομοίωση διακοπών (όπως η άφιξη SMS μηνύματος ή τηλεφωνικής κλήσης).
- Προσομοίωση καθυστέρησης και απώλειας στο κανάλι δεδομένων.
- Προσομοίωση λήψης δεδομένων θέσης από την συσκευή GPS.

1.3.3.Άλλα εργαλεία του Android

Το Android SDK περιλαμβάνει μερικά ακόμη εργαλεία για την ανάπτυξη εφαρμογών:

- Το Dalvik Debug Monitor Service (DDMS) το οποίο επιτρέπει την διαχείριση των διεργασιών στον εξομοιωτή ή στην συσκευή. Συγκεκριμένα δίνεται η δυνατότητα port-forwarding υπηρεσιών, λήψη screenshots, εμφάνιση πληροφοριών για τον σωρό και τα νήματα, logcat, εμφάνιση πληροφοριών ράδιο και πληροφοριών διεργασιών, προσομοίωση εισερχόμενων κλήσεων και μηνυμάτων, προσομοίωση δεδομένων θέσης κ.α.
- Την Android Debug Bridge (adb) η οποία επιτρέπει την διαχείριση της κατάστασης του εξομοιωτή ή της συσκευής. Μέσω του adb είναι δυνατή η

εκτέλεση εντολών φλοιού, η διαχείριση της προώθησης θυρών και η αντιγραφή από και προς την συσκευή ή τον εξομοιωτή.

- Το Android Asset Packaging Tool (aapt) το οποίο δίνει την δυνατότητα δημιουργίας .apk αρχείων τα οποία περιέχουν τα εκτελέσιμα και τους πόρους μιας εφαρμογής.
- Την Android Interface Description Language (aidl) η οποία επιτρέπει την δημιουργία κώδικα που επιτρέπει σε δύο διεργασίες σε μια συσκευή βασισμένη στο Android να συνομιλούν χρησιμοποιώντας διαδιεργασιακή επικοινωνία.
- Το sqlite3 το οποίο επιτρέπει την πρόσβαση στα δεδομένα της SQLite που δημιουργούνται από τις διάφορες εφαρμογές.
- Το Traceview το οποίο επιτρέπει την γραφική προβολή της ανάλυσης των trace log data που δημιουργούν οι διάφορες εφαρμογές.
- Το mksdcard το οποίο βοηθά στην δημιουργία εικονικού δίσκου ο οποίος μπορεί να χρησιμοποιηθεί από τον εξομοιωτή για την προσομοίωση της παρουσίας εξωτερικής αποθηκευτικής κάρτας (όπως η SD card).
- Το dx tool το οποίο μετατρέπει τα αρχεία .class από java bytecode σε Android bytecode.
- Το UI/Application Exerciser Monkey το οποίο είναι ένα πρόγραμμα που τρέχει στον εξομοιωτή και παράγει ψευδό-τυχαίες σειρές από συμβάντα χρήστη όπως clicks, touches, gestures καθώς επίσης και έναν αριθμό από συμβάντα συστήματος.
- Το activitycreator το οποίο είναι ένα script που δημιουργεί Ant build αρχεία τα οποία μπορούν να χρησιμοποιηθούν για την μεταγλώττιση των εφαρμογών.

1.4.Μοντέλο προγραμματισμού

1.4.1.Μοντέλο εφαρμογών

Στα περισσότερα συστήματα υπάρχει μια στενή συσχέτιση μεταξύ του εκτελέσιμου αρχείου στο οποίο υπάρχει μια εφαρμογή, της διεργασίας στην οποία τρέχει και στο περιβάλλον μέσα στο οποίο ο χρήστης αλληλεπιδρά με αυτήν. Στο Android αυτή η συσχέτιση είναι πιο χαλαρή λόγω της ευέλικτης

φύσης των εφαρμογών οι οποίες είναι γραμμένες για αυτό. Για μια εφαρμογή Android γίνεται ο εξής διαχωρισμός:

- Ένα android package (.apk) είναι το αρχείο το οποίο περιέχει τον εκτελέσιμο κώδικα και τους πόρους μιας εφαρμογής. Είναι το αρχείο που διαμοιράζεται και χρησιμοποιούν οι χρήστες προκειμένου να εγκαταστήσουν την εφαρμογή στην συσκευή τους.
- Ένα task είναι αυτό που αντιλαμβάνεται ο χρήστης σαν μια εφαρμογή η οποία μπορεί να εκκινήσει. Συνήθως ένα task έχει το δικό του εικονίδιο μέσω του οποίου ο χρήστης έχει πρόσβαση σε αυτό και είναι προσβάσιμο ως αντικείμενο του πιο υψηλού επιπέδου που μπορεί έρθει στο προσκήνιο μπροστά από άλλα tasks.
- Μια process είναι μια χαμηλού επιπέδου διεργασία του πυρήνα στην οποία τρέχει ο κώδικας της εφαρμογής. Συνήθως όλος ο κώδικας που περιέχεται σε ένα .apk εκτελείται μέσα σε μια διεργασία, αφιερωμένη για το συγκεκριμένο .apk. Παρόλα αυτά η ετικέτα διεργασίας μπορεί να χρησιμοποιηθεί για να τροποποιηθεί το που θα εκτελεστεί ο κώδικας είτε για ολόκληρο το .apk είτε για ένα μόνο στοιχείο του package.

1.4.2.Δομικά στοιχεία μιας εφαρμογής

Τα βασικά δομικά στοιχεία μιας εφαρμογής Android:

1. Δραστηριότητα (Activity)
2. Δέκτης Εκπεμπόμενων Προθέσεων (Broadcast Intent Receiver)
3. Υπηρεσία (Service)
4. Πάροχος Περιεχομένου (Content Provider)
5. Πρόθεση και Φίλτρο Προθέσεως (Intent και Intent Filter)
6. Ειδοποίηση (Notification)
7. Όψη (View)
8. AndroidManifest.xml

Δεν είναι απαραίτητο κάθε εφαρμογή να περιέχει όλα αυτά στοιχεία, αλλά μια εφαρμογή είναι ένας συνδυασμός μερικών από αυτά. Παρακάτω γίνεται μια σύντομη περιγραφή του κάθε ενός.

1.4.2.1.Δραστηριότητα

Η Δραστηριότητα (Activity) αποτελεί το πιο κοινό από τα δομικά στοιχεία μιας εφαρ μογής. Συνήθως είναι μια μόνο ξεχωριστή οθόνη σε μια εφαρμογή. Κάθε Δραστηριότητα υλοποιείται μέσα σε μια ξεχωριστή κλάση που επεκτείνει την κλάση Activity. Παρουσιάζει την διαπροσωπεία χρήστη (User Interface) και ανταποκρίνεται σε διάφορα συμβάντα.

1.4.2.2.Δέκτης Εκπεμπόμενων Προθέσεων

Ένας Δέκτης Εκπεμπόμενων Προθέσεων (BroadcastReceiver) μπορεί να χρησιμοποιηθεί όταν χρειάζεται να εκτελεστεί κώδικας μιας εφαρμογής ως αποτέλεσμα ενός εξωτερικού συμβάντος. Ο receiver δεν εμφανίζει user interface αν και μπορεί να χρησιμοποιήσει τον Διαχειριστή Ειδοποιήσεων (NotificationManager) για να ειδοποιήσει τον χρήστη. Δεν είναι απαραίτητο μια εφαρμογή η οποία περιέχει έναν Δέκτη Εκπεμπόμενων Προθέσεων να τρέχει για να ενεργοποιηθεί ο receiver. Το σύστημα θα ενεργοποιήσει μια εφαρμογή, αν είναι απαραίτητο, όταν ενεργοποιηθεί κάποιος Δέκτης Εκπεμπόμενων Προθέσεων. Τέλος μια εφαρμογή μπορεί να στείλει τα δικά της intent broadcasts σε άλλες εφαρμογές.

1.4.2.3.Υπηρεσία

Μια Υπηρεσία (Service) είναι κώδικας ο οποίος τρέχει χωρίς user interface. Άλλα στοιχεία μιας εφαρμογής μπορούν να συνδεθούν με μια Υπηρεσία και να επικοινωνήσουν μαζί του διαμέσου μιας διαπροσωπείας που παρέχεται από την Υπηρεσία.

1.4.2.4.Πάροχος Περιεχομένου

Οι εφαρμογές μπορούν να αποθηκεύουν τα δεδομένα τους σε αρχεία, βάσεις δεδομένων SQLite ή με κάποιον άλλον μηχανισμό. Ένας Πάροχος Περιεχομένου (Content Provider) όμως είναι χρήσιμος όταν χρειάζεται μια εφαρμογή να μοιράζεται τα δεδομένα της με άλλες εφαρμογές. Ο Πάροχος Περιεχομένου είναι μια κλάση που υλοποιεί ένα αριθμό από μεθόδους που επιτρέπουν στις εφαρμογές να αποθηκεύουν και να επαναφέρουν δεδομένα του συγκεκριμένου τύπου που χειρίζεται ο Πάροχος Περιεχομένου.

1.4.2.5.Πρόθεση και Φίλτρο Προθέσεως

Ένα αντικείμενο Πρόθεσης (Intent) είναι ένα αντικείμενο το οποίο περιγράφει τι θέλει να κάνει μια εφαρμογή. Τα βασικά στοιχεία μιας Πρόθεσης είναι ποια ενέργεια θέλει η εφαρμογή να εκτελεστεί και τα δεδομένα πάνω στα οποία θα εκτελεστεί η συγκεκριμένη ενέργεια. Ενώ ένα αντικείμενο Πρόθεσης αποτελεί μια πρόθεση να γίνει κάτι ένα αντικείμενο Φίλτρο Προθέσεως (Intent Filter) αποτελεί μια περιγραφή του τι είδους Προθέσεις είναι δυνατόν να εξυπηρετηθούν.

1.4.2.6.Ειδοποίηση

Μια Ειδοποίηση (Notification) αποτελεί ένα μικρό εικονίδιο που εμφανίζεται στην μπάρα καταστάσεων. Ο χρήστης μπορεί να αλληλεπιδράσει με το εικονίδιο αυτό για να λάβει περισσότερες πληροφορίες.

1.4.2.7.Όψη

Η Όψη (View) είναι ένα αντικείμενο το οποίο εμφανίζεται στην οθόνη. Το user interface δημιουργείται με χρήση Όψεων.

1.4.2.8.AndroidManifest.xml

Το AndroidManifest.xml αρχείο είναι το αρχείο ελέγχου το οποίο υπάρχει στον κεντρικό φάκελο κάθε εφαρμογής και μέσα στο οποίο περιγράφονται καθολικές ιδιότητες της εφαρμογής.

1.5.Ο κύκλος ζωής μιας εφαρμογής Android

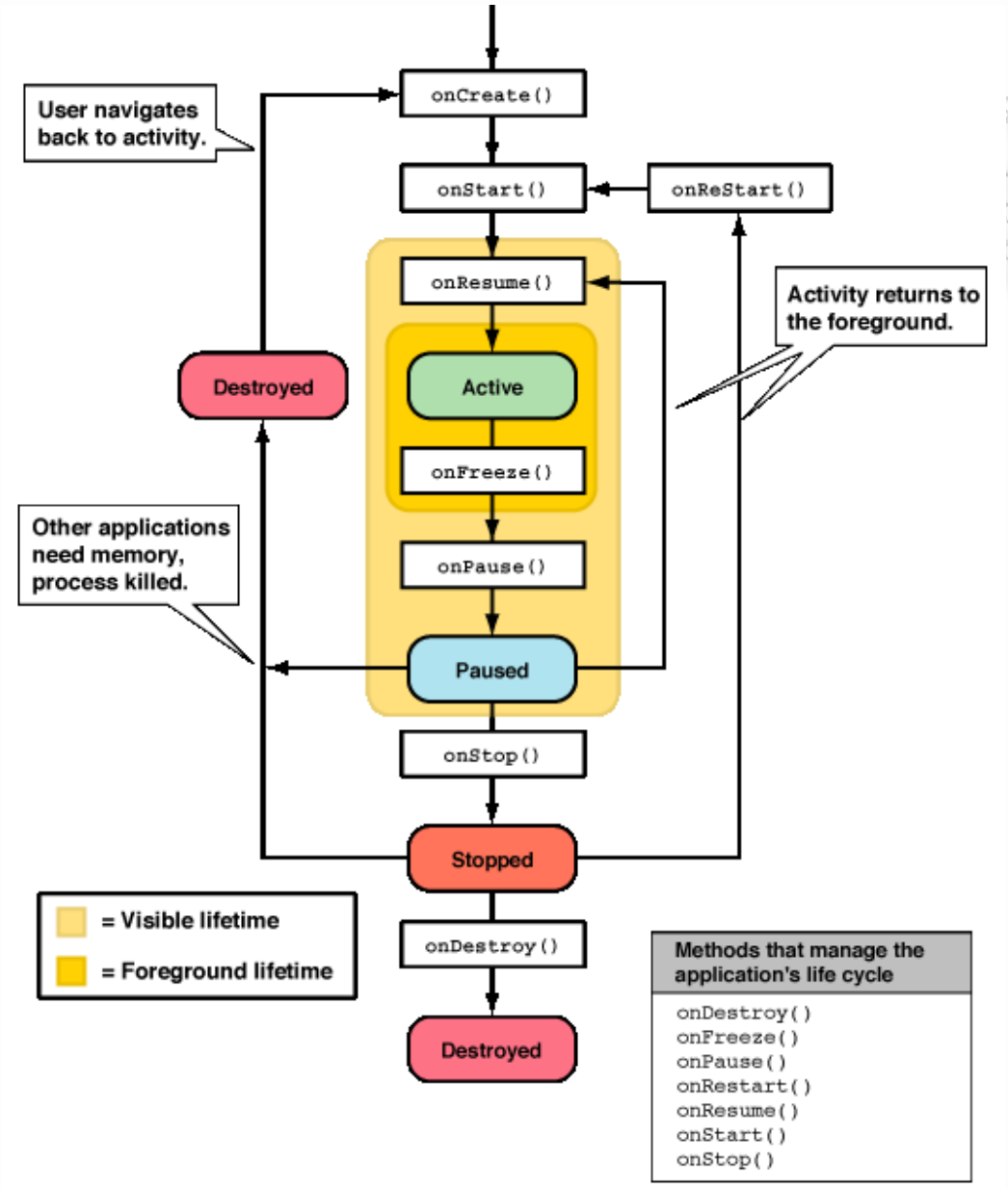
Στις περισσότερες περιπτώσεις, κάθε εφαρμογή Android τρέχει στην δική της ξεχωριστή διεργασία. Η διεργασία αυτή δημιουργείται όταν χρειάζεται να εκτελεστεί κάποιο μέρος του κώδικα της εφαρμογής, συνεχίζει να υπάρχει μέχρι ωσότου δεν είναι χρήσιμη πλέον και το σύστημα χρειάζεται να ελευθερώσει την μνήμη που καταλαμβάνει για χρησιμοποίηση από άλλες εφαρμογές.

Ένα βασικό χαρακτηριστικό του Android είναι ότι μια εφαρμογή δεν ελέγχει άμεσα τον κύκλο ζωής της. Αντιθέτως το σύστημα αποφασίζει για τον κύκλο ζωής μιας εφαρμογής αναλόγως με το ποια μέρη της εφαρμογής τρέχουν, πόσο σημαντικά είναι αυτά για τον χρήστη και πόση είναι η διαθέσιμη μνήμη

του συστήματος.

Η απόφαση του συστήματος για το ποια διεργασία πρέπει να τερματιστεί σε περίπτωση έλλειψης μνήμης βασίζεται σε μια ιεράρχηση των διεργασιών. Τα επίπεδα αυτής της ιεράρχησης είναι τα εξής:

1. Διεργασία Προσκηνίου (foreground process). Μια διεργασία θεωρείται διεργασία προσκηνίου όταν ισχύει κάποια από τις εξής συνθήκες:
    - Εκτελεί μία δραστηριότητα (Activity) με την οποία ο χρήστης αλληλεπιδρά (έχει καλεσθεί η μέθοδος onResume()).
    - Εκτελεί κάποια υπηρεσία (Service) που συνδέεται με την δραστηριότητα (Activity) με την οποία ο χρήστης αλληλεπιδρά.
    - Περιέχει μία υπηρεσία (Service) που καλεί συναρτήσεις δημιουργία, εκκίνησης, καταστροφής (έχει καλέσθει μία από τις μεθόδους: onCreate(), onStart(), onDestroy()).
    - Περιέχει ένα Δέκτη εκπομπών που καλεί συναρτήσεις λήψης (έχει καλεσθεί η μέθοδος onReceive())
2. Ορατή Διεργασία (visible process). Μια διεργασία που περιέχει μια Δραστηριότητα (Activity) που είναι ορατή στην οθόνη αλλά δεν βρίσκεται στο προσκήνιο (έχει καλεσθεί η μέθοδος onPause()).
3. Διεργασία Υπηρεσίας (service process). Μια διεργασία που περιέχει μια Υπηρεσία (Service) η οποία έχει ξεκινήσει (έχει εκτελέσει την μέθοδο startService()).
4. Διεργασία Παρασκηνίου (background process). Μια διεργασία που περιέχει μια δραστηριότητα (Activity) η οποία δεν είναι ορατή στον χρήστη (έχει καλεσθεί η μέθοδος onStop()).
5. Άδεια Διεργασία (empty process). Μια διεργασία που δεν περιέχει κανένα ενεργό στοιχείο (διατήρηση κοινής μνήμης για βελτίωση του χρόνου εκκίνησης όταν ξαναχρησιμοποιηθεί).



Σχημα 2 : Κύκλος ζωής εφαρμογής Android

1.6.Ασφάλεια στο Android

Το μεγαλύτερο μέρος της διασφάλισης της ασφάλειας μεταξύ των εφαρμογών και του συστήματος το αναλαμβάνει ο πυρήνας του Linux. Ένας επιπλέον μηχανισμός ασφάλειας που παρέχεται από το Android είναι τα Permissions (Άδειες) τα οποία θέτουν περιορισμούς σε συγκεκριμένες ενέργειες που μπορεί να εκτελέσει μια διεργασία.

Η φιλοσοφία του μηχανισμού αυτού είναι ότι καμία εφαρμογή δεν έχει

δικαίωμα να εκτελέσει οποιαδήποτε ενέργεια που επηρεάζει δυσμενώς κάποια άλλη εφαρμογή, το λειτουργικό σύστημα ή τον χρήστη. Μερικές από αυτού του είδους της ενέργειες είναι η ανάγνωση και εγγραφή των δεδομένων του χρήστη, η ανάγνωση και εγγραφή αρχείων άλλων εφαρμογών, πρόσβαση στο δίκτυο, διατήρηση της συσκευής ανοικτής κ.α. Προκειμένου μια εφαρμογή να εκτελέσει μια τέτοιου είδους ενέργεια πρέπει να έχει δηλώσει ότι απαιτεί το αντίστοιχο permission. Αυτή η δήλωση γίνεται στατικά μέσα στην εφαρμογή (συγκεκριμένα μέσα στο AndroidManifest.xml) έτσι ώστε είναι γνωστή κατά την εγκατάσταση της εφαρμογής και δεν αλλάζει. Τα permissions που δηλώνει ότι απαιτεί μια εφαρμογή διαχειρίζονται από το λειτουργικό σύστημα με διάφορους τρόπους. Συνήθως είτε γίνονται αυτομάτως αποδεκτά ή μη αποδεκτά βάσει πιστοποιητικών, είτε ζητείται από τον χρήστη να αποφασίσει για την αποδοχή τους. Στην συνέχεια, θα δούμε με μεγαλύτερη λεπτομέρεια το μηχανισμό ασφάλειας που παρέχεται από το Android.

1.7.Φιλοσοφία σχεδίασης εφαρμογών Android

Η φιλοσοφία της σχεδίασης μιας εφαρμογής Android είναι η διατήρηση τριών βασικών στοιχείων: της ταχύτητας, της αποκρισιμότητας και της συνοχής.

Μια εφαρμογή Android πρέπει να είναι γρήγορη. Ο όρος γρήγορη χρησιμοποιείται με την έννοια της αποδοτικότητας. Υπάρχει η τάση στην επιστήμη των υπολογιστώ να θεωρείται ότι τελικά ο νόμος του Moore θα λύσει όλα τα προβλήματα. Όσο αναφορά όμως τα ενσωματωμένα συστήματα είναι πιο πολύπλοκος στην εφαρμογή του. Δεν εφαρμόζεται με τον ίδιο τρόπο στις κινητές συσκευές όπως στους υπολογιστές γραφείου ή στους διακομιστές.

Ο νόμος του Moore αφορά την πυκνότητα των τρανζίστορ που σημαίνει ότι μπορούν να πακεταριστούν περισσότερα κυκλώματα στο ίδιου μεγέθους chip με το πέρασμα του χρόνου. Για τους υπολογιστές γραφείου ή για τους διακομιστές αυτό σημαίνει ότι μπορεί να πακεταριστεί περισσότερη «ταχύτητα» σε ένα chip του ίδιου μεγέθους με αποτέλεσμα την αύξηση της επίδοσης. Στην περίπτωση των κινητών συσκευών ο νόμος του Moore χρησιμοποιείται για την κατασκευή μικρότερων chip. Με την χρήση μικρότερων chip μειώνεται η κατανάλωση ενέργειας με αποτέλεσμα την

μείωση του μεγέθους της κινητής συσκευής και την αύξηση της διάρκειας της μπαταρίας. Άρα οι επιδόσεις των ενσωματωμένων συστημάτων, όπως οι κινητές συσκευές, αυξάνονται με πολύ πιο αργό ρυθμό από τους υπολογιστές γραφείου.

Για αυτόν τον λόγο είναι σημαντικό ο κώδικας μιας εφαρμογής android να είναι αποδοτικός. Αυτό σημαίνει την όσο το δυνατόν μικρότερη δέσμευση μνήμης και την αποφυγή ορισμένων προγραμματιστικών ιδιωμάτων τα οποία μπορούν να καταστρέψουν την απόδοση.

Μια εφαρμογή android πρέπει να έχει αποκρισιμότητα. Μπορεί μια εφαρμογή να είναι αποδοτική και παρόλα αυτά να μην είναι φιλική προς τον χρήστη. Υπάρχουν εφαρμογές οι οποίες δεν ανταποκρίνονται αρκετά, δηλαδή φαίνεται πως λειτουργούν αργά ή παγώνουν για σημαντικά χρονικά διαστήματα ή κάνουν πολύ χρόνο να επεξεργαστούν την είσοδο από τον χρήστη.

Μια εφαρμογή android η οποία δεν έχει επαρκή αποκρισιμότητα θα προκαλεί συχνά την εμφάνιση του μηνύματος "Application Not Responding" από το σύστημα. Αυτό γενικά συμβαίνει όταν μια εφαρμογή δεν μπορεί να ανταποκριθεί σε κάποια είσοδο χρήστη. Για παράδειγμα η εφαρμογή σταματάει σε κάποια I/O διεργασία (συχνά μια πρόσβαση στο δίκτυο). Τότε το κύριο νήμα της εφαρμογής δεν θα μπορεί να χειριστεί τα όποια συμβάντα προκαλεί ο χρήστης. Μετά από κάποιο χρονικό διάστημα το σύστημα θα θεωρήσει ότι η εφαρμογή έχει κολλήσει και δίνει την δυνατότητα του τερματισμού της.

Παρομοίως αν μια εφαρμογή χρειάζεται πολύ χρόνο για την εκτέλεση υπολογισμών τότε πάλι το σύστημα θα θεωρήσει ότι η εφαρμογή έχει κολλήσει. Για τον λόγο αυτόν τέτοιοι υπολογισμοί πρέπει να γίνονται όσο το δυνατόν αποδοτικότερα. Αλλά ακόμα και ο πιο αποδοτικός υπολογισμός ορισμένες φορές χρειάζεται χρόνο για να εκτελεστεί. Σε τέτοιες περιπτώσεις συνήθως προτιμάται η δημιουργία ενός νήματος-παιδιού το οποίο εκτελεί την χρονοβόρα διεργασία-υπολογισμό. Με αυτόν τον τρόπο το κύριο νήμα το οποίο διαχειρίζεται τα συμβάντα χρήστη συνεχίζει να τρέχει και αποτρέπει το σύστημα από το να θεωρήσει οτι η εφαρμογή έχει κολλήσει.

Τέλος ακόμα και αν μια εφαρμογή είναι γρήγορη και έχει αποκρισιμότητα μπορεί να μην είναι φιλική προς τους χρήστες. Μια εφαρμογή πρέπει να

συνεργάζεται ομαλά με το σύστημα και με τις υπόλοιπες εφαρμογές και να ολοκληρώνεται πλήρως στο σύστημα.

1.8.Πακέτα του Android

1.8.1.Android Location API

Το Android SDK περιλαμβάνει δύο πακέτα τα οποία παρέχουν την απαραίτητη υποστήριξη για την δημιουργία location-based υπηρεσιών: το πακέτο android.location και το com.google.android.maps.

android.location Αυτό το πακέτο περιέχει μια πληθώρα κλάσεων που σχετίζονται με τις υπηρεσίες θέσης της πλατφόρμας του Android. Η πιο σημαντική κλάση που παρέχει είναι η LocationManager η οποία παρέχει μια διαπροσωπεία για τον καθορισμό της θέσης εάν η υποκείμενη συσκευή υποστηρίζει αυτήν την λειτουργία. Ο LocationManager δεν πρέπει να αρχικοποιείται από τον χρήστη. Αντιθέτως πρέπει να λαμβάνεται μια αναφορά με την χρήση της μεθόδου getSystemService μιας και αποτελεί μια από τις υπηρεσίες του συστήματος. Αν μια εφαρμογή έχει μια αναφορά στον LocationManager μπορεί να κάνει τα παρακάτω τρία πράγματα:

- Ερώτηση για όλους τους παρόχους θέσης (LocationProviders) οι οποίοι είναι γνωστοί στον LocationManager για την τελευταία γνωστή περιοχή.
- Λήψη περιοδικών ενημερώσεων για την τρέχουσα θέση από έναν πάροχο θέσης (LocationProvider) (είτε με βάση κάποια κριτήρια, είτε με βάση το όνομα του)
- Δυνατότητα για μια δοσμένη Πρόθεση (Intent) να ενεργοποιηθεί εάν η συσκευή εισέλθει σε μια ορισμένη περιοχή.

com.google.android.maps Αυτό το πακέτο περιέχει ένα σύνολο από κλάσεις που σχετίζονται με την παρουσίαση, και τον έλεγχο ενός Google Map σε μια Activity. Η πιο σημαντική κλάση του πακέτου είναι η MapView η οποία εμφανίζει αυτόματα τον βασικό Google Map. Για την ορθή χρήση αυτού του πακέτου πρέπει να αποκτηθεί ένα κλειδί MapView API από την υπηρεσία Google Maps ώστε να γίνεται δυνατή η φόρτωση των χαρτών. Μετά την δημιουργία ενός αντικειμένου MapView μπορεί να γίνει η ανάκτηση ενός

MapController για τον έλεγχο και την κίνηση του χάρτη και η προσθήκη πληροφοριών στον χάρτη.

1.8.2.Android WiFi API

Το WiFi API περιέχεται στο πακέτο android.net.wifi. Αποτελείται από ένα σύνολο κλάσεων με βασικότερη την WifiManager η οποία παρέχει μια διαπροσωπεία για τον χειρισμό όλων των πτυχών μιας WiFi σύνδεσης. Ο WifiManager όπως και ο LocationManager δεν πρέπει να αρχικοποιείται από τον χρήστη αλλά ως μια υπηρεσία συστήματος πρέπει να λαμβάνεται μέσω της μεθόδου getSystemService. Μέσω του WifiManager μπορεί να γίνει:

- Διαχείριση της λίστας των δικτύων. Αυτή η λίστα μπορεί απλά να διαβαστεί, να ανανεωθεί καθώς επίσης ιδιότητες μεμονωμένων εγγραφών της μπορούν να αλλαχτούν.
- Εύρεση του τρέχοντος ενεργού WiFi δικτύου, εάν υπάρχει. Μπορεί να γίνει σύνδεση ή αποσύνδεση από αυτό και να ανακτηθούν δυναμικές πληροφορίες για την κατάσταση του δικτύου.
- Διαχείριση των αποτελεσμάτων της αναζήτησης για σημεία πρόσβασης (access points), τα οποία περιέχουν αρκετές πληροφορίες ώστε να ληφθεί η απόφαση σε ποιο από τα σημεία πρόσβασης θα γίνει σύνδεση.
- Εκπομπή συγκεκριμένων Προθέσεων (Intents) σε περίπτωση κάποιας αλλαγής στην κατάσταση του WiFi.

1.8.3.Άλλα πακέτα

Το Android API περιέχει αρκετά ακόμα πακέτα. Μερικά από αυτά είναι:

- Το Media API που περιέχεται στο android.media το οποίο παρέχει υποστήριξη για την αναπαραγωγή ήχου και βίντεο, καθώς επίσης και streaming ήχο ή βίντεο.
- Το OpenGL API που περιέχεται στο android.opengl και αποτελεί μια διαπροσωπεία για την χρήση της βιβλιοθήκης OpenGl.

1.9.Διαδικασία εγκατάστασης

1.9.1.Εγκατάσταση του Android

Ο χρήστης κάποιας κινητής συσκευής βασισμένης στο android δεν χρειάζεται να κάνει τίποτα για να το εγκαταστήσει στην συσκευή του αφού βρίσκεται προεγκατεστημένο σε αυτή. Ο προγραμματιστής εφαρμογών βασισμένων στο android από την άλλη χρειάζεται να κατεβάσει τις βιβλιοθήκες και τα εργαλεία του android από το επίσημο site (τρέχουσα έκδοση την στιγμή των γραφομένων είναι η Android SDK 1.0). Με την βοήθεια αυτών των εργαλείων καθώς και με την χρήση του ολοκληρωμένου περιβάλλοντος eclipse όπως αναφέρθηκε και παραπάνω η διαδικασία συγγραφής μιας android εφαρμογής απλοποιείται κατά πολύ.

1.9.2.Εγκατάσταση εφαρμογών Android

Για να μπορέσει να διαμοιραστεί μια εφαρμογή γραμμένη σε android πακετάρεται σε ένα αρχείο *.apk (android package). Το πακέτο αυτό περιέχει τόσο τις κλάσεις της εφαρμογής μεταγλωττισμένες για να τρέχουν στην Dalvic Virtual Machine, όσο και τους πόρους της εφαρμογής (γραφικά, ήχο κλπ). Ο κάθε χρήστης που θέλει να χρησιμοποιήσει την εφαρμογή στην κινητή συσκευή του πρέπει να αντιγράψει το αρχείο-πακέτο της εφαρμογής στην συσκευή του έτσι ώστε να εγκατασταθεί σε αυτήν. Μετά την αντιγραφή ο χρήστης μπορεί άμεσα να χρησιμοποιήσει την εφαρμογή επιλέγοντας το εικονίδιο της από το menu των εφαρμογών.

1.10.Μοντέλο ασφάλειας του Android

Το Android είναι μία πλατφόρμα Linux ενισχυμένη με τους μηχανισμούς ασφαλείας που χρειάζονται για ένα κινητό περιβάλλον. Το Android συνδυάζει όλα τα χαρακτηριστικά γνωρίσματα ενός λειτουργικού συστήματος, αποδοτική κοινή μνήμη (shared memory), υποστήριξη πολυπρογραμματισμού (multi-tasking), αναγνώριση χρηστών (Unix User Identifiers – UIDs), δικαιώματα αρχείων (file permissions) και μία ασφαλή γλώσσα προγραμματισμού όπως είναι η Java. Το μοντέλο ασφάλειας Android είναι πολύ πιο βελτιωμένο (σαν ένας multi-user server) από το sandbox που συναντάμε στο J2ME ή τις

πλατφόρμες Blackberry. Σε αντίθεση με τους υπολογιστές γραφείου όπου όλες οι εφαρμογές τρέχουν με το ίδιο UID, οι εφαρμογές Android τρέχουν η καθεμία χωριστά. Οι εφαρμογές Android τρέχουν ως ξεχωριστές διαδικασίες (processes) με ξεχωριστά UIDs και διαφορετικά δικαιώματα. Τα προγράμματα δεν μπορούν ούτε να γράψουν ούτε να διαβάσουν δεδομένα και κώδικα το ένα από το άλλο εκτός αν ο σχεδιαστής της εφαρμογής έχει ορίσει ρητά ότι θέλει ανταλλαγή με συγκεκριμένες άλλες εφαρμογές. Το GUI περιβάλλον του Android έχει μερικές νέες ιδιότητες ασφαλείας που βοηθούν την υποστήριξη αυτής της απομόνωσης.

Οι κινητές πλατφόρμες με το πέρασμα του χρόνου αποκτούν μεγάλη σημασία και καλούνται να καλύψουν όλο και πιο σύνθετες απαιτήσεις. Το Android υποστηρίζει την δημιουργία εφαρμογών που χρησιμοποιούν υπηρεσίες τηλεφώνου ενώ παράλληλα προστατεύουν τους χρήστες, ελαχιστοποιώντας τις συνέπειες από λάθη και κακόβουλα λογισμικά (bugs, malicious software, κ.α.). Η απομόνωση των διαδικασιών δίνει την ευελιξία στις εφαρμογές να χρησιμοποιήσουν το βασικά χαρακτηριστικά της πλατφόρμας χωρίς να παραβιάζουν την ασφάλεια του Android ή να δίνουν στην εφαρμογή επιπλέον δικαιώματα.

Τα δικαιώματα του Android είναι δικαιώματα που δίνονται στις εφαρμογές για να τους επιτραπεί να κάνουν πράγματα όπως να πάρουν εικόνες ή να χρησιμοποιήσουν το GPS ή να κάνουν φωνητικές κλήσεις. Όταν εγκαθίστανται, στις εφαρμογές δίνεται ένα μοναδικό UID και η εφαρμογή θα τρέχει πάντα με αυτό το UID στην συγκεκριμένη συσκευή. Το UID μιας εφαρμογής χρησιμοποιείται για να προστατεύσει τα δεδομένα της και οι σχεδιαστές των εφαρμογών θα πρέπει να είναι ρητοί για το αν θέλουν ανταλλαγή δεδομένων με άλλες εφαρμογές. Οι εφαρμογές μπορούν να διασκεδάσουν τους χρήστες με γραφικά, παίζοντας μουσική και να εκκινήσουν άλλα προγράμματα χωρίς πρόσθετη άδεια.

Τα κακόβουλα λογισμικά είναι μία πραγματικότητας για τις δημοφιλείς πλατφόρμες και το Android μέσα από τα χαρακτηριστικά του προσπαθεί αν ελαχιστοποιήσει τον αντίκτυπο και συνέπειες από τέτοια κακόβουλα λογισμικά. Εντούτοις, τα μη εξουσιοδοτημένα κακόβουλα λογισμικά εφόσον

εγκατασταθούν σε μία Android συσκευή (με την προσποίηση ότι είναι χρήσιμες εφαρμογές) μπορούν προσωρινά να δημιουργήσουν πρόβλημα στον χρήστη. Οι χρήστες σε τέτοιες ανεπιθύμητες περιπτώσεις θα πρέπει να εντοπίσουν και να αφαιρέσουν την κακόβουλη εφαρμογή. Το Android βοηθάει τους χρήστες να εντοπίσουν τις κακόβουλες εφαρμογές και να περιορίσουν την έκταση μίας πιθανής κατάχρησης, μέσα από την απαίτηση της άδειας χρήστη (user permission) από προγράμματα που κάνουν επικίνδυνα πράγματα όπως:

- Κλήση που μπορεί να εμπεριέχει επιπλέον χρέωση
- Αποκάλυψη των ιδιωτικών στοιχείων του χρήστη
- Καταστροφή τηλεφωνικού καταλόγου, ηλεκτρονική αλληλογραφίας, κ.α.

Η απάντηση του χρήστη σε ένα ενοχλητικό, γεμάτο προβλήματα ή κακόβουλο λογισμικό είναι απλά η απεγκατάσταση του. Εάν το κακόβουλο λογισμικό έχει αναστατώσει αρκετά το τηλέφωνο και ο χρήστης δεν μπορεί να κάνει απεγκατάσταση του, ο χρήστης μπορεί να κάνει επανεκκίνηση (reboot) του τηλεφώνου (προαιρετικά σε safe mode, όπου σταματάει να τρέχουν προγράμματα ή κώδικας που δεν ανήκει στο σύστημα) και να αφαιρέσει το λογισμικό αυτό προτού τρέξει πάλι.

1.10.1.Υπευθυνότητες των σχεδιαστών λογισμικού

Όλοι οι σχεδιαστές Android εφαρμογών, θα πρέπει να σκέφτονται πώς θα εξασφαλίσουν την ασφάλεια των χρηστών όπως ακριβώς σκέφτονται πώς θα αντιμετωπίσουν την περιορισμένη μνήμη, την περιορισμένη δυνατότητα των επεξεργαστών και την χρήση μπαταρίας. Οι σχεδιαστές Android εφαρμογών θα πρέπει να προστατεύσουν όλα τα δεδομένα που εισάγουν οι χρήστες στις εφαρμογές τους και να μην επιτρέψουν σε κακόβουλα λογισμικά να αποκτήσουν πρόσβαση στις εφαρμογές τους. Το πώς επιτυγχάνεται αυτό συσχετίζεται εν μέρει με ποια χαρακτηριστικά γνωρίσματα της πλατφόρμας χρησιμοποιούνται από την κάθε εφαρμογή.

Ένα από τα πολυπλοκότερα πράγματα που πρέπει να καταλάβουμε για το Android είναι ότι κάθε εφαρμογή τρέχει με διαφορετικό UID. Χαρακτηριστικά σε ένα υπολογιστή γραφείου, κάθε χρήστης έχει ένα μοναδικό UID και κάθε εφαρμογή χρησιμοποιεί το UID του χρήστη της. Στο Android, το σύστημα δίνει

σε κάθε εφαρμογή, και όχι σε κάθε χρήστη, ένα μοναδικό UID. Για παράδειγμα, όταν γίνεται εκκίνηση μίας εφαρμογής, η νέα διαδικασία δεν θα τρέξει με το UID του χρήστη της, αλλά με το δικό μοναδικό UID. Είναι σημαντικό ότι εάν ένα πρόγραμμα τρέξει με κακές παραμέτρους, ο σχεδιαστής της εφαρμογής αυτής θα πρέπει να έχει εξασφαλίσει ότι η εφαρμογή δεν θα βλάψει το σύστημα ή δεν θα κάνει κάτι που ο χρήστης δεν επιθυμεί. Οποιαδήποτε εφαρμογή μπορεί να ζητήσει από τον Activity Manager να κάνει εκκίνηση μίας άλλης εφαρμογής που θα τρέξει με το δικό της μοναδικό UID.
Ευτυχώς, τα μη έμπιστα σημεία εισόδου (entry-points) των εφαρμογών περιορίζονται από τα ιδιαίτερα χαρακτηριστικά γνωρίσματα της πλατφόρμας που επιλέγονται να χρησιμοποιηθούν από την εφαρμογή και η ασφάλεια τους εξασφαλίζεται με ένα συνεπή τρόπο. Οι εφαρμογές Android δεν έχουν μία απλή κεντρική μέθοδο (main function) από την οποία περνάει πάντα η εφαρμογή όταν ξεκινάει. Αντίθετα, τα αρχικά σημεία εισόδου των εφαρμογών είναι βασισμένα στην εγγραφή των δραστηριοτήτων (activities), των υπηρεσιών (services), των δεκτών εκπεμπόμενων προθέσεων (Broadcast Receivers) ή των παρόχων περιεχομένου (Content Providers) με το σύστημα. Μετά από μια ανανέωση των Android δικαιωμάτων και προθέσεων, καλύπτουμε με ασφάλεια όλους αυτούς που χρησιμοποιούν αυτά τα γνωρίσματα της πλατφόρμας.
Το Android απαιτεί από τους μηχανικούς ανάπτυξης λογισμικού σε android να υπογράφουν τον κώδικα τους. Ο κώδικας Android υπογράφεται χρησιμοποιώντας αυτό-υπογεγραμμένα πιστοποιητικά, όπου οι μηχανικοί ανάπτυξης λογισμικού μπορούν να παράγουν χωρίς την βοήθεια ή την άδεια κανενός. Οι υπογραφές αυτές επιτρέπουν στους μηχανικούς ανάπτυξης λογισμικού να ανανεώσουν το λογισμικό τους χωρίς την δημιουργία περίπλοκων αδειών. Οι εφαρμογές που υπογράφονται με το ίδιο κλειδί (και επομένως από τον ίδιο μηχανικό ανάπτυξης λογισμικού) μπορούν να ζητήσουν να τρέξουν με το ίδιο UID. Αυτό επιτρέπει στους δημιουργούς μίας εφαρμογής να μπορούν να την αναβαθμίσουν ή να επιδιορθώσουν εύκολα. Η υπογραφή είναι διαφορετική από την κανονική υπογραφή Jar ή Authenticode, δεδομένου ότι η πραγματική ταυτότητα του δημιουργού μίας εφαρμογής δεν

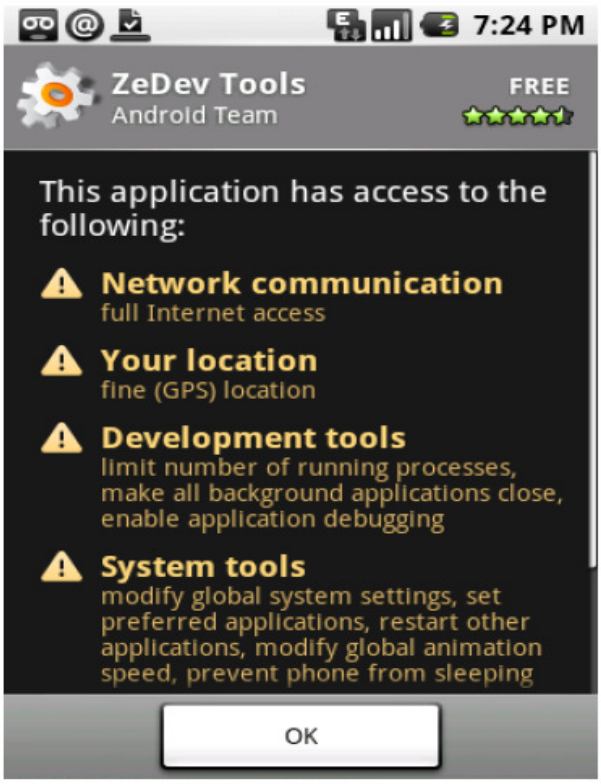
επικυρώνεται από κάποιο τρίτο. Οι μηχανικοί ανάπτυξης λογισμικού κερδίζουν μία καλή φήμη με την παραγωγή καλών προϊόντων. Τα πιστοποιητικά αποδεικνύουν μόνο τον δημιουργό της εφαρμογής. Οι μηχανικοί ανάπτυξης λογισμικού Android δεν είναι έμπιστοι. Αυτή η προσέγγιση είναι νέα και μπορεί να πετύχει, παρόλα αυτά δεν θα ήταν τεχνικά δύσκολο να προστεθούν κανόνες για την χορήγηση υπογραφών, εάν αποδειχτεί επιθυμητό.

1.10.2.Επισκόπηση των δικαιωμάτων Android

Οι εφαρμογές χρειάζονται έγκριση για να κάνουν πράγματα, όπως να στείλουν σύντομα γραπτά μηνύματα (SMS), να χρησιμοποιήσουν τη φωτογραφική μηχανή ή να αποκτήσουν πρόσβαση στην κατάλογο επαφών του χρήστη. Το Android χρησιμοποιεί τα Manifest δικαιώματα για να γνωρίζουν τι ο χρήστης επιτρέπει στην εφαρμογή να κάνει. Τα δικαιώματα κάθε εφαρμογής πρέπει να καταγράφονται στο δικό της AndroidManifest.xml και ο χρήστης συμφωνεί με αυτά από την στιγμή που θα εγκαταστήσει την εφαρμογή. Κατά την διάρκεια της εγκατάστασης κάθε εφαρμογής, ο χρήστης έχει την δυνατότητα να σκεφτεί και να αποφασίσει αν θα εμπιστευτεί το συγκεκριμένο λογισμικό βασισμένος στις κριτικές, την φήμη του δημιουργού και τα δικαιώματα που ζητάει η εφαρμογή. Τα δικαιώματα καλούνται μερικές φορές ως "Manifest Δικαιώματα" ή "Android Δικαιώματα" για να υπάρχει μία διάκριση με τα δικαιώματα των αρχείων.

Τα δικαιώματα θα πρέπει να συνδέονται με κάποιο στόχο που ο χρήστης καταλαβαίνει. Για παράδειγμα, μια εφαρμογή χρειάζεται την READ_CONTACTS άδεια για να διαβάσει τον κατάλογο επαφών του χρήστη. Μία εφαρμογή διαχείρισης επαφών χρειάζεται την άδεια READ_CONTACTS, σε αντίθεση με ένα παιχνίδι που δεν χρειάζεται τέτοια άδεια. Είναι πιθανό να εξασφαλιστεί ενός ενιαίου είδους άδειας από πολλούς διαφορετικούς μηχανισμούς Android IPC (inter-process communication). Εκκίνηση δραστηριοτήτων, εκκίνηση και σύνδεση με μία υπηρεσίας, πρόσβαση στον πάροχο περιεχομένου και παραλαβή και αποστολή εκπεμπόμενων προθέσεων μπορεί να θέλουν την ίδια άδεια. Επομένως, ο χρήστης δεν θα πρέπει να καταλάβει κάτι περισσότερο από το παρακάτω: "Μία εφαρμογή διαχείρισης επαφών χρειάζεται να διαβάζει τις επαφές". Οι χρήστες δεν καταλαβαίνουν πώς δουλεύει η συσκευή και η

εφαρμογή, επομένως θα πρέπει οι μηχανικοί ανάπτυξης λογισμικού να κρατούν τα δικαιώματα απλά και να αποφεύγουν τους τεχνικούς όρους όπως "Binder", "Activity", "Intent" όταν περιγράφουν τα δικαιώματα στον χρήστη. Μετά την εγκατάσταση μίας εφαρμογής, τα δικαιώματα της εφαρμογής αυτής δεν μπορούν να αλλάξουν. Με την ελαχιστοποίηση των δικαιωμάτων που απαιτούνται για κάθε εφαρμογή, περιορίζονται οι συνέπειες των πιθανών λαθών ασφαλείας από την εφαρμογή και κάνουν τους χρήστες να αισθάνονται καλύτερα για την εγκατάσταση της εφαρμογής αυτής. Όταν γίνεται εγκατάσταση μίας εφαρμογής Android, οι χρήστες βλέπουν τα απαιτούμενα δικαιώματα σε ένα παράθυρο όπως βλέπουμε στην παρακάτω εικόνα. Η εγκατάσταση ενός λογισμικού είναι πάντα ένας κίνδυνος και οι χρήστες θα πρέπει να μην εγκαθιστούν λογισμικά που δεν ξέρουν, ειδικά αν ζητούν αρκετά δικαιώματα.



Dialog showing Application permissions to users.

(Chu, 2008)

Σχημα 3 : Δικαιώματα εφαρμογών Android

Από την πλευρά των δημιουργών λογισμικού, τα δικαιώματα είναι συμβολοσειρές που σχετίζονται με εφαρμογή τους και το UID της. Οι δημιουργού του λογισμικού μπορούν να χρησιμοποιήσουν την μέθοδο της Context Class:

```
checkPermission(String Permission, int pid, int uid)
```

για να ελέγξουν προγραμματιστικά αν μία διαδικασία (και το αντίστοιχο UID) έχουν ένα συγκεκριμένο δικαίωμα, όπως READ_CONTACTS.

Παρακάτω, βλέπουμε ένα παράδειγμα ορισμού δικαιώματος. Να σημειώσουμε εδώ ότι η περιγραφή (description) και η ετικέτα (label) είναι πόροι που βοηθούν στον εντοπισμό της εφαρμογής.

```
<permission
xmlns:android="http://schemas.android.com/apk/res/android"
android:name="com.isecpartners.android.ACCESS_SHOPPING_LIST"
android:description="@string/access_perm_desc"
android:protectionLevel="normal"
android:label="@string/access_perm_label">
</permission>
```

Τα Manifest δικαιώματα όπως αυτό παραπάνω, έχει μερικές βασικές ιδιότητες. Απαραίτητα χρειάζεται η ετικέτα (μικρό κείμενο) και η περιγραφή της εφαρμογής (μεγαλύτερο κείμενο) που χρησιμοποιούνται κατά την εγκατάσταση της εφαρμογής. Όλα τα δικαιώματα πρέπει να έχουν ένα συνολικά μοναδικό όνομα. Το μοναδικό αυτό όνομα είναι ένας προσδιοριστής δικαιώματος που χρησιμοποιείται από τους προγραμματιστές. Τα δικαιώματα έχουν επίσης ένα επίπεδο προστασίας (protection level όπως παρουσιάζεται παραπάνω).

Υπάρχουν μόνο τέσσερα επίπεδα προστασίας για τα δικαιώματα:

- **Κανονικό – Normal:**

Δικαιώματα για γνωρίσματα της εφαρμογής όπου οι πιθανές επικίνδυνες συνέπειες είναι μικρές, όπως VIBRATE που επιτρέπουν στην συσκευή να δονηθεί. Κατάλληλο για δικαιώματα που δεν ενδιαφέρουν σημαντικά τους τελικούς χρήστες. Οι χρήστες μπορούν να διαβάσουν τα δικαιώματα αυτά, όμως δεν πρόκειται να προειδοποιηθούν ρητά από την εφαρμογή.

- **Επικίνδυνο – Dangerous:**

Τα δικαιώματα όπως WRITE_SETTINGS ή SEND_SMS είναι επικίνδυνες αφού μπορούν να τροποποιήσουν τις ρυθμίσεις του κινητού ή μπορούν να χρεώσουν υπερβολικά τον χρήστη. Κατάλληλο για δικαιώματα που ενδιαφέρουν τους χρήστες. Το Android θα προειδοποιήσει τους χρήστες για την ανάγκη χρήσης αυτών των δικαιωμάτων κατά την διάρκεια της εγκατάστασης.

- **Υπογραφή – Signature:**

Τα δικαιώματα αυτά μπορούν μόνο να χορηγηθούν σε άλλες εφαρμογές που υπογράφονται με το ίδιο κλειδί με αυτό του προγράμματος. Αυτό επιτρέπει την ασφαλή συνεργασία χωρίς την ανάγκη μίας δημόσιας διεπαφής.

- **Υπογραφή του συστήματος – Signature Of System:**

Παρόμοιο με την υπογραφή με την διαφοροποίηση ότι τα προγράμματα του συστήματος ζητάνε την δυνατότητα πρόσβασης. Αυτό επιτρέπει στα Android συστήματα που τρέχουν σε πελάτες να παίρνουν τα απαραίτητα δικαιώματα. Αυτή η προστασία βοηθάει την εύκολη ενσωμάτωση των Android συστημάτων με άλλα συστήματα και δεν χρειάζεται για τους μηχανικούς ανάπτυξης λογισμικού σε Android.

Εάν προσπαθήσετε να χρησιμοποιήσετε μία διεπαφή που δεν έχετε την άδεια, θα λάβετε ένα σφάλμα ασφάλειας (Security Exception). Μπορείτε επίσης να δείτε ένα μήνυμα λάθους που θα σας προσδιορίζει ποίο δικαίωμα πρέπει να ενεργοποιήσετε. Εάν το πρόγραμμα σας απαιτεί κάποια ιδιαίτερα δικαιώματα θα πρέπει να σκεφτείτε να έχετε προνοήσει για μηνύματα λάθους ώστε οι προγραμματιστές που χρησιμοποιούν την εφαρμογή σας να μπορούν να εντοπίσουν εύκολα το πρόβλημα. Μερικές φορές οι αποτυχίες σε παροχή δικαιωμάτων είναι σιωπηλές. Η ίδια η πλατφόρμα ούτε προειδοποιεί τους χρήστες όταν αποτυγχάνουν οι έλεγχοι άδειας, ούτε επιτρέπει τη χορήγηση δικαιωμάτων στις εφαρμογές μετά την εγκατάσταση τους.

Εκτός από την ανάγνωση και γράψιμο δεδομένων, πολλά δικαιώματα επιτρέπουν στις εφαρμογές να καλούν υπηρεσίες του συστήματος ή να ξεκινούν δραστηριότητες με απόρρητα αποτελέσματα που πρέπει να προστατευθούν. Για παράδειγμα, με τα σωστά δικαιώματα, ένα παιχνίδι μπορεί να πάρει τον συνολικό έλεγχο της οθόνης ή ένας Dialer μπορεί να αναγκάσει

το τηλέφωνο να σχηματίσει έναν αριθμό χωρίς την αίτηση του χρήστη.

1.10.3.Δημιουργία νέων Manifest δικαιωμάτων

Οι εφαρμογές μπορούν να ορίσουν τα δικά τους δικαιώματα εάν υπάρχει πρόθεση από άλλες εφαρμογές να έχουν προγραμματιστική πρόσβαση σε αυτές. Χρησιμοποιώντας τα Manifest δικαιώματα επιτρέπουμε στον τελικό χρήστη να αποφασίσει ποια προγράμματα έχουν πρόσβαση. Για παράδειγμα, μία εφαρμογή που διαχειρίζεται έναν κατάλογο αγορών θα μπορούσε να ορίσει ένα δικαίωμα με όνομα "ACCESS_SHOPPING_LIST". Εάν μία εφαρμογή καθορίσει ένα αντικείμενο ShoppingList, το δικαίωμα ACCESS_SHOPPING_LIST χρειάζεται για να έχει πρόσβαση στον κατάλογο αγορών. Το δικαίωμα αυτό θα απαιτείται από όλες τις εφαρμογές που θα προσπαθήσουν να δουν ή ενημερώσουν τον κατάλογο αγορών. Μόνο τα προγράμματα και οι εφαρμογές που δηλώνουν ότι θα χρησιμοποιήσουν αυτό το δικαίωμα θα μπορούν να έχουν πρόσβαση στον κατάλογο, δίνοντας στον χρήστη την δυνατότητα να αποτρέψουν την μη ταυτοποιημένη πρόσβαση.

Η προσθήκη δικαιωμάτων θα πρέπει να αποφευχθεί όποτε δεν είναι απαραίτητο. Για παράδειγμα, θα μπορούσαμε να καθορίσουμε μία δραστηριότητα που προσθέτει ένα νέο στοιχείο στον κατάλογο αγορών. Όταν μία εφαρμογή καλέσει το startActivity για να μπορέσει να εισάγει ένα νέο στοιχείο στον κατάλογο αγορών, η δραστηριότητα θα μπορούσε να ζητήσει την άδεια από τον χρήστη για να συνεχίσει με την εισαγωγή του νέου στοιχείου, από το να απαιτούσε την χορήγηση συγκεκριμένου δικαιώματος ή άδειας. Αυτό κρατάει το σύστημα απλό για τους χρήστες και αποφεύγουμε άσκοπο προγραμματιστικό κόπο. Εάν υπήρχε απαίτηση για την δραστηριότητα να γίνονται οι αλλαγές στον κατάλογο αγορών απευθείας, χωρίς την προειδοποίηση του χρήστη θα απαιτούσε να ακολουθήσουμε αναπόφευκτα την προσέγγιση με την εισαγωγή νέου Manifest δικαιώματος.

Η δημιουργία νέων Manifest δικαιωμάτων μπορεί να βοηθήσει στην ελαχιστοποίηση των απαιτήσεων σε δικαιώματα για τη εφαρμογές που χρησιμοποιούν το πρόγραμμα σας προγραμματιστικά. Για παράδειγμα, εάν μία εφαρμογή χρειάζεται άδεια για να στείλει SMS μηνύματα και να αποκτήσει

πρόσβαση στα δεδομένα θέσης του χρήστη, θα μπορούσε να οριστεί ένα νέο δικαίωμα όπως "SEND_LOCATION_MESSAGE". Αυτό θα ήταν το μοναδικό δικαίωμα που θα απαιτούνταν από τις εφαρμογές που θέλουν να χρησιμοποιήσουν την υπηρεσία σας, κάνοντας την εγκατάσταση τους πιο εύκολη και καθαρότερη στον χρήστη.

1.10.4.Εκπομπή συγκεκριμένων προθέσεων (Intents)

Η εκπομπή προθέσεων (Intent) είναι ένας μηχανισμός του Android για την μεταφορά δεδομένων μεταξύ των διαφορετικών διαδικασιών του Android και είναι το σημαντικότερο χαρακτηριστικό του Android IPC. Δεν έχουν ενσωματωμένη κάποια πολιτική ασφάλειας, αλλά αγγελιοφόρος είναι υπεύθυνος να ακολουθεί τα απαραίτητα όρια ασφάλειας. Για να επιτραπεί η επικοινωνία η εκπομπή προθέσεων πρέπει να γίνεται μέσω Binder interfaces (δεδομένου ότι εφαρμόζουν την Parcelable interface). Γενικότερα, όλα τα Android IPC έχουν υλοποιηθεί μέσω του Binder, αν και τις περισσότερες φορές αυτό το επίπεδο είναι κρυμμένο.

1.10.4.1.Προεπισκόπηση εκπομπής προθέσεων

Η εκπομπή προθέσεων χρησιμοποιείται με διάφορους τρόπους από το Android:

- Για να αρχίσει μία δραστηριότητα:
    - Χρησιμοποιώντας την μέθοδο startActivity() της κλάσης Context.
- Εκπομπή προθέσεων για να ενημερώσει τα ενδιαφερόμενα προγράμματα για αλλαγές ή γεγονότα:
    - Χρησιμοποιώντας τις μεθόδους της κλάσης Context sendBroadcast(), sendStickyBroadcast(), sendOrderedBroadcast().
- Σαν τρόπος για να αρχίσει, σταματήσει ή να επικοινωνήσει με υπηρεσίες στο παρασκήνιο (Background services):
    - Χρησιμοποιώντας τις μεθόδους της κλάσης Context startService(), stopService(), bindService().
- Για να έχει πρόσβαση στα δεδομένα μέσω παρόχων δεδομένων (Content Providers), όπως οι επαφές του χρήστη:
    - Χρησιμοποιώντας τη μέθοδο getContentResolver() της κλάσης Context.

- Σαν επανακλήσεις (call backs) για τον χειρισμό γεγονότων, όπως την επιστροφή αποτελεσμάτων ή λαθών ασύγχρονα με PendingIntents από τους clients στους servers μέσω του Binder interface.

Η εκπομπή προθέσεων έχει πολλές λεπτομέρειες υλοποίησης, αλλά η βασική ιδέα είναι ότι αντιπροσωπεύουν μπλοκ δεδομένων που μπορούν να μεταφερθούν ανάμεσα σε προγράμματα. Η εκπομπή προθέσεων έχουν συνήθως μία δράση, η οποία είναι μια συμβολοσειρά όπως "android.intent.action.VIEW" που προσδιορίζει κάποιο ιδιαίτερο στόχο και συχνά κάποια δεδομένα με την μορφή URI. Επίσης, η εκπομπή προθέσεων έχει κάποιες προαιρετικές ιδιότητες (category, extra, κ.α.). Γενικότερα, τα APIs τα οποία χειρίζονται προθέσεις μπορούν να περιοριστούν μέσω των Manifest δικαιωμάτων. Αυτό επιτρέπει να δημιουργήσετε δραστηριότητες, δέκτες προθέσεων, παρόχους δεδομένων ή υπηρεσίες που έχουν πρόσβαση μόνο εφαρμογές στις οποίες ο χρήστης έχει χορηγήσει αυτή την άδεια.

1.10.4.2.Φίλτρα εκπομπής προθέσεων

Η εκπομπή προθέσεων μπορεί να γίνει από τον Activity Manager του Android. Για παράδειγμα, η εκπομπή προθέσεων μπορεί να χρησιμοποιηθεί για την έναρξη μίας δραστηριότητας με την κλήση της Context.startActivity(). Η δραστηριότητα που πρέπει να ξεκινήσει βρίσκεται από τον Activity Manager μέσα από την αναζήτηση του καλύτερου ταιριάσματος ανάμεσα στο εισερχόμενο φίλτρο εκπομπής προθέσεων και του φίλτρου που έχει καταχωρηθεί για όλες τις δραστηριότητες στο σύστημα. Εντούτοις, η εκπομπή προθέσεων μπορεί να αγνοήσει το ταίριασμα του φίλτρου από τον Activity Manager. Οποιαδήποτε δραστηριότητα μπορεί να ξεκινήσει την εκπομπή προθέσεων με οποιοδήποτε τιμή σαν Action, Data, Category, Extra, κ.α. Το φίλτρο εκπομπής προθέσεων είναι μία δικλίδα ασφαλείας από την πλευρά του δέκτη προθέσεων. Στην περίπτωση της εκκίνησης μίας δραστηριότητας, ο καλών αποφασίζει ποιο στοιχείο θέλει να ξεκινήσει και εκπέμπει την πρόθεση που στην συνέχεια θα λάβει ο δέκτης. Ο καλών μπορεί να επιλέξει να ρωτήσει βοήθεια από τον Activity Manager σχετικά με το που πρέπει να πάει η πρόθεση, αλλά δεν είναι υποχρεωμένος να το κάνει.

Οι παραλήπτες όπως δραστηριότητες, υπηρεσίες και δέκτες προθέσεων πρέπει

να είναι έτοιμοι να αντιμετωπίσουν τους ενδεχόμενους εχθρικούς επισκέπτες, και ένα φίλτρο εκπομπής προθέσεων δεν φιλτράρει τις κακόβουλες προθέσεις. Τα φίλτρα εκπομπής προθέσεων βοηθούν το σύστημα να υπολογίσει το σωστό χειρισμό για κάθε πρόθεση ξεχωριστά, αλλά δεν αποτελεί φιλτράρισμά εισόδου στο σύστημα. Επειδή τα φίλτρα αυτά δεν είναι μία ασφαλιστική δικλίδα δεν μπορούν να συνδεθούν με τις άδειες. Όπως θα δούμε και παρακάτω κανένας μηχανισμός IPC δεν μπορεί να χρησιμοποιηθεί για επικύρωση σφαλμάτων εισόδου.

Στις προθέσεις μπορούν να προστεθούν κατηγορίες (Categories), κάνοντας το σύστημα πιο επιλεκτικό για ποια πρόθεση καλείται να αντιμετωπίσει. Οι κατηγορίες μπορούν επίσης να προστεθούν στα φίλτρα εκπομπής προθέσεων για να επιτρέψουν στις προθέσεις να περάσουν, δηλώνοντας ξεκάθαρα ότι το φιλτραρισμένο αντικείμενο υποστηρίζει τους περιορισμούς της κατηγορίας αυτής. Αυτό είναι χρήσιμο στις περιπτώσεις όπου στέλνουμε μία πρόθεση της οποίας ο παραλήπτης καθορίζεται από το Android, όπως κατά την έναρξη μίας δραστηριότητας ή την μετάδοση μίας πρόθεσης.

Η προσθήκη μίας κατηγορίας στην πρόθεση την περιορίζει στο τι μπορεί να κάνει. Παραδείγματος χάριν, ένα φίλτρο εκπομπής προθέσεων που έχει σαν κατηγορία "android.intent.category.BROWSABLE" δείχνει ότι είναι ασφαλές να κληθεί από την μηχανή αναζήτησης ιστού. Μελλοντικές κατηγορίες θα μπορούσαν να δείξουν ότι μία πρόθεση προέρχεται από απομακρυσμένη μηχανή ή μη-έμπιστη μηχανή αλλά επειδή η κατηγορία αυτή δεν θα ταιριάζει με το φίλτρο εκπομπής προθέσεων που βάζουμε σήμερα, το σύστημα δεν θα παραδώσει την πρόθεση στην εφαρμογή μας. Αυτό προστατεύει την εφαρμογή μας από απροσδόκητες συμπεριφορές όταν έχουμε αλλαγές στο λειτουργικό σύστημα.

1.10.5.Δραστηριότητες (Activities)

Οι δραστηριότητες επιτρέπουν στις εφαρμογές να καλούν η μία την άλλη, να επαναχρησιμοποιούν η μία την άλλη και να επιτρέπουν την αντικατάσταση ή βελτίωση των μεμονωμένων κομματιών του συστήματος όποτε ο χρήστης το θέλει. Οι δραστηριότητες συνήθως τρέχουν ως ξεχωριστές διαδικασίες, με το δικό τους UID, και κατά συνέπεια δεν έχουν πρόσβαση στα δεδομένα του

καλών (A-side) εκτός από ορισμένα δεδομένα που δίνονται στην πρόθεση που χρησιμοποιείται για να καλέσει την δραστηριότητα.

Οι δραστηριότητες δεν μπορούν να στηριχθούν σε φίλτρα εκπομπής προθέσεων (tag <intent-filter> στο AndroidManifest.xml) για να εξασφαλίσουν ότι ο καλών δεν θα περάσει στην εφαρμογή λανθασμένα δεδομένα πρόθεσης. Αυτή είναι μία κλασσική και συχνή πηγή λαθών. Από την άλλη πλευρά, οι δημιουργοί των δραστηριοτήτων μπορεί να στηριχθούν στους ελέγχους δικαιωμάτων σαν έναν αποδοτικό μηχανισμό ασφάλειας. Θέτοντας την ιδιότητα android:permission θα αποτρέψει προγράμματα που δεν έχουν την συγκεκριμένη άδεια να ξεκινήσουν απευθείας την συγκεκριμένη δραστηριότητα. Ορίζοντας ένα Manifest δικαίωμα που ο καλών πρέπει να έχει, δεν κάνεις το σύστημα να επιβάλει ένα φίλτρο εκπομπής και δεν εξασφαλίζεις ότι δεν θα υπάρξουν προθέσεις με απροσδόκητα δεδομένα, γι' αυτό το λόγο θα πρέπει πάντα να επικυρώνονται τα δεδομένα εισόδου.

Αυτός ο κώδικας παρουσιάζει την έναρξη μίας δραστηριότητας με μία πρόθεση. Ο Activity Manager θα αποφασίσει να ξεκινήσει τον Web Browser για να το χειριστεί, διότι ο Web Browser είναι μία δραστηριότητα που έχει καταχωρηθεί στο σύστημα με το κατάλληλο φίλτρο εκπομπής προθέσεων.

```
Intent i = new Intent(Intent.ACTION_VIEW);
i.setData(Uri.parse(http://www.isecpartners.com));
this.startActivity(i);
```

Ο παρακάτω κώδικας καταδεικνύει και φανερώνει τον καταναγκασμό της δραστηριότητας του Web Browser για να χειριστεί και μία πρόθεση με μία δράση και δεδομένα που δεν επιτρέπονται με βάση το φίλτρο εκπομπής προθέσεων.

```
//Το φίλτρο εκπομπής προθέσεων δεν συμφωνεί με αυτό την δράση
Intent I = new Intent("Cat-Farm Aardvark Pidgen");
//Το φίλτρο εκπομπής προθέσεων δεν συμφωνεί με αυτό το URI scheme
i.setData("marshmaellow:potatochip?");
//Η δραστηριότητα Web Browser πρόκειται να το πάρει οπωσδήποτε!
i.setComponent(new ComponentName("com.android.browser",
"com.android.browser.BrowserActivity"));
this.startActivity(i);
```

Εάν τρέξετε αυτό τον κώδικα θα δείτε την έναρξη της δραστηριότητας Web Browser, αλλά ο Browser θα αγνοήσει αυτή την παράξενη πρόθεση.

Ο παρακάτω κώδικας δείχνει ένα AndroidManifest παράδειγμα που ορίζει μια δραστηριότητα με όνομα ".BlankShoppingList". Αυτή η δραστηριότητα καθορίζει τον τρέχοντα κατάλογο αγορών και δίνει στον χρήστη έναν κενό κατάλογο για να αρχίσει τις επιλογές του. Επειδή το καθάρισμα είναι κάτι καταστρεπτικό, και συμβαίνει χωρίς επιβεβαίωση χρηστών, αυτή η δραστηριότητα θα πρέπει να περιοριστεί μόνο στους αξιόπιστους χρήστες. Το δικαίωμα "ACCESS_SHOPPING_LIST" επιτρέπει στα προγράμματα να διαγράψουν ή να προσθέσουν στοιχεία στον κατάλογο αγορών. Η περιγραφή του δικαιώματος αυτού εξηγεί επίσης στους χρήστες ότι χορηγώντας το δικαίωμα αυτό σε μία εφαρμογή δίνει στην εφαρμογή αυτή την δυνατότητα να διαβάσει και να αλλάξει τον κατάλογο αγορών.

```
<activity
      android:name=".BlankShoppingList"
      android:permission="com.example.ACCESS_SHOPPING_LIST"
      <intent-filter>
            <action
            android:name="com.example.shopping.CLEAR_LIST"/>
      </intent-filter>
</activity>
```

Όταν ορίζουμε δραστηριότητες, εκείνες που ορίζονται χωρίς κάποιο συγκεκριμένο φίλτρο εκπομπής προθέσεων ή μία android:exported ιδιότητα δεν είναι δημόσια προσιτές, που σημαίνει ότι άλλες εφαρμογές δεν μπορούν να τις ξεκινήσουν με Context.startActivity(Intent intent). Αυτές οι δραστηριότητες είναι πιο ασφαλής από όλες, όμως οι άλλες εφαρμογές δεν θα έχουν την δυνατότητα να επαναχρησιμοποιήσουν τις δραστηριότητες αυτές.

Οι μηχανικοί ανάπτυξης λογισμικού θα πρέπει να είναι προσεκτικοί όχι μόνο κατά την υλοποίηση των δραστηριοτήτων αλλά και κατά τη έναρξη των δραστηριοτήτων αυτών. Θα πρέπει να αποφεύγετε η εισαγωγή δεδομένων στις προθέσεις που χρησιμοποιούνται για την έναρξη δραστηριοτήτων που θα μπορούσαν να τραβήξουν το ενδιαφέρον των επιτιθέμενων. Ένας κωδικός πρόσβασης ή περιεχόμενα μηνύματος θα ήταν πρωταρχικό παράδειγμα

δεδομένων που δεν θα πρέπει να συμπεριλαμβάνονται. Παραδείγματος χάριν, ένα Malware θα μπορούσε να καταχωρηθεί με υψηλότερης προτεραιότητας φίλτρο εκπομπής προθέσεων και να καταλήξει να σταλούν τα ευαίσθητα δεδομένα του χρήστη στην δραστηριότητα του επιτιθέμενου αντί της κανονικής.

Κατά την έναρξη μίας δραστηριότητας εάν γνωρίζετε την λειτουργία που θέλετε να ξεκινήσετε, μπορείτε να το διευκρινίσετε αυτό στην πρόθεση με την κλήση της μεθόδου setComponent(). Αυτό αποτρέπει το σύστημα να ξεκινήσει κάποια άλλη δραστηριότητα σαν απάντηση στην πρόθεση σας. Ακόμα και σε αυτή την κατάσταση, είναι ακόμα επικίνδυνο να περαστούν ευαίσθητα προσωπικά δεδομένα στην πρόθεση. Μπορείτε να σκεφτείτε την πρόθεση που χρησιμοποιείται για την έναρξη μίας δραστηριότητας όπως τα command line arguments ενός προγράμματος που και σε αυτή την περίπτωση δεν θα πρέπει να περιέχουν ευαίσθητα δεδομένα.

1.10.6.Μεταδόσεις (Broadcasts)

Οι μεταδόσεις είναι ένας τρόπος ώστε οι εφαρμογές και τα τμήματα του συστήματος να επικοινωνήσουν ασφαλώς και αποτελεσματικά. Τα μηνύματα στέλνονται ως προθέσεις και το σύστημα τα διαχειρίζεται, συμπεριλαμβάνοντας το ξεκίνημα των δεκτών και επιβάλλοντας δικαιώματα.

1.10.6.1.Λήψη εκπεμπόμενων προθέσεων

Οι προθέσεις μπορεί να είναι μεταδόσεις σε δέκτες εκπεμπόμενων προθέσεων (Broadcast Receivers), που επιτρέπει την ανταλλαγή μηνυμάτων μεταξύ των εφαρμογών. Με την καταχώρηση ενός δέκτη εκπεμπόμενων προθέσεων στο AndroidManifest.xml της εφαρμογής σας, η κλάση του δέκτη θα ξεκινήσει όταν κάποιος ξεκινήσει μία μετάδοση. Ο Activity Manager χρησιμοποιεί τα φίλτρα εκπεμπόμενων προθέσεων των καταχωρημένων εφαρμογών για να καθορίσει ποιο πρόγραμμα θα χρησιμοποιήσει για κάθε δεδομένη μετάδοση. Όπως συζητήσαμε και παραπάνω, τα φίλτρα εκπεμπόμενων προθέσεων δεν είναι ένας μηχανισμός ασφάλειας. Όπως με τις δραστηριότητες, ένας αποστολέας μίας μετάδοσης μπορεί να στείλει σε ένα δέκτη μια πρόθεση, που δεν θα περνούσε το φίλτρο, ορίζοντας ρητά τον δέκτη. Οι δέκτες θα πρέπει να

είναι έτοιμοι να αντιμετωπίσουν απροσδόκητες προθέσεις ή λανθασμένα δεδομένα. Όπως απαιτείται για ασφαλή προγραμματισμό IPC, τα προγράμματα θα πρέπει προσεκτικά να επικυρώνουν τα δεδομένα εισόδου.

Οι δέκτες εκπεμπόμενων προθέσεων (Broadcast Receivers) ορίζονται στο AndroidManifest.xml με την ετικέτα <receiver>. Εξορισμού δεν μπορούν να κληθούν από άλλα προγράμματα, αλλά αυτό αλλάζει θέτοντας την ιδιότητα android:exported=”true”. Όπως και στις δραστηριότητες, οι προθέσεις που λαμβάνει ο δέκτης μπορεί να μην ταιριάζει με το φίλτρο που έχουν καταχωρήσει. Για να μπορέσουμε να περιορίσουμε ποιος μπορεί να στείλει στον δέκτη μας μια πρόθεση χρησιμοποιούμε την ιδιότητα android:permission στην ετικέτα του δέκτη για να ορίσουμε ένα συγκεκριμένο δικαίωμα. Όταν οριστεί ένα δικαίωμα για έναν δέκτη, ο Activity Manager είναι υπεύθυνος να επικυρώσει ότι ο αποστολέας έχει το συγκεκριμένο δικαίωμα πριν παραδώσει την πρόθεση. Τα δικαιώματα είναι ο σωστός τρόπος να εξασφαλίσουμε ότι ο δέκτης παίρνει μόνο τις προθέσεις από κατάλληλους αποστολείς, όμως δεν εξασφαλίζουν τις ιδιότητες των προθέσεων που θα λάβουν.

1.10.6.2.Ασφαλή αποστολή εκπεμπόμενων προθέσεων

Κατά την μετάδοση, οι μηχανικοί ανάπτυξης λογισμικού σε Android συμπεριλαμβάνουν ορισμένες ευαίσθητες πληροφορίες. Εάν τα δεδομένα που στέλνουμε είναι ευαίσθητα, θα πρέπει να είναι προσεκτικοί ως προς το ποίος θα λάβει τα δεδομένα αυτά. Ο απλούστερος τρόπος για να προστατεύσουμε τα δεδομένα σε ένα δυναμικό σύστημα είναι η απαίτηση από τον δέκτη ενός ειδικού δικαιώματος ή άδειας. Με το πέρασμα ενός Manifest δικαιώματος (Receiver Permission είναι το όνομα της παραμέτρους) στις μεθόδους του BroadcastIntent(), μπορούμε να απαιτήσουμε από τους παραλήπτες να έχουν την συγκεκριμένη άδεια. Αυτό δίνει στους μηχανικούς ανάπτυξης λογισμικού Android την δυνατότητα να έχουν τον έλεγχο σχετικά με το ποιες εφαρμογές μπορούν να λάβουν κάθε πρόθεση. Κατά συνέπεια, οι σχεδιαστές λογισμικού, όταν αποστέλλουμε ευαίσθητα δεδομένα, θα πρέπει να χρησιμοποιούν αυτό τον μηχανισμό.

Για παράδειγμα, μία εφαρμογή σύντομων μηνυμάτων μπορεί να θέλει να ενημερώσει άλλες ενδιαφερόμενες εφαρμογές, όταν έχει λάβει ένα σύντομο

μήνυμα, στέλνοντας τους μία πρόθεση. Μπορούμε να περιορίσουμε τους αποδέκτες αυτής της πρόθεσης μόνο στις εφαρμογές που έχουν το δικαίωμα RECEIVE_SMS, ορίζοντας το δικαίωμα αυτό κατά την αποστολή της πρόθεσης. Εάν μία εφαρμογή στείλει τα περιεχόμενα ενός σύντομου μηνύματος σε άλλες εφαρμογές, στέλνοντας μία πρόθεση χωρίς την απαίτηση του δικαιώματος RECEIVE_SMS, τότε οποιαδήποτε μη εξουσιοδοτημένη εφαρμογή θα μπορούσε να λάβει τα δεδομένα αυτά, δημιουργώντας έτσι μία τρύπα στην ασφάλεια. Οι εφαρμογές μπορούν να λάβουν τις εκπεμπόμενες προθέσεις χωρίς κάποιο ιδιαίτερο δικαίωμα. Επομένως, οι εφαρμογές θα πρέπει να απαιτούν από τους αποδέκτες να έχουν ένα συγκεκριμένο δικαίωμα πριν την αποστολή μίας πρόθεσης που περιέχει ευαίσθητα δεδομένα.

1.10.6.3.Sticky Μεταδόσεις (Sticky Broadcasts)

Οι Sticky μεταδόσεις είναι συνήθως ενημερωτικές και σχεδιασμένες για να πουν σε άλλες διαδικασίες πράγματα σχετικά με την κατάσταση του συστήματος. Οι εφαρμογές θα πρέπει να έχουν το δικαίωμα BROADCAST_STICKY για να στείλουν ή να αφαιρέσουν μία Sticky πρόθεση. Δεν μπορεί να απαιτηθεί μία συγκεκριμένη άδεια ή δικαιώματα κατά την αποστολή μίας Sticky πρόθεσης. Επομένως, δεν θα πρέπει να χρησιμοποιούνται για ανταλλαγή ευαίσθητων δεδομένων. Επίσης, οποιοδήποτε εφαρμογή έχει το δικαίωμα BROADCAST_STICKY μπορεί να αφαιρέσει μία Sticky πρόθεση που έχει αποστείλει κάποια άλλη εφαρμογή. Κατά συνέπεια, θα πρέπει να σκεφτόμαστε πριν παραχωρήσουμε το δικαίωμα αυτό σε μία εφαρμογή.

1.10.7.Υπηρεσίες (Services)

Οι υπηρεσίες είναι διαδικασίες που τρέχουν στο παρασκήνιο για μεγάλο χρονικό διάστημα και παρέχονται από το Android για την υποστήριξη εργασιών παρασκηνίου, όπως την αναπαραγωγή μουσικής κ.α. Οι υπηρεσίες μπορούν να αρχίσουν με μία πρόθεση και προαιρετικά να επικοινωνήσουν μέσω του Binder Interface καλώντας τη μέθοδο bindService() της κλάσης Context. Οι υπηρεσίες είναι παρόμοιες με τους δέκτες εκπεμπόμενων προθέσεων και τις δραστηριότητες ως προς το γεγονός ότι μπορούν να

ξεκινήσουν ανεξάρτητα από τα φίλτρα εκπεμπόμενων προθέσεων με την κλήση απευθείας μίας συγκεκριμένης υπηρεσίας που θέλουμε να εκκινήσουμε (εάν είναι exported). Επίσης, οι υπηρεσίες μπορούν να εξασφαλιστούν με την προσθήκη ενός ελέγχου άδειας στην ετικέτα τους στο AndroidManifest.xml. Η μακράς διάρκειας σύνδεση που παρέχεται από το bindService() δημιουργεί ένα γρήγορο IPC κανάλι βασισμένο στο Binder Interface. Τα Binder Interfaces μπορούν να ελέγχουν τα δικαιώματα των εφαρμογών που τα καλούν, επιτρέποντας την χρήση περισσότερων του ενός δικαιώματος την φορά ή διαφορετικών δικαιωμάτων σε διαφορετικά αιτήματα. Επομένως, οι υπηρεσίες παρέχουν πολλούς τρόπους επιβεβαίωσης ότι αυτός που τις καλεί είναι έμπιστος, παρόμοια με τις δραστηριότητες, τους δέκτες εκπεμπόμενων προθέσεων και τις Binder διεπαφές.

Εάν οι σχεδιαστές λογισμικού Android θέλουν να κάνουν κλήσεις υπηρεσιών με την εμπλοκή ευαίσθητων δεδομένων, όπως την αποθήκευση κωδικών πρόσβασης ή προσωπικών μηνυμάτων, θα πρέπει να επικυρώνουν ότι η υπηρεσία που σκοπεύουν να συνδεθούν είναι η σωστή και όχι κάποιο εχθρικό πρόγραμμα που δεν θα πρέπει να έχει πρόσβαση στις πληροφορίες αυτές. Εάν ο σχεδιαστής του λογισμικού ξέρει το ακριβές στοιχείο συστήματος με το οποίο θέλει να συνδεθεί, μπορεί να το προσδιορίσει ρητά στην πρόθεση που θα χρησιμοποιήσει για να συνδεθεί. Διαφορετικά, μπορεί να ελεγχτεί από την εφαρμογή το όνομα που παρέχεται στο onServiceConnected της κλάσης ServiceConnection (ComponentName name, IBinder service). Αυτός δεν είναι τόσο δυναμικός τρόπος.

Μέσω της μεθόδου onServiceConnected λαμβάνουμε το package name του στοιχείου συστήματος και αυτό το package name είναι συνδεδεμένο με τα δικαιώματα των εφαρμογών. Κατά συνέπεια, αυτό το όνομα θα μπορούσαμε να χρησιμοποιήσουμε για την επαλήθευση των δικαιωμάτων, όπως βλέπουμε παρακάτω:

```
Res = getPackageManager().checkPermission(
                                permToCheck, name.getPackageName());
```

1.10.8.Πάροχος περιεχομένου (Content Provider)

Το Android παρέχει μηχανισμό παροχής περιεχομένου που επιτρέπει στις εφαρμογές να μοιράζονται δεδομένα ή εικόνες ή ήχους κ.α. Ο πάροχος περιεχομένου χρησιμοποιείται από τις εφαρμογές για να εκθέσουν τα δεδομένα τους σε όλο το υπόλοιπο σύστημα, όπου το <provider> tag στο AndroidManifest.xml ορίζει ένα πάροχο ως διαθέσιμο και ορίζει δικαιώματα πρόσβασης σε αυτόν.

Ο οδηγός ασφάλειας του Android αναφέρει ότι υπάρχουν διαφορετικά δικαιώματα για γράψιμο και για διάβασμα («…έχοντας δικαίωμα γραψίματος δεν σημαίνει ότι έχεις την δυνατότητα διαβάσματος…» - Google Inc. 2008). Όσοι είναι εξοικειωμένοι με την SQL, πιθανώς υα συνειδητοποιούν ότι δεν είναι γενικά δυνατό να υπάρχουν write-only SQL queries. Για παράδειγμα, μία κλήση μίας updateQuery() ή deleteQuery() μπορεί να περιέχει μία επερώτηση διαβάσματος όταν εμπεριέχει κλήσεις τύπου "where …". Αυτό είναι δυνατόν να συμβεί ακόμα και αν ο καλών έχει μόνο δικαιώματα γραψίματος. Μία κλήση ενός updateQuery() ή deleteQuery() που χρησιμοποιεί "where …" δεν επιστρέφει απευθείας δεδομένα, όμως δίνει την δυνατότητα για αλλαγές με βάση τα αποθηκευμένα δεδομένα και εμμέσως αποκαλύπτει τα δεδομένα αυτά. Μέσω της παρατήρησης μίας σειράς updateQuery() ή deleteQuery() με έξυπνες κλήσεις "where …", κάποια εφαρμογή που δεν έχει δικαίωμα διαβάσματος μπορεί πολύ αργά να συμπεράνει τα δεδομένα που είναι αποθηκευμένα. Ο μηχανικός ανάπτυξης λογισμικού θα μπορούσε να δημιουργήσει ένα πάροχο περιεχομένου που δεν θα επιτρέπει περιπτώσεις όπως οι παραπάνω, όμως αυτό είναι μια επιλογή του σχεδιαστή του αντίστοιχου λογισμικού.

Η δήλωση των δικαιωμάτων αυτών γραψίματος και διαβάσματος γίνεται στο AndroidManifest.xml στην ετικέτα <provider>. Οι ετικέτες είναι η android:readPermission και android:writePermission. Αυτά τα δικαιώματα επιβάλλονται την στιγμή πρόσβασης στον πάροχο περιεχομένου.

Η δημιουργία ενός παρόχου περιεχομένου που μοιράζεται ανάμεσα σε πολλές εφαρμογές εμπεριέχει κάποιους κινδύνους, για παράδειγμα, θα πρέπει να υπάρχει σωστός συγχρονισμός πρόσβασης στα δεδομένα κ.α. Ο πάροχος

περιεχομένου είναι ένα πολύ ισχυρό όπλο, όμως δεν χρειαζόμαστε πάντα όλη αυτή την δύναμη.

1.10.8.1.Αποφυγή SQL injection

Για να μπορέσουμε να αποφύγουμε SQL injections, θα πρέπει να ξεχωρίσουμε τα SQL statement και τα δεδομένα που περιέχουν. Εάν τα δεδομένα που αποτελούν μέρος του SQL statement είναι λανθασμένα τότε το SQL injection που θα προκύψει μπορεί να δημιουργήσει σημαντικές τρύπες ασφάλειας για τα δεδομένα του χρήστη. Τα SQL injection αποφεύγονται εύκολα στις σύγχρονες πλατφόρμες όπως το Android με την χρήση επερωτήσεων που διαχωρίζουν τα δεδομένα από την λογική των επερωτήσεων. Οι μέθοδοι query(), update(), delete() (Content Provider) και οι μέθοδοι managedQuery() (Activities) υποστηρίζουν αυτό τον διαχωρισμό. Αυτές οι μέθοδοι παίρνουν την παράμετρο "String[] SelectionArgs", ένα σύνολο τιμών που αντικαθιστούν τους χαρακτήρες "?". Αυτό παρέχει ένα σαφή διαχωρισμό μεταξύ του περιεχόμενο του SQL statement στην παράμετρο "selection", και των δεδομένων που περιλαμβάνονται.

Θα μπορούσε να συμβεί SQL conjection στις περιπτώσεις που λαμβάνονται δεδομένα και μετά χρησιμοποιούνται σε επερωτήσεις, όπου εννοούμε δεδομένα από Binder Interfaces ή από εκπεμπόμενες προθέσεις ή από υπηρεσίες και δραστηριότητες, όπου θα μπορούσαν να αποτελούν πιθανοί στόχοι για διάφορα malware. Πάντα θα πρέπει να είστε προσεκτική για πιθανά SQL injection, αλλά ιδιαίτερη προσοχή πρέπει να δίνεται στα σημεία του κώδικα όπου τα στοιχεία και δεδομένα που θα χρησιμοποιηθούν στο SQL statement προέρχονται από μακρινές πηγές (RSS feeds, web pages, κ.α.).

1.10.9.Αντανάκλαση προθέσεων (Intent Reflection)

Ένας κοινός ιδιωματισμός της επικοινωνία στον Android είναι η λήψη μίας επανάκλησης (callback) μέσω μίας πρόθεσης. Σαν ένα πρακτικό παράδειγμα αυτού του ιδιωματισμο, θα μπορούσαμε να κοιτάξουμε τον Διαχειριστή Θέσης (Location Manager), που είναι ένα προαιρετικό χαρακτηριστικό. Ο διαχειριστής θέσης είναι μία Binder διεπαφή με την μέθοδο

LocationManager.addProximityAlert(). Αυτή η μέθοδος παίρνει ένα PendingIntent και αφήνει τον καλών να διευκρινίσει πώς θα τον ειδοποιήσει. Τέτοιες επανακλήσεις μπορούν να χρησιμοποιηθούν οπουδήποτε. Εάν το πρόγραμμα σας πρόκειται να στείλει μία πρόθεση όταν καλείται, πρέπει να αποφύγετε ο καλών να σας ξεγελάσει και να στείλετε μία πρόθεση που δεν θα του επιτρέπονταν. Η κλήση ενός άλλου για να στείλει μία πρόθεση για μένα αποτελεί την λεγόμενη αντανάκλαση προθέσεων. Αυτή η λειτουργία αποτελεί μία βασική χρήση της κλάσης PendingIntent που εισήχθη στο SDK 9.0.
Ένα η εφαρμογή σας παρέχει μια διεπαφή επιτρέποντας στον χρήστη της να ειδοποιηθεί λαμβάνοντας μία πρόθεση, θα πρέπει πιθανώς να την αλλάξετε να δέχεστε PendingIntent αντί για Intent. Τα PendingIntent στέλνονται με βάση την διαδικασία (process) που τα δημιούργησε. Το σύστημα θα διαχειριστεί και αντιμετωπίσει το αίτημα ως αίτημα του χρήστη της διεπαφής, δηλαδή του καλών. Αυτό μεταφέρει τον κίνδυνο από την υπηρεσία στον καλών. Πλέον, ο χρήστης της διεπαφής θα πρέπει να εμπιστευτεί την υπηρεσία με την δυνατότητα να στείλει για αυτό μία πρόθεση. Οι οδηγίες σχετικές με τα PendingIntent συστήνει σοφά να περιορίζουμε το PendingIntent στο συγκεκριμένο στοιχείο συστήματος που είναι σχεδιασμένο να στέλνει την επανάκληση με χρήση της setComponent(). Αυτό ελέγχει την αποστολή της πρόθεσης.

1.10.10.Δικαιώματα αρχείων (Files and Preferences)

Τα δικαιώματα των αρχείων που χρησιμοποιούνται στα συστήματα Unix είναι αυτά που συναντάμε και στο Android για τα συστήματα αρχείων που είναι σχεδιασμένα για να υποστηρίζονται, όπως το Root σύστημα αρχείων. Κάθε εφαρμογή έχει τη δική της περιοχή στο σύστημα αρχείων όπως ακριβώς τα προγράμματα που έχουν ένα Home Directory και έναν user ID. Μία δραστηριότητα ή μια υπηρεσία έχει πρόσβαση στον κατάλογο αυτό με τις μεθόδους getFilesDir(), getDir(), openFileOutput(), openFileInput(), getFileStreamPath(). Η παράμετρος "mode" χρησιμοποιείται για την δημιουργία ενός αρχείου με συγκεκριμένο σύνολο αδειών (που αντιστοιχούν στις άδειες αρχείων του UNIX). Για παράδειγμα, ένα "mode" = MODE_WORLD_WRITABLE | MODE_WORLD_READABLE δίνει την δυνατότητα

γραψίματος και διαβάσματος. Παρακάτω βλέπουμε ένα παράδειγμα δημιουργία ενός Word Readable αρχείο. Το FileOutputStream που παράγεται μπορεί να τροποποιηθεί μόνο από την ίδια την διαδικασία, αλλά μπορεί να διαβαστεί από οποιαδήποτε άλλη διαδικασία.

```
Fos = openFileOutput("PublicKey", Context.MODE_WORLD_READABLE);
```

Γενικά, θα πρέπει να είμαστε πολύ προσεκτικοί με κώδικα που δημιουργεί δεδομένα που είναι εύκολα προσβάσιμα και θα πρέπει να εξετάζουμε τα παρακάτω:

- Υπάρχουν ευαίσθητα δεδομένα γραμμένα σε αυτό το αρχείο?
- Θα μπορούσε μία αλλαγή στα στοιχεία να προκαλέσει κάτι δυσάρεστο ή απροσδόκητο να συμβεί?
- Εάν είναι Word writable files, καταλαβαίνετε ότι ένα κακό πρόγραμμα θα μπορούσε να γεμίσει την τηλεφωνική μνήμη και η εφαρμογή σας θα έπαιρνε όλο το φταίξιμο?

Προφανώς εκτελέσιμος κώδικας όπως scripts, libraries, configuration files, sites, folders θα ήταν κακό να είχαν World writable δικαιώματα. Αντίθετα, Log αρχεία, βάσεις δεδομένων ή εργασίες σε αναμονή θα ήταν κακό να είχαν World readable δικαιώματα.

1.10.10.1.Μαζική αποθήκευση (Mass storage)

Οι συσκευές είναι πιθανό να έχουν ένα περιορισμένο ποσό μνήμης στο εσωτερικό σύστημα αρχείων. Άλλες συσκευές μπορούν να υποστηρίξουν μεγαλύτερο σύστημα αρχείων με την χρήση εξωτερικών καρτών μνήμης. Η αποθήκευση στοιχείων σε αυτά τα συστήματα είναι λίγο μπερδεμένη. Για να διευκολύνει τους χρήστες να κινούν τα δεδομένα πέρα δώθε μεταξύ των φωτογραφικών μηχανών, τους υπολογιστές και το Android, χρησιμοποιούνται VFAT κάρτες. Το VFAT είναι ένα παλιό πρότυπο που δεν υποστηρίζουν τους ελέγχους προσπέλασης των Linux, έτσι τα στοιχεία που αποθηκεύονται σε αυτές είναι μη προστατευμένα.

Θα πρέπει να γίνεται ενημέρωση στους χρήστες ότι η μαζική αποθήκευση μοιράζεται σε όλα τα προγράμματα της συσκευής και να τους αποθαρρύνετε να αποθηκεύουν ευαίσθητα δεδομένα εκεί. Εάν θέλετε να αποθηκεύσετε εμπιστευτικά στοιχεία μπορείτε να τα κρυπτογραφήσετε και να αποθηκεύσετε

το κλειδί στην περιοχή αρχείων της εφαρμογής, και τα κρυπτογραφημένα δεδομένα μπορείτε να τα σώσετε στην κάρτα μνήμης. Εφόσον, ο χρήστης δεν θέλει να χρησιμοποιήσει την κάρτα μνήμης για να μετακινήσει δεδομένα σε άλλο σύστημα, τότε αυτή η λύση θα δουλέψει. Θα πρέπει να παρέχετε κάποιο μηχανισμό για να αποκρυπτογραφήσετε τα στοιχεία και να διαβιβάσετε τον κλειδί στον χρήστη εάν επιθυμεί να χρησιμοποιήσει την κάρτα μνήμης για να μετακινήσει τα εμπιστευτικά δεδομένα σε άλλο σύστημα.

1.10.11.Binder Διεπαφές

Το Binder είναι οδηγος συσκευών του πυρήνα (Kernel), που χρησιμοποιεί χαρακτηριστικά κοινή μνήμης του Linux για να επίτευξει αποδοτική και ασφαλή διαδιεργασιακή επικοινωνία. Οι υπηρεσίες συστημάτων είναι γνωστές ως διεπαφές Binder και το AIDL (Android Interface Definition Language) χρησιμοποιείται όχι μόνο για να ορίσουμε διεπαφές συστήματος, αλλά για να επιτρέψει στους σχεδιαστές λογισμικού να δημιουργήσουν Binder Πελάτες (Clients) και Διακομιστές (Servers). Η ορολογία μπορεί να δημιουρεί σύνχυση, αλλά οι Διακομιστές (Servers) χρησιμοποιούν την μέθοδο onTransct() της (android.os.Binder) και οι πελάτες (Clients) καλούν την δική τους transact() μέθοδο (android.os.IBinder). Και οι δύο μέθοδοι onTransact() και transact() χρησιμοποιούν αντικείμενα της κλάσης android.os.Parcel για να ανταλλάξουν δεδομένα με αποδοτικότητα. Το Android υποστηρίζει την διεπαφή Parcelable. Αντικέιμενα τύπου Parcelable μπορούν να μεταφερθούν μεταξύ των διεργασιών μέσω ενός Binder.

Στην πραγματικότητα, μία διεπαφή Binder είναι ένας περιγραφέας (Descriptor) που διατηρείται από μία Binder συσκευή (που είναι ένας οδηγός συσκευών πυρήνα). Η Binder διεργασιακή επικοινωνία μπορεί να χρησιμοποιηθεί για να περάσουμε ή να επιστρέψουμε δεδομένα, όπως αντικείμενα Parcelable, περιγραφείς αρχείου (File descriptors) και Binders. Κάνοντας χρήση μίας Binder διεπαφής έχουμε την δυνατότητα να χρησιμοποιήσουμε τις διεπαφές της (δηλαδή, κλήση της transact() και μια αντίστοιχη κλήση της onTransact() από την πλευρά του διακομιστή – Server), αλλά δεν εγγυάται ότι η υπηρεσία που χρησιμοποιεί αυτή την διεπαφή θα κάνει αυτό ακριβώς που ο καλών ζήτησε. Για παράδειγμα, οποιοδήποτε

πρόγραμμα μπορεί να έχει πρόσβαση στον Binder της υπηρεσίας συστήματος Zygote και να καλέσει την μέθοδο για τρέξει την εφαρμογή ως κάποιος άλλος χρήστης, όμως το Zygote θα αγνοήσει τις αιτήσεις που προέρχονται από μη εξουσιοδοτημένους χρήστες.

Η ασφάλεια του Binder έχει δύο βασικούς μηχανισμούς ασφάλειας: με τον έλεγχο της ταυτότητας του καλών και από την ασφάλεια των αναφορών Binder.

1.10.11.1.Ασφάλεια με έλεγχο των δικαωμάτων του καλών ή την ταυτότητα του

Όταν καλεστεί μία διεπαφή Binder, η ταυτότητα του καλών δίνεται με ασφάλεια από τον πυρήνα (kerner). Το Android συσχετίζει την ταυτότητα του καλών με το νήμα (thread) το οποίο θα εξυπηρετήσει το αίτημα. Αυτό επιτρέπει στον παραλήπτη να χρησιμοποιήσει τις παρακάτω μέθοδους checkCallingPermission (String permission) και checkCallingPermissionOrSelf (String permission) της κλάσης Context για να ταυτιποιήσει τα δικαιώματα του καλών. Οι εφαρμογές θέλουν συνήθως να επιβάλλουν τις άδειες που δεν έχουν στον καλών και η μέθοδος checkCallingPermissionOrSelf (String permission) επιτρέπει στην εφαρμογή να κληθεί ακόμα και αν στερείται τα κανονικά αναγκαία δικαιώματα. Οι Binder υπηρεσίες είναι ελεύθερες να κάνουν και άλλες Binder κλήσεις, αλλά όλες αυτές οι κλήσεις θα εμφανίζονται με την ταυτότητα της υπηρεσίας (UID και PID) κα όχι με την ταυτότητα του καλών.

Οι Binder υπηρεσίες έχουν πρόσβαση στην ταυτότητα του καλών χρησιμοποιόντας της μεθόδους getCallingUid() και getCallingPid()της κλάσης Binder. Αυτές οι μέθοδοι επιστρέφουν το UID και PID της διεργασίας που κάνει την Binder κλήση. Οι πληροφορίες ταυτότητας επικοινωνούνται με ασφάλεια στον χρήστη της Binder διεπαφής από τον πυρήνα (kernel).

Μια διεπαφή Binder μπορεί να υλοποιηθεί με πολλούς διαφορετικούς τρόπους. Ο πιο απλός τρόπος είναι η χρήση του μεταγλωττιστή AIDL. Στο εσωτερικό των μεθόδων, ο καλών αυτόματα συσχετίζεται με το νήμα που θα εξηπηρετήσει το αίτημα. Επομένως με την κληση Binder.getCallingUid()

ταυτοποιούμε τον καλών. Οι σχεδιαστές λογισμικού που χρησιμοποιούν την onTransact() θα πρέπει να συνειδητοποιήσουν ότι η ταυτότητα του καλών είναι συνδεδεμένη με το νήμα από το οποίο το αίτημα παραλήφθηκε και πρέπει αυτό να καθοριστεί πριν γίνει μεταπήδηση σε άλλο νήμα για να χειρισθεί το αίτημα. Μία κλήση της μεθόδου Binder.clearCallingIdentity() θα εμποδίσει τις μεθόδους getCallingUid() και getCallingPid() να είναι σε θέση να προσδιορίζουν τον καλών. Η μέθοδος checkPermission(String permission, int pid, int uid) της κλάσης Context χρησιμοποιείται για ελέγχους δικαιωμάτων ακόμα και αφού εχει καθαριστεί η ταυτότητα του καλών, εφόσον έχουμε αποθηκεύσει τις τιμές του UID και PID.

1.10.11.2.Ασφάλεια των Binder αναφορών

Οι Binder αναφορές μπορούν να μετακινηθούν μέσω μίας Binder διεπαφής. Οι μέδοδοι Parcel.writeStrongBinder() και Parcel.readStrongBinder() επιτρέπουν αυτό και προσφέρουν κάποιο επίπεδο ασφάλειας. Κατά την ανάγνωση μίας Binder αναφοράς από ένα Parcel με χρήση της readStrongBinder(), ο παραλήπτης είναι βέβαιως ότι ο συγγραφέας αυτού του Binder είχε μία αναοφορά στην Binder αναφορά που έλαβε. Αυτό αποτρέπει την πιθανότητα εξαπάτησης των Servers από τους καλούντες μέσω της αποστολής υποθετικών αριθμητικών τιμών τους οποίους θα χρησιμοποιήσει ο Server για να αντιπροσωπεύσει ένα Binder που ο καλών δεν έχει στην πραγματικότητα.

Η παροχή μίας αναφοράς σε ένα Binder δεν είναι πάντα θετικό. Επειδή ο κεντρικός Server μπορεί να πεί εάν ένας καλών έχει ένα συγκεκριμένο Binder και να αποφασίσει να μην δώσει Binder αναφορές, αυτό μπορεί να χρησιμοποιηθεί και σαν ένα μέτρο ασφάλειας. Όσο η Zygote δεν μπορεί να προστατεύσει τις Binder διεπαφές, πολλά Binder αντικείμενα κρατιούνται ιδιωτικά. Για να χρησιμοποιηθεί η ασφάλεια των αναφορών, οι διεργασίες θα πρέπει να εξασφαλήσουν την αποκάλυψη των Binder αντικειμένων. Μόλις λάβει μία διαδικασία ένα Binder, μπορεί να τον κάνει οτιδήποτε θέλει, να το περάσει σε άλλους ή να καλέσει την δικιά της transact() μέθοδο.

Τα Binder είναι συνολικά μοναδικά, που σημαίνει ότι αν δημιουργήσετε ένα Binder, κανένας αλλός δεν μπορεί να δημιουργήσει ένα ίδιο Binder με αυτό. Το Binder μπορεί να επικοινωνηθεί μεταξύ συνεργαζόμενων διεργασιών. Μία

υπηρεσία μπορεί να έχει έναν Binder σαν κλειδί για πολλούς καλούντες, γνωρίζοντας ότι μόνο εκείνοι που λαμβάνουν το κλειδί (ή τους έχει σταλεί) θα μπορούσαν αργότερα να το στείλουν πίσω. Αυτό λειτουργεί σαν μία απλή υλοποίηση κωδικού πρόσβασης. Ο διαχειριστής δραστηριοτήτων (Activity Manager) χρησιμοποιεί τις Binder αναφορές για τον έλεγχο και την διαχείριση των δραστηριότητων.

2.Η εφαρμογή

2.1.Περιγραφή κινητής συσκευής

Σχημα 4 : Τυπική συσκευή Android

Προκειμένου ο χρήστης να αλληλεπιδράσει με κάθε εφαρμογή android που έχει εγκατεστημένη στην συσκευή του πρέπει να έχει μια γενική γνώση των βασικών πλήκτρων με τα οποία είναι εφοδιασμένη μια κινητή συσκευή android. Μια τυπική συσκευή android φαίνεται στο σχήμα 1 και μια τυπική περιγραφή δίνεται παρακάτω.

Το μεγαλύτερο μέρος της συσκευής το καταλαμβάνει η οθόνη της συσκευής. Η οθόνη λειτουργεί με δύο τρόπους. Όπως κάθε οθόνη χρησιμεύει ως μια συσκευή εξόδου για να παρουσιάζεται ένα γραφικό περιβάλλον στον χρήστη. Ταυτόχρονα όμως αποτελεί και μια συσκευή εισόδου αφού είναι ευαίσθητη στην αφή (touchscreen) και μπορεί να ανιχνεύει συμβάντα όπως clicks, touches, gestures τα οποία παράγει ο χρήστης με απλή επαφή του με την οθόνη.

Στο κέντρο της συσκευής κάτω από την οθόνη βρίσκεται το D-pad το οποίο αποτελεί στην ουσία έναν σταυρό τεσσάρων κατευθύνσεων με ένα επιπλέον πλήκτρο στο κέντρο του. Με τα πλήκτρα κατευθύνσεων ο χρήστης είναι σε θέση να μετακινείται ανάμεσα στα διάφορα menu της εκάστοτε εφαρμογής. Με το κεντρικό πλήκτρο του D-Pad μπορεί να ενεργήσει έτσι ώστε να ενεργοποιήσει ένα επιλεγμένο αντικείμενο (πχ να πατήσει το επιλεγμένο πλήκτρο). Πρέπει να σημειωθεί οτι αυτή είναι η συνήθης λειτουργία αυτών των πλήκτρων και από εφαρμογή σε εφαρμογή μπορεί να διαφοροποιείται. Επιπλέον η κινητή συσκευή περιλαμβάνει ένα πλήκτρο με την ετικέτα MENU το οποίο όταν πατηθεί εμφανίζει στην οθόνη ένα αναδυόμενο menu αν αυτό είναι διαθέσιμο από την εκάστοτε εφαρμογή. Εκτός από τα πλήκτρα που φαίνονται στο σχήμα κάθε συσκευή android είναι εφοδιασμένη με ένα πληκτρολόγιο QERTY για να μπορεί ο χρήστης να εισάγει χαρακτήρες.

2.2.Περιγραφή της εφαρμογής

Τώρα έχοντας εισαχθεί στο Android και στις βασικές του τεχνολογίες, είναι το κατάλληλο σημείο να δούμε αναλυτικότερα την εφαρμογή που υλοποιήθηκε στα πλαίσια αυτής της διπλωματικής εργασίας. Σε αυτή την εφαρμογή θα δούμε να εφαρμόζονται πολλά από τα βασικά χαρακτηριστικά του Android. Η εφαρμογή μας λειτουργεί με μία εφαρμογή ιστοχώρου που διαχειρίζεται δεδομένα τα οποία οι χρήστες μπορούν να τα διαχειριστούν μέσω της Android εφαρμογής από το κινητό τους τηλέφωνο. Ο βασικός σκοπός της εφαρμογής αυτής είναι η παρουσίαση μίας σύνθετης εφαρμογής βασισμένη σε πραγματικές απαιτήσεις. Η εφαρμογή μας είναι σχεδιασμένη για τεχνικούς που δουλεύουν σε μία εταιρεία που παρέχει τις υπηρεσίες της σε πελάτες που βρίσκονται σε πολλά διαφορετικά μέρη. Την μία μέρα οι χρήστες της εφαρμογής μας (δηλαδή, οι χρήστες του "Mobile Worker") μπορεί να βρίσκονται σε έναν τοπικό πελάτη και την επόμενη μέρα να βρίσκονται σε μία χώρα του εξωτερικού.

2.3.Βασικές απαιτήσεις της εφαρμογής

Παρακάτω παραθέτουμε κάποιες βασικές απαιτήσεις που απασχόλησαν την

υλοποίηση της εφαρμογής μας:

- Ο "Mobile Worker" δίνεται από την κεντρική διοίκηση της εταιρείας που φροντίζει να δίνει πρωτεραιότητες στις εργασίες και να αναθέτει τις εργασίες αυτές σε κάθε τεχνικό.
- Ο "Mobile Worker" έχει μία συσκευή Android και έχει πλήρη υπηρεσία δεδομένων, δηλαδή μία συσκευή που μπορεί να πλοηγηθεί σε μία ποικιλία από διαδικτυακές πληροφορίες. Η συσκευή πρέπει να έχει πρόσβαση στο διαδίκτυο για την μεταφορά των δεδομένων.
- Το κεντρικό σύστημα και οι "Mobile Workers" ανταλλάσουν δεδομένα μέσω μίας ασύρματης σύνδεσης στο διαδίκτυο της κινητής συσκευής Android.
- Μία επιχειρηματική απαίτηση που θέσαμε είναι η απόδειξη της ολοκλήρωσης μίας εργασίας με την καταγραφή μίας υπογραφής του πελάτη. Σίγουρα μία ηλεκτρονική υπογραφή θα ήταν προτιμότερη.
- Το σύστημα της κεντρική εταιρείας επιθυμεί να λαμβάνει τις πληροφορίες ολοκλήρωσης μίας εργασίας το συντομότερο δυνατόν για να μπορεί να προχωρήσει στην τιμολόγηση της.
- Ο "Mobile Worker" θα πρέπει να είναι σε θέσει να λαμβάνει και να έχει πρόσβαση άμεσα στις νέες πληροφορίες για τις εργασίες.
- Ο "Mobile Worker" χρειάζεται όσο γίνεται περισσότερες πληροφορίες σχετικά με το πρόβλημα που καλείται να επιλύσει.
- Ο "Mobile Worker" χρειάζεται την παροχή πληροφοριών σχετικά με τον τόπο της εργασίας και οδηγίες για να φτάσει εκεί.
- Ο "Mobile Worker" θα πρέπει να είναι σε θέση να γνωρίζει την λίστα με τα ονόματα, τα τηλέφωνα και το "status" ("AVAILABLE" ή "UNAVAILABLE") των υπόλοιπων "Mobile Worker".
- Ο "Mobile Worker" θα πρέπει να έχει την δυνατότητα να ανανεώνει το "status" του, από "AVAILABLE" σε "UNAVAIBLE" ή "UNAVAILABLE" σε "AVAILABLE".
- Τελευταίο όμως αρκετά σημαντική απαίτηση είναι η εφαρμογή να είναι εύκολη και απλή στην χρήση της.

Η ασφάλεια σε τέτοιου είδους εφαρμογές έχει δύο θέματα. Το πρώτο θέμα είναι η φυσική ασφάλεια της συσκευής. Η υπόθεση μας είναι ότι η ίδια η

συσκευή κλειδώνει και μόνο ο εξουσιοδοτημένος χρήστης μπορεί να την χρησιμοποιήσει. Για πολλούς αυτό το επίπεδο ασφάλειας δεν είναι αποδεκτό, όμως ο στόχος της συγκεκριμένης εφαρμογής δεν είναι η μελέτη της ασφάλειας. Το δεύτερο θέμα ασφάλειας είναι η ασφαλής διαβίβαση των δεδομένων μεταξύ της κινητής συσκευής Android και του κεντρικού συστήματος. Ο πιο απλός τρόπος για να διασφαλίσουμε αυτό είναι η χρήση Secure Sockets Layer (SSL) στην σύνδεση αυτή.

2.4.Πρότυπο δεδομένων (Data Model)

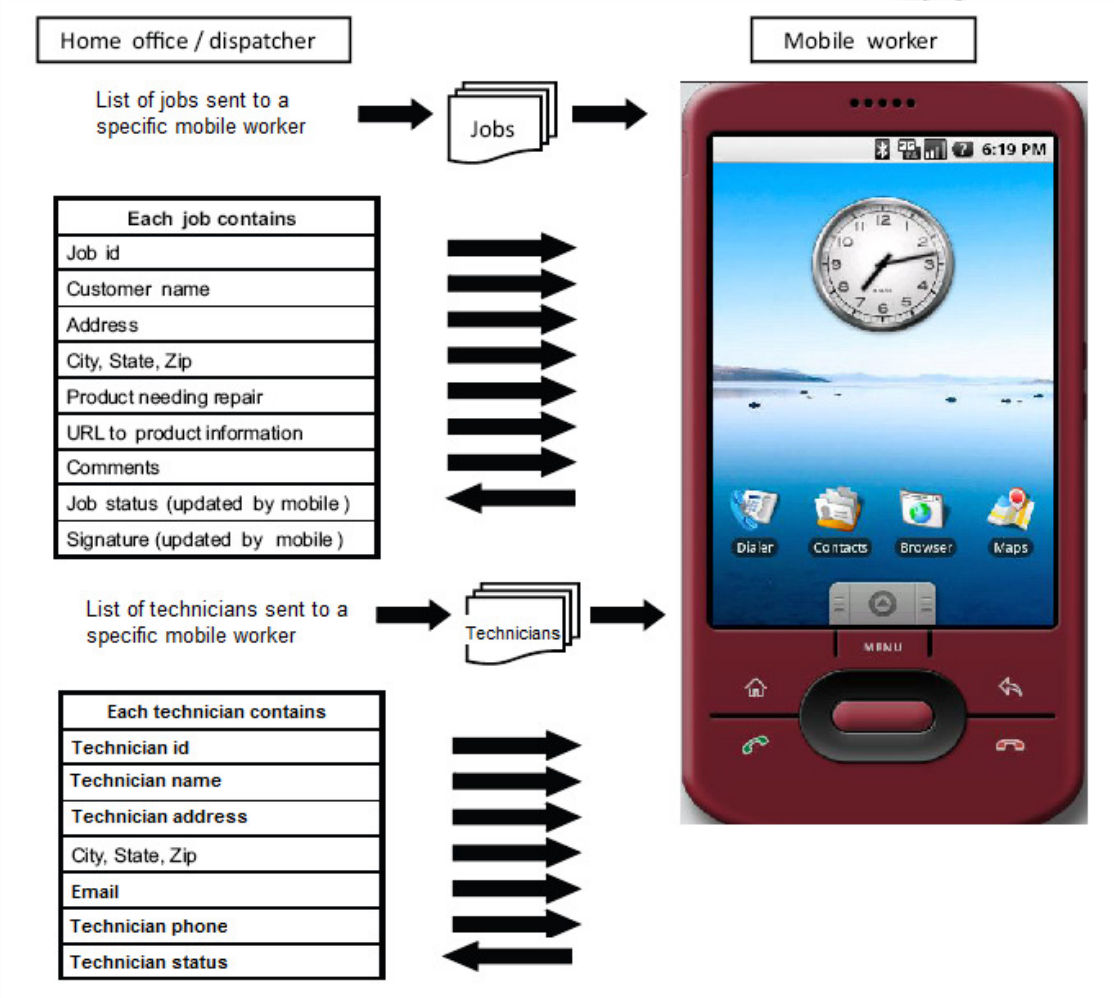
Ο όρος "Job" αναφέρεται σε μία συγκεκριμένη εργασία ή γεγονός όπου ο "Mobile Worker" συμμετέχει. Για παράδειγμα, το αίτημα για μία βλάβη ενός προϊόντος σε έναν πελάτη είναι ένα "Job". Το αίτημα για αναβάθμιση του Firmware ενός τηλεφώνου είναι και αυτό ένα άλλο "Job". Το κεντρικό σύστημα είναι αυτό που καταχωρεί μία ή περισσότερες "Jobs" σε κάθε τεχνικό. Υπάρχουν ορισμένα δεδομένα που αφορούν κάθε "Job", τα οποία θα μπορούσαν να βοηθήσουν τον τεχνικό στην αποτελεσματική και γρήγορη επίλυση του προβλήματος. Αυτές οι πληροφορίες εισάγονται από το κεντρικό σύστημα. Πού το κεντρικό σύστημα βρίσκει αυτές τις πληροφορίες δεν είναι θέμα το οποίο μας αφορά.

Στην συγκεκριμένη εφαρμογή υπάρχουν μόνο δύο πληροφορίες σχετικά με κάθε "Job" που ο "Mobile Worker" είναι υπεύθυνος για την υποβολή του στο κεντρικό σύστημα. Η πρώτη απαίτηση είναι ο "Mobile Worker" να μπορεί να διαβιβάσει στο κεντρικό σύστημα ότι ένα "Job" έχει κλίσει (CLOSED). Μία δεύτερη απαίτηση είναι η συλλογή μιας ηλεκτρονικής υπογραφής από τον πελάτη σαν επιβεβαίωση ότι η εργασία πραγματικά έχει ολοκληρωθεί.

Ο όρος "Technician" αναφέρεται στους υπόλοιπους τεχνικούς που κάνουν χρήση "Mobile Worker". Οι πληροφορίες που είναι σχετικές με τον τεχνικό εισάγονται από το κεντρικό σύστημα.

Στην συγκεκριμένη εφαρμογή υπάρχει μόνο μία πληροφορία σχετικά με κάθε "Technician" που ο "Mobile Worker" είναι υπεύθυνος για την υποβολή της στο κεντρικό σύστημα. Η απαίτηση είναι ότι ο "Mobile Worker" να μπορεί να ανανεώσει το "status" του. Μια επιπλέον απαίτηση είναι η δυνατότητα τηλεφωνικής επικοινωνίας ανάμεσα στους "Technicians".

Το παρακάτω σχήμα απεικονίζει αυτές τις ροές στοιχείων:



Σχημα 5 : Πρότυπο δεδομένων εφαρμογής

Φυσικά υπάρχουν πρόσθετες πληροφορίες σχετικά με κάθε "Job" που θα μπορούσαν να είναι χρήσιμες, όπως η αριθμός τηλεφώνου του πελάτη, η προσδοκώμενη διάρκεια της εργασίας, τα απαιτούμενα ανταλλακτικά που απαιτούνται για την επισκευή (με χρήση tracking numbers), κ.α. Ενώ όλα αυτά είναι αρκετά σημαντικά για μια εφαρμογή του πραγματικού κόσμου, στην συγκεκριμένη εφαρμογή δεν τα έχουμε συμπεριλάβει.

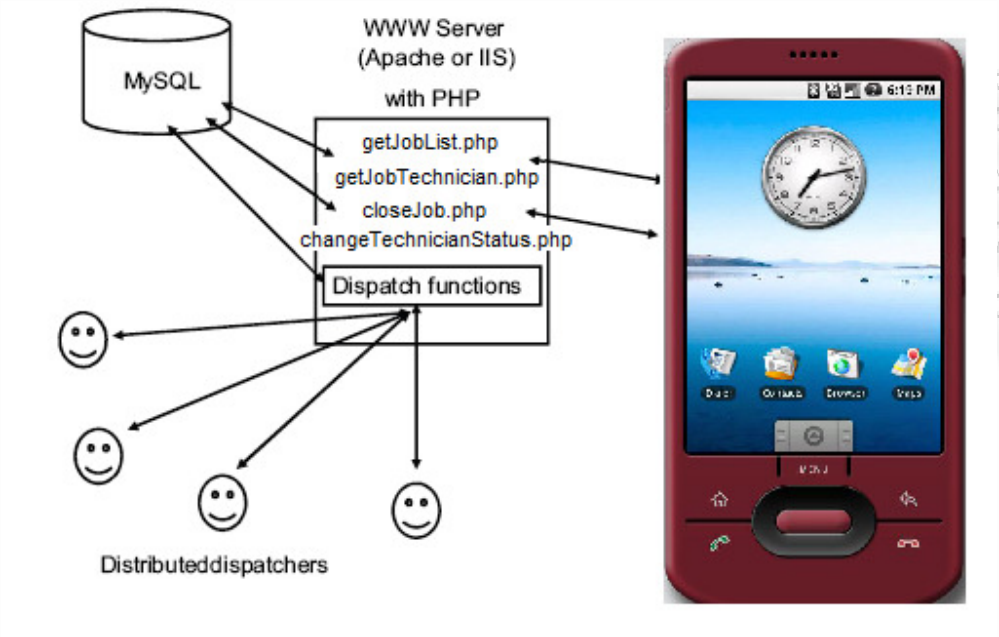
2.5.Αρχιτεκτονική της εφαρμογής

Τώρα που ξέρουμε ποίες οντότητες είναι υπεύθυνες για τα δεδομένα μας και με ποία κατεύθυνση ρέουν, θα εξετάσουμε πώς τα δεδομένα μας αποθηκεύονται και ανταλλάσονται.

Το κεντρικό σύστημα θα πρέπει να διαχειριστεί τα δεδομένα από διαφορετικούς "Mobile Workers". Το καλύτερο εργαλείο για αυτό τον λόγο είναι η χρήση μίας σχεσιακής βάσης δεδομένων. Οι επιλογές σε αυτό το θέμα είναι πολυάριθμες, αλλά λάβαμε την απόφαση να χρησιμοποιήσουμε MySQL, μίας δημοφιλής ανοιχτής βάσης δεδομένων. Όχι μόνο υπάρχουν διαφορετικοί "Mobile Workers", αλλά η οργάνωση που χτίσαμε αυτή την εφαρμογή είναι τέτοια ώστε το προσωπικό που καταχωρεί τα "Jobs" απο το κεντρικό σύστημα να μπορεί να βρίσκεται σε διαφορετική χώρα ή σε διαφορετικό Time zone. Λόγω αυτού αποφασίσαμε να φιλοξενήσουμε την βάση δεδομένων σε ένα "Data Center" όπου θα μπορεί να είναι προσβάσιμη μέσω μία "browser-based" εφαρμογής. Για την εφαρμογή μας το κεντρικό σύστημα είναι πολύ απλό, γραμμένο σε PHP.

Οι απαιτήσεις αποθήκευσης στην συσκευή είναι πολύ μέτριες. Οι εργασίες μπορούν να καταχωρηθούν οποιαδήποτε στιγμή επομένως ο "Mobile Worker" είναι καλό να ανανεώνει την λίστα με τα "Jobs" και τους "Technicians" ανά τακτά χρονικά διαστήματα. Επειδή έχουμε μικρή ανάγκη για κοινή μνήμη ανάμεσα σε πολλαπλές εφαρμογές, δεν έχουμε ανάγκη να χτίσουμε κάποιο πάροχο περιεχομένου (Content Provider). Επομένως, η απόφαση που έχουμε να κάνουμε είναι η χρήση ενός αποθηκευμένου XML αρχείου στα αρχεία συστήματος ως ένα προσωρινό αποθηκευτικό μέσο για την λίστα των "Jobs" και των "Technicians".

Η εφαρμογή χρησιμοποιεί HTTP για να ανταλλάξει στοιχεία με το κεντρικό σύστημα. Επίσης, η PHP χρησιμοποιείται για να δημιουργήσει τα Transactions για την ανταλλαγή των πληροφοριών. Ενώ μπορούν να υλοποιηθούν και πιο σύνθετα πρωτόκολλα όπως Simple Object Access Protocol (SOAP), αυτή η εφαρμογή απλά ζητάει ένα XML αρχείο της λίστας των "Jobs" και υποβάλει ένα αρχείο εικόνας που αντιπροσωπεύει την υπογραφή του πελάτη. Αυτή η αρχιτεκτονική φαίνεται και στο παρακάτω σχήμα:



Σχημα 6 : Αρχιτεκτονική εφαρμογής

Κάθε "Mobile Worker" θα πρέπει να δίνει την ταυτότητα του. Με βάση αυτή την ταυτότητα, η εφαρμογή μας μπορεί να ανακτήσει το σωστό κατάλογο εργασιών και από το κεντρικό σύστημα μπορούμε να καταχωρήσουμε εργασίες στον κατάλληλο τεχνικό. Επίσης, η κινητή συσκευή μπορεί να πρέπει να επικοινωνήσει με διαφορετικούς κεντρικούς υπολογιστές, ανάλογα με το πού βρίσκεται ο τεχνικός γεωγραφικά. Για παράδειγμα, ένας "Mobile Worker" μπορεί να θέλει να χρησιμοποιήσει τον κεντρικό υπολογιστή που βρίσκεται στο Σικάγο, ενώ ένας "Mobile Worker" που βρίσκεται στο Ηνωμένο βασίλειο μπορεί να θέλει να χρησιμοποιήσει τον κεντρικό υπολογιστή που βρίσκεται στο Cambridge. Λόγω αυτών των απαιτήσεων, ο χρήστης θα πρέπει να δίνει την ταυτότητα του και τον κεντρικό υπολογιστή που θέλει να επικοινωνήσει. Σε εμπορικές εφαρμογές, θα έπρεπε να υπάρχει μεγαλύτερη ασφάλεια για τα δεδομένα αυτά.

2.6.Ροή εφαρμογής

Σε αυτό το σημείο θα εξετάσουμε την ροή της εφαρμογής για να κατανοήσουμε καλύτερη την σχέση μεταξύ των λειτουργιών της εφαρμογής, των διεπαφών του χρήστη και των κλάσεων που χρησιμοποιούνται προγραμματιστικά για αυτές τις λειτουργίες. Με τον τρόπο αυτό θα

μπορέσουμε να βεβαιωθούμε ότι εφαρμογή ανταποκρίνεται στις απαιτήσεις των χρηστών και θα είναι πιο εύκολα προγραμματιστικά να ορίσουμε τις κλάσεις μας κατά την διάρκεια της υλοποίησης. Το παρακάτω σχήμα παρουσιάζει την ροή της εφαρμογής μέσω Android δραστηριοτήτων (Activity).

Σχημα 7 : Ροής εφαρμογής

Η εφαρμογή μας με βάση το σχήμα αυτό έχει τα παρακάτω βήματα:

- Application Launch: Η εφαρμογή επιλέγεται από την οθόνη έναρξης των εφαρμογών (Application Launch Screen) στην συσκευή του Android.
- Splash Screen (Splash Activity): Εμφανίζεται η Splash screen. Ορισμένες εφαρμογές χρειάζονται χρόνο για να την έναρξη τους και την αρχικοποίηση των δομών τους γι’ αυτό τον λόγο υλοποιήσαμε την συγκεκριμένη Splash screen.
- Main Screen (Field Service Activity): Στην οθόνη αυτή εμφανίζονται οι ρυθμίσεις (όνομα χρήστη και του όνομα κεντρικού υπολογιστή) του τελευταίου χρήστη της εφαρμογής. Επιπλέον, εμφανίζονται πέντε κουμπιά.
- Refresh Jobs (Refresh jobs Activity): Το κουμπί για “Refresh Jobs” ξεκινάει μία διαδικασία διαβάσματος των “Jobs” που υπάρχουν στον κεντρικό υπολογιστή που έχει ορίσει ο χρήστης. Κατά την διάρκεια του διαβάσματος

των εγγραφών αυτών εμφανίζεται ένα "ProgressDialog".

- Settings (Show Setting Activity): Το κουμπί "Settings" εμφανίζει στον χρήστη μία οθόνη μέσω της οποίας μπορεί να ορίσει το όνομα του χρήστη της εφαρμογής και του κεντρικού υπολογιστή.
- Manage Jobs (Manage Jobs Activity): Το κουμπί "Manage Jobs" εμφανίζει στον χρήστη μια οθόνη με όλες τις διαθέσιμες εργασίες που έχει στο όνομα του και δίνεται στον χρήστη η δυνατότητα να συνεχίσει σε επιπλέον βήματα εφόσον επιλέξει μία συγκεκριμένη εργασία.
- Show Job details (Show Job Activity): Επιλέγοντας μία συγκεκριμένη εργασία από την λίστα με της εργασίες που εμφανίζονται στην οθόνη "Manage Jobs", εμφανίζεται η οθόνη "Show Job Details" που περιέχει όλες τις λεπτομέρειες που υπάρχουν για την συγκεκριμένη εργασία. Αυτή η οθόνη εμφανίζει στον χρήστη όλες τις διαθέσιμες πληροφορίες για την εργασία αυτή και δίνει στον χρήστη τρείς επιλογές με τρία διαφορετικά κουμπιά.
- Map Job Location (Launch Google Maps): Το κουμπί "Map Job Location" εκτελεί μία γεωγραφική επερώτηση στην συσκευή μέσω μίας πρόθεσης (Intent). Ο προεπιλεγμένος διαχειριστής για αυτή την πρόθεση είναι η εφαρμογή χαρτών (Maps Application).
- Look up Product Info (Launch Browser): Επειδή ο "Mobile Worker" μπορεί να μην γνωρίζει πολλά για το προϊόν που καλείται να επιδιορθώσει, κάθε "Job" περιέχει ένα URL με πληροφορίες για το προϊόν. Αν ο χρήστης πατήσει το κουμπί "Lookup Product Info", εμφανίζεται ο διαδικτυακός πόρος που ορίζεται μέσω του URL σε ένα νέο "Browser". Αυτό ο διαδικτυακός πόρος μπορεί να είναι ένας απλός οδηγός χρήσης ή ένα εκπαιδευτικό video.
- Job Closed? : Η συμπεριφορά του κουμπιού αυτού εξαρτάται από την τρέχουσα κατάσταση της εργασίας.
- Capture Signature (Close Job Activity): Εάν το "Job" είναι "OPEN", γίνεται έναρξη της διαδικασίας ολοκλήρωση της εργασίας. Κατά την διαδικασία ολοκλήρωσης της εργασίας, εμφανίζεται ένα κενό πλαίσιο μέσα στο οποίο ο πελάτης μπορεί να υπογράψει για να επιβεβαιώσει την επιτυχή

ολοκλήρωση της εργασίας με την χρήση ενός ειδικού στυλό (χρειάζεται η Android συσκευή να είναι εφοδιασμένη με touch screen). Η επιλογές σε εκείνη της οθόνη είναι "Sign and Close" ή "Cancel". Εάν ο χρήστης επιλέξει την επιλογή "Sign and Close", η εφαρμογή στέλνει την υπογραφή σαν μία JPEG εικόνα στον κεντρικό υπολογιστή και ο κεντρικός υπολογιστής γυρνάει το συγκεκριμένο "Job" σε "CLOSED". Επιπλέον, το τοπικό αντίγραφό της "Job" μαρκάρεται ως "CLOSED". Εάν ο χρήστης επιλέξει την επιλογή "CANCEL", γίνεται ακύρωση της διαδικασίας ολοκλήρωσης της εργασίας.

- Display Signature (Launch Browser): Εάν το "Job" είναι "CLOSED", εμφανίζεται η αποθηκευμένη υπογραφή ολοκλήρωσης μέσω ενός νέου "Browser".
- Refresh technicians (Refresh Technicians Activity): Το κουμπί για "Refresh Technicians" ξεκινάει μία διαδικασία διαβάσματος των "Technicians" που υπάρχουν στον κεντρικό υπολογιστή που έχει ορίσει ο χρήστης. Κατά την διάρκεια του διαβάσματος των εγγραφών αυτών εμφανίζεται ένα "ProgressDialog".
- Manage Technicians (Manage Technicians Activity): Το κουμπί "Manage Technicians" εμφανίζει στον χρήστη μια οθόνη με όλους τους τεχνικούς και δίνεται στον χρήστη η δυνατότητα να συνεχίσει σε επιπλέον βήματα εφόσον επιλέξει ένα συγκεκριμένο τεχνικό.
- Show Technician details (Show Technician Activity): Επιλέγοντας ένα συγκεκριμένο τεχνικό από την λίστα με τους τεχνικούς που εμφανίζονται στην οθόνη "Manage Technicians", εμφανίζεται η οθόνη "Show Technician Details" που περιέχει όλες τις λεπτομέρειες σχετικά με τον τεχνικό. Αυτή η οθόνη εμφανίζει στον χρήστη όλες τις διαθέσιμες πληροφορίες για τον τεχνικό και δίνει στον χρήστη δύο επιλογές με δύο διαφορετικά κουμπιά.
- Call technician (Simple phone call): Το κουμπί "Call Technician" εκτελεί μία φωνητική κλήση μέσω μίας πρόθεσης (Intent).
- Change Technician Status (Change Technician Status Activity): Το κουμπί "Change Technician Status" κάνει έναρξη της διαδικασίας αλλαγής "status". Ο χρήστης μπορεί να επιλέξει το νέο "status" (AVAILABLE ή

UNAVAILABLE) και να επιλέξει “Save” ή “Cancel”. Εάν ο χρήστης επιλέξει την επιλογή “Save”, η εφαρμογή στέλνει το νέο “status” στον κεντρικό υπολογιστή και ο κεντρικός υπολογιστής αλλάζει το status του τεχνικού. Επιπλέον, το τοπικό αντίγραφό του “technician” μαρκάρεται με το νέο “status”. Εάν ο χρήστης επιλέξει την επιλογή “CANCEL”, γίνεται ακύρωση της διαδικασίας αλλαγής “status”.

Στην συνέχεια, αφού ήδη έχουμε κατανοήσει ποιες είναι οι απαιτήσεις της εφαρμογής μας και πώς θα μπορέσουμε να ικανοποιήσουμε τις απαιτήσεις αυτές σε επίπεδο λειτουργιών και ροής εφαρμογής, θα δούμε πως οι απαιτούμενες λειτουργίες υλοποιήθηκαν σε επίπεδο κώδικα.

2.7.Χάρτης του κώδικα

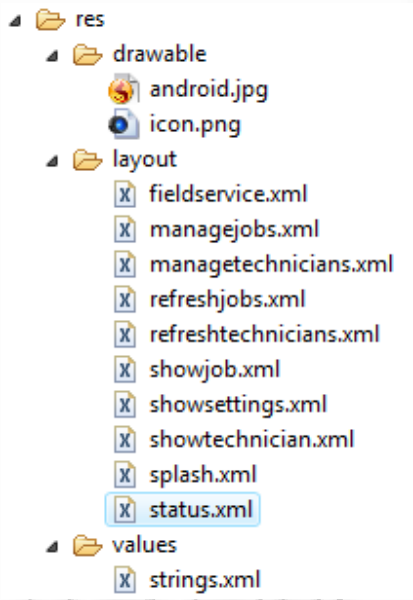
Ο κώδικας της εφαρμογής μας αποτελείται από 19 Java αρχεία, ένα από τα οποία είναι το R.java, το οποίο παράγεται αυτόματα με βάση τους πόρους της εφαρμογής μας. Σε αυτό το σημείο θα προσπαθήσουμε να κάνουμε μία γρήγορη εισαγωγή σε κάθε ένα από αυτά τα αρχεία:

Source Filename	Description
Splash.java	Activity που παρέχει την λειτουργία της splash screen
ShowSettings.java	Activity που παρέχει διαχείριση του ονόματος χρήση και της URL διεύθυνσης του κεντρικού υπολογιστή
FieldService.java	Activity που παρέχει την κεντρική οθόνη της εφαρμογής
RefreshJobs.java	Activity που αλληλεπιδρά με τον κεντρικό υπολογιστή για να διαβάσει την λίστα με τα "jobs"
RefreshTechnicians.java	Activity που αλληλεπιδρά με τον κεντρικό υπολογιστή για να διαβάσει την λίστα με τους "technicians"
ManageJobs.java	Activity που παρέχει πρόσβαση στην λίστα με τα "jobs"
ManageTechnicians.java	Activity που παρέχει πρόσβαση στην λίστα με τους "technicians"
ShowJob.java	Activity που παρέχει πληροφορίες σχετικά με μία συγκεκριμένη "job"
ShowTechnician.java	Activity που παρέχει πληροφορίες σχετικά με έναν συγκεκριμένο "technician"
CloseJob.java	Activity που συλλέγει την ηλεκτρονική υπογραφή και αλληλεπιδρά με τον κεντρικό υπολογιστή για να αποθηκεύσει την εικόνα και να μαρκάτει την εργασία σε "CLOSED"
ChangeTechnicianStatus.java	Activity που αλληλεπιδρά με τον κεντρικό υπολογιστή και αλλάζει το το "status" του τεχνικού
R.java	Αρχείο που δημιουργείται αυτόματα από το σύστημα
Prefs.java	Βοηθητική κλάση που συμπεριλαμβάνει τα SharedPreferences
TechnicianEntry.java	Κλάση που αναπαριστά ένα "technician". Συμπεριλαμβάνει βοηθητικές συναρτήσεις που χρησιμοποιούνται για πέρασμα αντικειμένων "TechnicianEntry" από την μια δραστηριότητα στην άλλη
TechnicianList.java	Κλάση που αναπαριστά ολόκληρη την λίστα με αντικείμενα "TechnicianEntry"
TechnicianListHandler.java	Κλάση που χρησιμοποιείται για το XML parsing δεδομένων των "technician"
JobEntry.java	Κλάση που αναπαριστά ένα "job". Συμπεριλαμβάνει βοηθητικές συναρτήσεις που χρησιμοποιούνται για πέρασμα αντικειμένων "JobEntry" από την μια δραστηριότητα στην άλλη
JobList.java	Κλάση που αναπαριστά ολόκληρη την λίστα με αντικείμενα "JobEntry"
JobListHandler.java	Κλάση που χρησιμοποιείται για το XML parsing δεδομένων των "jobs"

Σχημα 8 : Χάρτης κώδικα εφαρμογής

Η εφαρμογή στηρίζεται επίσης στα “Layout Resources” για τον ορισμό του

οπτικής διεπαφής με τον χρήστη. Εκτός από τα "Layout xml files", η εικόνα που χρησιμοποιείται από την δραστηριότητα Splash αποθηκεύεται σε ένα υποκατάλογο του καταλόγου res μαζί με την εικόνα "Android icon". Αυτή η εικόνα χρησιμοποιείται για την "Application Launch Screen".



Σχημα 9 : Layout xml Files της εφαρμογής

Σε αυτό το σημείο θα προσπαθήσουμε να κάνουμε μία γρήγορη εισαγωγή στα "Resource files" που έχουν χρησιμοποιηθεί για την εφαρμογή μας:

Filename	Description
Android.jpg	Εικόνα που χρησιμοποιείται από το Splash Activity
icon.jpg	Εικόνα που χρησιμοποιείται από τον Application Launcher
fieldservice.xml	Layout για την κεντρική οθόνη της εφαρμογή, Field Service Activity
managejobs.xml	Layout για την λίστα των "Jobs", Manage Jobs Activity
managetechnicians.xml	Layout για την λίστα των "Technicians", Manage Technicians Activity
refreshjobs.xml	Layout για αυτά που δείχνει η οθόνη όσο γίνεται ανανέωση της λίστας των "Jobs", Refresh Jobs Activity
refreshtechnicians.xml	Layout για αυτά που δείχνει η οθόνη όσο γίνεται ανανέωση της λίστας των "Jobs", Refresh Jobs Activity
showjob.xml	Layout για τις πληροφορίες ενός "Job", Show Job Activity
showsettings.xml	Layout για την οθόνη με τις ρυθμίσεις, Show Settings Activity
showtechnician.xml	Layout για τις πληροφορίες ενός "Technician", Show technician Activity
splash.xml	Layout για την splash οθόνη, Splash Activity
status.xml	Layout για την οθόνη επιλογής "technician" status, Change Technician Status Activity
strings.xml	Περιέχει το τίτλο του Application

Σχημα 10 : Resource Files της εφαρμογής

2.8.AndroidManifest.xml

Κάθε εφαρμογή Android χρειάζεται ένα "manifest file". Για να δούμε το

Manifest αρχείο της εφαρμογής μας:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.msi.manning.UnlockingAndroid">
    <application android:icon="@drawable/icon">
        <activity android:name=".Splash" android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".FieldService" >
        </activity>
        <activity android:name=".RefreshJobs" android:label="@string/app_name">
        </activity>
        <activity android:name=".ManageJobs" android:label="@string/app_name">
        </activity>
        <activity android:name=".ShowJob" android:label="@string/app_name">
        </activity>
        <activity android:name=".CloseJob" android:label="@string/app_name">
        </activity>
        <activity android:name=".ShowSettings" android:label="@string/app_name">
        </activity>
        <activity android:name=".RefreshTechnicians" android:label="@string/app_name">
        </activity>
        <activity android:name=".ManageTechnicians" android:label="@string/app_name">
        </activity>
        <activity android:name=".ShowTechnician" android:label="@string/app_name">
        </activity>
        <activity android:name=".ChangeTechnicianStatus" android:label="@string/app_name">
        </activity>

    </application>
    <uses-permission android:name="android.permission.INTERNET"></uses-permission>
</manifest>
```

Splash Activity είναι η αρχική δραστηριότητα της εφαρμογής μας

Intent Filter για την εμφάνιση του Main Launcher

Λίστα με τα Activities της εφαρμογής μας

Απαιτούμενα δικαιώματα για internet access

Σχημα 11 : AndroidManifest.xml της εφαρμογής

2.9.Συστατικά στοιχεία της εφαρμογής

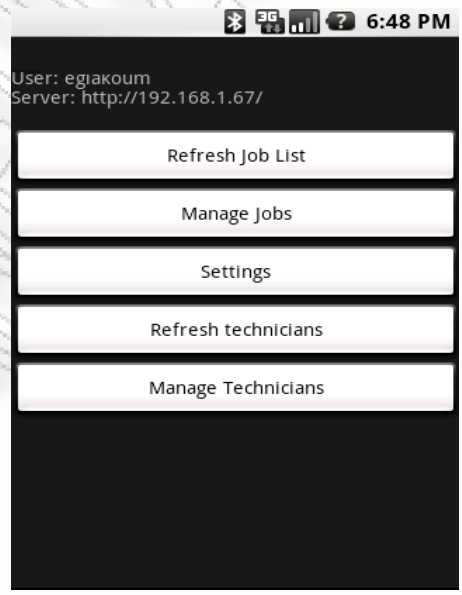
2.9.1.Splash Activity

Οι περισσότεροι είναι εξοικειωμένοι με "Splash Screen" μίας εφαρμογής (μια εικόνα που εμφανίζεται όσο φορτώνει μία εφαρμογή). Ενεργεί σαν μία κουρτίνα που κρύβει τις σημαντικές λειτουργίες που γίνονται πίσω από αυτή. Συνήθως οι "Splash screen" είναι ορατές έως ότου η εφαρμογή να είναι έτοιμη. Στην παρακάτω εικόνα μπορούμε να δούμε την splash screen που τρέχει στον εξομοιωτή του Android.

Σχημα 12 : Splash activity

2.9.2.Field Service Activity

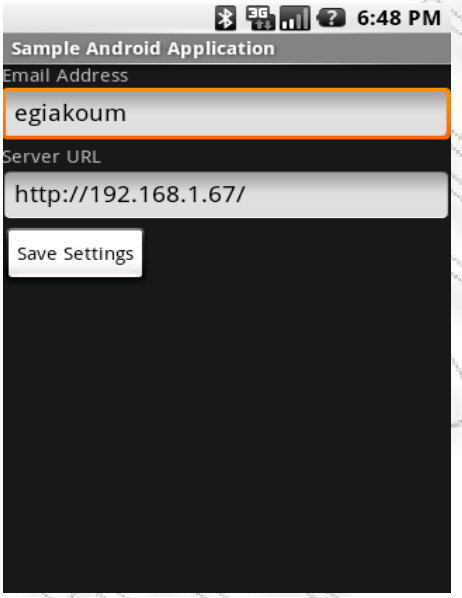
Ο στόχος της δραστηριότητας "Field Service" είναι να παραθέσει στον χρήστη όλες τις λειτουργίες του "Mobile Worker" και να του δώσει την δυνατότητα να επιλέξει με ευκολία μία από αυτές τις λειτουργίες. Συχνά, οι εφαρμογές που μπορούν να χρησιμοποιηθούν με το ένα χέρι είναι οι πιο εύχρηστες εφαρμογές για το κινητό. Ο χρήστης μέσα από την δραστηριότητα "Field Service" έχει την δυνατότητα να επιλέξει ανάμεσα στις παρακάτω λειτουργίες που βλέπουμε στην εικόνα



Σχημα 13 : Field service activity

2.9.3.Settings Activity

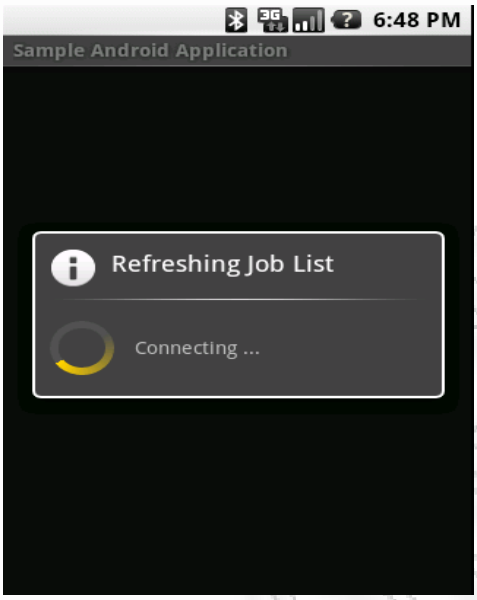
Όταν η χρήστης επιλέγει την επιλογή "Settings" γίνεται εκκίνηση μίας δραστηριότητας που επιτρέπει στον χρήστη να επιλέξει user ID (technician ID) και το URL του κεντρικού υπολογιστή, όπως βλέπουμε στην παρακάτω εικόνα.



Σχημα 14 : Setting activity

2.9.4.Refresh Jobs Activity

Η δραστηριότητα "Refresh Jobs" εκτελεί ένα σημαντικό κομμάτι της εφαρμογής μας. Όποτε ο χρήστης επιλέξει αυτή την λειτουργία, η δραστηριότητα αυτή διαβάζει από τον κεντρικό υπολογιστή όλη την λίστα με τις "Jobs". Η διεπαφή του χρήστη είναι εξαιρετικά απλή – μία μαύρη οθόνη και ένα ProgressDialog που ενημερώνει τον χρήση για την εξέλιξη της λειτουργίας όπως φαίνεται και στην εικόνα παρακάτω.



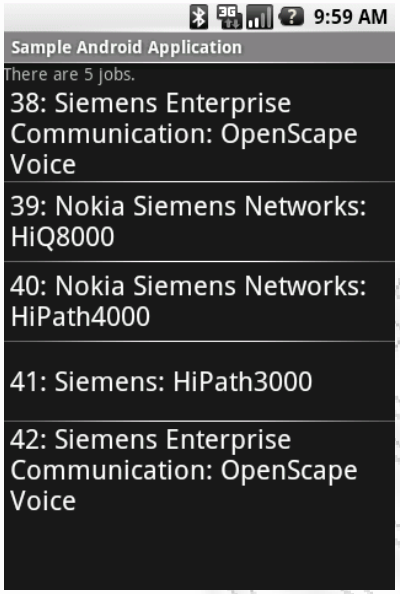
Σχημα 15 : Refresh jobs activity

2.9.5.Manage Jobs Activity

Η δραστηριότητα "Manage Jobs" εμφανίζει στον χρήστη την λίστα με τα "Jobs". Στην κορυφή της οθόνης έχουμε μία απλή περίληψη που δείχνει τον συνολικό αριθμό "Jobs" στην λίστα και κάθε μεμονωμένο "Job" απαριθμείται στην "ListView". Θα αναφέρουμε την σημασία της μεθόδου toString() της κλάσης JobEntry:

```
public String toString()
{
      return this._jobid + ": " + this._customer + ": " +
this._product;
}
```

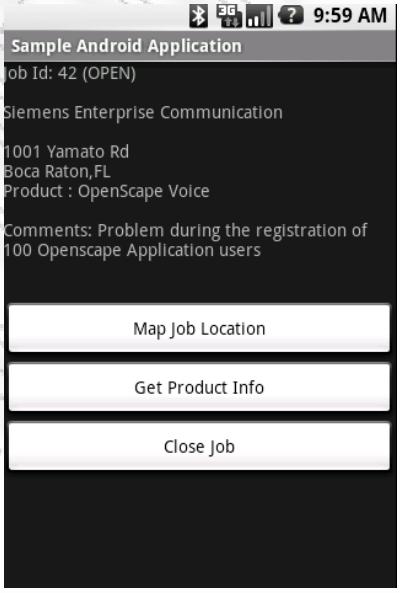
Η μέθοδος αυτή δημιουργεί την συμβολοσειρά που χρησιμοποιείται για την απεικόνιση ενός "JobEntry" στην "ListView", όπως φαίνεται και στο παρακάτω σχήμα.



Σχημα 16 : Manage jobs activity

2.9.6.Show Job Activity

Η δραστηριότητα "Show Job" είναι αναμφισβήτητα το πιο ενδιαφέρον κομμάτι της εφαρμογής μας και η πιο χρήσιμη οθόνη για τον χρήστη. Παρακάτω μπορούμε να παρατηρήσουμε την οθόνη αυτή.

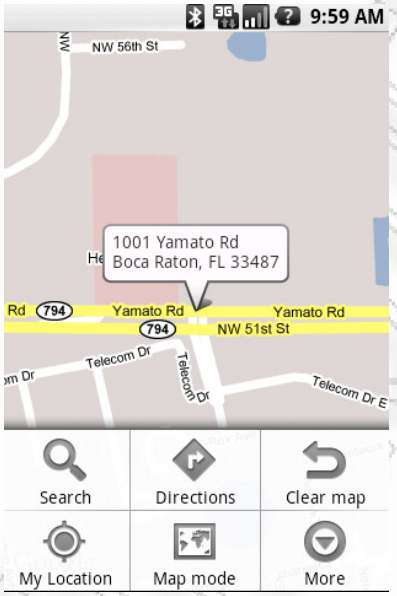


Σχημα 17 : Show job activity

Ένα "TextView" χρησιμοποιείται για την εμφάνιση των λεπτομερειών της κάθε εργασίας, όπως διεύθυνση, προϊόν που χρειάζεται επιδιόρθωση και τα σχόλια.

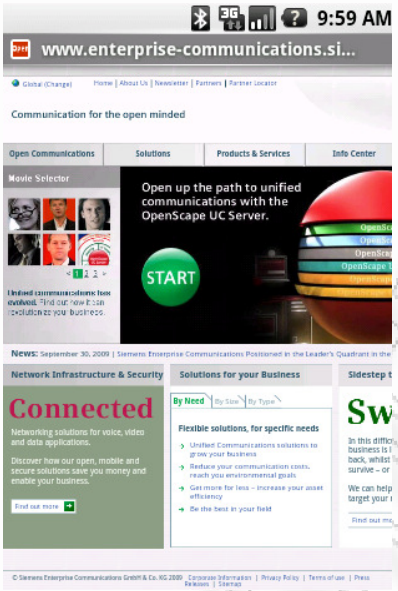
Το τρίτο κουμπί αλλάζει την ιδιότητα του ανάλογα με το "status" του "Job" ("OPEN" ή "CLOSED"). Εάν το "status" του "Job" είναι "CLOSED", η λειτουργία του τρίτου κουμπιού θα αλλάξει.

Για να υποστηρίξουμε την λειτουργία "Map Job Location", θα πρέπει να έχουμε μια νέα δραστηριότητα που θα παρουσιάσει την διεύθυνση της εργασίας σε έναν χάρτη όπως φαίνεται και στο παρακάτω οθόνη.



Σχημα 18 : Map job location

Το δεύτερο κουμπί "Get Product Info" ανοίγει έναν Browser μέσα από το οποίο ο χρήστης μπορεί να πάρει αναλυτικές πληροφορίες σχετικές με το προϊόν που καλείται να επιδιορθώσει, όπως φαίνεται στην παρακάτω οθόνη.

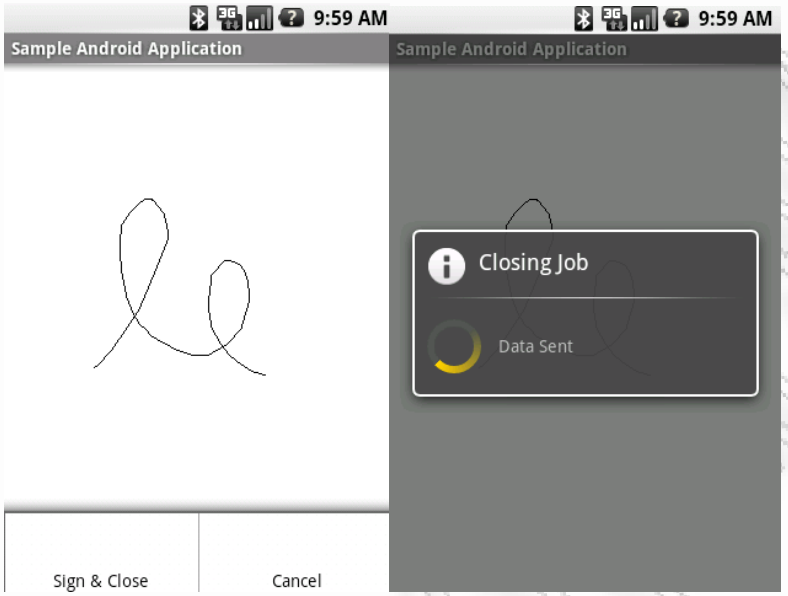


Σχημα 19 : Get product info

Το τρίτο κουμπί δίνει στον χρήστη την δυνατότητα να κλείσει μία εργασία (Close Job Activity) ή να δει την υπογραφή ολοκλήρωσης εργασίας εάν η εργασία είναι ήδη κλειστή.

2.9.7.Close Job Activity

Μετά την ολοκλήρωση μίας εργασίας, ο τεχνικός πρέπει να κλείσει την εργασία αυτή ζητώντας μία υπογραφή από τον πελάτη σαν πιστοποιητικό ολοκλήρωσης της εργασίας αυτής. Για το κλείσιμο της εργασίας η εφαρμογή εμφανίζει στον χρήστη μία κενή λευκή οθόνη στην οποία ο πελάτης πρέπει να υπογράψει χρησιμοποιώντας ένα ειδικό πενάκι για οθόνες αφής, επιβεβαιώνοντας με αυτό τον τρόπο την επιτυχή ολοκλήρωση της εργασίας. Στην συνέχεια, η υπογραφή αυτή στέλνεται για να φυλαχθεί στον κεντρικό υπολογιστή. Η παρακάτω οθόνες δείχνουν αυτή την διαδικασία.



Σχημα 20 : Close job activity

Αυτή η δραστηριότητα μπορεί να αναλυθεί σε δύο βασικές λειτουργίες:

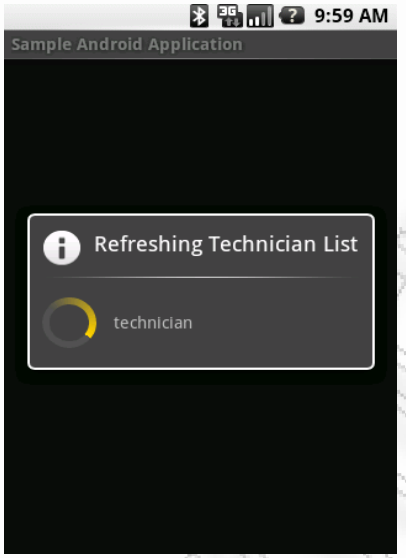
- Σύλληψη της υπογραφής
- Μεταφορά δεδομένων στον κεντρικό υπολογιστή

Το ενδιαφέρον είναι ότι το "User Interface" του χρήστη δεν έχει "Layout Resources". Όλα τα στοιχεία του "User Interface" παράγονται δυναμικά όπως μπορούμε να δούμε στον κώδικα του "CloseJob.java". Επιπλέον, το "ProgressDialog" που είχαμε χρησιμοποιήσει στην δραστηριότητα "Refresh Jobs" χρησιμοποιείται και από αυτή την δραστηριότητα για να ενημερώσει τον χρήστη ότι η υπογραφή στέλνεται στον κεντρικό υπολογιστή, εφόσον ο χρήστης έχει επιλέξει "Sign & Close". Εάν ο χρήστης επιλέξει "Cancel", η δραστηριότητα "Show Job" παίρνει ξανά τον έλεγχο.

2.9.8.Refresh Technicians Activity

Όποτε ο χρήστης επιλέξει αυτή την λειτουργία, η δραστηριότητα "Refresh Technicians" διαβάζει από τον κεντρικό υπολογιστή όλη την λίστα με τους "Technicians". Η διεπαφή του χρήστη είναι εξαιρετικά απλή – μία μαύρη οθόνη και ένα ProgressDialog που ενημερώνει τον χρήση για την εξέλιξη της λειτουργίας όπως φαίνεται και στην εικόνα παρακάτω (παρόμοια με την οθόνη

της δραστηριότητα “Refresh Jobs”).



Σχημα 21 : Refresh technicians activity

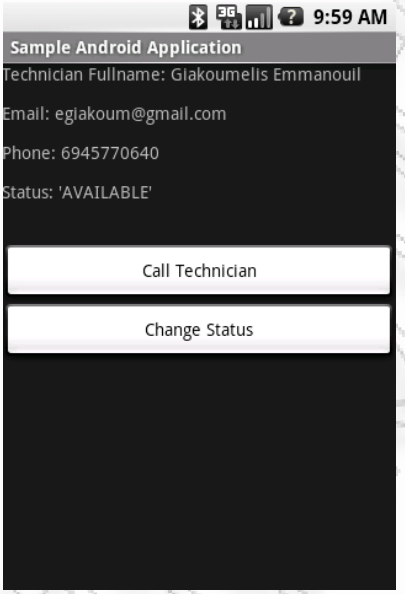
2.9.9.Manage Technicians Activity

Η δραστηριότητα “Manage Technicians” εμφανίζει στον χρήστη την λίστα με τους “Technicians”. Στην κορυφή της οθόνης έχουμε μία απλή περίληψη που δείχνει τον συνολικό αριθμό “Technicians” στην λίστα, όπως φαίνεται και στην παρακάτω οθόνη (παρόμοια λογική με δραστηριότητα “Manage Jobs”).

Σχημα 22 : Manage technicians activity

2.9.10.Show Technician Activity

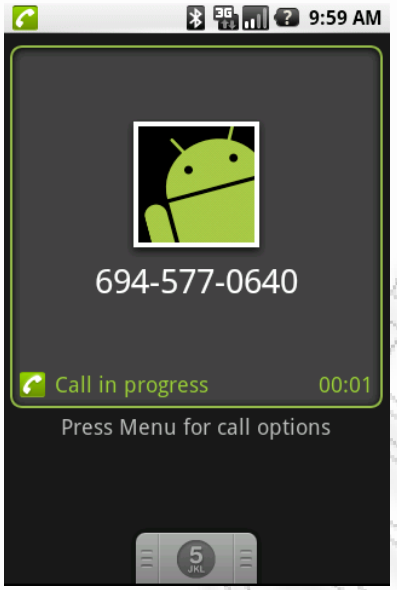
Η δραστηριότητα "Show Technician" είναι αναμφισβήτητα μία πολύ χρήσιμη οθόνη για τον χρήστη. Παρακάτω μπορούμε να παρατηρήσουμε την οθόνη αυτή.



Σχημα 23 : Show technician activity

Ένα "TextView" χρησιμοποιείται για την εμφάνιση των λεπτομερειών του κάθε τεχνικού, όπως ονοματεπώνυμο, ηλεκτρονική διεύθυνση, τηλέφωνο και "status". Το τρίτο κουμπί αλλάζει την ιδιότητα του ανάλογα με το "status" του "Technician" ("Available" ή "Unavailable").

Το πρώτο κουμπί "Call Technician" δίνει στον χρήστη την δυνατότητα να καλέσει οποιοδήποτε "Technician", όπως φαίνεται και στην παρακάτω οθόνη.

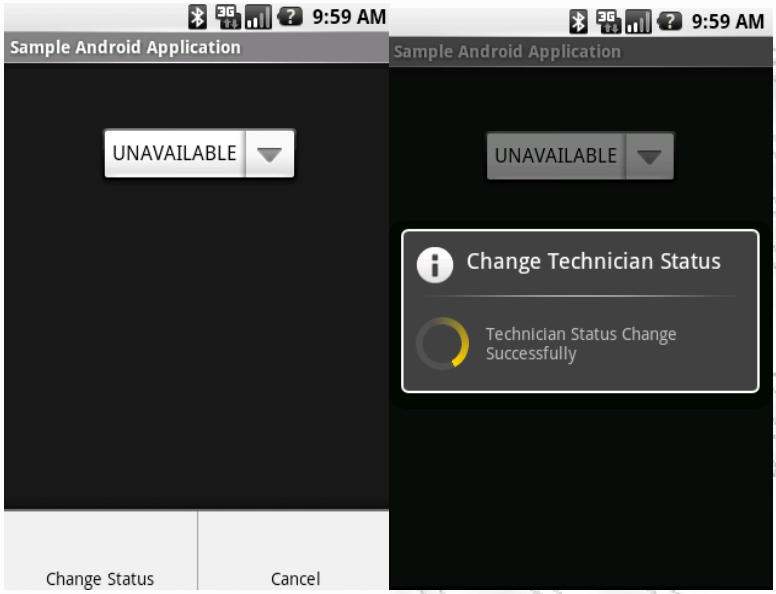


Σχημα 24 : Call technician

Το δεύτερο κουμπί δίνει στον χρήστη την δυνατότητα να αλλάξει το "status" του (δραστηριότητα "Change Technician Status").

2.9.11.Change Technician Status Activity

Η δραστηριότητα "Change Technician Status" δίνει στον χρήστη την δυνατότητα να αλλάξει το "status" του από "Available" σε "Unavailable" και αντίστροφα από "Unavailable" σε "Available". Τα δεδομένα αυτά στέλνονται στο κεντρικό υπολογιστή και αλλάζει συνολικά το "status" του συγκεκριμένου τεχνικού σε όλους τους "Mobile Workers", όπως φαίνεται και στις παρακάτω οθόνες.



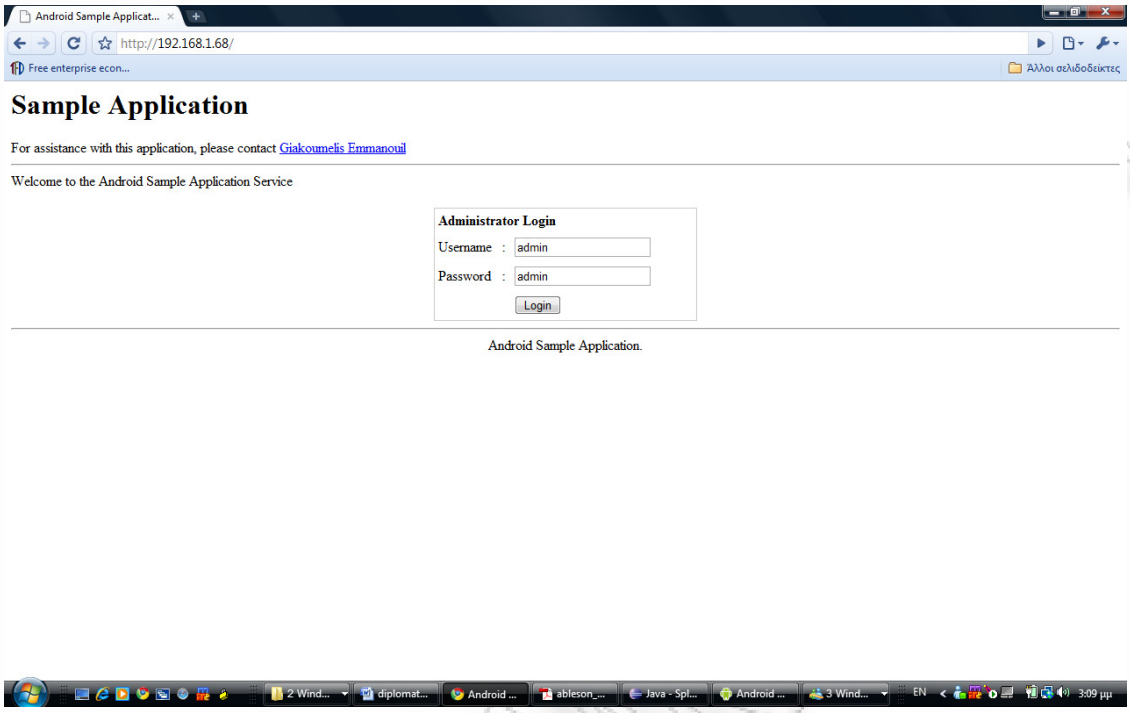
Σχημα 25 : Change technician status activity

2.10.Server side

Μια κινητή εφαρμογή στηρίζεται συχνά σε δεδομένα προερχόμενα από ένα κεντρικό υπολογιστή (server). Η εφαρμογή που υλοποιήσαμε δεν αποτελεί μία εξαίρεση. Στην συνέχεια θα παρουσιάσουμε εν συντομία τον κώδικα που τρέχει από την πλευρά του κεντρικού υπολογιστή.

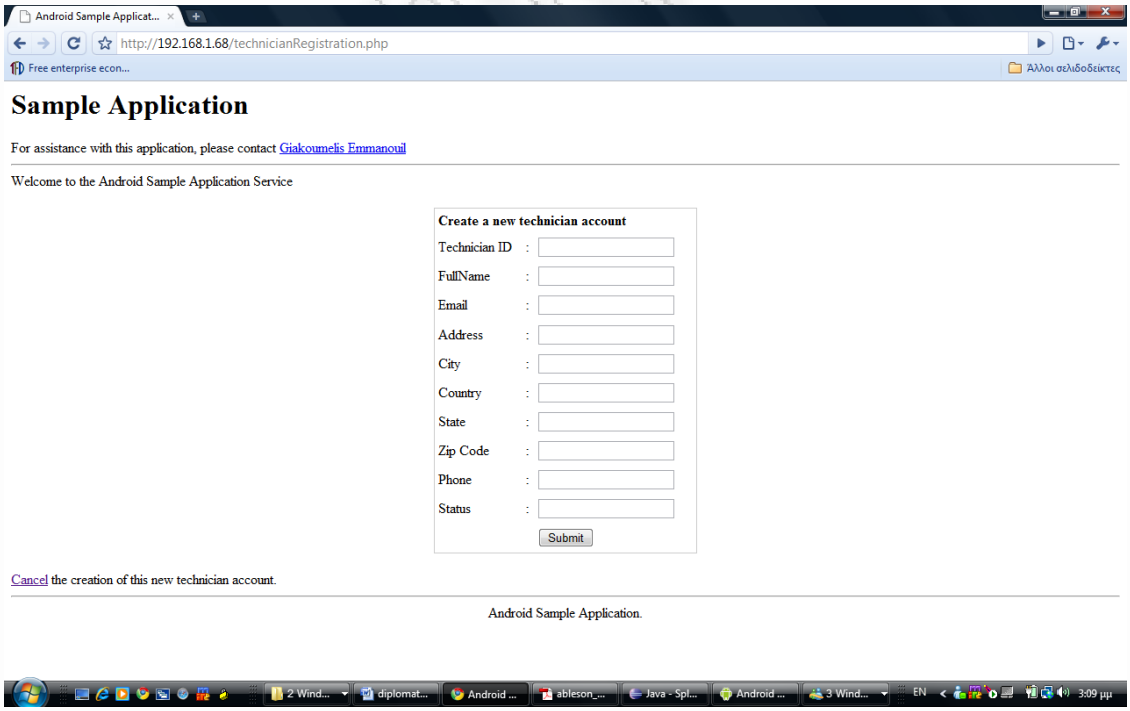
2.10.1.Διεπαφή χρηστών

Πριν προχωρήσουμε σε μεγαλύτερες λεπτομέρειες σχετικά με τον κώδικα που τρέχει από την πλευρά του κεντρικού υπολογιστή, είναι σημαντικό να κατανοήσουμε πως η εφαρμογή έχει οργανωθεί. Ο διαχειριστής εισάγει τα δεδομένα σχετικά με τους τεχνικούς και καταχωρεί εργασίες σε κάθε τεχνικό ξεχωριστά. Ο διαχειριστής για να έχει πρόσβαση στην εφαρμογή πρέπει να γνωρίζει το username/password:



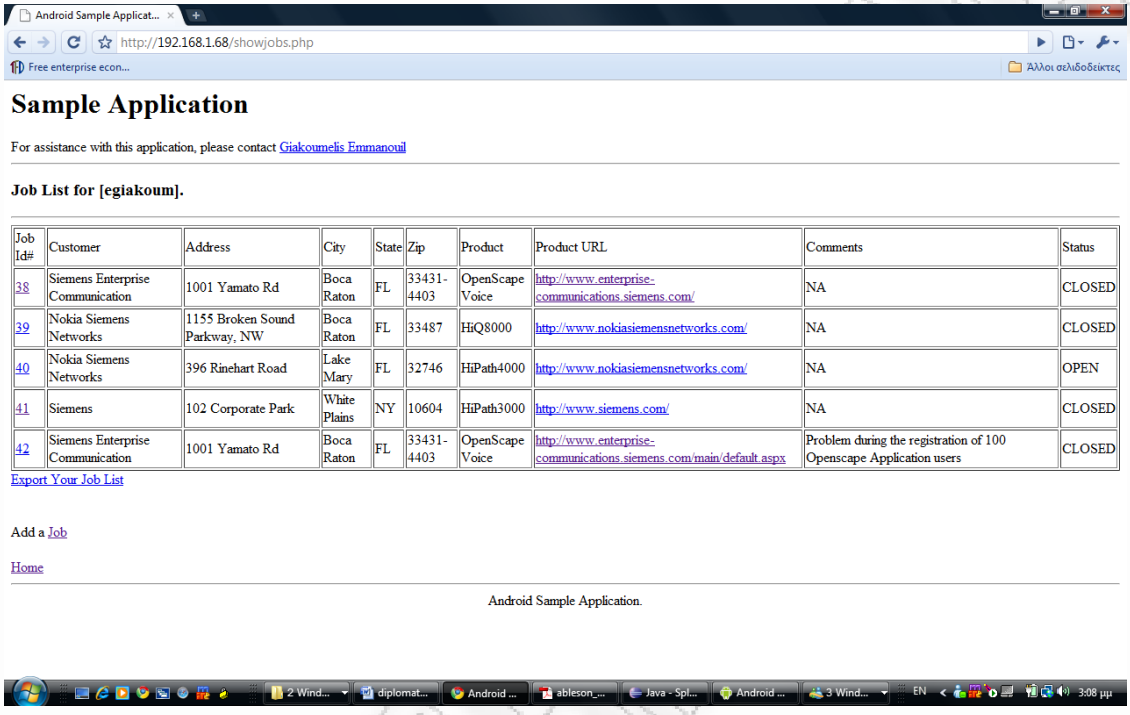
Σχημα 26 : Login to server

Ο διαχειριστής μπορεί να δημιουργήσει ένα νέο τεχνικό εισάγοντας τα στοιχεία του και παραχωρώντας του ένα μοναδικό Technician ID:



Σχημα 27 : Δημιουργία ενός Technician

Στην συνέχεια, ο διαχειριστής μπορεί να χρησιμοποιήσει αυτό το Technician ID για να διαχειριστεί τα "Jobs" κάθε τεχνικού.

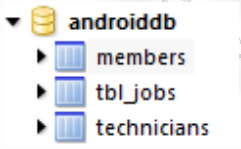


Σχημα 28 : Διαχείριση Jobs

2.10.2.Βάση δεδομένων

Όπως έχουμε αναφέρει ήδη, η βάση δεδομένων που έχουμε χρησιμοποιήσει είναι Mysql. Οι πίνακες που χρησιμοποιήθηκαν για την εφαρμογή μας είναι τρεις:

- Table members: Χρήστες του κεντρικού υπολογιστή
- Table technicians: Χρήστες των "Mobile Worker" ("Technicians")
- Table tbl_jobs: Πληροφορίες σχετικά με τις εργασίες ("Jobs")



Σχημα 29 : Βάση δεδομένων εφαρμογής

Το SQL για την δημιουργία αυτών των πινάκων φαίνεται παρακάτω:

```
DROP TABLE IF EXISTS `androiddb`.`members`;
CREATE TABLE `androiddb`.`members` (
  `id` int(10) NOT NULL AUTO_INCREMENT,
  `username` varchar(60) DEFAULT NULL,
  `password` varchar(60) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=3 DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `androiddb`.`technicians`;
CREATE TABLE  `androiddb`.`technicians` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `technicianID` varchar(32) NOT NULL,
  `fullname` varchar(32) NOT NULL,
  `email` varchar(32) NOT NULL,
  `address` varchar(32) NOT NULL,
  `city` varchar(32) NOT NULL,
  `country` varchar(32) NOT NULL,
  `state` varchar(32) NOT NULL,
  `zip` varchar(32) NOT NULL,
  `phone` varchar(32) NOT NULL,
  `STATUS` varchar(60) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `technicianID` (`technicianID`)
) ENGINE=MyISAM AUTO_INCREMENT=13 DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `androiddb`.`tbl_jobs`;
CREATE TABLE  `androiddb`.`tbl_jobs` (
  `jobid` int(11) NOT NULL AUTO_INCREMENT,
  `status` varchar(10) NOT NULL DEFAULT 'OPEN',
  `identifier` varchar(50) NOT NULL,
  `address` varchar(50) NOT NULL,
  `city` varchar(30) NOT NULL,
  `state` varchar(2) NOT NULL,
```

```
  `zip` varchar(10) NOT NULL,
  `customer` varchar(50) NOT NULL,
  `product` varchar(50) NOT NULL,
  `producturl` varchar(100) NOT NULL,
  `comments` varchar(100) NOT NULL,
 UNIQUE KEY `jobid` (`jobid`)
) ENGINE=MyISAM AUTO_INCREMENT=43 DEFAULT CHARSET=latin1;
```

2.10.3.PHP Dispatcher Code

Ο κώδικας του κεντρικού υπολογιστή έχει υλοποιηθεί σε PHP και αποτελείται από τα παρακάτω αρχεία τα οποία συνεργάζονται για να λειτουργήσει η τελική εφαρμογή.

Name	Date modified	Type	Size	Tags
sigs	18/10/2009 2:02 μμ	File Folder		
accountError.php	25/4/2009 5:27 μμ	PHP File	1 KB	
accountOk.php	25/4/2009 5:26 μμ	PHP File	1 KB	
addjob.php	5/8/2008 9:41 μμ	PHP File	1 KB	
changeTechnicianStatus.php	19/7/2009 9:18 πμ	PHP File	1 KB	
checklogin.php	25/4/2009 2:15 μμ	PHP File	2 KB	
closejob.php	19/7/2009 12:29 μμ	PHP File	1 KB	
db.php	23/4/2009 4:59 μμ	PHP File	1 KB	
emailExists.php	25/4/2009 5:24 μμ	PHP File	1 KB	
export.php	5/8/2008 9:41 μμ	PHP File	1 KB	
footer.php	24/4/2009 1:53 μμ	PHP File	1 KB	
getjoblist.php	5/8/2008 9:41 μμ	PHP File	1 KB	
getTechnicianList.php	12/7/2009 5:52 μμ	PHP File	1 KB	
header.php	24/4/2009 2:01 μμ	PHP File	1 KB	
index.php	25/4/2009 1:09 μμ	PHP File	1 KB	
login_fail.php	25/4/2009 1:06 μμ	PHP File	1 KB	
login_success.php	25/4/2009 12:54 μμ	PHP File	1 KB	
logout.php	25/4/2009 2:25 μμ	PHP File	1 KB	
main_page.php	25/4/2009 6:01 μμ	PHP File	3 KB	
manage.php	5/8/2008 9:41 μμ	PHP File	1 KB	
php.php	25/4/2009 12:13 μμ	PHP File	0 KB	
posttransaction.php	5/8/2008 9:41 μμ	PHP File	1 KB	
Register.php	12/7/2009 10:08 μμ	PHP File	2 KB	
savejob.php	5/8/2008 9:41 μμ	PHP File	1 KB	
showjob.php	5/8/2008 9:41 μμ	PHP File	1 KB	
showjobs.php	25/4/2009 5:55 μμ	PHP File	1 KB	
technicianExists.php	25/4/2009 6:21 μμ	PHP File	1 KB	
technicianRegistration.php	12/7/2009 10:07 μμ	PHP File	2 KB	
testclose.html	5/8/2008 9:41 μμ	HTML Document	1 KB	
updatejob.php	5/8/2008 9:41 μμ	PHP File	1 KB	
utils.php	19/7/2009 9:10 πμ	PHP File	8 KB	

Σχημα 30 : Κώδικας του server

2.10.4.PHP Mobile Integration Code

Όταν το Android τρέχει την δραστηριότητα Refresh Jobs, η πλευρά του κεντρικού υπολογιστή δημιουργεί "XML stream". Χωρίς να μπούμε σε υπερβολικές λεπτομέρειες σχετικές με τον κώδικα που τρέχει στον κεντρικό υπολογιστή, παρακάτω θα προσπαθήσουμε να εξηγήσουμε πώς λειτουργεί το αρχείο getjoblist.php.

```
<?
require('db.php'); #1
require('utils.php'); #2
$theid = $_GET['identifier']; #3
print (getJobsXML($theid)); #4
?>
```

Το αρχείο db.php περιέχει τον σύνδεσμο με την βάση δεδομένων. Το αρχείο utils.php περιέχει όλες τις απαραίτητες ρουτίνες για την αλληλεπίδραση με την βάση δεδομένων. Για παράδειγμα η μέθοδος getJobsXML() έχει υλοποιηθεί στο αρχείο utils.php. Επιπλέον, η μεταβλητή $theid λαμβάνεται από το "QUERY string" ("GET request"). Τέλος, η μέθοδος getJobsXML() ανακτά δεδομένα από την βάση δεδομένων και σχηματίζει XML απεικονίσεις για κάθε σειρά δεδομένων ("data row"). Αυτές οι XML απεικονίσεις πρέπει να είναι τέτοιες ώστε να μπορεί να τις κατανοήσει η εφαρμογή Android, οι οποίες θα διαβαστούν από τον "SAX parser" της κλάσης JobListHandler.

Η άλλη συναλλαγή που είναι σημαντική για την εφαρμογή μας είναι το αρχείο closejob.php, το οποίο θα εξετάσουμε παρακάτω.

```
<?
require('db.php');
require('utils.php');
$data = file_get_contents('php://input');
$jobid = $_GET['jobid'];
$f = fopen("C:\Apache\htdocs\sigs/".$jobid.".jpg","w");
fwrite($f,$data);
fclose($f);
print(closeJob($_GET['jobid']));
?>
```

Τα POST-ed δεδομένα εικόνας διαβάζονται μέσω της συνάρτησης file_get_contents(). Το μυστικό βρίσκεται στην προσθήκη του 'php://input', το οποίο μας επιτρέπει να κάνουμε δυαδικό διάβασμα. Τα δεδομένα αυτά διαβάζονται σε μία μεταβλητή που λέγεται $data. Το $jobid λαμβάνεται από το "QUERY string". Επιπλέον, τα αρχεία με τις εικόνες φυλάσσονται σε έναν κατάλογο που περιέχει τις υπογραφές σαν JPEG αρχεία (όνομα αρχείου =

<jobid>.jpg). Όταν μία εργασία έχει κλείσει και ζητείται από την εφαρμογή μας η υπογραφή, αυτό το JPEG αρχείο εμφανίζεται στον "Browser". Τέλος, η συνάρτηση closeJob() ενημερώνει την βάση δεδομένων ότι η εργασία πλέον είναι "CLOSED".

Το αρχείο getTechnicianList.php λειτουργεί με την ίδια λογική του getJobsList.php. Ο κώδικας του αρχείου getTechnicianList.php φαίνεται παρακάτω.

```
<?
require('db.php');
require('utils.php');
print (getTechniciansXML());
?>
```

Το αρχείο changeTechnicianStatus.php λαμβάνει το "id" και το "status" από το "QEURY string" του "GET request" και στην συνέχεια εκτελεί την συνάρτηση changeTechnicianStatus() (utils.php), η οποία ενημερώνει την βάση δεδομένων με το καινούργιο "status" του τεχνικού.

```
<?
require('db.php');
require('utils.php');
print(changeTechnicianStatus($_GET['id'],$_GET['status']));
?>
```

2.11.Οι μελλοντικοί πρωταγωνιστές των Mobile OS

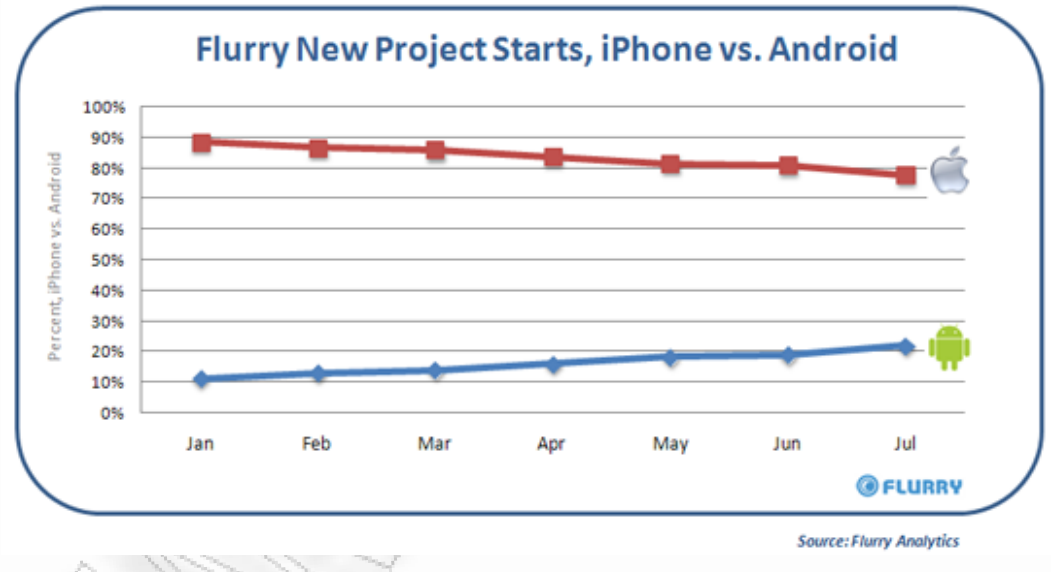
Κανείς δεν μπορεί να αμφισβητήσει την αυτοκρατορία της Apple στην παγκόσμια αγορά των smartphone. Έχει όμως ο καιρός γυρίσματα; Φαίνεται πως έχει!

Για να γίνουμε πιο ακριβείς. Η Apple πούλησε 5,2 εκκατομύρια iPhone το τελευταίο τρίμηνο, σημειώνοντας αύξηση της τάξης του 626% σε σχέση με το προηγούμενο. Έχει περισσότερες από 65.000 εφαρμογές στο App Store. Η ίδια η Apple ισχυρίζεται πως κατέχει το 32% του τζίρου τις βιομηχανίας των κινητών τηλεφώνων. Ομολογουμένως, εντυπωσιακά νούμερα! Αλλά για πόσο ακόμα;

Δεν είναι κρατικό μυστικό ότι μεγάλο ποσοστό της επιτυχίας του iPhone οφείλεται στους developers. Όλοι θέλουν να έχουν, να αναπτύξουν, να πουλήσουν την εφαρμογή τους στο App Store. Ε λοιπόν αυτή η «κοινότητα»

βρήκε και αλλού ενδιαφέρον. Οι developers άρχιζουν σιγά σιγά να στρέφονται και προς την πλατφόρμα του Android.

Σύμφωνα με την Flurry (mobile analytics) παρουσιάζεται μια στροφή των προγραμματιστών στο Android. Πιο συγκεκριμένα από την αρχή της χρονιάς το Android έχει διπλασιάσει τις εφαρμογές του, εν αντιθέση με το iPhone που μήνα με τον μήνα παρουσίαζει πτωτικές τάσεις. Αρκετά νωρίς για συμπεράσματα; Ναι. Εάν δεν περάσει αρκετά ακόμα ο καιρός δεν είναι δυνατόν να εξαχθούν ασφαλή συμπεράσματα για τις τάσεις τις αγοράς και το μέλλον αυτών των δύο πλατφορμών. Μπορεί να είναι απλά μα παροδική τρέλα, μόδα, μπορεί να είναι ή ήδη μεγάλη πληρότητα των εφαρμογών του iPhone, μπορεί να είνα ιη ευκολία ανάπτυξης εφαρμογών του Android κτλ. Μόνο ο χρόνος θα δείξει. Εν αναμονή, λοιπόν, των εξελίξεων αφού προβλέπεται πως θα απασχολούν αρκετά από εδώ και πέρα και αυτές οι δύο πλατφόρμες.



Σχημα 31 : iPhone vs Android

Για όποιον έτυχε να ασχοληθεί έστω και λίγο με τον χώρο της στατιστικής, γνωρίζει πως η πρόβλεψη αποτελεί μεγάλο βραχνά. Σε ένα κόσμο που αλλάζει μέρα με την μέρα, η πρόβλεψη για τεχνολογικά προϊόντα φαντάζει γόρδιος δεσμός. Δεν συζητάμε καθόλου το ενδεχόμενο της μεγάλης τελικής απόκλισης από αυτά που έχουν προβλεφθεί. Τους κινδύνους αυτούς τους

αγνόησε ο Ken Dulaney της Gartner που θέλησε να προβλέψει το ποσοστό επί της αγοράς που θα έχει το κάθε mobile OS το 2012. Η πρόβλεψη αυτή κάποιους θα ικανοποιήσει και κάποιους θα τους δυσαρεστήσει ανάλογα με ποίο OS υποστηρίζουν.

Κυρίαρχος των OS του 2012 θα παραμείνει το Symbian που φαίνεται να μην αντικαθίσταται ποτέ. Η έκπληξη είναι το γεγονός ότι το Android OS να είναι δεύτερο σε εισχώρηση OS στην αγορά. Με 76 εκατομμύρια συσκευές και 14,5% είναι πάνω από το iPhone που η πρόβλεψη το θέλει στο 13,7%. Ακολουθούν τα υπόλοιπα OS από κοντά. Κάτι που κάνει την πρόβλεψη αρκετά δύσκολη λόγο των μικρών διαστημάτων ασφαλείας. Πιο συγκεκριμένα:

- Symbian – 39% (203 εκατομμύρια συσκευές),
- Android – 14.5% (76 εκατομμύρια συσκευές),
- iPhone – 13.7% (71.5 εκατομμύρια συσκευές),
- Windows Mobile – 12.8% (εκατομμύρια συσκευές),
- BlackBerry OS – 12.5% (εκατομμύρια συσκευές),
- Various Linux devices – 5.4% (εκατομμύρια συσκευές),
- Palm webOS – 2.1% (εκατομμύρια συσκευές)

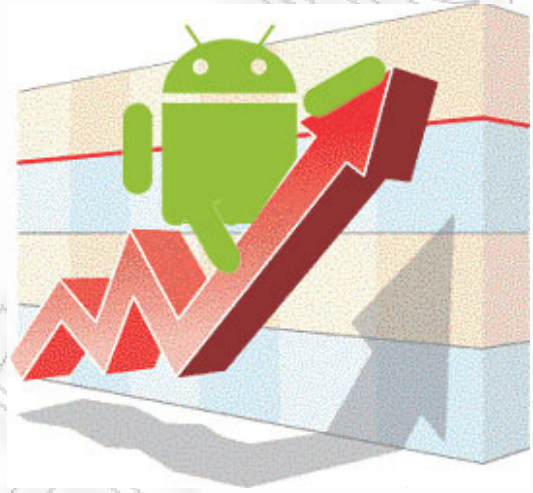
Όλα τα παραπάνω μοιάζουν πολύ ρευστά. Δεν μπορείς να προβλέψεις την αγορά και τους ανταγωνιστές. Βέβαια αυτό δεν αναιρεί την δυναμικότητα του Android. Το μόνο ασφαλές συμπέρασμα που μπορούμε να βγάλούμε από την συγκεκριμένη πρόβλεψη είναι πως το Android OS όχι μόνο θα μας απασχολήσει τα επόμενα χρόνια, αλλά θα πρωταγωνιστήσει στον χώρο.

Σε συνέχεια της παραπάνω μελάτης, ο Ken Dulaney της Gartner αναφέρει τους 5+1 λόγους που πιστεύει πως θα μας οδηγήθούμε σε αυτό το αποτέλεσμα το 2012.

Οι λόγοι για τους οποίους το Android OS θα είναι πάνω από iPhone, BlackBerry και Windows Mobile είναι:

- Το γεγονός ότι ένα μεγάλο όνομα όπως η Google είναι πίσω από το Android και το υποστηρίζει.

- Βελτιώνεται πολύ γρήγορα. Έχουμε συχνά updates και εάν συνυπολογίσουμε το xda ακόμα περισσότερα.
- Πρόκειται για ανοικτό λογισμικό, κάτι που βοηθά στην εύκολη-γρήγορη αποδοχή από τους προγραμματιστές. Λόγο της δωρεάν φύσης του υποστηρίζεται από πολλούς κατασκευαστές.
- Συνδυάζει ότι καλύτερο υπάρχει στην αγορά. Μενού στυλ iPhone, εικονίδια από Windows Mobile και Multi-tasking από BlackBerry.
- Ακόμα και οι ίδιες οι εταιρίες έχουν μπει σε παιχνίδι ανταγωνισμού για το ποια θα προσφέρει το καλύτερο extension για το Android OS στις συσκευές της. Έχουμε ήδη δει το Sense Ui της HTC, το MotoBlur της Motorola και το TouchWiz της Samsung.



Σχημα 32 : Το μέλλον του Android

3.ΒΙΒΛΙΟΓΡΑΦΙΚΕΣ ΑΝΑΦΟΡΕΣ

1. Android site, "Android SDK Reference", [September 2009], Available at HTTP: http://www.android.com/
2. Jesse Burns, "Developing Secure Mobile Applications for Android", [October 2008]
3. W. Frank Ableson, Charlie Collins, Robi Sen, "Unlocking Android", [April 2009]
4. Reto Meier, "Professional Android Application Development", [November 2008]
5. Jerome Dimarzio, "Android: A programmer's guide", [July 2008]