ΠΕΡΙΛΗΨΗ

Τα κινητά τηλέφωνα τύπου smartphone (έξυπνα τηλέφωνα) έχουν γίνει πλέον η προέκταση του ανθρώπινου χεριού. Αντιπροσωπεύουν την κοινωνική θέση, το είδος της εργασίας αλλά και την κοινωνική συμπεριφορά του καθενός. Υπάρχουν παντού, ενώ καθημερινά ολοένα και περισσότερα μοντέλα όλων των εταιρειών κατακλύζουν την αγορά. Όμως τα λειτουργικά συστήματα δύο εταιρειών επικρατούν στις περισσότερες συσκευές.

Ο λόγος για το Android OS της Google και το iOS της Apple. Στη διατριβή αυτή γίνεται αναφορά στα δύο λειτουργικά συστήματα που αναφέρθηκαν πιο πάνω και κυρίως στην πορεία ανάπτυξής τους, τα τεχνικά χαρακτηριστικά, την ασφάλεια, την αγορά αλλά και θέματα εκτέλεσης εφαρμογών και λειτουργιών συστήματος.

Οι πηγές που χρησιμοποιήθηκαν αφορούν κυρίως επιστημονικά άρθρα αλλά και τις επίσημες ιστοσελίδες των λειτουργικών συστημάτων. Μιας και δεν υπάρχει κάτι αντίστοιχο στη βιβλιογραφία για τη σύγκριση των δύο αυτών λειτουργικών συστημάτων ακολουθήθηκε μια μεθοδολογία βασισμένη στα κυριότερα θέματα τα οποία δίνουν και την ακριβή τεχνική σύγκριση. Στόχος, η εργασία αυτή να βοηθήσει οποιονδήποτε ενδιαφέρεται να σχηματίσει μια καλύτερη εικόνα για τα δύο λειτουργικά συστήματα έτσι ώστε να βοηθηθεί στην ανάπτυξη εφαρμογών ή λογισμικού για αυτά ή ακόμα και για να αποφασίσει μεταξύ των δύο για αγορά.

Στην τελευταία ενότητα παρουσιάζονται γραφήματα σχετικά με την απήχηση στην αγορά των δύο ενώ παραθέτονται και τα τελικά συμπεράσματα στηριζόμενα σε αντικειμενικά κριτήρια που προκύπτουν από την έρευνα που έχει γίνει.

ΤΕΧΝΙΚΗ ΣΥΓΚΡΙΣΗ ΤΩΝ ΛΕΙΤΟΥΡΓΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ iOS ΤΗΣ APPLE ΚΑΙ ANDROID OS ΤΗΣ GOOGLE

Έλια Κουζάρη

Η Διατριβή αυτή

Υποβλήθηκε προς Μερική Εκπλήρωση των

Απαιτήσεων για την Απόκτηση

Τίτλου Σπουδών Master

στην Επιστήμη της Πληροφορικής

στο

Πανεπιστήμιο Κύπρου

Συστήνεται προς Αποδοχή

από το Τμήμα Πληροφορικής

Ιούνης, 2011

ΣΕΛΙΔΑ ΕΓΚΡΙΣΗΣ

Διατριβή Master

ΤΕΧΝΙΚΗ ΣΥΓΚΡΙΣΗ ΤΩΝ ΛΕΙΤΟΥΡΓΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ iOS ΤΗΣ APPLE ΚΑΙ ANDROID OS ΤΗΣ GOOGLE

Παρουσιάστηκε από

Έλια Κουζάρη

Ερευνητικός Σύμβουλος _____

Όνομα Ερευνητικού Συμβούλου

Μέλος Επιτροπής _____

Όνομα Μέλους Επιτροπής

Μέλος Επιτροπής _____

Όνομα Μέλους Επιτροπής

Πανεπιστήμιο Κύπρου

Ιούνης, 2011

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον κύριο Γιώργο Παπαδόπουλο, ερευνητικό μου σύμβουλο, για την καθοδήγησή του σχετικά με τα θέματα που έπρεπε να καλυφθούν αλλά και την ελευθερία που μου έδωσε σε όλη την πορεία της έρευνας και συγγραφής της διατριβής.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΓΡΑΦΗΜΑΤΩΝ

Κεφάλαιο 1

Εισαγωγή

1.1 Κίνητρο

Το λειτουργικό σύστημα που τρέχει στις κινητές συσκευές είναι αυτό το οποίο ελέγχει τη συσκευή και τις λειτουργίες της όπως ακριβώς κάνει και το λειτουργικό σύστημα των υπολογιστών. Πρόκειται όμως για ένα πιο απλό σύστημα που έχει να κάνει με πολυμέσα, ασύρματες μορφές επικοινωνίας και διαφορετικές μεθόδους εισόδου.

Η συνεχής ανάπτυξη της τεχνολογίας αλλά και οι νέοι τρόποι επικοινωνίας απασχολούσαν πάντα τις εταιρείες κινητής τηλεφωνίας. Μετά από αρκετές δοκιμές, αρκετές εταιρείες κατάφεραν να δώσουν στο κοινό αντιπροσωπευτικά δείγματα των νέων τεχνολογιών στηρίζοντας τις λειτουργίες τους σε γνωστά λειτουργικά συστήματα μεταξύ των οποίων: Apple iOS, Google Android OS, Blackberry OS, Windows Mobile 7 και Nokia Symbian OS. Στην εργασία αυτή επιλέχθηκε η τεχνική σύγκριση δύο εξ' αυτών. Πρόκειται για την Apple με το λειτουργικό σύστημα iOS για κινητά τηλέφωνα iPhone καθώς και iPod και iPad συσκευές και για την Google με το λειτουργικό σύστημα Android για κινητά τηλέφωνα και tablets από διάφορους κατασκευαστές. Ο λόγος για τον οποίο επιλέχθηκαν τα δύο αυτά λειτουργικά συστήματα είναι η απήχησή τους που έχουν στο κοινό, όπως θα παρουσιαστεί και στα κεφάλαια που ακολουθούν καθώς και το ενδιαφέρον που προσελκύουν σε δημιουργούς εφαρμογών με την ανάπτυξης εκατοντάδων χιλιάδων για το κάθε ένα. Επιπλέον, είναι τα δύο λειτουργικά συστήματα που ακόμα και σήμερα και παρά την παγκόσμια οικονομική

κρίση, αυξάνουν τον κύκλο εργασιών τους σε αναπτυσσόμενες οικονομίες σε όλο τον κόσμο.

Όπως θα αναφερθεί και στα κεφάλαια που ακολουθούν, τα λειτουργικά αυτά συστήματα παρουσιάζουν αρκετές διαφορές, η κυριότερη από τις οποίες είναι ο τρόπος ανάπτυξης του λογισμικού. Στη μεν Apple το λογισμικό είναι κλειστού τύπου, στη δε Google το λογισμικό είναι ανοικτού κώδικα. Αυτή η διαφορά και μόνο παρουσιάζει πολύ ενδιαφέρον γιατί αποτελεί την κεντρική ιδέα στην οποία στηρίζεται η όλη φιλοσοφία των εταιρειών. Γίνεται αναφορά σε τεχνολογικά χαρακτηριστικά, ζητήματα ασφάλειας, υλικό, λογισμικό αλλά και στην εμπορική πτυχή του καθενός.

Στόχος είναι να γίνει μια σύγκριση σε όλες τις πτυχές των θεμάτων που θα μελετηθούν, θέτοντας σε αντιπαράθεση υλικό, λογισμικό αλλά και διαδικασίες ανάπτυξης και διάθεσης για τα δύο λειτουργικά συστήματα. Στο τέλος, θα παρουσιαστούν και κάποια ανοικτά ζητήματα τα οποία θα τεθούν όσον αφορά το κάθε ένα ξεχωριστά.

1.2 Μεθοδολογία που ακολουθείται

Το θέμα το οποίο αναλύεται σε αυτήν την ερευνητική διατριβή έχει να κάνει με την τεχνική σύγκριση και ανάλυση των δύο λειτουργικών συστημάτων της Apple και της Google. Το ένα είναι το iOS της Apple που τρέχει σε κινητά τηλέφωνα iPhone, σε iPad και iPod και το άλλο είναι το Android OS της Google που τρέχει σε κινητές συσκευές διάφορων κατασκευαστών.

Στο κεφάλαιο 2 γίνεται αναφορά στην εξέλιξη των επικοινωνιών ακολουθώντας τα βήματα προς τις γενιές συστημάτων επικοινωνίας 1G, 2G, 3G και πολύ πρόσφατα 4G. Ακολούθως, αναφέρεται ο ορισμός του Κινητού Υπολογισμού (Mobile Computing) μαζί με κάποιες γενικές πληροφορίες ενώ στο τέλος του κεφαλαίου παρουσιάζονται τα έξυπνα τηλέφωνα (smartphone) όπως τα συναντάμε στη

σημερινή καθημερινότητα με όλες τις ευκολίες που παρέχουν στο σύγχρονο άνθρωπο.

Στο κεφάλαιο 3 παρουσιάζεται η πορεία ανάπτυξης του iOS της Apple και του Android OS της Google. Για το κάθε λειτουργικό σύστημα δίνονται πληροφορίες σχετικά με τη σύλληψή του, την πορεία ανάπτυξης αλλά και τη διαδικασία διάθεσης στο κοινό όλων των εκδόσεων λογισμικού μέχρι και τον Μάρτιο του 2011.

Ακολούθως από το κεφάλαιο 4 μέχρι και το κεφάλαιο 9 γίνεται σύγκριση των δύο λειτουργικών συστημάτων σε διάφορες πτυχές τους από την αρχιτεκτονική, μέχρι το λογισμικό, τα τεχνικά χαρακτηριστικά, την ασφάλεια αλλά και τις αγορές εφαρμογών τους. Στο κεφάλαιο 4 παρουσιάζονται οι αρχιτεκτονικές των λειτουργικών συστημάτων. Πιο συγκεκριμένα, δίνονται εικόνες για την καλύτερη αντίληψη των επιπέδων που χρησιμοποιούν αλλά παρουσιάζονται και οι τεχνολογίες που υλοποιούνται στο κάθε επίπεδο ξεχωριστά.

Στο κεφάλαιο 5, γίνεται αναφορά στη διαχείριση συστήματος στο iOS. Αναπτύσσονται θέματα που αφορούν τη διαχείριση μνήμης, την πορεία ανάπτυξης των εφαρμογών αλλά και τα κύρια συστατικά τους. Παρέχονται επίσης εικόνες για την καλύτερη κατανόηση της εκτέλεσης των εφαρμογών ενώ παρουσιάζονται και κάποιοι μηχανισμοί χειρισμού διακοπής εφαρμογών, παρεμβολής εφαρμογών κτλ.

Αντίστοιχα, στο κεφάλαιο 6 παρουσιάζεται η διαχείριση συστήματος στο Android OS. Και στην ενότητα αυτή καλύπτονται θέματα διαχείρισης μνήμης, η πορεία ανάπτυξης και εκτέλεσης των εφαρμογών καθώς και τα κύρια συστατικά τους.

Στο κεφάλαιο 7, παρουσιάζονται θέματα που έχουν να κάνουν με την ασφάλεια των δύο λειτουργικών συστημάτων. Δίνονται στοιχεία σχετικά με τους μηχανισμούς που υλοποιεί το καθένα ενώ αναφέρονται και παραδείγματα χειρισμού επιθέσεων που έχουν συμβεί. Επίσης, παρατίθενται και τρόποι εξασφάλισης της ακεραιότητας των δεδομένων μέσω πολιτικών ασφαλείας.

Στο κεφάλαιο 8 συγκρίνονται τα κύρια χαρακτηριστικά του iOS με το Android OS. Οι παράμετροι της σύγκρισης αποτελούνται από τη διεπαφή χρήστη, το υλικό που

υποστηρίζεται από τις συσκευές που τρέχουν αυτά τα λογισμικά, το SDK στο οποίο γίνεται η ανάπτυξη εφαρμογών για το καθένα και τον τρόπο με τον οποίο γίνεται η ενημέρωση λογισμικού και εφαρμογών. Επίσης αναφέρονται οι μηχανισμοί επικοινωνίας με το διαδίκτυο καθώς και ο χειρισμός της ισχύς από το ίδιο το λογισμικό. Τέλος, παρατίθεται και σύγκριση σχετικά με το κόστος αγοράς τους.

Στο κεφάλαιο 9 γίνεται αναφορά στις αγορές εφαρμογών για το κάθε σύστημα. Τίθενται λοιπόν στο μικροσκόπιο οι πτυχές του App Store και του Android Market σε θέματα που έχουν να κάνουν με την ασφάλεια, την παροχή εφαρμογών αλλά και την ανάπτυξη.

Στο κεφάλαιο 10 παρουσιάζονται κάποια στατιστικά στοιχεία από το 2008 μέχρι και τον Μάρτιο του 2011 που απεικονίζουν τις τάσεις της αγοράς και του κοινού σε θέματα αγοράς συσκευών, λειτουργικού συστήματος που προτιμάται αλλά και τα μερίδια αγορών για τις εφαρμογές του καθενός. Στο τέλος του κεφαλαίου αυτού παρέχεται και ένας συγκριτικός πίνακας που συνοψίζει τα κυριότερα χαρακτηριστικά και τις τάσεις για το iOS και το Android OS.

Τέλος, στο κεφάλαιο 11 παρουσιάζονται κάποια ανοικτά ζητήματα όπως αυτά προκύπτουν από τα θέματα που εξετάστηκαν στα προηγούμενα κεφάλαια καθώς και τα εξαγόμενα συμπεράσματα.

Κεφάλαιο 2

4G, Κινητός Υπολογισμός, Έξυπνα Τηλέφωνα: Η εξέλιξη

2.1 Η εξέλιξη από το 1G στο 4G

Σε αυτή την ενότητα, στόχος είναι να γίνει μια σύντομη αναφορά στους πιο σημαντικούς τύπους δικτύων που χρησιμοποιούσαν τα κινητά τηλέφωνα και στη συνέχεια αυτά που χρησιμοποιούν τα τηλέφωνα της σημερινής εποχής. Με σκοπό την παροχή μιας γενικής εικόνας στα συστήματα, παρουσιάζονται οι ρυθμοί μεταφοράς που επιτυγχάνονται και το αν υποστηρίζουν ή όχι εφαρμογές βασισμένες σε δίκτυο για λειτουργία σε κινητά τηλέφωνα.

Η έννοια της κινητής τηλεφωνίας ξεκίνησε στις αρχές της δεκαετίας του 1970 και εξελίχθηκε σημαντικά μέχρι τις αρχές της επόμενης δεκαετίας. Τα πρώτα αυτά συστήματα κινητής τηλεφωνίας χρησιμοποιούσαν αναλογική τηλεφωνία και επειδή αποτελούσαν την πρώτη γενιά τέτοιων συστημάτων ονομάστηκαν χαρακτηριστικά 1G (First Generation). Τα αναλογικά συστήματα αυτής της γενιάς χρησιμοποιούσαν διαμοίραση συχνότητας πολλαπλής πρόσβασης όπου κάθε κανάλι ενός χρήστη χρησιμοποιούσε μια «αφοσιωμένη» σε αυτό ζώνη συχνοτήτων.

Αντιπροσωπευτικό σύστημα 1G θεωρείται το σύστημα AMPS (Advance Mobile Phone Service), που αναπτύχθηκε στην Αμερική το 1983 και χρησιμοποιούσε ένα κανάλι συχνοτήτων εύρους 30 kHz για κάθε χρήστη. Στη συνέχεια επάνω σε αυτό το σύστημα βασίστηκαν αρκετές βελτιώσεις όπως για παράδειγμα τα Narrow AMPS (NAMPS) και CDPD με αποτέλεσμα η ταχύτητα μεταφοράς των δεδομένων να ανέλθει στα 19.2 kbps [1].

Λόγω της ασυμβατότητας που υπήρχε σε αυτά τα συστήματα πρώτης γενιάς αλλά και στους χαμηλούς ρυθμούς δεδομένων που υποστήριζαν ποτέ δεν προσέλκυσαν αρκετά την προσοχή από εμπορικής άποψης. Βέβαια, για εκείνο το χρονικό διάστημα τέτοιες ταχύτητες ήταν αρκετές ώστε να γίνεται η αποστολή δεδομένων φωνής από ένα σημείο σε ένα άλλο ασύρματα. Όμως ο αριθμός των χρηστών κινητών τηλεφώνων αυξανόταν με ραγδαίους ρυθμούς με αποτέλεσμα οι συχνότητες να μην επαρκούν πλέον για αποδοτική επικοινωνία. Έτσι η τεχνολογία προχώρησε ένα βήμα παραπέρα. Υπήρχε πλέον η ανάγκη για αυξημένη χωρητικότητα στο δίκτυο και αυτό επιτεύχθηκε με τη δημιουργία των ψηφιακών πλέον συστημάτων που είχαν εφευρεθεί για την αντικατάσταση των αναλογικών αντίστοιχων. Τέτοια συστήματα ήταν το Ευρωπαϊκό παγκόσμιο σύστημα για κινητή επικοινωνία GSM (Global System for Mobile communication) και το αντίστοιχο των Η.Π.Α. CDMA (Code Division Multiple Access).Τα συστήματα αυτά αποτέλεσαν τη δεύτερη πλέον γενιά συστημάτων κινητής τηλεφωνίας και για το λόγο αυτό περιγράφονται χαρακτηριστικά ως 2G συστήματα.

Το σύστημα GSM είναι ένα σύστημα που χρησιμοποιεί διαμοίραση χρόνου όσον αφορά την πολλαπλή πρόσβαση (TDMA). Έτσι επιτυγχάνονται κανάλια συχνοτήτων εύρους 200kHz. Κάθε ζώνη συχνοτήτων αποτελείται στα συστήματα αυτά από 8 θέσεις. Πρόκειται λοιπόν για circuit switched συστήματα στα οποία δεσμεύεται αποκλειστικά ένα κανάλι για την επικοινωνία 2 χρηστών. Με τον τρόπο αυτό 8 κυκλώματα είναι δυνατόν να είναι ενεργά ταυτόχρονα επιτρέποντας έτσι την υποστήριξη 8 συνδρομητών στην ίδια ζώνη συχνοτήτων με διαμοίραση καναλιού των 200KHz.

Στα συστήματα δεύτερης γενιάς γεννήθηκε και η πρώτη εφαρμογή κινητών τηλεφώνων για μεταφορά δεδομένων και όχι μόνο φωνής. Πρόκειται για την εφαρμογή αποστολής σύντομων μηνυμάτων SMS (Short Message Service). Η χωρητικότητα ήταν βέβαια περιορισμένη (αποστολή μέχρι και 160 χαρακτήρων) αφού ένα και μόνο κανάλι μπορούσε να χρησιμοποιηθεί για τη μεταφορά των

δεδομένων κάθε φορά. Για το σκοπό αυτό αναζητήθηκαν εναλλακτικές λύσεις ή και τροποποιήσεις στα τότε υπάρχοντα συστήματα. Ανάμεσα στις βελτιώσεις που εφαρμόστηκαν με επιτυχία ήταν και η χρήση HSCSD (High-Speed Circuit Switched Data) όπου περισσότερες από μια θέσεις ήταν δυνατό να δεσμευτούν για την επικοινωνία μέσω μιας κλήσης.

Όσον αφορά τους ρυθμούς μεταφοράς δεδομένων στα συστήματα αυτά εξακολουθούσε να υπάρχει πρόβλημα μιας και αυτή περιορίζονταν από μόλις 9.6 με 14.4 Kbps. Ο λόγος για τον οποίο ο ρυθμός μεταφορά ήταν τόσο χαμηλός ήταν η circuit-switched φύση του GSM δικτύου οπότε υπήρχε η ανάγκη περεταίρω βελτίωσης.

Με την εξέλιξη της τεχνολογίας και την ανάγκη υποστήριξης επιπλέον υπηρεσιών μεταφοράς δεδομένων όπως e-mail, μηνύματα πολυμέσων MMS (Multimedia Messages) και πρόσβαση δικτύου, μια νέα γενιά συστημάτων κινητής επικοινωνίας εμφανίστηκε με την ονομασία 2.5G. Πρόκειται για το ασύρματο δίκτυο επικοινωνίας GPRS (General Packet Ratio Service) που είναι ένα σύστημα μετάδοσης πακέτων που επικαλύπτει το GSM και συνεργάζεται με εξωτερικά δίκτυα πακέτων δεδομένων όπως είναι και το ίντερνετ. Σε ένα GPRS σύστημα, σε κάθε κινητή συσκευή (τερματικό) ανατίθεται μια διεύθυνση IP. Η ανάθεση μπορεί να γίνει είτε στατικά όπως θα την ορίσει ο διαχειριστής του δικτύου, είτε δυναμικά και ανά σύνδεση. Όταν το τερματικό βρίσκεται σε λειτουργία είναι πάντοτε συνδεδεμένο με το GPRS και ο κάτοχος της συσκευής χρεώνεται ανάλογα με το ποσό των δεδομένων που μεταφέρεται και όχι με το χρονικό διάστημα στο οποίο βρίσκεται σε λειτουργία σε ρυθμούς που ανέρχονται στα 72.4Kbps.

Ακόμα και με αυτή την εξέλιξη της τεχνολογίας, η ανάγκη για μεγαλύτερη χωρητικότητα, υψηλότερους ρυθμούς δεδομένων και καλύτερη ποιότητα υπηρεσίας οδήγησαν στην εμφάνιση των συστημάτων επικοινωνίας τρίτης γενιάς, 3G τα οποία κυριαρχούν μέχρι και σήμερα. Κύριο λόγο στην επικράτηση των νέων αυτών συστημάτων έπαιξε και η ανάγκη για συμβατότητα μεταξύ των κυριότερων προτύπων

επικοινωνίας, όπως των GSM και cdmaOne. Με τη νέα αυτή γενιά οι ρυθμοί μεταφοράς των δεδομένων μεταξύ των τερματικών είναι κατά πολύ μεγαλύτεροι, αγγίζοντας μέχρι και 2Mbps. Με τον τρόπο αυτό είναι πλέον δυνατή και η μεταφορά αρχείων βίντεο μέσω δικτύου πράγμα που δεν μπορούσε να γίνει με τα προηγούμενα συστήματα.

Η νέα αυτή γενιά συστημάτων επικοινωνίας συνόδεψε και την «έκρηξη» των έξυπνων τηλεφώνων αφού με ικανοποιητικές ταχύτητες πρόσβασης στο δίκτυο είναι πια γεγονός η χρήση εφαρμογών που διαχειρίζονται δεδομένα πραγματικού χρόνου.

Και ενώ το 3G μοντέλο κυριαρχεί ακόμα στην επικοινωνία, η τέταρτη γενιά συστημάτων, η λεγόμενη 4G έχει εμφανιστεί (στο iPhone 4G της Apple) και αναμένεται να φέρει ακόμα πιο μεγάλες ταχύτητες που θα φτάνουν μέχρι και τα 1000Mbps. Τα συστήματα 4G είναι αποκλειστικά packet-switched και για το λόγο αυτό αναφέρονται σε όλα τα IP δίκτυα. Ενσύρματες αλλά και ασύρματες υπηρεσίες είναι πλέον εφικτό να υποστηριχθούν τη στιγμή που τα δίκτυα υποστηρίζουν πλέον ρυθμούς μεταφοράς δεδομένων από 20Mbps, 100Mbps και 200Mbps. Η υποστήριξη εφαρμογών βίντεο γίνεται ακόμα πιο εύκολη και συνάμα συναρπαστική για τους χρήστες αφού η ανταλλαγή μεγάλων αρχείων δεδομένων παύει να είναι πρόβλημα. Τέλος, ως συνέπεια της ύπαρξης ενός καθ' όλα IP δικτύου, η φωνή θα μεταφέρεται μέσω της ίδιας υποδομής μεταφοράς δεδομένων (packet-switched) με τη μορφή Voice over IP (VoIP).

2.2 Κινητός Υπολογισμός

Ο Κινητός Υπολογισμός (Mobile Computing) διέπεται από την αρχή της δημιουργίας πλατφορμών διαχείρισης πληροφοριών, που είναι ανεξάρτητες από χωρικούς και χρονικούς περιορισμούς [2]. Με την ανεξαρτησία που επιτυγχάνεται, οι χρήστες τέτοιων συσκευών μπορούν πλέον να έχουν πρόσβαση αλλά και να επεξεργάζονται τις πληροφορίες και τα δεδομένα τους παντού και πάντοτε. Η

κατάσταση του χρήστη (για παράδειγμα εν κινήσει ή όχι) δεν επηρεάζει τη δυνατότητα που παρέχει η κινητή, υπολογιστική του πλατφόρμα για διαχείριση των πληροφοριών και των δεδομένων. Έτσι, η επικράτηση του κινητού υπολογισμού δημιουργεί την ψευδαίσθηση ότι τα επιθυμητά δεδομένα καθώς και η υπολογιστική ισχύς που απαιτείται για την επεξεργασία τους είναι διαθέσιμη στο σημείο που βρίσκεται ο χρήστης, τη στιγμή που στην πραγματικότητα μπορεί να είναι χιλιόμετρα μακριά. Όπως προκύπτει λοιπόν, ο κινητός υπολογισμός δεν είναι απλά μια τεχνολογία. Είναι ουσιαστικά ένα σύνολο από λύσεις που επιτρέπουν τη μετακίνηση του χρήστη ενώ παράλληλα παρέχουν πρόσβαση σε δεδομένα οπουδήποτε και από οποιαδήποτε τοποθεσία.

Στις πλατφόρμες του κινητού υπολογισμού, η πληροφορία που μεταφέρεται μεταξύ των διαφόρων τμημάτων για επεξεργασία ρέει μέσω ασύρματων καναλιών. Τα τμήματα αυτά που επεξεργάζονται τα διάφορα δεδομένα είναι ανεξάρτητα από χρονικούς και χωρικούς περιορισμούς επίσης (για παράδειγμα ένας client στο μοντέλο client/server). Αυτό σημαίνει ότι μια υπολογιστική μονάδα, όπως είναι και ένας client, μπορεί ανά πάσα στιγμή να μετακινηθεί στο χώρο αλλά την ίδια στιγμή να παραμένει συνδεδεμένος με το server του.

Στη διάδοση και επικράτηση του κινητού υπολογισμού οδήγησαν οι προσωπικές συσκευές PDA (Personal Digital Assistant), οι φορητοί υπολογιστές αλλά και τα έξυπνα τηλέφωνα για τα οποία γίνεται αναφορά στην επόμενη ενότητα.

Χρησιμοποιώντας ένα PDA ή ένα κινητό τηλέφωνο, μέσω του web browser ο χρήστης μπορεί να έχει πρόσβαση σε δεδομένα που βρίσκονται στο ίντερνετ. Από τεχνικής πλευράς, η κινητή συσκευή του χρήστη που βρίσκεται συνδεδεμένη με το κυψελώδες σύστημα στέλνει μια αίτηση σε κάποιο συνδεδεμένο server [3]. Ο server αυτός δρα ως μια πύλη που μεταφράζει το σήμα από τη συσκευή του χρήστη σε γλώσσα την οποία μπορεί να κατανοήσει το web χρησιμοποιώντας πρωτόκολλα πρόσβασης και επικοινωνίας. Ένα τέτοιο πρωτόκολλο που είναι και δημοφιλές είναι το WAP (Wireless Application Protocol). Από την πλευρά του ο server, προωθεί την

αίτηση στο ίντερνετ για να πάρει τις κατάλληλες πληροφορίες. Η ιστοσελίδα στην οποία απευθύνεται ο server απαντά στην αίτηση και προωθεί την πληροφορία διαμέσου της σύνδεσης που διατηρεί με το server. Αυτή τη φορά η απάντηση μετατρέπεται σε μια γλώσσα σήμανσης που ονομάζεται WML (Wireless Markup Language) για να είναι δυνατή η εμφάνιση της πληροφορίας στη συσκευή του χρήστη. Η μεταφρασμένη αυτή πληροφορία στέλνεται στο σύστημα επικοινωνίας και από εκεί στη συσκευή του χρήστη.

Όσον αφορά τον κινητό υπολογισμό, χαρακτηρίζεται από 4 περιορισμούς. Αυτοί έχουν να κάνουν με τους κινητούς σταθμούς, τη μεταφερσιμότητα, τη φορητή συνδεσιμότητα και την περιορισμένη πηγή ενέργειας [4]. Καταρχήν, όσον αφορά τους κινητούς σταθμούς, θεωρούνται φτωχότεροι σε πόρους σε σχέση με τους στατικούς σταθμούς. Αυτό συμβαίνει γιατί για δεδομένο κόστος και επίπεδο τεχνολογίας, θέματα που αφορούν το βάρος, την ενέργεια, το μέγεθος αλλά και την εργονομία θα προκαλέσουν απώλειες σε υπολογιστικούς πόρους όπως ταχύτητα επεξεργαστή, μέγεθος μνήμης και χωρητικότητα δίσκου.

Συνεχίζοντας με τη μεταφερσιμότητα, επικεντρώνεται κανείς στη σύγκριση των πιθανοτήτων κλοπής μιας φορητής συσκευής και ενός σταθερού υπολογιστή. Είναι εύκολα αντιληπτό ότι η πιθανότητα κλοπής ενός φορητού υπολογιστή ή ενός έξυπνου τηλεφώνου είναι κατά πολύ μεγαλύτερη τη στιγμή που ένας σταθερός υπολογιστής βρίσκεται κλειδωμένος στο σπίτι ή σε ένα γραφείο. Συνήθως, τέτοιες φορητές συσκευές περιέχουν όχι μόνο μεγάλη ποσότητα πληροφορίας αλλά και σημαντικά προσωπικά δεδομένα με αποτέλεσμα τυχόν ζημιά σε αυτά ή απώλεια να προκαλεί μεγάλη ανησυχία στον κάτοχο.

Προχωρώντας στη φορητή συνδεσιμότητα, είναι ξεκάθαρο ότι αυτή είναι που θα διασφαλίσει την επιθυμητή απόδοση και αξιοπιστία για τις εργασίες του χρήστη. Πρόκειται όμως για ασύρματα δίκτυα στα οποία η συσκευή αποκτά διαδοχικά πρόσβαση ανάλογα με την τοποθεσία της. Έτσι, τη στιγμή που το ανοικτό δίκτυο μιας μεγάλης εταιρείας μπορεί να προσφέρει αξιόπιστη και γρήγορη πρόσβαση σε

δεδομένα και υπηρεσίες, θα εναλλάσσονται και μικρά δίκτυα με μικρότερες ταχύτητες που πιθανό να προκαλούν καθυστέρηση στην επεξεργασία και την πρόσβαση σε πληροφορία. Βέβαια, υπάρχει και το ενδεχόμενο ξαφνικά να μην υπάρχει ανοικτό διαθέσιμο δίκτυο με ακόμα πιο σοβαρά προβλήματα ιδίως όταν ο χρήστης εν κινήσει χειρίζεται ηλεκτρονικά μηνύματα και τραπεζικές εντολές.

Τέλος, σημαντική ανησυχία για τον κινητό υπολογισμό είναι και οι περιορισμένες πηγές ενέργειας που έχουν οι φορητές συσκευές. Η χωρητικότητα των μπαταριών συνεχώς αυξάνεται ενώ ταυτόχρονα μειώνεται το μέγεθος, αυτό όμως δεν εξαλείφει την ανάγκη για χαμηλούς ρυθμούς κατανάλωσης. Η επίλυση και αυτού του θέματος θεωρείται δύσκολη αφού απαιτείται προσοχή σε επίπεδα αποδοτικότητας τόσο σε υλικό όσο και σε λογισμικό.

Δυστυχώς, η φορητότητα επιδεινώνει την σύζευξη που υπάρχει μεταξύ της αυτονομίας και της αλληλεξάρτησης, πράγμα που αποτελεί χαρακτηριστικό όλων των κατανεμημένων συστημάτων. Υπάρχει όμως η ανάγκη η επικοινωνία να ξεπεράσει τα προβλήματα που προκαλούν τα αναξιόπιστα και χαμηλής ταχύτητας δίκτυα καθώς και να υπάρχει ευαισθησία στην κατανάλωση της ενέργειας των συστημάτων. Έτσι μια βιώσιμη προσέγγιση προς τον κινητό υπολογισμό θα πρέπει να εξασφαλίζει τη χρυσή τομή μεταξύ αυτών των αντικρουόμενων θεμάτων. Καταλήγει λοιπόν κανείς στο συμπέρασμα ότι καθώς η κατάσταση ενός client είναι μεταβλητή, οι κινητοί clients πρέπει να προσαρμόζονται. Δηλαδή, πρέπει να γίνεται δυναμικά ανακατανομή των ευθυνών του client και του server ανάλογα με τις συνθήκες που επικρατούν στο δίκτυο. Για να μπορέσει να γίνει αυτό, ο ίδιος ο client πρέπει να διαισθάνεται τις αλλαγές στο περιβάλλον του, να κάνει υποθέσεις σχετικά με την αιτία που προκάλεσε αυτές τις αλλαγές αλλά και να αντιδρά σε αυτές κατάλληλα. Με άλλα λόγια, ο client πρέπει να κάνει γενικούς υπολογισμούς βασισμένος σε τοπικές παρατηρήσεις.

Ολοκληρώνοντας το κομμάτι αυτό προκύπτει λοιπόν το συμπέρασμα ότι ο κινητός υπολογισμός είναι μεν πλέον στην καθημερινότητα των ανθρώπων, υπολείπονται όμως ακόμα τρόποι για την αντιμετώπιση των σημαντικών

προβλημάτων που προκύπτουν κυρίως λόγω της φορητότητας των συσκευών. Πλέον, είναι αντιληπτό ότι το κλειδί για τη φορητότητα και τον κινητό υπολογισμό είναι η συνεχής αναπροσαρμογή του client στις συνθήκες που προκύπτουν.

2.3 Έξυπνα Τηλέφωνα

Δεν αποτελούν ακόμα μακρινό παρελθόν οι εικόνες μεγάλων, βαριών και άκομψων κινητών τηλεφώνων. Κι όμως, τα κινητά τηλέφωνα μέσα σε λίγα μόλις χρόνια έχουν εκτοξεύσει τις δυνατότητές τους συνδυάζοντας πλέον ενσωματωμένες, ψηφιακές φωτογραφικές μηχανές, web browsers και mp3 players. Και όλα αυτά σε συσκευές αρκετά μικρές που να χωράνε στην παλάμη του χεριού ενός ενήλικα.

Η ιδέα ήταν και παραμένει απλή ακόμα και σήμερα: Συνδυασμός ενός κινητού τηλεφώνου με μια συσκευή οργάνωσης PDA. Οι δυνατότητες που προκύπτουν, πολλές: πραγματοποίηση κλήσεων, έλεγχος email, οργάνωση ημερολογίου και πολλά άλλα. Τα «έξυπνα τηλέφωνα» (smartphones) δίνουν την εντύπωση της πρόσβασης στο γραφείο ενός ατόμου από παντού. Επιπλέον, με τις φωτογραφικές μηχανές, τη μουσική αλλά και το ίντερνετ ο χρόνος περνά πάντα γρήγορα κι ευχάριστα μετατρέποντας τη συσκευή του τηλεφώνου σε ένα ζωντανό και φορητό κέντρο ψυχαγωγίας.

Μια «έξυπνη» συσκευή περιλαμβάνει κυρίως ένα προηγμένο και περίπλοκο λειτουργικό σύστημα το οποίο επιτρέπει τη λειτουργία third-party εφαρμογών στη συσκευή. Τέτοιο ανάλογο λειτουργικό σύστημα κατέχουν και οι συσκευές PDA οι οποίες όμως στερούν από το χρήστη τη δυνατότητα διαχείρισης της συσκευής του και ως κινητό τηλέφωνο. Όπως έχει ήδη αναφερθεί, ένα έξυπνο τηλέφωνο ενθυλακώνει και εμπορικές ηλεκτρονικές διευκολύνσεις που γίνονται προσβάσιμες μέσω ενσωματωμένων ψηφιακών μηχανών, mp3 players και πιο πρόσφατα δέκτες GPS. Συγκεκριμένα, με τον GPS δέκτη, ένα τηλέφωνο αυτής της κατηγορίας δίνει την ικανότητα στον κάτοχό του να καθορίσει με ακρίβεια τη θέση του μέσω των σημάτων που αποστέλλονται από τους δορυφόρους. Τα σήματα αυτά λειτουργούν και ως

είσοδος σε αρκετές εφαρμογές τους για το σχεδιασμό διαδρομών μετακίνησης αλλά και την εύρεση σημείων στο χάρτη. Η αλληλεπίδραση αυτών των υπηρεσιών γίνεται μέσω ενός API πρωτοκόλλου στις third-party εφαρμογές που χρησιμοποιούνται.

Σήμερα και ενώ η τεχνολογία της κινητής τηλεφωνίας έχει προχωρήσει κατά πολύ, αρκετές είναι οι εταιρείες που ανταγωνίζονται στην αγορά. Οι Microsoft, NOKIA, Sony Ericsson, Samsung αλλά και πιο πρόσφατα η Apple ξοδεύουν εκατομμύρια δολάρια το χρόνο στην προσπάθεια να καινοτομήσουν και να ξεπεράσουν τα εμπόδια που προκύπτουν από τον ισχυρό ανταγωνισμό. Στο πολύ πρόσφατο παρελθόν ανήκει και η προσπάθεια της Google η οποία ξεκίνησε να προμηθεύει την αγορά με ένα νέο λειτουργικό σύστημα για κινητά τηλέφωνα με την ονομασία Android Smartphone OS.

Παρόλο που δεν θα γίνει αναφορά στα διάφορα μοντέλα των έξυπνων τηλεφώνων που κυκλοφορούν στην αγορά, είναι σημαντικό να σημειωθεί ότι η τεχνολογία που χρησιμοποιεί το καθένα ξεχωριστά δίνει τη δυνατότητα ύπαρξης χαρακτηριστικών που τα διακρίνουν σε κατηγορίες. Ένα τέτοιο χαρακτηριστικό είναι και ο τρόπος εισαγωγής πληροφορίας μέσα στο σύστημα που είναι δυνατό να γίνει με 3 κυρίως τρόπους. Ο πρώτος αφορά την ύπαρξη πλήρους πληκτρολογίου Querty, ο δεύτερος την είσοδο πληροφορίας αποκλειστικά μέσω οθόνης αφής και ο τρίτος την ύπαρξη λίγων μόνο πλήκτρων με πολλαπλή πίεση. Βέβαια, είναι δυνατόν και ο συνδυασμός όλων σε μια μόνο συσκευή.

Ο Richard Webb, διευθυντής του τμήματος ανάλυσης με την ονομασία WiMax, Microwave and Mobile Devices στο Λονδίνο, είχε πει το 2008: «Τα έξυπνα τηλέφωνα αναμένεται να έχουν ραγδαία αύξηση πωλήσεων το 2009 και θα παραμείνουν ως ο μόνος τομέας της κινητής τηλεφωνίας που θα διατηρήσει τον κύκλο εργασιών του για τα επόμενα 5 χρόνια και ο μόνος τομέας που θα αποφέρει ετήσια κέρδη σε διψήφιο ποσοστό από το 2011 μέχρι και το 2013» [5].

Στην εργασία αυτή θα γίνει εκτενής αναφορά στους δύο τύπους λογισμικού για έξυπνα τηλέφωνα που επικράτησαν στην αγορά τα τελευταία 2 χρόνια. Πιο

συγκεκριμένα, θα μελετηθεί προσεκτικά το λογισμικό της Apple που βρίσκεται στα τηλέφωνα iPhone της εταιρείας αλλά και το λειτουργικό σύστημα Android της Google το οποίο χρησιμοποιείται σε συσκευές αρκετών εταιρειών σήμερα. Βαρύτητα θα δοθεί στα τεχνολογικά χαρακτηριστικά και στον τρόπο ανάπτυξης του καθενός ώστε να εντοπιστούν ομοιότητες αλλά και διαφορές που να δικαιολογούν την τάση της αγοράς. Σε μικρότερο βαθμό θα γίνει απλή αναφορά στις οικονομικές πτυχές του θέματος που αφορούν κυρίως τον ανταγωνισμό τους όχι μόνο σε τεχνολογικό επίπεδο αλλά και σε οικονομικό.

Κεφάλαιο 3

Πορεία ανάπτυξης λογισμικού Apple's iOS for iPhone και Google's Android

3.1 Η πορεία ανάπτυξης του λογισμικού της Apple για τα τηλέφωνα iPhone

Όλα άρχισαν στις 9 Ιανουαρίου του 2007 όταν ο Steve Jobs (Apple CEO) ανέβηκε στο βήμα της Macworld Expo στο Σαν Φρανσίσκο για να προαναγγείλει την άφιξη ενός νέου κινητού τηλεφώνου, του Apple iPhone. Ο ενθουσιασμός που τον διακατείχε στην αναγγελία του αυτή εντυπωσίασε τους επίδοξους αγοραστές ανά το παγκόσμιο. «Μαζί θα γράψουμε ιστορία. Σήμερα η Apple θα ξανά-εφεύρει το κινητό τηλέφωνο!» είχε πει τότε. Και παρόλο που το προϊόν δεν είχε ακόμα κυκλοφορήσει στην αγορά, έγινε ένα από τα πιο πολυσυζητημένα τεχνολογικά προϊόντα, ξεπερνώντας κατά πολύ οποιαδήποτε αναφορά είχε γίνει μέχρις στιγμής για προϊόντα της ίδιας κατηγορίας. Για το λόγο αυτό, το iPhone ήταν μέχρι πρόσφατα το πιο εμπορικό «έξυπνο τηλέφωνο» από τη στιγμή που κυκλοφόρησε.

Το λειτουργικό σύστημα του iPhone αναπτύχθηκε και έγινε διαθέσιμο από την εταιρεία Apple. Είναι το ίδιο λειτουργικό σύστημα που χρησιμοποιεί το iPod Touch και το iPad και προήλθε από το λειτουργικό σύστημα Mac OX X [6]. Το συγκεκριμένο λειτουργικό σύστημα αξίζει να σημειωθεί ότι δεν είχε καμιά ονομασία μέχρι τις 6 Μαρτίου 2008 όπου η πρώτη έκδοση του iPhone SDK κυκλοφόρησε. Η πρώτη έκδοση του λειτουργικού αυτού συστήματος κυκλοφόρησε στις 29 Ιουνίου του 2007 και την ακολούθησαν σε σύντομο χρονικό διάστημα οι ακόλουθες: Η έκδοση 1.0.2 που αρχικά ήταν διαθέσιμη για το iPod touch και η έκδοση 1.1.1 με ανανεωμένη διεπαφή χρήστη στις εφαρμογές υπολογιστικής μηχανής. Η έκδοση αυτή υποστήριζε

και TV out καθώς και προσαρμογή ήχου στο μικρόφωνο και τα ηχεία. Στη συνέχεια κυκλοφόρησε η έκδοση 1.1.2 με ειδικό σήμα προειδοποίησης για το επίπεδο φόρτισης της μπαταρίας και υποστήριξη διεθνών γλωσσών στο λεξικό του. Ακολούθησε η έκδοση 1.1.3 η οποία παρείχε επιπλέον χαρακτηριστικά που δεν υπήρχαν σε προηγούμενες εκδόσεις όπως το ηλεκτρονικό ταχυδρομείο, οι χάρτες, οι μετοχές, ο καιρός και οι σημειώσεις. Στην έκδοση αυτή αξίζει να σημειωθεί ότι αυξήθηκε και η χωρητικότητα των SMS μηνυμάτων από 1000 σε 75000 ενώ ενσωμάτωσε και την εφαρμογή “Locate me” η οποία ήταν σε θέση να δείξει την τοποθεσία του τηλεφώνου. Οι εκδόσεις 1.1.4 και 1.1.5 διαδέχθηκαν την προηγούμενη και επέλυαν κυρίως προβλήματα λογισμικού (bugs) που είχαν εκδηλωθεί μέχρι στιγμής με αποτέλεσμα τη βελτίωση της ταχύτητας του UI.

Σταθμός στην πορεία ανάπτυξης του λογισμικού του iPhone στάθηκε η έκδοση 2.0 η οποία ήταν διαθέσιμη με το νέο μοντέλο, iPhone 3G, στις 11 Ιουλίου 2008. Η κύρια τροποποίηση του είχε γίνει για την υποστήριξη ικανοτήτων 3G δικτύου. Βελτίωσε το λειτουργικό σύστημα εισάγοντας νέα χαρακτηριστικά όπως τη δυνατότητα έναρξης του WiFi στο προφίλ πτήσης του τηλεφώνου. Υποστήριζε SVG και CiscoIPsec VPN. Έγιναν επίσης κι άλλες διεθνείς γλώσσες διαθέσιμες στο πληκτρολόγιο του τηλεφώνου. Τη έκδοση 2.0 διαδέχθηκε η 2.1 η οποία παράλληλα με τη διόρθωση κάποιων προβλημάτων βελτίωσε και την απόδοση του συστήματος. Περιλάμβανε νέα χαρακτηριστικά λειτουργικού συστήματος όπως: αλλαγμένο 3G, ενδείξεις σχετικά με EDGE και GPRS καθώς και κλείσιμο της κάμερας μέσω της σελίδας των περιορισμών. Παρείχε επίσης νέες εκδόσεις για λίστες μουσικής του iPod. Η έκδοση 2.2 βελτίωσε την ταχύτητα του ηλεκτρονικού ταχυδρομείου και των χαρτών και βελτίωσε τη σταθερότητα και την ποιότητα του Safari (web browser).

Ακολούθησε η έκδοση 3.0 με το iPhone 3GS στις 17 Ιουνίου του 2009. Οι ενημερώσεις 3.1 και 3.2 υποστήριζαν και το νέο προϊόν της Apple, το iPad και εισήγαγαν νέα όψη χάρτη στην αντίστοιχη εφαρμογή. Υποστήριξε πλέον και την αποστολή MMS μηνυμάτων ενώ ταυτόχρονα βελτίωσε υπάρχουσες εφαρμογές.

Στις 24 Ιουνίου 2010 κυκλοφόρησε και η τέταρτη γενιά iPhone τηλεφώνων. Συνοδεύτηκαν με τη νέα έκδοση λειτουργικού συστήματος, το iPhone 4G το οποίο χρησιμοποιεί δύο κάμερες στο τηλέφωνο, χαρακτηριστικό που δεν υπήρχε στις προηγούμενες εκδόσεις. Παρόλα αυτά, σε αυτό το λογισμικό απουσιάζει η χρήση multi-touch οθόνης που όπως θα παρουσιαστεί στις επόμενες ενότητες είναι και ένα από τα καινοτόμα χαρακτηριστικά που εισήγαγε το iPhone στην τεχνολογία του αρχικά. Λίγους μήνες αργότερα κυκλοφόρησε ενημέρωση του λογισμικού με την έκδοση 4.1 η οποία βελτίωνε θέματα απόδοσης της συσκευής 4G. Τη διαδέχθηκε μέσα σε ελάχιστο χρόνο η έκδοση 4.2 η οποία εκτός από την εισαγωγή νέων χαρακτηριστικών, έδινε τη λύση και σε ζητήματα ασφάλειας που είχαν προκύψει με τις προηγούμενες εκδόσεις. Τη στιγμή που γράφονται αυτές οι γραμμές, η έκδοση 4.3 έχει ήδη κυκλοφορήσει (από τις 9 Μαρτίου 2011) και πρόκειται για την πρώτη έκδοση λειτουργικού της Apple που αφήνει εκτός της υποστήριξη των τηλεφώνων iPhone 3G (εκδόσεις λειτουργικού συστήματος 2.x) [7].

3.2 Η πορεία ανάπτυξης του λειτουργικού συστήματος Google Android

Η εταιρεία Android ιδρύθηκε στην Καλιφόρνια τον Οκτώβριο του 2003 με στόχο «… την ανάπτυξη πιο έξυπνων κινητών συσκευών που έχουν περισσότερη αίσθηση σχετικά με την τοποθεσία του χρήστη και τις προτιμήσεις του.» σύμφωνα με τον Andy Rubin που άνηκε στην ομάδα ίδρυσης [7]. Η εταιρεία αυτή έδρασε με μυστικότητα αφού το μόνο που αποκάλυψε ήταν ότι επικεντρώνονταν σε λογισμικό για κινητά τηλέφωνα.

Η Google αγόρασε την Android τον Αύγουστο του 2005 ενώ μέχρι τότε δεν ήταν σχεδόν τίποτα γνωστό για την εταιρεία αυτή. Πολλοί υπέθεσαν τότε ότι η Google σκόπευε να εισέλθει με αυτή της την κίνηση στην αγορά των κινητών τηλεφώνων. Στην Google η ομάδα ανάπτυξης παρουσίασε μια πλατφόρμα για κινητές συσκευές

βασισμένη στον πυρήνα του Linux (Linux Kernel) την οποία έδωσαν στην αγορά με στόχο τους κατασκευαστές τηλεφώνων και τις εταιρείες κινητής τηλεφωνίας. Στο σημείο αυτό, δίνονταν εγγυήσεις παροχής ενός ευέλικτου και αναβαθμίσιμου συστήματος ενώ ταυτόχρονα είχαν δοθεί και σειρές από συστατικά υλικού αλλά και λογισμικού. Με τον τρόπο αυτό, η Google ήθελε να αποδείξει στην αγορά ότι η πλατφόρμα που εισήγαγε ήταν ανοικτή σε διάφορους βαθμούς συνεργασίας σε σχέση με το λογισμικό και το υλικό που είχαν ήδη οι εταιρείες.

Ακολούθησε αρκετή φημολογία σχετικά με τις προθέσεις τις Google προς την αγορά των τηλεπικοινωνιών. Το Δεκέμβριο του 2006 το BBC αλλά και η εφημερίδα The Wall Street επισήμαναν ότι η Google επιθυμούσε την αναζήτησή της αλλά και τις εφαρμογές της σε κινητά τηλέφωνα και ότι δούλευαν σκληρά για να το πετύχουν αυτό. Τις φήμες αυτές ακολούθησαν και αυτές που ήθελαν την Google να αναπτύσσει ένα Google-branded κινητό τηλέφωνο ενώ παράλληλα έδινε πρωτότυπα σε κατασκευαστές τηλεφώνων και διαχειριστές δικτύων.

Στις 5 Νοεμβρίου του 2007 αποκαλύφθηκε το Open Handset Alliance που αποτελείται από εταιρείες όπως οι: HTC, Google, Intel, LG, Motorola, NVidia, Samsung, Sony Ericsson αλλά και αρκετές άλλες. Στόχος του ήταν η ανάπτυξη ανοικτών προτύπων για κινητές συσκευές. Την ίδια μέρα, ο οργανισμός αυτός ανακοίνωσε και το πρώτο του προϊόν, το Android, μια πλατφόρμα για κινητές συσκευές κτισμένη στον πυρήνα του λειτουργικού συστήματος Linux.

Το Android ήρθε για να φέρει ένα πλήρες σύνολο λογισμικού για κινητές συσκευές. Περιλαμβάνει λειτουργικό σύστημα, ενδιάμεσο λογισμικό (middleware) και εφαρμογές. Κατασκευάστηκε εξαρχής έτσι ώστε να επιτρέπει σε αυτούς που δημιουργούν εφαρμογές να κατασκευάζουν εφαρμογές που εκμεταλλεύονται πλήρως όλες τις δυνατότητες του κάθε κινητού τηλεφώνου. Είναι πλήρως ανοικτού κώδικα έτσι ώστε η κάθε εφαρμογή να μπορεί να καλέσει οποιαδήποτε λειτουργικότητα του πυρήνα στο εκάστοτε τηλέφωνο. Με τον τρόπο αυτό δίνεται η δυνατότητα ενεργοποίησης της κάμερας, η αποστολή γραπτών μηνυμάτων κτλ επιτρέποντας

στην κάθε ομάδα ανάπτυξης να δημιουργήσει εφαρμογές που θα εξυπηρετούν ακόμα και τους πιο απαιτητικούς χρήστες.

Το λογισμικό Android είναι διαθέσιμο ως λογισμικό ανοικτού κώδικα από τον Οκτώβριο του 2008. Υπό την άδεια ανοικτού κώδικα Apache [54], οι εταιρίες έχουν τη δυνατότητα να προσθέσουν τις δικές τους εφαρμογές και επεκτάσεις σε αυτό και να τις προωθήσουν στην αγορά χωρίς να χρειάζεται πρώτα να τις υποβάλουν στην κοινότητα ανοικτού κώδικα.

Η ιστορία του λειτουργικού συστήματος Android άρχισε ουσιαστικά με την έκδοση 1.0 στις 23 Σεπτεμβρίου του 2008. Όλες οι αναβαθμίσεις και ενημερώσεις έκτοτε στηρίζονται κυρίως στη βάση του λειτουργικού συστήματος, στην επιδιόρθωση κάποιων bugs αλλά και στην εισαγωγή νέων χαρακτηριστικών. Κάθε έκδοση αναπτύσσεται κάτω από ένα κωδικοποιημένο όνομα που αναφέρεται κυρίως σε κάποιο γλυκό! Έτσι, ξεκινώντας από την πρώτη έκδοση, την 1.0 άρχισαν οι αναβαθμίσεις και οι ενημερώσεις. Η έκδοση 1.1 στις 9 Φεβρουαρίου 2009 ήταν διαθέσιμη μόνο για το τηλέφωνο Ti-Mobile G1. Οι αλλαγές περιορίστηκαν σε θέματα που αφορούσαν το API καθώς και σε θέματα μετάβασης του τηλεφώνου σε κατάσταση αναμονής.

Ακολούθησε η πρώτη μεγάλη αναβάθμιση με την έκδοση 1.5 με την ονομασία “cupcake”. Η έκδοση αυτή του λογισμικού παρουσίαζε νέα χαρακτηριστικά αλλά και ενημερώσεις για τη διεπαφή χρήστη. Έδινε πλέον τη δυνατότητα λήψης και αποθήκευσης βίντεο από το κινητό τηλέφωνο, τη δυνατότητα προβολής τους στο διαδίκτυο μέσω YouTube καθώς και το ανέβασμα φωτογραφιών μέσω του κινητού τηλεφώνου αυτόματα. Υπήρξαν επίσης βελτιώσεις όσον αφορά τη χρήση και συνδεσιμότητα του Bluetooth της συσκευής. Λίγους μόνο μήνες αργότερα και συγκεκριμένα το Σεπτέμβριο του 2009 η έκδοση 1.5 αντικαταστάθηκε από την 1.6 με την ονομασία “donut”. Οι ενημερώσεις της έκδοσης αυτής αφορούσαν κυρίως την προβολή του ανανεωμένου Android Market, βελτιώσεις παρουσίασης για τα άλμπουμ

φωτογραφιών αλλά και εισαγωγή νέων τεχνολογιών για τη συνδεσιμότητα και επικοινωνία των συσκευών όπως CDMA, 802.1x κτλ.

Ένα μόνο μήνα αργότερα, στις 26 Οκτωβρίου 2009, ανακοινώθηκε η νέα έκδοση του Android με την ονομασία 2.0 – “Eclair”. Με την έκδοση αυτή οι χρήστες είχαν τη δυνατότητα να κινούνται με ακόμα μεγαλύτερη ταχύτητα ανάμεσα στις λειτουργίες του τηλεφώνου, να απολαμβάνουν πλουσιότερα γραφικά καλύτερης ποιότητας αλλά και Bluetooth 2.1. Την έκδοση αυτή ακολούθησαν οι 2.0.1 και 2.1 μέχρι τον Ιανουάριο του 2010. Μέχρι το Μάιο όμως του 2010 τα τηλέφωνα με λειτουργικό σύστημα Android είχαν την ευκαιρία να λειτουργούν με την έκδοση 2.2 του συστήματος με την ονομασία “Froyo”. Αρκετά ήταν τα ζητήματα που αντιμετωπίστηκαν με την έκδοση αυτή. Έγιναν καταρχήν βελτιώσεις σχετικά με την ταχύτητα του λειτουργικού συστήματος, τη μνήμη και την απόδοση. Ανανεώθηκε στη συνέχεια και η αγορά εφαρμογών του Android, ενώ έγινε ευκολότερη και η εναλλαγή μεταξύ γλωσσών του πληκτρολογίου και του λεξικού τους. Τέλος, ήταν πλέον δυνατή η αποστολή επαφών μέσω Bluetooth.

Στις 6 Δεκεμβρίου 2010 έγινε διαθέσιμη στο κοινό η έκδοση 2.3 του Android με την ονομασία “Gingerbread”. Μεταξύ άλλων, οι αλλαγές περιλάμβαναν ανανεωμένη γραφική διασύνδεση χρήστη, υποστήριξη τηλεφώνων με μεγαλύτερες οθόνες και καλύτερη ανάλυση, υποστήριξη για VoiP τηλεφωνία καθώς και αναπαραγωγή βίντεο μέσω δικτύου. Επίσης, εισήχθηκαν στοιχεία που αφορούν ήχο, εικόνα και μεθόδους εισαγωγής στοιχείων στο τηλέφωνο για τη διευκόλυνση των ομάδων ανάπτυξης εφαρμογών αλλά και garbage collector για βέλτιστη απόδοση. Η τελευταία έκδοση αυτού του γκρουπ είναι η έκδοση 2.3.3 την οποία ενσωματώνουν σήμερα όλα τα κινητά τηλέφωνα Android με δωρεάν αναβάθμιση μέσω της ιστοσελίδας του λειτουργικού συστήματος.

Η τελευταία διαθέσιμη έκδοση είναι η 3.0 με το όνομα “Honeycomb” που δόθηκε στους χρήστες στις 22 Φεβρουαρίου του 2011. Προς το παρόν, η έκδοση αυτή είναι διαθέσιμη μόνο για tablets που λειτουργούν με Android και συγκεκριμένα για το

Motorola Xoom tablet που κυκλοφόρησε στις αγορές στις 24 του ίδιου μήνα. Μιας και επικεντρώνεται στα tablets, η έκδοση αυτή δίνει βέλτιστη υποστήριξη tablet με ένα νέο UI αλλά και τρισδιάστατα desktop με νέα χαρακτηριστικά. Επίσης, υπάρχει υποστήριξη για επεξεργαστές πολλαπλών πυρήνων.

Μέσα στους επόμενους μήνες και πριν το τέλος του 2011 αναμένεται και η επόμενη έκδοση λογισμικού Android με την ονομασία “IceCreamSandwich”.

Κεφάλαιο 4

Αρχιτεκτονική του λογισμικού Apple iOS για iPhone και του Google Android

4.1 Η αρχιτεκτονική λογισμικού Apple iOS για τα τηλέφωνα iPhone

Όπως έχει ήδη αναφερθεί στην ενότητα 3, το λογισμικό της Apple είναι κλειστού τύπου και για το λόγο αυτό δεν είναι όλες οι λεπτομέρειες υλοποίησής του γνωστές. Παρόλα αυτά, θα γίνει αναφορά στα κύρια συστατικά-μέρη του αλλά και σε κάποια τμήματα υλικού που είναι υπεύθυνα για αρκετές από τις δημοφιλείς λειτουργίες του.

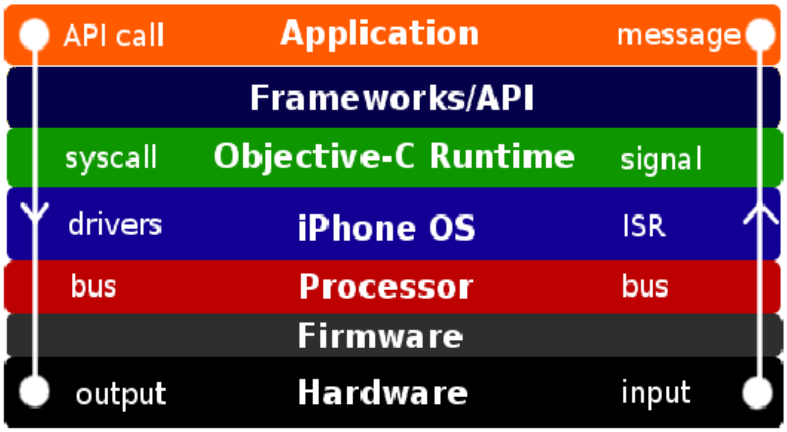
Το λειτουργικό σύστημα διαχειρίζεται το υλικό της συσκευής αλλά παρέχει και τις απαραίτητες τεχνολογίες για την υλοποίηση εφαρμογών από τρίτους στο τηλέφωνο. Το λειτουργικό σύστημα του iPhone στηρίζεται στο UNIX και έτσι πολλές από τις τεχνολογίες που συνθέτουν τα κατώτερα επίπεδα του λειτουργικού συστήματος προέρχονται από τεχνολογίες ανοικτού κώδικα. Επιπλέον, τα Interfaces αυτών των τεχνολογιών είναι ελεύθερα στην standard βιβλιοθήκη και τα interface directories.

Το λογισμικό iOS χρησιμοποιεί μια αρκετά απλή στοίβα λογισμικού. Στο κάτω μέρος αυτής της στοίβας βρίσκεται ο Mach πυρήνας και τα drivers του υλικού τα οποία διαχειρίζονται την εκτέλεση των προγραμμάτων στη συσκευή. Από πάνω από το επίπεδο αυτό υπάρχουν επιπλέον επίπεδα τα οποία περιλαμβάνουν τεχνολογίες πυρήνα και διεπαφές που χρησιμοποιούνται στην ανάπτυξη επιπλέον εφαρμογών όπως θα αναφερθεί λεπτομερώς στη συνέχεια της ενότητας. Παρόλο που το λογισμικό του iPhone δεν αποκαλύπτει καθόλου τον πυρήνα και τις διεπαφές των

drivers, αποκαλύπτει τεχνολογίες που βρίσκονται στα υψηλότερα επίπεδα της στοίβας [8].

Ενώ η αρχιτεκτονική του iOS δε χρησιμοποιείται ευρέως για τη δημιουργία εφαρμογών για το iPhone, παρουσιάζει αρκετή απλότητα όσον αφορά σχεδιαστικές επιλογές που βασίζονται στη θεωρία του λειτουργικού συστήματος. Βέβαια, η κατανόησή της οδηγεί στον καλύτερο χειρισμό ενός iPhone. Στην εικόνα 4.1 παρουσιάζεται η αρχιτεκτονική του iOS για το τηλέφωνο iPhone όπως παρέχεται από αρκετούς αναλυτές [6].

.



Εικόνα 4. 1 : Τα επίπεδα της αρχιτεκτονικής του iOS για τα κινητά τηλέφωνα iPhone

Όπως φαίνεται και στο σχήμα, υπάρχουν 7 επίπεδα αναφοράς. Αρχίζοντας από κάτω προς τα πάνω θα γίνει αναφορά στο τι αφορά κάθε επίπεδο. Αρχικά συναντάται το επίπεδο hardware το οποίο αναφέρεται κυρίως στα πραγματικά chips των περιφερειακών αισθητήρων του τηλεφώνου, όπως είναι η οθόνη και το accelerometer. Ο πραγματικός επεξεργαστής βρίσκεται στην ουσία κάτω από αυτό το επίπεδο αλλά το σύνολο των εντολών που χρησιμοποιούνται και οι πίνακες περιγραφών της μνήμης περιλαμβάνονται στο επίπεδο του processor. Είναι γνωστό ότι στο τηλέφωνο iPhone 3GS CPU είναι η ARM Cortex-A8 ενώ χρησιμοποιείται 32KB L1 cache και 13 stage pipeline [8].

Στη συνέχεια υπάρχει το επίπεδο Firmware. Αν και συνήθως με τον όρο firmware αναφέρεται κανείς σε ολόκληρο το λειτουργικό σύστημα, σε αυτή την αρχιτεκτονική αναφέρεται στον κώδικα που αφορά κάθε chip ξεχωριστά. Αυτό, είτε περιλαμβάνεται μέσα ή γύρω από τον περιφερειακό αισθητήρα τον ίδιο είτε μέσα στο ειδικό πεδίο (drive) που υπάρχει για κάθε συγκεκριμένο αισθητήρα.

Ακολούθως βρίσκεται το επίπεδο Processor όπου δεν αφορά τόσο το ARM chip που περιλαμβάνεται στο επίπεδο Hardware. Στην πραγματικότητα αναφέρεται στο σύνολο εντολών ARM και στον πίνακα περιγραφής διακοπών (interrupts) όπως αυτός ορίζεται από το λειτουργικό σύστημα του iPhone κατά τη διάρκεια της εκκίνησης και της αρχικοποίησης των drivers.

Ανεβαίνοντας ένα βήμα πιο πάνω αποκαλύπτεται το επίπεδο iPhone OS το οποίο περιλαμβάνει τον πυρήνα, τα drivers αλλά και τις υπηρεσίες που αποτελούν το λειτουργικό σύστημα του iPhone. Το λειτουργικό σύστημα iOS βρίσκεται μεταξύ του περιβάλλοντος χρήστη και του hardware.

Ακολούθως βρίσκεται το επίπεδο Objective-C Runtime το οποίο αποτελείται από τις δυναμικές βιβλιοθήκες της Objective-C αλλά και τις βασικές βιβλιοθήκες της C. Η βιβλιοθήκη της C χτίζει το περιβάλλον εκτέλεσης της Objective-C και για το λόγο αυτό περιλαμβάνονται στο ίδιο επίπεδο αρχιτεκτονικής, ενώ αμέσως πιο πάνω ακολουθεί το επίπεδο των Frameworks/API.

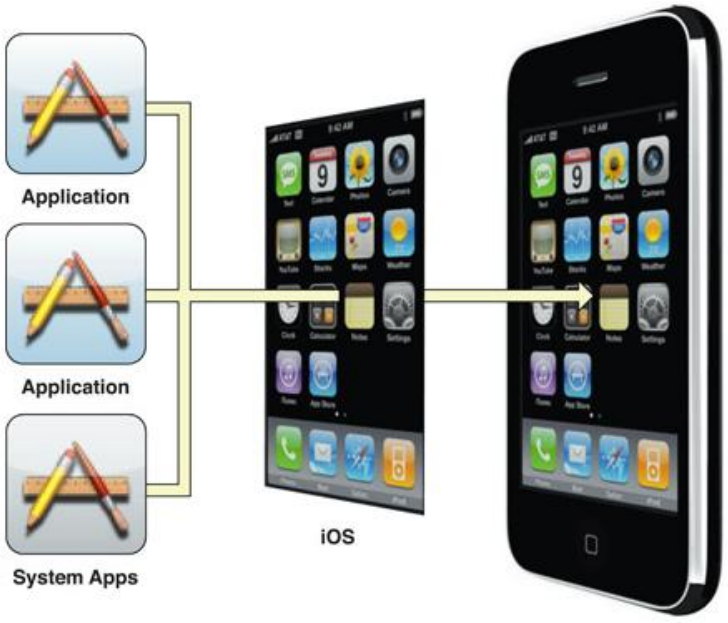
Στο επίπεδο Frameworks/API περιλαμβάνονται τα cocoa touch, οι κλήσεις ανώτερων επιπέδων OpenGL κτλ. Το cocoa touch είναι ένα πρωτόκολλο για την κατασκευή προγραμμάτων λογισμικού που τρέχει στο iPhone, στο iPod και στο iPad και το παρέχει η Apple. Το cocoa touch παρέχει ένα επιπλέον επίπεδο αφαίρεσης στο iOS και είναι γραμμένο σε Objective-C. Επιτρέπει τη χρήση υλικού και χαρακτηριστικών που δεν είναι διαθέσιμα στους MAC OS X υπολογιστές. Το πρωτόκολλο αυτό ακολουθεί Model View Controller αρχιτεκτονική (MVC) [55]. Όλα τα πρωτόκολλα που περιλαμβάνονται σε αυτό το επίπεδο διανέμονται από την Apple

μαζί με το iPhone SDK σε συνδυασμό με δυναμική σύνδεση που παρέχεται στο χρόνο εκτέλεσης.

Το ανώτερο επίπεδο που εμφανίζεται στην αρχιτεκτονική του iPhone είναι το επίπεδο των εφαρμογών, ή αλλιώς Application layer. Στο επίπεδο αυτό βρίσκονται οι εφαρμογές που «τρέχουν» ανά πάσα στιγμή στο τηλέφωνο και που εγκαταστάθηκαν σε αυτό μέσα από το appStore. Όλες οι εφαρμογές που βρίσκονται στο τηλέφωνο, έχουν μεταφραστεί σε native code μέσω του μεταγλωττιστή της Apple και έχουν συνδεθεί με το περιβάλλον εκτέλεσης μέσω Objective-C. Επίσης, οι εφαρμογές που βρίσκονται σε αυτό το επίπεδο, δηλαδή σε λειτουργία μια συγκεκριμένη στιγμή, τρέχουν αποκλειστικά στο περιβάλλον του χρήστη που φτιάχνεται από το λειτουργικό σύστημα του τηλεφώνου.

Το iOS διαχειρίζεται το υλικό της συσκευής και παρέχει τις απαραίτητες τεχνολογίες που απαιτούνται για την υλοποίηση εφαρμογών από τρίτους. Στην πράξη, το λειτουργικό αυτό σύστημα παραδίδεται στο χρήστη με αρκετές εφαρμογές συστήματος, όπως είναι οι τηλεφωνικές κλήσεις, το ηλεκτρονικό ταχυδρομείο και ο browser Safari οι οποίες παρέχουν τις απαραίτητες υπηρεσίες. Από την άλλη, το iOS SDK περιλαμβάνει τα εργαλεία και τα ενδιάμεσα λογισμικά που είναι απαραίτητα για την ανάπτυξη, εγκατάσταση, εκτέλεση και έλεγχο των εφαρμογών που φτιάχνονται από τρίτους. Για την κατασκευή τέτοιων εφαρμογών απαιτείται η χρήση των frameworks του συστήματος του iOS καθώς και η γλώσσα Objective-C. Οι εφαρμογές αυτές εκτελούνται κατευθείαν σε iOS λογισμικό.

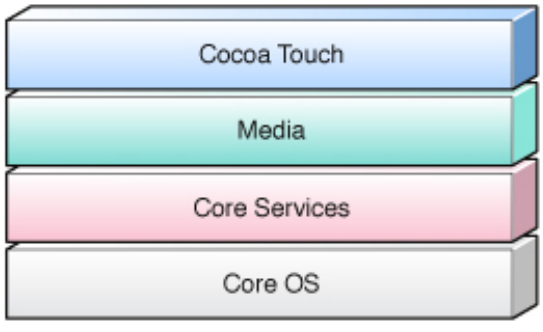
Η αρχιτεκτονική του iOS είναι παρόμοια με αυτή του Mac OS X. Στο υψηλότερο επίπεδο το λογισμικό δρα ως ενδιάμεσο μεταξύ του υλικού και των εφαρμογών που εμφανίζονται στην οθόνη όπως φαίνεται και στην εικόνα 4.2 [9]



Εικόνα 4. 2 Οι εφαρμογές που τρέχουν "από πάνω" από ένα iOS

Οι εφαρμογές που δημιουργεί ο χρήστης σπάνια απευθύνονται στο υλικό κατευθείαν. Αντίθετα, οι εφαρμογές επικοινωνούν με το υλικό μέσω ενός συνόλου καλώς ορισμένων διεπαφών του συστήματος. Ο λόγος για τον οποίο συμβαίνει αυτό είναι για την προστασία της εφαρμογής του χρήστη από τυχόν αλλαγές στο υλικό. Έτσι με αυτή την αφαίρεση γίνεται πιο εύκολη η δημιουργία εφαρμογών που δουλεύουν ανεπηρέαστες σε συσκευές με διαφορετικές ικανότητες υλικού.

Όσον αφορά την ανάπτυξη εφαρμογών για iOS η υλοποίηση των τεχνολογιών μπορεί να γίνει αντιληπτή ως ένα σύνολο επιπέδων όπως παρουσιάζεται και στην εικόνα 4.3. Στα κατώτερα επίπεδα του συστήματος βρίσκονται οι θεμελιώδεις λειτουργίες/υπηρεσίες και τεχνολογίες στις οποίες στηρίζονται όλες οι εφαρμογές. Στα υψηλότερα επίπεδα περιλαμβάνονται πιο εξειδικευμένες υπηρεσίες και τεχνολογίες.



Εικόνα 4. 3 Τα επίπεδα του iOS που πρέπει να ληφθούν υπόψη για την ανάπτυξη νέων εφαρμογών

Τα frameworks που υπάρχουν στα υψηλότερα επίπεδα στόχο έχουν να παρέχουν αφαιρέσεις για υπηρεσίες χαμηλότερων επιπέδων. Αυτές οι αφαιρέσεις κάνουν ευκολότερη τη συγγραφή κώδικα λόγω του ότι μειώνουν το ποσό του κώδικα που πρέπει να γραφτεί και ενθυλακώνουν πιθανά σύνθετα χαρακτηριστικά όπως είναι τα sockets και τα threads.

Το επίπεδο Cocoa Touch περιλαμβάνει τα σημαντικά frameworks που χρησιμοποιούνται στην κατασκευή iOS εφαρμογών. Καθορίζει τη βασική υποδομή των εφαρμογών και υποστηρίζει τις κυριότερες τεχνολογίες όπως multitasking, touch-based input, push notifications καθώς και άλλες υπηρεσίες συστήματος υψηλού επιπέδου. Με άλλα λόγια, καθοδηγεί τις αντιδράσεις του χρήστη στο iOS. Έχει κατασκευαστεί επάνω από το Model-View-Controller και παρέχει μια συμπαγή βάση για τη δημιουργία εφαρμογών σε συνδυασμό με το εργαλείο Interface Builder. Υπάρχουν αρκετά frameworks στο Cocoa Touch εκ των οποίων τα σημαντικότερα είναι: Address Book UI Framework, Event Kit UI Framework, Game Kit Framework, iAd Framework, Map Kit Framework, Message UI Framework και UIKit Framework.

Ένα επίπεδο πιο κάτω βρίσκεται το επίπεδο Media που περιλαμβάνει τα γραφικά, τον ήχο και τις τεχνολογίες δικτύου. Ο τρόπος με τον οποίο όλα αυτά βρίσκονται συγκεντρωμένα μαζί δίνει τη δυνατότητα για δημιουργία εφαρμογών πολυμέσων στην κινητή συσκευή. Οι τεχνολογίες αυτού του επιπέδου σχεδιάστηκαν έτσι ώστε να είναι εύκολα κατανοητές από οποιονδήποτε θελήσει να αναπτύξει εφαρμογές για το iPhone.

Τα γραφικά υψηλής ποιότητας είναι σημαντικό μέρος όλων των iOS εφαρμογών. Ο πιο εύκολος τρόπος για τη δημιουργία τέτοιων εφαρμογών είναι η ενσωμάτωση πολλών εικόνων μαζί με τη χρήση του UIKit framework που αναφέρθηκε στο προηγούμενο επίπεδο. Όσον αφορά τις τεχνολογίες που έχουν να κάνουν με τον ήχο, είναι σχεδιασμένες να παρέχουν την ικανότητα αναπαραγωγής υψηλής ποιότητας ήχου, ηχογράφηση υψηλής ποιότητας ήχου αλλά και την ενεργοποίηση της δόνησης τηλεφώνου σε συγκεκριμένες συσκευές. Το σύστημα παρέχει αρκετούς

τρόπους για την αναπαραγωγή και ηχογράφηση περιεχομένου με ήχο. Αυτό επιτυγχάνεται μέσω κάποιων υλοποιημένων frameworks που βρίσκονται σε αυτό το επίπεδο.

Όσον αφορά τις τεχνολογίες βίντεο, το σύστημα παρέχει αρκετούς τρόπους αναπαραγωγής και λήψης περιεχομένου ανάλογα με τις ανάγκες του χρήστη. Υπάρχει η επιλογή χρήσης πρωτοκόλλων υψηλότερων επιπέδων όπου απαιτείται λιγότερη προγραμματιστική δουλειά από το χρήστη αλλά και η επιλογή χρήσης πρωτοκόλλων χαμηλότερου επιπέδου για τη δημιουργία εξειδικευμένων εφαρμογών. Και σε αυτή την περίπτωση όλες οι επιλογές παρέχονται μέσω υλοποιημένων Frameworks.

Προχωρώντας ακόμα ένα επίπεδο πιο κάτω συναντάμε το Core Services επίπεδο, δηλαδή το επίπεδο το οποίο περιέχει τις σημαντικότερες υπηρεσίες συστήματος τις οποίες όλες οι εφαρμογές χρησιμοποιούν. Ακόμα και αν κάποιες εφαρμογές δεν τις χρησιμοποιούν απευθείας, σίγουρα υπάρχουν κομμάτια τους που στηρίζονται σε αυτές για τη λειτουργία τους. Οι σημαντικότερες από τις τεχνολογίες του επιπέδου παρουσιάζονται στη συνέχεια.

Αρχικά, τα Block Objects. Τα Block Objects εισήχθησαν στο iOS 4.0. και είναι στην πράξη τμήματα γλώσσας C που είναι δυνατόν να ενσωματωθούν μέσα στον κώδικα C και Objective-C. Ένα τέτοιο αντικείμενο αποτελείται στην ουσία από μια ανώνυμη συνάρτηση και τα δεδομένα που τη συνοδεύουν. Συνήθως χρησιμεύουν ως callbacks ή σε μέρη όπου απαιτείται ευκολία στο συνδυασμό τόσο του κώδικα που θα εκτελεστεί όσο και στα δεδομένα που σχετίζονται με αυτόν. Στο iOS τα blocks χρησιμοποιούνται κυρίως στα ακόλουθα σενάρια: Ως αντικατάσταση των delegate methods και των callback συναρτήσεων, για υλοποίηση completion handlers για πράξεις που θα γίνουν μόνο μια φορά, για τη διευκόλυνση της πραγματοποίησης μιας ενέργειας σε όλα τα αντικείμενα μιας συλλογής αλλά και μαζί με dispatch queues για την πραγματοποίηση ασύγχρονων εργασιών.

Στη συνέχεια υπάρχει το Grand Central Dispatch (GCD) όπου και αυτό χρησιμοποιήθηκε για πρώτη φορά στο iOS 4.0. Πρόκειται για μια BSD επιπέδου τεχνολογία την οποία χρησιμοποιεί κανείς για τη διαχείριση των εργασιών σε μια εφαρμογή. Συνδυάζει ένα ασύγχρονο προγραμματιστικό μοντέλο μαζί με ένα βέλτιστο πυρήνα για την παροχή εναλλακτικής και πιο εύκολης λύσης για τη διαχείριση των νημάτων (threading). Παρέχει επίσης εναλλακτικές λύσεις για αρκετές εργασίες χαμηλού επιπέδου όπως το διάβασμα και το γράψιμο περιγραφών αρχείων, η υλοποίηση χρονομέτρων, η παρακολούθηση σημάτων και η επεξεργασία γεγονότων.

Στην έκδοση 3.0 του iOS εμφανίστηκε για πρώτη φορά το In-App Purchase το οποίο δίνει στο χρήστη του τηλεφώνου την ικανότητα να συναλλάσσεται περιεχόμενο και υπηρεσίες μέσα από την εφαρμογή του. Αυτό το χαρακτηριστικό υλοποιείται μέσω του Store Kit framework το οποίο παρέχει την κατάλληλη υποδομή για την επεξεργασία οικονομικών συναλλαγών χρησιμοποιώντας το λογαριασμό του χρήστη στο iTunes.

Σε αυτό το επίπεδο υπάρχει και το SQLite. Η βιβλιοθήκη του SQLite επιτρέπει την ενσωμάτωση μιας «ελαφριάς» SQL βάσης δεδομένων μέσα στην εφαρμογή του χρήστη χωρίς να χρειαστεί η εκτέλεση επιπλέον διεργασιών του server της βάσης δεδομένων. Από την εφαρμογή, ο χρήστης μπορεί να δημιουργήσει τοπικές βάσεις αρχείων και να διαχειριστεί τους πίνακες και τις εγγραφές αυτών των βάσεων. Η βιβλιοθήκη έχει σχεδιαστεί για γενικού σκοπού βάσεις δεδομένων αλλά συνεχώς βελτιστοποιείται για την παροχή γρήγορης πρόσβασης στις εγγραφές της εκάστοτε βάσης δεδομένων.

Τέλος, υπάρχει και υποστήριξη για XML. Το Foundation framework παρέχει την κλάση NSXML Parser για την ανάκτηση στοιχείων από ένα XML έγγραφο. Επιπλέον υποστήριξη για τη διαχείριση XML περιεχομένου παρέχεται μέσω της βιβλιοθήκης libXML2. Πρόκειται για μια ανοικτού κώδικα βιβλιοθήκη που επιτρέπει τη μεταγλώττιση ή τη συγγραφή πηγαίων XML δεδομένων γρήγορα και τη μετατροπή του XML περιεχομένου σε HTML.

Στο επίπεδο Core Services υπάρχουν και αρκετά υλοποιημένα frameworks Που παρέχουν σημαντικές λειτουργίες. Τα σημαντικότερα από αυτά είναι: Address Book framework, CFNetwork framework, Core Data framework, Core Foundation framework, Core Location framework, Core Media framework, Core Telephony framework, EvenKit framework, Foundation framework, Mobile Core Services framework, Quick Look framework, Store Kit framework και System Configuration framework.

Τέλος, υπάρχει το επίπεδο Core OS το οποίο περιλαμβάνει τα χαμηλού επιπέδου χαρακτηριστικά στα οποίο στηρίζονται οι περισσότερες από τις τεχνολογίες που υλοποιούνται στα ανώτερα επίπεδα. Ακόμα και αν αυτές οι τεχνολογίες δεν χρησιμοποιούνται ξεκάθαρα στις εφαρμογές, χρησιμοποιούνται από τεχνολογίες τις οποίες χρησιμοποιούν οι εφαρμογές. Ακόμα και σε περιπτώσεις όπου υπάρχει η ανάγκη αντιμετώπισης θεμάτων ασφάλειας και επικοινωνίας με ένα εξωτερικό τμήμα υλικού χρησιμοποιούνται τα frameworks του επιπέδου αυτού.

Μεταξύ άλλων υλοποιούνται και τα ακόλουθα frameworks σε αυτό το επίπεδο. Αρχικά το Accelerate framework όπου χρησιμοποιήθηκε για πρώτη φορά στο iOS 4.0. Περιλαμβάνει τις διεπαφές που απαιτούνται για την πραγματοποίηση μαθηματικών πράξεων και σύνθετων υπολογισμών. Το πλεονέκτημα από τη χρήση αυτού του framework είναι ότι είναι ήδη βελτιστοποιημένο για όλες τις iOS συσκευές. Έτσι αν ο χρήστης γράψει μια φορά τον κώδικα της εφαρμογής του χρησιμοποιώντας το framework αυτό μπορεί να το επαναχρησιμοποιήσει πολλές φορές με την ασφάλεια ότι αυτό θα λειτουργεί χωρίς κανένα πρόβλημα σε όλες τις iOS συσκευές.

Υπάρχει επίσης και το External Accessory framework το οποίο χρησιμοποιήθηκε για πρώτη φορά στο iOS 3.0. Παρέχει υποστήριξη για επικοινωνία με εξαρτήματα υλικού που συνδέονται σε μια iOS συσκευή. Τα εξαρτήματα αυτά μπορούν να είναι συνδεδεμένα μέσω ενός 30-pin dock συνδετήρα συσκευής ή ασύρματα με χρήση Bluetooth. Επιπλέον, το framework αυτό παρέχει ένα τρόπο λήψης πληροφοριών σχετικές με κάθε εξάρτημα και την αρχικοποίηση επικοινωνίας με αυτά. Ακολούθως,

ο χρήστης είναι σε θέση να χειριστεί το εξάρτημα χρησιμοποιώντας οποιεσδήποτε εντολές υποστηρίζει αυτό.

Πολύ σημαντικό framework είναι και το Security framework. Ενώ υπάρχουν ενσωματωμένα χαρακτηριστικά ασφάλειας στο iOS, παρέχεται επίσης αυτό το ανεξάρτητο framework το οποίο μπορεί να χρησιμοποιήσει ο χρήστης για την εγγύηση ασφάλειας των δεδομένων τα οποία διαχειρίζεται η εφαρμογή του. Παρέχει τις διεπαφές που απαιτούνται για τη διαχείριση των πιστοποιητικών, των δημόσιων και ιδιωτικών κλειδιών και τις πολιτικές εμπιστοσύνης. Υποστηρίζει τη δημιουργία κρυπτογραφημένων, ασφαλών και ψευδό-τυχαίων αριθμών. Επίσης υποστηρίζει την αποθήκευση των πιστοποιητικών και των κρυπτογραφημένων κλειδιών στο keychain που είναι μια ασφαλής τοποθεσία αποθήκευσης ευαίσθητων δεδομένων του χρήστη.

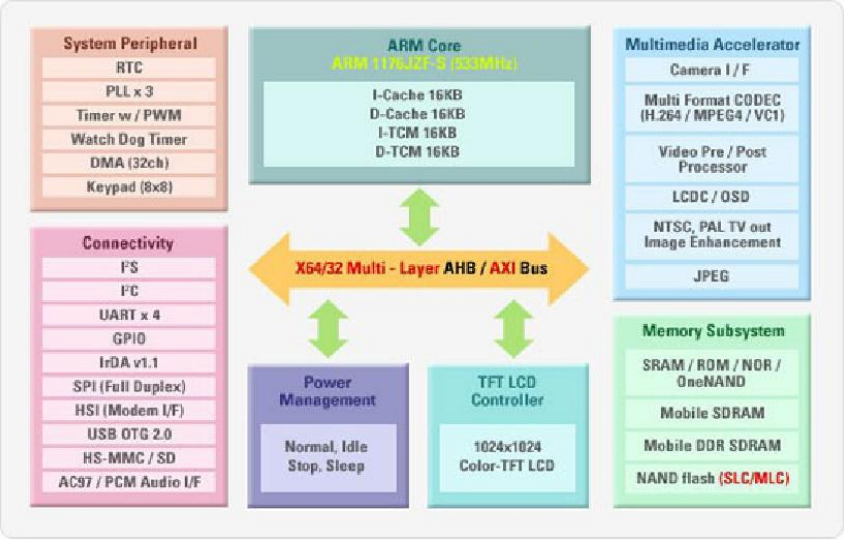
Τέλος, υπάρχει το σύστημα σε αυτό το επίπεδο το οποίο καλύπτει το περιβάλλον του πυρήνα, τα drivers και τις χαμηλού επιπέδου UNIX διεπαφές του λειτουργικού συστήματος. Ο πυρήνας βασίζεται σε Mach και είναι υπεύθυνος για κάθε πτυχή του λειτουργικού συστήματος. Διαχειρίζεται την εικονική μνήμη του συστήματος, τα νήματα, το σύστημα αρχείων, το δίκτυο και την επικοινωνία μεταξύ των διεργασιών. Τα drivers που βρίσκονται σε αυτό το επίπεδο παρέχουν επίσης τη διεπαφή μεταξύ του διαθέσιμου υλικού και των Frameworks του συστήματος. Για σκοπούς ασφαλείας η πρόσβαση στον πυρήνα και στα drivers είναι περιορισμένη σε ένα μικρό σύνολο frameworks συστήματος και εφαρμογές. Το iOS παρέχει ένα σύνολο διεπαφών για την πρόσβαση σε πολλά χαμηλού επιπέδου χαρακτηριστικά μέσω της βιβλιοθήκης LibSystem. Οι διεπαφές βασίζονται σε γλώσσα C και παρέχουν υποστήριξη για τα ακόλουθα: threading, networking, πρόσβαση σε σύστημα αρχείων, standard I/O, τοπικές πληροφορίες, δέσμευση μνήμης και μαθηματικούς υπολογισμούς.

Οι εφαρμογές του iPhone εμφανίζονται ως εικόνες στην οθόνη του χρήστη. Αντίθετα από τις εφαρμογές του διαδικτύου που τρέχουν μέσω του Safari browser, μια εφαρμογή στο iOS τρέχει απευθείας ως ανεξάρτητο και εκτελέσιμο τμήμα της συσκευής. Τέτοιες εφαρμογές έχουν πρόσβαση σε όλα τα χαρακτηριστικά του iPhone

όπως multi-touch, location service κτλ. Επιπλέον, μπορούν να αποθηκεύσουν πληροφορίες και δεδομένα στο τοπικό σύστημα αρχείων και να επικοινωνήσουν με άλλες εγκαταστημένες εφαρμογές μέσω τυπικών σχημάτων URL [8]. Οι εφαρμογές αυτές αναπτύσσονται με το framework UIKit το οποίο παρέχει την θεμελιώδη υποδομή μιας εφαρμογής αλλά και την προκαθορισμένη συμπεριφορά της. Εκτός άλλων, τα frameworks παρέχουν τμήματα τα οποία ο χρήστης μπορεί να προσωποποιήσει για να επεκτείνει αυτή τη συμπεριφορά.

Κάθε εφαρμογή του iPhone φτιάχνεται μέσω αυτού του Framework και για το λόγο αυτό έχει την ίδια αρχιτεκτονική πυρήνα με τις άλλες εφαρμογές. Το UIKit παρέχει τα αντικείμενα που απαιτούνται για την εκτέλεση της εφαρμογής και το συντονισμό της εισόδου του χρήστη και της εμφάνισης του περιεχομένου στην οθόνη. Το σημείο στο οποίο διαφέρουν οι εφαρμογές είναι ο τρόπος με τον οποίο διαμορφώνουν αυτά τα αντικείμενα αλλά και το πώς αλληλεπιδρούν τα αντικείμενα για τις ανάγκες τις εφαρμογής όσον αφορά την εμφάνιση αλλά και τη συμπεριφορά.

Στην εικόνα 4.4 [10] παρουσιάζεται η αρχιτεκτονική του iPhone από ένα υψηλότερο επίπεδο. Σύμφωνα με την εικόνα αυτή, στο iPhone χρησιμοποιείται ARM επεξεργαστής ο οποίος είναι RISC-based. Μπορεί να γίνει διαμόρφωση σε χαμηλά ή υψηλά επίπεδα για πρόσβαση στα δεδομένα. Το iPhone τρέχει αυτό τον επεξεργαστή σε little-endian mode με αποτέλεσμα όταν μια τιμή στον καταχωρητή είναι 0x12345678 εμφανίζεται στη μνήμη ως 0x78 0x56 0x34 0x12 [10].



Εικόνα 4. 4 Η αρχιτεκτονική του iPhone από ένα υψηλότερο επίπεδο

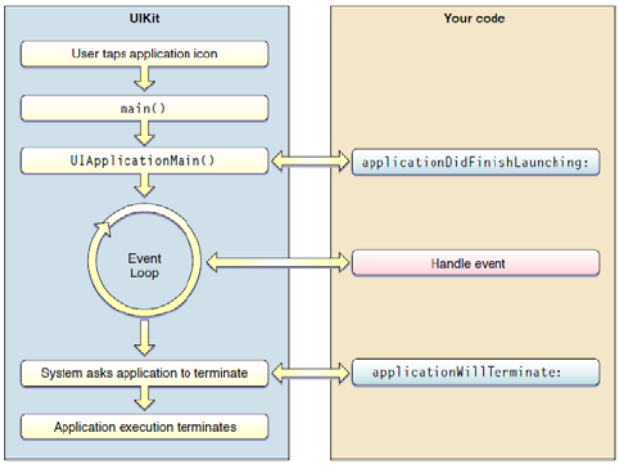
Ακόμα ένα σημαντικό χαρακτηριστικό της αρχιτεκτονικής του ARM είναι το γεγονός ότι η στοίβα είναι μη-εκτελέσιμη σε αντίθεση με την αρχιτεκτονική x86. Η στοίβα λοιπόν επεκτείνεται προς τα κάτω στον ARM.

Πρόκειται στην ουσία για ένα επεξεργαστή υψηλής απόδοσης όσον αφορά το σύστημα μνήμης. Υποστηρίζει 4-64K μεγέθη cache [11]. Το σύστημα μνήμης ARMv6 επιταχύνει την αλλαγή περιεχομένου του λειτουργικού συστήματος. Προς το παρόν η RAM μνήμη που υποστηρίζεται στο iPhone ανέρχεται στα 256MB και η ταχύτητα επεξεργαστή στα 600MHZ.

Στη συνέχεια παρουσιάζονται στοιχεία που αφορούν την εκτέλεση εφαρμογών στο iOS και τον κύκλο ζωής τους. Στο iOS κάθε εφαρμογή αποτελείται από ένα ή περισσότερα νήματα (threads). Κάθε νήμα αναπαριστά ένα μοναδικό μονοπάτι εκτέλεσης. Κάθε εφαρμογή αρχίζει με ένα μόνο νήμα το οποίο εκτελεί την main συνάρτηση της εφαρμογής. Οι εφαρμογές είναι πιθανό να έχουν περισσότερα νήματα τα οποία εκτελούν το καθένα μια ξεχωριστή συνάρτηση. Όταν μια εφαρμογή παράγει ένα νέο νήμα, αυτό μετατρέπεται σε ανεξάρτητη οντότητα μέσα στο χώρο διεργασιών. Κάθε νήμα έχει τη δική του στοίβα εκτέλεσης και ο πυρήνας προγραμματίζει τον χρόνο εκτέλεσής του. Καθώς τα νήματα βρίσκονται στον ίδιο χώρο διεργασιών μπορούν να επικοινωνούν με άλλα νήματα και διεργασίες. Όλα τα νήματα μιας εφαρμογής μοιράζονται τον ίδιο χώρο εικονικής μνήμης και έχουν τα ίδια δικαιώματα πρόσβασης όπως και η διεργασία. Κάθε νήμα απαιτεί δέσμευση μνήμης τόσο στον πυρήνα όσο και στο πρόγραμμα. Ο πυρήνας είναι αυτός που διαχειρίζεται το νήμα και συντονίζει το χρονοπρογραμματισμό του. Ο χώρος της στοίβας του νήματος και τα δεδομένα του αποθηκεύονται στο χώρο μνήμης του προγράμματος. Όταν το νήμα δημιουργείται, δημιουργούνται και αρχικοποιούνται οι περισσότερες δομές του. Για τη δημιουργία ενός νήματος χαμηλού επιπέδου, απαιτείται μια συνάρτηση ή μέθοδος για να δράσει ως το κύριο σημείο εισόδου του νήματος ενώ στη συνέχεια μια από τις διαθέσιμες μεθόδους νημάτων χρησιμοποιείται για την εκκίνηση του νήματος. Ένα νήμα μπορεί να δημιουργηθεί χρησιμοποιώντας διάφορες

μεθόδους όπως NSThread, POSIX threads, NSObjec. Μετά τη δημιουργία του νήματος απαιτείται αρχικοποίηση του μεγέθους της στοίβας, του τοπικού χώρου αποθήκευσης αλλά και ορισμός της κατάστασης του νήματος και της προτεραιότητάς του [6].

Είναι γνωστό ότι κάθε εφαρμογή έχει το δικό της κύκλο ζωής σε όλα τα λειτουργικά συστήματα. Έτσι, και στο iOS κάθε εφαρμογή ακολουθεί τη δική της πορεία από τη δημιουργία της μέχρι και τον τερματισμό της. Στο iOS ο κύκλος ζωής μιας εφαρμογής αποτελείται από μια ακολουθία γεγονότων που προκύπτουν μεταξύ της έναρξης και του τερματισμού της όπως φαίνεται ξεκάθαρα και στην εικόνα 4.5. Για την εκκίνηση μιας εφαρμογής ο χρήστης επιλέγει το αντίστοιχο εικονίδιό του από την αρχική οθόνη. Αμέσως μετά, το σύστημα εμφανίζει κάποια γραφικά στην οθόνη ενώ από αυτό το σημείο και μετά καλεί την κύρια συνάρτηση της εφαρμογής (main) για να την εκκινήσει. Ακολουθεί η αρχικοποίηση της εφαρμογής που συμβαίνει αφού ο έλεγχος περάσει στο UIKit. Έτσι, φορτώνεται η διεπαφή της εφαρμογής και ετοιμάζεται ο βρόγχος του γεγονότος.



Εικόνα 4. 5 Ο κύκλος ζωής μιας εφαρμογής στο iOS

Κατά τη διάρκεια του βρόχου του γεγονότος το UIKit συντονίζει την παράδοση των γεγονότων στα προσωποποιημένα αντικείμενα του χρήστη και απαντά σε εντολές που εκδίδονται από την εφαρμογή. Όταν ο χρήστης πραγματοποιήσει μια

ενέργεια που θα προκαλέσει το κλείσιμο της εφαρμογής το UIKit ειδοποιεί την εφαρμογή και αρχίζει τον τερματισμό της διεργασίας.

Όσον αφορά το περιβάλλον εκτέλεσης μιας εφαρμογής υπάρχουν 4 κύριες συνιστώσες που το αποτελούν. Αρχικά υπάρχει η αρχή του “Fast Launch, Short Use” κατά την οποία μόνο μια εφαρμογή τρέχει στο προσκήνιο ανά πάσα στιγμή. Κάθε εφαρμογή πρέπει να εκτελεστεί και να αρχικοποιηθεί από μόνη της και σε σύντομο χρονικό διάστημα και να είναι προετοιμασμένη να τερματίσει γρήγορα. Ακολούθως υπάρχει το Application Sandbox όπου είναι ένα σύνολο από καλώς ορισμένους ελέγχους που περιορίζουν την πρόσβαση κάθε εφαρμογής σε αρχεία, προτιμήσεις, πηγές δικτύου και υλικό. Για λόγους ασφαλείας το iOS περιορίζει την εφαρμογή σε μια μοναδική τοποθεσία στο σύστημα αρχείων.

Μια τρίτη πτυχή του περιβάλλοντος εκτέλεσης εφαρμογών αποτελεί και το εικονικό σύστημα μνήμης σύμφωνα με το οποίο κάθε πρόγραμμα εξακολουθεί να έχει το δικό του, εικονικό χώρο διευθύνσεων αλλά η χρησιμοποιήσιμη εικονική μνήμη του περιορίζεται από το ποσό της διαθέσιμης φυσικής μνήμης. Τέλος, υπάρχει το Automatic Sleep Timer για την εξοικονόμηση ενέργειας. Όμως τα άτομα που αναπτύσσουν εφαρμογές για τα iPhone μπορούν να το απενεργοποιούν. Έτσι, μόνο εφαρμογές που δεν στηρίζονται σε εισόδους από την οθόνη αλλά που χρειάζεται να εμφανίζουν εικονικό περιεχόμενο στην οθόνη της συσκευής το χρησιμοποιούν. Αν το σύστημα δεν ανιχνεύσει γεγονότα αφής (touch events) για εκτεταμένη περίοδο χρόνου χαμηλώνει το φωτισμό της οθόνης και στη συνέχεια την σβήνει εντελώς για εξοικονόμηση μπαταρίας.

4.2 Η αρχιτεκτονική λογισμικού του Google Android

Στόχος του Android OS ήταν από την αρχή η ενσωμάτωσή ενός λογισμικού βασισμένου σε Linux και μιας στοίβας λογισμικού σε πολλές συσκευές, διαφόρων

κατασκευαστών, παράλληλα με τη διάθεση ενδιάμεσου λογισμικού (middleware) για τη διευκόλυνση της ανάπτυξης εφαρμογών για κινητά τηλέφωνα.

Με μια πρώτη ματιά η αρχιτεκτονική του λογισμικού Android φαίνεται αρκετά πολύπλοκη. Παρουσιάζει αρκετά επίπεδα όπως φαίνεται και στην εικόνα 4.6 και τα οποία αναλύονται λεπτομερώς στη συνέχεια της ενότητας. Στην προσπάθεια να προσελκύσει τους κατασκευαστές κινητών τηλεφώνων παρέχει ένα ψευδό-πρωτότυπο περιβάλλον ανάπτυξης εφαρμογών το οποίο τρέχει Android εφαρμογές σε συσκευές διάφορων κατασκευαστών [12].

Εικόνα 4. 6 Η αρχιτεκτονική του λογισμικού Android

Το λογισμικό Android αποτελείται από 5 συστατικά-μέρη: Applications, Application Framework, Libraries, Android Runtime και Linux Kernel. Ξεκινώντας από τον πυρήνα, ο πυρήνας 2.6 του Linux αποτελεί τη βάση της αρχιτεκτονικής του Android. Βασίζεται σε αυτόν όσον αφορά τις σημαντικές υπηρεσίες πυρήνα όπως η ασφάλεια, διαχείριση μνήμης και διεργασιών, στοίβα δικτύου και driver model [12]. Ο πυρήνας δρα και ως ένα επίπεδο αφαίρεσης μεταξύ του υλικού και των υπολοίπων τμημάτων της στοίβας λογισμικού [6]. Ο πυρήνας έχει τροποποιηθεί έτσι ώστε τα

drivers να υποστηρίζουν Qualcomm MSM7200 chipsets τα οποία παρέχουν λειτουργικότητα όπως: WCDMA/HSUPA και EGPRS δικτύωση, Bluetooth και WiFi, υποστήριξη ψηφιακού ήχου, Java hardware acceleration, κάμερα αρκετών megapixels καθώς και GPS [13]. Όσον αφορά τη διαχείριση ενέργειας, έχει ενσωματωθεί στον πυρήνα ένα «ελαφρύ» driver που βασίζεται στο αρχικό driver διαχείρισης μνήμης του λειτουργικού συστήματος Linux. Η απόφαση αυτή πάρθηκε έτσι ώστε η KME (CPU) να μην καταναλώνει ενέργεια αν δεν υπάρχουν ενεργές εφαρμογές ή υπηρεσίες που απαιτούν ενέργεια αλλά και έτσι ώστε οι εφαρμογές και οι υπηρεσίες να ζητούν ενέργεια μέσω «κλειδωνιών» (locks).

Ο μηχανισμός ενδο-επικοινωνίας στο Android βασίζεται στο OpenBinder framework. Αυτό επιτρέπει στις διεργασίες να παρουσιάζουν τις διεπαφές τους η μια στην άλλη για κλήσεις συναρτήσεων. Το OpenBinder προσφέρει thread pools προ-επεξεργασίας για την επεξεργασία αιτήσεων και χειρίζεται τις μετρήσεις αναφορών και την αναδρομή πίσω στα αρχικά νήματα. Ακόμα ένα σημαντικό θέμα που αντιμετωπίζεται στο επίπεδο του πυρήνα είναι η διαχείριση μνήμης. Ο τροποποιημένος πυρήνας παρέχει ένα Low Memory Killer ο οποίος παρέχει ευελιξία τερματίζοντας («σκοτώνοντας») εφαρμογές όταν η μνήμη είναι περιορισμένη. Αυτές οι αποφάσεις λαμβάνονται μέσω ανάδρασης που παίρνει σε αντίθεση με τον Standard Out of Memory Killer ο οποίος τερματίζει εφαρμογές όταν το σύστημα δεν έχει πλέον καθόλου διαθέσιμη μνήμη [13]. Επιπλέον χρησιμοποιείται ένας διαχειριστής φυσικής μνήμης για την παροχή συνεχόμενων περιοχών φυσικής μνήμης σε βιβλιοθήκες στο χώρο του χρήστη που αλληλεπιδρούν με τον επεξεργαστή και άλλες συσκευές για τη βελτίωση της απόδοσης.

Το αμέσως επόμενο επίπεδο που αναπαριστάται με πράσινο στην εικόνα 4.6, περιλαμβάνει τις τοπικές βιβλιοθήκες (native libraries). Αυτές είναι γραμμένες σε C ή C++ για βελτίωση της απόδοσης και παρέχουν τυπική λειτουργικότητα βιβλιοθηκών πυρήνα όπως κλάσεις για βασικές πράξεις προς τη Γραφική Διασύνδεση Χρήστη, media codecs ή και πιστοποιητικά ασφαλείας για απόδοση πηγών. Οι βιβλιοθήκες

αυτού του επιπέδου είναι στην ουσία κατασκευαστικά κομμάτια και επίπεδα αφαίρεσης προς τα κατώτερα επίπεδα του συστήματος. Για το λόγο αυτό χρησιμοποιούνται από αρκετά συστατικά του Android συστήματος και είναι διαθέσιμες στους χρήστες που επιθυμούν να αναπτύξουν εφαρμογές για αυτό το λειτουργικό σύστημα. Το τμήμα Standard C library είναι η Bionic libC που βασίζεται στην BSD standard C βιβλιοθήκη μαζί με υλοποίηση νημάτων POSIX. Η βιβλιοθήκη αυτή είναι κατάλληλη για μικρές και ενσωματωμένες εγκαταστάσεις σε περιορισμένο υλικό. Ο surface manager δουλεύει ως μια αφαίρεση του driver της οθόνης. Είναι υπεύθυνος για τη σύνθεση διάφορων σχεδιαστικών επιφανειών (2D, 3D) από διάφορες διεργασίες και την εμφάνισή τους στην οθόνη. Τα διάφορα επίπεδα περνούν μέσω της OpenBinder IPC framework.

Το framework των πολυμέσων είναι το OpenCore που παρέχεται από την PacketVideo που είναι επίσης μέλος του Open Headset Alliance. Είναι ένα σύνολο από βιβλιοθήκες με παγκόσμια δομή για πολυμεσικές εφαρμογές κινητών τηλεφώνων. Παρέχει αναπαραγωγή αλλά και ηχογράφηση πολλών από τα γνωστά πρότυπα ήχου. Ο διαχειριστής ήχου (audio manager) χρησιμοποιείται για την επεξεργασία πολλών ροών ήχου μέσα σε PCM μονοπάτια εξόδου.

Όσον αφορά την αποθήκευση των δεδομένων (data storage), το Android OS χρησιμοποιεί SQLite. Πρόκειται για μια «ελαφριά» σχεσιακή βάση δεδομένων για ενσωματωμένα συστήματα. Είναι στην ουσία ένα συστατικό βιβλιοθήκης και όχι μια standalone υπηρεσία. Δίνει στις εφαρμογές πρόσβαση σε λειτουργικότητες αποθήκευσης μέσω απλών κλήσεων προς συναρτήσεις. Οι βάσεις δεδομένων αποθηκεύονται σε μοναδικά αρχεία τα οποία είναι ανεξάρτητα πλατφόρμας. Μπορεί η SQLite να μην παρέχει ένα κανονικό σύστημα σχεδίασης βάσεων δεδομένων αλλά η επιλογή για τη χρήση της στο Android OS ταιριάζει μια χαρά ως προς την απόδοση και την απλότητα των κινητών συσκευών.

Ένα ακόμα συστατικό αυτού του επιπέδου είναι το WebKit. Όπως ακριβώς και ο Safari στο iOS, έτσι και ο φυλλομετρητής που βρίσκεται ενσωματωμένος στο Android

OS είναι βασισμένος στην ανοικτού κώδικα Webkit rendering engine [56] . Έχουν γίνει κάποιες τροποποιήσεις για την ορθή εμφάνιση στις μικρές οθόνες των κινητών τηλεφώνων και υποστηρίζει επίσης και μοντέρνες τεχνολογίες όπως JavaScript, DOM, CSS και AJAX. Οι εφαρμογές του προεπιλεγμένου φυλλομετρητή του Android και οι χάρτες της Google βασίζονται γύρω από την WebKit μηχανή.

Το επόμενο συστατικό του Android OS είναι το Android Runtime ή αλλιώς ο χώρος εκτέλεσης. Αποτελείται από την Dalvik Virtual Machine και τις θεμελιώδεις βιβλιοθήκες πυρήνα σε Java που αναπτύχθηκαν από την Google και αποτελούν συστατικά-κλειδιά της στοίβας λογισμικού του Android OS. Οι συναρτήσεις πυρήνα σε Java παρέχουν τις κοινές δομές δεδομένων και τους αλγόριθμους που χρησιμοποιούνται σε υλοποιημένες βιβλιοθήκες. Αυτές περιλαμβάνουν λειτουργικότητες Εισόδου/Εξόδου, συλλογές, containers, utilities κτλ. Αυτό το τμήμα φανερώνει κάτι ασυνήθιστο για τα τυπικά συστήματα που βασίζονται σε Linux επειδή πρόκειται για μια Java Virtual Machine. Η JVM είναι υπεύθυνη για την εκτέλεση όλων των εφαρμογών στο περιβάλλον του Android. Ούτε οι εφαρμογές που αναπτύσσονται από τρίτους αλλά ούτε και οι εφαρμογές πυρήνα όπως το email, το ημερολόγιο δεν μπορούν να είναι γραμμένες σε γλώσσα C ή C++. Πρέπει να έχουν αναπτυχθεί σε Java. Με τον τρόπο αυτό η εικονική μηχανή λειτουργεί ως ένα επίπεδο αφαίρεσης μεταξύ των συστατικών του πυρήνα του συστήματος και του framework που χρησιμοποιείται για την ανάπτυξη εφαρμογών. Στο σημείο αυτό εντοπίζονται και κάποιες ομοιότητες με το λειτουργικό σύστημα της πλατφόρμας Blackberry. Αυτό που κάνει το Android να ξεχωρίζει από όλες τις υπόλοιπες πλατφόρμες που βασίζονται σε Java είναι η εικονική μηχανή που ονομάζεται Dalvik Virtual Machine η οποία δεν παρουσιάζει καθόλου ομοιότητες με τη Sun JVM ή KVM [12]. Η Dalvik VM έχει σχεδιαστεί για να τρέχει ειδικά σε περιορισμένο υλικό όπου ο επιπλέον χώρος δεν υπάρχει και η διάρκεια της μπαταρίας, ο επεξεργαστής, η μνήμη και ο χώρος αποθήκευσης είναι σημαντικά προς αντιμετώπιση θέματα [12]. Οι κυριότερες διαφορές αυτής της εικονικής μηχανής είναι: Αρχικά, η Dalvik VM είναι βασισμένη

στους καταχωρητές και όχι βασισμένη σε στοίβα. Με τον τρόπο αυτό απαιτεί λιγότερες εντολές για την αναπαράσταση του ίδιου κώδικα υψηλού επιπέδου απ' ότι θα χρειαζόταν στην περίπτωση της στοίβας. Χρησιμοποιεί μια ιδιωτική μορφή byte-code που ονομάζεται Dalvic Executable (.dex). Τα αρχεία .dex παράγονται από αρχεία .jar ή .class μέσω του dx εργαλείου. Για να είναι δυνατή η εκτέλεση σε Android των εφαρμογών που μεταγλωττίζονται στο desktop με JVM , πρώτα ο bytecode τους πρέπει να μετασχηματιστεί σε μορφή .dex η οποία είναι βελτιστοποιημένη με τέτοιο τρόπο ώστε να αφήνει το ελάχιστο δυνατό αποτύπωμα στη μνήμη. Αυτό συμβαίνει με το dx εργαλείο που συμπεριλαμβάνεται στο Android SDK και πραγματοποιείται είτε χειροκίνητα είτε αυτόματα στο build time. Όταν γίνεται η παραγωγή του .dex αρχείου το εργαλείο dx ενσωματώνει πολλαπλά αρχεία κλάσεων σε ένα μόνο .dex αρχείο όπου όλα τα αλφαριθμητικά που αναφέρονται περισσότερες από μια φορές ενσωματώνονται μόνο μια και γίνεται αναφορά σε αυτά σε global επίπεδο για την εξοικονόμηση χώρου. Επιπλέον, οι βιβλιοθήκες της Dalvik VM δεν αποτελούν προσαρμοσμένα πρότυπα. Οι βιβλιοθήκες πυρήνα που χρησιμοποιούνται από το περιβάλλον εκτέλεσης του Android είναι είτε Java ME συμβατές είτε Java SE συμβατές. Το μεγαλύτερο τμήμα των πρωτοκόλλων Java είναι βιβλιοθήκες Java SE. Από την έκδοση Micro Edition (Java ME) υλοποιείται μόνο το Java binding για OpenGL ES. Οι υπόλοιπες βιβλιοθήκες που περιλαμβάνονται στην πλατφόρμα Android προέρχονται από έργα ανοικτού κώδικα όπως Apache Commons, JUnit και Google.

Κάθε εφαρμογή Android τρέχει στη δική της διεργασία με το δικό της στιγμιότυπο της Dalvik VM. Η Dalvik VM στηρίζεται στον πυρήνα Linux για επιπλέον λειτουργικότητα όπως τη διαχείριση νημάτων και τη διαχείριση μνήμης σε χαμηλό επίπεδο. Επιπλέον, όταν μια εφαρμογή εγκαθίσταται στη συσκευή γίνονται κάποιες βελτιστοποιήσεις όπως byte order swapping static linking, απομάκρυνση των κενών μεθόδων κτλ. Η Dalvik VM έχει σχεδιαστεί έτσι ώστε πολλά στιγμιότυπα να τρέχουν στην ίδια συσκευή ταυτόχρονα και αποδοτικά [14]. Στην εικονική αυτή μηχανή του

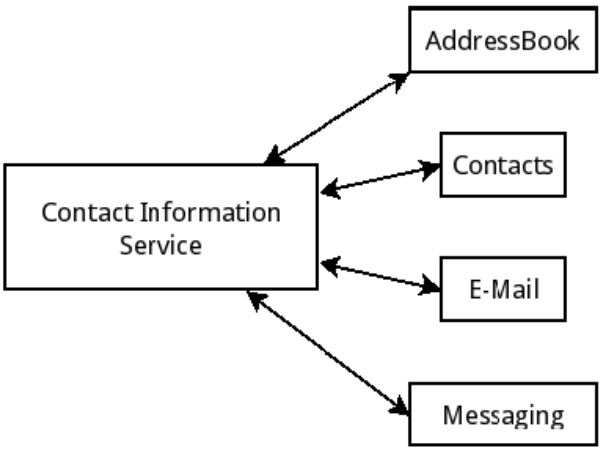
Android περιλαμβάνεται επίσης και ένας Garbage Collector – GC ο οποίος αρχικοποιείται για κάθε διεργασία. Διαχειρίζεται τη δέσμευση και αποδέσμευση του σωρού για κάθε εφαρμογή και σέβεται τα δεδομένα που δεν είναι διαμοιράσιμα ανάμεσα στις διάφορες εφαρμογές που εκτελούνται. Επιπλέον, η Dalvik VM δεν έχει Just-In-Time (JIT) μεταγλωττιστή ο οποίος βελτιώνει την απόδοση των μεταφρασμένων εφαρμογών και του bytecode. Ως αποτέλεσμα, εξοικονομείται ενέργεια στο Android OS. Ακόμα ένας λόγος για τον οποίο ο JIT δεν είναι απαραίτητος στο λογισμικό αυτό είναι ότι οι περισσότερες βιβλιοθήκες που χρησιμοποιούνται συχνά από τον επεξεργαστή είναι χαμηλού επιπέδου και γραμμένες σε C ή C++ κρατώντας την επιβάρυνση ως προς την απόδοση σε χαμηλά επίπεδα.

Προχωρώντας ακόμα πιο πάνω βρίσκεται το επίπεδο Application Framework. Πρόκειται για ένα σύνολο βιβλιοθηκών Java που παρέχει κατασκευαστικά κομμάτια κώδικα για εύκολη και γρήγορη ανάπτυξη εφαρμογών για την πλατφόρμα Android. Όλες οι εφαρμογές του Android είναι γραμμένες στη γλώσσα Java ενώ το ίδιο το λειτουργικό σύστημα δίνεται στην αγορά με ένα σύνολο θεμελιωδών εφαρμογών πυρήνα όπως email client, SMS πρόγραμμα, ημερολόγιο, χάρτες, φυλλομετρητή, επαφές κ.α. Το Android OS δίνει τη δυνατότητα στους χρήστες να αναπτύξουν διαφόρων πεδίων εφαρμογές με «ανοικτή ανάπτυξη». Δηλαδή παρέχεται στους χρήστες πλήρη πρόσβαση στα πρωτόκολλα των frameworks τα οποία χρησιμοποιούν όλες οι εφαρμογές, ακόμα και αυτές με τις θεμελιώδεις λειτουργίες που αναφέρθηκαν πιο πάνω. Η αρχιτεκτονική των εφαρμογών έχει σχεδιαστεί έτσι ώστε να απλοποιεί την επαναχρησιμοποίηση συστατικών. Έτσι, κάθε εφαρμογή μπορεί να δημοσιεύσει τις ικανότητές της και κάθε άλλη εφαρμογή μπορεί να αξιοποιήσει αυτές τις δυνατότητες. Ο ίδιος μηχανισμός επιτρέπει την αντικατάσταση συστατικών από άλλα που ταιριάζουν καλύτερα ανά πάσα στιγμή. Οι εφαρμογές περιλαμβάνουν όψεις που μπορούν να χρησιμοποιηθούν για την κατασκευή

εφαρμογών, λιστών, περιοχών κειμένου, κουμπιών ή ακόμα και ενσωματωμένων φυλλομετρητών.

Ο διαχειριστής των δραστηριοτήτων (activity manager) διαχειρίζεται τον κύκλο ζωής όλων των εφαρμογών και παρέχει μια κοινή διεπαφή πλοήγησης για τις εφαρμογές που τρέχουν σε διαφορετικές διεργασίες στη συσκευή. Από την άλλη ο package manager παρακολουθεί τις εγκατεστημένες εφαρμογές σε μια συσκευή μαζί με τις εξαρτήσεις τους και διαχειρίζεται την απομάκρυνση αλλά και την ανανέωσή τους όποτε είναι αναγκαίο ή όταν απαιτηθεί από το χρήστη. Ο διαχειριστής παραθύρων (window manager) είναι μια αφαίρεση της Java πάνω από την οθόνη και τα επιφανειακά συστήματα που παρέχει διαχείριση παραθύρων για τις εφαρμογές. Διαχειρίζεται τα διαφορετικά παράθυρα των διαφορετικών εφαρμογών μεταφέροντας, μετατρέποντας και εναλλάσσοντάς τα επιλέγοντας κάθε φορά την κατάλληλη σειρά και το παράθυρο στο οποίο θα επικεντρωθεί.

Ο παροχέας περιεχομένου (content provider) είναι επίσης ένα επίπεδο αφαίρεσης τοποθετημένο πάνω από τον OpenBilder που παρέχει στις εφαρμογές ένα μηχανισμό διαμοίρασης των δεδομένων και των λειτουργιών τους με άλλες εφαρμογές που τρέχουν στην ίδια συσκευή. Την ίδια στιγμή οι εφαρμογές μπορούν οι ίδιες να χρησιμοποιούν δεδομένα και λειτουργικότητες από άλλες εγκατεστημένες εφαρμογές. Αυτή είναι μια καλή πολιτική επαναχρησιμοποίησης συστατικών που εξοικονομεί χώρο, μνήμη και επεξεργαστική ισχύ. Ένα παράδειγμα του content provider είναι η υπηρεσία διαχείρισης επαφών (Contacts Manager) που παρέχεται από την εφαρμογή AddressBook του Android. Αυτή η υπηρεσία παρέχει λειτουργικότητες αποθήκευσης και ανάγνωσης πληροφοριών από τις επαφές σε μια συσκευή κινητού τηλεφώνου. Στη συνέχεια η εφαρμογή Phone και η εφαρμογή Messaging μπορούν να ζητήσουν και να χρησιμοποιήσουν την ίδια υπηρεσία για πρόσβαση σε πληροφορίες που αφορούν τις επαφές χωρίς να χρειαστεί ξανά η υλοποίηση της βάσης δεδομένων των επαφών όπως φαίνεται και στην εικόνα 4.7 [15][16].



Εικόνα 4. 7 Η υπηρεσία παροχής πληροφοριών σχετικά με τις επαφές

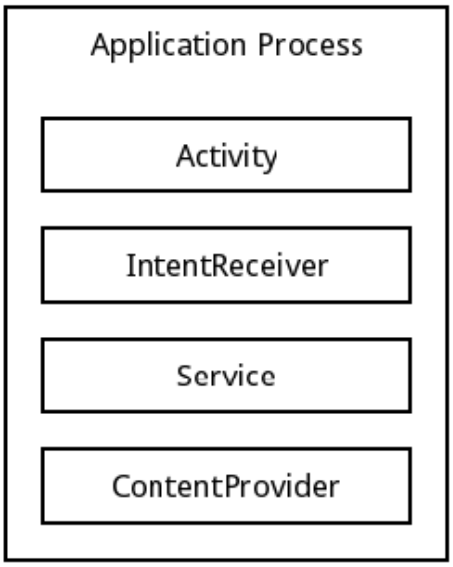
Ο διαχειριστής των πόρων (resource manager) παρέχει ένα συστατικό που βοηθά τις εφαρμογές να διαχειρίζονται τους πόρους τους όπως εικόνες, αρχεία δεδομένων, γραφικά αλλά και άλλα τμήματα της εφαρμογής. Το σύστημα εμφάνισης (view system) είναι στην ουσία ένα συστατικό γραφικής διασύνδεσης χρήστη που παρέχει τη βάση για τα γραφικά στοιχεία του GUI και τα widgets όπως οι λίστες, τα κουμπιά και τα πλαίσια κειμένου. Οι κύριες όψεις περιεχομένου (content views) που είναι οι αρχικές όψεις κάθε εφαρμογής, προσθέτονται στο παράθυρο της εφαρμογής (τη δραστηριότητα - activity) και μπορούν να περιέχουν όψεις-παιδιά. Με αυτό τον τρόπο δημιουργείται μια ιεραρχία όψεων (γραφικών στοιχείων της διασύνδεσης χρήστη) και δίνει το πλεονέκτημα της δυνατότητας πρόσθεσης animations και γραφικών στοιχείων στα στοιχεία του GUI.

Επίσης, στο επίπεδο αυτό περιλαμβάνεται και ο Location Manager και η XMPP υπηρεσία. Ο location manager επιτρέπει στις εφαρμογές να ανακτήσουν πληροφορίες που έχουν να κάνουν με γεωγραφική τοποθεσία ή να ενεργοποιήσουν γεγονότα που έχουν να κάνουν με εφαρμογές όταν η συσκευή εισέλθει σε συγκεκριμένη τοποθεσία. Η XMPP υπηρεσία (Extensible Messaging and Presence Protocol) παρέχει ένα εύκολα επεκτάσιμο πρωτόκολλο για άμεση συνομιλία (instance messaging). Επεκτείνεται με VoIP και αποτελεί τη βάση του GoogleTalk της Google. Χρησιμοποιείται στο Android OS για την ανταλλαγή ομότιμων μηνυμάτων (peer-to-

peer) μεταξύ διαφορετικών συσκευών. Αυτές οι δύο υπηρεσίες χρησιμοποιούνται για την κατασκευή καινοτόμων εφαρμογών εντοπισμού τοποθεσίας και εφαρμογών που έχουν να κάνουν με το περιεχόμενο κάνοντας τη φορητότητα των συσκευών ευκολότερη.

Τέλος, ο notification manager παρέχει μια διασύνδεση για την εμφάνιση ειδοποιήσεων από το σύστημα προς το χρήστη της συσκευής. Παραδείγματα ειδοποιήσεων αποτελούν τα επίπεδα ενέργειας της μπαταρίας, ο ερχομός ενός νέου email κτλ. Ο telephony manager είναι υπεύθυνος για να δίνει τη δυνατότητα στις εφαρμογές να πραγματοποιούν τηλεφωνικές κλήσεις.

Το τελευταίο επίπεδο της αρχιτεκτονικής του Android OS είναι το επίπεδο των εφαρμογών (applications). Οι εφαρμογές του Android φτιάχνονται χρησιμοποιώντας τα frameworks και τις βιβλιοθήκες που παρέχονται και είναι δυνατό να αποτελούνται από αρκετά συστατικά όπως φαίνεται και στην εικόνα 4.8.



Εικόνα 4. 8 Συστατικά μιας Android εφαρμογής

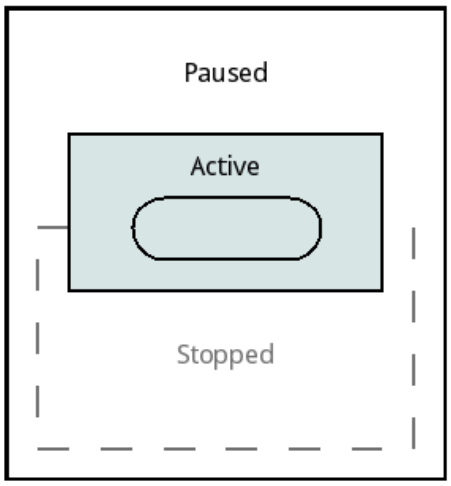
Μετά τη συγγραφή τους, οι εφαρμογές πακετάρονται χρησιμοποιώντας το aapt εργαλείο σε .apk αρχεία που διανέμονται και εγκαθίστανται σε Android συσκευές. Όταν μια εφαρμογή τρέχει σε μια αντίστοιχη συσκευή, αποτελεί μια μοναδική Linux διεργασία που έχει το δικό της στιγμιότυπο της Dalvik VM και ανατίθεται σε αυτήν μια

μοναδική ταυτότητα. Έτσι παίρνει το πλεονέκτημα της καλύτερης ασφάλειας, της προστατευόμενης μνήμης και του multi-tasking [17]. Έχει ήδη αναφερθεί ότι η κάθε εφαρμογή μπορεί να εκδίδει τη δική της λειτουργικότητα και δεδομένα και να χρησιμοποιεί τη λειτουργικότητα και τα δεδομένα άλλων εφαρμογών μέσω των content providers.

Κύρια συστατικά αυτού του επιπέδου είναι: Activity (δραστηριότητα), Intent Receiver, Service (υπηρεσία) και Content Provider (παροχέας περιεχομένου). Μια δραστηριότητα σε ένα βασικό επίπεδο είναι μια γραφική διασύνδεση χρήστη ενός παραθύρου της εφαρμογής. Συσχετίζεται συνήθως με την οθόνη. Ο Intent Receiver είναι αυτός που ξυπνά μια διεργασία και εκτελεί μια προκαθορισμένη ενέργεια μέσω ενός εξωτερικού γεγονότος. Αυτό καθορίζεται μέσω XML αρχεία περιγραφής που περιλαμβάνονται στους πόρους της κάθε εφαρμογής. Ο δημιουργός της εφαρμογής μπορεί να γράψει κώδικα για την εφαρμογή και να τον καταχωρήσει μέσω XML για να καλείται όταν ένα συγκεκριμένο εξωτερικό γεγονός συμβαίνει. Για παράδειγμα όταν η συνδεσιμότητα επιτυγχάνεται, όταν προκύψει εισερχόμενη κλήση ή όταν η μπαταρία τελειώνει.

Μια υπηρεσία είναι μια διεργασία που τρέχει στο παρασκήνιο και που δε βρίσκεται σε άμεση επαφή με το χρήστη. Αυτό είναι χρήσιμο αφού επιτρέπει στο χρήστη την πλοήγηση σε άλλες εφαρμογές όταν επεξεργάζεται η τρέχουσα διεργασία. Ως παράδειγμα μπορεί να αναφερθεί το Music player όπου η μουσική πρέπει να συνεχίσει να παίζει ακόμα και αν ο χρήστης αλλάξει εφαρμογή. Με τον τρόπο αυτό επιτυγχάνεται και λιγότερη προσπάθεια στην υλοποίηση της ίδιας λειτουργικότητας. Έτσι, συνεχίζοντας στο ίδιο παράδειγμα, ο χρήστης μπορεί να προσπαθήσει να δημιουργήσει ένα νέο περιβάλλον διασύνδεσης με την εφαρμογή του Music player χωρίς να ανησυχεί ότι θα υλοποιήσει την ίδια τεχνολογία αναπαραγωγής ήχου. Τέλος, ο content provider είναι το συστατικό στην κάθε εφαρμογή που διαμοιράζεται τα δεδομένα της εφαρμογής και τη λειτουργικότητά της με άλλες εφαρμογές που τρέχουν στο σύστημα.

Όσον αφορά τον κύκλο ζωής της κάθε εφαρμογής, όπως και στο iOS έτσι και στο Android OS ακολουθείται μια διαδικασία. Ο χρήστης θέλει να κινείται από εφαρμογή σε εφαρμογή και να επιστρέφει πίσω σε κάποιες άλλες με απολύτως φυσικό, γρήγορο και εύκολο τρόπο. Η πλατφόρμα Android παρέχει ένα τρόπο αφαίρεσης για να χειρίζεται multi-tasking και πολλαπλές ανοικτές δραστηριότητες ταυτόχρονα σε συσκευές με περιορισμένο υλικό. Κάθε δραστηριότητα μπορεί να βρίσκεται σε κάθε μια από τις ακόλουθες τρεις καταστάσεις όπως φαίνεται και στην εικόνα 4.9: α) Active – όπου η δραστηριότητα είναι ορατή και εκτελείται, β) Paused – όπου η δραστηριότητα είναι ορατή αλλά δεν εκτελείται στο προσκήνιο (μια άλλη δραστηριότητα βρίσκεται από πάνω της) και γ) Stopped – όπου η δραστηριότητα επικαλύπτεται από άλλη δραστηριότητα και δεν είναι πλέον ορατή.



Εικόνα 4. 9 Οι διάφορες καταστάσεις στις οποίες είναι δυνατό να περιέλθει μια δραστηριότητα

Οι paused και stopped δραστηριότητες μπορούν να αφαιρεθούν από τη μνήμη μέσω του συστήματος ανά πάσα στιγμή. Για να είναι δυνατή η αποκατάσταση αυτών των δραστηριοτήτων στις ίδιες καταστάσεις που ήταν πριν να τερματιστούν, πρέπει το σύστημα να έχει αποθηκεύσει τις καταστάσεις τους. Το σύστημα αποθηκεύει την κατάσταση της τρέχουσας δραστηριότητας όταν μια νέα δραστηριότητα αρχίζει. Ένα σενάριο εναλλαγής δραστηριοτήτων μπορεί να είναι ο έλεγχος των email ενός χρήστη

μέσω της εφαρμογής Email. Την επόμενη χρονική στιγμή έστω ότι ο χρήστης επιλέγει να διαβάσει ένα ηλεκτρονικό μήνυμα και επιλέγει ένα σύνδεσμο που περιέχεται σε αυτό. Αυτή η ενέργεια του χρήστη προκαλεί το άνοιγμα του φυλλομετρητή και της WebBrowser εφαρμογής ως μια νέα δραστηριότητα κ.ο.κ. Η μόνη εφαρμογή που τρέχει συνέχεια είναι η εφαρμογή Home και δεν μπορεί να τερματιστεί αφού χρησιμοποιείται για σκοπούς αλληλεπίδρασης και συνεννόησης με το σύστημα.

Όταν δεν υπάρχει πλέον διαθέσιμη μνήμη, το σύστημα επιλέγει να τερματίσει κάποια διεργασία. Παρόλο που ο τερματισμός μιας τυχαίας διεργασίας μπορεί να φαίνεται ως σωστό, το σύστημα πρέπει να επιλέξει προσεκτικά ποια διεργασία να τερματίσει. Συνήθως επιλέγεται για τερματισμό η διεργασία που είναι η πιο απομακρυσμένη από την τρέχουσα αφού είναι και η λιγότερο πιθανή να χρησιμοποιηθεί αρκετά σύντομα. Όταν το σύστημα αποθηκεύει καταστάσεις δε φαίνεται να αποθηκεύει το περιεχόμενο της δραστηριότητας. Αποθηκεύει μόνο τις τρέχουσες ιδιότητες της δραστηριότητας. Το περιεχόμενο φορτώνεται ξανά όταν η εφαρμογή ή η δραστηριότητα αποκατασταθεί και η αποθηκευμένη κατάσταση εφαρμόζεται στο νέο στιγμιότυπο.

Κεφάλαιο 5

Διαχείριση συστήματος στο Apple iOS για iPhone

Στο κεφάλαιο αυτό γίνεται μια λεπτομερής συζήτηση σε θέματα διαχείρισης συστήματος στο iOS για τα κινητά τηλέφωνα iPhone. Γίνεται αναφορά σε θέματα διαχείρισης μνήμης, στην πορεία ανάπτυξης των εφαρμογών του iOS καθώς και στα κυριότερα συστατικά της κάθε εφαρμογής. Εξηγούνται επίσης διαδικασίες μετάβασης καταστάσεων των εφαρμογών, χειρισμού διακοπών (interrupts) αλλά και θέματα γραφικών, διασύνδεσης και ήχου.

5.1 Διαχείριση Μνήμης

Επειδή το iPhone πρέπει πάντα να έχει τη δυνατότητα να λαμβάνει εισερχόμενες κλήσεις, η εφαρμογή Phone έχει πάντοτε μεγαλύτερη προτεραιότητα από όλες τις άλλες εφαρμογές. Για να τρέχει η εφαρμογή αυτή στο παρασκήνιο το σύστημα απαιτεί να υπάρχουν αρκετές διαθέσιμες πηγές ελεύθερες. Για το λόγο αυτό όλες οι εφαρμογές που γράφονται για το iPhone πρέπει να γράφονται με σεβασμό στους κανόνες χαμηλής μνήμης του λειτουργικού συστήματος. Όταν το λειτουργικό σύστημα ανιχνεύσει ότι η διαθέσιμη μνήμη μειώνεται, προειδοποιεί όλες τις εφαρμογές που εκτελούνται ήδη, ακόμα και αυτές που τρέχουν στο παρασκήνιο (Phone, Safari, Mail κτλ). Όταν μια εφαρμογή λάβει προειδοποίηση για χαμηλή μνήμη πρέπει είτε να ελευθερώσει όλους τους πόρους που μπορεί αλλιώς το iOS θα προκαλέσει άμεσα τον τερματισμό της [18].

Όταν ο χρήστης βρίσκεται σε μια εφαρμογή είναι συχνό φαινόμενο να εναλλάσσεται ανάμεσα σε λίστες, παράθυρα ή οθόνες. Η αναπαράσταση αυτών των παραθύρων και των οθόνων απαιτεί πολύ περισσότερο χρόνο επεξεργασίας απ' ότι η απλή κλήση τους όταν βρίσκονται ήδη αποθηκευμένες στην RAM. Έτσι οι περισσότερες εφαρμογές κρατάνε τις μη ενεργές οθόνες τους στη RAM. Όταν προκύψει μια προειδοποίηση για χαμηλή μνήμη παρόλο που η εφαρμογή πρέπει να ελευθερώσει άμεσα πόρους από τα προσχεδιασμένα δεδομένα της οθόνης το μόνο που απαιτείται να επαναληφθεί για την επανεμφάνισή τους είναι μια επανάκλησή τους όταν ο έλεγχος επιστρέψει σε αυτή την εφαρμογή. Ως παράδειγμα μπορεί να θεωρηθεί η εφαρμογή Calendar. Όταν ο χρήστης εναλλάσσεται από την εμφάνιση όψεις ημερών σε όψεις μηνών η εφαρμογή πρέπει να χειριστεί 3 διαφορετικές όψεις. Την πρώτη φορά που εμφανίζεται η κάθε όψη απαιτείται λίγος χρόνος μέχρι να σχεδιαστεί στην οθόνη. Τις επόμενες φορές που ο χρήστης επιστρέφει σε όψη που είχε εμφανιστεί προηγουμένως αυτή εμφανίζεται σχεδόν άμεσα αφού βρίσκεται ήδη αποθηκευμένη στη RAM. Όσα περισσότερα δεδομένα αποθηκεύονται στη RAM ανά πάσα στιγμή, τόσο πιο γρήγορη είναι η πλοήγηση του χρήστη στις εφαρμογές.

Ως αποτέλεσμα, η αύξηση της μνήμης RAM του iPhone από 128MB που ήταν αρχικά σε 256MB στο iPhone 3GS είναι πολύ σημαντικό πράγμα. Κάθε εφαρμογή που έχει γραφτεί για τα τηλέφωνα με 128MB RAM πλέον λειτουργούν με πολύ μεγαλύτερη ταχύτητα. Υπάρχει βέβαια πάντα το ενδεχόμενο η μνήμη να καταρρεύσει, ιδίως όταν μια εφαρμογή δεν είναι γραμμένη με τον ορθό τρόπο και δεν ακολουθεί τις ορθές πρακτικές διαχείρισης μνήμης. Η επιτυχία στηρίζεται και στη συνεχή προσπάθεια της κάθε ομάδας ανάπτυξης εφαρμογών προς τη βελτιστοποίηση τους. Ακόμα και αν κάποιοι επιχειρήσουν να ελευθερώσουν στην αγορά εφαρμογές που καταναλώνουν συνεχώς πολλή μνήμη, η Apple δεν πρόκειται να επιτρέψει την κυκλοφορία τους στο AppStore.

Οι χρήστες που αναπτύσσουν συνήθως εφαρμογές στη γλώσσα Java δεν έχουν συνεχώς την προσοχή τους στη διαχείριση μνήμης αφού για το σκοπό αυτό υπάρχει

ο Garbage Collector. Έτσι, ο χρήστης μπορεί να δημιουργεί ανενόχλητος μεταβλητές τις οποίες διαχειρίζεται ο GC όταν δε χρησιμοποιούνται πλέον. Στο iOS δεν υπάρχει GC και έτσι ο χρήστης πρέπει να φροντίζει για τη διαγραφή των αχρησιμοποίητων μεταβλητών. Σε περίπτωση που αυτό δε συμβεί, κάποια στιγμή το σύστημα θα καταρρεύσει λόγω έλλειψης μνήμης. Η λειτουργία αυτή όμως δε χρειάζεται να γίνει εντελώς χειροκίνητα. Η κλάση NSObject περιέχει μεθόδους που ελέγχουν πόσα αντικείμενα χρησιμοποιούνται ανά πάσα στιγμή χρησιμοποιώντας το αντικείμενο στην ερώτηση “retain count”.

Ο κανόνας της διαχείρισης μνήμης είναι να εξασφαλίζει ο χρήστης ότι μέχρι τη στιγμή που το πρόγραμμα θα τερματίσει την εκτέλεσή του, ο αριθμός των “ownership methods” που έχουν κληθεί σε ένα αντικείμενο πρέπει να ισούται με τον αριθμό των “loss-of-ownership” μεθόδων [19].

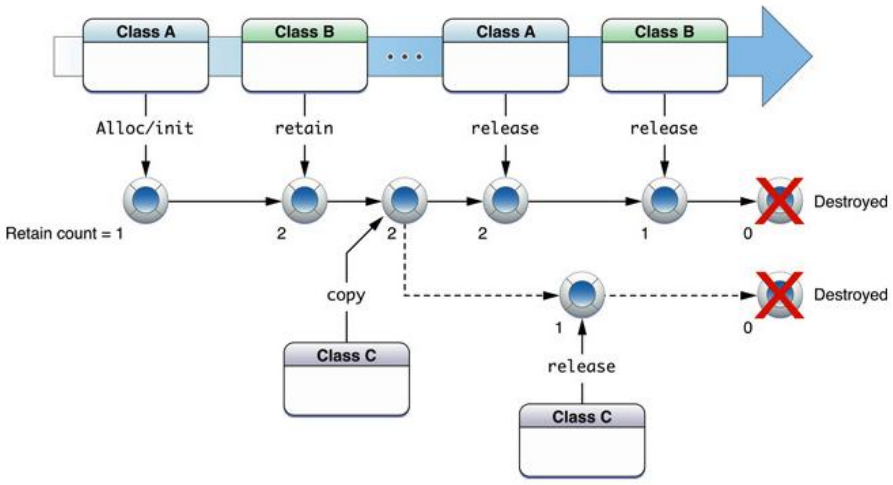
Η ανάπτυξη εφαρμογών για το iOS γίνεται όπως έχει αναφερθεί και στο προηγούμενο κεφάλαιο μέσω του iPhone SDK. Το SDK είναι πολύ καλά σχεδιασμένο και αρκετά γρήγορο. Η δυσκολία ανάπτυξης εφαρμογών αντιμετωπίζεται συνήθως στη γλώσσα Objective C και στο Cocoa σε θέματα που αφορούν κυρίως τη διαχείριση μνήμης. Η Objective C χρησιμοποιεί “reference counting” ως την κύρια τεχνική διαχείρισης μνήμης. Κάθε αντικείμενο κρατά ένα εσωτερικό μετρητή σχετικά με το πόσες φορές έχει χρειαστεί. Το σύστημα εξασφαλίζει ότι τα αντικείμενα που χρειάζονται δεν διαγράφονται ενώ αντίθετα, αντικείμενα που δεν είναι πλέον χρήσιμα, απομακρύνονται [20][21]. Η διαφορά με τον Garbage Collector είναι ότι στον αυτόματο GC της Java ένα ξεχωριστό τμήμα κώδικα τρέχει περιοδικά στο παρασκήνιο για να εντοπίσει τα αντικείμενα τα οποία αναφέρονται και ποια δε χρειάζονται πλέον. Στη συνέχεια διαγράφει όλα τα αχρησιμοποίητα αντικείμενα αυτόματα χωρίς να απαιτούνται ειδικοί χειρισμοί από τον προγραμματιστή. Στη μέθοδο “reference counting” ο προγραμματιστής έχει την υπευθυνότητα να δηλώνει το πότε χρειάζεται ένα αντικείμενο και πότε τελειώνει με αυτό. Η διαγραφή του

αντικειμένου συμβαίνει άμεσα όταν το αντικείμενο πάψει να χρησιμοποιείται, δηλαδή όταν ο μετρητής αναφοράς του πάει στο 0.

Πολύ σημαντικό στη διαχείριση μνήμης είναι και η ιδιοκτησία των αντικειμένων. Στην Objective C ο ιδιοκτήτης ενός αντικειμένου είναι κάποιος άνθρωπος ή ένα τμήμα κώδικα που έχει το αποκλειστικό δικαίωμα να πει ανά πάσα στιγμή «Χρειάζομαι το αντικείμενο, μην το διαγράψεις». Αυτό το άτομο ή το τμήμα κώδικα είναι αυτός που δημιούργησε αρχικά το αντικείμενο. Σε άλλη περίπτωση, ιδιοκτήτης ενός αντικειμένου μπορεί να είναι χρήστης ή τμήμα κώδικα ο οποίος δεν το δημιούργησε μεν, έχει όμως δηλώσει ότι το χρειάζεται. Με τον τρόπο αυτό, ένα αντικείμενο είναι δυνατό να έχει περισσότερους από ένα ιδιοκτήτες. Ο αριθμός των ιδιοκτητών που έχει ένα αντικείμενο είναι ο reference counter του.

Οι ιδιοκτήτες ενός αντικειμένου έχουν την ευθύνη να ενημερώνουν το αντικείμενο όταν τελειώσουν μαζί του. Από τη στιγμή που πλέον δεν το χρειάζονται, παύουν να συγκαταλέγονται ανάμεσα στους ιδιοκτήτες του ενώ αν δεν έχει παραμείνει κανένας ιδιοκτήτης για το αντικείμενο, αυτό διαγράφεται. Υπάρχει βέβαια και η δυνατότητα να χρησιμοποιήσει κανείς το αντικείμενο χωρίς να είναι ιδιοκτήτης του όμως μόνο για προσωρινή χρήση. Σε περίπτωση που κανείς χρειάζεται ένα αντικείμενο μακροχρόνια, πρέπει να αποκτήσει την ιδιοκτησία του.

Τα μηνύματα που μπορούν να σταλούν στα αντικείμενα σχετικά με τη διαχείριση μνήμης είναι τα ακόλουθα ενώ αναπαριστούνται και σχηματικά στην εικόνα 5.1:



Εικόνα 5. 1 Διαχείριση μνήμης μέσω δέσμευσης και αποδέσμευσης αντικειμένων

alloc: Αυτό δεσμεύει το στιγμιότυπο ενός αντικειμένου. Επίσης, θέτει τον reference counter τον αριθμό 1. Αυτός που θα καλέσει την εντολή alloc είναι πλέον και ο ιδιοκτήτης του αντικειμένου. Απαιτείται η αποδέσμευση του αντικειμένου όταν ο ιδιοκτήτης παύει να το χρειάζεται.

new: Εφαρμόζονται οι ίδιοι κανόνες με το μήνυμα alloc

retain: Το μήνυμα αυτό καλείται όταν έχει ήδη δηλωθεί ένα αντικείμενο και ο χρήστης επιθυμεί να δηλώσει ότι το χρειάζεται επίσης. Οπότε δίνεται το μήνυμα ότι όταν τελειώσει ο ιδιοκτήτης με το αντικείμενο, επιθυμεί να το χρησιμοποιήσει ο ίδιος οπότε αυτό δεν θα διαγραφεί. Στην περίπτωση αυτή ο μετρητής του αντικειμένου αυξάνεται κατά 1 και πλέον ο χρήστης θα προστεθεί στους ιδιοκτήτες του. Απαιτείται και σε αυτή την περίπτωση απελευθέρωση του αντικειμένου όταν πλέον δεν το χρειάζεται.

release: Το μήνυμα αυτό καλείται όταν ο χρήστης που είναι ένας από τους ιδιοκτήτες του αντικειμένου παύει να το χρειάζεται. Ο χρήστης παύει την ίδια στιγμή να είναι ιδιοκτήτης του αντικειμένου και ο μετρητής του αντικειμένου μειώνεται κατά 1. Αν ο μετρητής είναι πλέον 0 τότε το αντικείμενο καταστρέφεται αυτόματα και ελευθερώνεται η μνήμη την οποία είχε δεσμεύσει. Η κλήση αυτού το μηνύματος είναι υποχρεωτική για τους ιδιοκτήτες.

autorelease: Το μήνυμα αυτό δηλώνει ότι κάποιος χρήστης χρειάζεται το αντικείμενο προσωρινά και για το λόγο αυτό δεν απαιτείται να γίνει ιδιοκτήτης του.

copy: Με τον τρόπο αυτό δημιουργείται ένα αντίγραφο του αντικειμένου ανάλογα με το πώς το μήνυμα αυτό έχει υλοποιηθεί στο κάθε αντικείμενο. Το αντίγραφο του αντικειμένου που δημιουργείται ανήκει πλέον στο χρήστη που κάλεσε το μήνυμα και γίνεται πλέον ιδιοκτήτης του αντικειμένου.

Είναι κατανοητό ότι όταν ασχολείται κανείς με δείκτες και αντικείμενα στην Objective C είναι σημαντικό να θυμάται την αποστολή των κατάλληλων μηνυμάτων για κάθε λειτουργία. Αν κανείς είναι ιδιοκτήτης του αντικειμένου πρέπει να δηλώσει ότι το χρειάζεται δεσμεύοντάς το (allocate, new, retain, copy) και να το ελευθερώσει

(release) όταν δεν το χρειάζεται πλέον. Αν κανείς δεν είναι ιδιοκτήτης του αντικειμένου τότε δεν θα το ελευθερώσει. Η διατήρηση του αντικειμένου από κάποιον ιδιοκτήτη πρέπει να συμβαίνει μόνο αν θα το χρειαστεί για μεγάλο χρονικό διάστημα.

Αν κανείς χρειαστεί ταυτόχρονα πολλά προσωρινά αντικείμενα (autoreleased) τότε είναι καλό να δημιουργήσει short-term autorelease pools για να μην υπάρξει πρόβλημα με τη μνήμη. Αν παρόλα αυτά εξαντληθούν τα όρια της μνήμης στο iPhone η εφαρμογή θα τερματιστεί αμέσως αφού εναπόκειται στο χρήστη η αποδοτική δέσμευση και αποδέσμευση μνήμης στη διάρκεια του χρόνου.

5.2 Πορεία ανάπτυξης εφαρμογών για το iOS της Apple

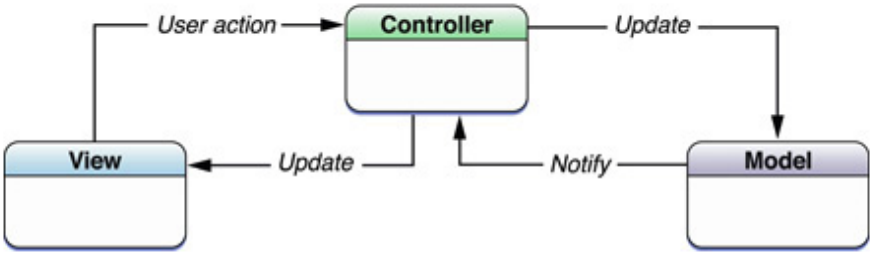
Κάθε εφαρμογή για το iOS φτιάχνεται χρησιμοποιώντας το UIKit framework και έχει την ίδια αρχιτεκτονική πυρήνα. Το UIKit παρέχει τα απαραίτητα αντικείμενα για την εκτέλεση της κάθε εφαρμογής, για το συντονισμό και το χειρισμό της εισόδου του χρήστη και για την εμφάνιση του περιεχομένου στην οθόνη. Ο τρόπος με τον οποίο διαφοροποιείται η κάθε εφαρμογή από την άλλη είναι ο τρόπος με τον οποίο χρησιμοποιούν τα αντικείμενα αυτά αλλά και η συμπεριφορά που επιλέγει ο κάθε χρήστης να δώσει. Υπάρχουν πολλές αλληλεπιδράσεις που συμβαίνουν μεταξύ του πυρήνα του συστήματος και μιας εν εκτέλεση εφαρμογής. Πολλές από αυτές τις αλληλεπιδράσεις χειρίζονται αυτόματα από την υποδομή του UIKit. Υπάρχουν βέβαια και στιγμές όπου η εφαρμογή πρέπει να είναι έτοιμη να αντιληφθεί γεγονότα που θα προκύψουν από το σύστημα. Για παράδειγμα, όταν ο χρήστης τερματίσει μια εφαρμογή πρέπει πρώτα αυτή να ειδοποιηθεί έτσι ώστε να μπορέσει να αποθηκεύσει σχετικά δεδομένα πριν μεταφερθεί στο παρασκήνιο. Για τις καταστάσεις αυτές, το UIKit παρέχει κομμάτια κώδικα τα οποία ο χρήστης μπορεί να εκμεταλλευτεί και να χρησιμοποιήσει για να παρέχει την επιθυμητή συμπεριφορά.

Η σχεδίαση του UIKit framework συνδυάζει πολλά από τα σχεδιαστικά πρότυπα που συναντώνται στις Cocoa εφαρμογές του Mac OS X. Η κατανόηση αυτών των

σχεδιαστικών προτύπων είναι κρίσιμη όσον αφορά τη δημιουργία iOS εφαρμογών. Στη συνέχεια αναφέρονται τα σχεδιαστικά πρότυπα Model View Controller, Block Objects, Delegation, Target-action, Managed memory model και Threads and concurrent programming [22].

Το σχεδιαστικό πρότυπο Model View Controller (MVC) είναι ένας τρόπος διαχωρισμού του κώδικα της εφαρμογής σε ανεξάρτητες, λειτουργικές περιοχές. Το τμήμα του μοντέλου (model) καθορίζει τη μηχανή δεδομένων που βρίσκεται στα θεμέλια της εφαρμογής και είναι υπεύθυνο για τη διατήρηση της ακεραιότητας αυτών των δεδομένων. Το τμήμα της όψης (view) καθορίζει τη γραφική διασύνδεση χρήστη με την εφαρμογή και δεν έχει καθόλου γνώση σχετικά με την καταγωγή των δεδομένων που εμφανίζονται στη διασύνδεση. Τέλος, το τμήμα του ελεγκτή (controller) δρα ως γέφυρα μεταξύ του μοντέλου και της όψης και συντονίζει το πέρασμα των δεδομένων μεταξύ τους. Το MVC αναθέτει στα αντικείμενα της κάθε εφαρμογής έναν από τους τρεις ρόλους που υπηρετεί. Το πρότυπο καθορίζει όχι μόνο τους ρόλους τους οποίους παίζει κάθε αντικείμενο στην εφαρμογή αλλά και τον τρόπο με τον οποίο τα αντικείμενα επικοινωνούν μεταξύ τους. Κάθε αντικείμενο είναι διαχωρισμένο από τα άλλα με αφηρημένα όρια και επικοινωνεί με αντικείμενα άλλων τύπων μέσω αυτών των ορίων. Η συλλογή των αντικειμένων ενός συγκεκριμένου MVC τύπου σε μια εφαρμογή αναφέρεται συνήθως ως επίπεδο π.χ. επίπεδο μοντέλου, επίπεδο όψης, επίπεδο ελεγκτή.

Το MVC είναι κρίσιμο όσον αφορά την καλή σχεδίαση για μια Cocoa εφαρμογή. Τα πλεονεκτήματα της χρήσης αυτού του προτύπου είναι αρκετά. Πολλά αντικείμενα σε αυτές τις εφαρμογές τείνουν να είναι πιο εύκολα επαναχρησιμοποιήσιμα και οι διεπαφές τους τείνουν να είναι καλύτερα ορισμένες. Οι εφαρμογές που υιοθετούν MVC σχεδίαση είναι πιο εύκολες στην επέκταση. Επιπλέον, πολλές Cocoa τεχνολογίες και αρχιτεκτονικές βασίζονται σε MVC και απαιτούν τα αντικείμενα της κάθε εφαρμογής να έχουν ένα από τους τρεις ρόλους.



Εικόνα 5. 2 Η επικοινωνία των μερών του Model View Controller

Τα model objects ενθυλακώνουν τα δεδομένα αποκλειστικά σε μια εφαρμογή και καθορίζουν τη λογική και τους υπολογισμούς που διαχειρίζονται και επεξεργάζονται εκείνα τα δεδομένα. Για παράδειγμα ένα αντικείμενο αυτής της κατηγορίας μπορεί να αναπαριστά ένα χαρακτήρα σε ένα παιχνίδι ή μια επαφή σε ένα τηλεφωνικό κατάλογο. Επίσης μπορεί να έχει σχέσεις ένα-προς-πολλά με άλλα αντικείμενα του ίδιου τύπου και για το λόγο αυτό κάποιες φορές το επίπεδο μοντέλου μιας εφαρμογής είναι ένας ή περισσότεροι γράφοι αντικειμένων. Πολλά από τα δεδομένα που είναι μέρος μιας σταθερής κατάστασης της εφαρμογής πρέπει να παραμένουν στα αντικείμενα του μοντέλου όταν τα δεδομένα φορτώνονται στην εφαρμογή. Επειδή τα αντικείμενα μοντέλου αναπαριστούν γνώση και εμπειρία σχετικά με ένα συγκεκριμένο πεδίο προβλήματος μπορούν να επαναχρησιμοποιηθούν σε προβλήματα παρόμοιας κατηγορίας. Ιδανικά, ένα αντικείμενο μοντέλου δεν πρέπει να έχει καθόλου εξαρτήσεις και συνδέσεις με τα αντικείμενα όψης που αναπαριστούν τα δεδομένα του ενώ πρέπει να επιτρέπει στους χρήστες να επεξεργάζονται τα δεδομένα τους. Την ίδια στιγμή, δεν πρέπει να ασχολούνται καθόλου με διεπαφές χρήστη και θέματα εμφάνισης.

Όσον αφορά την επικοινωνία, οι ενέργειες του χρήστη στο επίπεδο όψης που δημιουργούν ή τροποποιούν τα δεδομένα, επιτυγχάνονται μέσω ενός αντικειμένου controller και έχουν ως αποτέλεσμα τη δημιουργία και ενημέρωση ενός αντικειμένου μοντέλου. Όταν ένα αντικείμενο μοντέλου αλλάζει (για παράδειγμα όταν λαμβάνονται νέα δεδομένα μέσω μιας σύνδεσης δικτύου) ειδοποιεί το αντικείμενο ελεγκτή το οποίο ενημερώνει τα κατάλληλα αντικείμενα όψης.

Ένα View object είναι ένα αντικείμενο το οποίο οι χρήστες μπορούν να δουν. Ένα τέτοιο αντικείμενο ξέρει πώς να σχεδιάζει τον εαυτό του και να αντιδρά σε ενέργειες του χρήστη. Ένας από τους κύριους σκοπούς των αντικειμένων αυτής της κατηγορίας είναι να εμφανίζουν δεδομένα από τα αντικείμενα μοντέλου της εφαρμογής και να επιτρέπουν την επεξεργασία αυτών των δεδομένων. Ανεξάρτητα από αυτό, τα αντικείμενα όψης συνήθως είναι αποσυνδεδεμένα από τα αντικείμενα μοντέλου σε μια MVC εφαρμογή. Επειδή συνήθως ο συγγραφέας μιας εφαρμογής επαναχρησιμοποιεί και επανακαθορίζει αυτά τα αντικείμενα, τα αντικείμενα όψης παρέχουν συνέπεια μεταξύ των εφαρμογών. Τόσο το UIKit framework όσο και το AppKit framework παρέχουν συλλογές από view classes ενώ ο Interface Builder προσφέρει δεκάδες αντικείμενα στη βιβλιοθήκη του. Όσον αφορά την επικοινωνία, τα αντικείμενα όψης μαθαίνουν για τις αλλαγές στα δεδομένα του μοντέλου μέσω των αντικειμένων ελεγκτή της εφαρμογής και χειρίζονται τις αλλαγές που προκαλούν οι χρήστες (για παράδειγμα εισαγωγή κειμένου σε πεδίο κειμένου) μέσω των αντικειμένων ελεγκτή.

Ένα controller object δρα ως ενδιάμεσο μεταξύ ενός ή περισσότερων view objects και ενός ή περισσότερων model objects. Τα αντικείμενα ελεγκτή είναι στην πράξη ένας αγωγός μέσω του οποίου τα αντικείμενα μαθαίνουν σχετικά με τις αλλαγές στα αντικείμενα μοντέλου και αντίστροφα. Επίσης, τα αντικείμενα αυτής της κατηγορίας μπορούν να πραγματοποιούν εγκατάσταση και συντονισμό πράξεων για μια εφαρμογή και να διαχειρίζονται τον κύκλο ζωής των άλλων αντικειμένων. Όσον αφορά την επικοινωνία, ένα αντικείμενο ελεγκτή μεταφράζει τις ενέργειες του χρήστη που πραγματοποιούνται σε αντικείμενα όψης και μεταφέρει τα νέα ή τα αλλαγμένα δεδομένα στο επίπεδο μοντέλου. Όταν ένα αντικείμενο μοντέλου αλλάζει, ένα αντικείμενο ελεγκτή μεταφέρει τα νέα δεδομένα στα αντικείμενα όψης έτσι ώστε αυτά να μπορέσουν να τα εμφανίσουν.

Στη συνέχεια υπάρχουν και τα Block Objects τα οποία αποτελούν έναν εύκολο τρόπο ενσωμάτωσης κώδικα και μεταβλητών στοίβας σε μια μορφή που μπορεί να

εκτελεστεί στη συνέχεια. Η υποστήριξη block objects είναι διαθέσιμη για το iOS 4 και μετά όπου τα Blocks συχνά δρουν ως callbacks για ασύγχρονες λειτουργίες. Τα block objects είναι ένα χαρακτηριστικό επιπέδου C που επιτρέπει στο χρήστη που αναπτύσσει εφαρμογές για iOS να συνθέσει εκφράσεις συναρτήσεων που μπορούν να περαστούν ως ορίσματα, που προαιρετικά μπορούν να αποθηκευτούν, και να χρησιμοποιηθούν από πολλά νήματα. Η έκφραση της συνάρτησης μπορεί να αναφέρει και να διατηρήσει πρόσβαση σε τοπικές μεταβλητές. Ένα block χρησιμοποιείται όταν κανείς θέλει να δημιουργήσει τμήματα κώδικα που μπορούν να περαστούν λες και είναι τιμές. Προσφέρουν ευέλικτο προγραμματισμό και περισσότερη δύναμη. Είναι κατάλληλα για τη συγγραφή callbacks και για την πραγματοποίηση πράξεων σε όλα τα αντικείμενα μιας συλλογής.

Το delegation σχεδιαστικό πρότυπο είναι μια λύση τροποποίησης σύνθετων αντικειμένων χωρίς να χρειαστεί subclassing σε αυτά. Τα αντικείμενα χρησιμοποιούνται ως έχουν και οποιοσδήποτε κώδικας ενσωματώνεται σε ένα ξεχωριστό αντικείμενο για να τροποποιηθεί η συμπεριφορά του εκάστοτε αντικειμένου. Το εναλλακτικό αντικείμενο στο οποίο ενσωματώνεται ο κώδικας ονομάζεται delegate object (αντιπρόσωπος). Σε προκαθορισμένες στιγμές, το σύνθετο αντικείμενο καλεί τις μεθόδους του delegate αντικειμένου για να του δώσει την ευκαιρία να τρέξει το δικό του κώδικα.

Είναι αντιληπτό ότι πρόκειται για ένα απλό και παντοδύναμο πρότυπο με το οποίο ένα αντικείμενο σε ένα πρόγραμμα δρα εκ μέρους ή σε συνδυασμό με κάποιο άλλο αντικείμενο. Το delegating object (αντιπροσωπευόμενο) κρατά μια αναφορά στο αντικείμενο delegate και στον καθορισμένο χρόνο στέλνει μήνυμα σε αυτό. Το μήνυμα πληροφορεί το delegate για γεγονός το οποίο το delegating πρόκειται να χειριστεί ή έχει μόλις χειριστεί. Το delegate μπορεί να απαντήσει στο μήνυμα ανανεώνοντας την εμφάνιση ή την κατάσταση του εαυτού του ή άλλων αντικειμένων στην εφαρμογή ή ακόμα και να επιστρέψει μια τιμή που επηρεάζει το πώς ένα γεγονός θα τύχει χειρισμού. Η κύρια τιμή μιας αντιπροσωπείας (delegation) είναι ότι

επιτρέπει την εύκολη τροποποίηση της συμπεριφοράς αρκετών αντικειμένων σε ένα κεντρικό αντικείμενο. Το delegating αντικείμενο είναι συνήθως ένα framework object ενώ το delegate object είναι συνήθως ένα controller object. Το delegating object διατηρεί μια χαλαρή αναφορά στο delegate.

Ακολουθεί το target-action σχεδιαστικό πρότυπο το οποίο χρησιμοποιείται από τους ελέγχους για την ειδοποίηση της εφαρμογής για τις ενέργειες του χρήστη. Όταν ο χρήστης αλληλεπιδρά με κάποιον έλεγχο με ένα προκαθορισμένο τρόπο (π.χ. αγγίζοντας ένα κουμπί), ο έλεγχος στέλνει ένα μήνυμα (την ενέργεια) σε ένα αντικείμενο το οποίο έχει από πριν καθοριστεί. (ο στόχος). Όταν το μήνυμα ενέργειας ληφθεί, το αντικείμενο-στόχος μπορεί να απαντήσει με τον κατάλληλο τρόπο.

Το target-action πρότυπο επιτρέπει στο αντικείμενο να κρατήσει τις απαραίτητες πληροφορίες για την αποστολή ενός μηνύματος όταν ένα γεγονός συμβεί. Η αποθηκευμένη πληροφορία αποτελείται από δύο αντικείμενα δεδομένων: τον εκλογέα ενέργειας(action selector) ο οποίος αναγνωρίζει τη μέθοδο στην οποία θα παρέμβει και ένα στόχο (target) ο οποίος είναι το αντικείμενο το οποίο θα λάβει το μήνυμα. Το μήνυμα που αποστέλλεται όταν συμβεί το γεγονός ονομάζεται action message. Παρόλο που ο στόχος μπορεί να είναι οποιοδήποτε αντικείμενο, ακόμα και ένα framework object, είναι συνήθως ένας τυπικός ελεγκτής που χειρίζεται τα action messages με ένα τρόπο κατάλληλο για την κάθε εφαρμογή.

Το action message που μπορεί να προκαλέσει ένα γεγονός μπορεί να είναι οτιδήποτε . Το αντικείμενο που θα στείλει το μήνυμα μπορεί επίσης να είναι οποιοδήποτε αντικείμενο. Όταν ένας χρήστης χειρίζεται ένα control object στέλνει ένα μήνυμα στο συγκεκριμένο αντικείμενο. Το control object είναι ένα στιγμιότυπο υποκλάσης του UIControl. Τόσο ο action selector όσο και το αντικείμενο στόχος είναι ιδιότητες του control object ή στην περίπτωση του AppKit framework είναι ιδιότητες του control’s cell object.

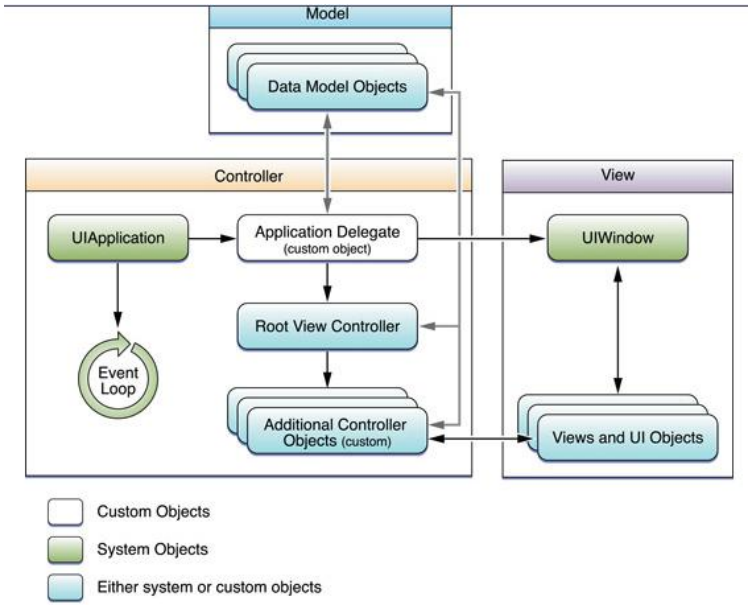
Όσον αφορά το managed memory model, η γλώσσα Objective C χρησιμοποιεί ένα σχήμα μετρητών αναφοράς για τον καθορισμό του πότε θα απελευθερώνονται τα

αντικείμενα από τη μνήμη. Ισχύουν όλες οι αρχές που αναφέρθηκαν στην ενότητα διαχείρισης μνήμης.

Τέλος, όσον αφορά τα νήματα και τον ασύγχρονο προγραμματισμό, όλες οι εκδόσεις του iOS υποστηρίζουν τη δημιουργία operation objects και δευτερευόντων νημάτων. Από το iOS 4 και μετά, οι εφαρμογές μπορούν επίσης να χρησιμοποιήσουν Grand Central Dispatch (GCD) για την εκτέλεση λειτουργιών ασύγχρονα.

Από τη στιγμή που η εφαρμογή θα εκτελεστεί από το χρήστη στο τηλέφωνο μέχρι τη στιγμή που αυτή θα τερματίσει, το UIKit framework διαχειρίζεται την περισσότερη συμπεριφορά πυρήνα της εφαρμογής. Για παράδειγμα, μια εφαρμογή iOS λαμβάνει γεγονότα συνεχώς από το σύστημα και πρέπει να απαντά σε αυτά. Η παραλαβή των γεγονότων είναι αρμοδιότητα του UIApplication object αλλά η απάντηση στα γεγονότα είναι αρμοδιότητα του κώδικα που έχει γράψει ο δημιουργός της εφαρμογής. Παρόμοιες σχέσεις υπάρχουν και σε άλλα τμήματα της εφαρμογής όπου το σύστημα χειρίζεται την όλη διεργασία και ο κώδικας του δημιουργού επικεντρώνεται στην υλοποίηση της συγκεκριμένης συμπεριφοράς της εφαρμογής.

Για να είναι κατανοητή η δράση των UIKit αντικειμένων και η συνεργασία τους με τον κώδικα του χρήστη απαιτείται η κατανόηση των αντικειμένων που συμμετέχουν στην εφαρμογή. Η εικόνα 5.3 [23] παρουσιάζει τους ρόλους των πιο συχνά χρησιμοποιούμενων αντικειμένων στις iOS εφαρμογές.



Εικόνα 5. 3 Οι ρόλοι των πιο συχνά χρησιμοποιούμενων αντικειμένων στις iOS εφαρμογές

Το UIApplication object διαχειρίζεται τον βρόχο γεγονότων της εφαρμογής και συντονίζει τις υψηλού επιπέδου συμπεριφορές της εφαρμογής. Αυτό το αντικείμενο χρησιμοποιείται όπως ακριβώς είναι κυρίως για την επαλήθευση διάφορων μεταβλητών που αφορούν διάφορες πτυχές της εμφάνισης της εφαρμογής. Ο τροποποιημένος κώδικας επιπέδου εφαρμογής βρίσκεται στο delegate αντικείμενο της εφαρμογής το οποίο δουλεύει διαδοχικά με αυτό το αντικείμενο.

Ένα application object είναι υπεύθυνο για την αρχική δρομολόγηση των γεγονότων του χρήστη και την όλη διαχείριση της εφαρμογής που εκτελείται. Όταν μια εφαρμογή τρέχει, δημιουργεί το application object στη main συνάρτησή της. Μέσα στο βρόχο γεγονότων της main, το application object δέχεται ένα εισερχόμενο γεγονός, που αναπαριστά την ενέργεια του χρήστη, και το δρομολογεί στο παράθυρο που περιέχει την όψη στην οποία επικεντρώνεται η ενέργεια. Επίσης λαμβάνει μηνύματα ενέργειας από ελέγχους και τα προωθεί στους κατάλληλους στόχους. Διατηρεί μια λίστα με τα παράθυρά του και διαχειρίζεται τις τρέχουσες καταστάσεις τους.

Επιπλέον, ένα application object λαμβάνει ειδοποιήσεις από το λειτουργικό σύστημα για το πότε εξωτερικά γεγονότα επηρεάζουν την ίδια την εφαρμογή (για παράδειγμα όταν η μνήμη είναι περιορισμένη). Το application object καταγράφει τη βοήθεια που δέχεται από τα delegate στο χειρισμό τέτοιων εξωτερικών γεγονότων καθώς και γεγονότων που έχουν να κάνουν με τον κύκλο ζωής της εφαρμογής. Πληροφορεί τα delegate για τα γεγονότα αυτά και σε κάποιες περιπτώσεις δρα στις απαντήσεις που αυτά δίνουν στα μηνύματα που στους αποστέλλει. Κάθε εφαρμογή έχει ένα μόνο application object το οποίο είναι διαθέσιμο σε όλα τα αντικείμενα της εφαρμογής. Στο iOS το application object είναι ένα στιγμιότυπο της κλάσης UIApplication.

Όσον αφορά το application delegate object, είναι ένα αντικείμενο που παρέχεται στο χρόνο εκτέλεσης της εφαρμογής όταν έχει ενσωματωθεί στο main nib αρχείο της. Η κύρια δουλειά αυτού του αντικειμένου είναι η αρχικοποίηση της εφαρμογής και η

παρουσία του παραθύρου της στην οθόνη. Το UIApplication object ειδοποιεί αυτό το αντικείμενο όταν συγκεκριμένα γεγονότα επιπέδου εφαρμογής προκύπτουν, όπως για παράδειγμα όταν μια εφαρμογή πρέπει να διακοπεί ή να μετακινηθεί στο παρασκήνιο.

Τα data model objects αποθηκεύουν τα περιεχόμενα της εφαρμογής και είναι με τον τρόπο αυτό εξειδικευμένα για κάθε εφαρμογή. Για παράδειγμα μια τραπεζική εφαρμογή μπορεί να αποθηκεύει μια βάση δεδομένων που περιλαμβάνει οικονομικές συναλλαγές ενώ μια εφαρμογή ζωγραφικής μπορεί να αποθηκεύει αντικείμενα εικόνων.

Τα view controller objects διαχειρίζονται την παρουσίαση του περιεχομένου της εφαρμογής. Τυπικά αυτό περιλαμβάνει τη δημιουργία όψεων για την παρουσίαση του περιεχομένου και τη διαχείριση των αλληλεπιδράσεων μεταξύ των όψεων και των δεδομένων των model objects της εφαρμογής. Η κλάση UIViewController είναι η βασική κλάση για όλα τα view controllers objects. Παρέχει την προκαθορισμένη λειτουργικότητα για τη ζωντανή εικόνα των όψεων, τη διαχείριση των περιστροφών της συσκευής και άλλων βασικών συμπεριφορών συστήματος.

Το UIWindow object συντονίζει την παρουσίαση ενός ή περισσότερων όψεων στην οθόνη της συσκευής ή σε μια εξωτερική οθόνη. Οι περισσότερες εφαρμογές έχουν μόνο ένα παράθυρο το περιεχόμενο του οποίου παρέχεται από μια ή περισσότερες όψεις. Μια εφαρμογή αλλάζει το περιεχόμενο αυτού του παραθύρου αλλάζοντας το τρέχον σύνολο όψεων με τη βοήθεια ενός view controller object. Επιπρόσθετα με τις hosting views, τα παράθυρα είναι επίσης υπεύθυνα για την παράδοση των γεγονότων σε αυτές τις όψεις και το χειρισμό των view controllers.

Το window object αναπαριστά το γραφικό περιεχόμενο σε μια ορθογώνια περιοχή της οθόνης και παίζει το ρόλο του διανομέα γεγονότων. Αυτό το αντικείμενο κληρονομεί από το UIWindow του Cocoa Touch.Τα window objects είναι στην ουσία περιέκτες όψεων και έχουν σημαντικές υπευθυνότητες που σχετίζονται με την εμφάνιση και το χειρισμό γεγονότων.

Ένα window object βρίσκεται στη ρίζα της ιεραρχίας των όψεων. Όταν ένα παράθυρο εμφανίζει το περιεχόμενό του, προχωρά προς τα κάτω στην ιεραρχία ζητώντας από κάθε όψη με τη σειρά να ζωγραφίσει το δικό της περιεχόμενο. Η όψη που βρίσκεται στο πάνω μέρος της ιεραρχίας είναι η content view η οποία καταλαμβάνει ολόκληρη την περιοχή εμφάνισης περιεχομένου στο παράθυρο. Εξυπηρετεί με τον τρόπο αυτό ως σκηνικό για όλες τις άλλες όψεις στο παράθυρο.

Το σύστημα συντονισμού των παραθύρων που ονομάζεται base coordinate system σχετίζεται με την παρουσίαση του περιεχομένου. Ό,τι ζωγραφίζεται από τις όψεις σε ένα παράθυρο ιδανικά αναφέρεται στο base coordinate system. Όμως όταν ένα παράθυρο από μόνο του τοποθετείται και παίρνει το μέγεθος του στην οθόνη χρησιμοποιεί το δικό του σύστημα συντονισμού. Οι κλάσεις των window objects καθορίζουν μεθόδους για μετατροπή μεταξύ του τοπικού συστήματος συντονισμού και του βασικού συστήματος. Επιπλέον, ένα window object διανέμει τα εισερχόμενα γεγονότα στις όψεις στην ιεραρχία του που είναι οι πιο κατάλληλοι παραλήπτες για αυτά τα γεγονότα. Για παράδειγμα, γεγονότα που έχουν να κάνουν με το ποντίκι (mouse events) έχουν παραλήπτη την όψη κάτω από τον mouse pointer. Για γεγονότα-κλειδιά και μηνύματα ενεργειών που δεν έχουν κάποιο στόχο, ο παραλήπτης είναι ο πρώτος ο οποίος θα απαντήσει σε αυτά, δηλαδή το αντικείμενο που πρώτο θα ανταποκριθεί στο γεγονός ή στο μήνυμα. Το window object είναι υπεύθυνο να εντοπίσει τον πρώτο που θα απαντήσει.

Παρόλο που τα παράθυρα στο Cocoa Touch διαμοιράζονται τις υπευθυνότητες, έχουν διαφορετικά χαρακτηριστικά και συμπεριφορές. Οι διαφορές αυτές προκύπτουν από τα διαφορετικά περιβάλλοντα χρήστη όπως για παράδειγμα ένα desktop system σε σχέση με την οθόνη μιας συσκευής χεριού με περιορισμένη περιοχή οθόνης. Τα διαφορετικά περιβάλλοντα χρήστη οδηγούν σε διαφορετικά πρότυπα χρήσης και απαιτήσεις.

Τέλος, υπάρχουν τα view, control και layer objects. Οι όψεις και οι ελεγκτές παρέχουν την οπτική αναπαράσταση του περιεχομένου της εφαρμογής. Μια όψη

είναι ένα αντικείμενο το οποίο σχεδιάζει το περιεχόμενο σε μια ορθογώνια περιοχή και απαντά σε γεγονότα εντός αυτής της περιοχής. Οι ελεγκτές είναι ένας εξειδικευμένος τύπος όψης, που είναι υπεύθυνοι για την υλοποίηση γνώριμων interfaces objects όπως είναι τα κουμπιά, τα πεδία κειμένου και τα toggle switches. Το UIKit framework παρέχει πρότυπες όψεις για την παρουσίαση πολλών ειδών περιεχομένου. Επίσης, ο δημιουργός της κάθε εφαρμογής είναι ελεύθερος να δημιουργήσει τη δική του όψη εξειδικεύοντας την κλάση UIView απευθείας.

Τα layer objects είναι αντικείμενα δεδομένων που αναπαριστούν οπτικό περιεχόμενο. Οι όψεις χρησιμοποιούν layer objects πίσω από τις οθόνες για να παρουσιάσουν το περιεχόμενό τους. Ο χρήστης μπορεί να προσθέσει τα δικά του layer objects στη διασύνδεση της εφαρμογής του για να υλοποιήσει σύνθετα animations και άλλους τύπους εξειδικευμένου περιεχομένου.

Τα αντικείμενα μιας εφαρμογής συνθέτουν ένα σύνθετο περιβάλλον τα περιεχόμενα του οποίου καθορίζουν την εφαρμογή. Όπως φαίνεται στην εικόνα 5.3 τα περισσότερα αντικείμενα σε μια εφαρμογή είναι είτε πλήρως είτε μερικώς παραμετροποιήσιμα. Μια εφαρμογή που κτίζεται πάνω από τις υπάρχουσες UIKit κλάσεις λαμβάνει ένα σημαντικό ποσό υποδομής χωρίς καμιά επιβάρυνση. Τότε ο δημιουργός πρέπει απλά να εξειδικεύσει την τυπική, προκαθορισμένη συμπεριφορά και να υλοποιήσει αυτήν που επιθυμεί.

5.3 Τα κύρια συστατικά μιας εφαρμογής για το iOS της Apple και η πορεία εκτέλεσης της

Η συνάρτηση main: Όπως οποιαδήποτε εφαρμογή στη γλώσσα C, έτσι και στις iOS εφαρμογές, το κύριο σημείο εισόδου στην εκκίνηση της εφαρμογής είναι η συνάρτηση main. Σε μια iOS εφαρμογή, η συνάρτηση main χρησιμοποιείται ελάχιστα. Η κύρια υπευθυνότητά της είναι να παραδίδει τον έλεγχο στο UIKit framework. Έτσι, κάθε νέο project που δημιουργείται στο Xcode έρχεται με διαφορετική συνάρτηση

main. Με ελάχιστες εξαιρέσεις, οι δημιουργοί iOS εφαρμογών δεν πρέπει να αλλάζουν καθόλου τον κώδικα υλοποίησης αυτής της συνάρτησης.

Η συνάρτηση UIApplicationMain στην καρδιά της συνάρτησης main σε μια iOS εφαρμογή, δέχεται τέσσερις παραμέτρους και τις χρησιμοποιεί για την αρχικοποίηση της εφαρμογής. Επιπρόσθετα με τη δημιουργία του application object και τη δημιουργία ή φόρτωση του application delegate, η συνάρτηση UIApplicationMain φορτώνει επίσης το nib αρχείο της main. Όλα τα Xcode projects περιέχουν nib αρχείο το οποίο συνήθως συμπεριλαμβάνει το παράθυρο της εφαρμογής και το application delegate αντικείμενο. Το UIKit παίρνει το όνομα του nib αρχείου από μια μεταβλητή που το περιέχει στην εφαρμογή. Παρόλο που δε χρειάζεται συχνά, ο δημιουργός μιας εφαρμογής έχει την ευχέρεια, αν το επιθυμήσει, να καθορίσει το δικό του nib αρχείο αλλάζοντας την τιμή της παραμέτρου που περιέχει το όνομά του και δίνοντας ένα άλλο ως τιμή.

The application delegate: Η παρακολούθηση της συμπεριφοράς υψηλού επιπέδου του συστήματος για μια εφαρμογή είναι υπευθυνότητα του αντικειμένου αυτού. Πρόκειται για ένα παραμετροποιημένο αντικείμενο το οποίο παρέχει ο δημιουργός της εφαρμογής. Ο delegation μηχανισμός χρησιμοποιείται για την αποφυγή subclassing σύνθετων UIKit αντικειμένων, όπως αναφέρθηκε και πιο πάνω. Αντί να γίνει subclassing και επανακαθορισμός των μεθόδων αυτών των σύνθετων αντικειμένων, χρησιμοποιείται το αντικείμενο όπως ακριβώς είναι μαζί με τον κώδικα τον οποίο ο χρήστης τοποθετεί μέσα στο delegate αντικείμενο. Το application delegate object είναι υπεύθυνο για το χειρισμό αρκετών κρίσιμων μηνυμάτων συστήματος και πρέπει να υπάρχει σε κάθε iOS εφαρμογή. Το αντικείμενο μπορεί να είναι στιγμιότυπο οποιασδήποτε κλάσης φτάνει να κληρονομεί από το UIApplicationDelegate πρωτόκολλο. Οι μέθοδοι που περιέχονται σε αυτό το πρωτόκολλο καθορίζουν τα σημεία του κύκλου ζωής της εφαρμογής τα οποία θα υλοποιηθούν με συγκεκριμένη συμπεριφορά.

Καταστάσεις και μεταβάσεις μιας εφαρμογής: Οι εφαρμογές που τρέχουν σε iOS 4 και μετά, μπορούν να βρίσκονται σε μια από αρκετές καταστάσεις κάθε συγκεκριμένη χρονική στιγμή. Οι εφαρμογές που τρέχουν σε iOS 3.2 και πιο πριν, δεν τίθενται στο παρασκήνιο ή σε ανασταλμένες καταστάσεις. Οι καταστάσεις στις οποίες μπορεί να περιέλθει μια εφαρμογή ακολουθούν.

Αρχικά, η κατάσταση Not Running (Δεν εκτελείται). Σε αυτήν, η εφαρμογή δεν έχει ακόμα εκτελεστεί ή έχει ήδη εκτελεστεί αλλά έχει τερματιστεί από το σύστημα. Στη συνέχεια υπάρχει η Inactive (Μη ενεργή) όπου η εφαρμογή εκτελείται στο προσκήνιο αλλά προς το παρόν δε λαμβάνει γεγονότα. Μια εφαρμογή παραμένει σε αυτή την κατάσταση μόνο για λίγο χρονικό διάστημα όταν μεταφέρεται από μια κατάσταση σε κάποια άλλη. Η μόνη χρονική στιγμή στην οποία παραμένει μη ενεργή για οποιοδήποτε χρονικό διάστημα είναι όταν ο χρήστης κλειδώσει την οθόνη ή όταν το σύστημα απαιτεί από τον χρήστη να απαντήσει σε κάποιο γεγονός (π.χ. απάντηση τηλεφωνικής κλήσης ή άνοιγμα εισερχόμενου μηνύματος). Ακολούθως υπάρχει και η κατάσταση Active (Ενεργή) κατά τη διάρκεια της οποίας η εφαρμογή εκτελείται στο προσκήνιο και λαμβάνει γεγονότα. Υπάρχει επίσης η κατάσταση Background (Παρασκήνιο) όπου η εφαρμογή βρίσκεται στο παρασκήνιο και εκτελείται κάποιος κώδικας. Οι περισσότερες εφαρμογές εισέρχονται σε αυτή την κατάσταση για λίγο χρονικό διάστημα πριν εισέλθουν στην κατάσταση αναστολής. Παρόλα αυτά, μια εφαρμογή που απαιτεί επιπλέον χρόνο εκτέλεσης είναι δυνατόν να παραμείνει σε αυτή την κατάσταση για αρκετό χρόνο. Επιπλέον, μια εφαρμογή που εκτελείται απευθείας από το παρασκήνιο μπαίνει σε αυτή την κατάσταση αντί για την κατάσταση Μη ενεργή. Η κατάσταση αυτή είναι διαθέσιμη μόνο στο iOS 4 και μετά και για συσκευές που υποστηρίζουν multitasking. Αν η κατάσταση αυτή δεν είναι διαθέσιμη, οι εφαρμογές τερματίζονται και μεταφέρονται στη μη εκτελέσιμη κατάσταση. Τέλος, υπάρχει η κατάσταση Suspended (Σε αναστολή), όπου η εφαρμογή βρίσκεται στο παρασκήνιο αλλά δεν εκτελείται καθόλου κώδικας. Το σύστημα μεταφέρει την εφαρμογή αυτόματα σε αυτή την κατάσταση όταν χρειάζεται.

Καθώς η εφαρμογή βρίσκεται σε αναστολή, είναι «παγωμένη» στην τρέχουσα κατάστασή της και δεν εκτελεί καμιά συνάρτηση ή λειτουργία. Σε συνθήκες περιορισμένης διαθέσιμης μνήμης, το σύστημα μπορεί να υποχρεώσει σε τερματισμό τις εφαρμογές που βρίσκονται σε αναστολή χωρίς ειδοποίηση έτσι ώστε να δημιουργηθεί διαθέσιμος χώρος για τις εφαρμογές που βρίσκονται στο προσκήνιο. Και αυτή η κατάσταση είναι διαθέσιμη μόνο στο iOS 4 και μετά και για συσκευές που υποστηρίζουν multitasking. Σε περίπτωση που η κατάσταση αυτή δεν είναι διαθέσιμη, οι εφαρμογές τερματίζονται και μεταφέρονται στην κατάσταση Μη εκτελέσιμη.

Εκτέλεση εφαρμογής: Κατά το χρόνο εκτέλεσης μιας εφαρμογής, η εφαρμογή μεταφέρεται από την κατάσταση Μη εκτελέσιμη στην κατάσταση Ενεργή ή Στο Παρασκήνιο. Μια εφαρμογή που εκτελείται πρέπει να προετοιμάσει τον εαυτό της για να εκτελεστεί και στη συνέχεια να ελέγξει αν το σύστημα την έχει εκτελέσει για την πραγματοποίηση μιας λειτουργίας. Κατά τη διάρκεια της ακολουθίας αρχικοποίησης η εφαρμογή καλεί τις μεθόδους delegate ακολουθούμενες από τις μεθόδους applicationDidBecomeActive ή applicationDidEnterBackground.

Η delegate μέθοδος της εφαρμογής είναι υπεύθυνη για την περισσότερη δουλειά κατά τη διάρκεια της εκτέλεσης και έχει τις ακόλουθες αρμοδιότητες: αρχικοποίηση των δομών της εφαρμογής και φόρτωση ή δημιουργία του κύριου παραθύρου της εφαρμογής και των όψεων. Η μέθοδος αυτή πρέπει να είναι όσο το δυνατόν πιο «ελαφριά» για το σύστημα. Δίνονται περίπου 5 δευτερόλεπτα για αρχικοποίηση και επιστροφή σε αυτή τη μέθοδο ενώ σε περίπτωση που δεν ολοκληρώσει τις λειτουργίες της το σύστημα μπορεί να την τερματίσει λόγω ανευθυνότητας. Η μετατροπή της μεθόδου αυτής σε «ελαφριά» είναι πολύ σημαντική ιδίως για εφαρμογές που βασίζονται σε λειτουργίες δικτύου, μιας και αυτές υπάρχει περίπτωση να χρειαστούν ενδιάμεσο χρόνο για να ολοκληρωθούν.

Όταν η μέθοδος application:didFinishLaunchingWithOptions καλείται, η ιδιότητα applicationState του UIApplication object έχει ήδη πάρει την κατάλληλη κατάσταση

της εφαρμογής ως τιμή. Αν η ιδιότητα έχει πάρει την τιμή UIApplicationStateInactive η εφαρμογή βρίσκεται στη Μη ενεργή κατάσταση και πρόκειται να μεταφερθεί στο προσκήνιο. Αν η κατάσταση της είναι UIApplicationStateBackground η εφαρμογή πρόκειται να μετακινηθεί στο παρασκήνιο. Ανάλογα με την κατάσταση στην οποία βρίσκεται η εφαρμογή προετοιμάζεται κατάλληλα.

Οι εφαρμογές τρέχουν στο παρασκήνιο όταν χρειάζεται για το χειρισμό ενός εισερχόμενου γεγονότος. Όταν τρέχουν στο παρασκήνιο η εφαρμογή έχει περιορισμένο χρόνο εκτέλεσης και πρέπει να αποφύγει να κάνει οποιαδήποτε δουλειά δεν είναι σχετική με την επεξεργασία του παρασκηνιακού γεγονότος.

Όταν ο χρήστης επιλέξει το Home κουμπί ή όταν το σύστημα τρέξει μια άλλη εφαρμογή, η εφαρμογή που ήταν στο προσκήνιο μεταφέρεται αρχικά στη Μη ενεργή κατάσταση και στη συνέχεια στην κατάσταση Παρασκήνιο. Αυτές οι ενέργειες οδηγούν στην κλήση των delegate μεθόδων της εφαρμογής applicationWilResignActive και applicationDidEnterBackground.Οι περισσότερες εφαρμογές που βρίσκονται στο παρασκήνιο μεταφέρονται στην κατάσταση Σε αναστολή όταν επιστρέψουν από τη μέθοδο applicationDidEnterBackground. Αν η εφαρμογή απαιτεί περισσότερο χρόνο εκτέλεσης ή αν δηλώνει ότι θέλει να εκτελεστεί στο παρασκήνιο επιτρέπεται να συνεχίσει να τρέχει μετά και την επιστροφή της μεθόδου. Όταν μια μέθοδος μεταφέρεται στο παρασκήνιο, όλα τα θεμελιώδη αντικείμενα της εφαρμογής παραμένουν στη μνήμη και είναι διαθέσιμα για χρήση. Αυτά τα αντικείμενα περιλαμβάνουν τα παραμετροποιημένα αντικείμενα του χρήστη και τις δομές δεδομένων μαζί με τα αντικείμενα ελεγκτή, παραθύρων, όψεων και επιπέδων. Όμως το σύστημα δεν ελευθερώνει πολλά από τα αντικείμενα που χρησιμοποιούνται στο παρασκήνιο για την υποστήριξη της εφαρμογής. Πιο συγκεκριμένα, το σύστημα εκτελεί τις παρακάτω παρασκηνιακές εφαρμογές: Ελευθερώνει το χώρο υποστήριξης για όλα τα core animation επίπεδα, πράγμα το οποίο αποτρέπει το περιεχόμενο αυτών των επιπέδων να εμφανιστεί στην οθόνη χωρίς να αλλάξει τις ιδιότητες του τρέχοντος επιπέδου. Δηλαδή δεν ελευθερώνει τα

ίδια τα layer objects. Επίσης ελευθερώνει όλες τις αναφορές προς cashed εικόνες καθώς και κάποια cached δεδομένα που διαχειρίζεται το σύστημα.

Και αυτή η μέθοδος έχει 5 δευτερόλεπτα περίπου μέχρι να επιστρέψει αλλιώς η εφαρμογή τερματίζεται χωρίς ειδοποίηση και αφαιρείται από τη μνήμη. Όλες οι multitasking εφαρμογές πρέπει να συμπεριφέρονται υπεύθυνα όταν μετακινούνται στο παρασκήνιο ανεξάρτητα από το αν θα συνεχίσουν να τρέχουν στο παρασκήνιο ή αν θα τεθούν Σε αναστολή αμέσως μετά την επιστροφή τους από το παρασκήνιο. Έτσι υπάρχουν δύο πράγματα τα οποία πρέπει να γίνονται στη μέθοδο applicationDidEnterBackground. Αρχικά πρέπει η εφαρμογή να είναι έτοιμη για «λήψη φωτογραφίας». Όταν η μέθοδος επιστρέψει, το σύστημα κρατά την εικόνα που έχει η διασύνδεση χρήστη της εφαρμογής και τη χρησιμοποιεί για τα transition animations. Αν κάποιες από τις όψεις της εφαρμογής περιέχουν ευαίσθητη πληροφορία πρέπει να αποκρυφτούν ή να τροποποιηθούν οι όψεις αυτές πριν επιστρέψει η μέθοδος.

Το δεύτερο που πρέπει να κάνει είναι να αποθηκεύσει τα δεδομένα του χρήστη και τις πληροφορίες που έχουν να κάνουν με την κατάσταση της εφαρμογής. Όλες οι μη αποθηκευμένες αλλαγές πρέπει να γραφούν στο δίσκο όταν η εφαρμογή θα εισέλθει στο παρασκήνιο. Το βήμα αυτό είναι απαραίτητο επειδή η εφαρμογή μπορεί να τερματιστεί ενόσω βρίσκεται στο παρασκήνιο για αρκετούς λόγους.

Απάντηση σε διακοπές: Όταν προκύψει μια προσωρινή διακοπή (π.χ. μια εισερχόμενη κλήση) η εφαρμογή μεταφέρεται προσωρινά στην κατάσταση Μη ενεργή. Παραμένει σε αυτή την κατάσταση μέχρι ο χρήστης να αποφασίσει αν θα αποδεχθεί ή θα αγνοήσει τη διακοπή. Όταν μεταφέρεται στην κατάσταση Μη ενεργή μια εφαρμογή, πρέπει να θέσει τον εαυτό της σε μια όσο το δυνατόν πιο «αθόρυβη» κατάσταση. Αν ο χρήστης αγνοήσει τη διακοπή, η εφαρμογή επανα-ενεργοποιείται σε χρόνο στον οποίο μπορεί να συνεχίσει με την κανονική της εκτέλεση. Παρόλα αυτά, όταν ο χρήστης δεχτεί τη διακοπή, η εφαρμογή μεταφέρεται στην κατάσταση Παρασκήνιο.

Ανάλογα με το τι κάνει ο χρήστης μέχρι να απαντήσει σε μια διακοπή, το σύστημα μπορεί να επιστρέψει στην εφαρμογή όταν η διακοπή τερματίσει. Για παράδειγμα αν ο χρήστης δεχθεί μια κλήση και στη συνέχεια την τερματίσει, το σύστημα ξανατρέχει την εφαρμογή. Αν κατά τη διάρκεια της κλήσης ο χρήστης επιστρέψει στην Home οθόνη ή εκτελέσει μια άλλη εφαρμογή, το σύστημα δεν θα επιστρέψει στην εφαρμογή.

Αν ο χρήστης πιέσει το sleep/wake κουμπί στη συσκευή καθώς τρέχει η εφαρμογή το σύστημα καλεί την delegate μέθοδο της εφαρμογής applicationWillResignActive και σταματά την παράδοση touch events ενώ ακολούθως κλειδώνει την οθόνη. Όταν ο χρήστης ξεκλειδώσει την οθόνη, το σύστημα καλεί την delegate μέθοδο applicationDidBecomeActive και αρχίζει την παράδοση γεγονότων ξανά στην εφαρμογή. Όσο η οθόνη παραμένει κλειδωμένη, οι εφαρμογές που βρίσκονται στο προσκήνιο και στο παρασκήνιο συνεχίζουν να εκτελούνται.

Επιστρέφοντας στη εκτέλεση στο προσκήνιο: Όταν ο χρήστης επιλέξει για εκτέλεση μια εφαρμογή που βρίσκεται στο παρασκήνιο, το σύστημα μετακινεί την εφαρμογή στην κατάσταση Μη ενεργή και στη συνέχεια στην κατάσταση Ενεργή. Αυτές οι ενέργειες καταλήγουν στην κλήση των delegate μεθόδων applicationWillEnterForeground και applicationWillBecomeActive. Όταν η εφαρμογή μετακινείται στο προσκήνιο ξαναρχίζει όλες τις υπηρεσίες που είχαν σταματήσει και ετοιμάζει τον εαυτό της για να χειρίζεται και πάλι γεγονότα.

Όταν η εφαρμογή είναι στην κατάσταση Σε αναστολή, το σύστημα εντοπίζει τα γεγονότα που στο σύνολό τους μπορεί να έχουν επίδραση σε αυτή την εφαρμογή όταν αυτή εκτελεστεί ξανά. Μόλις η εφαρμογή ξεκινήσει και πάλι να εκτελείται το σύστημα παραδίδει αυτά τα γεγονότα σε αυτήν. Για τα περισσότερα από αυτά τα γεγονότα η υπάρχουσα υποδομή της εφαρμογής πρέπει να αντιδρά κατάλληλα.

Αντιδρώντας στον τερματισμό μιας εφαρμογής: Παρόλο που οι εφαρμογές συνήθως μεταφέρονται στο παρασκήνιο και θέτονται σε αναστολή, αν κάποια από τις ακόλουθες συνθήκες είναι αληθής η εφαρμογή τερματίζεται και αφαιρείται από τη

μνήμα αντί να μεταφερθεί στο παρασκήνιο. Οι συνθήκες είναι: η εφαρμογή είναι συνδεδεμένη με έκδοση λογισμικού πριν από το iOS 4.0, η εφαρμογή τρέχει σε συσκευή με έκδοση iOS πριν από την 4.0, η τρέχουσα συσκευή δεν υποστηρίζει multitasking και η εφαρμογή περιλαμβάνει το κλειδί UIApplicationExistsOnSuspend στο αρχείο Info.plist.

Αν η εφαρμογή τρέχει (στο προσκήνιο ή στο παρασκήνιο) σε χρόνο τερματισμού τότε το σύστημα καλεί τη delegate μέθοδο applicationWillTerminate για να επιτευχθεί το απαραίτητο «καθάρισμα τιμών». Αυτή η μέθοδος μπορεί να χρησιμοποιηθεί για την αποθήκευση δεδομένων χρήστη ή πληροφορίας που αφορά την κατάσταση της εφαρμογής έτσι ώστε να μπορέσει να επιστρέψει σε αυτήν στη συνέχεια. Και αυτή η μέθοδος έχει στη διάθεσή της πέντε δευτερόλεπτα να επιστρέψει αλλιώς θα τερματιστεί και θα αφαιρεθεί από τη μνήμη. Η μέθοδος applicationWillTerminate δεν καλείται αν η εφαρμογή βρίσκεται στην κατάσταση Σε αναστολή.

Ακόμα και αν η ανάπτυξη εφαρμογών γίνεται με το iOS SDK 4 και μετά ο δημιουργός της εφαρμογής πρέπει να είναι προετοιμασμένος ότι η εφαρμογή μπορεί να τερματιστεί ανά πάσα στιγμή χωρίς προειδοποίηση. Ο χρήστης μπορεί να τερματίσει εφαρμογές χρησιμοποιώντας multitasking UI. Επιπλέον, αν η μνήμη περιοριστεί το σύστημα μπορεί να αφαιρέσει εφαρμογές από τη μνήμη για να ελευθερωθεί χώρος. Αν η εφαρμογή είναι σε αναστολή, το σύστημα σκοτώνει την εφαρμογή και την αφαιρεί από τη μνήμη χωρίς καμιά ειδοποίηση. Αν όμως η εφαρμογή τρέχει προς το παρόν στην κατάσταση Παρασκήνιο (δεν είναι δηλαδή σε αναστολή), το σύστημα καλεί την delegate μέθοδο applicationWillTerminate. Η εφαρμογή δεν μπορεί να ζητήσει επιπλέον χρόνο εκτέλεσης στο παρασκήνιο από τη μέθοδο αυτή.

Multitasking: Από την iOS 4.0 έκδοση και μετά πολλές εφαρμογές μπορούν να βρίσκονται στη μνήμη και να εκτελούνται ταυτόχρονα. Μόνο μια εφαρμογή όμως τρέχει στο προσκήνιο ενώ οι υπόλοιπες περιμένουν στο παρασκήνιο. Οι εφαρμογές

που τρέχουν σε τέτοιο περιβάλλον πρέπει να είναι σχεδιασμένες να χειρίζονται μεταβάσεις μεταξύ παρασκηνίου και προσκηνίου.

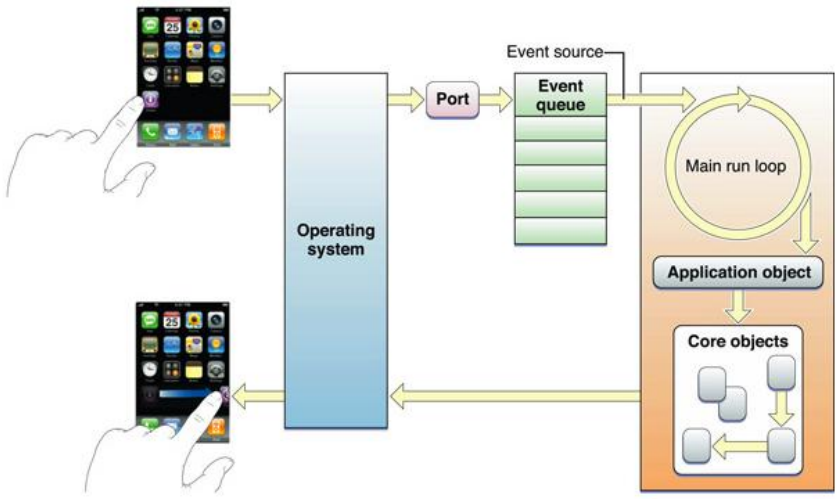
Απαντώντας σε αλλαγές συστήματος ενώ βρίσκεται στο παρασκήνιο: Όταν μια εφαρμογή βρίσκεται σε κατάσταση αναστολής δεν λαμβάνει γεγονότα που σχετίζονται με το σύστημα και έχουν κάποιο ενδιαφέρον. Όμως τα πιο σχετικά γεγονότα συλλαμβάνονται από το σύστημα και μπαίνουν στην ουρά για να παραδοθούν αργότερα στην εφαρμογή. Για την αποφυγή υπερφόρτωσης της εφαρμογής από ειδοποιήσεις όταν αυτή ξαναρχίσει, το σύστημα συγκεντρώνει τα γεγονότα και παραδίδει ένα μόνο γεγονός που αντιστοιχεί στην αλλαγή του δικτύου από τη στιγμή που η εφαρμογή τέθηκε σε αναστολή. Στις περισσότερες περιπτώσεις τα γεγονότα παραδίδονται με τη μορφή ενός αντικειμένου ειδοποίησης. Κάποια γεγονότα όμως μπορεί να «παρεξηγηθούν» από την αρχιτεκτονική του συστήματος και να παραδοθούν στην εφαρμογή με άλλο τρόπο. Εκτός και αν υπάρξει οποιαδήποτε τροποποίηση, όλα τα γεγονότα παραδίδονται ανεξάρτητα από το αν η εφαρμογή ξαναρχίζει στο προσκήνιο ή στο παρασκήνιο.

Όταν η εφαρμογή ξαναρχίζει, όλα τα γεγονότα που περιμένουν στη σειρά παραδίδονται μέσω του κύριου βρόχου της εφαρμογής. Επειδή τα γεγονότα αυτά μπαίνουν στη σειρά αμέσως, τυπικά παραδίδονται πριν από touch events ή άλλες εισόδους του χρήστη. Οι περισσότερες εφαρμογές πρέπει να είναι ικανές να χειριστούν αυτά τα γεγονότα αρκετά γρήγορα έτσι ώστε να μην προκαλέσουν σημαντική καθυστέρηση με την επανεκκίνηση. Αν όμως η εφαρμογή δυσκολεύεται στην απάντηση προς την είσοδο του χρήστη όταν ξαναρχίσει πρέπει να γίνει έλεγχος του κατά πόσο ο κώδικας του χρήστη προκαλεί την καθυστέρηση.

Το σύστημα χειρισμού γεγονότων: Στο iOS το σύστημα χειρισμού γεγονότων είναι υπεύθυνο για τον εντοπισμό touch και motion events και την παράδοσή τους στην εφαρμογή. Τα περισσότερα γεγονότα παραδίδονται στην εφαρμογή μέσω του UIApplication object το οποίο διαχειρίζεται τη σειρά των εισερχόμενων γεγονότων και τα μεταφέρει σε άλλα τμήματα της εφαρμογής. Ο πιο σημαντικός τύπος γεγονότων

που είναι δυνατό να παραληφθούν είναι τα touch αλλά και άλλοι τύποι γεγονότων μπορούν επίσης να παραχθούν και να παραδοθούν.

Όταν το σύστημα εκτελέσει μια εφαρμογή δημιουργεί τόσο διεργασία αλλά και νήμα για την εφαρμογή. Αυτό το αρχικό νήμα γίνεται το κύριο νήμα της εφαρμογής. Σε αυτό, το UIApplication object θέτει τον κύριο βρόχο εκτέλεσης και επιβεβαιώνει τον κώδικα χειρισμού γεγονότων που περιέχει όπως φαίνεται και στην εικόνα 5.4. Καθώς τα γεγονότα touch εισέρχονται στην εφαρμογή, μπαίνουν στην ουρά μέχρι η εφαρμογή να είναι σε θέση να τα επεξεργαστεί. Η εφαρμογή επεξεργάζεται τα γεγονότα στον κύριο βρόχο εκτέλεσης για να διασφαλίσει ότι χειρίζονται ακολουθιακά με τη σειρά που φτάνουν. Ο πραγματικό χειρισμός ενός δεδομένου γεγονότος συνήθως προκύπτει σε άλλα αντικείμενα όπως είναι τα application views και τα view controllers.



Εικόνα 5. 4 Επεξεργασία γεγονότων στον κύριο βρόχο εκτέλεσης

Ο βρόχος εκτέλεσης παρακολουθεί τις πηγές εισόδου για ένα δεδομένο νήμα εκτέλεσης. Η ουρά γεγονότων της εφαρμογής αναπαριστά μια από αυτές τις πηγές εισόδου. Όταν φτάσει ένα γεγονός, ο βρόχος εκτέλεσης ξυπνά (weak up) το νήμα και δίνει τον έλεγχο στο σχετικό χειριστή. Στην περίπτωση touch events χειριστής είναι το UIApplication object. Όταν ο χειριστής τελειώσει, ο έλεγχος περνά πίσω στο βρόχο

εκτέλεσης ο οποίος επεξεργάζεται στη συνέχεια κι άλλο γεγονός, άλλες πηγές εισόδου ή θέτει το νήμα σε κατάσταση sleep αν δεν υπάρχει κάτι άλλο να κάνει.

Τα περισσότερα γεγονότα που στέλνονται σε μια εφαρμογή ενθυλακώνονται σε ένα event object που είναι στιγμιότυπο της κλάσης UIEvent. Σε περίπτωση γεγονότων που σχετίζονται με ενέργειες αφής, το event object περιλαμβάνει ένα ή περισσότερα touch objects που αναπαριστούν τα δάκτυλα που αγγίζουν την οθόνη. Καθώς ο χρήστης τοποθετεί τα δάκτυλά του στην οθόνη και τα μετακινεί, το σύστημα καταγράφει τις αλλαγές από κάθε δάκτυλο στο αντίστοιχο touch object.

Ο διαμοιρασμός και η διαχείριση των γεγονότων είναι υπευθυνότητα των responder objects τα οποία είναι στιγμιότυπα της κλάσης UIResponder. Οι κλάσεις UIViewController, UIApplication, UIWindow και UIView βρίσκονται όλες από κάτω της UIResponder. Όταν ένα γεγονός βγει από την ουρά γεγονότων, η εφαρμογή ελευθερώνει το γεγονός στο UIWindow object στο οποίο συνέβηκε. Το window object με τη σειρά του, προωθεί το γεγονός στον πρώτο ο οποίος θα απαντήσει σε αυτό. Σε περίπτωση touch events, ο πρώτος που θα απαντήσει είναι συνήθως το view object στο οποίο συνέβηκε το γεγονός.

Αν ο πρώτος που θα απαντήσει δεν μπορεί να χειριστεί το γεγονός, το προωθεί στον επόμενο διαθέσιμο που είναι συνήθως η όψη-γονέας ή ο view controller. Αν και αυτό το αντικείμενο δεν μπορεί να χειριστεί το γεγονός, το προωθεί στον επόμενο κ.ο.κ. μέχρι το γεγονός να τύχει χειρισμού. Αυτή η ακολουθία συνδεδεμένων responder objects ονομάζεται responder chain. Τα μηνύματα συνεχίζουν να ταξιδεύουν στην αλυσίδα προς υψηλότερου επιπέδου responder objects όπως είναι το παράθυρο, η εφαρμογή κτλ μέχρι το χειρισμό του γεγονότος. Αν το γεγονός δεν βρει κανένα χειριστή τότε απορρίπτεται.

Το σύστημα Γραφικών και Σχεδιασμού: Υπάρχουν δύο βασικοί τρόποι με τους οποίους μια iOS εφαρμογή μπορεί να σχεδιάσει το περιεχόμενό της. Ο πρώτος είναι να χρησιμοποιήσει «ντόπιες» σχεδιαστικές τεχνολογίες όπως το Core Graphics και το UIKit και ο δεύτερος είναι να χρησιμοποιήσει OpenGL ES.

Οι υπάρχουσες σχεδιαστικές τεχνολογίες στηρίζονται στην υποδομή που παρέχεται από τις όψεις και τα παράθυρα για τη διανομή και σχεδιασμό του περιεχομένου. Όταν μια όψη εμφανίζεται για πρώτη φορά το σύστημα την ρωτά για το σχεδιασμό του περιεχομένου. Οι όψεις του συστήματος σχεδιάζουν τα περιεχόμενά τους αυτόματα αλλά οι όψεις των εφαρμογών του χρήστη πρέπει να υλοποιούν τη μέθοδο drawRect. Μέσα σε αυτή τη μέθοδο χρησιμοποιούνται οι υπάρχουσες σχεδιαστικές τεχνολογίες για το σχεδιασμό σχημάτων, εικόνων κειμένου και άλλο οπτικό περιεχόμενο.

Όταν χρησιμοποιείται το OpenGL ES για το σχεδιασμό του περιεχομένου της εφαρμογής, ο δημιουργός της εφαρμογής πρέπει να δημιουργήσει ένα παράθυρο και μια όψη για το χειρισμό του περιεχομένου όμως αυτά τα αντικείμενα απλά παρέχουν την επιφάνεια για το OpenGL σχεδιαστικό περιεχόμενο. Όταν το περιεχόμενο αυτό είναι διαθέσιμο, η εφαρμογή είναι υπεύθυνη για την αρχικοποίηση των σχεδιαστικών ενημερώσεων ανά προκαθορισμένα τακτά χρονικά διαστήματα.

Το σύστημα κειμένου: Στο iOS το σύστημα κειμένου παρέχει ό,τι απαιτείται για τη λήψη εισόδου από το χρήστη και την εμφάνιση του κειμένου στην εφαρμογή. Από την πλευρά της εισόδου, το σύστημα χειρίζεται το εισερχόμενο κείμενο από το πληκτρολόγιο του συστήματος το οποίο συνδέεται με το πρώτο responder object. Παρόλο που αυτό αναφέρεται ως πληκτρολόγιο, η εμφάνισή του δεν μοιάζει πάντα με ένα παραδοσιακό πληκτρολόγιο. Διαφορετικές γλώσσες έχουν διαφορετικές απαιτήσεις εισόδου κειμένου και για το λόγο αυτό το πληκτρολόγιο προσαρμόζεται ανάλογα για την υποστήριξή τους.

Οι εφαρμογές που επιθυμούν να αντικαταστήσουν το πληκτρολόγιο μπορούν να το κάνουν χρησιμοποιώντας μια τροποποιημένη όψη εισόδου. Η εμφάνιση του πληκτρολογίου πυροδοτείται από ένα αντικείμενο το οποίο γίνεται και ο πρώτος που θα απαντήσει σε αυτό. Αν ανατεθεί σε ένα responder object μια τροποποιημένη όψη εισόδου, τότε αυτή η όψη θα εμφανίζεται.

Οι εφαρμογές που αφορούν την εμφάνιση κειμένου έχουν αρκετές επιλογές. Για απλή εμφάνιση κειμένου και επεξεργασία μπορούν να χρησιμοποιηθούν οι κλάσεις UILabel, UITextField και UITextView. Επίσης μπορεί να γίνει απλός σχεδιασμός αλφαριθμητικών χρησιμοποιώντας επεκτάσεις της κλάσης NSString που παρέχονται από το UIKit. Για ακόμα πιο εξειδικευμένη εμφάνιση μπορεί να χρησιμοποιηθούν οι Core Graphics υπηρεσίες κειμένου ή το Core Text. Και τα δύο framework παρέχουν σχεδιαστικές αρχές για την εμφάνιση κειμένου. Επιπλέον, το Core Text framework παρέχει μια εξειδικευμένη μηχανή εμφάνισης για υπολογισμό της θέσης των γραμμών και πληροφοριών που έχουν να κάνουν με τη μορφή της οθόνης.

Υποστήριξη ήχου και βίντεο: Για εφαρμογές που χρησιμοποιούν ήχο και βίντεο το iOS παρέχει αρκετές τεχνολογίες για την υποστήριξη των αναγκών του χρήστη. Για αναπαραγωγή βίντεο μπορεί να χρησιμοποιηθεί το Media Player framework και το AV Foundation framework. Για αναπαραγωγή ήχου μπορούν να χρησιμοποιηθούν τα ίδια frameworks αλλά και τα Core Audio και OpenAL frameworks. Με τη χρήση αυτών των frameworks μπορούν να υλοποιηθούν χαρακτηριστικά όπως: υψηλής ποιότητας ηχογράφηση, αναπαραγωγή και streaming, live voice chat, αναπαραγωγή περιεχομένου από τη βιβλιοθήκη του iPod, πλήρες και μερικής οθόνης αναπαραγωγή βίντεο και βιντεογράφηση βίντεο σε συσκευές που το υποστηρίζουν.

Κεφάλαιο 6

Διαχείριση συστήματος στο Google Android

Στο κεφάλαιο αυτό όπως και στο προηγούμενο, θα γίνει αναφορά στη διαχείριση συστήματος στο Android OS. Γίνεται παρουσίαση των τεχνολογιών διαχείρισης μνήμης καθώς και της πορείας ανάπτυξης των εφαρμογών. Επίσης παρουσιάζονται τα κυριότερα συστατικά της κάθε εφαρμογής καθώς και τρόποι ενεργοποίησης αυτών.

6.1 Διαχείριση Μνήμης

Όπως έχει αναφερθεί και στο κεφάλαιο 3, το Android είναι μια στοίβα λογισμικού για κινητές συσκευές που περιλαμβάνει λειτουργικό σύστημα, middleware και εφαρμογές. Το Android SDK παρέχει τα εργαλεία και τα πρωτόκολλα που είναι απαραίτητα για την ανάπτυξη εφαρμογών στην πλατφόρμα Android χρησιμοποιώντας τη γλώσσα προγραμματισμού Java. Όλες οι βασικές πράξεις, ανάμεσά τους και η διαχείριση μνήμης, χειρίζονται από τον πυρήνα Linux του Android.

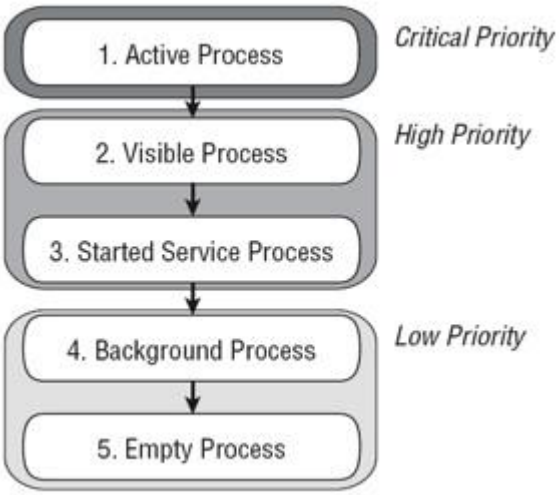
Το Android επεξεργάζεται και διαχειρίζεται τη μνήμη με κάπως ασυνήθιστο τρόπο. Όπως η Java και η .Net έτσι και το Android χρησιμοποιεί το δικό του χρόνο εκτέλεσης και τη δική του εικονική μηχανή για να διαχειρίζεται τη μνήμη των εφαρμογών. Όμως σε αντίθεση με τις δύο άλλες πλατφόρμες που αναφέρθηκαν, στο χρόνο εκτέλεσης του Android διαχειρίζεται και ο κύκλος ζωής των εφαρμογών. Διασφαλίζει την ανταπόκριση της εφαρμογής σταματώντας και «σκοτώνοντας»

διεργασίες αν το κρίνει αναγκαίο για να ελευθερώσει πόρους που είναι απαραίτητοι για τις εφαρμογές υψηλής προτεραιότητας [24].

Κάθε Android εφαρμογή τρέχει σε ξεχωριστή διεργασία, με το δικό της Dalvik στιγμιότυπο, δίνοντας όλη την υπευθυνότητα για μνήμη και διαχείριση επεξεργασίας στον χρόνο εκτέλεσης του Android. Ο χρόνος εκτέλεσης του Android και του Dalvik βρίσκεται επάνω από τον πυρήνα που χειρίζεται τις χαμηλού επιπέδου αλληλεπιδράσεις στις οποίες συμπεριλαμβάνεται και η διαχείριση μνήμης. Αυτό επιτυγχάνεται μέσω ένα σύνολο πρωτοκόλλων που παρέχουν πρόσβαση σε όλες τις υπηρεσίες χαμηλότερου επιπέδου, τα χαρακτηριστικά και το υλικό.

Έχει ήδη παρουσιαστεί σε προηγούμενο κεφάλαιο η Dalvik Virtual Machine. Πρόκειται για μια εικονική μηχανή που είναι βασισμένη στους καταχωρητές και έχει βελτιστοποιηθεί έτσι ώστε να είναι ικανή να τρέξει πολλαπλά στιγμιότυπα ταυτόχρονα. Βασίζεται στον πυρήνα για threading και διαχείριση μνήμης. Η πρόσβαση στο υλικό και στις υπηρεσίες συστήματος διαχειρίζεται μέσω της Dalvik που λειτουργεί ως ενδιάμεσο επίπεδο. Χρησιμοποιώντας μια εικονική μηχανή για να φιλοξενήσει την εκτέλεση μιας εφαρμογής, ο δημιουργός μιας εφαρμογής δε χρειάζεται να ανησυχεί σχετικά με υλοποιήσεις που έχουν να κάνουν με το υλικό.

Η σειρά με την οποία οι διεργασίες τερματίζονται για την απόκτηση πόρων καθορίζεται από την προτεραιότητα της κάθε εφαρμογής που εκτελείται (εικόνα 6.1). Η προτεραιότητα μιας εφαρμογής ισούται με αυτή του συστατικού της που έχει τη μεγαλύτερη προτεραιότητα από τα υπόλοιπα. Όταν δύο εφαρμογές έχουν την ίδια προτεραιότητα, η διεργασία που βρισκόταν σε χαμηλότερη προτεραιότητα για μεγαλύτερο χρονικό διάστημα τερματίζεται πρώτη. Η προτεραιότητα των διεργασιών επηρεάζεται επίσης από εξαρτήσεις που πιθανόν να υπάρχουν ανάμεσα στις διεργασίες. Αν μια διεργασία έχει εξάρτηση από την Service ή τον Content Provider που παρέχει μια άλλη εφαρμογή τότε η εφαρμογή που προσφέρει την εξάρτηση πρέπει να έχει τουλάχιστον τόσο ψηλή προτεραιότητα όσο η εφαρμογή την οποία υποστηρίζει.



Εικόνα 6. 1 Οι διάφορες μορφές προτεραιότητας που είναι δυνατό να έχει κάθε διεργασία σε κάθε μια από τις καταστάσεις της

Όλες οι Android εφαρμογές παραμένουν σε εκτέλεση και στη μνήμη μέχρι το σύστημα να χρειαστεί τους πόρους για άλλες εφαρμογές. Είναι σημαντικό η εφαρμογή να είναι δομημένη με τέτοιο τρόπο έτσι ώστε να διασφαλίζεται ότι η προτεραιότητα είναι η κατάλληλη για τη λειτουργία η οποία εκτελείται. Αν αυτό δε συμβαίνει τότε η εφαρμογή είναι δυνατό να τερματιστεί καθώς βρίσκεται εν εξέλιξη κάνοντας κάτι σημαντικό.

Το Android διαχειρίζεται τις «ανοικτές» εφαρμογές που τρέχουν στο παρασκήνιο και γι' αυτό ο χρήστης δε χρειάζεται να ανησυχεί για αυτό. Αυτό σημαίνει ότι οι εφαρμογές τερματίζονται όταν το σύστημα χρειάζεται επιπλέον μνήμη. Υπάρχουν βέβαια χρήστες που δεν είναι ικανοποιημένοι με αυτό το χειρισμό επειδή υπάρχει το σύστημα να επιτρέπει τη συνεχή εκτέλεση αρκετών εφαρμογών προκαλώντας καθυστέρηση στη χρήση της συσκευής. Έτσι από την πλευρά του ο χρήστης έχει τη δυνατότητα να χρησιμοποιήσει ένα task killer ή task manager που κυκλοφορούν στο Android Market για την καλύτερη διαχείριση των εφαρμογών που εκτελούνται.

Οι Android εφαρμογές που εκτελούνται τουλάχιστον στο T-Mobile G1 περιορίζονται σε 16MB μέγεθος σωρού [25]. Πρόκειται για πολλή μνήμη για ένα τηλέφωνο αλλά για ελάχιστη για κάποιους που θέλουν να αναπτύξουν εντυπωσιακές

εφαρμογές. Ακόμα και αν κάποιος δε σκοπεύει να χρησιμοποιήσει όλη αυτή τη μνήμη για την εφαρμογή του, πρέπει να δεσμεύσει όσο το δυνατόν λιγότερη έτσι ώστε άλλες εφαρμογές να έχουν την ευχέρεια να εκτελεστούν χωρίς να προκληθεί ο τερματισμός τους.

Όσο πιο πολλές εφαρμογές μπορούν να κρατηθούν στη μνήμη του Android τόσο πιο γρήγορα θα εναλλάσσεται ο χρήστης μεταξύ των εφαρμογών. Υπάρχουν βέβαια και περιπτώσεις memory leaks που συνήθως οφείλονται στη διατήρηση για πολύ χρόνο αναφοράς σε ένα Context. Στο Android, ένα context χρησιμοποιείται σε πολλές λειτουργίες αλλά κυρίως για τη φόρτωση και πρόσβαση σε πόρους. Με τον τρόπο αυτό όλα τα widgets λαμβάνουν την παράμετρο context στον κατασκευαστή τους. Σε μια κανονική Android εφαρμογή συνήθως υπάρχουν δύο ειδών context: Activity και Application. Συνήθως ο χρήστης περνά το πρώτο σε κλάσεις και μεθόδους που χρειάζονται context.

Αυτό σημαίνει ότι οι όψεις έχουν αναφορές σε συγκεκριμένη δραστηριότητα και ως επακόλουθο σε ό,τι έχει να κάνει με τη δραστηριότητα αυτή που είναι συνήθως ολόκληρη η ιεραρχία των όψεων και οι πηγές της. Για το λόγο αυτό αν υπάρξει leak ως προς το context, με την έννοια ότι κρατείται μια αναφορά σε αυτό που αποτρέπει τον GC να το διαγράψει, χάνεται μεγάλη ποσότητα μνήμης. Απαιτείται λοιπόν προσοχή γιατί είναι εύκολη η δημιουργία αλυσίδας από leaked contexts η οποία καταναλώνει τη διαθέσιμη μνήμη πολύ γρήγορα προκαλώντας προβλήματα.

Υπάρχουν δύο τρόπο για αποφυγή memory leaks που έχουν να κάνουν με contexts. Ο πιο προφανής είναι η αποφυγή διαφυγής του context εκτός της περιοχής του. Με άλλα λόγια οι αναφορές από εσωτερικές προς εξωτερικές κλάσεις μπορούν να προκαλέσουν leak και είναι επικίνδυνες στη χρήση. Ο δεύτερος τρόπος είναι η χρήση του Application context. Αυτό το context θα ζήσει όσο διαρκεί η εφαρμογή και δεν εξαρτάται από τις δραστηριότητες του κύκλου ζωής. Αν λοιπόν ο χρήστης επιθυμεί να κρατήσει για μεγάλο χρονικό διάστημα «ζωντανά» κάποια αντικείμενα που απαιτούν context πρέπει να στραφεί προς το application.

Συμπεραίνει λοιπόν κανείς ότι για να αποφύγει τα memory leaks που αφορούν contexts πρέπει να θυμάται τα ακόλουθα: να μην διατηρεί μακροχρόνιες αναφορές σε context που αφορούν δραστηριότητα. Μια αναφορά σε μια δραστηριότητα πρέπει να έχει την ίδια διάρκεια ζωής με τη δραστηριότητα. Είναι προτιμότερη η χρήση context-application παρά context-activity. Πρέπει να αποφεύγονται οι μη στατικές εσωτερικές κλάσεις σε μια δραστηριότητα αν δεν ελέγχει ο χρήστης τον κύκλο ζωής τους. Τέλος, ένας Garbage Collector δεν αποτελεί ασφάλεια σε ό,τι αφορά τα memory leaks.

Πολύ σημαντικές στη διαχείριση μνήμης είναι και οι κατανομές της. Σε ένα κώδικα που παρουσιάζει ευαισθησία ως προς την απόδοση όπως για παράδειγμα μια σχεδιαστική μέθοδος ή μια όψη ή ο κώδικας της λογικής ενός παιχνιδιού, η κατανομή της μνήμης έχει κάποιο κόστος [26]. Μετά από αρκετές κατανομές, ο GC θα τερματίσει την εφαρμογή με σκοπό την απελευθέρωση μνήμης. Τον περισσότερο χρόνο, ο GC ενεργεί πολύ γρήγορα και ο χρήστης δεν το καταλαβαίνει. Όταν όμως ενεργήσει κατά τη διάρκεια που ο χρήστης περιηγείται σε μια συλλογή αντικειμένων ή καθώς ασχολείται με ένα παιχνίδι, αντιλαμβάνεται άμεσα τη μείωση της απόδοσης στην αντίστοιχη εφαρμογή. Συνήθως ένας GC παίρνει περίπου 100-200 ms για μια συλλογή (collection).

Τις περισσότερες φορές ο GC παρεμβαίνει λόγω πολλών, μικρών, σύντομων σε ζωή αντικειμένων. Υπάρχουν και κάποιοι GC όπως οι generational GC που βελτιστοποιούν τη συλλογή αυτών των αντικειμένων έτσι ώστε η εφαρμογή να μη διακόπτεται πολύ συχνά. Ο GC του Android δεν μπορεί να υποστεί τέτοιες βελτιστοποιήσεις και η δημιουργία σύντομων σε ζωή αντικειμένων σε σημαντικά τμήματα κώδικα όσον αφορά την απόδοση είναι καταστροφική και κοστίζει για την εφαρμογή.

Για την αποφυγή συχνών GC το Android SDK περιλαμβάνει το εργαλείο Allocation Tracker. Το εργαλείο αυτό αποτελεί μέρος του DDMS το οποίο χρησιμοποιείται για debugging. Για τη χρήση του allocation tracker απαιτείται η εγκατάσταση και εκτέλεση του DDMS. Καθώς το DDMS εκτελείται, ο χρήστης μπορεί

να επιλέξει τη διεργασία και να διαλέξει την καρτέλα Allocation tracker. Στη νέα όψη που εμφανίζεται ο χρήστης επιλέγει start tracking και στη συνέχεια χρησιμοποιεί την εφαρμογή του για την εκτέλεση του κώδικα τον οποίο θέλει να αναλύσει. Επιλέγοντας ακολούθως get allocations θα πάρει μια λίστα με τα allocated objects. Έτσι ανά πάσα στιγμή ο χρήστης γνωρίζει τι τύπου αντικείμενο είναι allocated, σε ποιο νήμα, κλάση, αρχείο και γραμμή.

Αν και δεν είναι πάντα απαραίτητο ή πιθανή η απομάκρυνση όλων των allocations, το εργαλείο αυτό βοηθά στην αναγνώριση σημαντικών θεμάτων στον κώδικα της εφαρμογής. Έτσι ο χρήστης μπορεί να εντοπίσει εύκολα περιπτώσεις στις οποίες δεν ήταν απαραίτητη η δημιουργία νέων allocated objects τη στιγμή που υπήρχαν άλλα, διαθέσιμα προς επαναχρησιμοποίηση.

6.2 Πορεία ανάπτυξης εφαρμογών για το Google’s Android

Οι Android εφαρμογές είναι γραμμένες σε γλώσσα προγραμματισμού Java. Τα εργαλεία του Android SDK μεταγλωττίζουν τον κώδικα αλλά και τα δεδομένα που τον συνοδεύουν μαζί με αρχεία που περιέχουν πόρους για την εφαρμογή σε ένα Android πακέτο με κατάληξη .apk. Όλος ο κώδικας σε ένα μόνο .apk αρχείο θεωρείται ως μιας εφαρμογή και είναι το αρχείο το οποίο οι Android συσκευές χρησιμοποιούν για την εγκατάσταση της εφαρμογής [27].

Από τη στιγμή που μια εφαρμογή Android θα εγκατασταθεί σε μια συσκευή, λειτουργεί στο δικό της security sandbox:

- Το λειτουργικό σύστημα Android είναι ένα Linux σύστημα το οποίο υποστηρίζει πολλούς χρήστες όπου κάθε εφαρμογή είναι και ένας διαφορετικός χρήστης.
- Είναι προκαθορισμένο έτσι ώστε το σύστημα να αναθέτει στην κάθε εφαρμογή μια μοναδική Linux User ταυτότητα που χρησιμοποιείται μόνο από το σύστημα και παραμένει άγνωστη στην εφαρμογή. Το σύστημα

θέτει δικαιώματα για όλα τα αρχεία μιας εφαρμογής έτσι ώστε μόνο η ταυτότητα του χρήστη που έχει ανατεθεί σε αυτή την εφαρμογή να μπορεί έχει πρόσβαση σε αυτήν.

- Κάθε διεργασία έχει τη δική της εικονική μηχανή έτσι ώστε ο κώδικας κάθε εφαρμογής να τρέχει σε απομόνωση από τις άλλες εφαρμογές..
- Είναι προκαθορισμένο έτσι ώστε η κάθε εφαρμογή να τρέχει τη δική της Linux διεργασία. Το Android αρχίζει τη διεργασία όταν οποιοδήποτε από τα συστατικά της εφαρμογής πρέπει να εκτελεστεί και στη συνέχεια σταματά τη διεργασία όταν πλέον δε χρειάζεται πουθενά ή όταν το σύστημα πρέπει να εξοικονομήσει μνήμη για άλλες εφαρμογές.

Με τον τρόπο αυτό το σύστημα Android υλοποιεί την αρχή των «λιγότερων δυνατών προνομίων» (The principle of least privilege). Έτσι, κάθε εφαρμογή έχει πρόσβαση μόνο στα συστατικά που απαιτεί για τη λειτουργία της και σε κανένα άλλο. Δημιουργείται λοιπόν ένα ασφαλισμένο περιβάλλον στο οποίο η εφαρμογή δεν μπορεί να πλησιάσει και να έχει πρόσβαση σε μέρη του συστήματος για τα οποία δεν της έχουν δοθεί δικαιώματα.

Υπάρχει όμως πάντα ο τρόπος με τον οποίο μια εφαρμογή μπορεί να διαμοιραστεί δεδομένα με άλλες εφαρμογές αλλά και να έχει πρόσβαση σε υπηρεσίες του συστήματος. Είναι δυνατόν να κανονιστεί δύο εφαρμογές να διαμοιράζονται την ίδια Linux user ταυτότητα έτσι ώστε να έχει η μια πρόσβαση στα αρχεία της άλλης. Για την εξοικονόμηση πόρων του συστήματος οι εφαρμογές με την ίδια ταυτότητα μπορούν να τρέχουν στην ίδια Linux διεργασία και να διαμοιράζονται την ίδια εικονική μηχανή. Επίσης, μια εφαρμογή μπορεί να ζητήσει άδεια για πρόσβαση στα δεδομένα μιας συσκευής όπως είναι οι επαφές, τα εισερχόμενα μηνύματα, η κάρτα SD, η κάμερα, το Bluetooth κτλ. Όλα τα δικαιώματα της εφαρμογής πρέπει να δοθούν από το χρήστη κατά τη διάρκεια εγκατάστασής της.

Πριν το Android σύστημα ξεκινήσει ένα συστατικό μιας εφαρμογής πρέπει να γνωρίζει ότι το συστατικό υπάρχει. Αυτό το εξακριβώνει διαβάζοντας το AndroidManifest.xml αρχείο της εφαρμογής. Η εφαρμογή πρέπει να δηλώσει όλα της τα συστατικά σε αυτό το αρχείο το οποίο πρέπει να βρίσκεται στη ρίζα του φακέλου που περιέχει το project της εφαρμογής. Το manifest κάνει αρκετά πράγματα εκτός από το να δηλώνει τα συστατικά μιας εφαρμογής. Αναγνωρίζει τα δικαιώματα χρήστη που απαιτεί η εφαρμογή όπως την πρόσβαση στο διαδίκτυο ή την ανάγνωση των επαφών. Επίσης δηλώνει το ελάχιστο επίπεδο πρωτοκόλλου που απαιτείται από την εφαρμογή βασισμένο στο ποια πρωτόκολλα χρησιμοποιεί η εφαρμογή. Ακόμα, καθορίζει τα χαρακτηριστικά του υλικού και λογισμικού που χρησιμοποιούνται ή απαιτούνται από την εφαρμογή όπως η κάμερα, οι υπηρεσίες Bluetooth κτλ. Τέλος, οι βιβλιοθήκες πρωτοκόλλων με τις οποίες πρέπει να συνδεθεί η εφαρμογή καθορίζονται στο manifest εκτός από τα Android framework APIs.

Ο κύριος λόγος ύπαρξης του manifest είναι να πληροφορήσει το σύστημα σχετικά με τα συστατικά της εφαρμογής. Για παράδειγμα ένα manifest αρχείο μπορεί να δηλώσει μια δραστηριότητα όπως φαίνεται και στην εικόνα 6.2.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest ... >
    <application android:icon="@drawable/app_icon.png" ... >
        <activity android:name="com.example.project.ExampleActivity"
                  android:label="@string/example_label" ... >
        </activity>
        ...
    </application>
</manifest>
```

Εικόνα 6. 2 Τμήμα ενός manifest αρχείου εφαρμογής για Android

Το στοιχείο <application> αποτελεί τη δήλωση της εφαρμογής. Περιλαμβάνει υπό-στοιχεία που δηλώνουν κάθε ένα από τα συστατικά της εφαρμογής και έχει χαρακτηριστικά που επηρεάζουν όλα τα συστατικά. Πολλά από αυτά τα

χαρακτηριστικά θέτουν προκαθορισμένες τιμές για τα αντίστοιχα χαρακτηριστικά των στοιχείων των συστατικών ενώ άλλα θέτουν τιμές για την εφαρμογή ως σύνολο και δεν μπορούν να επανακαθοριστούν από τα συστατικά. Το android:icon χαρακτηριστικό δείχνει την πηγή της εικόνας η οποία χαρακτηρίζει την εφαρμογή.

Το στοιχείο <activity> δηλώνει μια δραστηριότητα (μια υποκλάση του Activity) που υλοποιεί μέρος της γραφικής διασύνδεσης της εφαρμογής. Όλες οι δραστηριότητες πρέπει να αναπαριστούνται από <activity> στοιχεία στο manifest. Αν κάποια δε δηλώνεται στο αρχείο αυτό με αυτό τον τρόπο, τότε δεν θα γίνει αντιληπτή από το σύστημα και δεν θα εκτελεστεί ποτέ. Το χαρακτηριστικό android:name καθορίζει το πλήρες ορισμένο όνομα κλάσης της υποκλάσης της Activity ενώ το android:label χαρακτηριστικό καθορίζει ένα αλφαριθμητικό που χρησιμοποιείται ως ορατή ετικέτα για τη δραστηριότητα.

Με τον τρόπο αυτό πρέπει να δηλωθούν όλα τα συστατικά της εφαρμογής. <activity> στοιχεία για τις δραστηριότητες, <service> στοιχεία για τις υπηρεσίες, <receiver> στοιχεία για τους broadcast receivers και <provider> στοιχεία για τους παροχείς περιεχομένου.

Το στοιχείο <service> δηλώνει μια υπηρεσία ως ένα συστατικό της εφαρμογής. Αντίθετα με τις δραστηριότητες, οι υπηρεσίες δεν παρουσιάζουν γραφική διασύνδεση χρήστη. Χρησιμοποιούνται για την υλοποίηση μακροπρόθεσμων παρασκηνιακών λειτουργιών ή σύνθετα επικοινωνιακά πρωτόκολλα που καλούνται από άλλες εφαρμογές.

Το στοιχείο <receiver> δηλώνει ένα broadcast receiver ως ένα από τα συστατικά της εφαρμογής. Επιτρέπουν στις εφαρμογές να λαμβάνουν Intents που ανακοινώνονται από το σύστημα ή από άλλες εφαρμογές ακόμα και όταν άλλα συστατικά της εφαρμογής δεν εκτελούνται. Υπάρχουν δύο τρόποι για να γίνει ένας broadcast receiver γνωστός στο σύστημα. Ο πρώτος είναι να δηλωθεί στο manifest αρχείο με αυτό το στοιχείο. Ο δεύτερος είναι η δημιουργία του receiver Δυναμικά μέσω κώδικα και η καταχώρησή του με μια κατάλληλη μέθοδο.

Το στοιχείο <provider> δηλώνει έναν content provider που προμηθεύει δομημένη πρόσβαση στα δεδομένα που χειρίζεται η εφαρμογή. Όλοι οι Content providers που είναι μέρος μιας εφαρμογής πρέπει να αναπαριστώνται με αυτό το στοιχείο στο manifest αρχείο. Το σύστημα δεν μπορεί να δει και γι' αυτό δεν θα εκτελέσει contented provider ο οποίος δεν έχει δηλωθεί. Ο δημιουργός μιας εφαρμογής υποχρεούται να δηλώσει τους Content providers που έχει αναπτύξει και όχι αυτούς που έχουν αναπτυχθεί από άλλους και χρησιμοποιούνται στην εφαρμογή.

Όπως θα εξηγηθεί με περισσότερες λεπτομέρειες στην υπό-ενότητα 6.3, ένα Intent Μπορεί να χρησιμοποιηθεί για την εκκίνηση δραστηριοτήτων, υπηρεσιών και broadcast receives. Αυτό συμβαίνει ονομάζοντας απλώς το συστατικό-στόχο μέσα στην intent. Στην πραγματικότητα, η κύρια δύναμη των Intents έγκειται στο γεγονός ότι υπάρχουν ενέργειες (intent actions). Με τις ενέργειες αυτές, ο χρήστης μπορεί απλά να περιγράψει τον τύπο της ενέργειας που θέλει να εκτελέσει καθώς και τα δεδομένα στα οποία θα εφαρμοστεί και ακολούθως να αφήσει το σύστημα να βρει το συστατικό της συσκευής το οποίο θα εκτελέσει την ενέργεια. Αν υπάρχουν πολλά συστατικά που μπορούν να εκτελέσουν την ενέργεια τότε ο χρήστης επιλέγει ποιο προτιμά.

Ο τρόπος με τον οποίο το σύστημα αναγνωρίζει τα συστατικά τα οποία μπορούν να απαντήσουν σε ένα Intent παρέχεται στο manifest αρχείο άλλων εφαρμογών της συσκευής. Όταν ο χρήστης δηλώσει ένα συστατικό στο manifest αρχείο της εφαρμογής του μπορεί προαιρετικά να εισάγει intent φίλτρα τα οποία υποδηλώνουν τις ικανότητες του συστατικού. Έτσι το συστατικό θα είναι ικανό να αντιδράσει και σε άλλες ενέργειες. Το intent φίλτρο υποδηλώνεται προσθέτοντας ένα στοιχείο <intent-filter> ως παιδί του στοιχείο που ορίζει το συστατικό.

Υπάρχουν αρκετές συσκευές που λειτουργούν με Android OS και δεν παρέχουν όλες τα ίδια χαρακτηριστικά και ικανότητες. Για την αποτροπή της εγκατάστασης μια εφαρμογής σε μια συσκευή στην οποία δεν μπορεί να εκτελεστεί, πρέπει να ορίζεται ξεκάθαρα ένα προφίλ για τους τύπους των συσκευών και τους περιορισμούς

λογισμικού συστήματος τους οποίους υποστηρίζει μια εφαρμογή. Το προφίλ αυτό εμπεριέχεται επίσης στο αρχείο manifest. Οι περισσότεροι από αυτούς τους ορισμούς είναι απλά πληροφοριακοί και το σύστημα δεν τους διαβάζει. Τους διαβάζουν όμως εξωτερικές υπηρεσίες όπως το Android Market.

Για παράδειγμα αν μια εφαρμογή απαιτεί κάμερα και χρησιμοποιεί το API που εισήχθηκε στο Android 2.1 (επιπέδου 7) πρέπει να δηλωθεί στις απαιτήσεις εφαρμογής στο αρχείο manifest. Με τον τρόπο αυτό, οι συσκευές που δεν έχουν κάμερα και έχουν έκδοση Android λογισμικού χαμηλότερη από το 2.1 δεν θα μπορούν να εγκαταστήσουν την εφαρμογή από το Android Market.

Σημαντικά χαρακτηριστικά συσκευής τα οποία ο δημιουργός εφαρμογής πρέπει να λάβει υπόψη του κατά τη διάρκεια ανάπτυξης εφαρμογών είναι το μέγεθος και η πυκνότητα της οθόνης της συσκευής, τους τρόπους επιβεβαίωσης εισόδου, τα χαρακτηριστικά της συσκευής και η έκδοση του λογισμικού.

Όσον αφορά την οθόνη, για να είναι δυνατή η κατηγοριοποίηση των συσκευών ανάλογα με τον τύπο οθόνης τους το Android καθορίζει δύο χαρακτηριστικά για κάθε συσκευή, το μέγεθος και την πυκνότητα. Έτσι έχουν κατηγοριοποιηθεί σε ομάδες για την εύκολη αναγνώρισή τους. Τα μεγέθη οθονών είναι: μικρή, κανονική, μεγάλη και πολύ μεγάλη. Οι πυκνότητες είναι: χαμηλή, μεσαία, υψηλή και πολύ υψηλή. Αρχικά, η εφαρμογή είναι προκαθορισμένη έτσι ώστε να είναι συμβατή με όλα τα μεγέθη οθόνης και όλες τις πυκνότητες επειδή το Android κάνει τις κατάλληλες τροποποιήσεις στη διασύνδεση χρήστη. Παρόλα αυτά πρέπει να χρησιμοποιούνται εξειδικευμένα περιγράμματα οθόνης για συγκεκριμένα μεγέθη και πυκνότητες με εναλλακτικές πηγές εμφάνισης. Μέσα στο αρχείο manifest πρέπει να δηλώνονται ξεκάθαρα τα μεγέθη των οθονών που υποστηρίζει η εφαρμογή μέσω του στοιχείο <supports-screens>.

Όσον αφορά τις επαληθεύσεις των εισόδων, υπάρχουν συσκευές που παρέχουν διαφορετικό τύπο μηχανισμού εισόδου από το χρήστη όπως είναι το υλικό πληκτρολόγιο (αντί το εικονικά), η trackball, το navigation pad κτλ. Αν η εφαρμογή

του χρήστη απαιτεί ένα συγκεκριμένο είδος μηχανισμού εισόδου τότε και αυτό πρέπει να δηλωθεί ξεκάθαρα στο manifest μέσω του στοιχείου <uses-configuration>.

Όσον αφορά τα χαρακτηριστικά της συσκευής, υπάρχουν πολλά υλικά χαρακτηριστικά αλλά και χαρακτηριστικά λογισμικού που μπορεί να υπάρχουν (ή όχι) σε μια Android συσκευή. Παραδείγματα είναι η κάμερα, ο αισθητήρας φωτός, το Bluetooth, κάποια συγκεκριμένη έκδοση της OpenGL ή ακόμα και η οθόνη αφής. Δεν πρέπει ποτέ να παρθεί ως δεδομένο ότι μια συσκευή περιλαμβάνει όλα ανεξαιρέτως τα χαρακτηριστικά. Για το λόγο αυτό πρέπει να δηλώνονται τα χαρακτηριστικά που χρησιμοποιεί η εφαρμογή μέσω του στοιχείο <uses-feature> στο manifest.

Τέλος, όσον αφορά την έκδοση πλατφόρμας που χρησιμοποιείται, υπάρχει το ενδεχόμενο να μην έχουν όλες οι Android συσκευές την πιο πρόσφατη έκδοση και επομένως να τρέχει η κάθε συσκευή και διαφορετική έκδοση του Android OS. Κάθε νέα έκδοση συνήθως περιλαμβάνει και επιπλέον πρωτόκολλα τα οποία δεν είναι διαθέσιμα στις προηγούμενες εκδόσεις του λογισμικού. Για να καθοριστεί το ποιο σύνολο πρωτοκόλλων είναι διαθέσιμο, κάθε πλατφόρμα ορίζει ένα επίπεδο API. Πρέπει να ορίζεται στο manifest το επίπεδο των API που χρησιμοποιείται στην εφαρμογή μέσω του στοιχείο <uses-sdk>.

Είναι σημαντικό να δηλωθούν όλες αυτές οι απαιτήσεις για κάθε εφαρμογή επειδή όταν η εφαρμογή διανεμηθεί μέσω του Android Market η αγορά θα χρησιμοποιήσει αυτούς τους ορισμούς για να φιλτράρει τις κατάλληλες εφαρμογές για κάθε συσκευή. Έτσι οι εφαρμογές πρέπει να είναι διαθέσιμες μόνο για συσκευές οι οποίες πληρούν τους περιορισμούς.

Μια Android εφαρμογή αποτελείται από περισσότερα πράγματα παρά μόνο κώδικα. Απαιτεί και πηγές που βρίσκονται ξεχωριστά από τον πηγαίο κώδικα όπως εικόνες, αρχεία ήχου και οτιδήποτε σχετίζεται με την οπτική παρουσίαση της εφαρμογής. Για παράδειγμα πρέπει να καθοριστούν τα μενού, τα στυλ, τα χρώματα και η γραφική διασύνδεση χρήστη μέσω XML αρχείων. Χρησιμοποιώντας τέτοιες πηγές για κάθε εφαρμογή είναι πιο εύκολη η ενημέρωση διάφορων χαρακτηριστικών

της εφαρμογής χωρίς να χρειαστεί τροποποίηση στον κώδικα. Επίσης, από τη στιγμή που παρέχεται ένα σύνολο εναλλακτικών πηγών είναι και πιο απλή η βελτιστοποίηση της εφαρμογής για μια ποικιλία διαφορετικών συσκευών.

Για κάθε πηγή που συμπεριλαμβάνεται στο Android project το SDK φτιάχνει εργαλεία που καθορίζουν μια μοναδική, ακέραια ταυτότητα την οποία χρησιμοποιεί ο χρήστης για αναφορά στην πηγή από τον κώδικα ή από άλλες πηγές που καθορίζονται στο XML αρχείο. Μια από τις σημαντικότερες πτυχές παροχής πηγών ανεξάρτητα από τον πηγαίο κώδικα είναι η ικανότητα παροχής εναλλακτικών πηγών για διαφορετικά χαρακτηριστικά. Για παράδειγμα χρησιμοποιείται ξεχωριστή πηγή για τη διασύνδεση της ίδια εφαρμογή στα ελληνικά και ξεχωριστή πηγή για τη διασύνδεση της ίδιας εφαρμογής στα αγγλικά. Απλά πρέπει να καθοριστεί ποιο XML αρχείο θα χρησιμοποιείται σε κάθε περίπτωση κάνοντας αναφορά σε αυτό. Το Android υποστηρίζει πολλούς διαφορετικούς qualifiers για τις εναλλακτικές πηγές. Ο qualifier είναι ένα σύντομο αλφαριθμητικό το οποίο συμπεριλαμβάνεται στο όνομα των φακέλων με τις πηγές για νε καθοριστούν οι ιδιότητες της συσκευής για τις οποίες θα χρησιμοποιηθούν οι πηγές.

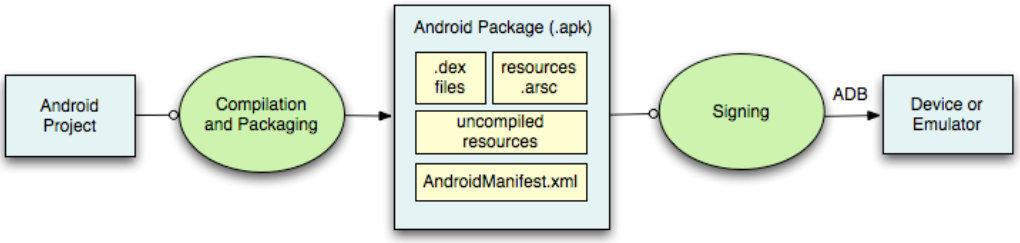
Κατά τη διάρκεια της διαδικασίας κατασκευής μιας εφαρμογής, το Android project πακετάρεται στο .apk αρχείο το οποίο περιέχει το binary της εφαρμογής. Περιέχει όλες τις πληροφορίες που είναι απαραίτητες για την εκτέλεση της εφαρμογής σε μια συσκευή ή σε έναν προσομοιωτή όπως για παράδειγμα τα .dex αρχεία, μια binary έκδοση του AndroidManifest.xml αρχείου, μεταγλωττισμένες πηγές και μη μεταγλωττισμένα αρχεία πηγών της εφαρμογής [28]. Όταν ο χρήστης αναπτύσσει εφαρμογή στο Eclipse, το ADT plugin χτίζει αυξητικά το project καθώς ο χρήστης κάνει αλλαγές στον πηγαίο κώδικα. Το Eclipse εξάγει ένα .apk αρχείο αυτόματα στο φάκελο bin του project έτσι ο χρήστης δε χρειάζεται να κάνει τίποτα απολύτως για την παραγωγή του αρχείου αυτού.

Σε περίπτωση που ο χρήστης δεν αναπτύσσει σε περιβάλλον Eclipse, μπορεί να κτίσει το project με τη βοήθεια του παραγόμενου build.xml Ant αρχείου το οποίο

βρίσκεται στο φάκελο του project. Το αρχείο αυτό καλεί αυτόματα τα εργαλεία κατασκευής.

Για την εκτέλεση μιας εφαρμογής σε προσομοιωτή ή σε συσκευή, η εφαρμογή πρέπει να είναι υπογεγραμμένη σε debug ή release mode. Όταν πρόκειται για απλό έλεγχο της εφαρμογής χρησιμοποιείται υπογραφή σε debug mode ενώ όταν η εφαρμογή θα ελευθερωθεί στην αγορά του Android απαιτείται υπογραφή σερelease mode για την οποία απαιτείται προσωπικό κλειδί.

Η εικόνα 6.3 παρουσιάζει τα συστατικά που εμπλέκονται στη διαδικασία κατασκευής και εκτέλεσης μιας εφαρμογής.



Εικόνα 6. 3 Τα συστατικά που εμπλέκονται στην κατασκευή και εκτέλεση μιας Android εφαρμογής

Η διαδικασία κατασκευής περιλαμβάνει αρκετά εργαλεία και διεργασίες που παράγουν ενδιάμεσα αρχεία για την παραγωγή του .apk. Τα περισσότερα από τα εργαλεία που χρησιμοποιούνται αποκρύπτονται από το χρήστη. Η γενική διεργασία κατασκευής παρουσιάζεται στη συνέχεια. Το Android Asset Packaging εργαλείο (aapt) παίρνει τα αρχεία με τους πόρους της εφαρμογής όπως είναι το AndroidManifest.xml και τα XML αρχεία για τις δραστηριότητες και τα μεταγλωττίζει. Παράγεται επίσης ένα R.java αρχείο το οποίο περιλαμβάνει τις αναφορές των πηγών που μπορούν να χρησιμοποιηθούν στον κώδικα.

Ακολουθεί το εργαλείο aidl που μετατρέπει όλα τα .aidl interfaces που υπάρχουν σε java interfaces. Όλος ο κώδικας που είναι γραμμένος σε java συμπεριλαμβανομένου του R.java αρχείου και των .aidl αρχείων μεταγλωττίζονται από τον compiler της Java για να εξαχθούν τα .class αρχεία.

Το εργαλείο dex μεταφράζει τα .class αρχεία σε Dalvik byte code. Όλες οι 3rd party βιβλιοθήκες και τα .class αρχεία που περιλαμβάνονται στο projects μετατρέπονται σε .dex αρχεία για να είναι δυνατή η αποθήκευσή τους μέσα στο τελικό .apk αρχείο.

Όλες οι μη μεταγλωττισμένες πηγές (όπως οι εικόνες), οι μεταγλωττισμένες πηγές και τα .dex αρχεία στέλνονται στο εργαλείο apkbuilder για να ενσωματωθούν σε ένα .apk αρχείο/

Όταν κατασκευαστεί το .apk πρέπει να υπογραφεί είτε με debug κλειδί είτε με release κλειδί πριν γίνει η εγκατάστασή του σε οποιαδήποτε συσκευή. Τέλος, αν η εφαρμογή υπογράφεται σε release mode πρέπει να ευθυγραμμιστεί το .apk αρχείο με το zipalign εργαλείο. Η ευθυγράμμιση του τελικού .apk ελαττώνει τη χρήση μνήμης όταν η εφαρμογή τρέχει σε μια συσκευή Android.

6.3 Τα κύρια συστατικά μιας εφαρμογής για το Android της Google και η πορεία εκτέλεσης της

Μια Android εφαρμογή μπορεί ανά πάσα χρονική στιγμή να βρίσκεται σε κάποια κατάσταση. Η κάθε κατάσταση, περιγράφεται από τα συστατικά τα οποία την συνθέτουν. Μια διεργασία μιας εφαρμογής μπορεί να βρίσκεται σε μια από τις ακόλουθες καταστάσεις: Active Process, Visible Process, Started Service Process, Background Process και Empty Process.

Active process (Ενεργή): μια ενεργή διεργασία βρίσκεται στο προσκήνιο και είναι αυτή η οποία φιλοξενεί εφαρμογές των οποίων τα συστατικά αλληλεπιδρούν με το χρήστη τη συγκεκριμένη στιγμή. Οι διεργασίες αυτής της κατηγορίας είναι αυτές τις οποίες το Android προσπαθεί να κρατήσει ζωντανές απαιτώντας πόρους από άλλες. Γενικά, υπάρχουν πολύ λίγες τέτοιες διεργασίες και είναι αυτές οι οποίες τερματίζονται τελευταίες όταν προκύψει ανάγκη. Οι ενεργές διεργασίες περιλαμβάνουν δραστηριότητες σε ενεργή κατάσταση που βρίσκονται δηλαδή στο

προσκήνιο και αντιδρούν στα γεγονότα του χρήστη. Περιλαμβάνουν επίσης δραστηριότητες, υπηρεσίες και broadcast receivers που προς το παρόν εκτελούν ένα χειριστή γεγονότων onReceive. Τέλος, περιλαμβάνουν υπηρεσίες που εκτελούν χειριστές γεγονότων onStart, onCreate και onDestroy.

Visible Process (Ορατή): Μια διεργασία που βρίσκεται σε αυτή την κατάσταση είναι ορατή αλλά είναι και μη ενεργή. Συνήθως τέτοιες διεργασίες φιλοξενούν ορατές δραστηριότητες οι οποίες είναι μεν ορατές αλλά δεν είναι στο προσκήνιο ή δεν αντιδρούν στις ενέργειες του χρήστη. Αυτό συμβαίνει όταν μια δραστηριότητα είναι μερικώς κρυμμένη από μια διαφανή δραστηριότητα ή μια δραστηριότητα που δεν καταλαμβάνει όλη την οθόνη. Υπάρχουν πολύ λίγες ορατές διεργασίες και τερματίζονται μόνο σε ειδικές περιπτώσεις για να επιτραπεί η συνέχιση της εκτέλεσης στις Ενεργές διεργασίες.

Started Service Process (Εκκίνηση Υπηρεσίας): Οι διεργασίες που έχουν να κάνουν με υπηρεσίες υποστηρίζουν τη συνεχή επεξεργασία η οποία πρέπει να συνεχίζεται χωρίς ορατή διασύνδεση. Επειδή οι υπηρεσίες δεν αλληλεπιδρούν απευθείας με το χρήστη, λαμβάνουν λίγο χαμηλότερη προτεραιότητα παρά οι ορατές διεργασίες. Εξακολουθούν να θεωρούνται διεργασίες που εκτελούνται στο προσκήνιο και δεν τερματίζονται εκτός και αν απαιτούνται πόροι για ενεργές ή ορατές διεργασίες.

Background Process (Παρασκήνιο): Μια διεργασία βρίσκεται στο παρασκήνιο όταν ασχολείται με δραστηριότητες που δεν είναι ορατές και που δεν έχουν υπηρεσίες που έχουν αρχίσει να εκτελούνται ήδη. Συνήθως υπάρχει ένας μεγάλος αριθμός από διεργασίες σε αυτή την κατάσταση και είναι αυτές που τερματίζονται συνήθως από το σύστημα όταν υπάρχει απαίτηση για επιπλέον πόρους. Οι διεργασίες αυτές τερματίζονται με την πολιτική last-seen-first-killed.

Empty Process (Άδεια): Για τη βελτίωση της όλης απόδοσης του συστήματος το android συχνά διατηρεί στη μνήμη εφαρμογές αφού έχουν φτάσει στο τέλος του κύκλου ζωής τους. Τις διατηρεί για να βελτιώσει το χρόνο εκκίνησης των εφαρμογών

όταν επιλέγονται για επανεκκίνηση. Τέτοιες διεργασίες τερματίζονται εντελώς όταν είναι απαραίτητο και με τυχαίο τρόπο.

Τα συστατικά μιας εφαρμογής: τα συστατικά μιας εφαρμογής είναι τα θεμελιώδη κατασκευαστικά μπλοκ μιας Android εφαρμογής. Κάθε συστατικό είναι ένα διαφορετικό σημείο μέσω του οποίου το σύστημα μπορεί να εισέλθει στην εφαρμογή. Δεν είναι όλα τα συστατικά πραγματικά σημεία εισόδου για το χρήστη. Κάποια σχετίζονται μεταξύ τους αλλά το καθένα υπάρχει από μόνο του ως μια οντότητα και έχει ένα ξεχωριστό ρόλο. Κάθε ένα είναι ένα μοναδικό κατασκευαστικό μπλοκ που βοηθά στον καθορισμό της συμπεριφοράς της εφαρμογής του χρήστη.

Υπάρχουν τέσσερις τύποι συστατικών εφαρμογών [27]. Κάθε τύπος εξυπηρετεί ένα διαφορετικό σκοπό και έχει ένα ξεχωριστό κύκλο ζωής που καθορίζει τον τρόπο με τον οποίο το συστατικό δημιουργείται και καταστρέφεται. Οι τέσσερις τύποι των συστατικών είναι Activities, Services, Content Providers και Broadcast Receivers.

Activities: Μια δραστηριότητα αναπαριστά μια μοναδική οθόνη με μια διασύνδεση χρήστη. Για παράδειγμα μια εφαρμογή email είναι δυνατό να έχει μια δραστηριότητα για την εμφάνιση των email, μια άλλη δραστηριότητα για τη σύνθεσή τους κτλ. Παρόλο που οι δραστηριότητες συνεργάζονται για να παρέχουν μια εμπειρία στο χρήστη όσον αφορά την εφαρμογή, η κάθε μια είναι ανεξάρτητη από τις άλλες. Επίσης, κάθε εφαρμογή μπορεί να εκκινεί οποιαδήποτε από αυτές τις δραστηριότητες αν η εφαρμογή τους το επιτρέπει. Μια δραστηριότητα υλοποιείται ως υποκλάση της κλάσης Activity.

Services: Μια υπηρεσία είναι ένα συστατικό που τρέχει στο παρασκήνιο για την πραγματοποίηση μακροχρόνιων λειτουργιών ή για την εξυπηρέτηση απομακρυσμένων διεργασιών. Μια υπηρεσία δεν παρέχει μια διασύνδεση χρήστη. Για παράδειγμα μια υπηρεσία μπορεί να αναπαράγει μουσική στο παρασκήνιο καθώς ο χρήστης βρίσκεται σε μια άλλη εφαρμογή. Ένα άλλο συστατικό, όπως μια δραστηριότητα για παράδειγμα, μπορεί να εκκινήσει μια υπηρεσία και να την αφήσει

να τρέχει ή να δεσμευτεί μαζί της για σκοπούς αλληλεπίδρασης. Μια υπηρεσία είναι υλοποιημένη ως υποκλάση της κλάσης Service.

Content Providers: ένας content provider διαχειρίζεται ένα σύνολο δεδομένων το οποίο διαμοιράζεται ανάμεσα σε κάποιες εφαρμογές. Δεδομένα μπορούν να αποθηκευτούν στο σύστημα αρχείων, στην SQLite βάση δεδομένων, στο διαδίκτυο ή σε οποιαδήποτε άλλη τοποθεσία στην οποία μπορεί να έχει πρόσβαση η εφαρμογή. Μέσω του content provider άλλες εφαρμογές μπορούν να κάνουν ερωτήσεις στα δεδομένα ή να τα τροποποιήσουν αν επιτρέπεται. Οι content providers είναι χρήσιμοι για την ανάγνωση και εγγραφή δεδομένων που είναι προσωπικά της εφαρμογής και όχι διαμοιραζόμενα. Ένας content provider υλοποιείται ως υποκλάση της κλάσης Content Provider και πρέπει να υλοποιεί ένα προκαθορισμένο σύνολο πρωτοκόλλων που επιτρέπουν στις άλλες εφαρμογές να κάνουν συναλλαγές.

Broadcast Receivers: Ένας broadcast receiver είναι ένα συστατικό το οποίο αντιδρά στις ανακοινώσεις του συστήματος. Πολλοί broadcasters προέρχονται από το σύστημα, για παράδειγμα ο broadcaster που είναι υπεύθυνος για την ανακοίνωση του γεγονότος ότι η οθόνη έχει σβήσει, ότι η μπαταρία τελειώνει ή ότι μια εικόνα έχει παρθεί μέσω της φωτογραφικής. Οι εφαρμογές μπορούν επίσης να αρχικοποιήσουν broadcasts για να ενημερώσουν π.χ. άλλες εφαρμογές ότι κάποια δεδομένα έχουν «κατέβει» στη συσκευή και ότι είναι διαθέσιμα για να τα χρησιμοποιήσουν. Παρόλο που οι broadcast receivers δεν εμφανίζουν διασύνδεση χρήστη, μπορεί να δημιουργήσουν μια μπάρα ειδοποιήσεων για να ενημερώνουν το χρήστη όταν ένα γεγονός broadcast συμβαίνει. Συνήθως, ένας broadcast receiver είναι απλά μια πύλη για άλλα συστατικά και έχει ως σκοπό να κάνει πολύ λίγη δουλειά. Για παράδειγμα, μπορεί απλά να αρχικοποιήσει μια υπηρεσία για να κάνει κάτι σχετικά με ένα γεγονός. Ο broadcast receiver υλοποιείται ως υποκλάση της κλάσης Broadcast Receiver και κάθε broadcast παραδίδεται ως ένα intent αντικείμενο.

Μια μοναδική πτυχή της σχεδίασης του Android συστήματος είναι ότι οποιαδήποτε εφαρμογή μπορεί να εκκινήσει ένα συστατικό κάποιας άλλης

εφαρμογής. Για παράδειγμα αν ο χρήστης θέλει να τραβήξει μια φωτογραφία με τη κάμερα η εφαρμογή του χρήστη θα εκκινήσει το αντίστοιχο συστατικό κάποιας άλλης εφαρμογής που κάνει αυτή τη δουλειά. Έτσι δεν απαιτείται να υλοποιηθεί ξανά η ίδια λειτουργικότητα. Το μόνο που πρέπει να γίνει είναι η εκκίνηση του συστατικού αυτού εκκινώντας την αντίστοιχη δραστηριότητα στην εφαρμογή της κάμερας.

Όταν το σύστημα εκκινεί ένα συστατικό, ξεκινά στην ουσία τη διεργασία εκείνης της εφαρμογής αν δεν εκτελείται ήδη και προκαλεί τη δημιουργία στιγμιότυπων των κλάσεων που απαιτούνται για το συστατικό. Για παράδειγμα, αν η εφαρμογή του χρήστη ξεκινά τη δραστηριότητα στην εφαρμογή της κάμερας η οποία είναι υπεύθυνη για τη λήψη φωτογραφιών, η δραστηριότητα εκείνη τρέχει στη διεργασία που ανήκει στην εφαρμογή της κάμερας και όχι στη διεργασία της εφαρμογής του χρήστη. Φαίνεται λοιπόν ότι σε αντίθεση με τις εφαρμογές άλλων συστημάτων, οι εφαρμογές του Android δεν έχουν ένα μοναδικό σημείο εισόδου (δεν υπάρχει main συνάρτηση). Λόγω του ότι το σύστημα τρέχει κάθε εφαρμογή σε διαφορετική διεργασία με περιορισμούς αρχείων που απαγορεύουν την πρόσβαση σε άλλες εφαρμογές, η εφαρμογή του χρήστη δεν μπορεί να προκαλέσει την απευθείας ενεργοποίηση ενός συστατικού άλλης εφαρμογής. Στο Android όμως αυτό μπορεί να γίνει.

Για την ενεργοποίηση ενός συστατικού άλλης εφαρμογής πρέπει να παραδοθεί ένα μήνυμα στο σύστημα που να καθορίζει την πρόθεση (intent) για εκκίνηση ενός συγκεκριμένου συστατικού. Ακολούθως το σύστημα ενεργοποιεί αυτό το συστατικό για την εφαρμογή που το ζήτησε.

Ενεργοποίηση συστατικών: Τρεις από τους τέσσερις τύπους συστατικών (activities, services, broadcast receivers) ενεργοποιούνται μέσω ασύγχρονων μηνυμάτων που ονομάζονται intent. Τα intents δεσμεύουν ανεξάρτητα συστατικά μεταξύ τους στη διάρκεια του χρόνου εκτέλεσης που μπορούν να ανήκουν στην εφαρμογή του χρήστη ή σε άλλες εφαρμογές. Ένα intent δημιουργείται από ένα intent αντικείμενο το οποίο καθορίζει το μήνυμα που θα ενεργοποιήσει είτε ένα

συγκεκριμένο συστατικό είτε ένα συγκεκριμένο τύπο συστατικού. Ένα intent μπορεί να είναι είτε explicit είτε implicit.

Για δραστηριότητες και υπηρεσίες, ένα Intent καθορίζει την ενέργεια που θα εκτελεστεί και μπορεί να καθορίζει το URL των δεδομένων στα οποία θα δράσει. Σε κάποιες περιπτώσεις μπορεί να ξεκινήσει μια δραστηριότητα για να λάβει ένα αποτέλεσμα. Η δραστηριότητα επίσης επιστρέφει το αποτέλεσμα ως intent.

Για τους broadcast receivers το Intent απλά καθορίζει την ανακοίνωση του ότι υπάρχει μια ανά-μετάδοση.

Ο τέταρτος τύπος συστατικού που είναι ο content provider δεν ενεργοποιείται από Intents. Ενεργοποιείται όταν είναι ο στόχος μιας αίτησης από ένα ContentResolver. Ο content resolver χειρίζεται όλες τις απευθείας συναλλαγές με τον content provider έτσι ώστε υο συστατικό που πραγματοποιεί συναλλαγές με τον provider να μην χρειάζεται και να καλεί απλά τις μεθόδους του contentResolver αντικειμένου. Αυτό αφήνει ένα επίπεδο αφαίρεσης μεταξύ του content provider και του συστατικού που ζητά κάποιες πληροφορίες.

Υπάρχουν ξεχωριστές μέθοδοι για την ενεργοποίηση κάθε τύπου αντικειμένου. Για την εκκίνηση μιας δραστηριότητας αρκεί να περαστεί ένα Intent στη μέθοδο startActivity() ή startActivityForResault(). Για την εκκίνηση μιας υπηρεσίας αρκεί να περαστεί ένα intent στη μέθοδο startService(). Επίσης μπορεί να δεσμευτεί η εφαρμογή σε μια υπηρεσία περνώντας ένα Intent στη μέθοδο bindService(). Για την αρχικοποίηση ενός broadcast αρκεί να περαστεί ένα Intent σε μεθόδους όπως sendBroadcast(), sendOrderedBroadcast() ή sendStickyBroadcast(). Είναι επίσης δυνατή η πραγματοποίηση ερωτήματος σε ένα content provider καλώντας τη μέθοδο query() σε ένα ContentResolver.

Κεφάλαιο 7

Ασφάλεια λογισμικού στο iOS της Apple και στο Android της Google

7.1 Ασφάλεια στο iOS της Apple

Έχοντας εξετάσει σε βάθος το iOS θα ήταν σωστό να γίνει και αναφορά στη δομή ασφαλείας του iPhone από ένα υψηλό επίπεδο. Έχουν εντοπιστεί ήδη κάποια ζητήματα σχετικά με την ασφάλεια στο iOS για iPhone τα οποία παρουσιάζονται στη συνέχεια [29]. Παράλληλα, και η ίδια η φιλοσοφία ασφάλειας του iPhone παρουσιάζει αρκετά λάθη.

Η προσέγγιση της Apple στο να κάνει το iPhone μια ασφαλή συσκευή ήταν το να ελαχιστοποιήσει την επιφάνεια επίθεσης στη συσκευή [30] και να ελαχιστοποιήσει την έκθεση της συσκευής σε ευπάθειες. Για να επιτύχει το στόχο της, η Apple επέτρεψε την πρόσβαση για εγγραφή μόνο στη sandbox περιοχή του συστήματος αρχείων και απαγόρευσε την εγκατάσταση third-party εφαρμογών που αναπτύσσονταν από άλλους. Αρκετά χαρακτηριστικά του Safari αφαιρέθηκαν από τον Mobile Safari που συναντάται στο iOS για το iPhone συμπεριλαμβανομένης και της ικανότητας χρήσης Plug-ins όπως το Flash καθώς και την ικανότητα εγκατάστασης συγκεκριμένων τύπων αρχείων από το διαδίκτυο.

Ο Mobile Safari είναι περιορισμένος στο να εκτελεί μόνο Javascript κώδικα και μόνο στη sandbox περιοχή. Προκύπτει λοιπόν το συμπέρασμα ότι αντί να επιτρέπεται η ελευθερία και η ευελιξία στο χρήστη κάνοντας το σύστημα εύρωστο και ασφαλές, η προσέγγιση της Apple προκάλεσε τη δημιουργία μιας ελεγχόμενης,

κλειστού τύπου συσκευής. Η φιλοσοφία και μόνο που ακολουθεί η Apple έχει πρόβλημα ενώ και η ίδια η αρχιτεκτονική παρουσιάζει μειονεκτήματα αφού κάνει πολύ λίγη προσπάθεια όσον αφορά την προστασία διαφορετικών τμημάτων της συσκευής ανεξάρτητα. Αυτό προκύπτει από το γεγονός ότι όλες οι σημαντικές διεργασίες εκτελούνται ως super user με δικαιώματα διαχειριστή, γεγονός που αποτελεί μειονέκτημα όσον αφορά τον τομέα της ασφάλειας.

Έτσι, ο εισβολέας μπορεί να ελέγξει ολόκληρο το iPhone αν καταφέρει να εκμεταλλευθεί μια από τις ευαισθησίες του iOS με κάποια από τις εφαρμογές του. Η ασφάλεια του όλου συστήματος μπορεί να παρομοιαστεί με την προσπάθεια που κάνει ένας ιδιοκτήτης στο να προστατέψει το σπίτι του. Αγοράζει μεν όλα τα συστήματα κλειδώματος από την αγορά, αφήνει όμως όλα τα υπόλοιπα δωμάτια εκτεθειμένα χρησιμοποιώντας μόνο ένα σημείο ασφαλείας. Παρόλο που θα είναι δύσκολο να μπει κανείς στο σπίτι, όταν μπει θα είναι ελεύθερος να κάνει ό,τι θέλει με το περιεχόμενο.

Ακόμα ένα σημαντικό πρόβλημα ασφάλειας προκύπτει από το γεγονός ότι το iPhone χρησιμοποιεί αρκετές εφαρμογές συμπεριλαμβανομένων των MobileSafari και MobileMail οι οποίες βασίζονται σε projects ανοικτού κώδικα. Επιπλέον, αρκετές ανοικτές βιβλιοθήκες χρησιμοποιούνται σε πολλές από τις εφαρμογές του iPhone. Για παράδειγμα η libtiff και η WebKit χρησιμοποιούνται και οι δύο στο MobileSafari και MobileMail. Η χρήση και η διαμοίραση projects ανοικτού κώδικα είναι πολύ ευνοϊκή φτάνει να βρίσκονται εγκαταστημένες οι τελευταίες εκδόσεις τους. Εξάλλου, παλιές εκδόσεις του libtiff έχουν ήδη υποστεί κακόβουλες επιθέσεις.

Ο λόγος για τον οποίο τα projects ανοικτού κώδικα παρουσιάζουν προβλήματα στην ασφάλεια είναι ότι το μόνο που χρειάζεται να κάνει ο εισβολέας είναι να κοιτάξει την τελευταία έκδοση του project καθώς και ακολούθως την έκδοση που χρησιμοποιείται αυτή τη στιγμή στο iOS. Σε περίπτωση που βρει ότι στη συσκευή είναι εγκατεστημένη παλαιότερη έκδοση θα εντοπίσει το σημείο το οποίο διορθώθηκε και θα εισβάλει με ευκολία στο σύστημα του iOS. Με το να χρησιμοποιούνται

παλαιότερες εκδόσεις των projects ανοικτού κώδικα στο iOS, η Apple έκανε ευκολότερη τη δουλειά των hackers στον εντοπισμό πιθανών επιθέσεων. Η Apple απέτυχε επίσης στο να δυσκολέψει τους hackers στο να εκμεταλλευθούν τις ευαισθησίες του λογισμικού. Με το να μην υλοποιεί γνωστές τεχνικές όπως Address Space Layout Randomization (ASLR) ή μη εκτελέσιμου σωρού στην έκδοση του OS X που υπάρχει στη συσκευή δεν προκαλεί καμιά δυσκολία στην ανάπτυξη και διανομή κακόβουλων προγραμμάτων.

Βέβαια, έχουν υιοθετηθεί και κάποιες καλές πρακτικές οι οποίες βοηθούν στην αύξηση της ασφάλειας στο iPhone. Όμως οι hackers βρίσκουν και πάλι τρόπους για να ξεπερνούν τα εμπόδια που τους θέτει η Apple. Για παράδειγμα, η στοίβα είναι μη εκτελέσιμη στο iOS έτσι ώστε ο εισβολέας να μην μπορεί εύκολα να προσθέσει φορτίο σε αυτήν μέσω buffer overflow [57] και να την εκτελέσει. Όμως, μια μη εκτελέσιμη στοίβα δεν παρέχει προστασία έναντι στην return-to-libc επίθεση η οποία υλοποιείται σε μια jailbreaking επίθεση. Επιπρόσθετα, η Apple κυκλοφόρησε νέες εκδόσεις firmware οι οποίες επιδιορθώνουν συγκεκριμένες ευαισθησίες για την αποφυγή jailbreaking. Η εταιρεία προσπάθησε επίσης να αποτρέψει το ξεκλείδωμα συσκευών χρησιμοποιώντας μια πιο νέα έκδοση του bootloader. Η προσπάθεια αυτή όμως απέτυχε επειδή οι hackers βρήκαν τον τρόπο να επιστρέφουν στην παλιά έκδοση του bootloader. Με όλα αυτά, μπορεί να δημιουργηθεί στον καθένα η απορία του πώς η Apple χρησιμοποίησε τόσο υψηλή τεχνολογία στην ανάπτυξη λογισμικού αλλά δεν κατάφερε παρόλα αυτά να δημιουργήσει μια ασφαλή συσκευή για το κοινό της.

Το iPhone είναι επίσης ικανό για την αποστολή και συλλογή δεδομένων με τη χρήση του EDGE δικτύου. Το δίκτυο αυτό παρέχεται από την AT&T. Εναλλακτικά, το iPhone μπορεί να στείλει δεδομένα μέσω ασύρματων συνδέσεων και μέσω Bluetooth. Ένα iPhone αρχικά ενεργοποιείται μέσα από την επικοινωνία του με το iTunes. Μέσα από το πρόγραμμα αυτό, το iPhone είναι ικανό να λαμβάνει δεδομένα και να συγχρονίζεται με τον υπολογιστή. Όπως αναφέρθηκε και πιο πάνω, η

αρχιτεκτονική ασφαλείας του iPhone επιχειρεί την ελάττωση της επιφάνειας που μπορεί να δεχθεί επίθεση. Αυτό επιτυγχάνεται περιορίζοντας τον αριθμό των εφαρμογών σε μια συσκευή καθώς και τη λειτουργικότητά των υπαρχόντων εφαρμογών. Η συσκευή δεν περιλαμβάνει κοινά binaries όπως Bash, ssh ή Ls.

Ο περιορισμός των τύπων δεδομένων που επεξεργάζονται από τη συσκευή είναι ένας αποδοτικός τρόπος μείωσης της έκθεσης σε πιθανές επιθέσεις. Συνεχίζοντας με την προοπτική της μείωσης έκθεσης, το iPhone όπως αναφέρθηκε πριν, δεν επιτρέπει την εγκατάσταση εφαρμογών από τρίτους στη συσκευή. Αυτό επιτυγχάνεται με άρνηση πρόσβασης στο σύστημα αρχείων και με τη μη ύπαρξη του κατάλληλου SDK [31]. Παρομοίως, υπάρχει τεχνική δυσκολία στη μη ύπαρξη ενός toolchain για την κατασκευή εφαρμογών για Mac OS X που τρέχει σε ARM επεξεργαστή. Δεν υπάρχει δηλαδή κανένας compiler εκτός περιβάλλοντος Apple ο οποίος να μπορεί να χτίσει εφαρμογές που θα τρέχουν σε iOS ακόμα και αν το αρχείο μπορεί να τοποθετηθεί στο σύστημα αρχείων της συσκευής. Με την απαγόρευση εγκατάστασης εφαρμογών από τρίτους, ο αριθμός των πιθανοτήτων εισβολής από τρίτους μειώνεται. Όταν όμως μια εφαρμογή του iOS παραβιαστεί από τον εισβολέα, δεν υπάρχουν πολλά πράγματα που θα αποτρέψουν τον εισβολέα από το να καταλάβει τον έλεγχο ολόκληρου του συστήματος.

Το γεγονός ότι δε χρησιμοποιείται η έννοια του τυχαίου στη θέση των διευθύνσεων μνήμης του συστήματος, κάθε φορά που μια διεργασία εκτελείται, η στοίβα, ο σωρός αλλά και ο εκτελέσιμος κώδικας βρίσκονται στο ίδιο ακριβώς σημείο μνήμης στο οποίο ήταν και στις προηγούμενες εκτελέσεις της ίδιας διεργασίας. Αυτό βοηθά τους εισβολείς στη συγγραφή κώδικα που αφού τους επιτρέπει να μαντέψουν τη μορφή της μνήμης στην εκτέλεση μιας εφαρμογής αλλά και από συσκευή σε συσκευή.

Επιπλέον, το γεγονός ότι ο σωρός είναι εκτελέσιμος, επιτρέπει στους εισβολείς να γράψουν τον κώδικά τους και απλά να τον τοποθετήσουν σε ένα σημείο του σωρού. Μόλις αποκτήσουν το έλεγχο της συσκευής μπορούν να τον καλέσουν για

την επίτευξη του στόχου τους. Θα ήταν ίσως προτιμότερο να μη δινόταν τόσο μεγάλη σημασία στην αποτροπή του ξεκλειδώματος του iPhone από την Apple ενώ περισσότερη ασφάλεια θα έπρεπε να προσφέρεται όσον αφορά την εκτέλεση των εφαρμογών από εισβολείς.

Υπάρχουν βέβαια και κάποια χαρακτηριστικά που χρησιμοποιούνται στο iOS και έχουν ως στόχο τη διαφύλαξη των προσωπικών δεδομένων του χρήστη και της συσκευής. Κάποια από τα γενικά χαρακτηριστικά ασφάλειας του iOS είναι τα ακόλουθα [32]:

Geo location: Αν κανείς χάσει την iPhone συσκευή του τότε το χαρακτηριστικό αυτό είναι πολύ χρήσιμο. Σε συνδυασμό με το MobileMe [58] που παρέχεται από την Apple, βοηθά στον εντοπισμό του κινητού τηλεφώνου. Η εφαρμογή MobileMe υποδεικνύει άμεσα την τοποθεσία του iPhone αν αυτό βρίσκεται σε λειτουργία. Το μόνο που απαιτείται είναι να είναι ενεργοποιημένος ο εντοπισμός τοποθεσίας του τηλεφώνου καθώς αυτό κινείται.

Auto Erase: Πρόκειται για ένα ενδιαφέρον χαρακτηριστικό που αφορά την προστασία των ευαίσθητων δεδομένων του χρήστη. Σε περίπτωση που το τηλέφωνο χαθεί, υπάρχει η δυνατότητα μέσω της Apple για τη διαγραφή των δεδομένων. Όμως, ο χρήστης του τηλεφώνου μπορεί να εντοπίσει και πάλι τα δεδομένα μέσω των back-ups που εμφανίζονται στο μενού του τηλεφώνου. Σε περίπτωση βέβαια που ακόμα και τα αντίγραφα δεδομένων περιλαμβάνουν κωδικό ασφαλείας, μετά από δέκα λανθασμένους κωδικούς τα δεδομένα διαγράφονται αυτόματα και δεν είναι δυνατή η ανάκτησή τους ποτέ ξανά.

Data Encryption: το iOS μπορεί να κρυπτογραφήσει τα δεδομένα του χρήστη αν του ζητηθεί. Αυτή η επιλογή είναι διαθέσιμη μόνο για χρήστες της συσκευής που χρησιμοποιούν κωδικό πρόσβασης σε αυτήν.

4 επίπεδα OS: Οι εφαρμογές του iOS δεν έχουν απευθείας πρόσβαση στο υλικό της συσκευής. Όλες οι αλληλεπιδράσεις με το υλικό ελέγχονται αποκλειστικά από ένα αριθμό διαφορετικών επιπέδων λογισμικού που δρουν ως ενδιάμεσα μεταξύ της

εφαρμογής και του υλικού της συσκευής. Με τον τρόπο αυτό διασφαλίζεται η ασφάλεια της συσκευής.

Όπως φαίνεται λοιπόν, δεν υπάρχουν μόνο μειονεκτήματα όσον αφορά την ασφάλεια του iOS. Το iPhone έχει τη δυνατότητα να παρέχει ασφαλή πρόσβαση στις υπηρεσίες και να προστατεύει με τον τρόπο αυτό τα δεδομένα της κάθε υπηρεσίας. Παρέχει δυνατή κρυπτογράφηση για μετάδοση δεδομένων καθώς και μεθόδους αυθεντικοποίησης για πρόσβαση σε υπηρεσίες συνεργασίας. Από το iPhone 3GS και έπειτα, παρέχεται και κρυπτογράφηση υλικού για όλα τα δεδομένα που αποθηκεύονται στη συσκευή. Επίσης, το iPhone παρέχει ασφαλή προστασία με τη χρήση κωδικών πρόσβασης και πολιτικών. Ακόμα και αν η συσκευή πέσει σε λάθος χέρια, υπάρχει η δυνατότητα καθαρισμού όλων των δεδομένων από τη συσκευή για πάντα. Για την προστασία λοιπόν των δεδομένων πρέπει να λαμβάνει κανείς υπόψη τις μεθόδους που απαγορεύουν την χρήση της συσκευής από άτομα που δεν έχουν αυτή το δικαίωμα αλλά και τα πρωτόκολλα δικτύου και την κρυπτογράφηση των δεδομένων καθώς αυτά μεταφέρονται [32] .

Μια πολύ καλή πολιτική που χρησιμοποιείται είναι να χρησιμοποιούνται κωδικοί στη συσκευή. Όταν υπάρχει κωδικός ασφαλείας σε μια συσκευή τότε αποτρέπονται χρήστες που δεν έχουν δικαίωμα πρόσβασης να τη χρησιμοποιήσουν και να έχουν πρόσβαση στα δεδομένα. Στο iOS υπάρχει η επιλογή κωδικού από ένα εκτεταμένο σύνολο απαιτήσεων έτσι ώστε ο κάθε χρήστης να επιλέξει κάποιο που θα εξυπηρετεί καλύτερα τις ανάγκες του. Μεταξύ αυτών των απαιτήσεων συμπεριλαμβάνονται timeout περίοδοι, δύναμη κωδικού αλλά και συχνότητα αλλαγής κωδικού.

Επιπλέον, υπάρχει και ασφαλής επιβεβαίωση συσκευής. Τα προφίλ επιβεβαίωσης είναι XML αρχεία που περιλαμβάνουν πολιτικές ασφαλείας της συσκευής και περιορισμούς. Περιλαμβάνουν επίσης πληροφορίες επιβεβαίωσης για VPN, ρυθμίσεις που αφορούν WiFi συνδέσεις, λογαριασμούς email και ημερολογίων. Η δυνατότητα που δίνεται για τη χρήση πολιτικών για κωδικούς σε συνδυασμό με διάφορες ρυθμίσεις που γίνονται στη συσκευή σε ένα προφίλ επιβεβαίωσης,

εξασφαλίζει ότι οι συσκευή θα είναι σωστά επιβεβαιωμένη και σύμφωνα με τα πρότυπα ασφαλείας που επιθυμεί ο χρήστης. Και επειδή τα προφίλ επιβεβαίωσης μπορούν να είναι και κρυπτογραφημένα και κλειδωμένα, οι ρυθμίσεις τους δεν μπορούν να τροποποιηθούν ή να διαμοιραστούν από τρίτους.

Υπάρχουν επίσης και περιορισμοί σχετικά με το ποια χαρακτηριστικά επιτρέπεται να χρησιμοποιούνται σε μια συσκευή έτσι ώστε να αυξάνεται η ασφάλειά της. Συνήθως χαρακτηριστικά που αφορούν σύνδεση στο διαδίκτυο αλλά και εφαρμογές διαδικτύου όπως ο Safari και το Youtube, εμπεριέχουν περισσότερα ρίσκα. Οι περιορισμοί που ένας χρήστης μπορεί να επιλέξει να θέσει στη συσκευή του μπορεί να περιλαμβάνουν περιορισμούς κάμερας, επαφών, μηνυμάτων κτλ.

Πολύ σημαντική είναι και η κρυπτογράφηση. Από το iPhone 3GS και έπειτα χρησιμοποιείται και κρυπτογράφηση βασισμένη στο υλικό. Αυτή χρησιμοποιεί AES 256 bit κωδικοποίηση για την προστασία όλων των δεδομένων στη συσκευή. Η κρυπτογράφηση είναι πάντα ενεργοποιημένη και ο χρήστης δεν μπορεί να την απενεργοποιήσει. Επιπλέον, τα δεδομένα που αποθηκεύονται για σκοπούς ασφαλείας στο iTunes μπορούν επίσης να κρυπτογραφηθούν αν το επιθυμεί ο χρήστης. Όταν ένα προφίλ κρυπτογράφησης αποθηκευτεί στη συσκευή, τότε η δυνατότητα αυτή ενεργοποιείται αυτόματα. Για την επιπλέον προστασία των δεδομένων οι χρήστες μπορούν να έχουν πρόσβαση σε πρωτόκολλα που τους επιτρέπουν την κρυπτογράφηση των δεδομένων μέσα στις εφαρμογές τους.

Το iPhone υποστηρίζει επίσης και SSL v3 καθώς και Transport Layer Security (TLS v1). Πρόκειται για την επόμενη γενιά προτύπων του διαδικτύου. Ο Safari, το ημερολόγιο, το Mail και άλλες εφαρμογές διαδικτύου εκκινούν αυτόματα αυτούς τους μηχανισμούς για να επιτρέπουν την ύπαρξη ενός κρυπτογραφημένου καναλιού επικοινωνίας μεταξύ του iPhone και των υπηρεσιών που εκτελούνται.

Όσον αφορά το περιβάλλον εκτέλεσης των εφαρμογών, οι εφαρμογές της συσκευής είναι sandboxed έτσι ώστε να μην μπορούν να έχουν πρόσβαση στα δεδομένα που αποθηκεύονται από άλλες εφαρμογές. Το σύστημα αρχείων, οι πηγές

και ο πυρήνας αποκρύπτονται από το χώρο εφαρμογών του χρήστη. Αν μια εφαρμογή χρειάζεται πρόσβαση στα δεδομένα άλλων εφαρμογών μπορεί να το κάνει χρησιμοποιώντας τα πρωτόκολλα και τις υπηρεσίες που παρέχονται από το iOS. Επίσης αποτρέπεται και η παραγωγή κώδικα από τις εφαρμογές.

Το iOS παρέχει επίσης ένα ασφαλείς και κρυπτογραφημένο χώρο για την αποθήκευση ψηφιακών ταυτοτήτων, ονομάτων και κωδικών. Συνήθως αυτό το τμήμα διασπάται έτσι ώστε τα πιστοποιητικά που αποθηκεύονται από άλλες εφαρμογές να μην έχουν πρόσβαση σε αυτά τα δεδομένα με πλασματικές ταυτότητες. Τέλος, οι δημιουργοί των εφαρμογών έχουν πρόσβαση σε πρωτόκολλα κρυπτογράφησης έτσι ώστε να τα χρησιμοποιούν για την περαιτέρω προστασία των δεδομένων των εφαρμογών τους. Τα δεδομένα μπορεί να είναι συμμετρικά κρυπτογραφημένα με τη χρήση αξιόπιστων μεθόδων όπως AES, RC4 και 3DES. Όταν ένας χρήστης επιχειρεί να αποκτήσει πρόσβαση σε εφαρμογές και δεδομένα μιας συσκευής, το iOS μπορεί να εξασφαλίσει ότι μόνο αξιόπιστοι χρήστες θα τα καταφέρουν.

7.2 Ασφάλεια του Android της Google

Το Android OS υιοθετεί ένα πρωτοφανές μοντέλο ασφάλειας το οποίο επικεντρώνεται στο να θέτει το χρήστη υπεύθυνο για τη συσκευή του. Όμως οι συσκευές Android δεν προέρχονται όλες από την ίδια εταιρεία αφού η ανοικτή φύση της πλατφόρμας επιτρέπει την ύπαρξη πολλών προτύπων, επεκτάσεων και αλλαγών σε αυτή. Οι επεκτάσεις αυτές θα μπορούσαν και να βοηθήσουν αλλά και να δυσκολέψουν την ασφάλεια με συνέπεια η ανάλυση του Android να αποτελεί ένα σημαντικό βήμα στην προστασία των πληροφοριών του συστήματος [33].

Το Android OS που είναι μια Linux πλατφόρμα που προγραμματίζεται στη γλώσσα Java, έχει τους δικούς του μηχανισμούς ασφαλείας [34]. Συνδυάζει στοιχεία λειτουργικού συστήματος όπως είναι η αποδοτική διαμοίραση μνήμης, multitasking

και δικαιώματα αρχείων. Το μοντέλο ασφαλείας το οποίο προκύπτει μοιάζει με ένα server πολλών χρηστών που συναντάται κυρίως σε J2ME και Blackberry πλατφόρμες.

Οι εφαρμογές του Android τρέχουν ανεξάρτητα και απομονωμένα η μια από την άλλη, σε διαφορετικές διεργασίες και με διαφορετικές ταυτότητες και διαφορετικά δικαιώματα η κάθε μια. Τα προγράμματα που εκτελούνται δεν μπορούν ούτε να διαβάσουν αλλά ούτε και να γράψουν το ένα στον κώδικα του άλλου ενώ η διαμοίραση δεδομένων μεταξύ τους πρέπει να γίνεται με προσοχή. Το περιβάλλον διασύνδεσης χρήστη του Android παρουσιάζει κάποια καινοτόμα χαρακτηριστικά τα οποία βοηθούν στην υποστήριξη της απομόνωσης μεταξύ των εφαρμογών. Το Android υποστηρίζει επίσης την κατασκευή εφαρμογών που χρησιμοποιούν τα χαρακτηριστικά του τηλεφώνου ενώ προστατεύουν τους χρήστες ελαχιστοποιώντας την ύπαρξη bugs και κακόβουλου κώδικα.

Η απομόνωση των διεργασιών του Android προλαμβάνει την ανάγκη για περίπλοκα αρχεία επιβεβαιώσεων για τα sandboxes. Έτσι οι εφαρμογές έχουν την ευελιξία να χρησιμοποιήσουν πηγαίο κώδικα χωρίς να μειώνουν την ασφάλεια του Android ή να δίνουν στις εφαρμογές επιπλέον δικαιώματα. Τα δικαιώματα που χρησιμοποιούνται στο Android δίνονται στην εφαρμογή για να της επιτρέψουν την πραγματοποίηση λειτουργιών όπως η λήψη φωτογραφιών, η χρήση του GPS και η πραγματοποίηση κλήσεων. Όταν εγκαθίσταται μια εφαρμογή παίρνει μια μοναδική ταυτότητα την οποία θα χρησιμοποιεί κάθε φορά που θα εκτελείται στη συγκεκριμένη συσκευή. Η ταυτότητα αυτή χρησιμοποιείται για την προστασία των δεδομένων ενώ ο δημιουργός της κάθε εφαρμογής πρέπει να είναι ξεκάθαρος σχετικά με το ποια δεδομένα μπορεί να διαμοιράζεται η εφαρμογή του.

Το κακόβουλο λογισμικό αποτελεί μια πραγματικότητα στις δημοφιλείς πλατφόρμες και τα χαρακτηριστικά του Android επιχειρούν να ελαχιστοποιήσουν τις επιδράσεις του. Όμως ακόμα και μια κακόβουλη εφαρμογή η οποία εγκαθίσταται στο Android τηλέφωνο ενός χρήστη χωρίς ιδιαίτερα σημαντικά δικαιώματα μπορεί να

καθυστερήσει κατά πολύ την εκτέλεση των διεργασιών. Σε αυτή την περίπτωση, ο ίδιος ο χρήστης της συσκευής είναι αυτός που πρέπει να εντοπίσει και να απομακρύνει από τη συσκευή του αυτή την εφαρμογή. Το Android από μόνο του προσπαθεί να προστατέψει τους χρήστες από εφαρμογές που εκτελούν επικίνδυνες λειτουργίες όπως: άμεση πραγματοποίηση τηλεφωνικών κλήσεων, διαγραφή δεδομένων, καταστροφή τηλεφωνικών καταλόγων κτλ ζητώντας την άδεια του χρήστη πριν την εκτέλεση των αντίστοιχων ενεργειών. Με τον τρόπο αυτό ο χρήστης υποψιάζεται τους σκοπούς της κακόβουλης εφαρμογής και προχωρά στην κατάργησή της.

Οι δημιουργοί εφαρμογών για Android συσκευές έχουν την ευθύνη της συγγραφής του κώδικά τους με τέτοιο τρόπο έτσι ώστε ο χρήστης να μην τεθεί σε κίνδυνο. Πρέπει λοιπόν να διασφαλίσουν τα δεδομένα που ο χρήστης πιθανόν να δίνει ως είσοδο στην εφαρμογή και να μην επιτρέπει σε κακόβουλες εφαρμογές να επηρεάσουν τα δικαιώματά τους ή να έχουν πρόσβαση στα δεδομένα τους.

Εκτός από τη μοναδική ταυτότητα κάθε εφαρμογής που εξασφαλίζει τη μεταξύ τους απομόνωση, το Android χρησιμοποιεί και μηχανισμούς δικαιωμάτων που επιβάλλουν περιορισμούς σε συγκεκριμένες λειτουργίες τις οποίες μπορεί να πραγματοποιήσει μια διεργασία [35].

Ένα κεντρικό σημείο σχεδίασης για την αρχιτεκτονική ασφαλείας του Android είναι το γεγονός ότι είναι προκαθορισμένο ότι καμιά εφαρμογή δεν έχει το δικαίωμα να εκτελέσει οποιεσδήποτε λειτουργίες που είναι δυνατόν να επηρεάσουν άλλες εφαρμογές, το λειτουργικό σύστημα ή το χρήστη. Αυτοί οι περιορισμοί περιλαμβάνουν την ανάγνωση/εγγραφή σε δεδομένα άλλων εφαρμογών, την ανάγνωση/εγγραφή των αρχείων των εφαρμογών, την πραγματοποίηση πρόσβασης στο δίκτυο και το να κρατούν σε ενεργή κατάσταση τη συσκευή συνεχώς.

Επειδή για τη διαμοίραση δεδομένων μεταξύ εφαρμογών απαιτείται η παραχώρηση δικαιωμάτων, αυτό γίνεται μέσω του χρήστη. Το Android δεν έχει μηχανισμό για τη δυναμική ανάθεση δικαιωμάτων επειδή με τον τρόπο αυτό θα έβαζε

σε κίνδυνο την ασφάλεια των δεδομένων του χρήστη. Ο πυρήνας είναι ο μόνος υπεύθυνος για την απομόνωση μεταξύ των εφαρμογών. Η Dalvik εικονική μηχανή δεν είναι ένα όριο ασφαλείας και κάθε εφαρμογή μπορεί να εκτελέσει δικό της κώδικα σε αυτήν. Όλοι οι τύποι των εφαρμογών (Java, native και υβριδικοί) τοποθετούνται σε sandbox [59] με τον ίδιο τρόπο και έχουν την ίδια βαθμίδα ασφαλείας.

Ακόμα μια δικλείδα ασφαλείας του Android είναι και η υπογραφή όλων των εφαρμογών του από το δημιουργό. Κάθε .apk αρχείο πρέπει να υπογραφεί με ένα πιστοποιητικό του οποίου το ιδιωτικό κλειδί δίνεται από το δημιουργό της εφαρμογής. Το πιστοποιητικό αυτό αρκεί να είναι self-signed, δηλαδή, μπορεί να είναι υπογεγραμμένο μόνο από τον εαυτό του. Ο λόγος για τον οποίο υπογράφονται οι Android εφαρμογές είναι για να αποκαλύπτονται οι δημιουργοί τους. Έτσι το σύστημα μπορεί να παραχωρήσει ή να αρνηθεί πρόσβαση μιας εφαρμογής σε δικαιώματα υψηλότερων επιπέδων και να αποδεχθεί ή να αρνηθεί την αίτηση μιας εφαρμογής να πάρει την ίδια ταυτότητα με μια άλλη εφαρμογή.

Μια βασική Android εφαρμογή δεν έχει καθόλου δικαιώματα που σχετίζονται μαζί της. Έτσι, δεν μπορεί να κάνει οτιδήποτε το οποίο θα μπορούσε να επηρεάσει το χρήστη ή τα δεδομένα της συσκευής. Για τη χρήση προστατευμένων χαρακτηριστικών της συσκευής πρέπει να συμπεριλαμβάνονται μέσα στο αρχείο AndroidManifest.xml ένα ή περισσότερα δικαιώματα τα οποία χρειάζεται η εφαρμογή. Κατά τη διάρκεια εγκατάστασης της εφαρμογής, τα δικαιώματα που απαιτεί η εφαρμογή θα της δοθούν από τον package installer αφού γίνει ο έλεγχος των υπογραφών της που ορίζουν αυτά τα δικαιώματα.

Μέχρι σήμερα έχουν γίνει αρκετές σημαντικές διορθώσεις που αφορούν την ασφάλεια από την έκδοση 2.3 του Android και έπειτα [36]. Πλέον, αποτρέπονται click jacking επιθέσεις από hackers. Με τον όρο click jacking αναφέρεται κανείς στην επίθεση η οποία ξεγελά το χρήστη σε μια ροή από clicks που είναι μεταμφιεσμένα με τέτοιο τρόπο ώστε να παρουσιάζονται στον χρήστη με τη μορφή χαρακτηριστικών της εφαρμογής. Επίσης, η πλατφόρμα είναι πλέον κατασκευασμένη έτσι ώστε να

συμπεριλαμβάνει ένα μηχανισμό φιλτραρίσματος ο οποίος ονομάζεται FilterTouchesWhenObscures για την αποφυγή επιθέσεων αυτού του τύπου. Ο μηχανισμός αυτός είναι υπεύθυνος να ειδοποιεί τους χρήστες όταν οι συσκευές τους δυσλειτουργούν σε κάποιες ευαίσθητες υπηρεσίες.

Μέχρι στιγμής έχουν εντοπιστεί αρκετά επικίνδυνα για εισβολή σημεία στο Android λειτουργικό σύστημα. Οτιδήποτε θεωρηθεί ύποπτο ή μη ασφαλές, αναφέρεται στην ομάδα της Google έτσι ώστε να μπορέσουν να αντιμετωπίσουν αυτά τα ζητήματα και να παρέχουν τη λύση τους στην επόμενη έκδοση λογισμικού ή στην επόμενη ενημέρωση για την τρέχουσα έκδοση [37]. Για παράδειγμα, στις 6 Μαρτίου 2008 εντοπίστηκε πρόβλημα στις βιβλιοθήκες επεξεργασίας εικόνων στο Android. Εντοπίστηκαν κάποια ευαίσθητα σημεία στο SDK τα οποία θα μπορούσαν να χρησιμοποιηθούν από hackers για εισβολή στο λογισμικό. Εκείνη την περίοδο είχε ανακοινωθεί στη συνέχεια ότι σε περίπτωση που κάποιος εντόπιζε αυτά τα σημεία και είχε κακόβουλες προθέσεις θα μπορούσε να αποκτήσει πλήρη έλεγχο στις συσκευές Android [38].

Λίγους μήνες αργότερα, στις 25 Οκτωβρίου 2008 εντοπίστηκε πρόβλημα στο φυλλομετρητή του Android ο οποίος επέτρεπε στα Trojans να εγκαθίστανται στην ίδια κατάτμηση ασφαλείας με το φυλλομετρητή. Με τον τρόπο αυτό, ένας εισβολέας θα μπορούσε να εγκαταστήσει λογισμικό το οποίο θα κατέγραφε τις ακολουθίες χαρακτήρων του χρήστη που αντιστοιχούσαν σε κωδικούς για να τις χρησιμοποιήσει στη συνέχεια για πρόσβαση σε προσωπικά δεδομένα [39].

Εντοπίστηκαν επίσης και bugs τα οποία θα μπορούσαν να οδηγήσουν σε Denial of Service επίθεση. Το πρώτο αφορά DoS ευαισθησία και αναφέρθηκε στις 24 Απριλίου 2009. Αυτό θα μπορούσε να οδηγήσει σε επανεκκίνηση των διεργασιών του συστήματος. Εντοπίστηκε στα πρωτόκολλα της Dalvik μηχανής και οι κακόβουλες εφαρμογές μπορούσαν να φτιαχτούν έτσι ώστε όταν τελείωνε η εγκατάστασή τους στη συσκευή να ξαναρχίζουν τις διεργασίες συστήματος προκαλώντας προβλήματα [40]. Το δεύτερο bug αφορά την απώλεια της συνδεσιμότητας και αναφέρθηκε για

πρώτη φορά στις 19 Ιουνίου 2009. Αφορά το χειρισμό γραπτών μηνυμάτων από το Android όπου ένα μήνυμα σε συγκεκριμένη, κακόβουλη μορφή θα μπορούσε να προκαλέσει απώλεια συνδεσιμότητας από το δίκτυο κινητής τηλεφωνίας στη συσκευή στην οποία υπάρχει. Το μετασχηματισμένο γραπτό μήνυμα αποτελείται κυρίως από ένα κακογραμμένο WAP Push μήνυμα το οποίο προκαλεί την εξαίρεση Java ArrayOutOfBoundsException στην εφαρμογή android.com.phone. Η εφαρμογή αυτή ξαναρχίζει μετά τη διακοπή «αθόρυβα» και χωρίς καμία ενημέρωση στο χρήστη οδηγώντας σε απώλεια συνδεσιμότητας[40].

Ένα από τα πλεονεκτήματα του Android είναι ότι είναι ανοικτού κώδικα. Παρόλο που αυτό μπορεί να προκαλέσει κάποιες ανησυχίες σχετικά με την ασφάλεια των προσφερόμενων εφαρμογών δεν παύει από το να είναι στην ουσία ένα αποτρεπτικό μέτρο εισβολής hackers. Η κοινότητα του Android αναφέρει οποιοδήποτε πρόβλημα εντοπίσει στην Google η οποία με τη σειρά της το αντιμετωπίζει εκδίδοντας ενημερώσεις αλλά και συνεχείς εκδόσεις ολοένα και πιο ασφαλούς λογισμικού.

7.3 Σύγκριση ανάμεσα στα δύο λειτουργικά συστήματα σε θέματα ασφάλειας παροχής εφαρμογών

Μετά από την εξέταση των κυριότερων θεμάτων ασφαλείας για το κάθε λειτουργικό σύστημα ξεχωριστά, ακολουθεί μια σύγκριση σχετικά με την ασφάλεια των εφαρμογών που διανέμει το καθένα. Η διανομή των εφαρμογών στο iOS γίνεται μέσω του Apple App Store και στο Android OS μέσω του Android Market τα οποία αναλύονται σε επόμενο κεφάλαιο.

Από την πλευρά της η Apple επιμένει και προσπαθεί να διασφαλίσει την εικόνα που έχει στην αγορά απαγορεύοντας την έκδοση και διανομή εφαρμογών συγκεκριμένου τύπου. Υποστηρίζει ότι μέσα στο App Store δεν υπάρχουν κακόβουλες εφαρμογές αλλά αυτό δεν αποδεικνύεται πάντοτε αληθές. Για να διανέμει ένας δημιουργός την εφαρμογή του στο App Store πρέπει να την καταχωρήσει για

έγκριση από την ομάδα ανάπτυξης. Επίσης, οι δημιουργοί εφαρμογών για το iPhone υποχρεούνται να πληρώνουν ένα ετήσιο ποσό συνδρομής. Ο λόγος είναι ότι η Apple επιθυμεί να έχει πολύ υψηλά επίπεδα εφαρμογών στο τι θεωρούν οι ίδιοι κατάλληλο για το App Store [32]. Η εταιρεία υποστηρίζει ότι οποιουδήποτε είδους κακόβουλη, ακατάλληλη αλλά και πιθανή για υποκλοπή προσωπικών δεδομένων εφαρμογή δεν θα καταλήξει σε κανένα από τους χρήστες των iPhone μέσω του App Store. Είναι επίσης γεγονός ότι η Apple απορρίπτει το 10% των αιτήσεων για εφαρμογές που έχει ετησίως για τους πιο πάνω λόγους.

Η Google από την άλλη δεν παρουσιάζεται τόσο αυστηρή όσον αφορά τον έλεγχο των νέων εφαρμογών. Παρόλο που υπάρχει η αγορά για Android εφαρμογές δωρεάν και μη, δεν απαγορεύει την έκδοση εφαρμογών οποιουδήποτε περιεχομένου και οποιαδήποτε λειτουργία και αν εκτελούν ή οποιοδήποτε περιεχόμενο και να παρέχουν. Όμως, παρουσιάζεται πανέτοιμη στην αντίδραση για οποιαδήποτε καταχώρηση σχετικά με κακόβουλες εφαρμογές ή ευαίσθητα σημεία του συστήματος. Έχει υπάρξει ακόμα και η κατασκευή μιας απομακρυσμένης εφαρμογής για Android η οποία χρησιμοποιήθηκε για την απομάκρυνση συγκεκριμένης κακόβουλης εφαρμογής από κινητά τηλέφωνα Android. Λόγω του ότι οποιαδήποτε εφαρμογή επιθυμεί μπορεί να διακινηθεί ελεύθερα στο Android Market χωρίς παρακολούθηση και έλεγχο οι χρήστες έχουν στη διάθεσή τους καινοτόμες εφαρμογές που στην περίπτωσή τους η Apple θα μπορούσε να τις απορρίψει.

Η ασφάλεια στις εφαρμογές του Android Market επιτυγχάνεται μέσω των ελέγχων που κάνει το σύστημα σχετικά με τους πόρους και τα δικαιώματα που διεκδικεί η κάθε εφαρμογή. Δίνει έτσι τη δυνατότητα στον καθένα να παρουσιάσει την εφαρμογή του, σε περίπτωση όμως εντοπισμού ανωμαλιών ενεργεί άμεσα με απομάκρυνσή της από την αγορά.

Το Android βέβαια στερείται κάποια χαρακτηριστικά ασφαλείας που υπάρχουν στο iOS όπως για παράδειγμα: multilayer protection, απομακρυσμένη διαγραφή δεδομένων καθώς και παρακολούθηση στην αγορά του. Οι νέες εκδόσεις και των δύο

λειτουργικών συστημάτων θα φανερώσουν τα επόμενα βήματα στην εξασφάλιση των χρηστών και των δεδομένων τους.

Ακολουθεί ένας συγκριτικός πίνακας στα θέματα ασφάλειας που εξετάστηκαν για τα δύο λειτουργικά συστήματα.

	iOS	**Android OS**
Προσέγγιση	Ελαχιστοποίηση της έκθεσης της συσκευής σε ευπάθειες	Ο χρήστης είναι υπεύθυνος για την ασφάλεια της συσκευής του
Third-party εφαρμογές	Απαγορεύεται η εγκατάστασή τους	Επιτρέπεται η εγκατάστασή τους
Μειονεκτήματα	• Όλες οι σημαντικές διεργασίες εκτελούνται ως super user με δικαιώματα διαχειριστή • Στοίβα, σωρός και εκτελέσιμος κώδικας βρίσκονται στο ίδιο σημείο μνήμης σε κάθε εκτέλεση	• Ο χρήστης είναι υπεύθυνος να εντοπίσει και να διαγράψει τις κακόβουλες εφαρμογές
Χαρακτηριστικά ασφάλειας	• Geo Location • Auto erase • Data encryption	• Μοναδική ταυτότητα για κάθε δημιουργό εφαρμογής • Auto erase (σε συγκεκριμένα μοντέλα)
Ασφάλεια που παρέχεται από το λειτουργικό σύστημα	• Χρήση πολιτικών ασφαλείας (security policies) • Προφίλ επιβεβαίωσης • Κρυπτογράφηση • Sandboxed εφαρμογές	• Sandboxed εφαρμογές • Κάθε εφαρμογή έχει τη δική της μοναδική ταυτότητα • Μηχανισμοί δικαιωμάτων

Πίνακας 7. 1 Σύγκριση των κυριότερων χαρακτηριστικών ασφάλειας των δύο λειτουργικών συστημάτων

.

Κεφάλαιο 8

Σύγκριση χαρακτηριστικών λειτουργίας του iPhone (Apple's iOS) και των συσκευών με λογισμικό Android (Google's)

8.1 Διεπαφή χρήστη (User interface)

8.1.1 Διεπαφή χρήστη στο iPhone

Η διεπαφή χρήστη του iOS βασίζεται στην κεντρική οθόνη του που ονομάζεται επίσης και Home-screen [41]. Η οθόνη αυτή περιέχει μια γραφική λίστα με όλες τις διαθέσιμες εφαρμογές. Οι εφαρμογές του iPhone τρέχουν συνήθως μια κάθε φορά εκτός από το iOS 4 όπου επιτρέπει την εκτέλεση εφαρμογών και στο παρασκήνιο. Η οθόνη Home είναι προσβάσιμη ανά πάσα στιγμή μέσω του κουμπιού που βρίσκεται από κάτω της όπως φαίνεται και στην εικόνα 8.1 τερματίζοντας την διεργασία της εφαρμογής που εκτελείται.

Εικόνα 8. 1 Η διεπαφή χρήστη του iPhone 4

Είναι προκαθορισμένο να εμφανίζονται στην οθόνη τα εικονίδια των εφαρμογών Messages, Calendar, Photos, Camera, YouTube, Stocks, Maps, Weather, Voice Memos, Notes, Clock, Calculator, Settings, iTunes, App Store και Compass. Στο κάτω μέρος της οθόνης εμφανίζονται πάντα τα τέσσερα εικονίδια για τις εφαρμογές Phone, Mail, Safari και iPod που αναπαριστούν και τις κύριες λειτουργίες του iPhone.

Στις 15 Ιανουαρίου του 2008 η Apple έδωσε στο κοινό την αναβάθμιση λογισμικού 1.1.3 η οποία έδινε τη δυνατότητα στους χρήστες να αλλάζουν την τοποθέτηση των εφαρμογών στη Home screen και μάλιστα σε μέχρι εννέα διαδοχικές οθόνες. Η εναλλαγή από οθόνη σε οθόνη γίνεται με οριζόντια ολίσθηση του χεριού στην οθόνη.

Στην κύρια οθόνη παρέχεται επίσης η δυνατότητα στους χρήστες να αλλάζουν τα εικονίδια των τεσσάρων πιο συχνών εφαρμογών που επιθυμούν. Κάθε οθόνη μπορεί να φιλοξενήσει μέχρι και δεκαέξι εικονίδια ενώ δεν χρειάζεται να γεμίσει η μια για να προχωρήσει στην επόμενη ο χρήστης, επιτρέποντας με τον τρόπο αυτό την κατηγοριοποίηση των εφαρμογών.

Σχεδόν όλη η είσοδος του χρήστη δίνεται μέσω της οθόνης αφής η οποία αντιλαμβάνεται και σύνθετες χειρονομίες αφού υποστηρίζει multi-touch. Οι τεχνικές αλληλεπίδρασης του iPhone επιτρέπουν στο χρήστη να μετακινεί το περιεχόμενο προς τα πάνω ή προς τα κάτω μέσω αγγίγματος και σύρσιμο του δακτύλου στην οθόνη.

Η διεπαφή χρήστη του iPhone προσομοιώνει τη φυσική υπόσταση ενός πραγματικού αντικειμένου. Επίσης η οθόνη μπορεί να αλλάξει τον προσανατολισμό του περιεχομένου της προς όποια κατεύθυνση επιλέξει ο χρήστης να κρατήσει τη συσκευή του. Τα μενού της κάθε εφαρμογής εμφανίζονται στο πάνω και στο κάτω μέρος της οθόνης όταν είναι αναγκαία. Οι επιλογές που παρέχονται σε κάθε μενού διαφέρουν ανάλογα με το είδος του προγράμματος αλλά ακολουθούν πάντοτε ένα σταθερό στυλ μοντελοποίησης. Σε ιεραρχίες μενού υπάρχει ένα κουμπί επιστροφής

(back) στην πάνω αριστερή γωνία της οθόνης που εμφανίζει το όνομα του γονιού του στοιχείου το οποίο εμφανίζει την τρέχουσα οθόνη.

8.1.2 Διεπαφή χρήστη στο Android

Το Android OS υποστηρίζεται από πολλές συσκευές διάφορων κατασκευαστών και για το λόγο αυτό υπάρχουν διαφοροποιήσεις όσον αφορά τις διεπαφές χρήστη που χρησιμοποιεί η κάθε συσκευή. Παρόλα αυτά, υπάρχουν κάποια κύρια χαρακτηριστικά που είναι κοινά σε όλες.

Αρχικά, η οθόνη σε μια android συσκευή δεν υποστηρίζει πάντα την εμφάνιση εικονιδίων των εφαρμογών. Συνήθως πρόκειται για μια οθόνη στην οποία ο χρήστης προσθέτει widgets με κύριες λειτουργίες και υπηρεσίες όπως: εμφάνιση ώρας, αλλαγή προφίλ κτλ. Το μενού του Android εμφανίζεται όταν ο χρήστης πιέσει το κεντρικό πλήκτρο της συσκευής το οποίο διαφέρει από μοντέλο σε μοντέλο.

Η οθόνη που εμφανίζεται περιλαμβάνει τις εφαρμογές που βρίσκονται εγκαταστημένες στο τηλέφωνο. Είναι προκαθορισμένο έτσι ώστε κάθε Android συσκευή να περιλαμβάνει αρχικά τις εφαρμογές: Contacts, Phone Calls, Messages, Internet, Camera, Album, Email, Market, Alarm Clock, Calendar, Backup and Synchronization, Music Player, Notes, Video, YouTube, Maps και Gmail. Όπως και στο iPhone, έτσι και στο Android η μετακίνηση μεταξύ των οθόνων γίνεται κυρίως μέσω αφής ενώ επιτρέπεται και η αναδιάρθρωση των εικόνων των εφαρμογών. Ο χρήστης μπορεί να κρατήσει στη συσκευή μέχρι και εφτά διαδοχικές οθόνες με εφαρμογές. Ανάλογα με το μέγεθος της οθόνης της συσκευής τοποθετούνται και τα ανάλογα εικονίδια σε κάθε μια.

Οι κυριότερες εφαρμογές τις οποίες ο χρήστης χρησιμοποιεί τοποθετούνται στην αρχική οθόνη και με τρόπο που διαφέρει από συσκευή σε συσκευή. Για παράδειγμα, όπως φαίνεται και στην εικόνα 8.2, στο Xperia x10 mini της SonyErricsson οι πιο συχνά χρησιμοποιούμενες εφαρμογές μπορούν να τοποθετηθούν στα τέσσερα άκρα

της οθόνης και είναι προσβάσιμες χωρίς να χρειάζεται το μενού εφαρμογών. Εννοείται ότι ο κάθε χρήστης μπορεί να βάλει τις εφαρμογές που θέλει αλλά υπάρχουν κάποιες εφαρμογές από το Android Market, που είναι συνήθως παιχνίδια, τα οποία δεν μπορούν να μπουν σε αυτή τη θέση.

Εικόνα 8. 2 Η διεπαφή χρήστη του Xperia x10 mini. Αριστερά εμφανίζεται η κύρια οθόνη και δεξιά το μενού εφαρμογών

Το σύστημα χρησιμοποιεί την μπάρα ειδοποιήσεων για να εμφανίζει στο χρήστη μηνύματα σχετικά με την κατάσταση του συστήματος. Αυτή η μπάρα επεκτείνεται εάν ο χρήστης την τραβήξει προς τα κάτω για να εμφανίσει όλες τις ειδοποιήσεις. Κάθε χρονική στιγμή στην μπάρα ειδοποιήσεων εμφανίζεται μόνο η τελευταία ειδοποίηση.

8.1.3 Σύγκριση των διεπαφών των δύο λειτουργικών συστημάτων

Είναι πολλές οι ομοιότητες που παρουσιάζουν τα δύο λειτουργικά συστήματα στις διεπαφές τους αλλά και αρκετές οι διαφορές τους. Καταρχήν, και τα δύο λειτουργικά συστήματα υποστηρίζουν την εγκατάσταση νέων εφαρμογών και επομένως και την οργάνωσή τους ανάλογα με τη σειρά που επιλέγει ο χρήστης να εμφανίζονται. Επίσης, και στα δύο λειτουργικά συστήματα υπάρχει η δυνατότητα καθορισμού των πιο χρήσιμων εφαρμογών έτσι ώστε ο χρήστης να μπορεί να τις

χρησιμοποιεί με ευκολία χωρίς να επιλέξει το μενού εφαρμογών. Τόσο στο iOS όσο και στο Android OS επιτρέπεται η οργάνωση των εφαρμογών σε σελίδες έτσι ώστε ο χρήστης να ομαδοποιεί εφαρμογές ανάλογα με την κατηγορία τους και τη συχνότητα χρήσης τους.

Το iOS στα τηλέφωνα iPhone δεν επιτρέπει στις αρχικές του ρυθμίσεις την εγκατάσταση θεμάτων (themes) στο τηλέφωνο, ενώ στο Android η προσωποποίηση (personalization) γίνεται με εύκολο τρόπο τόσο μέσω των διαθέσιμων θεμάτων που βρίσκονται προ-εγκατεστημένα σε κάθε συσκευή όσο και μέσω θεμάτων που υπάρχουν ως εφαρμογές στο Android Market. Επιπλέον, τα δύο λειτουργικά συστήματα παρουσιάζουν διαφορές στην ειδοποίηση του χρήστη σχετικά με συμβάντα που προκύπτουν κατά τη διάρκεια λειτουργίας των εφαρμογών. Για παράδειγμα, όπως έχει ήδη αναφερθεί, στις Android συσκευές χρησιμοποιείται η μπάρα ειδοποιήσεων στην οποία εμφανίζεται το αντίστοιχο μήνυμα για κάθε γεγονός, π.χ. εισερχόμενο μήνυμα, χαμένη κλήση, λήψη email κτλ. Από την άλλη, στο iPhone ο χρήστης ενημερώνεται για τα αντίστοιχα θέματα μέσω του εικονιδίου της κάθε εφαρμογής ξεχωριστά. Για παράδειγμα σε περίπτωση εισερχόμενου μηνύματος, ο αριθμός ‘1’ εμφανίζεται στο εικονίδιο της εφαρμογής μηνυμάτων για να ειδοποιηθεί ο χρήστης.

Η σύγκριση των διεπαφών των δύο λειτουργικών συστημάτων παρουσιάζεται και στο τέλος του κεφαλαίου στον Πίνακα 8.1.

8.2 Hardware που συναντάται

Στην ενότητα αυτή γίνεται αναφορά στις δύο πιο πρόσφατες συσκευές που αντιπροσωπεύουν τα λειτουργικά συστήματα που εξετάζονται. Για το iOS η συσκευή που παρουσιάζεται είναι το κινητό τηλέφωνο iPhone 4G [60] ενώ για το λειτουργικό σύστημα Android χρησιμοποιούνται τα χαρακτηριστικά του κινητού τηλεφώνου

Google Nexus S [61]. Συγκριτική ανάλυση των χαρακτηριστικών υπάρχει και στον Πίνακα 10.1 στο τέλος του κεφαλαίου 10.

8.2.1 Το υλικό που υπάρχει στο iPhone

Επεξεργαστής: 1 GHz ARM Cortex-A8 επεξεργαστής, PowerVR SGX535GPU, Apple A4 chipset.

Μνήμη και δευτερεύον αποθηκευτικός χώρος: Η μνήμη RAM στο iPhone είναι 512MB αλλά υποστηρίζει επιπλέον αποθηκευτικό χώρο μέχρι 32GB.

Οθόνη: Πρόκειται για μια οθόνη 9cm υγρών κρυστάλλων αρχικά και πλέον Led τεχνολογίας (στο iPhone 4). Περιέχει γυαλί που την προστατεύει από γδαρσίματα και απεικονίζει πλέον16 εκατομμύρια χρώματα. Η οθόνη αφής είναι multi-touch αίσθησης και οι χειρονομίες αναγνωρίζονται με τεχνολογία που αναπτύχθηκε από την FingerWorks.

Αισθητήρες: Το iPhone έχει τρεις αισθητήρες. Ο proximity sensor απενεργοποιεί την οθόνη για εξοικονόμηση μπαταρίας και την αποτροπή μη ηθελημένων εισόδων από το χρήστη. Ο ambient light sensor ρυθμίζει τη φωτεινότητα της οθόνης για εξοικονόμηση ενέργειας. Ο accelerometer sensor τριών αξόνων διαισθάνεται τον προσανατολισμό του τηλεφώνου και αλλάζει ανάλογα την οθόνη. Υπάρχει επίσης εγκατεστημένος και ο moisture sensor ο οποίος είναι ικανός να ανιχνεύσει κατά πόσο η συσκευή έχει επηρεαστεί ή έχει πάθει βλάβη από νερό.

Πολυμέσα και Δίκτυο: Τα τμήματα ήχου μπορούν να είναι δυναμικά προσβάσιμα από την αντίστοιχη εφαρμογή του iPhone. Επιτρέπουν στο χρήστη να προσθέσει ενδιαφέροντα ακουστικά χαρακτηριστικά. Το iPhone περιέχει τμήματα ήχου που υποστηρίζουν mixing, equalization, format conversion, I/O for recording και αναπαραγωγή. Το ενσωματωμένο Bluetooth2.x + EDR υποστηρίζει ασύρματα ακουστικά. Από την έκδοση 3.0 και αργότερα υποστηρίζεται στερεοφωνικός ήχος.

Μπαταρία: το iPhone περιλαμβάνει μια εσωτερική, επαναφορτιζόμενη μπαταρία η οποία δεν μπορεί να αντικατασταθεί από το χρήστη. Η μπαταρία φορτίζει όταν το

τηλέφωνο είναι συνδεδεμένο με υπολογιστή μέσω USB καλωδίου. Η μπαταρία είναι σχεδιασμένη να διατηρεί το 80% της αρχικής της χωρητικότητας μετά από 400 πλήρεις κύκλους φόρτισης. Επίσης είναι ικανή να προσφέρει μέχρι 250 ώρες αναμονής, μέχρι 24 ώρες αναπαραγωγής μουσικής, μέχρι 7 ώρες αναπαραγωγής βίντεο, μέχρι 6 ώρες web browsing και μέχρι 8 ώρες ομιλίας.

Κάμερα: το ιPhone έχει ενσωματωμένη κάμερα 5 megapixel (στο iPhone 4)που βρίσκεται στο πίσω μέρος του για τη λήψη ψηφιακών φωτογραφιών. Περιέχει πλέον οπτικό ζουμ, flash και auto-focus και από την έκδοση iPhone 3GS υποστηρίζεται και η λήψη βίντεο. Το βίντεο στη συνέχεια μπορεί να φορτωθεί στο YouTube ή σε άλλες υπηρεσίες απευθείας.

8.2.2 Το υλικό συναντάται στη συσκευή Google Nexus S

Το Android OS χρησιμοποιείται ως λειτουργικό σύστημα για πολλά κινητά τηλέφωνα διαφόρων εταιρειών όπως Motorola, CLIQ, Samsung, Sony Erricsson κτλ[42]. Μια από τις πιο πρόσφατες συσκευές που τρέχουν λειτουργικό σύστημα Android είναι η συσκευή Google Nexus S της οποίας και τα χαρακτηριστικά παρουσιάζονται στη συνέχεια.

Επεξεργαστής: 1GHz Cortex A8 (Hummingbird) επεξεργαστής της εταιρείας Samsung.

Μνήμη και δευτερεύον αποθηκευτικός χώρος: Έχει μνήμη RAM μέχρι 512 MB στις οποίες είναι δυνατή η αποθήκευση και εφαρμογών από το Android Market. Ο δευτερεύον αποθηκευτικός χώρος ανέρχεται στα 16GB (iNAND flash memory) και χρησιμοποιείται για την αποθήκευση εικόνων, μουσικών αρχείων και βίντεο.

Κάμερα: 5.0 megapixels κάμερα. Υποστηρίζεται επίσης και οπτικό ζουμ μέχρι 4x. Περιλαμβάνονται χαρακτηριστικά όπως image stabilization, real-time color effects, scene modes και location tagging.

Αισθητήρες: περιέχονται τρεις αισθητήρες: proximity sensor, ambient light sensor και eCompass sensor

Οθόνη: Διαφέρει επίσης από συσκευή σε συσκευή. Μπορεί να είναι από 2.55 cm μέχρι 4.3cm. Σχεδόν όλες οι συσκευές υποστηρίζουν 16 εκατομμύρια χρώματα και έχουν TFT οθόνες.

Μπαταρία: ανάλογα με τη συσκευή υποστηρίζονται διαφορετικοί χρόνοι αναμονής και συνεχής χρήσης. Υπάρχουν συσκευές στις οποίες η μπαταρία μπορεί να αντικατασταθεί αλλά και συσκευές στις οποίες αυτό δεν μπορεί να γίνει.

Πολυμέσα και Δίκτυο: Υποστηρίζει αναπαραγωγή ήχου και βίντεο καθώς και πολλές μορφές τους όπως: AAC, MP3, WAV, MIDI κτλ. Υπάρχει υποστήριξη Bluetooth, GPS και WiFi.

8.3 Διαθέσιμο SDK

8.3.1 iOS SDK

Με το SDK οι προγραμματιστές μπορούν να αναπτύξουν εφαρμογές που τρέχουν στο iPhone και στο iPod. Σε αυτό συμπεριλαμβάνονται Xcode, IDE, Instruments, iPhone simulator, frameworks and samples, compilers, Shark analysis tool κτλ. Το iOS SDK περιέχει περισσότερα από 1000 νέα πρωτόκολλα και παρέχει στους δημιουργούς εφαρμογών νέες δυνατότητες για να εμπλουτίσουν τη λειτουργικότητα των εφαρμογών τους. Τα νέα πρωτόκολλα παρέχουν επίσης και υποστήριξη για την επικοινωνία των εφαρμογών με τα αξεσουάρ στα οποία είναι συνδεδεμένη η συσκευή. Σημαντική λεπτομέρεια αποτελεί το γεγονός ότι οι ενδιαφερόμενοι για την ανάπτυξη εφαρμογών στο iOS πρέπει αρχικά να αγοράσουν το SDK. Επιπλέον, το SDK απαιτεί ένα Intel Mac υπολογιστή που τρέχει τουλάχιστον Mac OS S Leopard έκδοση λογισμικού ενώ λειτουργικά συστήματα της Microsoft δεν υποστηρίζονται.

8.3.2 Android SDK

Το Android SDK περιλαμβάνει ένα δυνατό σύνολο εργαλείων ανάπτυξης εφαρμογών που περιλαμβάνουν μεταξύ άλλων debugger, libraries, documentation, sample codes και tutorials. Υποστηρίζει τις υπολογιστικές πλατφόρμες με αρχιτεκτονική x-86 που τρέχουν Linux, Mac OS X, Windows XP, Windows Vista και Windows 7. Το SDK παρέχεται δωρεάν στους ενδιαφερόμενους από την ιστοσελίδα Android Developers [62] και απαιτεί το περιβάλλον ανάπτυξης να περιέχει το Java Development Kit. Το επίσημο IDE ανάπτυξης Android εφαρμογών είναι το Eclipse σε συνδυασμό με το Android Development Tools plug-in.

8.4 Ενημερώσεις λογισμικού και εφαρμογών

8.4.1 Ενημέρωση στο iPhone

Η ενημέρωση λογισμικού στο iPhone γίνεται με πολύ εύκολο τρόπο. Ο χρήστης της συσκευής συνδέει το τηλέφωνό του με τον υπολογιστή του και μπορεί να ενημερώσει το λογισμικό μέσω του iTunes [43]. Όταν ο χρήστης συνδέσει το τηλέφωνο και εκκινήσει το iTunes πρέπει να κάνει αναζήτηση για τυχόν ενημερώσεις. Σε περίπτωση που υπάρχει ενημέρωση για το λογισμικό. Σε περίπτωση που υπάρχει τότε ο χρήστης πρέπει να κατεβάσει το λογισμικό μέσω του iTunes. Υπάρχει η απαίτηση το τηλέφωνο να είναι συνεχώς συνδεδεμένο με τον υπολογιστή για όση ώρα διαρκεί η ενημέρωση.

Όσον αφορά την ενημέρωση εφαρμογών που βρίσκονται εγκαταστημένες στο τηλέφωνο, αυτή γίνεται αυτόματα. Ο πρώτος τρόπος είναι ο ίδιος με την ενημέρωση λογισμικού και γίνεται μέσω του iTunes όταν το τηλέφωνο συνδεθεί με τον υπολογιστή. Ο δεύτερος και πιο απλός είναι η ενημέρωση απευθείας από το τηλέφωνο μέσω του App Store. Όταν ο χρήστης είναι συνδεδεμένος στο διαδίκτυο η εφαρμογή App Store τον ενημερώνει αν υπάρχουν ενημερώσεις για τις εφαρμογές

του. Σε περίπτωση που υπάρχουν ο χρήστης επιλέγει την ενημέρωσή τους και αυτή γίνεται αυτόματα.

8.4.2 Ενημέρωση στο Android

Παρόμοια διαδικασία με το iPhone ακολουθείται και στις Android συσκευές. Ο χρήστης ενημερώνεται για διαθέσιμη ενημέρωση λογισμικού με ειδοποίηση στο τηλέφωνό του ή με ειδοποίηση στον υπολογιστή όταν ενώσει το τηλέφωνο. Κάθε εταιρεία διανέμει μαζί με την Android συσκευή και το αντίστοιχο δωρεάν λογισμικό υπολογιστή για το χειρισμό του τηλεφώνου.

Όταν ο χρήστης ενημερωθεί για νέα έκδοση λογισμικού υπάρχουν δύο πιθανά σενάρια. Το πρώτο αφορά στην απλή ενημέρωση της υπάρχουσας έκδοσης του λογισμικού. Σε αυτή την περίπτωση ο χρήστης επιλέγει την αυτόματη ενημέρωση του τηλεφώνου. Όταν αυτή ολοκληρωθεί, ο χρήστης πρέπει να αποσυνδέσει το τηλέφωνο από τον υπολογιστή και να το εκκινήσει. Συνήθως η πρώτη εκκίνηση μετά από ενημέρωση λογισμικού διαρκεί αρκετό χρονικό διάστημα. Σε αυτή την ενημέρωση δεν επηρεάζονται καθόλου τα δεδομένα του χρήστη και οι εφαρμογές που έχει εγκαταστήσει στο τηλέφωνο.

Στη δεύτερη περίπτωση βρίσκεται η εγκατάσταση στο τηλέφωνο της νεότερης έκδοσης Android λογισμικού. Προτού γίνει η ενημέρωση ο χρήστης καλείται να αποθηκεύσει όλα τα δεδομένα του κάνοντας backup μέσα από την αντίστοιχη εφαρμογή στο τηλέφωνο. Ακολούθως γίνεται η εγκατάσταση όπου παίρνει αρκετό χρόνο. Στη συνέχεια, ο χρήστης πρέπει να μεταφέρει τα δεδομένα του πίσω στο τηλέφωνο. Αυτή η διαδικασία μπορεί να διαρκέσει αρκετές ώρες, ανάλογα με τα δεδομένα του χρήστη. Το μειονέκτημα είναι ότι υπάρχει περίπτωση κάποιες από τις εφαρμογές να χρειάζονται εγκατάσταση από την αρχή. Επίσης ρύθμιση χρειάζονται και το ημερολόγιο, το ξυπνητήρι, τα προφίλ του χρήστη και γενικά οτιδήποτε είχε τροποποιηθεί στην προηγούμενη έκδοση του λογισμικού.

Λόγω των πολλών Android συσκευών που κυκλοφορούν από διάφορους κατασκευαστές, αντιμετωπίζεται ένα σημαντικό πρόβλημα σχετικά με την έκδοση λογισμικού μιας και αρκετές συσκευές δεν υποστηρίζουν πάντα την πιο πρόσφατη έκδοση λογισμικού που παρέχεται για το Android OS. Για το λόγο αυτό, η Google μαζί με το OHA προσπαθούν να καθορίσουν μια νέα «anti-fragmentation policy» σύμφωνα με την οποία θα καθορίζεται ένα συγκεκριμένο χρονικό διάστημα στο οποίο θα πρέπει όλες οι συσκευές να έχουν τη νέα έκδοση του λειτουργικού συστήματος που παρέχεται από την Google. Το γεγονός αυτό δεν απευθύνεται στους χρήστες των συσκευών αλλά στις εταιρείες που τα κατασκευάζουν οι οποίες θα πρέπει να ακολουθούν τις οδηγίες της Google και να φροντίζουν να παρέχουν τη νέα έκδοση λογισμικού στους πελάτες τους. Στόχος είναι να εκλείψει το έντονο χάσμα μεταξύ των εκδόσεων Android λογισμικού που χρησιμοποιούνται στις συσκευές.

Η ενημέρωση των εφαρμογών στο Android είναι πολύ εύκολη υπόθεση. Όταν ο χρήστης είναι συνδεδεμένος στο διαδίκτυο λαμβάνει ειδοποιήσεις ότι υπάρχουν διαθέσιμες ενημερώσεις για τις εφαρμογές που έχει εγκαταστήσει. Μέσα από το Android Market και πιο συγκεκριμένα στην κατηγορία «οι εφαρμογές μου» μπορεί να επιλέξει τις εφαρμογές για τις οποίες θα πάρει ενημέρωση.

8.5 Mobile Internet και Networking Support

Τόσο στο iPhone όσο και στις Android συσκευές χρησιμοποιούνται WiFi και Bluetooth για πρόσβαση στο ίντερνετ και για επικοινωνία [6].

WiFi: το WiFi είναι το πρότυπο 802.11. Πρόκειται για ένα πρωτόκολλο που λειτουργεί σε δύο συχνότητες. Αρχικά όλα τα WiFi χρησιμοποιούσαν 2.412 – 2.472 GHz. Το 802.11n πρωτόκολλο πρόσθεσε την ικανότητα χρήσης από 5.15 μέχρι 5.25 GHz όμως το iPhone είναι περιορισμένο στο 802.11b/g. Η στοίβα δικτύου στο OS περιλαμβάνει αρκετές διεπαφές για τις συσκευές iPhone και iPod. Από την πλευρά του το Android δίνεται στην αγορά με πλήρη υποστήριξη WiFi συνδεσιμότητας. Το

κύριο συστατικό του είναι ο WiFiManager ο οποίος εξασφαλίζει τη σύνδεση στο διαδίκτυο όταν του ζητηθεί.

Bluetooth: Το Bluetooth είναι ένα πρωτόκολλο το οποίο δουλεύει στις συχνότητες 2.40 – 2.485 GHz. Αποτρέπει τις παρεμβολές με άλλα συστήματα που βρίσκονται στην ίδια συχνότητα επειδή λειτουργεί με πολύ μικρή ισχύ. Έχει μέγιστη απόσταση κάλυψης τα δέκα μέτρα. Η πλατφόρμα Android υποστηρίζει τη στοίβα λογισμικού του Bluetooth η οποία επιτρέπει στη συσκευή να ανταλλάσσει δεδομένα ασύρματα με άλλες Bluetooth συσκευές. Το framework το εφαρμογών παρέχει πρόσβαση στις λειτουργίες του Bluetooth μέσω των πρωτοκόλλων του Android Bluetooth. Τα πρωτόκολλα αυτά επιτρέπουν την ασύρματη σύνδεση με άλλες συσκευές επιτρέποντας την επικοινωνία από σημείο σε σημείο αλλά και την επικοινωνία μεταξύ πολλών. Στο iPhone υποστηρίζεται η τεχνολογία Bluetooth, δεν επιτρέπεται όμως η αποστολή αρχείων μεταξύ των συσκευών. Χρησιμοποιείται δηλαδή μόνο για ασύρματη επικοινωνία και όχι για ανταλλαγή δεδομένων.

8.6 Διαχείριση ενέργειας

8.6.1 Διαχείριση ενέργειας στο iPhone

Το iPhone δεν έχει σύνολο εργαλείων για τη διαχείριση της κατανάλωσης ισχύς. Η λειτουργία αυτή ενσωματώνεται μέσα στο core επίπεδο (πυρήνα) γεγονός το οποίο εξοικονομεί ενέργεια θέτοντας το σύστημα σε αναμονή ή σε sleep κατάσταση όταν χρειάζεται. Τα προγράμματα τα οποίο καταναλώνουν την περισσότερη ενέργεια στο iPhone είναι το 3G radio σύστημα, το WiFi, το 2G radio σύστημα, το Bluetooth και το GPS.

Ο περιορισμός αυτών των εφαρμογών εξοικονομεί ενέργεια για το σύστημα. Όταν το iPhone περνά στην κατάσταση sleep, αυτόματα απενεργοποιούνται το WiFi και ο φωτισμός της οθόνης.

8.6.2 Διαχείριση ενέργειας στις Android συσκευές

Το Android υποστηρίζει τη δική του διαχείριση ενέργειας που είναι σχεδιασμένη έτσι ώστε να εξασφαλίζει ότι ο επεξεργαστής δεν θα καταναλώνει ενέργεια αν δεν απαιτείται από εφαρμογές και υπηρεσίες. Το Android απαιτεί οι εφαρμογές και οι υπηρεσίες να ζητούν πόρους μέσω wake locks από την υποδομή των Android εφαρμογών και τις βιβλιοθήκες του Linux. Σε περίπτωση που δεν υπάρχουν ενεργές wake locks, το σύστημα σταματά τον επεξεργαστή.

Η διαχείριση των κλήσεων για πόρους γίνεται μέσω του Power Manager. Σε περίπτωση που κάποια εφαρμογή ή υπηρεσία παρακάμψει τον Power Manager και λάβει από μόνη της πόρους θα οδηγήσει το σύστημα σε αποσταθεροποίηση. Όλες οι κλήσεις προς τον Power Manager πρέπει να περνούν από τα πρωτόκολλα του Power Manager.

8.7 Κόστος

Οι περισσότεροι νέοι σήμερα θέλουν να είναι εξοπλισμένοι με κινητά τηλέφωνα τελευταίας τεχνολογίας. Πλέον η συσκευή τηλεφώνου που έχει ο καθένας στην κατοχή του αποδεικνύει κατά κάποιο τρόπο την οικονομική του κατάσταση αλλά και την εξοικείωσή του με την τεχνολογία.

Δυστυχώς, στις περισσότερες περιπτώσεις το κόστος αγορά τέτοιων συσκευών είναι απαγορευτικό. Όσον αφορά τις συσκευές iPhone, η τιμή αγοράς ενός iPhone 4 κυμαίνεται γύρω στα 650 ευρώ. Η τιμή αυτή ανεβαίνει όταν η χωρητικότητα της συσκευής είναι περισσότερη. Βέβαια κυκλοφορεί ακόμα στην αγορά και η προηγούμενη γενιά iPhone που αντιπροσωπεύεται από τη συσκευή iPhone 3GS. Η τιμή αυτών είναι γύρω στα 500 ευρώ, τιμή που παραμένει υψηλή.

Όσον αφορά το Android λόγω του ότι προσφέρεται σε συσκευές διάφορων κατασκευαστών υπάρχει ανταγωνισμός με αποτέλεσμα οι τιμές να ξεκινούν από πολύ χαμηλά ποσά και να φτάνουν μέχρι πολύ υψηλά. Το φθηνότερο Android τηλέφωνο της αγοράς έχει τιμή 116 ευρώ. Πρόκειται για συσκευή που δεν ανήκει στους δημοφιλείς κατασκευαστές κινητών τηλεφώνων και είναι branded από την

εταιρεία κινητής τηλεφωνίας Vodafone. Τα ακριβότερα Android τηλέφωνα αυτή τη στιγμή είναι το HTC Desire HD και το Galaxy S της Samsung με τιμή που ανέρχεται στα 600 ευρώ. Είναι εύκολα αντιληπτό ότι μια Android συσκευή μπορεί να φτάσει στα χέρια κάποιου ευκολότερα. Το θέμα όμως τίθεται στο τι θα ικανοποιήσει τον καθένα και αυτό δεν είναι μόνο θέμα τιμής αλλά και εμφάνισης, μεγέθους, λειτουργιών και ευχρηστίας.

8.8 Συγκριτικός πίνακας συσκευών αντιπροσωπευτικών των δύο λειτουργικών συστημάτων

Στον πίνακα συνοψίζονται οι ομοιότητες και διαφορές σε θέματα hardware αλλά και χαρακτηριστικών που συναντώνται στις συσκευές iPhone και Android. Όπου υπάρχει εξειδίκευση σε συγκεκριμένο μοντέλο αυτό θα αφορά για το μεν iOS το κινητό τηλέφωνο iPhone 4G και για το Android το κινητό τηλέφωνο Google Nexus S.

	Android OS	**iOS**
Ομαδοποίηση εφαρμογών	Επιτρέπεται	Επιτρέπεται
Σύστημα ειδοποιήσεων για συμβάντα	Μέσω των εικονιδίων κάθε εφαρμογής	Μέσω μηνυμάτων στην μπάρα ειδοποιήσεων
Προσωποποίηση	Επιτρέπεται μέσω ενεργειών του χρήστη	Επιτρέπεται
Επεξεργαστής	1 GHz ARM Cortex-A8 processor	1GHz Cortex A8 (Hummingbird)
Μνήμη RAM	512MB	512MB
Δευτερεύον αποθηκευτικός χώρος	Μέχρι 32GB	16GB
Αισθητήρες	Accelerometer, GPS, proximity, ambient light, compass	Accelerometer, GPS, proximity, ambient light, compass
Συνδεσιμότητα	Wifi και Bluetooth	WiFi και Bluetooth
Κάμερα	5 megapixel	5 megapixel

SDK	iOS SDK (απαιτείται αγορά)	Android SDK (δωρεάν)
Ενημέρωση λογισμικού	Μέσω iTunes	Μέσω ειδοποίησης στο τηλέφωνο, δεν απαιτείται πάντα σύνδεση με υπολογιστή
Κόστος	Υψηλό (περίπου 500 ευρώ)	Κυμαίνεται από χαμηλό μέχρι και υψηλό, ανάλογα με τη συσκευή (από 115 ευρώ μέχρι 600)

Πίνακας 8. 1 Συγκριτικός πίνακας των χαρακτηριστικών που εξετάστηκαν στο κεφάλαιο 8

Κεφάλαιο 9

Διανομή εφαρμογών μέσω Apple App Store και Android Market

9.1 Apple's App Store

Το Apple App Store βρίσκεται προ-εγκαταστημένο στο iOS και επιτρέπει στους χρήστες να αναζητούν και να εγκαθιστούν εφαρμογές στο κινητό τους τηλέφωνο μέσω του iTunes Store. Το iTunes Store αναπτύχθηκε μέσω του iOS SDK και διανέμεται επίσης από την Apple. Ανάλογα με την εφαρμογή υπάρχει η επιλογή εγκατάστασης είτε δωρεάν είτε με κάποιο μικρό κόστος που τίθεται από το δημιουργό της [44]. Οι εφαρμογές μπορούν να εγκατασταθούν απευθείας σε μια συσκευή ή να «κατεβούν» πρώτα σε κάποιο PC ή Mac μέσω iTunes. Το 30% του τζίρου μιας εφαρμογής πηγαίνει στην Apple και το υπόλοιπο 70% πηγαίνει στο δημιουργό της εφαρμογής.

Το App Store άνοιξε για πρώτη φορά στις 10 Ιουλίου του 2008 μέσω μιας ανανέωσης που έγινε στο iTunes. Στις 11 Ιουλίου κυκλοφόρησε το iPhone 3G με προ-εγκατεστημένο το iOS 2.0.1 με υποστήριξη για το App Store. Από τις 20 Οκτωβρίου του 2010 είναι επίσημα διαθέσιμες στο App Store 300 000 εφαρμογές . Στις 22 Ιανουαρίου του 2011 η εφαρμογή με τον αριθμό δέκα δισεκατομμύρια «κατεβάστηκε» από το App Store. Ο μέσος τζίρος κατά εφαρμογή υπολογίζεται ότι είναι $8,700 δολάρια παρόλο που τα δεδομένα αυτά δεν είναι δημόσια διαθέσιμα και επίσημα.

Η μέση τιμή μιας μη δωρεάν εφαρμογής στο App Store κυμαίνεται από $3,5 δολάρια μέχρι και $4 δολάρια. Η κατανομή των τιμών ακολουθεί την power-law κατανομή από το νόμο του Zipf. Παρόλο που οι τιμές επιλέγονται αυθαίρετα από τον καθένα, οι περισσότεροι δημιουργοί επιλέγουν την τιμή της εφαρμογής τους να είναι το πολύ $4.99 δολάρια.

Μετά την επιτυχία του App Store και τη δημιουργία παρόμοιων υπηρεσιών από τους ανταγωνιστές της Apple (π.χ. NOKIA) ο όρος App Store έχει υιοθετηθεί και αναφέρεται για όλων των ειδών τις υπηρεσίες παρόμοιου τύπου άλλων εταιρειών. Βέβαια, η Apple έχει κάνει ήδη αίτηση από το 2008 για να δεσμεύσει τον όρο App Store ως Trademark (αποκλειστικά δική της χρήση και δικαιώματα). Η επιτυχία αυτή ήρθε και η εξασφάλιση του trademark ήρθε μόλις στις αρχές του 2011.

Λόγω του App Store και της απήχησης που είχε στο κοινό, η λέξη App πήρε το βραβείο της πιο συχνά αναφερόμενης λέξης για το 2010 (Word of the Year) από την Αμερικανική Κοινωνία Διαλέκτου. Η Apple παρόλα αυτά δεν έχει κάνει αίτηση και προφανώς ούτε προτίθεται για να πάρει τα δικαιώματα και αυτής της λέξης.

Στις 20 Οκτωβρίου 2010 η Apple ανακοίνωσε το Mac App Store που είναι παρόμοιο με αυτό των iOS συσκευών αλλά περιέχει εφαρμογές για τους Mac υπολογιστές. Ενώ το Mac App Store είναι προσβάσιμο μόνο από τον Mac OS X Snow Leopard υπολογιστή, το App Store είναι προσβάσιμο από τα iPhone, iPod και iPad μέσω μιας iOS εφαρμογής με το ίδιο όνομα. Είναι επίσης ο μόνος τρόπος για την εγκατάσταση εφαρμογών από τρίτους σε αυτές τις συσκευές χωρίς τη χρήση jailbreaking στη συσκευή.

Εφαρμογές διαδικτύου μπορούν επίσης να εγκατασταθούν σε αυτές τις συσκευές χωρίς να ληφθεί καθόλου υπόψη το App Store. Παρουσιάζουν όμως πολύ περιορισμένη λειτουργικότητα. Το App Store είναι επίσης προσβάσιμο και μέσω του iTunes και έτσι και σε κάθε λειτουργικό σύστημα στο οποίο είναι δυνατό να εγκατασταθεί το iTunes.

Το Φεβρουάριο του 2011 η Apple ανακοίνωσε τη νέα υπηρεσία της βάση συνδρομής η οποία θα επιτρέπει στους εκδότες να καθορίζουν το χρονικό διάστημα αλλά και την τιμή της συνδρομής τους. Η νέα αυτή υπηρεσία θα επιτρέπει στους εκδότες να πουλούν το περιεχόμενό τους μέσω των εφαρμογών τους επιτρέποντας στους χρήστες να λαμβάνουν νέο περιεχόμενο για συγκεκριμένο χρονικό διάστημα. Επίσης, η Apple θα επιτρέπει στους εκδότες όχι μόνο να πουλούν μέσω του iTunes με τα ποσοστά κέρδους που προαναφέρθηκαν αλλά θα τους επιτρέπει να διανέμουν το περιεχόμενό τους και μέσω των προσωπικών τους ιστοσελίδων χωρίς κανένα μερίδιο για την Apple.

Τον Απρίλιο του 2009 η Apple ανακοίνωσε τις εφαρμογές που εγκαταστάθηκαν τις περισσότερες φορές από τότε που δημιουργήθηκε το App Store. Από τις μη δωρεάν εφαρμογές η πιο δημοφιλής ήταν η Crash Bandicoot Nitro Kart 3D ενώ από τις δωρεάν ήταν οι Facebook(στην πρώτη θέση) και Google Earth (στη δεύτερη θέση).

Η Apple κατηγοριοποιεί τις εφαρμογές της με βάση το περιεχόμενό τους και καθορίζει με τον τρόπο αυτό το ηλικιακό γκρουπ για το οποίο είναι πιο κατάλληλες. Σύμφωνα με το iPhone OS 3.0 το iPhone θα επιτρέπει το μπλοκάρισμα ακατάλληλων εφαρμογών στις ρυθμίσεις του. Η Apple κατηγοριοποιεί τις εφαρμογές σε 4 γκρουπ: Αρχικά είναι το γκρουπ 4+ στο οποίο οι εφαρμογές δεν περιλαμβάνουν καθόλου κακό περιεχόμενο. Ακολουθεί η κατηγορία 9+ όπου είναι δυνατόν οι εφαρμογές να περιλαμβάνουν εμφανίσεις χαρακτήρων κινουμένων σχεδίων, φανταστικής ή πραγματικής βίας σε λογικά επίπεδα ή ακόμα και τρομακτικό περιεχόμενο το οποίο πιθανώς να μην είναι κατάλληλο για παιδιά κάτω των εννέα ετών. Στην κατηγορία 12+ περιλαμβάνονται οι εφαρμογές που είναι δυνατόν να περιέχουν άσχημη γλώσσα, έντονα γραφικά στοιχεία, φανταστική ή πραγματική βία σε πιο έντονο βαθμό αλλά και προσομοιωμένα παιχνίδια τζόγου. Τέλος είναι η κατηγορία των 17+ όπου περιλαμβάνονται εφαρμογές με όλα τα προαναφερόμενα χαρακτηριστικά καθώς και περιεχόμενο που έχει να κάνει με ακατάλληλες σκηνές, αλκοόλ, τσιγάρα και

ναρκωτικά. Πριν από την εγκατάσταση των εφαρμογών αυτής της κατηγορίας εμφανίζεται ένα μήνυμα στο χρήστη που τον ενημερώνει για το περιεχόμενο της εφαρμογής αλλά και την ηλικία στην οποία κατατάσσεται.

Οι εφαρμογές όπως αναφέρθηκε και στο κεφάλαιο 7 υποβάλλονται στην Apple για έλεγχο αξιοπιστίας και περαιτέρω ανάλυση. Οι εφαρμογές μπορούν να διανέμονται ad-hoc αν απορριφθούν αν ο δημιουργός τους υποβάλει αίτηση για να πάρει άδεια από την Apple έτσι ώστε η εφαρμογή να εγκατασταθεί σε συγκεκριμένα iPhone. Βέβαια, η Apple διατηρεί το δικαίωμα να πάρει πίσω ανά πάσα στιγμή αυτή την άδεια αν εντοπίσει κάτι ύποπτο σχετικά με την εφαρμογή.

Υπάρχουν ακόμα και εφαρμογές οι οποίες δεν είναι διαθέσιμες για εγκατάσταση εκτός του App Store των ΗΠΑ αν το επιθυμεί ο δημιουργός. Επίσης, η Apple αφαίρεσε την άδεια λογισμικού GPL από το App Store μετά από παράπονα από δημιουργούς εφαρμογών ότι οι όροι του App Store είναι ασυνεπείς με αυτούς της GPL.

9.2 Android Market

Το Android Market είναι ένα online κατάστημα λογισμικού το οποίο αναπτύχθηκε από την Google για τις Android συσκευές. Ένα πρόγραμμα εφαρμογής (app) που ονομάζεται Market είναι προ-εγκατεστημένο στις περισσότερες Android συσκευές και επιτρέπει στους χρήστες να αναζητούν και να κατεβάζουν εφαρμογές που προσφέρονται από δημιουργούς εκτός της ομάδας ανάπτυξης του Google και που φιλοξενούνται στο Android Market [45]. Οι χρήστες μπορούν επίσης να αναζητήσουν και να διαβάσουν λεπτομερή περιγραφή σχετικά με τις εφαρμογές που τους ενδιαφέρουν από την ιστοσελίδα του Android Market.

Το Android Market ανακοινώθηκε για πρώτη φορά στις 28 Αυγούστου του 2008 και έγινε διαθέσιμο στους χρήστες στις 22 Οκτωβρίου της ίδιας χρονιάς. Υποστήριξη σχετικά με τις εφαρμογές έναντι καταβολής κάποιου ποσού προστέθηκε για τις ΗΠΑ

και την Αγγλία στα μέσα Φεβρουαρίου 2009. Οι χρήστες από την Αγγλία πήραν το πράσινο φως για την αγορά μη δωρεάν εφαρμογών από τις 13 Μαρτίου 2009.

Σύμφωνα με τον τεχνικό διευθυντή της T-Mobile, στις 17 Μαρτίου 2009 ήταν διαθέσιμες για εγκατάσταση 2300 εφαρμογές στο Android Market. Μέχρι το Δεκέμβριο του 2009 πάνω από 20 000 εφαρμογές τοποθετήθηκαν στο Android Market. Οι εφαρμογές πολλαπλασιάστηκαν σε πολύ μικρό χρονικό διάστημα και έτσι μέχρι τον Αύγουστο του 2010 πάνω από 80 000 εφαρμογές ήταν διαθέσιμες ενώ πάνω από ένα δισεκατομμύριο είχαν ήδη εγκατασταθεί σε Android συσκευές. Ακολούθησε ρυθμός αύξησης περίπου 10 000 νέων εφαρμογών κάθε μήνα.

Τον Ιούλιο του 2010 κυκλοφόρησε ένα έγγραφο από την εταιρεία Distimo το οποίο πληροφορούσε το κοινό ότι το Android Market είχε το μεγαλύτερο ποσοστό δωρεάν εφαρμογών που ήταν 57%, διπλάσιο από το αντίστοιχο της Apple σχετικά με το App Store όπου μόλις το 28% ήταν δωρεάν εφαρμογές.

Το Δεκέμβριο του 2010 ανακοινώθηκε ότι το Android Market θα διαδεχόταν μια ενημέρωση στην οποία θα περιλαμβάνονται και μικρές αλλαγές μαζί με την πρόσθεση φιλτραρίσματος περιεχομένου. Η νέα ενημέρωση ήταν διαθέσιμη για όλες τις συσκευές Android που έτρεχαν λειτουργικό 1.6 ή πιο πρόσφατο.

Στις 31 Δεκεμβρίου του 2010 το Android Market έφτασε τις 200 000 εφαρμογές. Μόλις λίγο καιρό αργότερα, στις 2 Φεβρουαρίου του 2011, η Google παρουσίασε ένα νέο web client που παρείχε πρόσβαση στο Android Market και μέσω υπολογιστή. Οι εφαρμογές που επιθυμούσε πλέον ο χρήστης θα μπορούσαν να κατεβούν και να εγκατασταθούν απευθείας στην καταχωρημένη Android συσκευή.

Όσον αφορά το κέρδος από τη διανομή Android εφαρμογών στο Android Market, όπως και στην Apple, το 70% των κερδών πηγαίνει στο δημιουργό. Το υπόλοιπο 30% κατανέμεται μεταξύ των φορέων που παρέχουν την εφαρμογή. Το κέρδος που προκύπτει από το Android Market δίνεται στους δημιουργούς των εφαρμογών μέσω εμπορικών λογαριασμών Google Checkout. Η T-Mobile, η πρώτη συσκευή Android που κυκλοφόρησε, πρόσφατα ανανέωσε την αγορά της έτσι ώστε να επιτρέπει στην

Google να χρεώνει τις εφαρμογές απευθείας στον τηλεφωνικό λογαριασμό του χρήστη της συσκευής.

Όσον αφορά το Android Market, υπάρχουν χωρικοί περιορισμοί σύμφωνα με τους οποίους άτομα συγκεκριμένων χωρών δεν μπορούν να κατεβάζουν/αγοράζουν ή να πουλούν Android εφαρμογές. Όσον αφορά την Κύπρο, οι κάτοχοι των Android συσκευών μπορούν να κατεβάζουν τις δωρεάν εφαρμογές του Android Market, δεν μπορούν όμως να αγοράζουν εφαρμογές αλλά ούτε και να διαθέτουν τις δικές τους μέσω του Android Market με κάποια χρέωση. Προς το παρόν μόνο δημιουργοί εφαρμογών που βρίσκονται στις Αυστρία, Γαλλία, Γερμανία, Ολλανδία, Ισπανία, Αγγλία και ΗΠΑ έχουν τη δυνατότητα να πουλούν τις εφαρμογές τους στο Android Market.

Σε αντίθεση με το iPhone, δεν υπάρχει καμιά απαίτηση ότι οι Android εφαρμογές πρέπει να προμηθεύονται στους χρήστες μόνο από το Android Market. Οι χρήστες μπορούν να προμηθευτούν Android εφαρμογές από οποιαδήποτε πηγή συμπεριλαμβανομένων της ιστοσελίδας των δημιουργών τους αλλά και από οποιεσδήποτε εναλλακτικές αγορές Android εφαρμογών.

Αρκετές φορές μέχρι σήμερα χρειάστηκε η Google να αφαιρέσει εφαρμογές από το Android Market λόγω παραβίασης κανονισμών αλλά και για αναξιοπιστία. Οι εφαρμογές που απομακρύνθηκαν όμως είναι ακόμα διαθέσιμες από τις προσωπικές ιστοσελίδες των δημιουργών τους οπότε εναπόκειται στον κάθε χρήστη το αν θα τις εγκαταστήσει ή όχι.

Οι εφαρμογές του Android OS περιλαμβάνονται στα apk αρχεία. Το Android Market δεν εγκαθιστά τις ίδιες τις εφαρμογές. Αντίθετα, ζητά από την υπηρεσία PackageManagerService της συσκευής να τις εγκαταστήσει. Ο package manager μπορεί να δει απευθείας ότι ο χρήστης προσπαθεί να εγκαταστήσει ένα .apk αρχείο στη συσκευή του. Αρχικά οι εφαρμογές εγκαθίστανται στον εσωτερικό αποθηκευτικό χώρο της συσκευής ενώ στη συνέχεια είναι δυνατή και η αποθήκευσή τους στην κάρτα μνήμης υπό συγκεκριμένες συνθήκες.

Οι Android συσκευές μπορούν να εκτελέσουν εφαρμογές που έχουν δημιουργηθεί από τρίτους και διανέμονται μέσω του Android Market ή άλλων αγορών που υπάρχουν. Όταν οι δημιουργοί εφαρμογών πάρουν το προσωπικό τους κλειδί μπορούν να διαθέσουν άμεσα τις εφαρμογές τους χωρίς ιδιαίτερη διαδικασία έγκρισης.

Οι εταιρείες λογισμικού ασφαλείας έχουν αναπτύξει εφαρμογές για να διασφαλίζουν την ασφάλεια των Android συσκευών. Η SMobile Systems υποστηρίζει ότι 20% των εφαρμογών στο Android Market ζητούν δικαιώματα τα οποία θα μπορούσαν να χρησιμοποιηθούν για κακόβουλες ενέργειες ενώ 5% των εφαρμογών παίρνουν το δικαίωμα να πραγματοποιούν τηλεφωνικές κλήσεις χωρίς καμιά παρέμβαση του χρήστη σε αυτό. Αυτό καθαυτό δε σημαίνει ότι οι εφαρμογές αυτές είναι κακόβουλες όμως η πιθανότητα για παράνομη δραστηριότητα υπάρχει.

Στις αρχές Μαρτίου του 2011, το DroidDream, ένα Trojan rootkit κυκλοφόρησε στο Android Market με τη μορφή αρκετών δωρεάν εφαρμογών που ήταν στην πραγματικότητα πειρατικές εκδόσεις υπαρχόντων, μη δωρεάν εφαρμογών. Αυτό επέτρεψε στους hackers να κλέψουν πληροφορίες όπως αριθμούς IMEI και IMSI, μοντέλα τηλεφώνων, ταυτότητες εφαρμογών και service providers. Οι εφαρμογές εγκαθιστούσαν και μια «πίσω πόρτα» η οποία επέτρεπε στους hackers να κατεβάζουν περισσότερο κώδικα στη διασύνδεση της συσκευής. Αυτές οι εφαρμογές εγκαταστάθηκαν περισσότερο από 50 000 φορές πριν η Google ενεργήσει και τις απομακρύνει από το Android Market. Οι εφαρμογές αυτέ επηρέασαν συσκευές που έτρεχαν εκδόσεις λογισμικού μέχρι και Android 2.3. Σε πολλές περιπτώσεις η μόνη εγγυημένη μέθοδος αφαίρεσης της εφαρμογής και των περιεχομένων της από τις συσκευές ήταν η επαναφορά των εργοστασιακών ρυθμίσεων.

Παρόλα αυτά, η Google στις 5 Μαρτίου άρχισε την απομακρυσμένη αφαίρεση της κακόβουλης εφαρμογής από τις επηρεαζόμενες συσκευές και κυκλοφόρησε ταυτόχρονα την εφαρμογή Android Market Security Tool τον Μάρτιο του 2011 η οποία απομάκρυνε αυτόματα αυτή την ομάδα των εφαρμογών. Η εφαρμογή αυτή της

Google εγκαταστάθηκε αυτόματα σε όλες τις επηρεαζόμενες συσκευές και οι χρήστες των συσκευών αυτών ειδοποιήθηκαν μέσω email.

Η τάση που φαίνεται στην ανάπτυξη του Android Market είναι συνεχώς αυξητική ενώ όπως θα παρουσιαστεί και στην επόμενη ενότητα αναμένεται σε λίγο καιρό να ξεπεράσει τα ποσοστά του Apple App Store. Το μόνο σίγουρο είναι ότι οι χρήστες πρέπει να ελέγχουν τις εφαρμογές που επιλέγουν να εγκαταστήσουν στη συσκευή τους μιας και η Google προς το παρόν δεν παίρνει εκ των προτέρων μέτρα για την αποτροπή της κυκλοφορίας τους.

9.3 Συγκριτικός πίνακας των αγορών εφαρμογών

	Apple App Store	**Android Market**
Τρόπος εγκατάστασης στη συσκευή	Προ-εγκατεστημένο	Προ-εγκατεστημένο
Διαθέσιμες εφαρμογές	300 000+	200 000+
Κόστος	Δωρεάν/Μέσω Χρέωσης	Δωρεάν/Μέσω Χρέωσης
Εναλλακτικός τρόπος πρόσβασης	Μέσω iTunes (Υπολογιστή)	Μέσω ιστοσελίδας AndroidMarket
Έλεγχος εφαρμογών	Εξέταση και χορήγηση άδειας από την Apple	Δεν υπάρχει
Κατηγοριοποίηση εφαρμογών	Ναι	Ναι
Διάθεση Ad-Hoc	Μετά από άδεια της Apple και μόνο για συγκεκριμένα iPhone	Ελεύθερη
Χωρικοί περιορισμοί διάθεσης και αγοράς εφαρμογών	Ναι	Ναι

Πίνακας 9. 1 Σύγκριση των αγορών εφαρμογών των δύο λειτουργικών συστημάτων

Κεφάλαιο 10

Ποιος είναι ο κυρίαρχος της αγοράς;

10.1 Αποδοχή από την αγορά

Το κινητό τηλέφωνο αλλά και το διαδίκτυο έχουν μετατρέψει ήδη την κοινωνική ζωή των ανθρώπων και έχουν φέρει πολλές αλλαγές στην αγορά εργασίας. Από το πρώτο κιόλας κεφάλαιο τονίστηκε η σημασία των έξυπνων τηλεφώνων που σπάνε τα όρια του χώρου και δίνουν τη δυνατότητα στους χρήστες να έχουν παντού μαζί τους το κινητό τους τηλέφωνο. Η εξοικείωση με τις συσκευές των κινητών τηλεφώνων αλλά και η προσμονή εξερεύνησης όλων των χαρακτηριστικών τους δημιούργησε μια νέα κουλτούρα πολιτισμού που οδήγησε στην επανάσταση της κινητής τηλεφωνίας και των συσκευών της.

Τίποτα από τα προαναφερόμενα δεν θα είχε επιτευχθεί χωρίς την απαραίτητη τεχνολογική ανάπτυξη που έφερε το κοινό σε επαφή με την πραγματικότητα του 3G, 4G και της συνεχής πρόσβασης στο δίκτυο (always on). Μια τέτοια συσκευή πρέπει να έχει και την ανάλογη επεξεργαστική ικανότητα, την ανάλογη μνήμη αλλά και όλες τις απαραίτητες διεπαφές για την πραγματοποίηση κλήσεων και την εκτέλεση εφαρμογών.

Η άφιξη του iPhone και ειδικότερα της δεύτερης έκδοσής του (iPhone 3G), έδωσε το έναυσμα για την αρχή της νέας εποχής των κινητών συσκευών παρέχοντας όλα τα χαρακτηριστικά που εξετάστηκαν στα προηγούμενα κεφάλαια [46]. Υπήρξαν βέβαια και άλλες συσκευές παράλληλα που είχαν επίσης στόχο την παγκόσμια κυριαρχία μεταξύ των οποίων οι: RIM BlackBerry Storm, T-Mobile HTC Google και Nokia N96 16GB smartphone. Όμως το iPhone μαζί με την απόφαση της Apple να θέσει ως

στόχο ολόκληρη την αγορά κυριάρχησε από την πρώτη κιόλας μέρα κυκλοφορίας του.

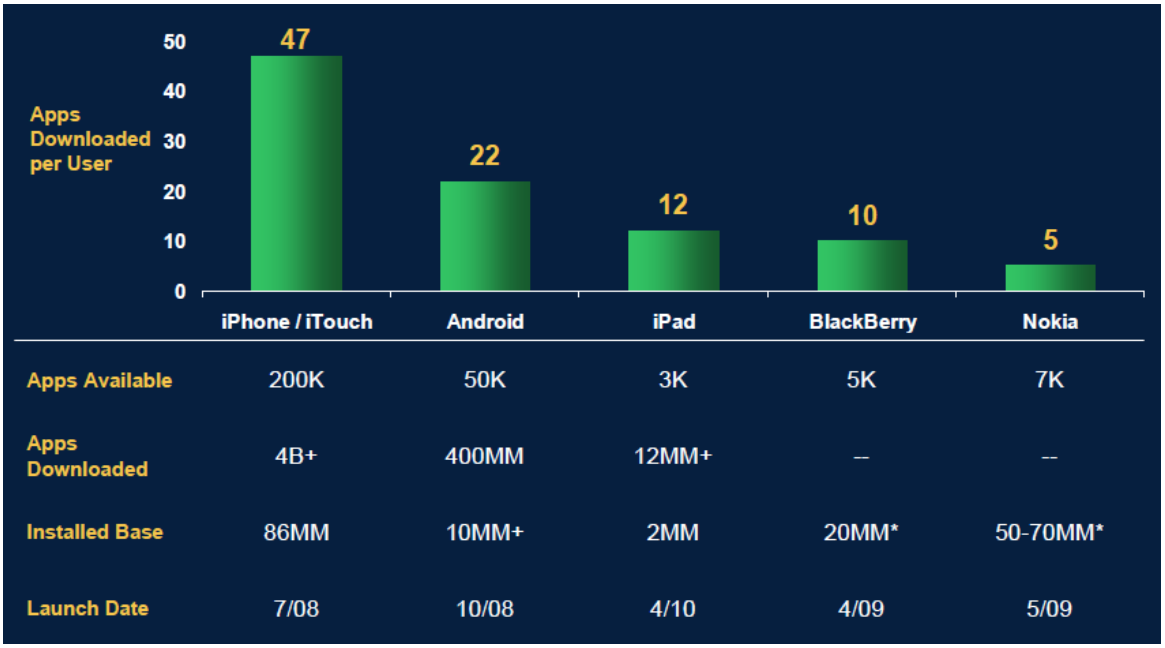
Από την ημέρα ανακοίνωσης της έλευσης του iPhone χιλιάδες ήταν εκείνοι που περίμεναν για να το αγοράσουν. Η Apple είχε πετύχει λοιπόν την τόσο σημαντική αναμονή κοινού. Μετά από αυτό όλα τα υπόλοιπα κύλισαν απίστευτα καλά για την Apple αφού παρόλο των υψηλών τιμών με τις οποίες κυκλοφορεί το iPhone στην αγορά, πάντοτε το κοινό σπεύδει να τις αποκτήσει. Ακόμα περισσότερο χρήμα μπήκε στα ταμεία μέσω του App Store ενώ το υψηλό κέρδος συνεχίζεται και σήμερα, παράλληλα με την επιτυχία του Google's Android.

Τα τηλέφωνα με λογισμικό Android που παρέχεται από την Google είναι ίσως η πρώτη μεγάλη απειλή για την αγορά του iPhone [47]. Αυτό συνέβηκε κυρίως γιατί η Google έβαλε ως στόχο της ακριβώς το ίδιο κοινό με αυτό της Apple για τα iPhone. Και σα να μην ήταν αυτό αρκετό, η Google έθεσε και ως στόχο να κερδίσει και τις καρδιές των δημιουργών εφαρμογών για κινητά τηλέφωνα. Με βάση λοιπόν το μερίδιο αγοράς και τον αριθμό των διαθέσιμων εφαρμογών, το Android θεωρείται επιτυχημένο. Ένας ακόμα λόγος που συντέλεσε στην επιτυχία και αποδοχή του ήταν ότι το κοινό μπορούσε να έχει ένα έξυπνο τηλέφωνο στα χέρια του με ένα πολύ καλό λειτουργικό σύστημα χωρίς να ξοδέψει πολλές εκατοντάδες ευρώ [48]. Πολλοί είναι επίσης αυτοί που υποστηρίζουν ότι η επιτυχία του Android δεν μπορεί ευθέως να συγκριθεί με αυτή του iOS για τα iPhone. Η άποψη αυτή στηρίζεται στο γεγονός ότι η Google, από τη στιγμή που δεν παράγει η ίδια τις συσκευές δεν την ενδιαφέρει καθόλου το πόσες συσκευές θα πωληθούν. Το μόνο που την ενδιαφέρει είναι οι συσκευές να τρέχουν λογισμικό Android [49]. Προκύπτει λοιπόν το συμπέρασμα ότι το Android έτυχε καλής υποδοχής από την αγορά, όπως και το iPhone.

10.2 Στατιστικά στοιχεία

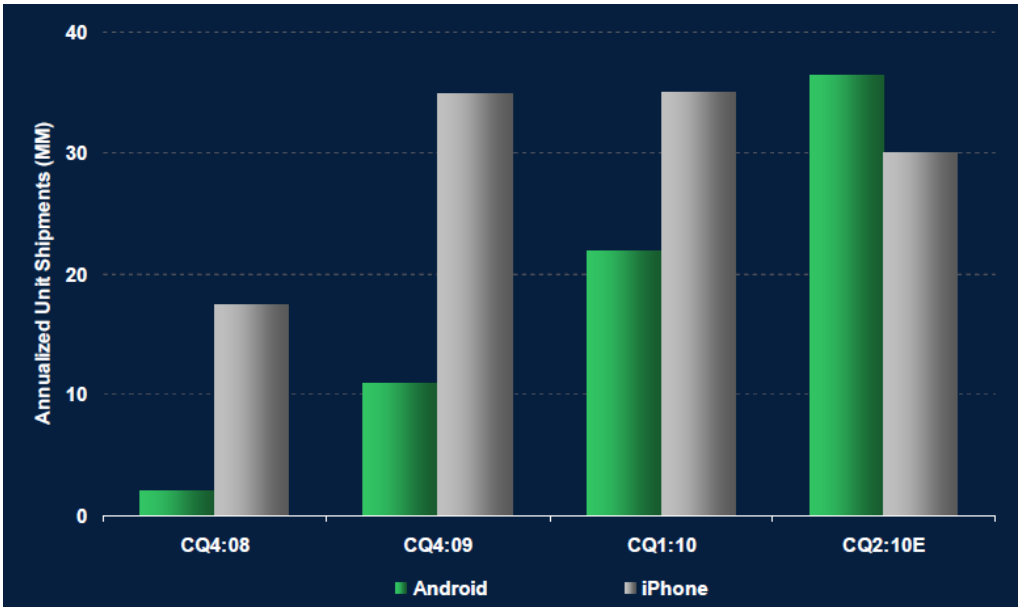
Στα γραφήματα που ακολουθούν αναφέρονται στοιχεία πωλήσεων για τις συσκευές που τρέχουν το iOS της Apple και το Android της Google κυρίως.

Το γράφημα 10.1 παρουσιάζει κατά μέσο όρο των αριθμό των εφαρμογών που κατεβάζει ο χρήστης της συσκευής στο κινητό του τηλέφωνο. Πρόκειται για στοιχεία που προέκυψαν τους μήνες Μάιο και Ιούνιο του 2010 [50]. Φαίνεται ξεκάθαρα πως τη συγκεκριμένη χρονική στιγμή οι χρήστες των iPhone είχαν ακόμη στη διάθεσή τους περισσότερες εφαρμογές στο App Store με αποτέλεσμα να παρουσιάζεται και μεγαλύτερη κινητικότητα.



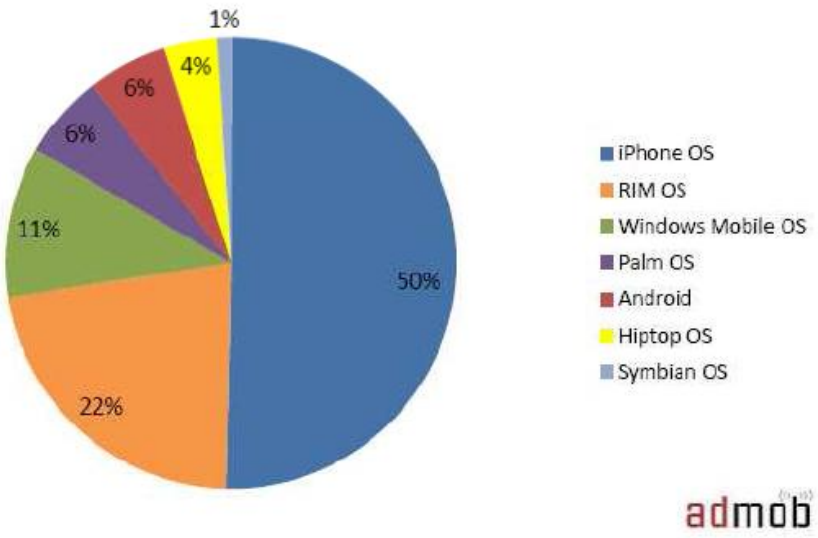
Γράφημα 10. 1 Η αναλογία εφαρμογών που οι χρήστες εγκαθιστούν στα κινητά τους τηλέφωνα μέσω των αγορών εφαρμογών.

Το γράφημα 10.2 παρουσιάζει τις πωλήσεις των κινητών τηλεφώνων που τρέχουν iOS και των κινητών τηλεφώνων που τρέχουν Android λογισμικό [50]. Πρόκειται για δεδομένα πωλήσεων του 2008, 2009 και του πρώτου εξαμήνου του 2010 με πρόβλεψη για τις πωλήσεις του δεύτερου εξαμήνου 2010 όπου και οι Android συσκευές ξεπερνούν σε πωλήσεις τα iPhone. Όπως παρουσιάζεται και στα γραφήματα που ακολουθούν πρόκειται για πρόβλεψη που επαληθεύτηκε.



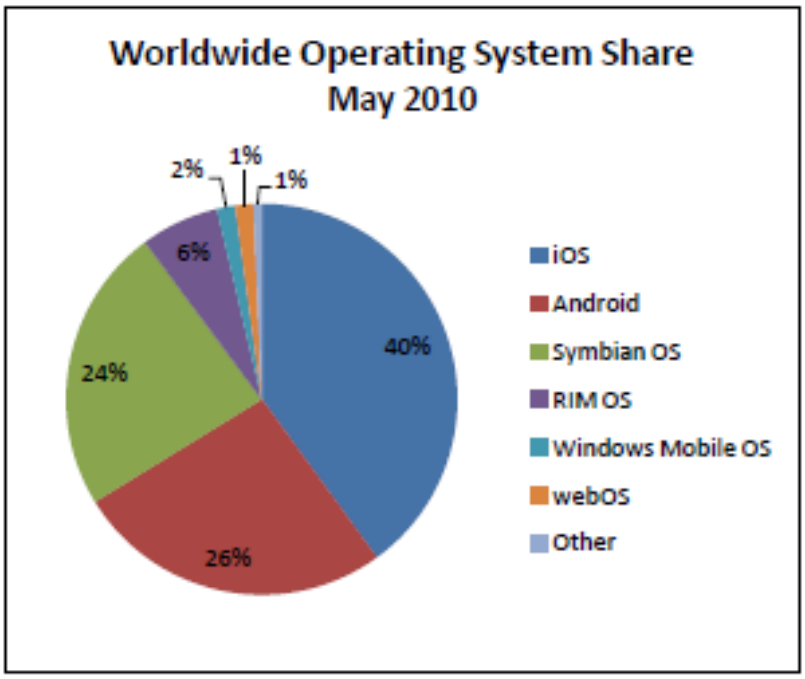
Γράφημα 10. 2 Οι ετήσιες πωλήσεις συσκευών με iOS και Android λογισμικό. Σύμφωνα με το γράφημα οι Android συσκευές ήταν αναμενόμενο να ξεπεράσουν για πρώτη φορά τις iOS συσκευές το δεύτερο εξάμηνο του 2010

Το γράφημα 10.3 παρουσιάζει τις διαφημίσεις που προωθούνταν στις αγορές εφαρμογών διαφόρων εταιρειών το Μάρτιο του 2009 στις ΗΠΑ[51]. Το μικρό μερίδιο αγορά που καταλαμβάνει το Android οφείλεται στο γεγονός ότι μέχρι τότε το Android Market δεν ήταν ευρέως διαδεδομένο όπως αναφέρθηκε και στο κεφάλαιο 8.



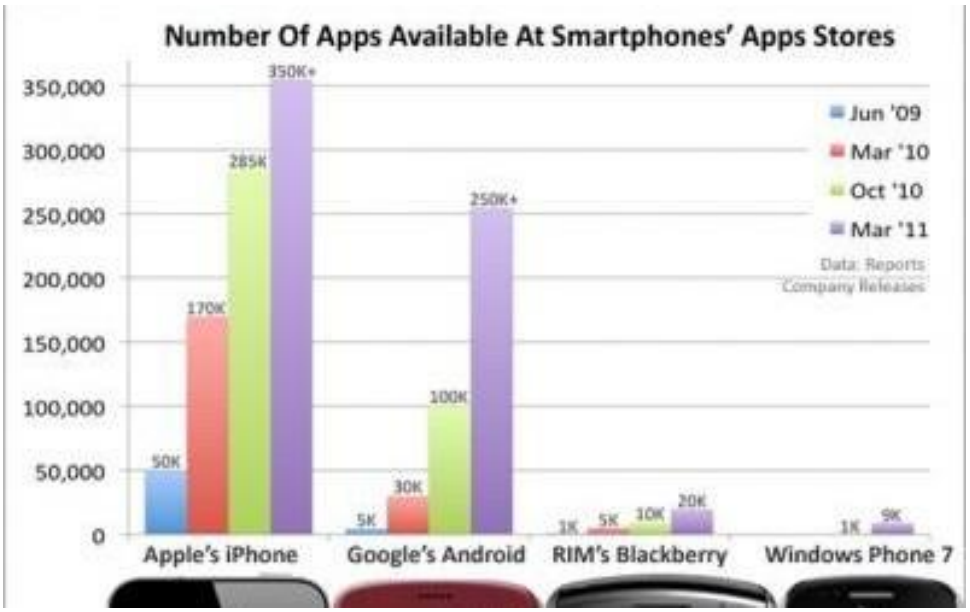
Γράφημα 10. 3 Το μερίδιο αγοράς των κυριότερων λειτουργικών συστημάτων όσον αφορά την προώθηση διαφημίσεων

Το γράφημα 10.4 παρουσιάζει το μερίδιο αγοράς των πιο γνωστών λειτουργικών συστημάτων στην παγκόσμια αγορά για ολόκληρο το 2010 [52]. Φαίνεται ξεκάθαρα ότι iOS και Android κυριαρχούν στην αγορά συγκεντρώνοντας μερίδιο πάνω από 65%.



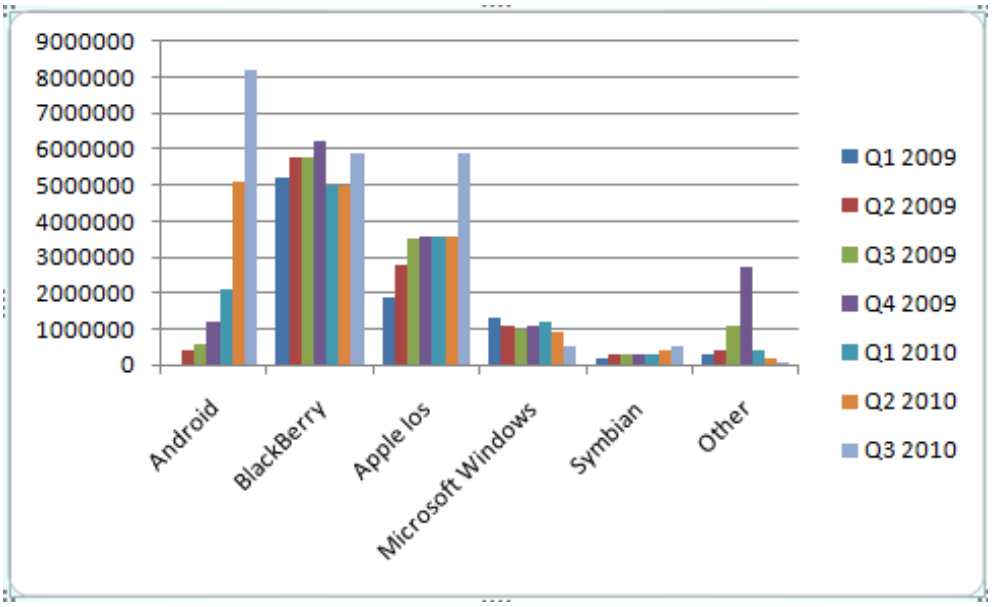
Γράφημα 10. 4 Το μερίδιο αγοράς των πιο γνωστών λειτουργικών συστημάτων για το 2010

Το γράφημα 10.5 παρουσιάζει την ανάπτυξη του Android Market μέχρι και σήμερα όπου φαίνεται να πλησιάζει πλέον τις επιδόσεις του App Store [53]. συστημάτων στην παγκόσμια αγορά για ολόκληρο το 2010 [52].



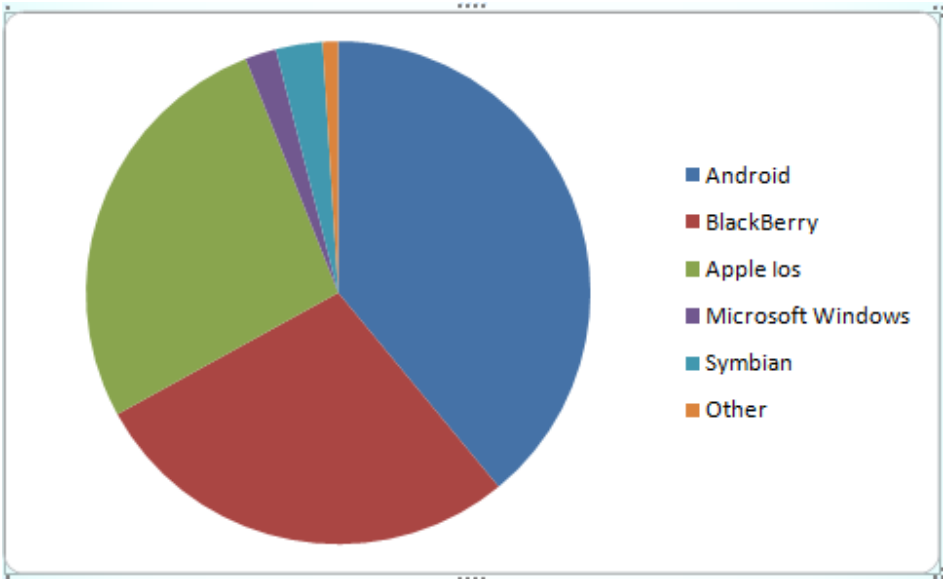
Γράφημα 10. 5 Ο αριθμός των διαθέσιμων εφαρμογών στις πιο γνωστές αγορές από τον Ιούνιο του 2009 μέχρι το Μάρτιο του 2011

Το γράφημα 10.6 παρουσιάζει τις πωλήσεις σε εκατομμύρια των συσκευών κινητών τηλεφώνων με τα λειτουργικά συστήματα Android, BlackBerry, Apple iOS, Microsoft Windows, Symbian κ.α. από το πρώτο τετράμηνο του 2009 μέχρι το τρίτο τετράμηνο του 2010 στη Νότια Αμερική[47]. Οι συσκευές με Android OS είναι οι μόνες που παρουσιάζουν σημαντική αύξηση πωλήσεων για κάθε τετράμηνο τονίζοντας την αυξητική τάση ανάπτυξής του. Συμπέρασμα που προκύπτει από το γράφημα αυτό είναι ότι σε κοινωνίες με αναπτυσσόμενη οικονομία, οι συσκευές Android παρουσιάζονται περισσότερο δημοφιλείς παρά οι συσκευές των υπολοίπων μεγάλων κατασκευαστών.



Γράφημα 10. 6 Οι πωλήσεις smartphones στη Νότια Αμερική κάθε τρίμηνο από το 2009 μέχρι τον Σεπτέμβριο του 2010

Το γράφημα 10.7 παρουσιάζει τα μερίδια αγοράς για την Νότια Αμερική όσον αφορά το κυρίαρχο λειτουργικό σύστημα [47]. Για πρώτη φορά στο τρίτο τετράμηνο του 2010 το Android OS κυριαρχεί στην αγορά με 39% μερίδιο.



Γράφημα 10. 7 Το μερίδιο αγοράς των smartphone στη Νότια Αμερική για το τρίτο τρίμηνο του 2010.

10.3 Συγκριτικός πίνακας χαρακτηριστικών

Στον πίνακα που ακολουθεί παρουσιάζονται συγκριτικά τα κυριότερα χαρακτηριστικά των δύο λειτουργικών συστημάτων που εξετάστηκαν. Λόγω της ραγδαίας ανάπτυξης και των δύο τα στοιχεία αυτά αλλάζουν συνεχώς. Τα στοιχεία του πίνακα είναι τα πιο πρόσφατα που ισχύουν μέχρι τον Απρίλιο του 2011.

	Android OS	**iOS**
Λειτουργικό σύστημα στο οποίο στηρίζεται	Linux	Mac OS X
Περιβάλλον ανάπτυξης εφαρμογών	Eclipse	Xcode
Γλώσσα ανάπτυξης	Java	Objective-C
Υποστήριξη εικονικών μηχανών	Ναι	Όχι
Άδεια διανομής	Ανοικτού Κώδικα	Κλειστού κώδικα
Multitasking	Επιτρέπεται	Δεν επιτρέπεται
Πιο πρόσφατη έκδοση	2.3 (για κινητά τηλέφωνα) και 3.0 (για tablets)	4.3

Πίνακας 10. 1 Συγκριτικός πίνακας των κυριότερων διαφορών των δύο λειτουργικών συστημάτων

10.4 Συγκριτικός πίνακας διαφορών σε χαρακτηριστικά ανάπτυξης του λειτουργικού συστήματος

Στον πίνακα που ακολουθεί παρουσιάζονται συγκριτικά τα κυριότερα χαρακτηριστικά που αφορούν την ανάπτυξη του συστήματος στα δύο λειτουργικά συστήματα.

	Android OS	iOS
Διαχείριση Μνήμης	• Dalvik VM και Garbage Collector • Οι διεργασίες τερματίζονται με βάση την προτεραιότητα της εφαρμογής και οι εφαρμογές μένουν ενεργές μέχρι να μην υπάρχουν πλέον πόροι • Ο χρήστης έχει τη δυνατότητα χρήσης Task Killer εφαρμογής	• Δεν έχει Garbage Collector και η διαγραφή μεταβλητών γίνεται μέσω της κλάσης NSObject • Οι εφαρμογές ειδοποιούνται για ελευθέρωση πόρων αλλιώς τερματίζονται • Ο χρήστης δεν μπορεί να χειριστεί τον τερματισμό διεργασιών
Ανάπτυξη εφαρμογών	• Ορίζεται μια εικονική μηχανή και μια διεργασία για κάθε εφαρμογή • Χρήση Android SDK για ανάπτυξη εφαρμογών • Η κάθε εφαρμογή μπορεί να έχει τη δική της αρχιτεκτονική ανάλογα με τη λειτουργία της • Διαμοίραση λειτουργιών και δεδομένων ανάμεσα σε εφαρμογές	• Χρήση UIKitFramework και AppKitFramework για ανάπτυξη εφαρμογών • Χρήση iPhone SDK για ανάπτυξη εφαρμογών • Κάθε εφαρμογή έχει την ίδια αρχιτεκτονική πυρήνα • Δεν υπάρχει διαμοίραση λειτουργιών και δεδομένων ανάμεσα στις εφαρμογές
Multitasking	• Επιτρέπεται σε όλες τις συσκευές	• Επιτρέπεται από την έκδοση iOS 4 και μετά

Πίνακας 10. 2 Συγκριτικός πίνακας χαρακτηριστικών ανάπτυξης για τα δύο λειτουργικά συστήματα

Κεφάλαιο 11

Ανοικτά ζητήματα και Συμπεράσματα

11.1 Ανοικτά ζητήματα

Παρόλο που τόσο το iOS όσο και το Android OS βρίσκονται σε υψηλά επίπεδα ανάπτυξης, υπάρχουν θέματα τα οποία πρέπει να αντιμετωπιστούν. Κυρίαρχο ζήτημα είναι βεβαίως αυτό της ασφάλειας το οποίο ενδιαφέρει όλους. Σε προηγούμενο κεφάλαιο έγινε αναφορά τόσο στους μηχανισμούς προστασίας στο καθένα όσο και στα ευαίσθητα σημεία τα οποία μπορούν να εκμεταλλευθούν οι hackers για την πραγματοποίηση κακόβουλων επιθέσεων.

Η Google μπορεί να είναι πιο ήσυχη μιας και το γεγονός ότι πρόκειται για λογισμικό ανοικτού κώδικα οδηγεί τους δημιουργούς κώδικα και εφαρμογών στην μεγαλύτερη προσοχή σε τμήματα λογισμικού που παρουσιάζουν περίεργη ή προβληματική συμπεριφορά. Αναφέρθηκαν επίσης και οι περιπτώσεις εντοπισμού δυσλειτουργιών και bugs οι οποίες έτυχαν της άμεσης ανταπόκρισης από την ομάδα ανάπτυξης της Google για επιδιόρθωση ή αντικατάσταση λειτουργιών. Βέβαια, υπάρχει συνεχώς περιθώριο βελτίωσης πράγμα που φανερώνουν και οι διαδοχικές εκδόσεις λογισμικού που δίνονται στους καταναλωτές.

Η Apple, έχει αντιμετωπίσει μέχρι στιγμής σοβαρά προβλήματα όσον αφορά την παραβίαση του συστήματός της και αναμένεται αυτό να συνεχίσει. Ο λόγος βρίσκεται στο γεγονός ότι δεν επιτρέπει την εγκατάσταση εφαρμογών από τρίτους με αποτέλεσμα αρκετοί να καταφεύγουν σε ξεκλείδωμα του τηλεφώνου για την προσθήκη λειτουργιών. Όπως έχει ήδη αναφερθεί, η Apple επικεντρώθηκε στην προστασία της συσκευής iPhone και δεν έδωσε παρόμοια σημασία στους μηχανισμούς αντιμετώπισης επιθέσεων σε περίπτωση που κάποιος καταφέρει

παρόλα αυτά να εισέλθει στο σύστημα. Εκατομμύρια δολάρια ξοδεύονται για το σκοπό αυτό χωρίς όμως να εγγυάται κανείς ότι και στο μέλλον δεν θα υπάρξουν εισβολές στο κλειστό κατά τα άλλα λογισμικό της εταιρείας.

Ανοικτά ζητήματα παραμένουν επίσης και στις αγορές εφαρμογών των δύο εταιρειών. Κάθε μια από αυτές αντιμετωπίζει από ένα μεγάλο πρόβλημα που στηρίζεται στον ίδιο παράγοντα: την παροχή εφαρμογών από ανθρώπους σε αυτές. Η Apple παρέχει πολύ αυστηρούς μηχανισμούς αποδοχής εφαρμογών για το App Store. Το πλεονέκτημα είναι ότι με τον τρόπο αυτό εξασφαλίζει κατά κάποιο τρόπο τους χρήστες των συσκευών από την εισβολή κακόβουλου λογισμικού στο σύστημα. Το μεγάλο μειονέκτημα όμως που την απασχολεί είναι η απογοήτευση που δίνει στους δημιουργούς εφαρμογών που δεν πληρούν τα κριτήριά της. Το πρόβλημα έχει γίνει εντονότερο, με το Android Market να πλησιάζει με γρήγορο ρυθμό την ανάπτυξη του App Store. Έτσι, η Apple στρέφεται στην αναζήτηση τρόπων υποστήριξης περισσότερων εφαρμογών έτσι ώστε οι δημιουργοί εφαρμογών να συνεχίσουν να αναπτύσσουν εφαρμογές για το iOS με όσο λιγότερες απώλειες γίνεται προς το Android OS.

Αυτό που για την Apple αποτελεί μειονέκτημα, το Android το παρέχει. Δηλαδή την ελεύθερη είσοδο εφαρμογών στο Android Market. Όμως το πλεονέκτημά της αυτό προκαλεί προβλήματα στην ομάδα ανάπτυξης μιας και ολοένα περισσότερες εφαρμογές μεταμφιέζονται ως χρήσιμες και ελκυστικές τη στιγμή που στην καλύτερη περίπτωση περιέχουν μόνο διαφημιστικό υλικό. Έχουν υπάρξει εφαρμογές με κακόβουλο λογισμικό που διακινήθηκαν στην αγορά με αποτέλεσμα της εισόδου ιών στις συσκευές των χρηστών. Η Google αντέδρασε στις περιπτώσεις αυτές όσο πιο γρήγορα μπορούσε απομακρύνοντας τις εφαρμογές και παρέχοντας λογισμικό επιδιόρθωσης στους χρήστες. Στρέφεται όμως πλέον στην αναζήτηση πιο αποδοτικών μεθόδων προτροπής εισόδου τέτοιων εφαρμογών στο Android Market χωρίς να μπει στη διαδικασία απόρριψης εφαρμογών από τους δημιουργούς όπως η Apple.

Εν αναμονή λοιπόν της βελτίωσης των θεμάτων αυτών η έρευνα ολοκληρώνεται. Το όνομα τόσο της Apple όσο και της Google μπορούν μόνο να εγγυηθούν ότι η λύση σε αυτά τα ζητήματα θα έρθει σύντομα και θα περιλαμβάνει ό,τι πιο νέο στην ασφάλεια λογισμικού και την εξασφάλιση των χρηστών των λειτουργικών τους συστημάτων.

11.2 Συμπεράσματα

Μέσα από την έρευνα αυτή έγινε αναφορά στις κυριότερες πτυχές των δύο λειτουργικών συστημάτων για κινητά τηλέφωνα που κυριαρχούν στην αγορά τα τελευταία 2 χρόνια. Όπως φαίνεται μέσα από την εξέλιξη της επικοινωνίας και των τεχνολογιών της η ανάπτυξη τέτοιων λειτουργικών συστημάτων ήταν φυσικό επακόλουθο. Από τη στιγμή όμως που εμφανίστηκαν οι πρώτες έξυπνες συσκευές, ο ρυθμός ανάπτυξής τους είναι ασύλληπτος. Εκατοντάδες νέα μοντέλα πλημμυρίζουν την αγορά παρόλο που μόνο αυτά με λογισμικό iOS και Android OS φαίνεται να ανταπεξέρχονται της οικονομικής κρίσης και να συνεχίζουν ακάθεκτα την πορεία τους.

Τόσο το iOS όσο και το Android OS στηρίζονται σε αρχιτεκτονικές που τους παρέχουν τα περισσότερα πλεονεκτήματα αλλά και κάποια από τα μειονεκτήματά τους. Από την πλευρά του το iOS δεν παρουσιάζει τόση ευελιξία αφού δεν επιτρέπει αυτοματοποιημένους μηχανισμούς διαχείρισης μνήμης αλλά και εκτέλεση εφαρμογών παράλληλα. Όμως, η Apple κάνει τα πάντα στον τομέα της ασφάλειας με στόχο της ούτε η συσκευή του iPhone αλλά ούτε και το iOS να δέχονται επιθέσεις. Προς το παρόν αυτό βέβαια δεν επιτυγχάνεται πλήρως.

Από την άλλη πλευρά, το Android OS ήταν το μόνο λειτουργικό σύστημα το οποίο από τη μέρα εμφάνισής του φάνηκε ικανό να μπορεί να απειλήσει την μέχρι τότε κυριαρχία της Apple. Πρόκειται για λογισμικό ανοικτού κώδικα το οποίο επιτρέπει σε όποιον ενδιαφέρεται να ασχοληθεί με την ανάπτυξη του και να το εξελίξει. Από την

άλλη όμως, λόγω της πληθώρας των συσκευών στα οποία προσφέρεται φαίνεται να υπάρχει δυσκολία στον καθορισμό ενός συνόλου λειτουργιών και χαρακτηριστικών που να προσφέρονται από όλες τις Android συσκευές. Το πρόβλημα αυτό δεν απασχολεί καθόλου την Apple μιας και έχει χρησιμοποιήσει πρώτη πολλές από τις νέες τεχνολογίες στα iPhone συμπεριλαμβανομένου του multi-touch και της περιστροφής του περιεχομένου της οθόνης όταν αλλάζει ο προσανατολισμό του τηλεφώνου.

Και τα δύο λειτουργικά συστήματα κυκλοφορούν στην αγορά με κάποιες προ-εγκατεστημένες εφαρμογές. Το κοινό όμως απαιτεί και κάποιες ποιο εξειδικευμένες εφαρμογές. Για το λόγο αυτό υπάρχουν το App Store και το Android Market με τα δικά του πλεονεκτήματα και τις δικές του αδυναμίες το καθένα όπως έχουν παρουσιαστεί. Μέχρι τον Απρίλιο του 2011 η ανάπτυξη του Android Market ήταν πολύ γρήγορη με αποτέλεσμα να πλησιάσει τον αριθμό των εφαρμογών του App Store. Σε όλη αυτή τη διαδικασία, εταιρείες όπως η BlackBerry και η NOKIA είναι απλοί θεατές, ανήμποροι να πετύχουν το ίδιο με τα δικά τους λειτουργικά συστήματα.

Κλείνοντας, κανείς δεν μπορεί να πει με σιγουριά ποια θα είναι η κατάληξη σε αυτή την κατά τ' άλλα μεγαλειώδη πορεία των δύο λειτουργικών συστημάτων. Όλες οι εταιρείες που συνεργάζονται με την Google προωθούν συνεχώς νέα μοντέλα στην αγορά διαφόρων μεγεθών, δυνατοτήτων και κόστους. Στόχος τους, ο κάθε άνθρωπος να μπορεί να έχει στα χέρια του μια Android συσκευή. Από την πλευρά της η Apple, ανακοίνωσε ήδη την προετοιμασία για ανάπτυξη του νέου της τηλεφώνου iPhone 5 συνεχίζοντας την προμήθεια της αγοράς με καινοτόμα προϊόντα. Στόχος της Apple, η διάθεση συσκευών στο κοινό με τεχνολογικά χαρακτηριστικά και υπηρεσίες που δεν έχουν σκεφτεί ακόμα οι αντίπαλοι και που δεν υπάρχουν σε καμιά συσκευή μέχρι στιγμής. Απομένει λοιπόν να φανεί ποιος θα είναι τελικά ο κυρίαρχος όσον αφορά το πιο δημοφιλές λειτουργικό σύστημα της αγοράς για κινητά τηλέφωνα με άμεση συνέπεια τη μεγάλη οικονομική επιτυχία που είναι και το ζητούμενο για τους κατασκευαστές.

Βιβλιογραφία

[1] A. Pashtan, “Wireless Terrestrial Communications: Cellular Telephony”, Aware networks, Inc., Buffalo Grove, Illinois, USA, © 2006 Eolls Publishers

[2] V. Kumar, **“**Mobile Computing: A Brief History of Personal Communication System”, University of Missouri-Kansas City

[3] F. Turisko, J. Case, “Wireless and Mobile Computing”, Prepared by First Consulting Group, October 2001, ISBN: 1-929008-72-4, Copyright© 2001 California Health Foundation

[4] M. Satyanarayanan, “Fundamental Challenges in Mobile Computing”, School of Computer Science, Carnegie Mellon University

[5] R. Godwin-Jones, “Emerging Technologies Mobile-Computing Trends: Lighter, Faster, Smarter”, Language Learning & Technology http://llt.msu.edu/vol12num3/emerging/, October 2008, Volume 12, Number 3 pp. 3-9, Copyright © 2008, ISSN 1094-3501

[6] http://developer.apple.com/, Last Accessed 26/5/2011

[7] Impact of the Apple iPhone on the Mobile Phone Industry, March 4,2007

[8] M. Ruggiero, “iPhone Programming”, University of Bologna, Italy

[9] http://developer.apple.com/devcenter/ios/index.action, Last Accessed 26/5/2011

[10] L. Cheng, “Analysis and Comparison with Android and iPhone Operating System"

[11] V. R. Pandya, “IPhone Security Analysis”, In Partial Fulfillment of the Requirements for the Degree Master of Computer Science, Department of Computer Science San Jose State University, May 2008

[12] C. P. Praher, **“**Mobile Service Oriented Architecture in the Context of Information Retrieval”, In Partial Fulfillment of the Requirements for the Degree Master of Computer Science, Department of Computer Science, Linz University, June 2008

[13]M. Frederic-Gerald, Android Architecture, German University in Cairo

[14] B. Arve, I. Hickson, “Opera Platform DOM Interface Specifcation 1.1, APR 2005”, URL: http://oxine.opera.com/documentation/dom-interface.html

[15] Apple Inc.: iPhone OS Programming Guide, URL https://developer.apple.com/iphone/library/documentation/iPhone/Conceptual/iPhoneOSProgrammingGuide/iPhoneOSProgrammingGuide.pdf

[16] Canalys research release 2008/021, URL http://www.canalys.com/pr/2008/r2008021.pdf

[17] Androidology – Application Lifecycle

[18] theiPodObserver, URL: http://www.ipodobserver.com/ipo/article/iphone_memory_management_from_a_users_perspective/, Last Accessed 26/5/2011

[19] http://mauvilasoftware.com/iphone_software_development/2008/01/iphone-memory-management-a-bri.html, Last Accessed 26/5/2011

[20] http://memo.tv/memory_management_with_objective_c_cocoa_iphone, Last Accessed 26/5/2011

[21]http://developer.apple.com/library/mac/#documentation/cocoa/conceptual/MemoryMgmt/MemoryMgmt.html, Last Accessed 26/5/2011

[22]Apple Developers, URL: http://developer.apple.com/library/ios/#documentation/iphone/conceptual/iphoneosprogrammingguide/CoreApplication/CoreApplication.html, Last Accessed 26/5/2011

[23]Apple Developers, URL: http://developer.apple.com/library/ios/#documentation/iphone/conceptual/iphoneosprogrammingguide/CoreApplication/CoreApplication.html, Last Accessed 26/5/2011

[24] http://mobworld.wordpress.com/2010/07/05/memory-management-in-android, Last Accessed 26/5/2011

[25] Android Developers, URL: http://developer.android.com/resources/articles/avoiding-memory-leaks.html, Last Accessed 26/5/2011

[26] Android Developers BlogSpot, URL: http://android-developers.blogspot.com/2009/02/track-memory-allocations.html, Last Accessed 26/5/2011

[27]Android Developers, URL: http://developer.android.com/guide/topics/fundamentals.html, Last Accessed 26/5/2011

[28] Android Developers, URL: http://developer.android.com/guide/developing/building/index.html, Last Accessed 26/5/2011

[29] V. R. Pandya, “IPhone Security Analysis”, In Partial Fulfillment of the Requirements for the Degree Master of Computer Science, Department of Computer Science San Jose State University, May 2008

[30] C. Miller, J. Honoroff, J. Mason, “Security Evaluation of Apple’sIPhone”, July 19, 2007

[31] http://www.securityevaluators.com/iphone/exploitingiphone.pdf, URL: http://www.androidauthority.com/android-vs-ios-security-features-compared-11065/, Last Accessed 26/5/2011

[32] Apple Inc, "iPhone in BusinessSecurity Overview", Copyright © 2009 Apple Inc.

[33] J. Burns, "Mobile Application Security on Android-Context on Android security", Black Hat, June, 2009

[34] J. Burns(2009, October) "Developing Secure Mobile Applications For Android", URL: http://www.isecpartners.com/files/iSEC_Securing_Android_Apps.pdf, October 2009

[35] Android Developers, URL: http://developer.android.com/guide/topics/security/security.html, Last Accessed 26/5/2011

[36] http://www.androidauthority.com/the-security-architecture-of-android-11063/, Last Accessed 26/5/2011

[37] Technical Report: "Analysis report on Android Application Framework and existing Security Architecture", Security Engineering Research Group, February 2010

[38] M. Reardon, "Vulnerability to image processing Libraries", URL http://www.builderau.com.au/news/soa/Security-flaws-unearthed-in-Google-s-Android/0,339028227, 339286533,00.htm.

[39] "Security hole in web browser", URL: http://mobile.slashdot.org/article

[40] "Bugs that leads to denial of service attack", URL: http://www.ocert.org/advisories/ocert-2009-014.html

[41] http://en.wikipedia.org/wiki/IPhone#Interface, Last Accessed 26/5/2011

[42] http://en.wikipedia.org/wiki/Comparison_of_Android_devices, Last Accessed 26/5/2011

[43] http://www.ehow.com/how_5115482_update-iphone-software.html, Last Accessed 26/5/2011

[44] http://en.wikipedia.org/wiki/Apple_App_Store, Last Accessed 26/5/2011

[45] http://en.wikipedia.org/wiki/Android_Market, Last Accessed 26/5/2011

[46] Article "The iPhone ushers in an age of true mobility", Global Telecoms Business CEO and CFO Guide to IP Transformation: November/December 2008

[47] M. Butler, "Android: Changing the Mobile Landscape", Pervasive Computing, Published by the IEEE CS, January-March 2011

[48] **"**Android by 2012 - A study on present and future of Google's Android", Dot Com Infoway - Position Paper

[49] D. Roth, "Google's Phone", July 2008

[50]M. Stanley, "Internet Trends", CM Summit, New York, June 2010

[51]A. Hand, “Android Vs iPhone” presentation, presented at MobileMonday Chicago, April 27, 2009

[52]AdMob Mobile Metrics, “Metrics Highlights”, URL: http://metrics.admob.com, May 2010

[53] http://www.thesmythgroup.com/2011/03/android-app-market-to-overtake-itunes-app-store-soon, Last Accessed 26/5/2011

[54] http://www.apache.org/licenses/LICENSE-2.0.html, Last Accessed: 26/5/2011

[55] http://warp.povusers.org/programming/mvc.html, Last Accessed: 26/5/2011

[56] http://www.webkit.org/, Last Accessed: 26/5/2011

[57]http://www.windowsecurity.com/articles/analysis_of_buffer_overflow_attacks.html, Last Accessed: 26/5/2011

[58] http://www.apple.com/mobileme/, Last Accessed: 26/5/2011

[59] http://en.wikipedia.org/wiki/Sandbox_%28computer_security%29, Last Accessed: 26/5/2011

[60] http://www.gsmarena.com/apple_iphone_4g-3275.php, Last Accessed: 26/5/2011

[61] http://www.google.com/nexus/#/tech-specs, Last Accessed: 26/5/2011

[62] http://developer.android.com/sdk/index.html, Last Accessed: 26/5/2011